A Methodology for the Numeric Time-Cost Forecast and Pareto Optimization of Large Injection Projects in Tunneling

von

Jan Onne Backhaus

2021

Herausgegeben von

J. Grabe

Technische Universität Hamburg
Institut für Geotechnik und Baubetrieb

Veröffentlichungen des Instituts für
Geotechnik und Baubetrieb

49

Herausgeber:

Univ.-Prof. Dr.-Ing. Jürgen Grabe
Technische Universität Hamburg
Institut für Geotechnik und Baubetrieb
Harburger Schloßstraße 36
D – 21079 Hamburg
E-Mail: *grabe@tuhh.de*

ISBN-13: **978-3-936310-51-1** **(Erstausgabe)**
DOI: **10.15480/882.3840** **(digital)**

2021

Druckerei

Druckzentrum Neumünster GmbH
Rungestraße 4
24537 Neumünster

In derselben Reihe erschienen:

1. J. Grabe (Hrsg.), 2000: Verbrennungsrückstände. Tagungsband, ISBN 3-936310-00-9
2. J. Grabe (Hrsg.), 2001: Schaden- und Risikomanagement im Tiefbau. Tagungsband, ISBN 3-936310-01-7
3. J. Grabe, 2003: Bodenmechanik und Grundbau. Skriptum, ISBN 3-936310-03-3
4. J. Grabe (Hrsg.), 2003: Euronormen in der Geotechnik – Was ändert sich? Tagungsband, ISBN 3-936310-04-1
5. J. Grabe (Hrsg.), 2003: Bodenverdichtung, Experimente - Modellierung - Geräteentwicklung - Baustellenberichte - F+E-Bedarf. Tagungsband, ISBN 3-936310-05-X
6. M. Kelm, 2004: Nummerische Simulation der Verdichtung rolliger Böden mittels Vibrationswalzen. Promotion, ISBN 3-936310-06-8
7. J. Grabe (Hrsg.), 2004: Kaimauern - Messungen und Numerik. Tagungsband, ISBN 3-936310-07-6
8. J. Stein, 2005: Experimentelle und numerische Untersuchungen zum Düsenstrahlverfahren. Promotion, ISBN 3-936310-09-2
9. J. Grabe (Hrsg.), 2005: Grenzschicht Wasser und Boden - Phänomene und Ansätze. Tagungsband, ISBN 3-936310-10-6
10. J. Grabe (Hrsg.), 2005: FEM in der Geotechnik - Qualität, Prüfung, Fallbeispiele. Tagungsband, ISBN 3-936310-11-4
11. B. Mardfeldt, 2006: Zum Tragverhalten von Kaikonstruktionen im Gebrauchszustand. Promotion, ISBN 3-936310-12-2
12. J. Grabe (Hrsg.), 2006: Optimierung in der Geotechnik - Strategien und Fallbeispiele. Tagungsband, ISBN-13: 978-3-936310-13-9
13. T. Bierer, 2007: Bodenschwingungen aus Straßenverkehr auf unebener Fahrbahn im Zeitbereich - experimentelle und theoretische Untersuchungen. Promotion, ISBN-13: 978-3-936310-14-6
14. J. Grabe (Hrsg.), 2007: Bemessen mit Finite-Elemente-Methoden. Tagungsband, ISBN-13: 978-3-936310-15-3
15. K.-P. Mahutka, 2008: Zur Verdichtung von rolligen Böden infolge dynamischer Pfahleinbringung und durch Oberflächenrüttler. Promotion, ISBN-13: 978-3-936310-16-0
16. J. Grabe (Hrsg.), 2008: Seehäfen für Containerschiffe zukünftiger Generationen. Tagungsband, ISBN-13: 978-3-936310-17-7
17. F. König, 2008: Zur zeitlichen Traglastentwicklung von Pfählen und der nachträglichen Erweiterung bestehender Pfahlgründungen. Promotion, ISBN-13: 978-3-936310-18-4
18. S. Henke, 2008: Herstellungseinflüsse aus Pfahlrammung im Kaimauerbau. Promotion, ISBN-13: 978-3-936310-19-1

19. J. Grabe (Hrsg.), 2009: Spundwände – Profile, Tragverhalten, Bemessung, Einbringung und Wiedergewinnung. Tagungsband, ISBN-13: 978-3-936310-20-7
20. J. Dührkop, 2009: Zum Einfluss von Aufweitungen und zyklischen Lasten auf das Verformungsverhalten lateral beanspruchter Pfähle in Sand. Promotion, ISBN-13: 978-3-936310-21-4
21. O. Möller, 2009: Zum Langzeit-Kompressionsverhalten weicher organischer Sedimente. Promotion, ISBN-13: 978-3-936310-22-1
22. J. Grabe (Hrsg.), 2011: Ports of container ships of future generations. Tagungsband, ISBN-13: 978-3-936310-23-8
23. S. Kinzler, 2011: Zur Parameteridentifikation, Entwurfs- und Strukturoptimierung in der Geotechnik mittels numerischer Verfahren. Promotion, ISBN-13: 978-3-936310-24-5
24. G. Qiu, 2012: Coupled Eulerian Lagrangian Simulations of Selected Soil-Structure Problems. Promotion, ISBN-13: 978-3-936310-25-2
25. X. Ma, 2013: Nutzung der oberflächennahen Geothermie mittels Energiepfählen und Erdwärmesonden. Promotion, ISBN-13: 978-3-936310-26-9
26. J. Grabe (Hrsg.), 2013: Proceedings of the Conference on Maritime Energy COME 2013. Tagungsband, ISBN-13: 978-3-936310-28-3
27. J. Grabe (Hrsg.), 2013: Bemessen mit numerischen Methoden. Tagungsband, ISBN-13: 978-3-936310-29-0
28. T. Pucker, 2013: Stoffmodell zur Modellierung von stetigen Materialübergängen im Rahmen der Optimierung geotechnischer Strukturen. Promotion, ISBN-13: 978-3-936310-30-6
29. S. Henke, 2013: Untersuchungen zur Pfropfenbildung infolge der Installation offener Profile in granularen Böden. Habilitation, ISBN-13: 978-3-936310-31-3
30. J. Grabe (Hrsg.), 2014: Ports for Container Ships of Future Generations. Tagungsband, ISBN-13: 978-3-936310-32-0
31. J. Grabe (Hrsg.), 2014: Offshore Basishäfen. Tagungsband, ISBN-13: 978-3-936310-33-7
32. C. Rudolph, 2015. Untersuchungen zur Drift von Pfählen unter zyklischer, lateraler Last aus veränderlicher Richtung. Promotion, ISBN-13: 978-3-936310-34-4
33. J. Grabe (Hrsg.), 2015: Morphodynamics 2015. Tagungsband, ISBN-13: 978-3-936310-35-1
34. T. Hamann, 2015: Zur Modellierung wassergesättigter Böden unter dynamischer Belastung und großen Bodenverformungen am Beispiel der Pfahleinbringung. Promotion, ISBN-13: 978-3-936310-36-8
35. B. Schümann, 2015: Beitrag zum dynamischen Dreiphasenmodell für Boden auf Basis der Finite-Elemente-Methode. Promotion, ISBN-13: 978-3-936310-37-5

36. M. Milatz, 2015: Untersuchungen zum Einfluss der Kapillarität auf das hydraulisch-mechanische Verhalten von granularer Tragschichten für Verkehrswege. Promotion, ISBN-13: 978-3-936310-38-2
37. H. Kaya, 2016: Bodenverschleppung und Spaltbildung infolge der Einbringung von Profilen in Dichtungsschichten aus Ton. Promotion, ISBN-13: 978-3-936310-39-9
38. J. Grabe (Hrsg.), 2017: Proceedings of the Conference on Maritime Energy COME 2017. Tagungsband, ISBN-13: 978-3-936310-40-5
39. B. Kocak, 2017: Zur numerischen Modellierung von hydraulisch-mechanisch gekoppelten Prozessen in gesättigten granularen Böden mittels Smoothed Particle Hydrodynamics. Promotion, ISBN-13: 978-3-936310-41-2
40. K. Siegl, 2017: Zur Pfahldynamik von gerammten Großrohrpfählen und der daraus resultierenden Wellenausbreitung in Wasser und im Meeresboden. Promotion, ISBN-13: 978-3-936310-42-9
41. J. Grabe (Hrsg.), 2017: Numerical Methods in Geotechnics. Tagungsband, ISBN-13: 978-3-936310-43-6
42. J. Grabe (Hrsg.), 2018: Digitale Infrastruktur und Geotechnik (DIG 2018). Tagungsband, ISBN-13: 978-3-936310-44-3
43. D. Osthoff, 2018: Zur Ursache von Schlosssprengungen und zu einbringbedingten Lageabweichungen von Spundwänden. Promotion, ISBN-13: 978-3-936310-45-0
44. E. Heins, 2018: Numerical based identification of the pile-soil interaction in terms of the axial pile bearing capacity. Promotion, ISBN-13: 978-3-936310-46-7
45. K.-F. Seitz, 2021: Zur Topologieoptimierung von geotechnischen Strukturen und zur Tragfähigkeitssteigerung des Baugrunds durch Scherfugenverfestigung. Promotion, ISBN-13 978-3-936310-47-4
46. D. Plenker, 2021: Physical and numerical investigations of the dynamic interaction of saturated granulates and fluid. Promotion, ISBN-13: 978-3-936310-48-1
47. J. Grabe, J. O. Backhaus, P. Vogel, 2021: Bauprojektmanagement. Skriptum, ISBN-13: 978-3-936310-49-8
48. M. Kanitz, 2021: Experimental and numerical investigations of particle-fluid systems in geotechnical engineering. Promotion, ISBN-13: 978-3-936310-50-4

Dedicated to

myself

Vorwort des Herausgebers

Die möglichst zutreffende Vorhersage der Bauzeit und der Baukosten ist seit jeher einer der Herausforderungen des Bauingenieurs. Die Abweichungen zwischen Soll und Ist sind, wie prominente Einzelfälle wie die Elbphilharmonie in Hamburg, der Flughafen Berlin-Brandenburg oder Stuttgart 21 belegen, teilweise eklatant. Es stellt sich die Frage, ob durch die heutigen Möglichkeiten, Daten in großer Anzahl in digitaler Form aufzunehmen und geeignet auszuwerten eine zutreffendere Prognose gelingt. Voraussetzung ist allerdings ein umfangreicher Datensatz, welcher Herrn Backhaus im Zuge der Injektionsarbeiten im Gipskeuper beim Tunnelbauprojekt Stuttgart 21 zur Verfügung stand. Es ist reizvoll verschiedenste mathematische Verfahren wie beispielsweise die Monte-Carlo-Methode, die Methode der Markow Ketten, die Graphentheorie, die numerische Optimierung sowie neuronale Netzwerke auf diese Daten anzuwenden. Da die Digitalisierung im Bauwesen mit großen Schritten voranschreitet, ist das Thema der Arbeit von hoher Relevanz für die Forschung und Entwicklung.

Die Ziele der Arbeit von Herrn Backhaus sind gemäß Kapitel 3:

1. Analyze and visualize the digitized data set to improve the understanding of the data, and the construction site's workings. 2. Forecast of the grout volume, which is injected in the tunnel. 3. Forecast of the work progress of an injection production systems, which consists of injection units of the same type. 4. Forecast of the construction time and cost of an injection production system, which includes two types of injection units. The methods shall take the mutual collaboration of injection units into account. Furthermore, the forecasting methods shall be used as goal functions to find Pareto optimal solutions of the number of machines employed on the construction site. 5. Improving the model developed under objective 4 to include the discrete location of injection units in the tunnel. Furthermore, the mutual obstruction of injection units in the narrow tunnel environment shall be taken into account, and a third type of injection unit shall be included in the simulation.

Herr Backhaus hat sich zur Klärung der Forschungsfragestellung mit mehreren mathematischen Ansätzen beschäftigt, diese in einen Programmcode implementiert und auf die von ihm im ersten Schritt bereinigten Daten angewendet. Die Arbeit ist methodisch im Gebiet der Bauinformatik angesiedelt, während das Randwertproblem dem Spezialtiefbau als Teilgebiet der Geotechnik zugeordnet ist. Es handelt sich also um eine Dissertation an der Grenze zwischen Geotechnik und Bauinformatik, eine Grenze, die zukünftig sicherlich immer mehr verschwimmen wird.

In Kapitel 1 führt Herr Backhaus zunächst in die Thematik und Motivation seiner Arbeit ein. Dabei war ein an der TUHH durchgeführter Workshop „Digitale Infrastruktur und Geotechnik“ ein Meilenstein, da er hierdurch den Kontakt zum vorgenannten Injektionsprojekt Stuttgart 21 mit einer großen Anzahl von bereits erfassten Datensätzen bekam.

Das Kapitel 2 gibt den Stand der Wissenschaft zur Vorhersage von Kosten und Bauzeiten im Bauwesen wieder. Er beschreibt kurz die in der Praxis vielfach verwendeten Netzpläne (Critical Path Method, Metra Potential Method, Graphical Evaluation and Review Technique). Dann geht er auf Petri Netze zur Beschreibung der Prozesse als Orte/Zustände

(Places) und Zustandsänderungen (transitions) ein. Es folgt eine Beschreibung der Monte-Carlo Simulation, des Agentenverfahren (Agent based simulation), der kontinuierlichen Ansätze wie System Dynamics, diskreter Ansätze (Discrete event models), Ansätze wie Fuzzy Logic, und weitere eher einfache, im Bauwesen zumeist verwendete Prognosemethoden. Im Abschnitt 2.5 bewertet er die vorgenannten Ansätze und entwickelt seinen Forschungsansatz.

In Kapitel 3 stellt er, wie bereits geschrieben, die Zielsetzung seiner Arbeit dar und erläutert den methodischen Ansatz seiner Arbeit.

Die mathematischen Grundlagen, der von ihm im Weiteren verwendeten Ansätze, werden in Kapitel 4 behandelt.

In Kapitel 5 beschreibt er die Baustelle mit den besonderen geologischen Randbedingungen im Gipskeuper sowie das Injektionsverfahren selbst. Bild 5.13 zeigt beispielhaft die bei einem Injektionsgerät aufgezeichneten Daten. Die Gesamtdaten mussten gesichtet und um Fehlzeiten und -vorgängen bereinigt werden. Tabelle 5.3 zeigt, welche enormen Datenmengen zur Auswertung und Prognose zur Verfügung standen. Nun folgt die Auswertung der Daten. Interessant ist der Abschnitt 5.6. Gelingt es die Injektionsmengen anhand der räumlich benachbarten Injektionsstellen vorherzusehen? Am ehesten mit den verwendeten Markow Ketten, wie in Abschnitt 5.6.4 gefolgert wird.

Die Vorhersage der Projektdauer auf der Basis der bis zum Zeitpunkt der Prognose bekannten Prozessdaten wird in Kapitel 6 behandelt. Er beginnt in Abschnitt 6.1 mit der Prognose des Fortschritts einer Injektionseinheit und erweitert dies in Abschnitt 6.2 auf zwei Einheiten. In Abschnitt 6.3. diskutiert Herr Backhaus die Konvergenz, der von ihm angewandten Monte Carlo Simulation. Bilder 6.8 und 6.9 zeigen, dass die mean/markov Prognose etwas besser abschneidet als die in der Praxis üblicherweise angewandte Prognose auf der Basis der Mittelwerte.

Im Kapitel 7 untersucht Herr Backhaus, wie aus dem optimalen Zusammenspiel zweier Einheiten eine Reduktion der Gesamtdauer und -kosten erzielt werden kann, siehe Bild 7.4. Dazu wendet er die numerische Optimierung mittels evolutionärer Algorithmen an. Eine Methode, die im Institut bereits für andere Optimierungsfragen erfolgreich Anwendung findet. Die Ergebnisse der Optimierung sind in den Bildern 7.19, 7.21 und 7.23 dargestellt.

Die vorgenannte Analyse wird noch um eine dritte Einheit erweitert, siehe Kapitel 8, wodurch weitere Freiheitsgrade entstehen, siehe Bild 8.9. Die Bilder 8.13, 8.14 und 8.16 zeigen, wie groß die Unterschiede in den Kosten und Projektdauern sind. So entsteht ein erheblicher Mehrwert zur Auswahl des bestmöglichen Geräteeinsatzes.

In Kapitel 9 stellt Herr Backhaus die wesentlichen Erkenntnisse seiner Arbeit dar und gibt einen Ausblick.

Die von Herrn Backhaus vorgelegte Arbeit zeigt, welches Potential die Digitalisierung im Bauwesen hat, sofern ausreichende Datenmengen wie im vorliegenden Fall vorliegen. Es gelingt ihm durch systematische Methodenauswahl, deren Implementierung in einem selbst geschriebenen Programm sowie dessen Anwendung auf die von ihm bereinigten Daten den Mehrwert hinsichtlich der Zeit- und Kostenprognose zu zeigen. Besonders deutlich ist der

Mehrwert bei der von ihm untersuchten Betrachtung der Pareto Optima beim Einsatz mehrerer Injektionseinheiten.

Es war mir eine große Freude Herrn Backhaus während seiner Promotionszeit zu begleiten. Für seinen weiteren Lebensweg wünsche ich viel Erfolg.

Hamburg 21.06.21

Jürgen Grabe

Vorwort des Authors

The current trend towards digitization in construction is already leading to the emergence of comprehensive databases describing construction sites. Although these data silos are already being successfully used to answer specific questions, their inherent potential to add value to the construction process has not yet been fully exploited. My research is driven by the desire to identify and leverage the data treasures inherent in these data silos to add value to construction sites.

In 2018, during my time at the Institute of Geotechnical Engineering and Construction Management, I organized the conference "Digital Infrastructure and Geotechnical Engineering 2018 (DIG2018)". At that conference, I was fortunate to make the acquaintance of Mr. Sewerin Sabew of Renesco GmbH. Mr. Sabew was the project manager of an injection site of the Stuttgart/21 project. With the support of the Viennese company eguana GmbH, Mr. Sabew had set up a system that digitized the many thousands of waterproofing injections on the construction site. He showed me the result of this digitization on his tablet PC in the form of clear visualizations. These revealed the daily status of all machines, both active and idle, on the project, and he could provide many thousands of injection logs on demand. We agreed that the database behind the visualization was an ideal foundation for developing and testing a methodology for further value creation on the construction site. Much of what grew out of that initial meeting is described on the following pages. However, left unmentioned are all those who supported me during my research process:

My thanks go to my doctoral advisor, Professor Dr. Jürgen Grabe, for the opportunity to work on such an exciting topic, and to Prof. Dr. Christian Ringle and Prof Dr. Konrad Nübel, who accompanied my doctorate as reviewers. My sincere gratitude goes to Mr. Sewerin Sabew for the many exciting and visionary discussions on digitalization in the construction industry. Thank you for the opportunity to accompany the experts from Renesco GmbH during their work in the tunnel. My thanks go to Mr. Flavio Piras from Renesco GmbH for the many telephone calls during which we discussed the details of his injection construction site. I would also like to thank Mr. Philipp Maroschek of eguana GmbH for providing the digital data required to perform my research.

For their support in proofreading this thesis, I would like to thank Anne Stark, Christopher Tinat, Dennis Heinrich, Jannik Beuße, Dr. Martin Rose, Dr. Marius Milatz, Maximilian Budau, Paul Vogel, and Pauline Kaminski.

I was alone in my research area at the Institute of Geotechnical Engineering and Construction Management. For this reason, professional feedback and the opportunity to discuss parts of the work were of particular value to me. I want to thank Professor Dr. Kathrin Fischer and Professor Dr. Hermann Lödding for the opportunity to discuss my ideas at their institute. My thanks likewise go to my sparring partners Tom Toerzs (LaTeX & MATLAB), Alexander Chmelnizkij (Hardware, Software, Mathematics), and Philip Zirfaß (MATLAB Deep Learning Toolbox). My thanks go to Prof. Dr. Markus Dahm and Mr. Valentin Richter for inspiring discussions on lean construction. Without the excellent introduction to graph theory by Dr. Martin Rose, I would not have been able to write this thesis, so my thanks go to him for sharing his wisdom in the realm of nodes and edges.

For me, completing a dissertation parallel to a part-time MBA study was, at many points, essentially a psychological challenge of endurance. For this reason, I want to mention those people who have supported me emotionally over the years. Thank you, Tom Toerzs, Dr. Manuela Kanitz, Dr. Marc Stapelfeldt, Dennis Heinrich, Florian Bettex, and Timon Burgwedel for the many fantastic journeys into the Pen&Paper worlds of DSA and FATE. Many thanks to Jan Syring, Henner Bach, Eric Bendels, Florian Bettex, and Volker Seifert for confused, insane, and silly diversions. My special thanks to Volker Seifert, Dr. Gabi Kaffka, Dr. Michael Hasse, and Dr. Martin Rose for your enthusiasm and interest in science, knowledge, and research, which provided me with the intrinsic motivation to start and finish this work. In addition to that, my thanks go to Dr. Michael Hasse for not giving up on teaching me the proper use of the English language, without which I would not have been allowed to study me trade or write me thesis.

At this point, I would also like to thank my parents for supporting me and, to a large part, financing my education.

My special thanks go to Joy Kraft for enduring the ups and downs during my doctorate with me and to Ella Fischer, whose joyful nature carried me through so many lows.

Hamburg 12.07.2021

Jan Onne Backhaus

Schlagwörter:

Baumanagement, Zeit- und Kostenvorhersage, Injektion, Simulation, Diskrete Ereignissimulation, Monte-Carlo Methode, Markow Kette Genetische Optimierung, Deep Learning, Graphentheorie

Keywords:

construction management, time- and cost forecast, injection, simulation, discrete event simulation, monte-carlo method, markov chain, genetic optimization, deep learning, graph theory

A Methodology for the Numeric Time-Cost Forecast and Pareto Optimization of Large Injection Projects in Tunneling

Vom Promotionsausschuss der
Technischen Universität Hamburg

zur Erlangung des akademischen Grades

Doktor-Ingenieur(in) (Dr.-Ing.)

genehmigte Dissertation

von
Jan Onne Backhaus

aus
Hannover

2021

1. Gutachter: Univ.-Prof. Dr.-Ing. Jürgen Grabe
2. Gutachter: Univ.-Prof. Dr.-Ing. Konrad Nübel
3. Gutachter: Univ.-Prof. Dr. rer. pol. Christian M. Ringle

Tag der mündlichen Prüfung: 01.06.2021

Contents

1 Introduction

> *Successful digital transformation is like a caterpillar turning into a butterfly. [...] Unfortunately, when it comes to digital transformation, many [...] are just thinking about fast caterpillars. And it's hard to keep up with your competitors if you're crawling ahead while they can fly.*
>
> *Westman (2017)*

The German construction industry is facing inevitable change as a result of its long time underperformance. That underperformance becomes apparent when comparing its *Gross Value Added* (GVA) with that of other industries. Figure 1.1 shows that since 1994 GVA is continually declining, seems constant after 2005, and shows a slight tendency to increase after 2015. This behavior is in stark contrast to other industries, such as the manufacturing and service industry, which managed to increase GVA since 1994 continuously.

Industry experts, such as Bektas (2013), Laufer and Tucker (1987), Rahm et al. (2015), VDI (2010), and Wallerang (2015), see a need for change in the way of production on construction sites where they perceive miscommunication, inefficiency, and downtime, as the rule rather than the exception. The result is a flood of civil lawsuits. Wallerang (2015) states, that about 30 percent of all civil actions in Germany deal with disputes with their roots on construction sites.

A study by Hertie Business School of Governance in cooperation with the *Organisation for Economic Co-operation and Development* (OECD) comes to another conclusion. According to the study, the reasons for the underperforming construction industry are to be found primarily on the legislative side. It is foremost poor policy-making, which has led to the problem; see Hertie School Of Governance (2016).

At the same time, political decision-makers claim that it is the construction industry's planning progress, which poses the most inefficient part and needs to be changed. With regard to major construction projects, the BMVI (2015a, pp. 5 sq.) points out that

- political motivated wrong estimation of building cost and the neglect of risks,
- imprecise determination of clients' wishes and insufficient consideration of the specifics of the project at the start of planning,
- insufficiently detailed planning of major projects and a lack of cooperation between stakeholders,
- missing risk management both before and during the building phase,

- missing management know-how for the execution of major projects on the side of owner and contractor,
- missing transparency of project status, cost risk, and due dates, and finally
- pure price-based decision-making during the tender phase,

are typical for major construction projects in Germany. The *Bundesministerium für Verkehr und Digitale Infrastruktur* (BMVI) regards these factors as decisive for the considerable cost increases, overruns of planned deadlines, cost-intensive rework, and the associated claims of major German construction projects.

The problem is recognized, and the solution sought in the construction industry's digital transformation. To increase the value creation on construction sites, the BMVI has published a phased plan for the introduction of *Building Information Modeling* (BIM). This plan is complemented by several pilot projects and has the ultimate goal to include BIM in all tenders for federal infrastructure projects starting from 2020; see BMVI (2015b, p. 5, 2017, pp. 5, 15 sq.) and Freiboth (2006).

Before going any further, the following section 1.1 explains the process commonly known as *Digital Transformation*. This clarification is necessary to place the presented work in the context of the construction industry's current digital development.

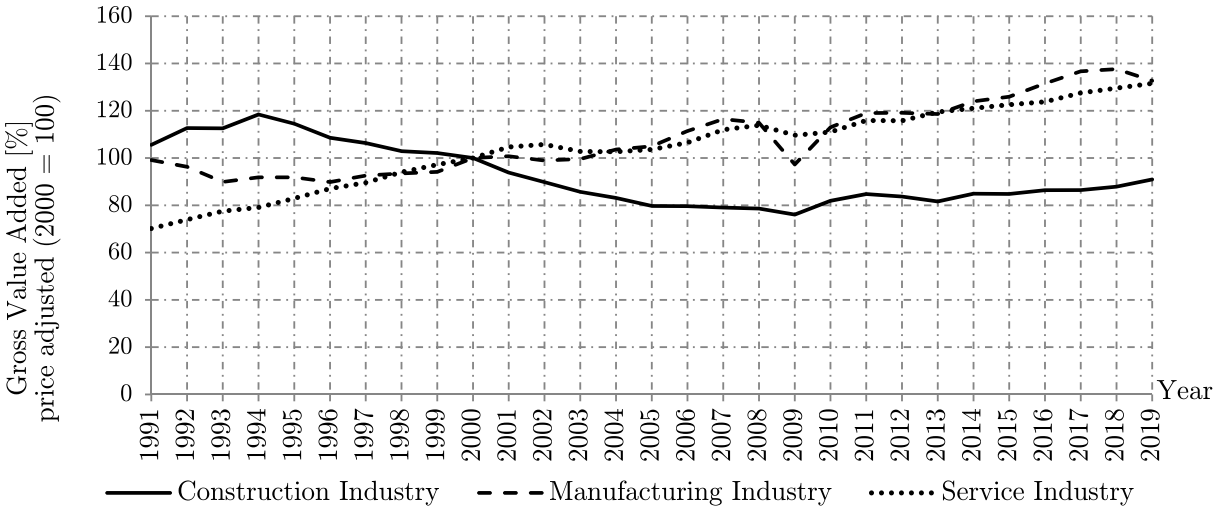


Figure 1.1: Gross value added, price adjusted (2000 = 100) of construction industry, manufacturing industry and service industry. In comparison the construction industry shows clear underperformance. Picture adapted from Backhaus and Dahm (2020) with data from Destatis (2019, p. 59).

1.1 Digitization, Digitalization and Digital Transformation

The terms *digitization*, *digitalization* and *digital business transformation*, albeit colloquially treated interchangeably, have a different meaning, Figure 1.2. They all fall under the current trend of *digital transformation*, which is a collective term for anything from the modernization of *Information Technology* (IT) infrastructure, i. e., replacing the file server with cloud computing, to the invention of new digital business models, such as mediating cheap transport solution via software application.

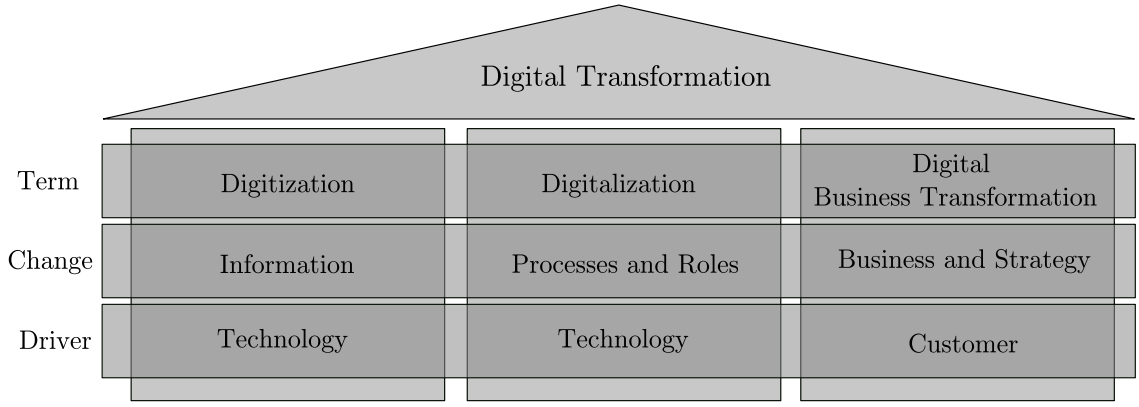


Figure 1.2: Digital Transformation is a current trend towards digitizing information, digitalization of processes, and roles, with the ultimate goal of digital business transformation of the business and its strategy.

Digitization refers to the process of "[...] changing from analog to digital form [...]", Gartner (2020). An example from the construction industry is *Computer-aided Design* (CAD), which is used to create analog drawings with the computer, instead of using the drawing board. Another example is the digitization of paper-based information, such as scanning invoices and storing them in a database. In digitization, however, it is always the information that is made digital. The business process, e. g., the checking of invoices, stays analog. In this sense, the current developments in BIM are efforts to digitize construction sites by making previous analog information available in digital form.

The term *digitalization* is harder to grasp. Gartner (2020) defines it as "[...] the use of digital technologies to change [an analog] business model [towards a digital business model] and provide new revenue and value-producing opportunities [...]", a rather vague definition. Digitalization builds on top of digitization as it uses digitized information to digitalize business processes. Muro et al. (2017, p. 7) gives examples of online supply chain management and internet market places, such as Workyard (2020), a platform mediating workers and offering payroll accounting. An example of the German construction industry's recent development is the introduction of the iTWO software by the Deutsche Bahn group, see RIB Software SE (2019), as the mandatory standard project management software. Instead of merely mirroring the analog world's analog objects, digitalization transforms the process itself.

Digitization and digitalization can finally lead to *digital business transformation*. In such a business, the processes have not only been digitalized. Digitization and digitalization are further exploited "to create a robust new digital business model", Gartner (2020). Such digital transformation is not just digitizing all physical objects and the digitalization of processes but a strategic decision. It requires an organization to develop the ability to change as an enabler to become a customer-driven end-to-end enterprise. The digital business transformation requires digitization and digitalization as enablers; both are essentially technology-focused. However, the digital business transformation itself is not about technology but the customer, Bloomberg (2018).

1.2 Problem Statement and Vision

The current trend in digitizing construction sites leads to more and more data being stored in digital databases. This development makes the data available for computer-aided processing, thus enabling the creation of additional digital value. The role models are the big internet companies such as Google and Facebook; see Facebook (2020) and Google (2020), who use the data stored on their customers' behavior to generate revenues that outpace other industries. The challenge for construction companies is how to generate value from the existing and available databases. These are in particular the currently growing BIM databases, be it in a commercial format, such as CAD models, as resource planning systems, or mere time stamps of digitized access control systems of construction sites.

An example of additional digital value creation for an injection project is shown in Figure 1.3. With sufficient real-time data and the right algorithms, a digital construction site manager would automatize and optimize decision-making based on numerical construction progress forecasts. Parallel to the construction site's value creation, all process data is digitized and handed to the digital construction manager. That digital manager computes forecasts and uses them in combination with optimization algorithms to derive concrete recommendations for best actions on the construction site. Such a system can provide meaningful assistance in construction site management's decision-making processes and serve as an early warning system, e. g., to avoid that butterfly effects transform small disturbances into large financial liabilities.

Under the assumption that construction vehicles will work autonomously in the near future, such a system would even replace on-site managers. It would continuously compute the best order of process execution and leave the execution itself to autonomous vehicles, which frequently report their status back to the control system. That such systems are not to be cast in the area of science fiction has already been proven by the stationary industry, in which Industry 4.0 describes the latest development in digitalization towards autonomous factories; see Sokolov et al. (2020).

Concerning the planning phase, simulation's ability to provide what-if analysis combined with numeric optimization is suited to significantly reduce both planning time and the risk inherent in planning. For instance, bottlenecks in the production can be identified, site layouts can be optimized to make the best use of the available spaces, or the machine park's optimal size can be predicted.

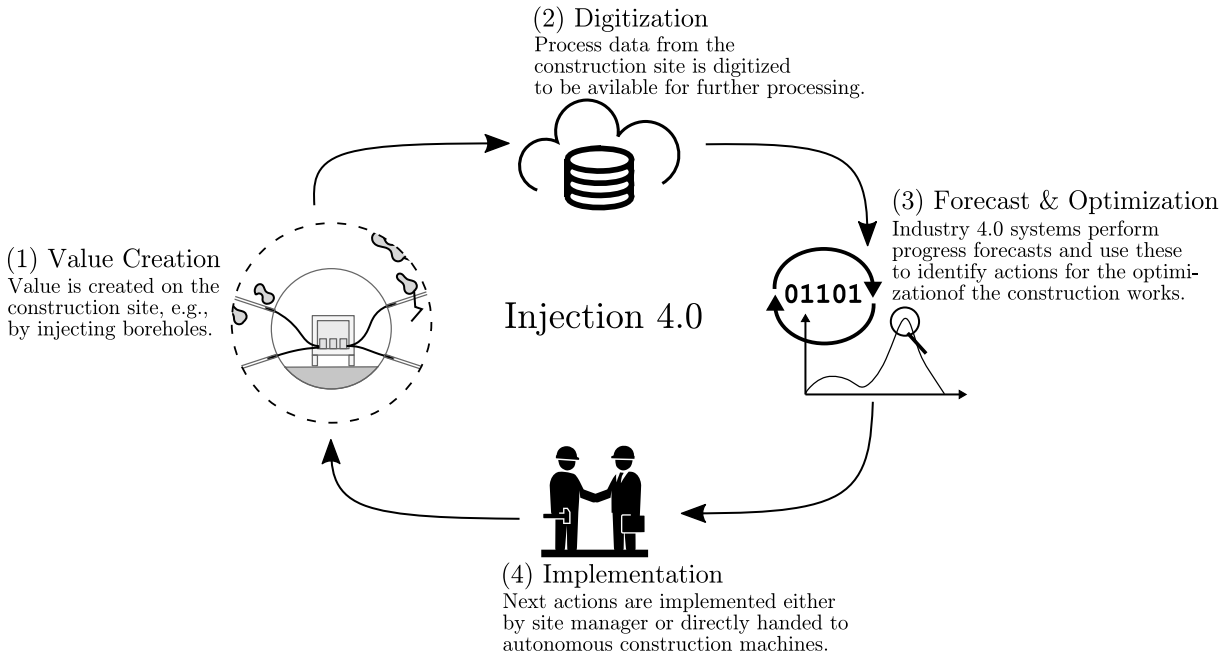


Figure 1.3: Vision: Construction Industry 4.0. *Value creation* happens on the construction site. Throughout the building phase, *digitization* of the construction site is done and made available for further processing, e. g., via cloud solution. Based on real-time data, process *forecasts* are calculated. *Optimization* algorithms provide improvement proposals to decision-makers, who *implement* the proposals on the construction site. With regard to industry 4.0 scenarios, those decision-makers may be digital construction managers commanding fleets of autonomous vehicles. Once implemented, the cycle of value creation, digitization, forecast & optimization and implementation starts anew.

Apart from support during the preparation and the execution phase, a Construction Industry 4.0 system also supports negotiations of claims. Whether in contractually agreed adjudication proceedings or before a German court, in both cases, the knowledge of the undisturbed construction project is required to determine the cost difference to the existing disturbed construction project; see also BGH (2019a,b). That undisturbed project can be simulated in a posteriori analysis. As a side effect of the increased transparency, it can be expected that the number of legal disputes decreases.

The motivation to write a thesis on the topic of digitalization of a major injection project stems from a discussion held at the conference “Digitale Infrastruktur und Geotechnik 2018” in Hamburg; see Grabe (2018). One of the guests carried a tablet computer with him. With that tablet, he graphically displayed the injection site’s real-time state he was in charge of as a project manager. The data he showed around was automatically measured by the injection units and stored on a cloud server. Finally, his device downloaded the finished analysis via a web browser. At that time, the data was primarily used for documentation and manual progress checks of the thousands of injections performed within the framework of the project. Beyond that, it was not used because, reportedly, the technology to process

the data into information with a higher value, i. e., into progress forecasts, did not exist. The general idea was to identify suitable methods for increasing the existing information's value, e. g., within the framework of a doctoral thesis. After this first meeting, it took less than a week to agree on a loose research alliance. The construction site's data was exchanged in return for the promise of occasional discussions about the possibilities of further digitalization of the construction site and access to the final research results.

This thesis deals with selecting, implementing, testing, and evaluating numerical methods to support large construction projects' construction management. Because the research alliance mentioned above provides digitized data of a large injection project, this is done on that project's example. The methods are bundled in a non-commercial expert tool to be used for later research projects. The tool, which is named GBPlan, uses real-time construction process data measured on the construction site to compute project time and cost forecasts. These predictions are then combined in a numerical optimization algorithm to determine the Pareto optimal number and operating time of construction site machinery. In this respect, this work represents the first step towards an Injection Construction Site 4.0.

Note that parts of this thesis have been pre-published. These parts are referenced in the text. Furthermore, a summary of all publications can be found in appendix E.

1.3 Thesis outline

The following chapter 2 provides an overview of existing work on simulation in the construction industry. In the first part, the most common modeling techniques and methods are being presented. The second part deals with popular *General Purpose Simulation*s (GPSs) and *Single Purpose Simulation*s (SPSs), which have been implemented to support construction management. Based on chapter 2, the motivation for this work and its goal is formulated in chapter 3. Chapter 4 introduces the methodological framework on which this thesis and the SPS GBPlan is built. Chapter 5 deals with the digitization of the construction site investigated in this thesis. A general description of the construction site and the tasks under investigation is given. The way manual construction processes are being digitized is described, and the first conclusions on the construction site's performance are drawn based on that data. As a first approach to construction performance simulation, three models for grout volume forecasts are introduced, and their performance is compared. The first model uses cross-correlation analysis, the second a stochastic model, and the third an *Artificial Neural Network* (ANN). The following three chapters focus on construction time and cost prediction and the time-cost Pareto optimization of machine deployment. In chapter 6, a time-discrete model for the construction time forecast with several production units of the same type is presented. The general way GBPlan models processes for the project under investigation is being introduced and compared against the construction site's data. In chapter 7, the model is extended by another machine type, and rules are introduced to model the collaboration of different injection units. Furthermore, the cost is being introduced as a new result dimension. With this, a time-cost Pareto optimization using *Multi Objective Genetic Algorithm* (MOGA) is being performed, and the results are

discussed compared to solutions developed on the construction site. In chapter 8, a third unit type is being introduced to the model. Furthermore, the tunnel's confined work environment added as an additional boundary condition. Graph Theory is applied to model the movement and mutual obstruction of machines in the narrow tunnel environment. Finally, that model is used to perform a Pareto optimization of the project's machine deployment strategy, and the results are being discussed compared to data collected from the construction site. Finally, in chapter 9, a summary of the main results and an outlook are given.

2 Simulation in Construction Management

> *Simulation is the imitation of the operation of a real-world process or system over time.*
>
> *Banks (1998b, p. 3)*

Looking back on the past 60 years of simulation and modeling in construction management, it is notable that the main focus of early methods, models, and simulation frameworks was to reproduce the real system's behavior. While still important today, that focus has changed over time and became more diverse. To reach out to a wider user-base, researchers increasingly aim to simplify the simulation process to make it accessible for users who are not knowledgeable in simulation techniques. This is being achieved by integrating modern *Graphical User Interface*s (GUIs) into simulation frameworks; simplifying the implementation and maintenance effort for simulation frameworks, e. g., by using *Object-Oriented* (OO) programming; and combining different types of simulations using modern *High Level Architecture* (HLA) structures.

A complete description of the field of science dealing with simulation would exceed the pages of this thesis. This is also true for the subset of publications dealing with simulation in construction management. For this reason, a firm introduction to simulation, in general, is given at the beginning of this chapter, section 2.1. The second part, section 2.2, covers the most important simulation and modeling techniques employed in the construction industry. This includes networking scheduling techniques, Petri Nets, several surrogate models, Agent Based Simulation, *Monte-Carlo Simulation* (MCS), Continuous Simulation and *Discrete Event Simulation* (DES), the latter of which is the fundamental simulation technology used for most parts of GBPlan. In the third part, section 2.3, an introduction to GPS frameworks which are popular in the construction industry is given. These frameworks have been designed to support construction managers regardless of the type of construction project they are in charge of. The fourth part, section 2.4, deals with SPS frameworks designed to support tunneling and injection projects.

2.1 The Simulation

Simulations imitate the operation of a real-world process or system over time, Banks (1998b, p. 3) and Law (2007, p. 1). An artificial history of observations of that system can be

calculated with this imitation. These observations can further be used to draw conclusions on the simulated system. The types of problems which can be solved by simulation are manifold and include: the description and analysis of the behavior of a system, what-if questions about the real system, and support in real-system design, for both, existing and conceptual systems, Banks (1998b, pp. 3 sq.). Simulation is one of many ways to

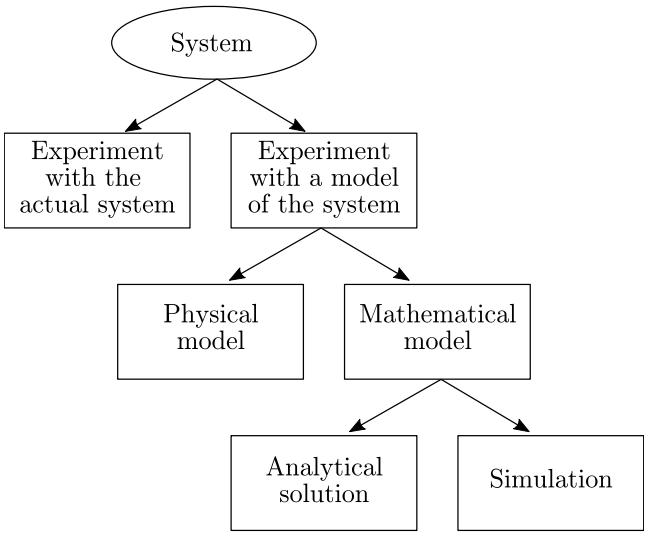


Figure 2.1: Ways to study a system. Figure adapted from Law (2007, Figure 1.1).

analyze a system. Whether or not it is the appropriate method depends on several factors, Figure 2.1. For instance, if the existing system cannot be used for experiments, e. g., because it has not been build yet, one might consider building a physical model. As those models' creation is often cost-intensive, particularly if a high number of configurations shall be considered, developers may look for an analytical solution to limit that number of configurations. If such an analytical solution is not available, performing a simulation is often a reasonable solution. Concerning large injection projects, the arguments named above explain why simulation is the only way to perform in-depth analysis. However, apart from pure necessity in the given case, simulation offers other advantages. Banks (1998b, pp. 10 sqq.); see also Law (2007, pp. 76 sqq.), summarizes these as:

Choose correctly Simulations support in decision-making as they allow the testing of different system setups without spending excessive resources.

Compress and expand time Simulations may give further insight into the system by speeding up or slowing down the system behavior.

Understand why In addition to how a system behaves, it is possible to understand why it behaves in a certain way. For instance, system inherent dependencies can be made visible by changing internal system parameters and observing others' influence.

Explore possibilities Once a real-world system has been correctly implemented, changes to that system can be added to the simulation model. The opportunity to quickly alter models provides an inexpensive way of exploring new system setups while the real-world continues to be productive.

Diagnose problems The additional insight into the internal parameters of complex systems and the possibility to change these in the simulation allow for quick debugging and improvement of complex real-world systems.

Identify constraints Simulations allow the identification of bottlenecks in manufacturing systems. These are often an effect and not the cause of a delay, resulting from other delays in the work process, information flow, material logistics, or other processes.

Develop understanding A simulation allows developing an understanding of how the production system works rather than trusting in someone's "experience", which can quickly reach its limit once something new happens.

Visualize the plan Dynamic visualization techniques allow a much faster understanding of dynamic systems than static drawings.

Build consensus It is often easier to accept tested simulation results than one person's opinion.

Prepare for change What-if scenarios can be simulated in order to develop measures if change happens, e. g., how to react if the input of a system is reduced by x percent due to constraints in the supply chain.

Invest wisely Often, changing or modifying a system is expensive. With simulation, the system's optimization can be performed without iterating costly modifications of the real-world system.

Train the team Simulations can help a team to understand how the real-world system behaves and allows the making of mistakes without having to face their real-world consequences – an essential component of learning.

Specific requirements By simulating different settings, the requirements of a machine can be assessed.

With regards to disadvantages that come with the application of simulation, Banks (1998b, p. 12) regards the following as most noteworthy:

Special training required Even though modeling software is becoming more user-friendly, building a simulation model requires training and experience.

Difficult interpretation of results Most systems work with random input variables, making it hard to assess whether the output results from the system interrelationships or just randomness.

Time-consuming and expensive Even though simulation can lead to enormous cost reduction (see above), it is not a cheap technology. Overambitious cost optimization for modeling and simulation might lead to insufficient models.

Inappropriate usage Simulation can be used even though an analytical solution is possible.

In summary, it can be stated that simulation is a powerful tool to analyze complex systems. Simulation can help to understand the system, drive down costs, and speed up the analytical process. However, to appropriately apply simulation technology, expert knowledge is required – an impediment that scientists try to overcome by creating simulation frameworks that guide users through the modeling, simulation, and interpretation process.

What type of simulation models and methods exist and how they are developed into simulation frameworks for the construction industry is being discussed in the following section.

2.2 Types of Simulation Methods and Models

This section summarizes a selection of simulation methods and models, which are important in construction management. The spotlight is on methods that have been applied in the past to forecast the duration and total cost of construction projects. Due to a lack of construction management simulations for injection grouting projects, this section's focus lies on tunneling. However, if relevant publications in that field were not found, examples from other trades are given. The following sections 2.3 and 2.4 discuss the development of GPS and SPS frameworks. These frameworks integrate the technologies discussed in this section to make them easier to use for stakeholders of the construction industry.

2.2.1 Network Scheduling Techniques

Until the 1950s, the bar chart, i. e., the Gantt chart, was the most used technique for scheduling in construction management, Baldwin and Bordoli (2014, p. 4). The technique is excellent at showing when activities occur, and it offers stakeholders a basis for discussion that enables an instinctive optimization of the construction process. However, it does not show interrelations between the single activities of the construction project, Baldwin and Bordoli (2014, p. 4). This flaw was addressed with the development of modeling techniques, which use graph theory, see section 4.4, to model these interrelations. These methods, known as network scheduling techniques, use network schedules which are normatively described in DIN 69900 (2009-01, p. 9) as a "graphical or tabular representation of a process structure consisting of operations or events and relationships". Apart from visualization and easy modification of interrelations between activities, these techniques feature the determination of early and late start dates for activities, the calculation of float time of individual activities, and in particular the ability to determine the critical path – i. e., the longest path or activity sequence through the project, Dori (2016b, pp. 16 sq.). A selection of the most common graph-based techniques is presented hereafter.

In 1956/57 the chemical group DuPont together with the computer manufacturer Remington Rand developed *Critical Path Method* (CPM) for the *Universal Automatic Computer I* (UNIVAC I) – one of the first commercial computers, O'Brien (1993, p. 5) and Swaine and Freiberger (2020). CPM belongs to the most commonly used modeling techniques in construction management. Its network representation is an *activity-on-arrow diagram* (AOA), i. e., each arrow represents an activity, and consecutive arrows represent consecutive activities, Figure 2.2. The weight of an arrow represents the duration of the activity. The nodes stand for events in the project. Each activity (arrow) has precisely one start and one end event (node). The CPM has precisely one source node representing the project start and one sink node to represent the project end. A detailed description on how CPMs are being calculated is given by Zimmermann et al. (2006, pp. 73 sqq.).

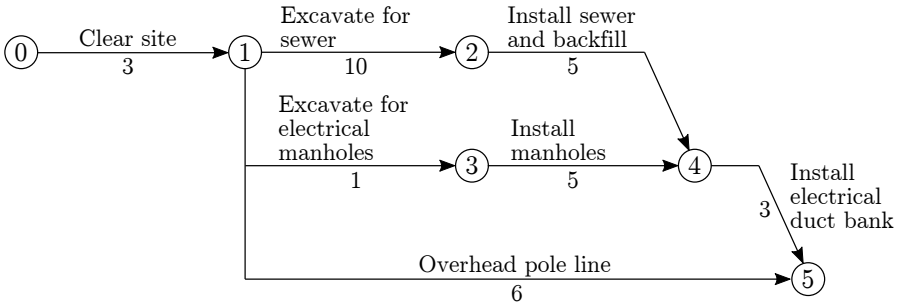


Figure 2.2: Example of CPM. Arrows represent activities, such as *Excavate for sewer* and nodes events, such as *Excavation for sewer finished*. Figure adapted from O'Brien (1993, Figure 5.1).

Parallel to the development of CPM the US Navy developed *Program Evaluation and Review Technique* (PERT) to ensure the completion of the Polaris missile project in the shortest possible time. PERT shares many similarities with CPM; for instance, the network representation of both is AOA. However, the fundamental difference is that while the duration of a process in CPM is deterministic, PERT uses a stochastic definition. Whenever a PERT activity is evaluated, its duration is drawn randomly from a probability distribution. The user has to decide beforehand which probability distribution fits its cause. In practice, the uniform, triangular or beta distribution are most commonly used; however, which distribution fits best remains part of the academic discussion even 50 years after PERT had been developed, Tesfaye et al. (2015). The use of probability distributions makes the calculation of the longest path between two nodes very complex. A good approximation can usually be achieved with a MCS; see section 2.2.3. Whether or not that approximation is accurate enough is often hard to assess a priori. This led scientists to model durations in PERT networks as fuzzy numbers, e. g., Rabetge (1990), DuBois and Prade (1989), and Herroelen and Leus (2005), which are able to model imprecise information, Józefowska and Weglarz (2006, p. 4). A good description of PERT can be found in Zimmermann et al. (2006, pp. 87 sqq.).

Around the same time, in 1959, the *Metra Potential Method* (MPM), Figure 2.3, was developed by the Metra group for the construction of the first French nuclear power plant and the french steamboat France, Alexandre (2020). It is the currently most widely used network scheduling technique, Zimmermann et al. (2006, p. 55). Designed as *activity-on-node diagram* (AON), each activity is represented as one node in the network, and the arrows between the nodes represent the interrelations between the activities. With the MPM it is possible to model all types of relations between activities easily, i. e., start-end, start-start, end-end and end-start relations, which distinguishes it from AOA networks, which make use of dummy activities, i. e., activities with a duration of zero, to model interrelations other than start-end. This makes AOA networks look complex and hard to handle. MPM nodes contain information, such as start and finish date and the deterministic duration, which makes it easier for users to understand the network. A good introduction to MPM is given by Noosten (2013) and Zimmermann et al. (2006, p. 55).

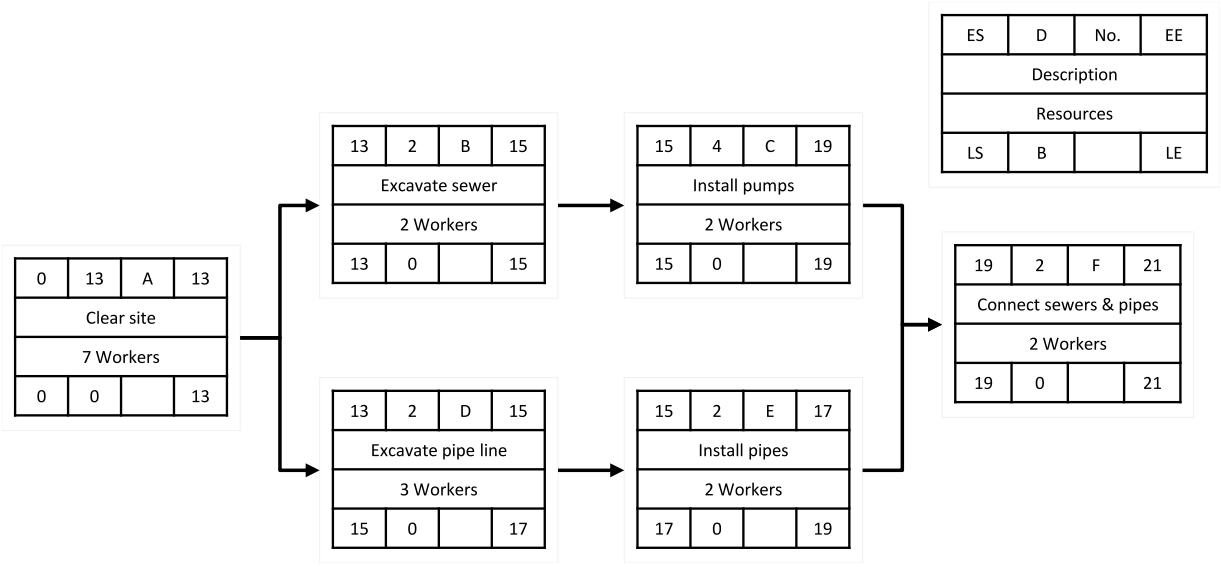


Figure 2.3: Example MPM. Each node contains a set of information, which supports in the understanding and editing of the schedule. Each node shows the duration (D), the earliest start (ES), the earliest end (EE), the latest start (LS) and the latest end (LE) of the activity. Additional information, such as the number of resources per process, a process description, or a process identification number (No.) can be added. With this notation, the critical path can be easily identified. A node n_i belongs to the critical path if $n_j \in C = \{n_i \mid ES_i = LS_i \wedge EE_i = LE_I\}$. Figure adapted from Backhaus (2018b, slide 53) and Noosten (2013, p. 40).

Graphical Evaluation and Review Technique (GERT) was first presented in 1966 by Dr. Alan Pritsker to analyze networks which contain activities that have a probability of occurrence associated with them and that treats the duration of activities as a random variable, Pritsker (1966). CPM, PERT and MPM use acyclic AOAs, i. e., each activity in the project is executed exactly one time and returning to already visited nodes of the network is not possible. For many projects, this kind of modeling is unsuitable. Zimmermann et al. (2006, pp. 95 sq.) give the example of a quality process executed during the production of semi-finished parts, Figure 2.4. Depending on a probability distribution, the process identifies a certain number of all parts as of low quality and sends them back to the work station for rework. The rework chance creates a feedback loop during which some parts are being processed several times, while others are not affected. GERT has been developed for the planning and monitoring of such stochastic processes. An introduction to GERT is provide by Zimmermann et al. (2006, pp. 95 sqq.).

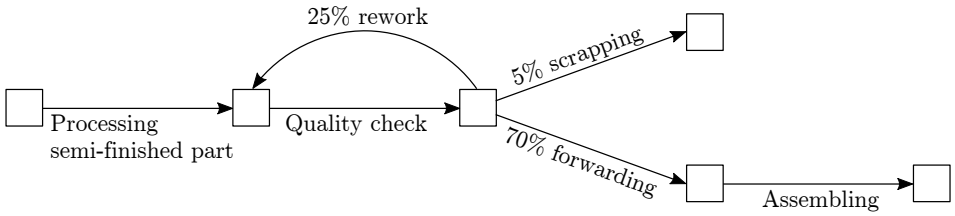


Figure 2.4: Example of GERT. Processes such as *rework* create cycles in the network. In the given example a quality check identifies semi-finished parts with a probability of $p = 25\,\%$ to be send back to the manufacturing station for rework. There is a chance of $5\,\%$ that the part is beyond repair and sent directly for scrapping. A chance of $70\,\%$ exists, that the part passes the quality check and is forwarded to the assembly station. Figure adapted from Zimmermann et al. (2006, Figure 1.50).

The presented methods belong to the most commonly used networking scheduling techniques in construction management. For further reading, in depth descriptions of these and other scheduling techniques are provided by Dori (2016b, pp. 25 sqq.), Rahm (2016, pp. 11 sqq.), Zimmermann et al. (2006, pp. 55 sqq.), Domschke et al. (2015), Haidar (2016, pp. 57 sqq.), and Runzheimer et al. (2005, pp. 143 sqq.). The presented techniques are well researched, relatively easy to understand, and can – at least for small problems – be handled with hand calculations. They can be used to compute a single schedule, which has, for instance, an optimal makespan.

A disadvantage of the presented network methods is that they cannot include additional boundary conditions, such as resource availability. If resource utilization should be considered, users have to manually push tasks forward in the schedule until all requirements are met – an iterative process that is often time-consuming to the extent that it becomes uneconomical. Another approach to include resource constraints to scheduling networks makes use of heuristic methods. For instance, Hegazy (1999a,b) uses a *Genetic Algorithm* (GA) to approximate best resource allocation based on a time-cost trade-off. Lu and Li (2003) presents a heuristic for resource allocation named *resource-activity critical-path method* (RACPM). The general procedure of these heuristics is alike. Tasks are ranked

according to their parameters, e. g., duration or earliest start time. Then the algorithms try to shift tasks according to their ranking order while staying within the resource boundaries; see also Willis (1985) and Lu and Lam (2008). These methods, however, are usually so complicated or time-consuming that they are of no use besides academic research, Dori (2016b, p. 47) and Willis (1985, p. 151). However, they remain of interest for researchers who do not tire of trying the latest optimization algorithms to produce optimal construction schedules, e. g., Dori (2016b), Hartmann et al. (2017), and Poshdar et al. (2018).

If the schedule contains many repetitive processes, network techniques quickly become overly complex and hard to handle. A solution poses Linear Scheduling Models, such as the well established *Line of Balance* (LOB) method. LOB is a visual method in which straight lines in a time-location-diagram show the linear development of repetitive tasks. Clashes between trades are easily identified at intersection points of the lines. The method is relatively simple, i. e., a spreadsheet is usually sufficient to create the output diagram, Agrama (2011). The limitations of this type of method are similar to those of the bar chart. They assume the same construction rate for all progresses, regardless at which location they take place – an approximation which provides a rough overview but is not suited for detailed project planning, Yamín and Harmelink (2001) and Baldwin and Bordoli (2014, p. 68). A good introduction to LOB is provided by Baldwin and Bordoli (2014, pp. 71 sqq.).

The described methods are still the object of active academic research. A selection of academic papers shall illustrate this:

Koo et al. (2007) present *Constraint-Loaded Critical Path Method* (CLCPM), a formal identification and re-sequencing process that supports the development of sequencing alternatives in CPM schedules. The authors combine knowledge on dependencies between limited resources and replaceable processes to identify valid schedules. CLCPM also supports re-scheduling.

Beran and Hromada (2008) present a *Microsoft* (MS) EXCEL tool implemented in *Visual Basics for Applications* (VBA) to simulate the project cost and duration based on an activity network. The tool computes dynamic progress charts if the input parameters, such as production rate, the work scope, the schedule, bonding conditions, maximum and minimum deviations from the work scope, and the production rate, are at hand. It visualizes the project's required cash flow and can plot 3D diagrams that compare cost and duration with their occurrence probability. The authors demonstrate their tool at a small project with eleven activities.

Rogalska et al. (2008) use a LOB in combination with an Evolutionary Algorithm, section 4.5.1, for the time-cost Pareto optimization of fictional, linear projects. Another approach using numerical optimization is presented by Devi and Ananthanarayanan (2015), who combine LOB with GA to create near-optimal schedules for a mass housing construction project in Nagapattinam, India.

An example of MPM in academic research is given by Christodoulou et al. (2009). The authors apply the physical principle of entropy to the resource allocation problem in scheduling resource-constraint projects. A project's entropy is a measure of its tendency to progress out of order and into a chaotic condition. The authors use the entropy as a

goal function to optimize resources in three fictional sample projects, which they modeled using MPM.

Mikulakova et al. (2010) present an approach for knowledge-based schedule generation and evaluation. The authors divide the construction project into construction parts, each of which requires a known sequence of tasks to be built. These are then connected with constraints to define their order in the overall building process. Given that the knowledge database contains the required information, the system converts BIM models to construction schedules. If alternative solutions exist, these are weighted using qualitative and quantitative ratings. The authors test their system on several theoretical small and mid-sized projects.

Network scheduling methods tend to become overly complex with the growing size of the project. Furthermore, practitioners soon realized that the application of networking scheduling techniques alone does not guarantee success. A study by the *National Economic Development Office, London, UK* (NEDO) which compared the construction performance of construction sites in the *United Kingdom* (UK), Europe and the *United States of America* (USA), came to the conclusion that a correlation between the on-time project completion and the application of network scheduling techniques does not exist, NEDO (1988) and Baldwin and Bordoli (2014, p. 5). This general flaw and the rise of computer technology led to the development of a range of simulation frameworks, which are being discussed in section 2.3 and 2.4.

2.2.2 Petri Nets

The *Petri Net* (PN) is named after Carl Adam Petri, who developed them in the mid-60s, Petri (1962, 1966). A PN is a model for discrete systems. The net provides a mathematical and graphical description of the system it models. It is a directed, weighted, bipartite graph, see section 4.4, which consist of four basic elements: *Places* (circles) represent states of the system; *Transitions* (squares) are activities in the system, which may change the state of the system when triggered; *Links* or arcs (arrows) connect places with transitions and describe the movement or flow of *Tokens* (black dots), Figure 2.5. Tokens can represent resources, and trigger transitions, Karatkevich (2007, pp. 10 sqq.). Wakefield and Sears

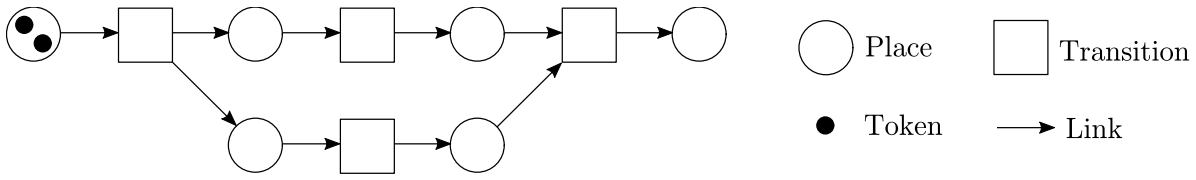


Figure 2.5: Petri nets consist of four elements: Places, transitions, tokens and links or arcs.

(1997) give the illustrative example of a PN modeling the work sequences a crane experiences when hoisting materials, Figure 2.6. The mathematical description of PNs allows methods and algorithms to identify flaws in their structure, such as blocked paths or insufficient synchronization of processes, Berkhahn (2007, p. 22).

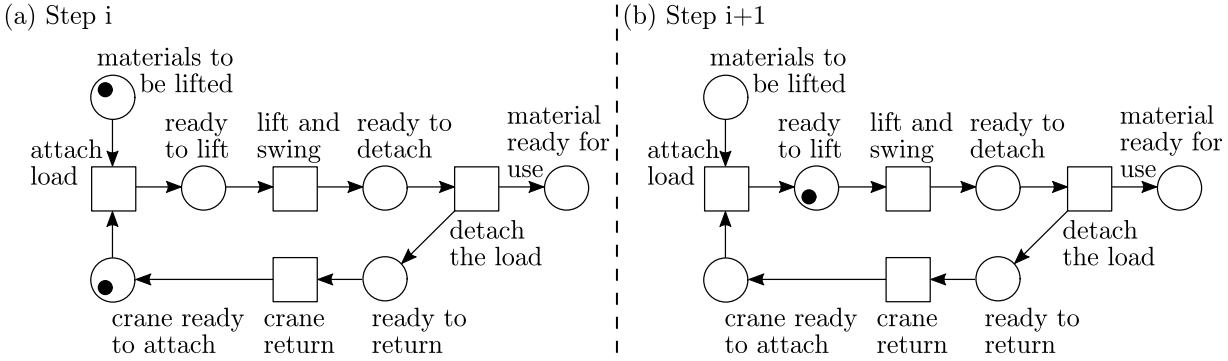


Figure 2.6: Example: Petri net model of crane hoisting materials. (1) Two conditions must be satisfied for the crane to lift and swing. First, the materials to be lifted must be accessible, and second, the crane must be ready to attach the materials. When these two conditions are fulfilled, then (2) the transition *attach load* fires, and the system is in the state *ready to lift*. Figures adapted from Wakefield and Sears (1997, Figure 1).

*Timed Petri Net*s (TPNs) model time by relating a time delay to transitions. This time delay can be deterministic or represented by any distribution function, Balbo and Chiola (1989). Cases of conflict, e. g., if two transitions compete for the same resources (Token), can be solved by giving priorities to transitions, Wakefield and Sears (1997, pp. 107 sq.). If the tokens of a PN are associated with different attributes, that PN is called a *Colored Petri Net*s (CPNs). Concerning the example in Figure 2.6, a CPN would be able to model different materials hoisted by the crane. These materials can be treated differently, depending on their parameters. For instance, bricks can be delivered to a wall while concrete is lifted to the next base plate. If the equipment fleet is made of different machines, a CPN can model queuing situations, Sawhney (1997) and Wakefield and Sears (1997). Limitations of PNs are named by Martinez and Ioannou (1999), who point out that bulk material cannot be adequately be represented. For instance, the transfer of soil or aggregate between two locations cannot be directly modeled. A comprehensive introduction to PNs is provided by Baumgarten (1996) and Reisig (2010).

Franz (1989) uses higher PNs to model the work chain excavator-truck and the transport of concrete material on construction sites, including stochastic parameters to model disturbances in the production chain. Wakefield and Sears (1997, pp. 107 sq.) model a subsystem of the earthworks construction system used in building contracts for the New Hong Kong International Airport and the placement of concrete slabs to demonstrate the capabilities of PNs. Berkhahn (2007) presents ProMiSE, a software tool to support the modeling, analysis, and control process based on PNs. The author demonstrates the tool on the planning process of a pile-raft-foundation; see also Berkhahn et al. (2005) and Rüppel (2007). Cheng et al. (2011) use a CPN to model earth moving operation for a high-rise residential project in Harbin, China. Sawhney and Vamadevan (2000) use a TPN to schedule a bridge project located in Portage, MI. Cheng et al. (2013) use a PN to model the building design process in construction under consideration of information constraint relations among different design activities to optimize resource allocation throughout the

development cycle. Despite the examples mentioned above, PNs are relatively rare in the construction industry. Dori (2016b) attributes this to their higher complexity compared to other simulation methods.

2.2.3 Monte-Carlo Simulation

The underlying principle of this type of simulation algorithms is to take repeated random samples to calculate a set of results for a given problem. A subsequent, statistical analysis of the result set is then used to estimate the result most likely to realize, Law (2007, pp. 73 sqq.). The method was originally developed to perform numerical integration of functions that were difficult to solve through analytical methods. One of its features is its high flexibility, which allows the user to solve complex problems with relatively simple techniques. A critical disadvantage offsets this advantage: MCSs are usually very cost-intensive, i. e., they have a long computation time.

Today MCS is widely used in risk analysis, in particular, to compute the risk inherent to complex financial products such as in derivatives and options trading, Dunn and Shultis (2012, pp. 2 sq.) and Choi and Kang (2013, pp. 12, 15). Examples of the application of MCS in the field of construction management are given in section 2.2.7.3. The mathematical foundation of MCSs, the *Monte-Carlo Method* (MCM), is being described in section 4.2. A special form of MCS, the *Markov-Chain Monte-Carlo Simulation* (MCMCS) in which MCS is is combined with Markov-Chains is being discussed in section 4.3.

A short introduction to MCS is given by Bättig (2015, pp. 100 sqq.) and Law (2007, p. 73). For an in depth discussion on MCS and its variations see Dunn and Shultis (2012), Robert and Casella (2004), and Monahan (2011, p. 257). For an enjoyable and exciting summary of the history of the MCS see Dunn and Shultis (2012, pp. 2 sqq.).

2.2.4 Agent Based Simulation

An *Agent Based Model* (ABM) creates an artificial world in which heterogeneous agents interact with each other or the world, e. g., with time and space. The model sets the agents' rules, which tell them what they can do depending on the current circumstances. A key feature of ABMs is that agents have different characteristics, i. e., they behave differently in the simulation environment. With abilities, such as autonomy, perception, reasoning, assessing, learning, goal processing Dori (2016b), Mohamed and AbouRizk (2005), and Oren et al. (2000) label agents as intelligent modules. An early ABM described by Epstein and Axtell (1996) is the Sugarscape model. Based on a 51×51 cell grid, each cell may contain a different amount of sugar units. Every time step of the simulation, agents try to find the closest cells with a sugar unit, move to that cell, and metabolize the sugar. Depending on the simulation's goal, agents may reproduce, die, or be born, transfer information, create pollution and transmit diseases. Squazzoni (2010) gives a summary of the development of ABMs since the mid-1990s. For an introduction on ABMs Gilbert and Troitzsch (2005) and Lynne Hamill (2016) is recommended.

The ABMs is a powerful tool to simulate complex systems with many individuals and their interactions, as is the case on construction sites. However, a clear disadvantage of ABMs is their complexity, which results in high development cost and a loss of transparency. The connection of Micro- to Macro level is not trivial. It is often not easy to assess a priori how changing the set of rules of single agents will reflect on the macro level, e. g., if they might bring the whole system to a standstill and thus render the model unusable. Furthermore, systems with a high number of agents may require extensive computation power, Kugler and Franz (2007).

An example of the application of ABM in construction management is given by Mohamed and AbouRizk (2005), who use that technology for the minimization of *Tunnel Boring Machine* (TBM) idle times in tunneling. Snyder et al. (2017) perform a simulation with an ABM to explore how the water supply system of ancient Constantinople had been constructed. Kugler and Franz (2007) present a concept for agent-based modeling of high-rise construction buildings, in which elements such as cranes and vehicles are modeled as individual agents. Building on Knotts et al. (2000) the authors of Horenburg et al. (2012) present an ABM which improves scheduling of construction projects by ensuring the intelligent allocation of resources. Kim and Kim (2010) present SIMCON, a construction site simulator which takes traffic flows (and its congestion) on the construction site into account. They demonstrate their model on the example of an earthmoving operation over multi-lane, traffic light controlled roads. Kooragamage et al. (2013) present an ABM for the planning of construction site logistics on the example of earthmoving operations of the London Gateway Port project. The model takes spatial time clashes between the construction objects and resources into account, i. e., it considers that the construction site layout is changing as the project progresses.

2.2.5 Continuous Simulation

System Dynamics (SD) was developed around 1960 by Jay W. Forrester to support decision-making for policy makers in economics, Forrester (1958, 1961, 1989, 2003) and Kleiner (2009). It is a modeling method that simulates the continuous development of a system over time. This continuity distinguishes it from DES where the state transitions of the model are assumed to happen instantly. With time passing continuously, the interdependence in SD models can be expressed through differential equations. If the systems have a complex structure, the solution can often not be expressed analytically, for instance, if the system contains a feedback loop. SD solves this problem by a simple but effective working principle, which consists of feedback loops and flow or growth rates. An example is given in Figure 2.7. A short introduction to the topic is provided by Law (2007, pp. 70 sqq.). For a comprehensive introduction to SD see Bala et al. (2017). Examples for SD software packages are products such as Power Sim by Powersim Software (2020), Stella by ieee Systems (2020) and Vensim by Ventana Systems (2020).

Despite its success in economics SD is rarely used for simulation in operational construction management, Han et al. (2014, p. 219). The reason might be the mutual lack of detail of the models on the operational level. Love et al. (2002) use SD to describe how changes and rework can impact the project management for construction projects. Han et al. (2005)

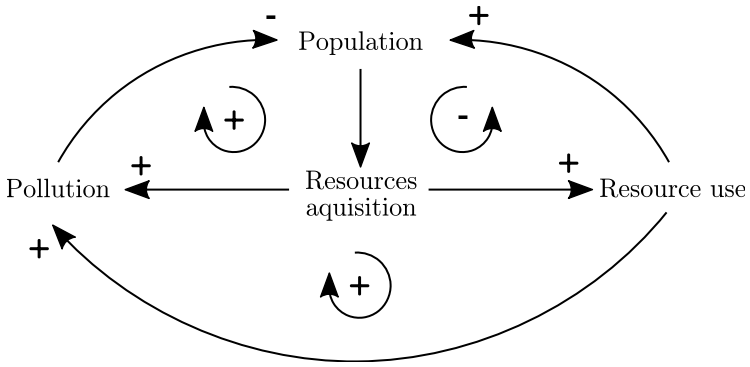


Figure 2.7: A simple SD model. Growth of the earth's population would lead to an increase in the number of resources to be requisitioned by that population. Acquiring more resources leads to an increase in pollution, which has a detrimental effect on the population. The same is true for the use of resources. However, the use of resources also has the effect of easing the existence of the population, which adds to its growth. These three loops determine the ultimate size of the population. If the effect of pollution from the use and acquisition of resources (negative loops) outcasts the benefits of using resources (positive loop), the size of the population declines. Figure adapted from Bala et al. (2017).

compare the performance of an SD model for earthmoving operations project with the same project modeled in STROBOSCOPE, a DES software. The authors conclude that the SD model can address the operational details as accurately and reliably as the DES model. Rahm et al. (2012) and Dang et al. (2018) use SD to model material flows on tunneling construction sites. Their hybrid models combines SD and DES.

Nasirzadeh et al. (2008) present an SD model to analyze risk management in construction projects. The model uses fuzzy logic to model the imprecision and uncertainty involved in the description of risks. Another SD risk model, however, without fuzzy logic elements, is presented by Wan and Liu (2014). Mohamed and Chinda (2011) present a method to model the interactions of safety culture enablers. The authors use the model to analyze construction sites' safety culture under the assumption that, by improving these enablers, there is also an improvement in safety performance on the construction site Jiang et al. (2015) use SD to understand the cause of unsafe behavior of construction workers. After identifying individual and environmental conditions that may affect safety, the authors establish the model and calibrate it during a five-week survey on a building construction project. Maryani et al. (2015) present a model for construction accidents. The model takes all supply chain members into account and can calculate the direct and indirect damage caused by construction site accidents. Based on this data, the simulation identifies operational safety and health components that need to be controlled and suggests improvements in subcontractors' and supervisors' supply chain.

A model for labor productivity in construction projects is presented by Nasirzadeh and Nojedehi (2013). Leon et al. (2018) describe their SD approach for forecasting the performance of construction projects under unit price contracts. The model uses eight perfor-

mance dimensions, including cost, schedule, quality, profitability, safety, environment, team satisfaction, and client satisfaction. The authors test the model on a road construction project to simulate and analyze four possible intervention scenarios for project managers.

Nasir and Hadikusumo (2019) model the contractual relationship between the owner and the contractor of construction projects. After testing the model on real data, it is used to analyze the effectiveness of different policies. An interesting finding is that the combined effect of policies made before the construction stage is more influential on relationships made during the construction stage than the combined effect of policies made during the construction stage. Yildiz et al. (2020) present an SD model for a dynamic Strategy Map based on the Balanced Score Card framework. The Balanced Score Card and the Strategy Map are both recognized management methods to plan and outline organizations' strategy, Kaplan and Norton (1992, 1996, 2000, 2004). The presented model is meant to support the implementation of valid strategies for construction projects. Gilkinson and Dangerfield (2013) propose an SD model to analyze the competitiveness of the construction sector. Based on their results, the authors provide a strategy guide to assist decision-making in contract procurement and execution.

A literature study by Liu et al. (2019) comes to the conclusion, that SD is increasingly applied in the construction sector; see also Xu and Zou (2020). The share of applications in the area of site and resource management is, however, rather small. The majority of research papers deal with project planning & control, risk analysis, sustainability, and performance analysis, which is backed by the findings of this literature review.

2.2.6 Discrete Event Simulation

*Discrete Event Model*s (DEMs) include detailed representations of their internal components and interactions. This sets DEMs apart from other models such as mathematical, statistical, and input-output models. These only provide explicit representations of their respective in- and outputs but not their internals. Banks and Cason (1984, p. 53) provide definitions of a number of general concepts used in DEMs. These are:

System A collection on entities which interact with each other towards a common goal.

Model The abstract representation of a system.

System state Collection of variables, which are suited to describe the system.

Entity An object in the system, e. g., a customer or a server.

Attributes Properties of an entity, such as the speed of a conveyor belt or the arrival time of a customer.

Set A collection of associated entities, such as all machines or all boreholes of the construction project.

Activity A duration such as service time, with a length that may follow a statistical distribution.

Delay A duration of indefinite length during which nothing happens. The duration of the delay can be discrete or statistically distributed.

A DES simulates the behavior of a DEM over time. The simulation lets the system's state change only at discrete points in time. Those states s_i are reflected in parameter sets which describe the states' features p_i. The change from state s_i of the system to the following state s_{i+1} is called a transition, is triggered by an event, takes place in an instant, and effectively changes the parameters of the system, such that $p_{i+1} = f(p_i)$, Choi and Kang (2013, p. 5), VDI 3633 Blatt 8 (2007, p. 49), and Banks (1998a, p. 8). Events originating from within the system are called endogenous events, and such that originate from the outside are called exogenous events, Banks (1998a, p. 6).

In general, four types of simulation strategies are being distinguished: Process-Interaction, Event-Scheduling, Activity Scanning, and the Three-Phase Method, Banks (1998b, p. 9). In *Process-Interaction*, the flow of an object through a system is simulated. The flow describes all states the object can take in sequence, from the moment the object enters the system until it leaves the system. The basic idea of *Event-Scheduling* is that time is advanced until something happens, i. e., if the system is idle and an event causes the system to be idle even longer, then time does not stop. The event is skipped and time advanced until a meaningful change to the system takes place. *Activity Scanning* also coined as the *Two-Phase Method* consists of independent modules waiting for a predefined condition to be met until they act. The model is scanned for these conditions at predefined time increments. If a condition is met, the module acts, and the system state changes. The fourth strategy is the *Three-Phase Method*. In the first phase, time is advanced until something meaningful happens, e. g., a state change of the system. In the second phase, i. e., when time stops, the system is scanned to identify all events that occur at that time. Furthermore, changes to the system are being made. Finally, during the third phase, new activities are started based on the system's current state. An in-depth explanation of the four simulation strategies is provided by Balci (1988).

The main difference between the four strategies is how they model time. More generally spoken, time in DES can be modeled as a continuously changing parameter t, which changes in fixed increments δt, such that $t_{i+1} = t_i + \delta t$. This approach, albeit intuitive, is very costly. This is because at most of the calculated time steps t_i, the model parameters p_i will not change, and thus computation time is wasted. Therefore it is expedient and common to model time steps with varying increment size δt. At each transition, the time increment until the next transition occurs is being calculated and added to the simulation model's internal clock.

Example: Figure 2.8 shows a single server model in which a storage unit is fed with semi-finished products. A machine removes items from the storage and processes them into finished goods. The parameters of the system are the number of items in stock $s \in \mathbb{Z}_0^+$ and the current status of the machine $m \in \{\text{busy}, \text{idle}\}$. The events which might occur in the system are $E \in \{\mathit{Arrive}, \mathit{Load}, \mathit{Unload}\}$. An internal clock keeps track of the time t_s in the simulation. The arrival time $t_{a,j}$ is the time at which the event *Arrive* takes place, and the j^{th} semi-finished product arrives at the storage hopper. The DES checks at each simulation step whether or not the storage hopper contains any semi-finished products. If so, the item j with $min(t_{a,j})$ is removed (i. e., *Loaded* to the machine) and its processing time t_p is calculated. Finally, as part of the *Unload* event, the internal clock is being updated: $t_{s,i+1} = t_{s,i} + \delta t = t_{s,i} + t_p$. In the case that the storage hopper contains no items, the

program simply sets the internal clock to $t_{s,i+1} = t_{a,j+1}$, i. e., to a time when the next *Arrive*-event takes place.

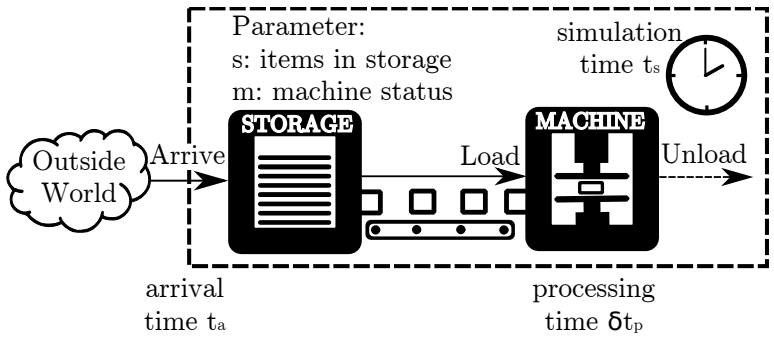


Figure 2.8: Single Server Model. Items arrive from the outside world with an arrival time interval t_a. Once within the system's boundaries, the item is stored in a storage hopper. From here the items are loaded to the machine. The machine processes the items in δt_p time units. Once processed the item is unloaded and the machine is ready to process the next item in the storage hopper.

In *hybrid models* the features of discrete and continuous simulation are being combined. The latter uses methods based on differential equations to model continuous change of the parameters sets, VDI 3633 Blatt 8.

The type of simulation promoted in this thesis is the DES. Banks (1998a,b), Banks and Carson (1984), Law (2007), Pidd (1989), and Schriber and Brunner (1998) give an introduction to the topic. Choi and Kang (2013) is a more recent textbook on the subject. A comprehensive selection of publications describing DES as means for simulation in construction management is presented in the following chapters.

2.2.7 Surrogate Models

This section discusses a selection of surrogate models that are not based on process simulation. In the shortage of good literature on the prediction of tunnel injection projects, the selected models deal with supporting tunneling project management.

2.2.7.1 Empirical / Analytical Models

Several analytical and empirical models for the prediction of TBM progress parameters have been developed in the past. An example of one of the most commonly applied models is the semi theoretical *Colorado School of Mines* (CSM) model. It was developed by Ozdemir et al. (1978) and further advanced by Rostami (1997). The model correlates TBM rock cutting wheel parameters with the penetration rate in millimeter per revolution. The model's input includes material parameters of the rock and geometric characteristics of the cutting wheels. A statistical analysis of a database obtained with full-scale tests serves as the basis for the model. Other popular models are the empirical *Norwegian University*

of Science (NTNU) model, Bruland (1998, 2000), and the QTBM-Model, Barton (2000) and Barton (1999). Further models from this category are presented by Bamford (1984), Gehring (1995), Gong and Zhao (2009), Hassanpour et al. (2011), Roxborough and Phillips (1975), and Yagiz (2002).

A model that directly forecasts project cost based on cost curves for the construction project's individual components is presented by Walski (1980). Furthermore, cost curves have been implemented in several computer programs developed in the 70s. Wheby and Cikanek (1973) present COSTUN, a computer program developed in 1973 by Harza Engineering Company. The program uses empirical equations based on regression analysis of previous tunnel projects. In addition to different structural tunnel characteristics, COSTUN allows distinguishing between several geologic conditions for the project's total cost forecast; see also Bennett (1981, pp. 17 sqq.). Another model of that decade and time is the *Tunnel Cost Model* (TCM) by Moavenzadeh et al. (1974); see also section 2.4.2, which incorporates uncertainty in its submodels. The *performance/cost Tunnel Model* (TM) was implemented in FORTRAN 4 for the CDC 6000 mainframe computer (Spicer, 2000) by Hibbard and Pietrzak (1972) is very similar to COSTUN and TCM. Like COSTUN it includes a discrete geological model. The model does not include uncertainty but allows for a three-dimensional description of soil profiles. The forecasting tool *Transportation Systems Center/Foster Miller Associates* (TSC/FMA) was written in BASIC for the Wang 2200 minicomputer and draws its input from two databanks, one with factors for effort values per unit and one with cost factors for labor, equipment, and materials. The data had been collected from 20 soft ground tunnel projects in Washington, D.C., the only type of tunnel TSC/FMA may be used for, Bennett (1981, pp. 32 sqq.).

A problem shared by analytical/empirical models is that they are usually not precise enough to have a meaningful impact on the forecast accuracy because they do not take various conditions and their related parameters into account, Koopialipoor et al. (2018, p. 3800).

2.2.7.2 Artificial Neural Network Models

Since the turn of the millennium, modeling with ANNs, section 4.6, has become increasingly popular. In the following, a selection of models is presented, which use this modeling method.

Grima et al. (2000) use a neuro-fuzzy approach to model penetration rates in TBM rock excavation. The authors use data from 640 TBM projects, which were recorded in Nelson et al. (1994) as input for the model. The data is made of a logical component based on fuzzy set theory and a numerical component using ANNs.

Benardos and Kaliampakos (2004) create an ANNs model to predict the advance rate of tunneling in soft ground. The model takes the geological site conditions into account. Inputs are based on historical data relating to the subsurface's geological characteristics and the specific site conditions. They explicitly exclude disturbances due to machine failure. The tool is customized for and tested on the example of constructing a section of the metro tunnels in Athene, Greece.

Yagiz et al. (2009) present an approach to build a non-linear multivariable prediction model to estimate TBM performance as a function of rock properties. The authors use a database of rock properties from a tunnel project in New York, NY, which had been collected as part of another research project; see Yagiz et al. (2010). Two different prediction tools are used with that database, and their results are compared: the ANN and the non-linear multivariable regression. The authors conclude that both models are suitable for the intended cause.

Javad and Narges (2010) use an ANN to model the penetration rate of hard rock TBMs. The authors collect data from three projects: the Queens Water Tunnel in the USA; the Karaj-Theran water transfer tunnel in Iran; and the Gilgel Gibe II hydroelectric project in Ethiopia and use the resulting database for the training of different ANNs, of which a five-layered ANN shows the best correlation. The results suggest that the method is suitable for TBM performance prediction.

Lau et al. (2010) combine ANN with radial basis function to forecast the production rate of the next production cycle of drill-and-blast tunneling operations. After each production cycle, the model is updated with the latest results and retrained. A case study on the example of a tunnel project in Hong Kong is performed.

Leu and Adi (2011) deal with the problem of the exact prediction of geological conditions, the knowledge of which would result in a considerable improvement of the prediction of performance values. Their approach combines a *Hidden Markov Model* (HMM) with ANN. The model works with the latest data from the construction site, i. e., it is regularly updated and retrained. The authors validate their project on a drainage tunnel project in Taipei, Taiwan.

Petroutsatou et al. (2012) discuss an early cost estimation system for road tunnel construction based on ANNs. The authors use geological, geometric, quantity-related, and cost-related data from 33 twin tunnels with a total length of 46 km, which was collected during the construction of the Egnatia Motorway in Greece. The authors compare the performance of two methods: a multi-layered *Feedforward Neural Network* (FNN) and a general regression ANN; see also section 4.6.1. They conclude that general regression ANNs are more accurate than the former. However, both models are robust enough to be used for early cost estimates.

Moore and Pham (2012) propose a context-aware tunneling system using hybrid ANNs for the prediction of TBM performance and risk response. The model uses real-time data in the form of data series and fuzzy rules. The hybrid ANN then combines fuzzy theory with self organized maps (a special form of ANN; see Agarwal and Skupin, 2008; Kamimura, 2011; Patan, 2019) to predict the TBM performance. The authors use data from several unspecified tunnel construction projects in the Asian region to test their model and conclude that they are suited to provide meaningful forecast results. However, it must be noted that 83 % of the input data had been used to train the model. Hence, only 17 % of the data were used for its validation. It remains to be seen whether the model can work in tunnel projects that are not related to the data set.

Mahdevari et al. (2014) describe a regression model to predict the penetration rate of a TBM in hard rock conditions using *Support Vector Regression* (SVR) – an *Artificial*

Intelligence (AI) algorithm; see Cortes and Vapnik (1995) and Vapnik et al. (1996). The authors use 80 % of the data collected from the Queens Water Tunnel project in New York City to train, and the remaining 20 % of data to test the model. They conclude that the SVR-model is useful and reliable if a suitable data set exists.

Yagiz and Karahan (2015) use historical data to develop several models for estimating TBM performance. The models are computed using a number of optimization algorithms including the hybrid harmony search (HS-BFGS) algorithm, Geem et al. (2001); differential evolution, Storn and Price (1997); and grey wolf optimization, Mirjalili et al. (2014). The authors compare their results in a case study involving the Queens Water Tunnel in New York and conclude that HS-BFGS is superior to the other algorithms, particularly if computational time and efficiency are critical.

Koopialipoor et al. (2018) present a method based on the *Group Method of Data Handling* (GMDH), a form of ANN, for predicting the penetration rate of a TBM. The authors compare the performance of the GMDH with those of five multiple regression models. They conclude that although both methods are applicable for estimating the penetration rate, the GMDH provides a higher degree of accuracy; see also Kalantary et al. (2009).

Zhou et al. (2019) compare the ability to forecast the TBM advance rate of two methods, namely ANN and genetic programming. The authors use statistical data from the Pahang-Selangor Raw Water Transfer Tunnel project in Malaysia as input for their models. They conclude that the genetic programming model yields significantly better results than the model using ANN technology.

Mahmoodzadeh et al. (2020) use two different AI techniques to forecast the geology, construction time, and the construction cost of the Garan road tunnel project in Iran. The *Gaussian Process Regression* (GPR), described by Pal and Deswal (2010) and Rasmussen and Williams (2005), and the SVR. Data collected during historical road tunnel projects were used as input for the training phase of the algorithms. Throughout the project, finished sections of the tunnel were added to the training data set to improve forecast accuracy continuously. The authors conclude that GPR is superior to SVR. However, both techniques yield good prediction results, which reduced the uncertainties in the project.

For further reading, Koopialipoor et al. (2018, p. 3801) provide a table which sorts a selection of publications according to the performance parameters used by the presented AI algorithm for the prediction of penetration rates or other utilization indices.

2.2.7.3 Fuzzy- and Stochastic Models

Apart from empirical/analytical and the more complex ANN models, scientists have made attempts to model the construction performance in tunneling using statistical methods and fuzzy logic. A selection of publications making use of these methods is presented hereafter.

Simões and Kim (2006) develop a TBM utilization predictor model using fuzzy logic. Rule-based (Mamdani model) and parametric-based (Sugeno model) fuzzy logic were adapted to model subjective and unquantified TBM field data sets. Data from three hard rock drives were used to correlate rock mass properties and utilization factors. After comparing

simulation results against field data, the authors conclude that the parametric-based model is superior due to its ability to provide more accurate and smoother results.

Girmscheid and Busch (2008, pp. 138 sqq.) discuss MCS for project cost and risk analysis in construction projects. The method is generally suited to provide meaningful decision support; however, due to the individual character of construction projects acquiring the input data for the MCS can be difficult if not impossible. As a workaround, the authors suggest a modified version of the Delphi Method, see Armstrong (2001) and Dalkey and Helmer (1963), to acquire missing inputs. The modified method differs from the original in that it contains fewer decision cycles, which are deliberately limited, to speed up the process.

Demmler (2009) presents a process model for risk management in mechanized tunneling projects following the *Fédération Internationale des Ingénieurs Conseils* (FIDIC) Red Book recommendations. The model summarizes risks from different categories, such as conceptual, technical, legal, financial, scheduling, environmental, and management risk, during the offer phase. Throughout the execution phase, the model is updated with data from the construction site reports to determine the current risk development, Demmler (2009, pp. 19, 191 sqq.). While such models provide a methodology for risk assessment, they heavily rely on experts' opinions, which is in the best case collected using structured communication methods, such as the Delphi Method. In particular, this impedes risk assessment for scheduling since it can be assumed that a sufficiently large group of experts is not always available to determine deadline risks.

Yagiz and Karahan (2011) use the particle swarm optimization method, see Kennedy and Eberhart (1995), to compute models for TBM performance prediction. The authors share the opinion that their models are superior to others using Fuzzy Theory or ANNs because their model's inner working is more transparent than others. However, if the accuracy of the predicted results measured their approach's quality, the judgment would be different. Their approach would produce its best results only if 80 % to 100 % of the data set were used to train their models. The required amount of training data limits their method to very late project phases or a posteriori analysis.

Oraee and Salehi (2011) compare the results of two prediction models for advance rates of TBMs based on real project data from the 16 km Karaj-Teheran water supply tunnel project. The first is an experimental model developed at the Norwegian Institute of Technology, and the second a mathematical model developed by the Colorado School of Mines. Based on the analyzed tunnel project, the authors conclude that experimental models are superior to mathematical models, partly because the latter are very sensitive to small changes in input parameters.

Špačková et al. (2013) and Špačková and Straub (2012) introduce a dynamic Bayesian network model for probabilistic assessment of tunnel construction performance. The model quantifies uncertainties in the construction process and the risk from ordinary events that may cause severe delays or damages. The risk model for geotechnical conditions is adapted from the *Decision Aids for Tunneling* (DAT) model described in section 2.4.2. Both risks from human and other external factors are addressed. The authors use data from three tunnel projects and one artificial test project to show their model's general qualification.

Ghasemi et al. (2013) use MATLAB to develop a fuzzy logic model to predict the penetration rate based on collected data from the Queens Water Tunnel project in New York City, USA – a hard rock TBM tunnel. The model uses rock properties such as uniaxial compressive strength, rock brittleness, the distance between planes of weakness, and the orientation of discontinuities in the rock mass to predict TBM penetration rates. The authors show that the fuzzy logic model yields better results than those computed with linear multivariate regression, non-linear multivariate regression, and particle swarm optimization techniques. They see the most significant advantage of their fuzzy model in its ability to include human judgment and intuition in the prediction process. Finally, the authors strain that the developed model works only for the Queen Water Tunnel project.

Vargas et al. (2014) use MCS to forecast the construction time of a 560 m tunnel built to access a Chilean copper mine. The tunnel is built with the drilling and blast method. The scientists use past projects' experience to compile the probability distributions for each construction process, namely drilling, load and blasting, ventilation, scaling, and mucking. The authors state that, apart from a reasonable estimation of the mean construction duration, the minimum and maximum duration computed by the MCS was a great help to decide on appropriate construction scenarios and minimize financial risk. In Vargas et al. (2015) the authors extend the method to a MCMCS by using a Markov chain to determine whether or not a process has been interrupted by a shift change and adding the additional restarting time to the duration of the interrupted process.

Maji and Theja (2017) use a stochastic model to predict TBM performance. The model assumes normal distributed input parameters and performs a MCS, see section 2.2.3, to calculate the statistical distribution of the penetration rate. The results are compared with not further described previous tunnel construction projects and the results of three other empirical models: the CSM model, the QTBM-Model and the NTNU model, mentioned in section 2.2.7.1. The authors conclude that the distribution obtained for each project compares well with actual penetration rates.

2.3 General Purpose Simulation Frameworks in the Construction Industry

GPS frameworks are created with the intent to simulate every type of construction operation. These frameworks had been developed to overcome the well-documented limitations of popular network models, such as CPM and PERT, MacCrimmon and Ryavec (1964), Levitt et al. (1988), and Pritsker (1989). The main problem is that these networks do not consider typical construction projects' features because they have been developed for the aerospace and manufacturing industry. Sawhney (1997) lists three major characteristics of construction projects that limit the use of traditional network techniques:

1. The many stakeholders in construction projects, e. g., architects, structural engineers, general contractor, and special contractors, complicate decision-making and flow of information. Hence, planning and scheduling are more complex; see also Forbes and Ahmed (2011, p. 60).

2. Compared to the stationary industry, construction projects are situated in a dynamic environment. Weather changes, labor productivity, skill fluctuations, and varying site conditions such as changing soil parameters, add a stochastic element to the construction process; see also Köster (2007, p. 64), Low Sui (2001, p. 417), Uhl (2011, p. 209), Forbes and Ahmed (2011, p. 60), Nübel et al. (2016, pp. 101 sq.), and Paez et al. (2005, p. 235).
3. Resources required for the work process are dynamically allocated. For instance, a tower crane is supporting different trades and tasks. It may be used to arrange steel parts, unload precast concrete slabs from a truck, or lift a concrete bucket to pour a wall.

These characteristics also distinguish the in-stationary construction industry from stationary industries and can be supplemented by at least three more, namely:

4. Contracts for construction projects usually contain a penalty clause in case of underperformance, Forbes and Ahmed (2011, p. 58). Such a clause is difficult to model in most network techniques.
5. The planning process is often separated from the project execution, which is why executing companies have only limited influence on the planning process, Köster (2007, pp. 64 sqq.). This separation is, at least for the German construction industry, implemented in regulations such as *Vergabe- und Vertragsordnung für Bauleistungen* (VOB) and *Verordnung über die Honorare für Architekten und Ingenieure* (HOAI) published in Bundesanzeiger (2013, 2016, 2020).
6. Construction works often undergo significant customer-initiated changes, resulting in many minor and major changes of the planning during project execution, Köster (2007, p. 64). This often results in the initial schedule becoming obsolete during the project.

In the following, one of the earliest GPS frameworks, *Cyclic Operation Network* (CYCLONE), is firmly being introduced and its development over the years outlined. In the second part of this section, other GPS which were inspired by the success of CYCLONE, are being described.

2.3.1 CYCLONE – A Popular GPS Framework

One of the first GPS frameworks is CYCLONE, which was presented in the 70s by Halpin (1977). It is a modeling technique designed to allow practitioners with limited simulation background to model and simulate complex construction projects, Halpin and Riggs (1992) and Lee et al. (2006, p. 222). The basic idea of CYCLONE revolves around the movement of these resources to participate in the execution of processes, Dozzi and AbouRizk (1993, p. 12), and is based on a GERT network, Pritsker (1966). CYCLONE had been developed for large mainframe computers in FORTRAN.

During the modeling process, the user has to define activities, relations between these activities, their duration, and resource requirements. Resources can either be active or idle and move from one activity to another. CYCLONE features include the ability to

simulate discrete systems with both deterministic or stochastic variables and an easy to read graphical representation of the network. The latter is made of only a few element types: *Squares* represent active tasks, which are constraint by resource requirements if the square features a triangle in its top-left corner. *Circles* define idle states or delays in the workflow. An empty circle contains functions, e. g., to consolidate or split resources. Circles with a little flag attached to them are counters. These are used to analyze the system. Finally, *Arrows* define the flow of entities. They connect the above-described elements. An example of a standard model for stripping works is shown in Figure 2.9. With the rise of the personal computer in the 90s *Microcomputer Cyclic Operation Network* (MicroCYCLONE), a port of CYCLONE to microcomputers, was presented, Halpin (1990). For an introduction to

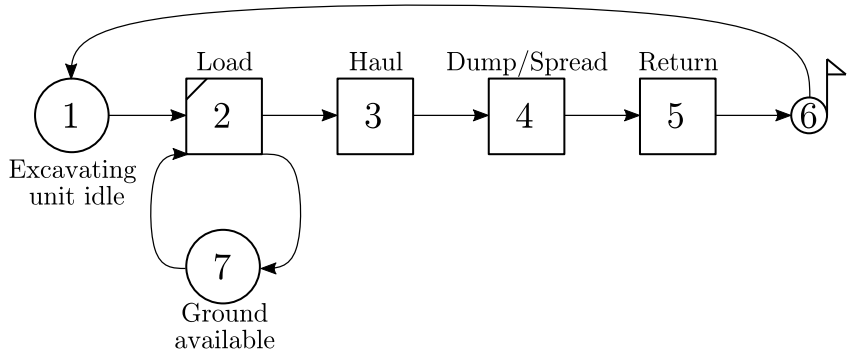


Figure 2.9: Standard Stripping CYCLONE model. (1) When the excavating unit is idle, the simulation proceeds to the next step, and (2) the excavation unit is loaded. The ground is then (3) hauled, (4) dumped, and spread. Finally, (5) the unit returns to the area of excavation, and the loop starts anew while a counter (6) keeps track of the number of executed loops. This process is repeated as long as there is both (7) ground and (1) an idle excavation unit available. Figure adapted from Halpin (1983, Figure 9).

the look & feel of CYCLONE and MicroCYCLONE, a short introduction with screenshots is provided on the website of Purdue University, Purdue (2020).

Even though CYCLONE is commonly associated with the simulation of construction projects of that time, other independent programs were developed in parallel. Examples are SCRAPESIM. Clemmens and Willenbrock (1978), an SPS for the simulation of scraper earthworks operations developed with the FORTRAN-based PASS simulation software; SIREN, Kavanagh (1985), a computer model of repetitive construction tasks implemented for the IBM-PC, IBM (2020), and featuring MCS; and an approach for the development of an SPS able to perform stochastic DES presented by Pilcher and Flood (1984). The same authors present COSMOS, a simulation software based on that approach. Four years later, a port of COSMOS to run on mainframe computers: *Interactive Construction Simulation* (ICONS), is presented at the University of Manchester, Pilcher and Flood (1984) and Alkoc and Erbatur (1998, p. 160).

CYCLONE seems to age well. Albeit presented over 40 years ago, it is still used by both practitioners and scientists to model and simulate construction projects. A selection of

publications on CYCLONE models includes: Touran and Asai (1987), who use CYCLONE to predict the tunnel advance rate of a tunnel with a length of several kilometers. Weigl (1993) presents a simulation model implemented in CYCLONE to forecast the performance of tunneling operations. The author builds his model with a strong focus on process durations based on data collected during two tunneling projects. However, unplanned standstills and disturbances are neglected; as Rahm (2016, p. 25) points out. Nido et al. (1999) analyze the impact of variations in soil compositions on the productivity of the operation and the utilization of labor resources in microtunneling. This is done with process simulation using the CYCLONE framework on the example of the Holes Creek Tunnel Project in Ohia, USA. Liu et al. (2010) describe a CYCLONE model for TBM construction process simulation and performance optimization. The model is somewhat abstract and couples TBM processes with those required for train bound material handling, in particular the transport of material from TBM to the dumping site. Furthermore, the model can calculate the optimum train to muck cart combination. The tool is evaluated on a 20 km water diversion tunnel project in Uygur Autonomous Region, China. The authors improve their model in Liu et al. (2015) by including geologic uncertainties and associated risks. These are adapted by the CYCLONE model, which is compared against the performance measures from a 13 km section of a TBM tunnel project. Zahran and Nassar (2013) use CYCLONE to model a pipeline project and Zhang et al. (2017) present a method to improve construction scheme planning in earth dam construction under consideration of stochastic rainfall impact.

Despite its ongoing popularity among some scientists, CYCLONE is not without critics. Upon these are Dori (2016b, p. 19) and Martínez (1996, p. 13), who agree that its shortcomings or deficiencies are:

- The framework is imposing limits, which do not allow the user to model the process at a level of detail that allows for sound decision-making,
- the inability of the model to recognize the difference between similar resources, i. e., similar resources cannot have different properties,
- the state of the simulated process cannot be evaluated during runtime and
- dynamic changes to the model cannot be performed during runtime.

Furthermore, Shi and AbouRizk (1997, pp. 1124 sq.) point out that the framework does not cater to practitioners, which usually got their knowledge through field training rather than academic research. Consequently, the application of methodologies that require extensive knowledge on simulation requires extensive learning of the tool, which renders GPS frameworks, such as CYCLONE, uneconomic.

The named deficiencies have been recognized by other scientists and led to the development of several CYCLONE inspired GPS frameworks, support-tools, and add-ons. These are described in the following section.

2.3.2 GPS Frameworks Inspired by CYCLONE

At least four different versions of CYCLONE and several programs inspired by the success of the CYCLONE software have been implemented in the past. The original CYCLONE itself had been implemented to run on large mainframe computers, Halpin and Woodhead (1976).

With the rise of the minicomputer during the late 70s, Kalk (1980) developed INSIGHT, a CYCLONE version converted to run on the PDP-11, the minicomputer for which the first UNIX operating system and the C programming language had been written, Ritchie (2020). INSIGHT featured extended modeling capabilities through time-lapse photography analysis and interactive computer graphics.

Six years later Chang (1986) presents *Resource based Queuing Network Simulation System* (RESQUE), a program which permits CYCLONE to differentiate between multiple resource types involved in the same task. RESQUE input is created via text commands. The text-input keeps the graphical CYCLONE model output simple but increases the complexity of the modeling process itself. Hence, it is more error-prone and increases overall modeling time; see also Chang and Hoque (1989, pp. 351 sq.).

The implementation of UM-CYCLONE, is a port of CYCLONE to run on processors with a modern x86 architecture, Intel (1990). The program allows for machine-user interaction via menu-driven GUI – a feature supposed to increase ease of use and the popularity of the simulation software, Ioannou (1989).

MicroCYCLONE is a port of CYCLONE to the Tandy TRS-80 and Apple II plus microcomputers, Reed (2020) and Weyhrich (2020), implemented by Lluch and Halpin (1982); see also Halpin (1990). A variety of sample simulations for MicroCYCLONE is provided on the website of Purdue University (Purdue, 2020). These include the simulation of the building of temporary cofferdams for construction, asphalt paving, pipeline process modeling, or segmented construction process models. Furthermore, Alkoc and Erbatur (1998) use MicroCYCLONE to compare two concreting methods. (1) with crane and bucket and (2) with a concrete pump. Their model can analyze the influence of limited resources, work crews' interaction due to space limitations, and equipment interaction due to the need to share tools between different processes.

Starting with the 90s, the OO design paradigm became more popular among scientists developing simulation software for construction problems. Zhang et al. (2005) name the advantages of the OO paradigm, which allows easier maintenance, modifying, and reusability of the simulation system. These advantages result from OO concepts, such as inheritance, encapsulation, polymorphism, dynamic binding, and parameterized typing; see also Dörn (2017), Eilebrecht and Gernot (2013), Goll (2014), and Joines and Roberts (1998).

COOPS, developed by Liu (1991), is an OO system that supplements CYCLONE. It provides a GUI to build and edit models, treats all resources as individual objects with own statistical characteristics, and implements a calendar to control the availability of resources; see also Liu (1995) and Liu and Ioannou (1992).

CIPROS, another OO approach, is presented by Odeh (1992). The program allows hierarchical modeling of simulation objects to test alternative construction plans. It provides the possibility to correlate design drawings with construction plans to analyze the latter regarding their quality and feasibility; see also Tommelein et al. (1994).

Oloufa (1993) presents MODSIM, an OO language with a focus on linking construction elements to simulation objects to simplify the changeability of the model in case of unforeseen events. Oloufa et al. (1998) describe a simulation library which enables users without knowledge of "any simulation programming language, syntax, or icon [sic]" (Oloufa et al., 1998, p. 325) to execute the simulation, draw conclusions on the analyzed project and quickly alter the model if the need may arise; see also Oloufa (1994).

In 1996 the OO simulation language STROBOSCOPE (State and Resource Based Simulation of Construction Processes) was presented by Martínez (1996). Like CYCLONE the inner logic of STROBOSCOPE follows an AOA; however, it integrates technologies, which were not available in the 70s when CYCLONE came to life. For instance, STROBOSCOPE's functionality can be extended through high-level languages such as C, C++, FORTRAN, or PASCAL. Users can integrate external libraries, which allows for solving a wide range of problems, influencing the simulation during runtime, and using resources that are treated differently according to their properties. In particular the integration of external libraries enable developers to create SPS frameworks based on STROBOSCOPE, Hajjar and AbouRizk (1999). The same author developed EZStrobe, which is wholly based on graphic elements, Martínez (2001). EZStrobe is meant to provide an easy entry into simulation without the need to learn a programming language, Ioannou (2020). Hassan and Gruber (2008) describe how they use an EZStrobe simulation of paving operations of an interstate highway in the USA to determine the optimum machine configuration. Kamat and Martinez (2003) present a case study in which they use the *Dynamic Construction Visualizer* (DCV) – a tool for the 3D visualization of STROBOSCOPE simulations – for an earthmoving project. The authors show that 3D visualization is a great help to understand the consequences of various construction alternatives.

SLAM II is a prototype of a resource-based model simulator described by Pritsker (1984). Operating processes of resources are defined as atomic models and stored in a model library. By linking the atomic models together, users can define simulation models without being proficient with simulation theory and simulation. Shi and AbouRizk (1997) demonstrate the tool's abilities on the example of a surface mining reclamation project. Gonzalez-Quevedo et al. (1993) compare SLAM II with MicroCYCLONE by modeling a horizontal earth-boring project. They conclude that the former is more complicated to learn and apply; however, it is also more versatile than the latter. A software product using the SLAM II modeling language is *Simulation-based project Control* (SimCon) by Chehayeb and AbouRizk (1998). SimCon has been developed to create schedules for construction projects. It builds on the existing CYCLONE fundamentals implemented in SLAM II. The implementation of the visualization follows the OO paradigm. Furthermore, the software features a relational database using MS ACCESS, Reading (2016). The authors see the main advantage of SimCon in using logical links between simulation elements. These logical links are supposed to create a more accurate representation of relationships between activities and simplify the construction sequencing methodology.

Project Integrated Cyclic Analysis of Serial System Operations (PICASSO) is presented by Senior and Halpin (1998). The software tool combines classic CPM with a CYCLONE model. PICASSO can calculate the float time of processes while taking resource interaction and resource sharing patterns into account. The authors illustrate the functionality of their tool on the example of a simplified housing project.

Owing to the low distribution of construction simulation frameworks outside the scientific community, Hajjar and AbouRizk (1999) analyzed three SPSs which were popular in the construction industry at that time: AP2_Earth (Hajjar and AbouRizk, 1996), CRUISER (Hajjar and AbouRizk, 1998) and *Construction Site Dewatering* (CSD) (Hajjar et al., 1998). They identify several requirements a simulation framework has to fulfill to get adapted by the construction industry. SIMPHONY, a discrete event simulator, was developed from the findings of that study. Its main features are described by AbouRizk and Mohamed (2002) and Hajjar and AbouRizk (1999) as:

- A GUI to create and edit the simulation model,
- Script-based modeling and access to higher-level programming languages to allow advanced users to bypass the GUI,
- General purpose constructs as well as specialized templates for specific constructs,
- Generation of customized output results as graphs and tables,
- Automated generation of project planning data in a standard format,
- A model library, in which the user can store commonly used models, and
- Modular and hierarchical modeling for complex or large projects.

SIMPHONY has been designed for users with limited simulation experience and developers who wish to create SPSs for those users. To this day, SYMPHONY is continuously improved. Its latest version, Simphony.NET, supports C♯ and builds on the Microsoft .NET framework, AbouRizk et al. (2014). The core libraries of Simphony.NET allow for external access, i. e., to be integrated with the HLA provided by COSYE (description see below), AbouRizk and Hague (2009). Mohamed and AbouRizk (2005) use a tunnel model prototyped with SYMPHONY to demonstrate an agent-based approach for the minimization of TBM idle time. Ebrahimy et al. (2011a) present a supply chain simulator based on SIMPHONY, which they use to model the supply chain of pre-cast concrete liner segments of the Glencoe Tunnel project in Calgary, Canada.

Shi (1999) introduces the *Activity-Based Construction* (ABC) modeling and simulation method, which uses one single element, the activity, to model general purpose processes. The simulation, ABC-Sim, follows a three-step approach. Namely, (1) select an activity, (2) advance simulation, and (3) release simulation entities. The reduction of modeling elements to a single entity is supposed to make the modeling process as easy as the CPM. Thang et al. (2002) extend that approach by modeling the activities as objects and adding an animation function to their software so that users may be supported in understanding the model's workings. The view that ABC and the three-step approach eases modeling is rejected by Zhang et al. (2005, p. 315), who point out that the modeling process is indeed tricky and time-consuming, especially if many activities are modeled in the system.

Furthermore, Martinez and Ioannou (1999, pp. 266 sqq.) remark that the graphical network, which is required for the modeling, is cumbersome to maintain for large networks.

Lu (2003) proposes *Simplified Discrete Event Simulation Approach* (SDESA) with the intent to make simulating construction systems as easy as the CPM. SDESA is a general purpose method which distinguishes between disposable and reusable resources and reduces the queuing structure into a single dynamic queue of flow entities. This way of modeling reduces the simulation to the interaction of two dynamic queues: One for the flow entities and one for the resource entities; see also Dori (2016b, p. 76).

Halpin et al. (2003) present the WebCYCLONE prototype, which offers the functionality of CYCLONE as a web service. The application is meant to support easier information exchange and peer collaboration when working with the CYCLONE simulation framework. Tam and Leung (2010) use WebCYCLONE to model and simulate the laying of water mains in Hong Kong. Najafi and Tiong (2015) simulate the installation of horizontal precast concrete parts. Kim et al. (2018a) present a method combining construction-process simulation with WebCYCLONE and vision-based context reasoning for productivity analysis of an earthmoving project. The scientists use convoluted networks technology – a form of AI described in Fischer (2016) and Lu et al. (2017) – to extract information on the state of construction equipment from the construction site's *Closed Circuit Television* (CCTV).

Zhang et al. (2008) propose a simulation-based methodology to handle time constraints such as cyclical breaks, preemption, and overtime. This is done by computing the start and end times of time-constrained activities and incorporating them into an activity-based simulation through Activity Scanning, section 2.2.6. Albeit a real case study has not been performed, the theoretical examples show promising results. First, they are sound, and second, building models for repetitive construction projects is simpler and more comfortable than it is with CPM or CYCLONE.

AbouRizk and Hague (2009) present COSYE, which supports the distribution of simulations using separate modules inter-linked via HLA. HLA is an IEEE standard (IEEE 1516, 2010) which was originally developed by the Defense Modeling and Simulation Office of the US Department of Defense to combine many small simulation models for the creation of one large. This combination of small models allows for the simulation of large scale industrial construction projects by dividing individual parts of that simulation into so-called *federations*. For instance, the construction federate may be using DES while the resource allocation federate makes use of AI. By dividing large-scale simulations into federations, the development process improves. This improvement is because the code is automatically structured, and inter-dependencies are reduced, resulting in higher transparency. Hence, the code contains fewer bugs, and it is easier to be maintained. AbouRizk et al. (2010) present a COSYE simulation for industrial plant construction, Xie et al. (2011) simulate a tunnel in Edmonton, Canada using real time data, which is manually collected via web interface of mobile phones. Hollermann et al. (2012) present the scheduling of a bridge construction project; see also Dori (2016b, p. 79).

Furthermore, several add-ons for CYCLONE have been implemented to improve its usability. A selection of these includes: CADSIM, a program which enables AUTOCAD to generate a visual representation of a CYCLONE network from CYCLONE code, Riggs

(1987). Paulson et al. (1983) present a method to extract input data for CYCLONE or INSIGHT from video-taped construction operations to be used in simulation environments; see also Paulson et al. (1987). Dabbas and Halpin (1982) present PROMAX, a tool to connect CYCLONE with a *Project Management* (PM) software favored by the developers. AbouRizk et al. (1990) present a feature for MicroCYCLONE to reduce the number of simulation runs without sacrificing the level of confidence of the output. Touran (1992) presents a tool intended to be used by construction managers without simulation experience. The tool produces simulation code for SLAM and CYCLONE to help users choose the fleet size for earthmoving works. Lutz et al. (1994) implement the Boeing learning curve, Tanner (1985), to CYCLONE to include learning effect for work crews in the simulation. Huang and Halpin (1994) present the DISCO interface, which visualizes the system dynamics of a MicroCYCLONE simulation. Huang et al. (1994) model a cable-stayed bridge using DISICO, a pre- and post processor for MicroCYCLONE. Sawhney and AbouRizk (1996) present a tool, which implements their HSM method. The program uses OO concepts and visual modeling to support the modeling process with CYCLONE.

2.4 Special Purpose Simulation Frameworks in Tunneling and Injection Grouting

SPS frameworks are simulation programs, which focus on a narrow domain, such as paving, pipeline installation, or tunneling. The advantages of tailoring a construction simulation software to a specific industry branch are, in particular, that such software products often gain wider acceptance with practitioners, require less experience in simulation theory, is more straightforward and thus more effectively to handle than GPS software, and allows the creation of standardized reports which can directly be used in the estimation and bidding process, Hajjar and AbouRizk (1996).

Within the scope of the literature research for this thesis, no relevant literature on SPS to support the project management of injection grouting projects had been found. The focus of past and current research seems to be on the analysis of technical questions, in particular in the field of material science, e. g., Holmboe et al. (2011), Padovnik and Bokan-Bosiljkov (2020), and Restuccia et al. (2017). Simulations are conducted foremost to find optimal technical parameters, such as the optimal injection pressure, Abdollahisharif and Bakhtavar (2018); the injection capacity of the injected media, Jorne et al. (2014); or the evaluation of the injection depth, Ganjalipour and Esmailzadeh (2019). For this reason, the literature research area is extended to SPS for process simulation in tunneling – the industry branch in which the research of this thesis takes place.

The most significant developments of SPSs for process simulation for the tunneling industry are being summarized in the following. Three research groups are most active in this field. The first develop the software package *Single Purpose Simulation for Utility Tunneling* (SUT), section 2.4.1. Professor AbouRizk leads the research group at the University of Alberta in the city of Edmond, Canada. The second group develops DAT under supervision of Professor Einstein at the Massachusetts Institute of Technology in Cambridge, USA,

section 2.4.2. The third group has not yet given their results a crisp acronym or name. It is part of the *Sonderforschungsbereich* (SFB) 837, see section 2.4.3, and led by Professor König and Professor Thewes at the Ruhr-University in Bochum, Germany. Outside of these three research groups, several scientists have developed individual SPS solutions. A selection of the most interesting developments is summarizes in section 2.4.4.

2.4.1 SPS for Utility Tunneling

Since Hajjar and AbouRizk (1999) introduced the GPS framework SYMPHONY, section 2.3.2, for the modeling of SPS templates for tunneling, the research group from Canada has refined their software package in various ways. Apart from providing accurate simulation results, the group's goal is to make SPS accessible for practitioners. In cooperation with the city of Edmonton, Canada, the group had several opportunities to put their software design to the test with both practitioners and scientists alike.

The SUT was developed by Ruwanpura (2001) and first present in AbouRizk et al. (1999). The tool supports decision-making in mechanized tunnel construction. It uses historical data to generate a potential construction schedule. Since its first appearance in 1999, the tool has been continuously improved. Ruwanpura and AbouRizk (2001) uses Markov chains, section 4.3, with two states to build a soil prediction model, an idea already proposed by Krumbein and Dacey (1969). The model uses historic borehole data to anticipate changes between two types of soil layers. Ruwanpura et al. (2004) improve the method and allow more than two types of soil for the prediction model. The tool estimates the project cost and project duration of the construction of one and two-way tunnels. Features of the program include the simulation of rail-bound material transport. A first real case study is described by Ruwanpura et al. (2001), who use the tool to support the planning of the Canadian South Edmonton Sanitary Sewer project. After the tunnel structure specification had changed during the project, the tool was used to evaluate alternatives.

Fernando et al. (2003) describe how SUT had been used in three different Canadian tunneling projects, namely

- the South Edmonton Sanitary Sewer (SESS) Tunnel (length: 2.5 km),
- the Calgary Trail Interchange Tunnel (CTIT) (length: 510 m), and
- the North Edmonton Sanitary Trunk (NEST) Tunnel (length: 1.6 km)

to estimate project time and cost and improve utilization of resources, i. e., the number of trains and carts used for the transport of soil from the TBM to the surface. Chung et al. (2006) uses Bayesian techniques based on the real project progress of the NEST project to improve the simulation's input data. Al-Battaineh et al. (2006) use the tool for the planning of the Glencoe Storm Sewer upgrade, a Canadian tunnel in Calgary, Alberta with a length of 935 m. Zhou et al. (2008) add an SPS template to model shaft constructions and apply it to a section of the NEST project.

The introduction of the COSYE framework, AbouRizk and Hague (2009), with its HLA allows the SUT to integrate the developed SYMPHONY templates with other features, such as 3D CAD modeling and progress visualization, Zhang et al. (2010). Ebrahimy et al.

(2011a,b) add the *SPS Supply Chain Simulator*, an SPS to analyze logistic supply chains and different information-sharing strategies for the construction site. The authors test the tool on the Glencoe Storm Sewer upgrade project in Canada. Moghani et al. (2011) add an extension to consider a sequential excavating method using either shotcrete or rib and lagging as preliminary supporting systems in the analysis and apply it on a project building two parallel tunnels with a total length of 764 m in the city of Edmonton, Canada. Xie et al. (2011) use COSYE to integrate real-time progress monitoring to provide real-time input data for the simulation. The feature is tested in a case study involving the NEST tunnel project. The application of the tool is described in detail by Al-Bataineh et al. (2013). A stochastic weather simulator is added to the tool by Shahin et al. (2013), to consider the harsh Canadian weather conditions which have hindered construction efforts in the past, e. g., by freezing the ground at $-40°C$ or disturbing lifting cycles of cranes at high wind speeds. RazaviAlvi and AbouRizk (2014) introduce a feature to support decision-making in site layout planning, and RazaviAlvi and AbouRizk (2015) add a genetic optimization algorithm to optimize the material yard laydown. Werner and AbouRizk (2015) add a feature to include delays of TBM progress from equipment breakdown and other unexpected conditions to the simulation. Zhang et al. (2015) combine the approach of Chung et al. (2006) and Xie et al. (2011) to continuously update the geological stochastic model of the tunnel during construction in order to eliminate the uncertainty of the original model and improve the simulation results. This approach is further improved by Werner et al. (2018) and Wu et al. (2020).

2.4.2 Decision Aids for Tunneling (DAT)

DAT is an SPS for tunneling operation, which is developed at Massachusetts Institute of Technology in Cambridge by a team of scientists lead by Prof. Einstein. In contrast to many other SPSs it does not rely on one of the popular GPS frameworks, such as CYCLONE or SYMPHONY. It consists of two major modules: the geology module and the construction module. Both consider uncertainties in the geological profile and assessment of associated risks. This issue has already been highlighted by the authors well before DAT was first named, e. g., by Moavenzadeh et al. (1974) who present TCM which includes empirical submodels for geological conditions and the tunnel structure; see also Bennett (1981, pp. 23 sqq.). In DAT, the *geological model* divides the tunnel in geological zones of uncertain length. Each zone is characterized by its geological parameters, including an estimation for their uncertainties described as a Markov chain process. The *construction model* simulates the construction process for each of the ground profiles. The simulation automatically adapts the tunneling method, which defines the cross-section to the geological conditions. The different tunneling methods are associated with cost and advance-rate distributions. The simulation is calculating a cost-duration pair for each of the possible geological profiles. Once finished, the uncertainty of the construction process is represented by these pairs and plotted in a cost-time scatter diagram, Haas and Einstein (2002). The DAT can furthermore include other influencing factors in the simulation, such as delays caused by method changes, learning curves, delays caused by operational aspects such as rail placements or different shift arrangements such as working and maintenance shifts,

Einstein (2001) and Min et al. (2003). An introduction of DAT is given by Einstein (2004), Einstein et al. (1999), and Haas and Einstein (2002).

Until today DAT is under constant development. In addition to cost and time estimation, the analysis of project resources is a core aspect of DAT, Einstein (2004) and Einstein et al. (1999). Sinfield and Einstein (1996) analyze the effects on cost and time of potential changes in micro tunneling and tunneling with full-face TBM.

Haas and Einstein (2002) introduce the option to update the simulation with real-time data using Bayesian statistics. Min et al. (2008) use this feature to analyze the 1.9 km Sucheon tunnel in Korea both before and during the construction phase. DAT was also used during the pre-exploration phase of the tunnel, described in Karam et al. (2007).

Ritter et al. (2013) adapt DAT to assess uncertainties affecting tunnel excavation material handling and perform a case study on a section of the Brenner Base Tunnel. The individual steps in the material handling process are modeled, starting with excavation and ending with the reuse or final depositing of the material. Shao et al. (2015) extend DAT to simulating underground cavern construction with complex networks, a particular case with high uncertainties in terms of project cost and time.

Min and Einstein (2016) present an approach to optimize resource scheduling with regard to time and cost, based on optimal construction cycles sets. The authors prioritize activities to reduce the number of possible resource allocation plans per cycle to a number that is small enough to be computed and compared in total. The model is tested on data of a finished project of a Korean two-tubed tunnel described in Min et al. (2008).

Moret and Einstein (2016) include three sources of uncertainty in DAT: variability in the construction process, correlations between the costs of repeated activities, and disruptive events. The authors use historical data and expert estimations to model the cost and duration uncertainty and validate their model on a 45.6 km section of the Portuguese high-speed rail network project, including five tunnels, six viaducts, and several cuts and embankments.

Paltrinieri et al. (2016) perform a simulations of TBM tunneling in highly fractured and faulted rocks. Naghadehi et al. (2016) calculate the risk construction time and cost of a part of the Tehran Metro Line 7, an urban tunneling project. Kim et al. (2018b) use DAT on a hard rock tunnel with multiple hard rock mass properties for scheduling and cost estimation.

Costa et al. (2018) combine DAT with *Opt*, an optimization tool for linear infrastructure projects, which uses maps derived from *Geographic Information System* (GIS) as input. With this, the authors perform a probabilistic 3D alignment optimization of the underground transport infrastructure based on GIS subsurface characterization.

Maruvanchery et al. (2020) use DAT not for tunneling but a large cavern construction project in Singapore as an early construction cost and time predicting tool.

2.4.3 Interaction Modeling in Mechanized Tunneling (SFB 837)

A research group at the Ruhr-University Bochum has recently published several papers on construction management simulation in mechanized tunneling. The research was performed in connection to the *Collaborative Research Center* (CRC) / SFB 837 on "Interaction modeling in mechanized tunneling", Sadri et al. (2013) and SFB (2015, 2018, 2019), in particular its subproject C3, Conrads et al. (2018b) and Jodehl et al. (2019).

Rahm et al. (2012) present a multi-method simulation model to investigate the advancement rate of tunnel boring machines. The model considers process-related disturbances and models parts of the TBM supply chain. The authors combine different simulation methods. Processes are simulated with DES and the propagation of the continuous flow of excavated soil is based on an SD model. The authors use the commercial software package AnyLogic, AnyLogic (2020), and describe their models with *System and Modeling Language* (SysML), SysML (2020). The SysML is a general-purpose architecture modeling language for systems engineering applications. It uses graphical objects with nine different types of diagrams to describe the system's requirements, its physical constraints, structure, and behavior, Dori (2016a), Holt and Perry (2008), and Nyamsi (2020). The research members of SFB 837 frequently use both SysML and AnyLogic to describe and model their projects.

Rahm et al. (2013) present an approach based on Fuzzy Logic to take the performance-related influence of wear of cutting tools on the advance rate of the TBM into account. König et al. (2014) and Rahm et al. (2015) add three classes of disturbances to the model: production disturbances, logistic problems, and reproductive effects. The latter are cascading effects, which might bring the entire tunnel drive to a standstill. Scheffer et al. (2014a,b) extend the supply-chain model to above-ground construction site logistics, which are later improved by Scheffer and Duhme (2018) who add a dynamic traffic simulation model. Scheffer et al. (2016) analyze the influence of construction site lay-out on project progress and include cascading effects of different types of disturbances.

Mattern et al. (2016) analyze different approaches of maintenance of TBMs in soft ground on the project duration. The authors have their focus on the performance-limiting factors of cutting tool condition. Rahm et al. (2016) continue with this approach to include the failure of other production-relevant parts, in particular wear out and abrasion of cutting tools mounted to the TBM and resulting maintenance interrupt; see also Conrads et al. (2016), Rahm (2016), Conrads et al. (2017, 2018a), and Conrads (2020).

Dang et al. (2013) describe an SPS with which they perform a post-project simulation of a microtunneling project in Recklinghausen, Germany. Their model includes differing soil compositions, varying process durations, and the allocation of resources. A GUI supports the user in the correct application of the tool. Dang et al. (2018) present *Microtunnelling: Statistics, Analysis and Simulation* (MiSAS), a software module designed for the simulation of microtunneling project. Following the research group's standard approach, the module is being described in SysML and then modeled using the AnyLogic software package. It combines DES and SD simulation techniques to analyze construction processes and identify the influence of disturbances and different soil compositions on productivity. The software performs a MCS to identify the best construction alternative. Furthermore, MiSAS features

a GUI to lower the entry barrier for beginners in simulation and has been successfully tested on a 145 m long microtunnel project in Recklinghausen, Germany.

2.4.4 Other Simulation Based Tools and Techniques

So far, results of selected research groups have been discussed due to their dominant presence in the research community. This section examines a selection of other publications that are ideally related to the ideas presented in this thesis.

An example of an early SPS is VEHSIM by Caterpillar Inc. – a manufacturer of machinery for the construction industry, Caterpillar (2020). VEHSIM had been developed in the 60s and is a static, stochastic simulation algorithm that computes haul and return times of vehicles. It is neither time-dependent nor are interactions with resources part of the simulation, Ashton (1989, pp. 31 sq.) and Zub (1981).

Structured Environment for Process Simulation (STEPS) was developed for the US Navy to support earthwork operations. The program features include resources of different size for the same task, the calculation of task durations during the simulation, the rule-based release of resources from queues, and statistics on each type and size of resources, McCahill and Bernold (1993).

Wales and AbouRizk (1996) use the commercial software SLAMSYSTEM to combine a CPM styled construction plan with DES and a weather condition simulation to generate stochastic weather conditions, which influence the process durations. By performing a MCS the authors can compute stochastically distributed project durations.

AP2_earth, CRUISE and CSD are three SPS which were popular in the late 90s. The experience made with these tools significantly influenced the design of STROBOSCOPE. AP2_Earth is a software tool to analyze earthmoving projects, Hajjar and AbouRizk (1996); CRUISER is an OO SPS for the simulation of crushed-stone aggregate production that features a GUI to create and experiment with the simulation model, Hajjar and AbouRizk (1998); and CSD is an SPS for construction site dewatering, Hajjar et al. (1998).

Likhitruangsilp and Ioannou (2003) present *ProbSched*, a model implemented with STROBOSCOPE. The authors seek to evaluate alternatives in the tunnel construction with respect to the expected performance. Simulation results provide probability distributions of tunnel advance rates and tunneling unit costs for the possible alternatives. Uncertainties enter the calculation through a probabilistic geological prediction model, a probabilistic tunnel cost estimation model, and a risk-sensitive dynamic decision model. However, Rahm (2016, p. 25) points out that some uncertainties come from subjective assessment rather than stochastic analysis. The authors continue to improve their model to take limited resources of the alternatives into account, Ioannou and Likhitruangsilp (2005). The authors evaluate their results on the Hanging Lake Project in Colorado, USA; see also Likhitruangsilp and Ioannou (2004, 2005).

Marzok et al. (2008) present *Tunnel_Sim*, a tool to assists contractors in estimating the time and cost required for the construction of tunnel projects. The tool analyzes different construction techniques to capture the construction of open/closed, rectangular,

and circular cross-section tunnels. The tunnels are divided into working zones, and the total duration and cost for each zone are calculated. Finally, the tool selects the best construction technique from a set of alternatives based on a decision-making method that utilizes fuzzy numbers. Later the authors add the option to plan microtunnels as well, Abdallah and Marzouk (2013) and Marzouk et al. (2010).

An approach to making computer-aided process simulation usable for construction site managers in the field is presented by Kim and Gibson (2003). *Knowledge-Embedded Modularized Simulation System* (KMOS) is an interactive simulation modeling approach that is supposed to provide step-by-step guidance for project managers to build their simulation models on the computer.

Szczesny and König (2015) proposes a method for reactive construction scheduling of house construction to ease the integration of real-time logistic data to the model. The method follows a four-step approach during which (1) fuzzy set theory and α-cut method is used to analyze uncertain data of logistics processes, (2) the data is integrated into a DES to perform a sensitivity analysis to analyze impacts on the construction schedule, (3) an actual schedule is generated, and (4) simulation-based optimization based on the NSGA-II evolutionary algorithm is used, if a significant deviation from pre-defined constraints is identified.

2.5 Research Gaps

The literature reviewed does not use complete, real-time, second-by-second data measurements to predict total injection volume, total project cost, and total project time for the construction project or provide in-depth analysis of the construction site's process data. Simulation models do not combine the mutual collaboration and location wise mutual obstruction of construction machinery with statistically distributed process durations, the possibility of machine failure, and the influence of statistically distributed geological conditions. Furthermore, such models do not use GA in combination with DES and MCS to calculate Pareto optimal solutions in the dimensions of project time and project cost. In particular, this has never been done on the example of an injection project.

The revealed research gaps are used in the following chapter to develop the objectives of this thesis.

3 Motivation and Objectives

'Would you tell me, please, which way I ought to go from here?'
'That depends a good deal on where you want to get to', said the Cat.

Caroll (1865)

This thesis's objectives are derived from the research gaps revealed by the previous chapter's literature review. The goal is to select, implement, test, and evaluate numerical methods to support large construction projects' construction management. These methods shall be bundled in a non-commercial SPS framework named GBPlan with the primary purpose to preserve the applied methods to be used in later research projects. To improve the source code's reusability, GBPlan shall be implemented in MATLAB, the programming language most frequently used by the research institute, which funded this thesis's preparation. The implementation of methods shall be performed in five steps, and each of these steps shall define an objective of this thesis. Since digitized data from an injection project in tunneling is available, the objectives will be worked using this project as an example. Its large size characterizes that injection project. Several injection units inject thousands of boreholes in a 3.2 km, twin-tubed railway tunnel over several years. During this time, the complete digital documentation of all injection units' processes is recorded in real-time.

The five objectives of this thesis are:

Objective 1 Analyze and visualize the digitized data set to improve the understanding of the data, and the construction site's workings.

Objective 2 Forecast of the grout volume, which is injected in the tunnel.

Objective 3 Forecast of the work progress of an injection production system's, which consists of injection units of the same type.

Objective 4 Forecast of the construction time and cost of an injection production system, which includes two types of injection units. The methods shall take the mutual collaboration of injection units into account. Furthermore, the forecasting methods shall be used as goal functions to find Pareto optimal solutions of the number of machines employed on the construction site.

Objective 5 Improving the model developed under Objective 4 to include the discrete location of injection units in the tunnel. Furthermore, the mutual obstruction of injection units in the narrow tunnel environment shall be taken into account, and a third type of injection unit shall be included in the simulation.

4 Methodological Framework

Deep in the human unconscious is a pervasive need for a logical universe that makes sense. But the real universe is always one step beyond logic.

Herbert (1990)

This chapter describes the methodical framework applied in this thesis. The first part, section 4.1, focuses on providing concise definitions for general statistical concepts, such as distribution functions and measures of statistical error. The second part, section 4.2, deals with the MCM – a statistical method to sample from unknown distribution functions. In the third part, section 4.3, *Markov Chain*s (MCs), which can be used to model stochastic processes, are being introduced. The fourth part, section 4.4, deals with Graph Theory. Graph Theory is used to calculate the shortest paths in the tunnel and understand the implementation of other methods, such as MCs, within the GBPlan framework. The sixth section, section 4.5, introduces numerical optimization, i. e., a method to find the parameters of a function that minimize its results. The section ultimately leads to mixed-integer MOGA, which is an optimization method implemented in GBPlan. Finally, section 4.6, gives an introduction to Deep Learning, a method that belongs to the field of Artificial Intelligence. The section's focus is on FNNs, which are later used to forecast grout volumes in the tunnel project under investigation.

All sub-section provide references to relevant literature. Also, references to publications using the described methods in a construction management context are provided. In that, this section complements the literature review presented in chapter 2.

4.1 Statistics

This section's focus is to provide definitions, which are used throughout the further discussion in this thesis.

4.1.1 Probability Density Function

The *Probability Density Function* (PDF) gives the probability of x for every unit of x. Two types of PDF are distinguished: continuous and discrete PDF.

If the PDF $f(x)$ is *continuous* the function gives the probability that x is within an interval dx, EQ 4.1.

$$f(x)dx = \text{Prob}\{x \leq x_i \leq x + dx\} \tag{4.1}$$

The continuous PDF has three properties: (1) f(x) is defined on an interval $[a, b]$ with $a > b$, (2) $\forall x \in [a, b], f(x) \geq 0$ and (3) the PDF is normalized such that $\int_{-\infty}^{+\infty} f(x)dx = 1$, Dunn and Shultis (2012, pp. 22 sq.).

In the *discrete* case $f_i \equiv f(x_i)$ is the probability of event i with $i = [1, n]$ and $i \in \mathbb{Z}^+_{\neq 0}$. The discrete PDF has two properties: (1) $f_i \leq 0$ and (2) $\sum_{i=1}^{n} f_i = 1$, Dunn and Shultis (2012, p. 29).

4.1.2 Cumulative Distribution Function

The *Cumulative Distribution Function* (CDF) gives the probability $F(x)$ that a random variable r takes a value less or equal to x,

$$\text{Prob}\{r \leq x\} = F(x). \tag{4.2}$$

Like with the PDF, a distinction is made between continuous and discrete CDF.

In the *continuous* case the CDF is defined as an integral over the PDF such that

$$F(x) \equiv \int_a^x f(x)dx \tag{4.3}$$

and is a measure for the probability that a random sample of a stochastic variable x has a value between a and x_i, i. e.,

$$\text{Prob}\{x_1 \leq x \leq x_2\} = \int_{x_1}^{x_2} f(x)dx = F(x_2) - F(x_1). \tag{4.4}$$

The CDF has three important properties, which are (1) $F(a) = 0$, (2) $F(b) = 1$, and $F(x)$ is monotone increasing, Dunn and Shultis (2012, p. 23).

The *discrete* CDF is defined as EQ 4.5

$$F_i \equiv \sum_{j=1} f_j \tag{4.5}$$

and gives the probability that one of the first i events occurs. It has three important properties: (1) $F_1 = f_1$, (2) $F_n = 1$, and (3) $F_i \leq F_j$ with $i \leq j$, Dunn and Shultis (2012, p. 29).

The *Inverse Cumulative Distribution Function* (ICDF) can be used to draw random samples from a CDF. This is possible because any CDF is monotone increasing and takes values between zero and one. Hence, if there exists an x_i for the PDF $f(x)$ of which the CDF is $F(x)$ and a y_i for the PDF $g(y)$ of which the CDF is $G(y)$ then

$$F(x_i) = G(y_i). \tag{4.6}$$

In order to sample from F one replaces G in EQ 4.6 with the unit rectangular CDF $U(\rho)$, EQ 4.7,

$$U(\rho) = \int_0^{\rho} d\rho' = \rho \tag{4.7}$$

such that EQ 4.8

$$F(x_i) = \rho_i \tag{4.8}$$

of which the ICDF F^{-1} is EQ 4.9.

$$x_i = F^{-1}(\rho_i). \tag{4.9}$$

In order to sample from F one simply has to sample ρ from the unified distribution $\mathcal{U}(0,1)$ and solve EQ 4.9, Figure 4.1. This method of sampling is very efficient but not

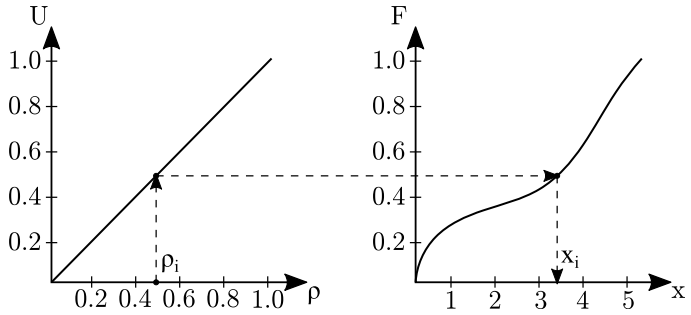


Figure 4.1: Sampling from the ICDF. First a random value $\rho_i \sim \mathcal{U}(0,1)$ is used to sample from the rectangular CDF U. Then that result is used to calculate x_i from the ICDF F^{-1}.

always feasible. The reason for this is that it implies that EQ 4.8 is solvable for x_i. If F is *continuous* this may not be feasible, Dunn and Shultis (2012, pp. 70 sqq.). In this case, a well known, however, very computationally intensive method is the Accept-Reject method – a trial and error approach first proposed by Neumann (1951); see also Dunn and Shultis (2012, pp. 74 sqq.) and Robert and Casella (2004, pp. 47 sqq.). If F is *discrete*, then the transformation from CDF to ICDF becomes a matter of correct counting, i. e., it is in general feasible. Sampling with the ICDF comes with the advantage that it is possible to sample from uncommon distributions. For instance, Allen (2011, 11, Fig 2.1) uses the method to sample from a temperature distribution which is either normally distributed around a high or a low temperature. For further reading on the application of ICDF for sampling refer to Forbes et al. (2011, pp. 7 sq.), Allen (2011, pp. sqq.), and Dunn and Shultis (2012, pp. 70 sqq.).

4.1.3 Statistical Measures

This section provides a firm description of statistical measures for populations, section 4.1.3.1, for samples, section 4.1.3.2, and for errors, section 4.1.3.3 used within the framework of GBPlan. For a comprehensive description refer to Dunn and Shultis (2012) and Pahl and Damrath (2000). If in need of a quick reference, refer to the online resource of Weisstein (2020e).

4.1.3.1 Population Measures

The *mean* μ and the *variance* σ^2 of a population can be calculated based on its PDF $f(x)$. It is also called the expected or average value and has a definition which varies depended on whether the distributions are continuous, EQ 4.10, or a discrete distribution, EQ 4.11.

$$\langle z \rangle \equiv \mu(x) \equiv \int_a^b x\, f(x)\, dx \qquad \text{(mean, continuous)} \tag{4.10}$$

$$\langle z \rangle \equiv \mu(x) \equiv \sum_{i=1}^{n} x_i\, f_i \qquad \text{(mean, discrete)} \tag{4.11}$$

The *variance* is a measure of how much the variable x scatters around the mean value. For continuous distributions it is defined as EQ 4.12 and for discrete distributions as EQ 4.13.

$$\begin{aligned} \sigma^2 \equiv \langle [x - \langle x \rangle]^2 \rangle \quad &= \int_a^b f(x) \cdot [x - \langle x \rangle]^2\, dx \qquad \text{(variance, continuous)} \\ &= \langle x^2 \rangle - \langle x \rangle^2 \end{aligned} \tag{4.12}$$

$$\sigma^2 \equiv \sum_{i=1}^{n} [x_i - \langle x \rangle]^2 f_i \qquad \text{(variance, discrete)} \tag{4.13}$$

The *standard deviation* $\sigma(z)$ is defined as the square root of the variance such that

$$\sigma(z) = \sqrt{\sigma^2(z)}. \tag{4.14}$$

The *median* $\tilde{x}$ of a continuous distribution $F(x_i)$ is calculated with its ICDF $G(y_i)$ as in EQ 4.15. In the case of a discrete distribution, the population's elements x_i are sorted according to their value. The median is the element in the middle of that sorted set, EQ 4.16, Weisstein (2020b,d).

$$G(\tilde{x}) = \frac{1}{2} \qquad \text{(median, continuous)} \tag{4.15}$$

$$\tilde{x} = \begin{cases} y_{(N+1)/2} & \text{, if } N \text{ is odd} \\ \frac{1}{2}\left(y_{N/2} + y_{1+N/2}\right) & \text{, if } N \text{ is even} \end{cases} \qquad \text{(median, discrete)} \tag{4.16}$$

with

$$y_i \in Y = \left\{ y_1 = \min_j x_j, y_{,s}, \ldots, y_{N-1}, y_N = \max_j x_j \right\}$$

4.1.3.2 Sample Measures

In order to estimate the measures of a population-based on N samples $z(x)$, which are drawn from that population, the following definitions can be used. The *sample mean* $\bar{z}$ is an estimation of the populations mean $\langle z \rangle$. It is defined as

$$\bar{z} \equiv \mu(x) \equiv \frac{1}{N} \sum_{i=1}^{N} z(x_i) \quad \text{(sample mean)}. \tag{4.17}$$

The *trimmed sample mean* $\bar{z}_{p\%}$ is used to exclude outliers from the sample. For the calculation of $\bar{z}_{p\%}$, $p/2$ of the upper and $p/2$ of the lower outliers are ignored, such that

$$\begin{aligned} \bar{z}_{p\%} &\equiv \frac{1}{b-a}\sum_{i=a}^{b} z(x_i) \quad \text{(sample trimmed mean)} \\ \text{with } a = 1+\left\lfloor \frac{p\,N}{2} \right\rfloor,\ b &= N-\left\lfloor \frac{p\,N}{2} \right\rfloor,\ p = \{x \in \mathbb{R} \mid 0 < x < 1\}\,. \end{aligned} \tag{4.18}$$

To calculate the *sample variance* $s^2(z)$, the unknown mean of the population $\langle z \rangle$ must be approximated by $\bar{z}$. In doing so one degree of freedom is used. Hence, the sample variance is defined as

$$\begin{aligned} s^2(z) &\equiv \frac{1}{N-1}\sum_{i=1}^{N}[z(x_i)-\bar{z}]^2 \quad \text{(sample variance)} \\ &= \frac{N}{N-1}\left(\bar{z^2}-\bar{z}^2\right). \end{aligned} \tag{4.19}$$

The *sample standard deviation* is calculated in accordance with EQ 4.14 as

$$s(z) = \sqrt{s^2(z)} \quad \text{(sample standard deviation)}. \tag{4.20}$$

4.1.3.3 Error Measures

Measures for the error between simulation results $x_i \in x$ and an equal number of reference values $x_{i,ref} \in x_{ref}$, which are used in GBPlan and the analysis of its output in this thesis, are listed in Table 4.1.

4.2 Monte-Carlo Method

The goal of the MCM is to calculate the estimate of an expected value $\langle z \rangle$ of a distribution with an unknown PDF, i. e.,

$$\langle z \rangle = \int_a^b z(x)\,f(x)\,dx. \tag{4.27}$$

The method is based on two mathematical theorems, Dunn and Shultis (2012, p. 21):

1. the *Law of Large Numbers* and
2. the *Central Limit Theorem*.

The description of these laws in the following sections leads to an introduction to MCS, a numerical method which applies the MCM to estimate unknown PDFs through repeated, numerical sampling.

For a complete discussion of the MCM refer to Dunn and Shultis (2012), who offer a full explanation of the method. Monahan (2011, pp. 251 sqq.) dedicates one chapter of his book on numerical methods of statistic to MCM. A very mathematical approach to explaining the MCM is taken by Robert and Casella (2004).

Table 4.1: Error types used within the GBPlan framework. The definitions are derived from Gneiting (2011), MathWorks (2019b), and Weisstein (2020c).

Error Type	Definition	
Absolute Error	$\Delta x = x_i - r_{i,ref}$	(4.21)
Relative Error	$\delta x = \frac{x - x_{ref}}{x_{ref}}$	(4.22)
Mean Absolute Percentage Error (MAPE)	$\delta_{\text{MAPE}} = \frac{1}{N}\sum_{i=1}^{N}\left\lvert\frac{x_i - x_{i,ref}}{x_{i,ref}}\right\rvert$	(4.23)
Mean Square Error (MSE)	$\delta_{\text{MSE}} = \frac{\lVert x - x_{ref}\rVert^2}{N}$	(4.24)
Normalized Root Mean Square Error (NRMSE)	$\delta_{\text{NRMSE}} = \frac{\lVert x_{ref} - x\rVert}{\lVert x_{ref} - \bar{x}_{ref}\rVert}$	(4.25)
Normalized Mean Square Error (NMSE)	$\delta_{\text{NMSE}} = \frac{\lVert x_{ref} - x\rVert^2}{\lVert x_{ref} - \bar{x}_{ref}\rVert^2}$	(4.26)

* N is the number of measurements and $\lVert$ denotes the 2-norm, i. e., $\lVert x\rVert = (\sum_{k=1}^{n} \lvert x_k\rvert^2)^{0.5}$

4.2.1 Law of Large Numbers

When expressing the estimate $\langle z\rangle$, EQ 4.27, as the mean $\bar{z}$ of results of a set of samples X with $x_i \in X | X \sim f(x)$, then

$$\bar{z} = \frac{1}{N}\sum_{i=1}^{N} z(x_i). \tag{4.28}$$

For this case, the *Law of Large Numbers* (LLN)

$$\lim_{N\to\infty} \bar{z} = \langle z\rangle \tag{4.29}$$

states, that for a sufficiently large number of measurements N the summation in EQ 4.28 approaches the expected value $\langle z\rangle$.

The LLN, however, does not give information on how large N should be. This problem is addressed by the *Central Limit Theorem* (CLT), Dunn and Shultis (2012, pp. 35 sqq., 41).

4.2.2 Central Limit Theoreme

While the LLN states that an estimate of an expected value can be obtained, the CLT gives information on how accurate this estimate is. Dunn and Shultis (2012, p. 37) presents the CLT in the form of

$$\lim_{N\to\infty} \text{Prob}\left\{\frac{\sqrt{N}\cdot|\bar{z}-\langle z\rangle|}{\sigma(z)} \leq \lambda\right\} = \frac{1}{\sqrt{2\pi}}\int_{-\lambda}^{\lambda} e^{-\frac{1}{2}u^2}\,du. \tag{4.30}$$

From this equation it is visible, that $|\bar{z}-\langle z\rangle|/[\sigma(z)/\sqrt{N}]$ is a unit normal distribution, i. e., $\bar{z}$ is distributed as normal distribution with mean $\mu = \langle z\rangle$ and standard deviation $\sigma(z)/\sqrt{N}$. Less obvious is that the CLT is not limited to a particular type of distribution function to generate N samples of z. As the sample mean $\bar{z}$ is calculated from z it can be concluded that, regardless of the distribution of z, $\bar{z}$ is always approximately normally distributed for large samples ($N \to \infty$). Of course, the only restriction is that it has a finite variance $\sigma(z)$.

The definition of LLN in EQ 4.29 corresponds with EQ 4.30, because the sample mean $\bar{z}$ approximates the true mean $\langle z\rangle$ with $N \to \infty$. This is further expressed on the right side of EQ 4.30, which approaches zero for $\lambda \to 0$.

The sample standard variance s^2, EQ 4.19 and 4.20, can be used to estimate the population's standard deviation $\sigma(z)$ in EQ 4.30. Furthermore, the uncertainty in the estimate of the expected value is proportional to $1/\sqrt{N}$. Because we usually deal with samples rather than the basic population, the population standard deviation $\sigma(z)$ in EQ 4.30 can be replaced by the sample standard deviation $s(z)$ and thus EQ 4.30 be transformed to

$$\text{Prob}\left\{\bar{z}-\lambda\frac{s(x)}{\sqrt{N}} \leq \langle z\rangle \leq \bar{z}+\lambda\frac{s(x)}{\sqrt{N}}\right\} \simeq \frac{1}{\sqrt{2\pi}}\int_{-\lambda}^{\lambda} e^{-\frac{1}{2}u^2}\,du. \tag{4.31}$$

EQ 4.31 allows the quantification of the probability that the calculated mean $\bar{z}$ lies within a confidence interval around the true mean $\langle z\rangle$. By integrating the right side of EQ 4.31, which is also called the confidence coefficient, the nominal confidence limits expressed in percentage can be calculated. Some examples for nominal confidence limits as function of λ are given in Table 4.2.

Table 4.2: Examples of Nominal Confidence Limits as function of λ.

λ	NCL*
0.25	20.00 %
1.00	68.00 %
2.00	95.00 %
4.00	99.99 %

* NCL: $\frac{1}{\sqrt{2\pi}}\int_{-\lambda}^{\lambda} e^{-\frac{1}{2}u^2}\,du$

This is best explained by an example: If we would run a MCS with 100 iterations to simulate the total project time of a construction project and the results of these 100 simulations would have a mean value of $\bar{z} = 320$ [days] and a sample standard deviation of $s(x) = 60$ [days], then we could state that with a confidence of 95 percent ($\lambda = 2$) the real mean $\langle z\rangle$ would be between $\bar{z} \pm 2 \cdot \frac{60}{\sqrt{100}} \Rightarrow 308 \leq \langle z\rangle \leq 332$.

4.2.3 Mone-Carlo Simulation

A MCS is a numerical model, that applies the MCM to calculate an estimate of a value $\langle z \rangle$, EQ 4.27, in which the underlying PDF is unknown. For instance, in a building project schedule, the time required to finish each task is not determined but follows a PDF. With many inter-dependencies between the single tasks, processes have to be executed in parallel or consecutive depending on their predecessors' random duration. In such a complex setting, computing the PDF of the total project duration by convolution, Weisstein (2020a), of the PDFs of single tasks may be impractical if not impossible. MCS offers a solution. In an *analog* MCS it numerically mimics the actual processes involved, i. e., by creating a numerical model of the physical process. A *nonanalog* MCS is used if the cost of computation for an analog model is too high. The nonanalog MCS is a simplified model that has been tweaked to give good results despite not modeling the physical process. In any case, the model of the processes includes randomness. For the example of a construction schedule, the MCS performs a number N of simulations. During each simulation run, the random variables are drawn anew. The LLN states, that when N is large enough, EQ 4.29, the sample mean $\bar{z}$ converges towards the true mean $\langle z \rangle$. The CLT provides a method to estimate the uncertainty of the resulting sample mean and variance. A special form of MCS is the MCMCS. In a MCMCS samples are drawn using a MC, a technology described in the following section.

4.3 Markov Chains

In the Bayesian approach, Bayes (1763), information from $x \in X$ with X having the probability distribution $f(x|\theta)$ is combined with prior information

$$x \in X \mid X \sim f(x|\theta). \tag{4.32}$$

This prior information has its own prior probability distribution with the density $\pi(\theta)$. The posteriori distribution $\pi(\theta|x)$ can be derived from the joint distribution $f(x|\theta)\pi(\theta)$. It extends the information extracted from the observed data by information observed prior to the experiment and is calculated according to Bayes formula

$$\pi(\theta|x) = \frac{f(x|\theta)\,\pi(\theta)}{\int f(x|\theta)\,\pi(\theta)\,d\theta}\,, \tag{4.33}$$

which is discussed in detail in Monahan (2011, p. 261) and Robert and Casella (2004, pp. 12 sq.).

Like in the Bayesian approach, MCs deal with conditional probabilities, i. e., the probability of an event based on knowledge of conditions, which might be related to that event. The term MC was coined by Bernstein (1927) after the mathematician Andrey Markov, who wrote his first paper on the topic over 100 years ago, Markov (1906). It is a series of random variables $x_i \in X, i = 1 \ldots n$ in which each variable x_n depends only on its predecessor x_{n-1}. Monahan (2011, p. 377) and Dunn and Shultis (2012, pp. 133 sqq.). The function which describes the relation between x_i and x_{i+1} is called the transition kernel K, Figure 4.2.

Robert and Casella (2004, p. 208) define the transition kernel K as a function on $\mathcal{X} \times \mathcal{B}(\mathcal{X})$. The kernel K has two essential characteristics: (1) EQ 4.34, which denotes that each element K is a measurement of the probability for the transition from 'x' to any other element '$\cdot$'. And (2) EQ 4.35, i. e., each element A in K can be measured.

$$\forall x \in \mathcal{X}, K(x, \cdot) \tag{4.34}$$

$$\forall A \in \mathcal{B}(\mathcal{X}), K(\cdot, A) \tag{4.35}$$

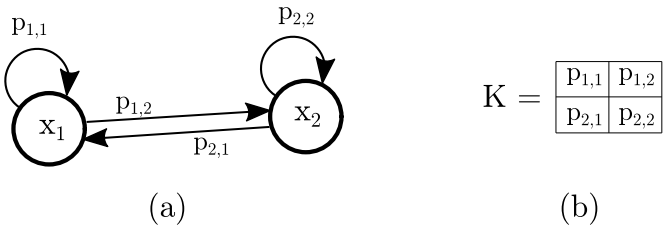


Figure 4.2: Relation between the (a) graph of the Markov-Chain and its (b) kernel K. Each element $p_{i,j} \in K$ represents the probability for the transition from x_i to x_j. Hence, the elements x_{ii} represent the probability that no transition takes place.

In the following section some of the more interesting properties of MCs are being discussed, section 4.3.1. Furthermore, operations on MC which are relevant for functions of GBPlan are explained in section 4.3.2. And finally a popular method to use MC in MCS is being outlined in section 4.3.3.

4.3.1 Properties

If $\mathcal{X}$ is *discrete*, then the transition kernel is called a transition matrix K. The elements of K are

$$P_{xy} = P(X_n = y \mid X_{n-1} = x), \; x, y \in \mathcal{X}. \tag{4.36}$$

The continuous case is discussed in Robert and Casella (2004, p. 208).

The MC is called *homogeneous* if the distribution of all past random variables $X_{t_i, i<0}$ is the same as the distribution of all future random variables $X_{t_i, i>0}$ with

$$\begin{aligned} P(X_{k+1} \in A \mid x_0, x_1, \ldots, x_k) &= P(X_{k+1} \in A \mid x_k) \\ &= \int K(x_k, dx). \end{aligned} \tag{4.37}$$

In other words: In a homogeneous MC the transition kernel is constant over time, while in an inhomogeneous MC it changes; see also Pahl and Damrath (2000, p. 965). Hence, the knowledge of the distribution of the initial state of the homogenous MC is sufficient for the construction of future elements of the chain.

After each transition, the new state m must be selected. For this reason the lines of the transition matrix K are normalized

$$\sum_{j=1}^{m} k_{ij} = 1, k_{ij} \in K. \tag{4.38}$$

Note that the probability to stay in the same state x_i is always $k_{ii} \in K$ and that a transition from x_i to x_j is not possible if $k_{ij} = 0$.

A discrete MC is called *irreducible* if all states of the chain communicate with each other, Robert and Casella (2004, p. 213). With respect to a kernel $K(x, y)$ this means that each element $y \in K$ can be reached from each element $x \in K$ within n transitions such that

$$\forall x \in \mathcal{X}, K^n(x, A) > 0 \tag{4.39}$$

and therefore

$$P_x(\tau_y < \infty) > 0, \quad \forall x, y \in \mathcal{X}. \tag{4.40}$$

In this equation P_x denotes the probability to reach y from x in τ_y transitions.

The chain is called *strongly irreducible* if $n = 1$ for all measurable A such that

$$\forall x \in \mathcal{X}, K^1(x, A) > 0, \tag{4.41}$$

i. e., each element of the MC is connected with itself over a path with the length of $n = 1$ transitions.

4.3.2 Calculating with Markov Chains

A question in practical application is often how great the probability is, that a target state is reached for the first time from an initial state after a number of transitions. With the statements made above, the probability p_{ij} of changing from state x_i to x_j in one step can be calculated as $p_{ij} = k_{ij}$. Consequential the probability of changing from state x_i to x_j after n steps is

$$p_{ij}^{(n)} = \sum_{k=1}^{m} k_{ik}\, k_{kj}. \tag{4.42}$$

Thus, the probability $\pi^{(n)}$ denotes that x has the value x_i after the n^{th} step is

$$\pi^{(n)} = \pi^{(0)} K^n. \tag{4.43}$$

Note that $\pi^{(0)}$ is the initial state of x_i, i. e., for $i = 2$ and $m = 3$ possible states the initial row vector is $\pi^{(0)} = (0, 1, 0)$.

So far we have discussed the case of a MC with discrete random variables. In the case of continuously distributed random variables, the transition probability is a function $p_{ij} = k_{ij} = K(x \to u)$ and the transition probability from one state to another can be computed as $K(x \to u)du$. Dunn and Shultis (2012, pp. 134 sq.) point out that like in the discrete case, EQ 4.38, the transition function is normalized, such that

$$\forall x, u \in [a, b] \int_a^b K(x \to u)du = 1. \tag{4.44}$$

4.3.3 Markov-Chain Monte-Carlo Simulation

A MC is called *ergodic* if the probability of transitioning from one state to another is dependent on when the transition takes place, Dunn and Shultis (2012, pp. 136 sq.). An ergodic MC converges towards a stationary or invariant distribution with $n \to \infty$. Therefore ergodic MCs have their own PDF f and sampling from the ergodic MC is the same as if sampling from f. This feature allows to sample from complex distributions, i. e., multidimensional distributions, and is so powerful that it denotes a subclass of MCS which is called MCMCS. The method goes back to Metropolis et al. (1953) who proposed to use MC to sample from complex distributions. Hastings (1970) became known for their development of the so-called Metropolis-Hastings algorithm, which performs such samplings. Within the framework of this thesis, it is sufficient to understand that MCMCS uses a MC to create samples for a MCS. As such, it is just a particular method to obtain random samples from a PDF. Dunn and Shultis (2012, p. 167) list the advantage of MCMCS as its general structure is easy to implement; it can be applied if only the shape of the distribution is specific; it can be applied to problems for which Bayes theorem is employed; and it can be applied to very complex and high multidimensional sampling problems. For further reading on the topic refer to Metropolis et al. (1953), Hastings (1970), Dunn and Shultis (2012, pp. 133 sqq.), Robert and Casella (2004, p. 267), and Monahan (2011, pp. 375 sqq.).

4.4 Graph Theory

One of the first publications on Graph theory was written in the 18$^{\text{th}}$ century by Leonhard Euler, who discusses a problem which became known as the *Seven Bridges of Königsberg*, Euler (1741) see also Shields (2012). The city of Königsberg has seven bridges that connect four areas of the city, Figure 4.3 a. The question of that time was whether or not it is possible to go for a stroll, crossing each bridge exactly once. After reducing the problem to nodes for each of the four areas and edges for each of the seven bridges, Figure 4.3 b, Euler was able to prove that such a walk is not possible for the given case. In his honor, paths which cross each node of a graph exactly once are called *Euler paths*.

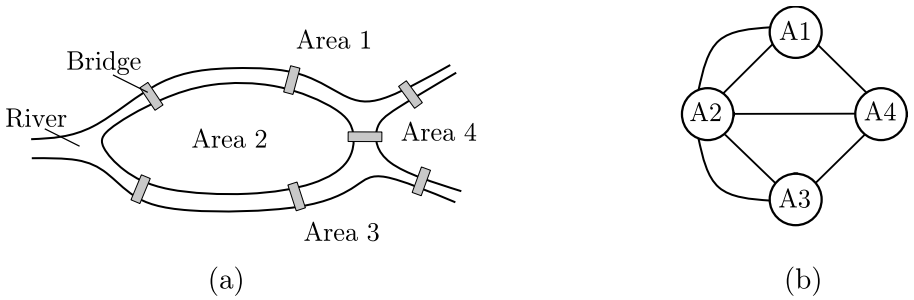


Figure 4.3: Seven Bridges of Königsberg. By transforming the topology of the city of Königsberg (a) to a graph representation (b), Euler was able to prove that there does not exist a path that crosses all bridges exactly once.

Graph theory uses sets and relations to describe graphs. These can be used to describe the relationship between elements through visualization. For this purpose the elements are called *nodes* and the relations are called *edges*. The visualization of such a relation is called a *graph*. The meaning of nodes and edges is depended on the case of application. In the context of this work these cases are in particular:

1. locations and connection between these locations, chapter 8, and
2. states of a system and the transition between these system states, e. g., as depicted in Figure 4.2 for a MC.

An edge between two nodes can be *directed* or *undirected*. In the former case the edge is visualized as an arrow in the graph diagram, Figure 4.4 a, and in the latter as a line, Figure 4.4 b. An edge which connects a node with itself is called a *noose*; see edge e2 in Figure 4.4 c. The geometrical orientation of nodes and edges in the graph diagram is arbitrary, i. e., relocation of nodes for better readability does not alter the meaning of the graph, Pahl and Damrath (2000, pp. 513, 544). Because a graph is based on relations, the algebra of sets can be applied to graphs. For finite sets the algebra of sets can be transformed to a Boolean vector algebra.

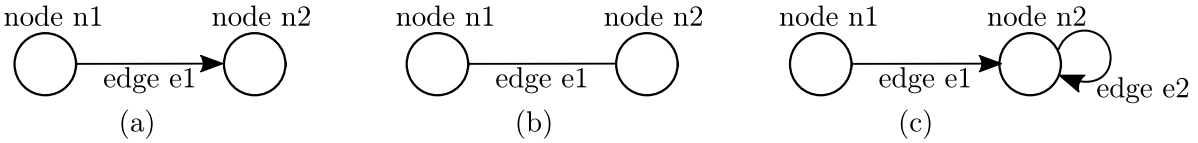


Figure 4.4: Examples of graphs. The two nodes $n1$ and $n2$ are connected with one edge $e1$. That edge can be (a) directed or (b) undirected. Edges which have the same start and end node, such as $e1$ in Figure (c), are called a noose.

This chapter describes the types and structures of graphs, section 4.4.1, and algorithms, in particular path search algorithms, section 4.4.3, which are relevant for this thesis.

A comprehensive description of graph theory for engineers can be found in Pahl and Damrath (2000, pp. 513 sqq.). Furthermore, Diestel (2017) introduces graph theory from a mathematical point of view, while Turau (2009) and Krumke and Noltemeier (2005) put their focus on algorithms for graphs.

4.4.1 Types of Graphs and their Sub-Structures

A graph $G = (V, E)$ consists of two sets. V contains the vertices or nodes $v_i \in V, i = 1 \dots n$ of the graph. E contains touples $e_{ij} = (v_i, v_j)$ which represent the connections or edges between two nodes such that $E \subseteq V \times V$. Two nodes v_i and v_j are adjacent to each other if the edge $e_{ij} \in E$ exists. The neighborhood relations in a graph are denoted in the $n \times n$ adjacency matrix A. In a *simple, undirected graph*, Figure 4.5 a, $a_{ij} \in A$ takes the Boolean value of 1 (true) if the edge e_{ij} exists, i. e., if n_i and n_j are neighbors. Otherwise $a_{ij} = 0$ (false). Consequently, the adjacency matrix of simple, undirected graphs are always symmetric. Edges in *simple, directed graphs* are drawn as arrows to symbolize

their direction, Figure 4.5 b. The elements a_{ij} in the adjacency matrix take the value $a_{ij} = 1$ if an arrow e_{ij} from v_i to v_j exist. Otherwise $a_{ij} = 0$. It is possible that an edge e_{ij} and e_{ji} exist at the same time. Consequently the adjacency matrix A can be symmetric but does not have to be symmetric. Some problems do not only require knowledge of the neighborhood relation of two nodes but also on how strong this neighborhood relation is. This is modeled by *simple, directed, weighted graphs*, Figure 4.5 c, in which an each edge e_{ij} is weighted with a weight $w_{ij} \mapsto W$ such that $W \mapsto E$. This is denoted in A by replacing the binary neighborhood relation used in un-weighted graphs with the weight w such that $a_{ij} = w_{ij}$. This shall be illustrated by an example. Let the nodes represent locations and the weight the time required to travel between these locations. Then $w_{ij} = 0$ would mean that traveling from v_i to v_j takes 0 time units, i. e., happens in an instant. $0 < w_{ij} < \infty$ means that the travel time is w_{ij} time units. And $w_{ij} = \infty$ indicates that there is no edge from v_i to v_j, which is why it would take infinite time units, i. e., forever, to reach v_j.

Graphs can be split into *sub-graphs* S such that $S \subseteq G$. An important type of sub-graph is the *simple path*, Figure 4.5 d. In a simple path, each $v \in V$ has exactly one or no outgoing edge and exactly one or no ingoing edge. All its nodes are linked through its edges such that the end node can be reached from the start node by crossing each node and each edge of the path exactly one time. Consequently, simple paths do not have parallel edges. Two important types of paths are the *shortest path* and the *longest path*. These contain the shortest path $G_s(n_i, n_j)$ and the longest path $G_l(n_i, n_j)$ between two nodes n_i and n_j in a graph. An example for a longest and shortest path in Figure 4.5 c are $G_s(n_1, n_4) = \{n_1 \to n_3 \to n_2 \to n_4\}$ and $G_l(n_1, n_4) = \{n_1 \to n_3 \to n_4\}$.

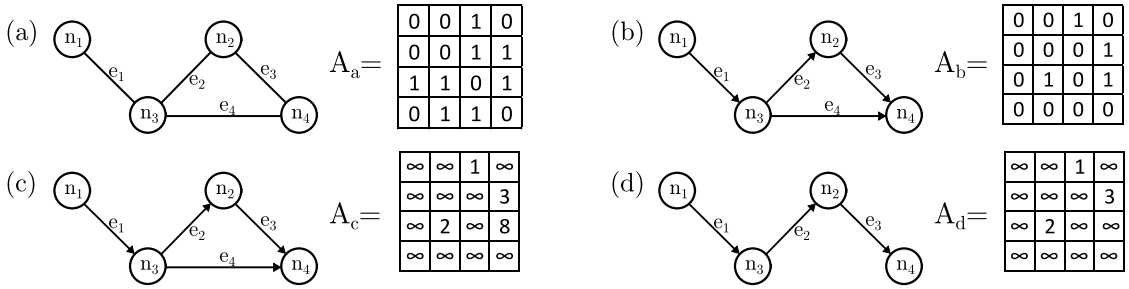


Figure 4.5: Types of graphs and corresponding adjacency matrix A. (a) Simple, undirected graph G_a. (b) Simple, directed graph G_b. (c) Simple, directed and weighted graph G_c. (d) Graph of the shortest path from v_1 to v_4 in G_d.

If the path is an unweighted graph, then the number of its edges is called the *length* of the path. In a weighted graph, the *length* of the path is the sum of its edges' weights. The path is called a *cyclic path* if its start and end nodes are the same. If the simple path contains only one node it is called a *loop*. For instance, in Figure 4.6 a, the sub-graph $S_{loop} = \{V, E\} = \{n_2, e_5\}$ is such a loop. A graph which contains a cyclic path is called a *cyclic graph*, Figure 4.6 c.

A node from which all other nodes of a directed graph can be reached is called *root node*. Searching all nodes of a graph which can be reached from a node results in a *root graph*,

Figure 4.6 b. Cyclic root graphs may contain more than one root node. However, acyclic root graphs, which contain exactly one root node, are of importance for this work. They are used when performing a search in a graph. This is discussed in more detail in section 4.4.3.

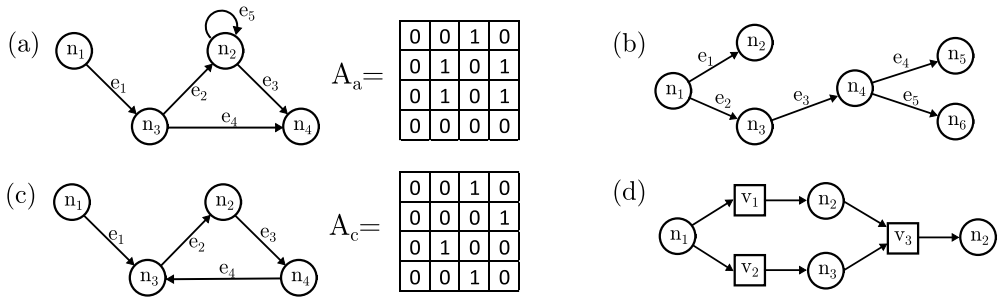


Figure 4.6: Structures of graphs. (a) Simple, directed graph G_a with a loop at n_2. (b) Root tree G_b. (c) Cyclic, directed graph G_c. (d) Bipart graph G_d

A graph with two disjunctive sets of nodes and two disjunctive sets of edges such that $G = (V_1, V_2, R_1, R_2)$ with $R_1 \subseteq V_1 \times V_2$ and $R_1 \subseteq V_2 \times V_1$ is called a *bipart graph*, Pahl and Damrath (2000, pp. 551 sqq.). Figure 4.6 d shows a sketch of a bipart graph. An example of this type of graph has been given in section 2.2.2 for the introduction of PNs.

If the weights of a weighted, directed graph represent probabilities $p_{ij} \in P$ ($P \mapsto E$), then the graph is called a *stochastic graph*. An example has been shown in Figure 4.2, where a stochastic graph is used to visualize a MC. The properties for kernels of MCs, section 4.3.1, are very similar to those of the probability matrix P of stochastic graphs. For instance, the sum of probabilities of all outgoing edges of a node n must equal one, compare EQ 4.45 and EQ 4.38, Pahl and Damrath (2000, pp. 958 sqq.).

$$\sum_{j=1}^{m} p_{j,n} = 1 \tag{4.45}$$

4.4.2 Transitive Hull

The *transitive hull* R_t of a relation R with $R \subseteq M \times M$ of the set M is calculated, by adding all element pairs $(a, b) \in M^2$ to R_t, which are connected through R, Pahl and Damrath (2000, p. 45). Hence, if there is a connection from node v_i to node v_j, then the element (v_i, v_j) exists in E_t. The transitive hull can be used to identify vertices that are connected with a path of the length m. Such a path from v_i to v_j exists, if the m^{th} power of E contains the element (v_i, v_j), i. e., $(v_i, v_j) \cap E^m \neq \varnothing$. The m^{th} power of E is calculated by joining E with itself m times, e. g., $E^3 = E \sqcup E \sqcup E$. With this rule all connections in V by E with a path length $m \leq q$ can be calculated by joining $E \sqcup E^2 \cdots \sqcup E^q$, Pahl and Damrath (2000, p. 540). This is best illustrated by an example. If E contains the elements

(a, b) and (b, c), then E^2 contains the element (a, c) because a path with the length 2 from a to c exists in E , e. g.,

$$E = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}, E^2 = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}. \tag{4.46}$$

4.4.3 Searching in Graphs

A common problem in graph theory is to search a graph. Let a be a node in the simple graph G, Figure 4.7 a. Then the root graph R with the root node a, which contains all successors of a, is called a *search graph* of a. To build the search graph, one starts at node a and then iterates through all successor of a. Two ways of iterating can be distinguished: *depth search* and *width search*.

4.4.3.1 Width Search

The width search starts with the node sequence F. At the beginning of the initial step, F contains only the root node. In a loop, the following steps are executed:

1. If the node n at the beginning of F has a successor s which has not already been visited, then s is added to the end of F.
2. If the node n at the beginning of F has no such successor, then n is removed from F.

The loop ends when F is empty. Figure 4.7 b shows how F develops during the search process of the graph in Figure 4.7 a. The nodes and edges visited during the execution of the width search form the width search graph. The width search graph is build by adding a node y with an edge from x to y, when a node y is added to the end of F with the start node x, Figure 4.7 c. For unweighted graphs, the width search graph contains a shortest path from the root node to each visited node.

4.4.3.2 Depth Search

The depth search starts with the node sequence F. At the beginning of the initial step, F contains only the root node.

1. If the node n at the end of F has a successor s which has not already been visited, then s is added to the end of F.
2. If the node n at the end of F has no such successor s, then n is removed from F.

The loop ends when F is empty. Figure 4.7 d shows how F develops during the search process of the graph depicted in Figure 4.7 a. The nodes and edges visited during the execution of the depth search form the depth search graph. The depth search graph is build by adding a node y with an edge from x to y, when a new node y is added to F with the end node x, Figure 4.7 e. For unweighted graphs, a depth search tree's depth is the longest path from the start node to a node without a successor.

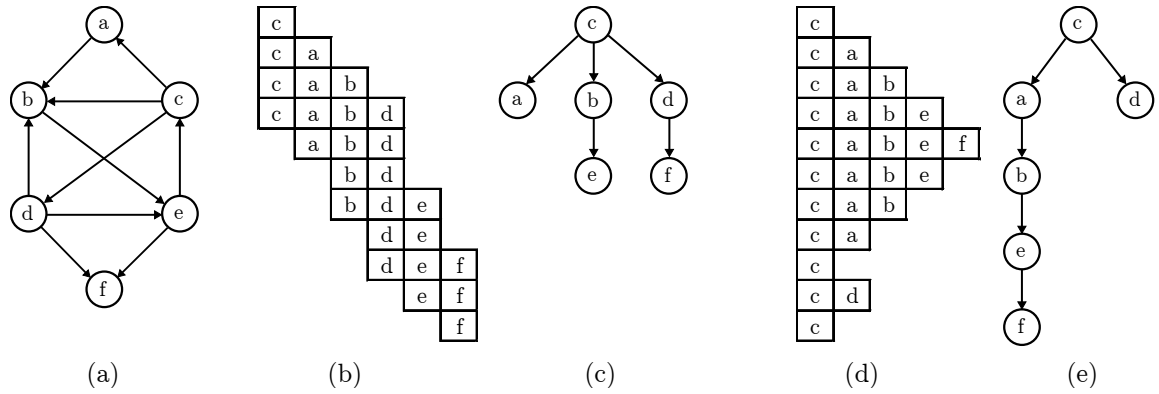


Figure 4.7: Search graphs. (a) A simple graph with the root node c from which all other nodes can be reached. (b) Width search sequence. Depicted is the node sequence F during the construction of the width search graph. (c) The width search graph. (d) Depth search sequence. Depicted is the node sequence F during the construction of the depth search graph. (e) Depth search graph. Example adapted from Pahl and Damrath (2000, pp. 610 sq.).

4.4.3.3 Shortest Path

A typical question in graph theory is what the shortest path between two nodes of a graph is. Furthermore, the shortest path search is a problem that had to be adapted to this work. This adaption is described in chapter 8.

Pahl and Damrath (2000) give an introduction to path algebra, which allows the calculation of shortest and longest paths. For this purpose, the authors define operations for the concatenation and linking of paths, Pahl and Damrath (2000, pp. 614 sqq.). With these operations and the graph's adjacency matrix, the authors create equation systems for minimum and maximum paths. These can be solved using direct methods, such as Gaussian elimination, Lindfield and Penny (2012, pp. 84 sqq.) and Pahl and Damrath (2000, pp. 657 sqq.), or the Dijkstra algorithm, Dijkstra (1959); or iterative methods, such as the method of Jacobi, Gauss-Seidel and Backwards- and Forward Chaining, Lindfield and Penny (2012, pp. 114 sqq.) and Pahl and Damrath (2000, pp. 661 sqq., 671 sqq.).

Without introducing a formal path algebra as in Pahl and Damrath (2000), shortest paths from a root node to all other, reachable nodes of simple, unweighted graphs can be calculated with the width search presented in section 4.4.3.1.

If the graph is weighted, then the number of edges of a path cannot be used to measure its length. The method applied in GBPlan is a modified deep search. The algorithm is nested as `.recursive_deep_search` of the `GBModel.tunnel.get_closest_unfinished_segment()` method. It searches the shortest path to the *next free segment* of a directed, weighted, cyclic graph. The next free segment is a node of the graph that satisfies a series of requirements defined in section 8.2.2. The general procedure is as follows: First, a start node is selected. From that start node, the function moves from node to node according to the above-described depth search procedure. It sums up the weighting for each edge it passes. This

sum s is the distance covered so far. At branches the method recursively calls itself to compute the length l_i of each of the n branches and returns $s^* = s + \min_i(l_i)$. If it finds the node labeled as next free segment, then the function returns the distance s to that node and the node itself. If it reaches a dead end, then the function returns $s = \infty$. If the algorithm reaches a node it has already visited, then that node is not revisited.

4.5 Numerical Optimization

Numerical optimization is an approach to identify the optimal parameters of a system. This usually means identifying a set of parameters X, which minimizes or maximizes the goal or objective function $f(X)$. If g calculates the execution time of a project or the risk inherent in a stock portfolio, those costs or risks are minimized by minimizing $f(X)$, Dori (2016b), Pai (2018), and Kienitz and Wetterau (2012, pp. 482 sq.). In the best case, g can be differentiated, and its extreme points are identified through the zeros of the goal function's derivation $f'(X) = 0$.

Exact methods provide exact solutions of the optimization problem given that enough iterations are being performed. Examples of well established methods of this type are *Linear Programming* (LP), a graphical method for linear optimization problems, which optimizes g under considerations of linear constraints, Domschke et al. (2015, pp. 17 sq.); the Simplex-Algorithm, which examines the boundary of a LP problem for an optimal solution, Domschke et al. (2015, pp. 26 sq.); and the Nelder-Mead Method, also Downhill-Simplex Method, which adapts the Simplex-Algorithm to problems with non-linear constraints, Domschke et al. (2015).

Heuristic Methods promise a solution to problems for which exact methods cannot calculate an optimal solution within an acceptable time. This type of method follows an approach that is adapted to the given problem. Albeit they outpace exact methods in terms of computation time, their clear disadvantage is that they do not guarantee that a best solution is found. However, they usually provide an acceptable and feasible solution, Domschke et al. (2015, pp. 13 sq.). A simple form of heuristic is the *greedy algorithm* which is designed to accept the first best solution in each iteration step without anticipating that future iterations might cast the current solution invalid. The solutions are usually not optimal but feasible. *Local search or improvement methods* start with a valid solution which is either identified using trial and error or a greedy algorithm and search in the proximity of that solution for better solutions. They can be divided into deterministic and stochastic methods. While the former calculates the same results if the input does not change, the latter contains an element of randomness. This leads to different solutions with the same input. The local search stops when a local minimum, which can be significantly worse than the global minimum, is found. *Meta-Strategies*, such as: *Simulated Annealing* (SA), Threshold Accepting and Tabu Search, Kirkpatrick et al. (1983), Aarts (1989), Ingber (1995), and Werra and Hertz (1989), solve this problem by accepting sub-optimal solution during some iterations in order to leave areas of local minima; see also Domschke et al. (2015, pp. 137 sq.). A meta-strategy used in GBPlan is the Evolutionary Algorithm. *Hybrdid-Strategies* combine different methods

to improve the optimization result. For instance, Sonmez and Bettemir (2012) combine GA with SA to solve time-cost trade-off problems.

4.5.1 Evolutionary Algorithms

The idea to adopt procedures from Darwinian evolution, Darwin (1859), to modern problem solving is not new. One of the first monographs written on the topic originates from the 60s, Fogel et al. (1966). Since then, Evolutionary Algorithms have been increasingly applied to various combinatorial optimization problems. Their principle is based on the creation of whole sets of solutions called population. By repeatedly selecting the best individuals of that population and consecutive recombination of the selected best individuals, the solutions improve, imitating the evolutionary process.

Examples of Evolutionary Algorithms are

- the Particle Swarm Method, which imitates the movement of animals towards a local point of attraction such as a food source, Kennedy and Eberhart (1995);
- the Ant Colony optimization, which models ants' ability to find shortest paths by using tracer pheromones, Solnon (2013); and in particular
- the GA, which imitates the iterative adaption of DNA over several generations.

The concept of GA was first presented by Holland (1992) and then used to solve optimization problems by De Jong (1975). Goldberg (1989) provides a detailed description of the implementation of GA. A short excursion on how GA is historically connected to other evolutionary methods such as

- Evolutionary Strategies, Rechenberg (1973) and Schwefel (1981);
- Evolutionary Programming, Fogel et al. (1966); and
- Genetic Programming, Banzhaf (1998) and Brameier and Banzhaf (2007);

is given by Eiben and Schoenauer (2020).

Examples of GA to solve geotechnical problems are given by Kinzler (2011), Pucker and Grabe (2011), Seitz et al. (2018), Seitz and Grabe (2016, 2018), and Seitz et al. (2016). Due to its importance for this work, GA is being discussed in more detail in the following sections.

4.5.2 Single-Objective Genetic Algorithms

The goal of single-objective GA is to find a set of parameters $x \in X$ that minimizes one objective or goal function $f(X)$. The basic principle is shown in Figure 4.8.

During the first step, the initial population is being created. A population P is a set of individuals $p \in P$. Each individual's characteristics are defined by its genome G, which

itself is composed of several genes $g \in G$. In genetic optimization the parameters x are treated as the genes g and consequently a parameter set X as a genome G such that

$$\begin{aligned} \text{(parameter)} \quad & x \Leftrightarrow g \quad \text{(gene)} \\ \text{(parameter set)} \quad & X \Leftrightarrow G \quad \text{(genome)}. \end{aligned} \tag{4.47}$$

In other words, the initial population contains a number of individuals with different parameters. Each of the individuals in that population represents a possible parameter combination, which can be used to evaluate the objective functions $f(x)$.

Once the initial population has been created, each individual p is evaluated based on the objective function f. This evaluation is done to check whether the population meets the optimization criteria and select the best individuals of that population, also known as elite individuals. The simplest criterion is that the algorithm stops if a predefined amount of computation time has passed. A more subtle criterion is that a predefined amount of time has passed without a significant improvement in the solution. If the criterion is met, the

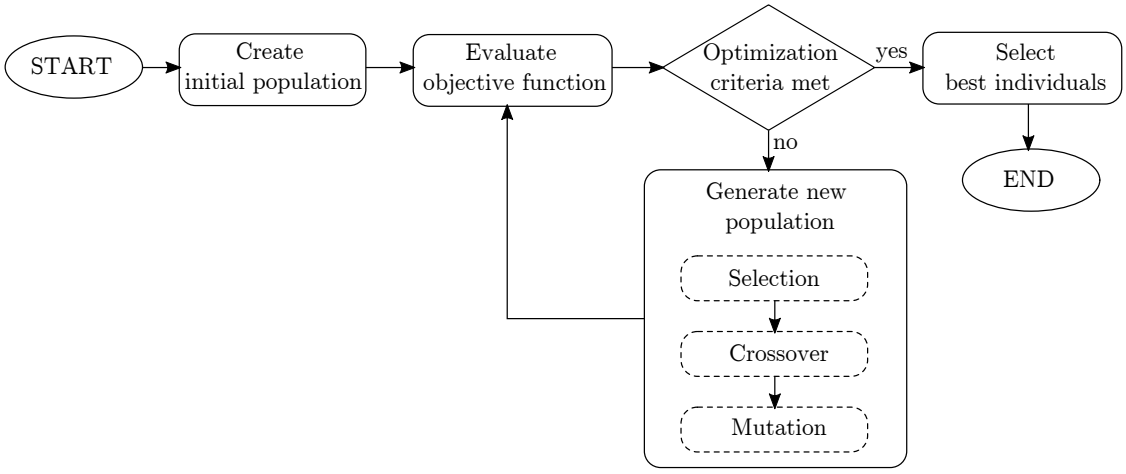


Figure 4.8: Flow chart of *Genetic Algorithm*. After creating the initial population, the algorithm iterates through the cycle of selection, crossover, mutation, and evaluation of the objective functions. If, after the evaluation, the optimization criteria are met, the loop stops, and the best individuals from the last population are selected as the return value.

algorithm stops and returns the best individuals, i. e., the parameter sets X, which yield the best results when being evaluated with the objective function f. Otherwise, a new population P_{i+1} is generated from the elite individuals of the prior population $P_{i=1}$. The creation of a new population is a three-step approach during which (1) the best individuals of P_i are being selected, (2) their genes are being combined according to a ruleset outline further down in this text and (3) finally the genes of the genomes are randomly mutated, i. e., changed within pre-set boundaries. Once the new population P_{i+1} has been created, its individuals are evaluated with the objective function. A check is being performed, whether or not the new population, together with the computation time or the passed number of computation steps, meets the optimization criterion. If so, the algorithm returns the best individuals, and if not, the process starts anew with the creation of a new population.

The process of generating a new population is at the core of GA and distinguishes it from other evolutionary algorithms. As stated above, it comprises of three steps (1) selection, (2) crossover, and (3) mutation. A goal of that three-step approach is to maintain a high diversity of individuals as long as possible to avoid that the algorithm gets stuck at a local minimum, Figure 4.9. During the selection process, each individual is rated with the

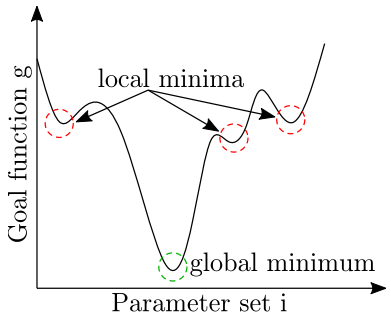


Figure 4.9: Local Minima Problem. Optimization algorithms must be able to leave local minima in order to find the best global solution. In the example shown in Figure 4.9, three local minima exist. Those local minima may attract the search algorithm to narrow down its search to the close proximity of one of the local minima, thus missing the global minimum. GA employs two tactics to avoid getting stuck at a local minimum. First, it searches in a wide range to increase the chance to search in the valley, which belongs to the global minimum. Second, its mutation function adds randomness to the algorithm. That randomness allows the algorithm to escape a local minimum's valley and increases the chance to find the global minimum.

objective function. Then, the individuals with the best ratings are selected to pass their genes to the next generation. These best individuals are also called elite children. The genomes of the elite children are being used in the crossover to generate new individuals for the next generation. The crossover is done by pairwise combining the genomes' genes. Figure 4.10 provides a sketch of this process. First, it is determined via roulette selection, which of three possible crossover methods is being applied to two individuals. Within the framework of this study 10 % of the genomes experience the *Single-Point Crossover* (SC), 20 % the *Two-Point Crossover* (TC) and 70 % the *Uniform Crossover* (UC). Then the actual crossover is performed. In the case of a SC the genomes are being cut at a random position. Then the two resulting gene-segments are swapped. The TC performs two cuts and swaps only the middle gene segment, and during the UC each gene is swapped with a chance of 50 %. The resulting genomes are called the offspring of elite children. These are further exposed to mutation, during which randomly selected genes of the elite children are randomly changed.

The combination of selection and crossover enables the algorithm to recombine the genes of the best individuals into offspring, which has the potential to be superior to their parent generation. The mutation ensures genetic variety, which increases the likelihood that superior individuals are being found. A detailed description of the implementation of the first population's creation, the crossover, and the mutation, as implemented in GBPlan is given in section 7.1.6.

For further in-depth reading on crossover and mutation Deep and Thakur (2007a,b) present two interesting papers on their implementation of these functions for GA. In the field of construction management several authors have applied GA to find near optimal solution for construction schedules, e. g., Chua et al. (1997), Damci et al. (2013), and Hegazy (1999a,b). Fahihi et al. (2014) propose an interesting approach in which they extract information from a BIM database to automatically create such near-optimal schedules using GA. Furthermore, the optimization of space is a problem, which Alanjari et al. (2015) discuss for the optimization of the material yard of construction sites and Azadivar and Wang (2000) for facility layouts. For a more general introduction to GA the interested reader may refer to Lindfield and Penny (2012, pp. 397 sqq.), Haupt and Haupt (2004), and Yang (2014, pp. 77 sqq.).

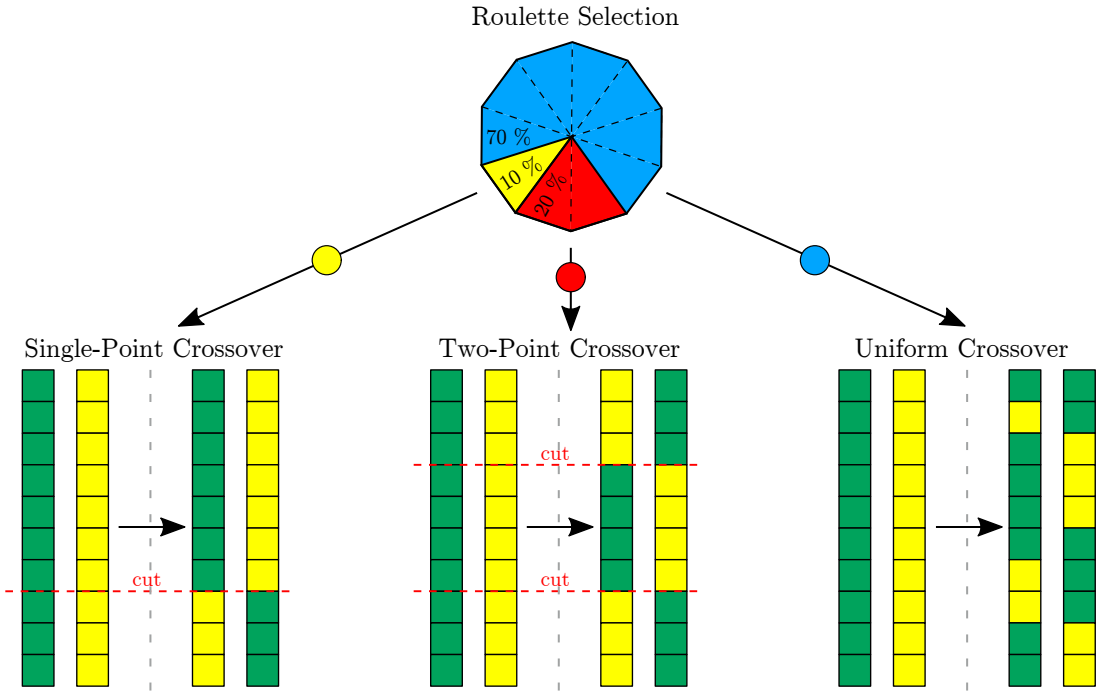


Figure 4.10: Crossover. During the Roulette-Selection process it is decided which type of crossover procedure each of the pairs of individuals experience. In 10 % the Single-Point Crossover, in 20 % the Two-Point and in 70 % of all cases the Uniform Crossover is selected. The main difference of the three methods is the number of positions at which the genes are being cut for the consecutive swap of gene sequences. For the Single- and Two-Point Crossover the position p of the cut is selected with a random variable $p \sim \mathcal{U}\{1, n-1\}$ with $p \in \mathbb{Z}^{+}_{\neq 0}$ and n being the number of genes in the genome. The Uniform Crossover calculates a random variable $u \sim \mathcal{U}(0, 1)$ for each gene and swaps the genes if $u < 0.5$.

4.5.3 Multi-Objective Genetic Algorithm

Instead of optimizing towards a single goal, MOGA is a search strategy that tries to find the Pareto optimal set of solutions – a set of solutions that contains the best compromises between multiple objectives. This set of Pareto optimal solutions is called the Pareto frontier. A typical example of such a set of objectives in construction management is time and cost. Figure 4.11 shows a sketch of a two-objective problem, where the goal is to minimize both objectives. MOGA tries to find solutions as close as possible to the Pareto frontier. A solution $x^* \in X$ is a noninferior solution and lies on the Pareto frontier, see MathWorks (2019c), if in the neighborhood of x^* there does not exist a Δx such that $(x^* + \Delta x) \in X$ and

$$\begin{aligned} &f_i(x^* + \Delta x) \leq f_i(x^*), i = 1, \ldots, m, \text{ and} \\ &f_j(x^* + \Delta x) < f_j(x^*) \text{ for at least one } j. \end{aligned} \tag{4.48}$$

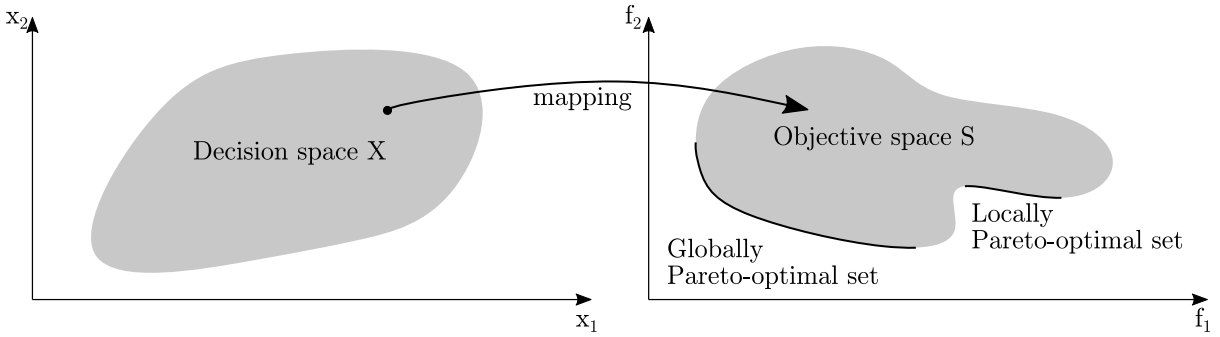


Figure 4.11: Decision space (left) and Search space with Pareto optimal set (right). The decision space X contains all possible combinations (x_1, x_2) of the decision variables. The objective space S comprises of all feasible combinations $(f_1(X), f_2(X)) \in S$ which may occur during the variation of the input parameter set X. Solutions outside the decision or objective space are not feasible, i. e., there does not exist a parameter set X, which yields a solution (f_1, f_2) outside the search space. If the goal is to minimize both goal functions f_1 and f_2, then the global Pareto optimal set contains all those solutions $s \in S$ for which one of the goal functions cannot be decreased without increasing the other. The problem of finding a local instead of the global Pareto optimal set exists for MOGA in the same way, as it does for local minima in single objective genetic optimization; see Figure 4.9.

When dealing with MOGA note, the method uses two search spaces: the decision space and the search or objective space. Even though a relationship between the two spaces exists through a unique mapping, the proximity of two points in the decision space does not implicate the same for two points in the objective space. It is important to remember this when trying to have a high diversity during searching because the diversity of decision space variables does not guarantee the same for objective space variables.

While searching through the search space S, the algorithm may find locally Pareto optimal sets. To avoid that the algorithm gets stuck at such a local set, it must find and evaluate

a set of solutions as diverse as possible. Furthermore, a diverse set of solutions is essential to converge towards the Pareto frontier and identify the whole front. Within GBPlan the selection methods of the MATLAB function `gamultiobj` assure diversity through the selection of individuals according to three principles: *Rank*, *Crowing distance*, and *spread*; see MathWorks (2019a). The *rank* is a measure of how many individuals dominate or are dominated by another individual. Individuals with rank 1 are not dominated by any other individuals. On rank 2, individuals are only dominated by rank 1 individuals, and on rank k, individuals are only dominated by individuals with rank $\leq k-1$. Individuals with a higher rank have a higher chance of taking part in creating the next generation. The *Crowding*

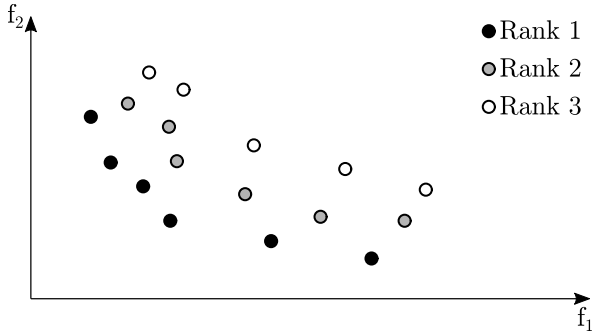


Figure 4.12: Rank. The rank is the same for all individuals on a Pareto frontier. A lower rank increases the chance of being selected to participate in the creation of the next generation.

distance measures how close the individuals of the same rank and population are together. By default, the distance is measured in objective space S, albeit using the decision space X is optional. The distance value of an individual is the distance to its sorted neighbors. Therefore, if $x_{m,i}$ is a list of sorted individuals with m being the dimension, i. e., the index for the objective function, and i being the index of the sorted neighbor, then

$$d_i = \sum_{i=1}^{m} \left(x_{m,i+1} - x_{m,i-1} \right). \tag{4.49}$$

Because a higher distance improves diversity, those individuals with a higher distance value within the same rank have a higher chance of being selected to create the next generation. The *Spread* is a measure of the change between consecutive Pareto sets. It is calculated as

$$\text{spread} = \frac{\mu + \sigma}{\mu + k^* \cdot \sigma}. \tag{4.50}$$

In this equation σ is the standard deviation, EQ 4.14, of the crowding distance, EQ 4.49, of points on the same Pareto frontier. For the calculation of μ, the algorithm sums up the norm of the distance between the minimum value of the current Pareto frontier and that of the Pareto frontier of the previous generation for each of the k dimensions or objective functions. The spread is small when both σ and μ are small, i. e., when the extreme objective function values do not change much between two generations and when the points on the Pareto frontier are spread evenly. The spread is used as a stopping

criterion. The optimization loop stops when the final spread is less than an average of recent spreads and does not change much. The presented three techniques to preserve diversity are only a small selection of possible methods. For a comprehensive introduction to the topic of diversity preservation, refer to Goh and Tan (2009, pp. 11 sqq.).

Examples of the application of MOGA in construction management are presented by Kim et al. (2016) and Senouci and Mubarak (2016) who optimize construction schedules with regards to the time-cost trade-off. Agrama (2015) does this optimization for nonidentical multi-story building projects, Senouci and Al-Derham (2008) show how to apply MOGA for the optimization of linear construction projects. Senouci and Eldin (2004) propose a model to optimize construction schedules under consideration of multiple crew strategies, total project cost minimization, and time-cost trade-off. Moon et al. (2014) minimizes clashes of resources on construction sites by extracting information from a 4D BIM system to create schedules with minimum resource time and space overlap automatically. Monghasemi et al. (2015) present a MOGA for the search of best Pareto solutions for discrete time-cost-quality trade-off problems. The authors test their algorithm on a fictional highway project modeled as 18-node AON network; see also Feng et al. (1997, 2000).

A very good introduction to MOGA is given by Deb (2009). With a focus on the multi-objective aspect Abraham et al. (2005) and Goh and Tan (2009) elaborate on MOGA and other multi-objective evolutionary algorithms. A rather mathematical discussion of Pareto optimality of multi-objective problems is presented by Censor (1977).

4.5.4 Mixed-Integer Genetic Algorithm

A mixed-integer problem is an optimization problem in which some or all decision variables are restricted to have integer values. Several methods exist to solve mixed-integer optimization problems, such as the branch and bound technique, the cutting planes technique, or the outer approximation technique; see Conforti et al. (2014), Cooper (1981), Floudas (1995), Grossmann (2002), and Marchand et al. (2002). Albeit these methods perform well for their particular class of problem, they may cut off the global optima when dealing with non-convex objective spaces, Deep et al. (2009). A robust implementation of mixed-integer GA is presented by Deep et al. (2009). An adaption of the MATLAB 2019a real MOGA function `gamultiobj` to serve as mixed-integer MOGA in the GBPlan SPS framework is presented in section 7.1.6.

4.6 Deep Learning

Deep Learning is a kind of Machine Learning, which itself is a kind of Artificial Intelligence. In Machine Learning training data is used to train a model. Once properly trained, the model calculates a correct output from input data, which differs from the training data, Figure 4.13. For successful training, a sufficient amount of unbiased training data must be available. A typical problem is overfitting, which occurs, when the model has been overly customized to the training data. In the case of overfitting, the model produces good

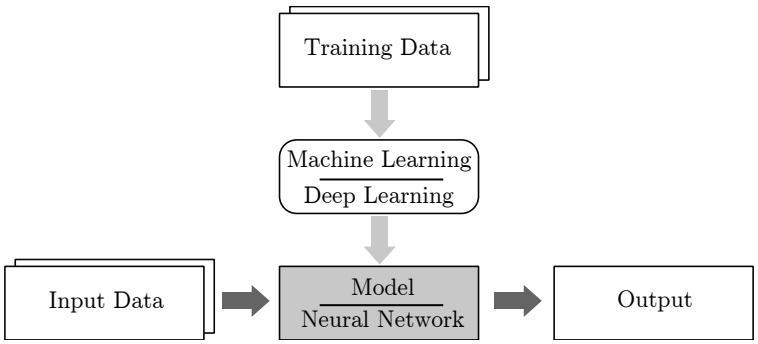


Figure 4.13: Deep Learning is a form of Machine Learning which uses multi-layer ANNs. Before an ANN can be used, it must be trained with tailored training functions and training data. Once properly trained, the model transforms input data which differs, from the training data, to correct output data. Figure adapted from Kim (2017, p. 5, Figure 1-4).

results for the training data but shows poor performance when used with data that is not part of the training set. Depending on the training method, Machine Learning can be distinguished in supervised, unsupervised, and reinforced learning. The three methods differ in the kind of training data used to train the model: {`input`, `correct output`} for supervised, just {`input`} for unsupervised and {`input`, `some output`, `weight for the output`} for reinforced learning. In the framework of this thesis, supervised learning is being applied. The problems tackled by supervised learning can be coarsely divided into classification problems, e. g., face recognition or spam mail filtering, and regression, e. g., finding a model that estimates income by age based on a data set containing income and age samples, Kim (2017, pp. 14 sqq.). Machine Learning becomes Deep Learning, when the applied model is a deep neural network, i. e., a multi-layer ANN with two or more hidden layers. The following sections introduce the Deep Learning technologies used in this thesis, section 5.6.3. This is in particular the FNN and the *Scaled Conjugated Gradient Algorithm* (SCGA) used for its training.

For further reading on deep learning, Kim (2017) and Hagan et al. (1996) give a very understandable introduction to the topic of ANNs and deep learning. Patan (2008) requires the reader to have an understanding of the basics of ANNs and aims at using ANNs for numerical fault diagnosis. Anastassiou (2011) writes solely about FNNs, the network type which is also used in this thesis.

4.6.1 Structure of Artificial Neural Networks

The brain stores information through the association between a great number of neurons. The neuron itself does not store information. Its job is to transmit signals coming from and going to other neurons. Hence, when the brain learns new information, it alters the association between the neurons to reflect the new information. ANNs imitate this method.

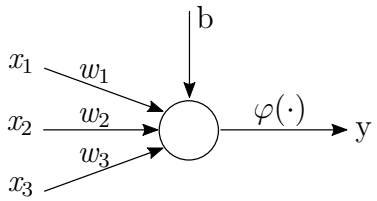


Figure 4.14: A single node of an ANN receiving three inputs x_1, x_2, x_3. These are weighted with the three weights w_1, w_2, w_3 and then translated into output y with the activation function φ, taking the bias b into account. Figure adapted from Kim (2017, p. 20, Figure 2-2).

Instead of neurons, these structures use nodes, and instead of the association between neurons ANNs apply weights to the connection between their nodes.

The signal flow in an ANN node with three input signals, is shown in Figure 4.14. x_1, x_2, x_3 are the input signals and w_1, w_2, w_3 their weight. The value b is a bias, which serves as another factor to store information. The output signal of the node is y. It is calculated as a function of the three input signals $y = \varphi(v)$ with v being the weighted sum of the input signals plus the bias, EQ 4.51. The function φ is called the activation function and determines the behavior of the node. Many different types of activation functions are possible.

$$\begin{aligned} v &= wx + b \\ \text{with } w &= \begin{bmatrix} w_1 & w_2 & w_3 \end{bmatrix}, \; x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \end{aligned} \tag{4.51}$$

The described structure provides ANNs with two essential features. First, they can deal with non-linear problems, and second, they can learn from historical training data. In doing this, they require little or no a priori knowledge about the process that lies behind the modeled real-world problem, Patan (2008, p. 29).

ANNs comprise of several layers which can be distinguished in: input layer, hidden layer(s) and output layer, Figure 4.15. Each layer contains several nodes connected to all other nodes of the layer before and after that layer. Nodes in the same layer, however, are not connected. The nodes on the input layer simply pass the input values to the next layer and do not apply the weighted sum or the activation function. The nodes in the output layer provide the final result of the ANN. The layers between the input and output layer are the hidden layers, called as such because they are usually not accessible from outside the network. ANNs without hidden layers are called single-layer networks, while those with hidden layers are called multi-layered networks. If the multi-layered ANN has just one hidden layer it is called a shallow ANN and with two or more hidden layers it is called a deep neural network. Note that the network in Figure 4.15 has the structure of a directed, acyclic graph, section 4.4, and that a layer l_i is connected exclusively to a layer l_{i+1}. This type of network is called a FNN and of particular importance because it can be used

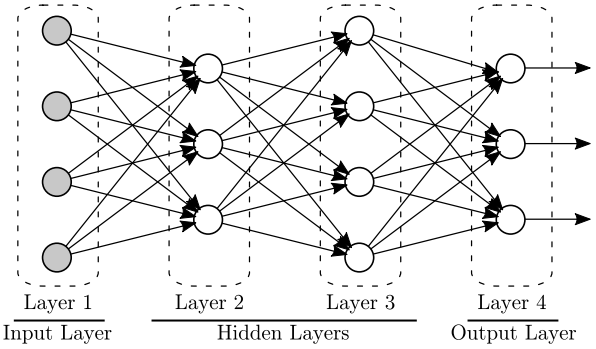


Figure 4.15: Signal flow in a *Feedforward Neural Network*. The signal moves in one direction, from layer i to layer $i+1$. In general, three types of layers are being distinguished: the input layer, the output layer and a number of hidden layers located between the former two. Connections within layers or from a layer i to layer $i-1$ do not exist.

to approximate any continuous function to any degree of accuracy by simply increasing the number of hidden neurons, Anastassiou (2011, p. 1). A good start configuration for building ANNs is given by Livshin (2019, p. 10), who proposes to use n_{hidden}, neurons per hidden layer before iterating to an acceptable network configuration

$$n_{hidden} = 2 \cdot n_{\text{input}} + n_{\text{output}}. \tag{4.52}$$

For further reading on the mathematical fundamentals on the approximation of functions with ANNs refer to Barron (1993), Cao et al. (2008), Chen and Chen (1995), Chui and Li (1992), Cybenko (1989), Ferrari and Stengel (2005), Funahashi (1989), Hahm and Hong (2004), Hornik et al. (1989, 1990), Leshno et al. (1993), Maiorov and Meir (1998), Makovoz (1998), Mhasjar and Micchelli (1992), Mhaskar and Micchelli (1995), Suzuki (1998), and Zongben and Feilong (2004).

Examples for other multi-layer networks are *Cascaded Forward ANNs*, which have additional weighted connections from each node of layer l_i to each downstream node in layer $l_{j>i}$, Al-Allaf (2012) and MathWorks (2019b); *Convoluted ANNs*, which have a two- or three-dimensional arrangement of the nodes, build their hidden layers from sub-structures called convoluted layers and are commonly used in image recognition Michelucci (2019) and Kim (2017, pp. 121 sqq.); or *Recurrent ANNs*, which allow connections between nodes of the same layer or nodes to those located in upstream layers and show good performance in recognition of handwriting or speech, Mandic and Chambers (2001) and Patan (2008, pp. 29 sqq.).

4.6.2 Training Artificial Neural Networks

ANNs must be adequately trained to yield useful results, i. e., the weights w_i must be set to a value that transforms the input to the desired output. In supervised learning, the network calculates output from the input. It compares the result with a set of correct output data. The error between output and correct output is calculated and, based on

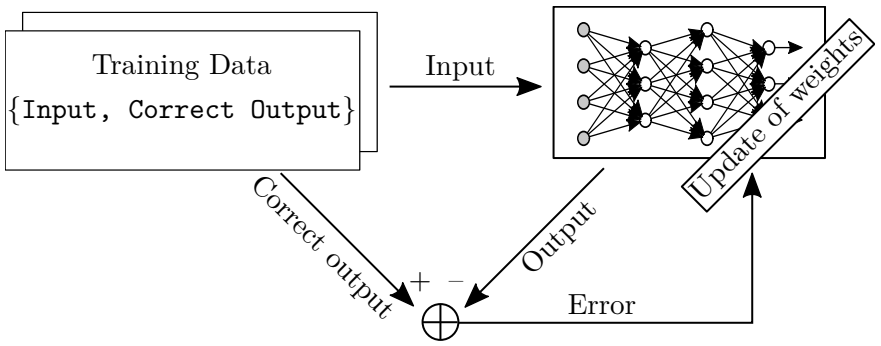


Figure 4.16: Principle sketch of the iterative process of supervised learning of an ANN. Training data is used to create the output of the ANN. From the difference between the output and the correct output, the error is calculated. The error is further used to update the weights of the ANN. This loop of output creation, error calculation, and update of weights is repeated until the error reaches a pre-set limit.

that error, the weights of the ANN are updated. Once this is done, the process starts anew until a stop criterion is met, e. g., the error falls below a pre-defined limit; see Figure 4.16. The systematic approach to modify the weights during the training process is called the learning rule.

The delta rule (also Adaline or Widrow-Hoff rule) expressed in EQ 4.53, is used in single-layer networks and calculates how much an input node x_j contributes to the error of an output node y_i, so that the weight w_{ij} can be updated accordingly, see EQ 4.54. The learning rate α scales the speed in which the weight w_{ij} is adapted in each iteration step k. If α is set too high the solution might not converge, while the convergence speed can be inappropriately slow if it is set too low, Kim (2017, pp. 27 sqq.).

$$\begin{aligned} \delta_i &= \varphi'(v_i) \cdot e_i \\ &= \varphi'(v_i) \cdot (d_i - y_i) \end{aligned} \tag{4.53}$$

$$w_{ij,t+1} = w_{ij,t} + \alpha \cdot \delta_i \cdot x_j \tag{4.54}$$

with t = Iteration step
x_j = Output from input node j
e_i = Error of output node i
d_i = Correct output at node i
y_i = Calculated output at node i
φ' = Derivative of the activation function φ
v_i = Weighted sum of the output node i
w_{ij} = Weight between node i and node j
α = Learning rate $(0 < \alpha \leq 1)$

Different training algorithms exist to train different types of networks. The fundamental algorithm for FNNs is the *Back-Propagation* (BP) algorithm, Rumelhart et al. (1985, 1986)

and Werbos (1974, 1994). This algorithm solves the problem of adjusting the weights of the hidden layers. In BP the calculation of the error term e_i, which is used for the application of the delta rule, EQ 4.54, differs for nodes located in hidden layers and those located in the output layer. BP first applies the delta rule as described above for the output layer l_i. Then the algorithm goes backwards through the hidden layers l_{i-1} and calculates the error based on the error of the next adjacent downstream layer l_i

$$\begin{aligned} \delta_{i-1} &= \varphi'(v_{i-1}) \cdot e_{i-1} \\ &= \varphi'(v_{i-1}) \cdot w_i \cdot \delta_i. \end{aligned} \tag{4.55}$$

With this the updated weight in the next iteration step $t+1$ becomes

$$\begin{aligned} \Delta w_{ij} &= \alpha \cdot \delta_i \cdot x_j \\ w_{ij,t+1} &= w_{ij,t} + \Delta w_{ij}. \end{aligned} \tag{4.56}$$

A challenge when training deep neural networks is to maintain a high stability while minimizing computation time. By introducing a momentum term m, the weight adjustment is influenced to achieve these two goals, Kim (2017, p. 65). However, Møller (1993, p. 527) remarks that the speed benefit of the momentum term is minor and the that the algorithm becomes less robust by introducing another user defined parameter. The term is added to the delta rule and is supposed to smoothly drive the weight adjustment into the desired direction instead of causing a sudden change. In this sense, m acts similarly to a physical momentum that interferes with the body's reaction to external forces, Kim (2017, p. 65). This is expresses in

$$\begin{aligned} \Delta w &= \alpha \cdot \delta \cdot x \\ m_t &= \Delta w + \beta \cdot m_{t-1} \\ w &= w + m_t \ , \end{aligned} \tag{4.57}$$

where m_{t-1} is the momentum from the last iteration step and $\beta = \{x \mid 0 < x < 1\}$ a constant value, which lets the influence of older momenta fade out as more and more training iterations are being computed.

The training algorithm used within the framework of this thesis is the SCGA. SCGA is implemented in the MATLAB function `trainscg` and uses backpropagation including a momentum term. It further implements a search strategy based on conjugate gradients, which Møller (1993) describes in detail. The strategy combines the Levenberg-Marquadt algorithm, a variation of the standard Newton algorithm, with the conjugated gradient approach, Fletcher (2000, pp. 80 sqq., 100 sqq.), to include second-order information when updating the weights. While usually not as fast as the Levenberg-Marquardt algorithm, the SCGA has the important advantage of significantly less memory usage.

5 Digitalization of Construction Site

The world's most valuable resource is no longer oil, but data.

The Economist (2017)

This chapter deals with knowledge discovery through digitalization. The applied *Knowledge Discovery Process* (KDP) has been proposed by Maletic and Marcus (2005) and follows a 7-step approach, in which the data collected at the construction site is refined until new knowledge can be generated from that data. The structure of this chapter's sections

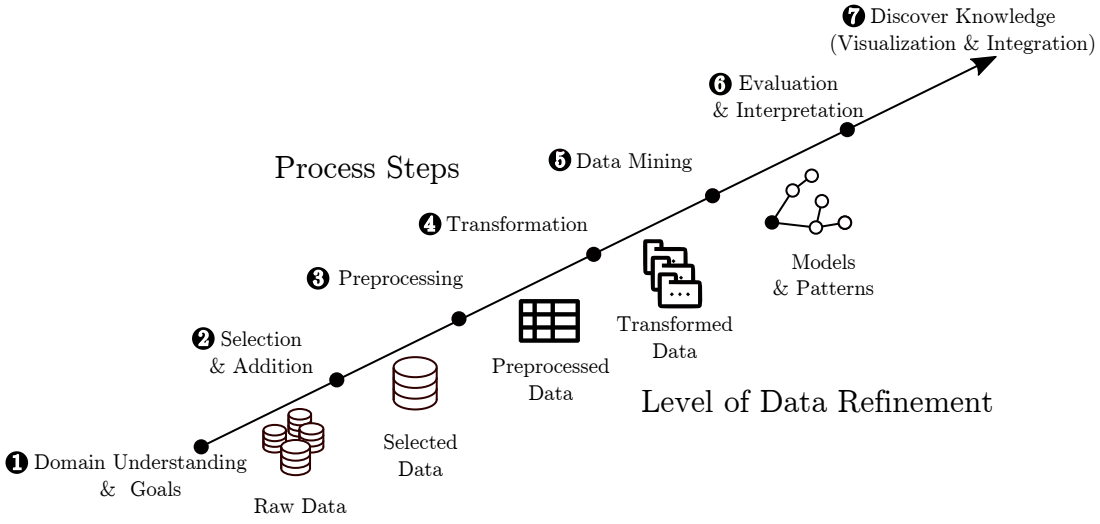


Figure 5.1: The process of knowledge discovery in databases. Figure adapted from Maletic and Marcus (2005, p. 3, Figure 1.1).

reflects that process, displayed in Figure 5.1, in which (1) an understanding of the domain (the construction site), in which the KDP takes place, is developed, section 5.1, (2) a raw data set on which the discovery process is being performed is defined, section 5.2, (3) the raw data collected in the previous step is being pre-processed, section 5.3, which includes, in particular, the cleaning of the raw data in order to improve its quality and prepare it for (4) the transformation of the data set to a format which suits the needs of the following steps, section 5.4, (5) during which the data set is mined, section 5.5, i. e., it is analyzed, to provide models with sufficient input of high quality to create meaningful output, section 5.6, which is further (6) evaluated and interpreted in order to (7) discover knowledge hidden in the data.

The KDP has been implemented in GBPlan. Taking the point of view of the data, Figure 5.2 describes its flow inside and outside of the program's framework. The data enters GBPlan via csv-file and is stored as a structured table named `rawData`. This table is further transformed into two GBPlan data structures: `structData` and `structBores`. During the transformation process from `rawData` table to GBPlan specific formats, GBPlan applies a number of rules to the data set in order to improve its quality for the analysis.

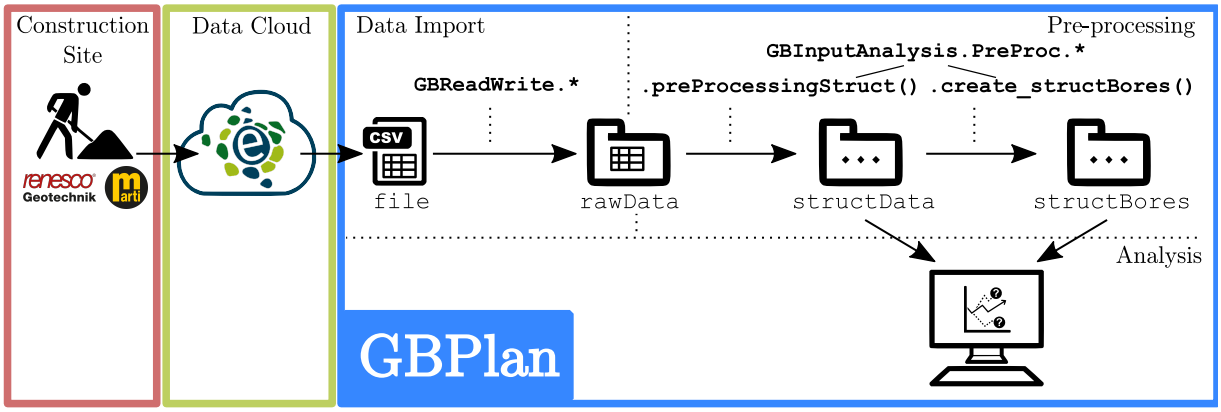


Figure 5.2: Flow of data in the GBPlan framework. The data is generated at the construction site, digitized and uploaded to the eguana cloud. GBPlan receives a reduced data set as csv-file, which is further processed to a MATLAB structured array, the `rawData` table. This table is further transformed to two data structures. `structData` sets its focus on the performance of machines and `structBores` on properties, which are linked to specific boreholes; see also section 5.4.

5.1 Description of Construction Site "Tunnel Feuerbach"

The construction site of Stuttgart 21 has been exhaustively described in many journal papers, newspaper articles, and of course, the internet. This chapter is based on the articles Lienhart et al. (2019), Sabew et al. (2019a,b), Wittke (2014), and Wittke et al. (2017) on the sub-trade of injection grouting at the tunnel Feuerbach, which has been managed by Renesco GmbH. In order to ease reading, citations have been limited to selected highlights.

Tunnel Feuerbach connects the Stuttgart central railway station with the railway station Feuerbach, Figure 5.3. It leads through an Anhydrite-bearing Gipskeuper, a rock that undergoes a swelling process when it comes in contact with water, section 5.1.1. To counteract the detrimental influence of the swelling on the tunnel structure, the planning company, WBI GmbH, has developed an Anhydrite concept for the tunnel construction, section 5.1.2. The contractor who executes WBI's concept through extensive injection works is Renesco GmbH, section D.1. Furthermore, Renesco GmbH subcontracted eguana GmbH, section D.2, with the development of a digital documentation tool to meet the high quality and documentation specification of the project. The tool digitizes the process data from the construction machines and stores it on the companies cloud server for further processing

into meta information such as *Key Performance Indicator* (KPI) slides, mass tables and injection protocols, section 5.1.4.

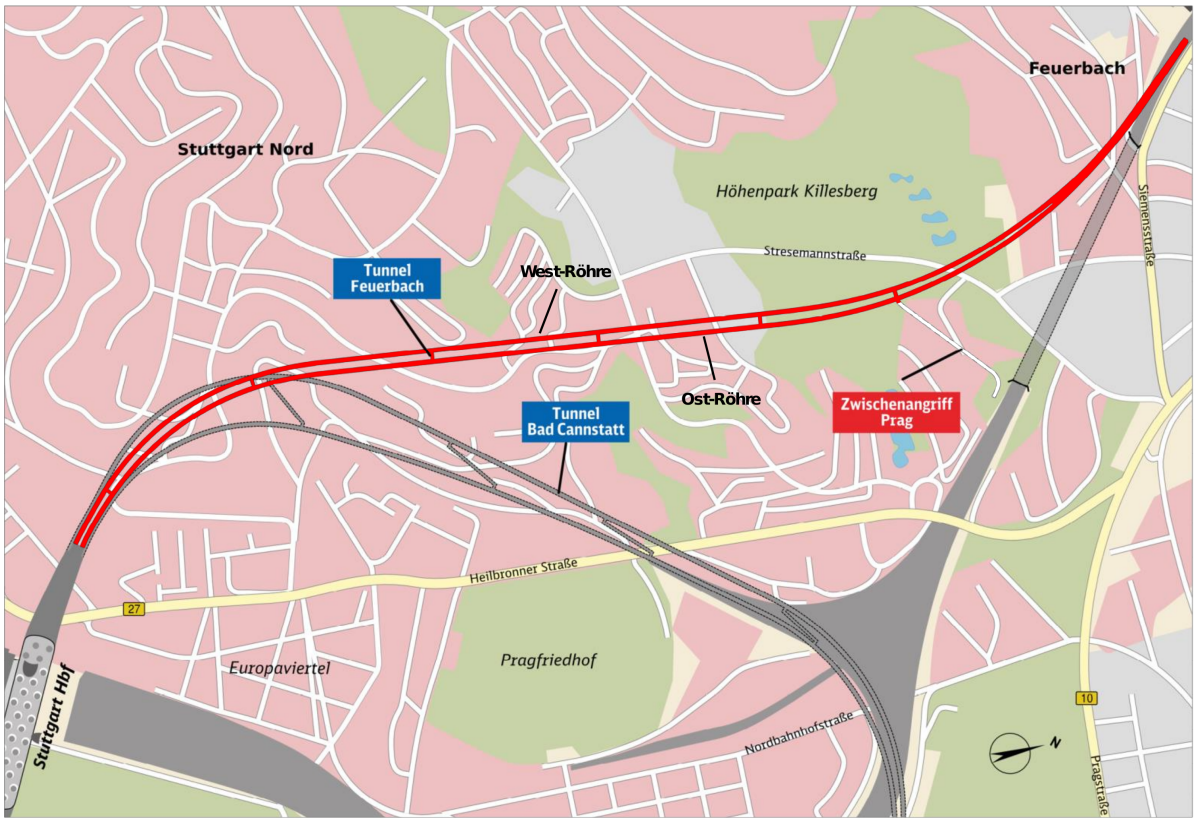


Figure 5.3: The tunnel Feuerbach connects the main station Stuttgart with the station Feuerbach. The tunnel consists of two parallel tubes, which are connected with several connection buildings. Construction equipment enters the tunnel construction site at Zwischenangriff Prag. Source DB Projekt Stuttgart-Ulm (2019).

5.1.1 Local Geology and Anhydrite Swelling

As part of the Stuttgart 21 project, the tunnel Feuerbach, an approximately 3.2 km long, twin-tube railway tunnel, is being built. The tunnel passes through layers of Anhydrite-bearing Gipskeuper. When in contact with water (H_2O) the Anhydrite ($CaSO_4$) goes into solution, EQ 5.1, and then crystallizes, EQ 5.2. The resulting product is Gypsum ($CaSO_4 \cdot 2H_2O$) which incorporates two water molecules into its chemical structure. This leads to an increased volume of the resulting gypsum body by approximately 61 percent.

$$CaSO_4 + 2H_2O \longrightarrow Ca^{++} + SO_4^{--} + 2H_2O \tag{5.1}$$

$$Ca^{++} + SO_4^{--} + 2H_2O \longrightarrow CaSO_4 \cdot 2H_2O \tag{5.2}$$

With about 30 percent of Anhydrite in the Tunnel's Gipskeuper, an increase of the rock volume by roughly 18 percent can be expected. The resulting swelling pressure of up to 10 MN/m^2 Anhydrite has led to considerable damages to tunnel structures in the past, Figure 5.4, Wittke et al. (2017, p. 205), Wittke (2014, pp. 753 sqq.), and Amstad and Kovári (2001, pp. 78 sqq.).

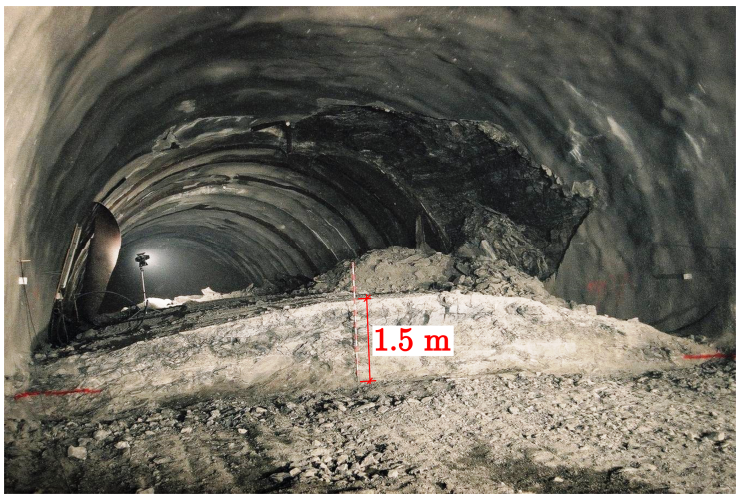


Figure 5.4: Ground uplift during the construction of the Chienberg road tunnel as a result of the swelling of Anhydrite. Photo courtesy of Aegert Bosshardt AG, Aegerter & Bosshardt (2019).

.

A principal sketch of the Gipskeuper formation is shown in Figure 5.5. The formation can be coarsely divided into a leached (upper) and an unleached (lower) zone. These two zones are divided by the leaching front. In the leached zone, the reaction from Anhydrite to gypsum has taken place during geological times. The bottom part of the leached zone's gypsum has been partly washed out due to groundwater flow, leaving water channels and karstic holes behind. As a result, the upper part of the leached zone's permeability is significantly lower than in the lower part. In the leaching front, the sulfate content dissolved, EQ 5.1, and the transformation to gypsum, EQ 5.2, takes place. The unleached zone contains Anhydrite that has not yet reacted; see Wittke et al. (2017, p. 205) and Sabew et al. (2019b, pp. 172 sqq.).

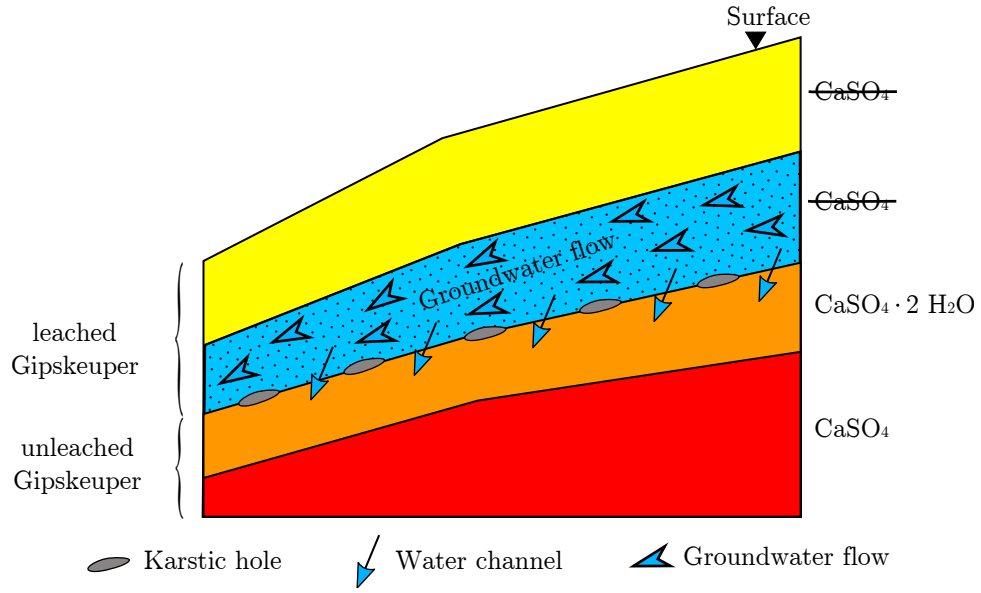


Figure 5.5: Gipskeuper, principle sketch. Four zones are distinguished. The upper two are part of the leached and the lower two of the unleached Gispkeuper. The uppermost zone (yellow) is exposed to the weather. Hence it does not contain Anhydrite. The second zone (blue) contains groundwater, which has leached the Gipskeuper of all Anhydrite. That groundwater enters the third zone (orange) and reacts with the Anhydrite. The fourth zone (red) had not yet been in contact with the water and contains most Anhydrite.

5.1.2 Anhydrite Concept

In order to counter the detrimental influence of Anhydrite swelling, the planning company, WBI GmbH, developed a novel Anhydrite concept with the primary goal to deny the Anhydrite containing unleached zones any inflow of water; compare Wittke et al. (2017, 206psq), Sabew et al. (2019b, pp. 172 sq.), and WBI (2019). In particular, four measures have been implemented:

1. *Sealing structures:* Seepage flow through the disturbed rock zone parallel to the tunnel structure is countered by so-called sealing structures. These are made of two concrete rings with a thickness of about 1 m and a length of 4 to 5 m. Chemical injections with TPH Pur-O-Stop FS-L 60 (TPH Bausysteme, 2019a), a *Polyurethane* (PU) synthetic resin, further seals the rock adjacent to the sealing structures, Figure 5.6.
2. *Dry tunneling and advanced grouting:* Advanced grouting with PU is used to seal the highly permeable zone of the leaching front, Figure 5.5. This method enables the TBM to pass through the leaching front without using any water, Figure 5.7.
3. *Radial grouting:* After completion of tunneling, selected zones of high permeability, Figure 5.8 b, are being sealed with a low-viscous chemical grout to prevent the inflow of water, Figure 5.8 c, and resulting swelling of Anhydrite, Figure 5.8 c. The applied grouting agent is the *Acrylate* (AC) gel: TPH Rubbertite B 40g, TPH Bausysteme

(2019b). These AC injections are supplemented by additional injections with PU agent.

4. *Adaption of profile cross-section:* Two cross-sections are being used for the tunnel structure to minimize the disturbance, which would create additional gaps in the rock, through which water might reach the unleached Anhydrite. (1) A circular cross-section is chosen in areas that are sufficiently distant from the leaching zone. (2) In areas in which the Anhydrite is located in the lower part of the cross-section, the tunnel cross-section takes a U-profile shape. The U-profile is supposed to minimize the inflow of water coming from above.

The injection project investigated in the framework of this study is constructing measure 3.

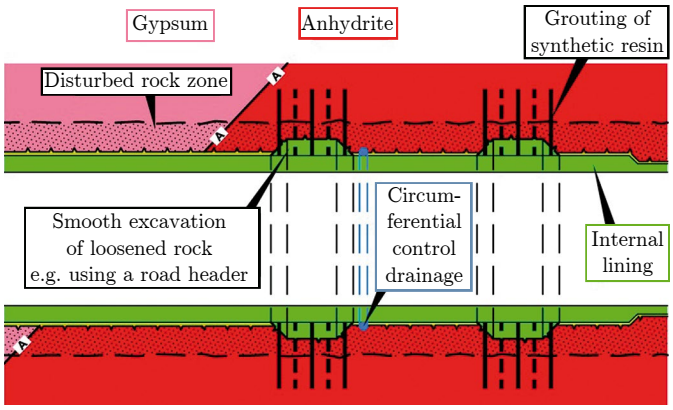


Figure 5.6: Tunnel Feuerbach, sealing structure. A combination of sealing structures and synthetic grouting is used to prevent water seepage. Control drainages are installed to detect possible failure of the implemented measures. Figure adapted from Wittke et al. (2017, p. 206, Fig. 6).

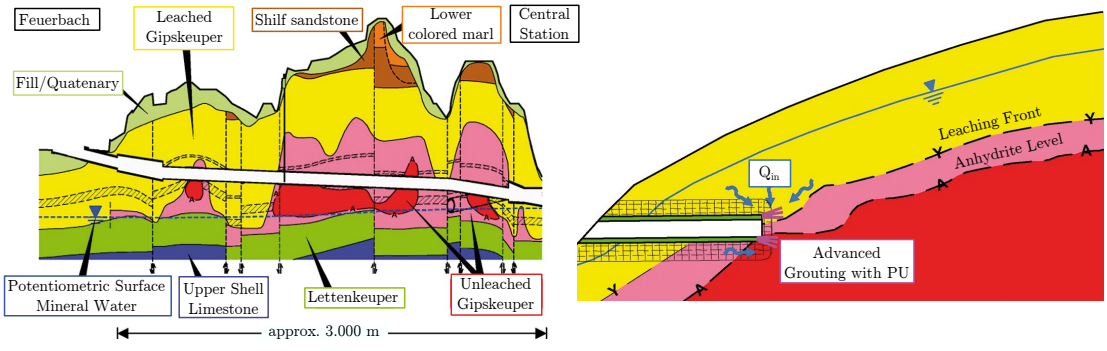


Figure 5.7: Tunnel Feuerbach, section cut. The tunnel leads through different rock formations, of which the unleached Gipskeuper must be denied the contact with water (left). Advance injections with PU agent are performed to prevent the ingress of water (right). Figure adapted from Wittke et al. (2017, p. 207, Figure 8, 9).

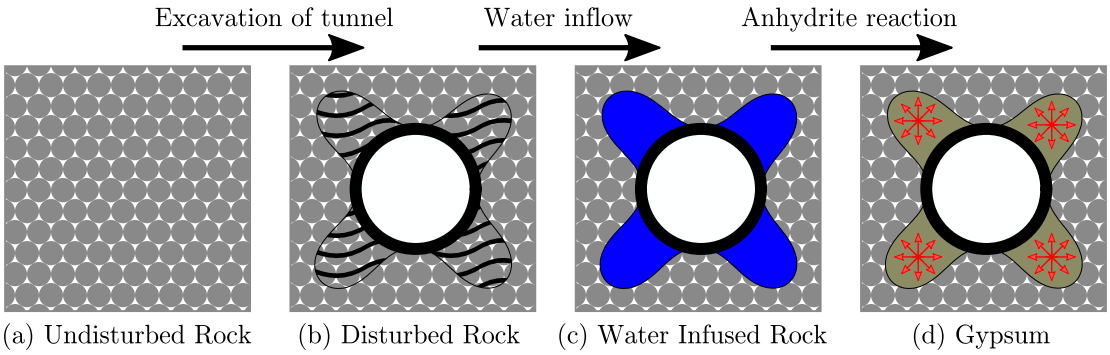


Figure 5.8: Swelling process after tunnel excavation. Tunnel excavation in undisturbed rock (a) leads to zones of disturbed rock (b) which allow the inflow of water (c) and consequently start the Anhydrite swelling reaction.

5.1.3 Process of radial injection grouting (measure no. 3)

In the following, the injection concept of the radial injections is summarized. A detailed description is presented in Sabew et al. (2019a,b) and Lienhart et al. (2019). A more generic description of the injection process is presented in Singh (1993, pp. 620, 724 sqq.).

Injections are made in radial, pre-made boreholes, which have been produced by a bore jumbo with air purge, Figure 5.9. In the process chain of tunnel construction, the drilling process is followed by the injection work, which precedes the internal formwork. The total

Figure 5.9: Exemplary picture of drilling jumbo. The drilling jumbo produces the boreholes, which are later being injected with chemical grouting material. Picture by Ellgaard (2009).

length of the tunnel section which is injected is about 3 km. The boreholes have a diameter of about 51 mm, a depth of up to 10 m, and are arranged in a grid of approximately 1 m×1 m. The injection sequence is determined by a complicated pilgrim procedure consisting of up to 12 steps during which the injection unit passes each borehole 11 times. A total of three

injections have been planned per borehole. A principal sketch of the procedure is shown in Figure 5.10. At first the boreholes of row *a*, then row *b* and finally row *c* are being injected. Each time the cross-section I to IV are being processed one after another. The planning company uses measurements from the drilling and the injection works to decide how to proceed during the upcoming phase.

To prevent that the rock fractures as a result of an injection, its pressure is limited to 5 bar. To assure infiltration of the injection agent at such low pressure an AC gel with a low viscosity of about 3 mPa · s is used, Lienhart et al. (2019, p. 462). The AC injections are carried out using pneumatically tensioned packers with ventilation; see also Hemphill (2012, pp. 223 sqq.). The packers have been designed to be reused after going through a cleaning process in a designated on-site washing zone.

The injection process stops once an injection volume of one liter over a holding time of 10 minutes is not exceeded. In contrast to pure flow-depended criteria, this method ensures that small peaks in the measured injection flow, i. e., due to shifting of the packer in the borehole, do not lead to a prolongation of the holding phase.

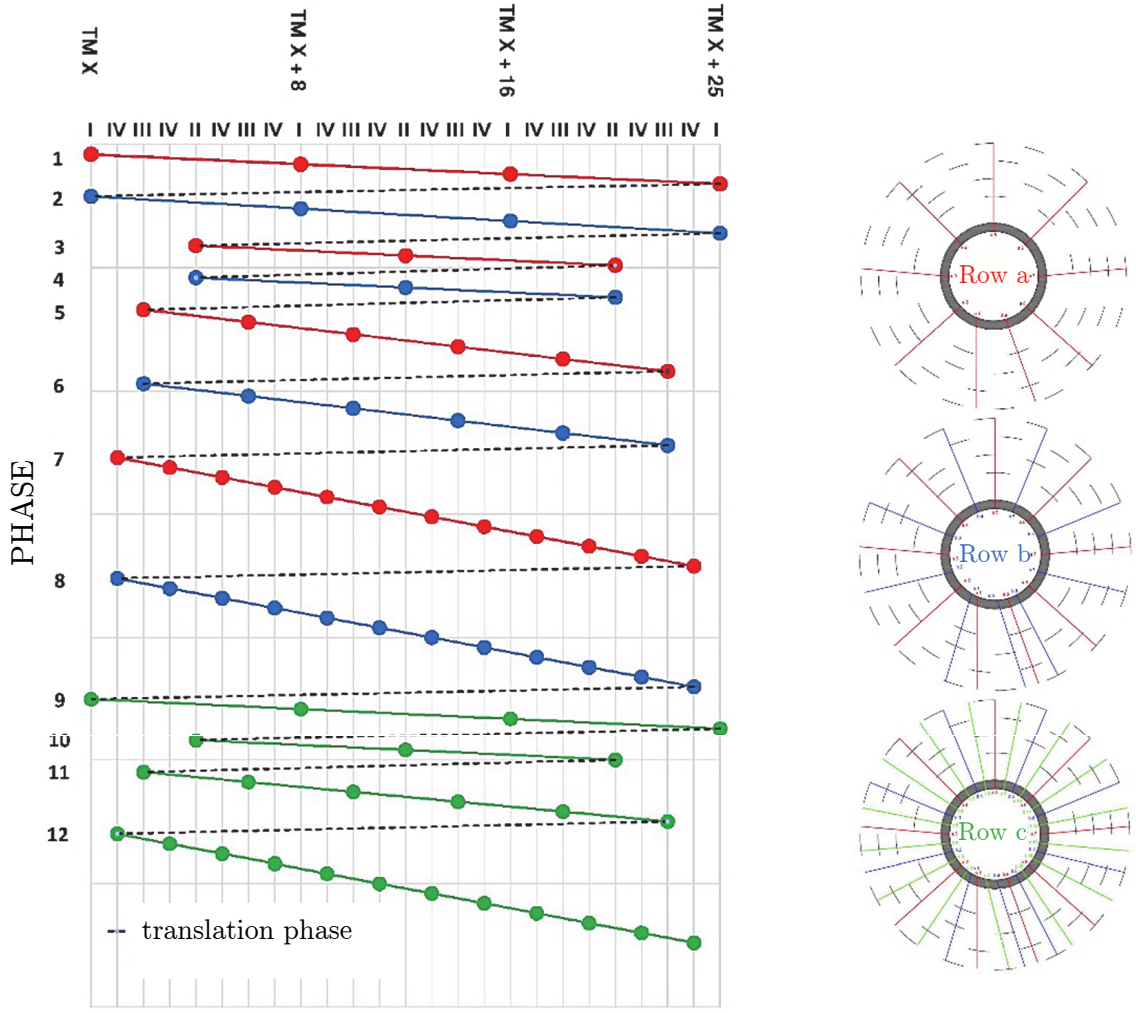
The venting and flushing process for the AC gel injection is integrated into the control loop. The former ensures that no air is being enclosed in the rock body. The latter protects the injection system from an inevitable denaturation of the injection gel. With a denaturation of approximately 10 minutes, it would otherwise clog the injection lines.

PU has a significantly higher viscosity of 150 mPa·s, which would result in higher pressures at the same injection rate. To ensure compliance with the 5 bar injection criterion, the injection computer dynamically regulates the injection rate based on flow resistance in the grouting line.

The injection fluids AC and PU are mixed from two components. The correct mixing ratio of these components is assured by continuously measuring their flow rates directly at the blender head, attached to the packer. A principle sketch of the digital system for measurement and control of the AC injections is shown in Figure 5.11.

One equipment unit, i. e., *Geräteeinheit Acrylat* (GEA), consists of up to three pumps installed together with the control technology in a container. The container itself is mounted on a truck. Such a GEA can be moved freely within the tunnel. Part of the GEA production system, Figure 5.12, are up to three Manitou lifting platforms, via which workers place the packers in the boreholes.

The pump operator starts and monitors the injection process from the control container and carries out quality assurance measures such as monitoring the multi-component injection agents' mixing ratio. The system automatically logs essential control parameters, such as pressure and flow rates, and each of the pumps process steps.



(a) Movement in tunnel section

(b) Movement in tunnel cross-section

Figure 5.10: Principle sketch of the pilgrim method of injections. In sub-figure (a) the abscissa shows the longitudinal location in the tunnel, and the ordinate counts the injection phases. Colors reference to sub-figure (b), which shows the radial location of injections. The dashed line in sub-figure (a) indicates the translation of the injection unit, i. e., it has to move back as part of the pilgrim method. Figure adapted from Sabew et al. (2019b, p. 174, Abb. 4).

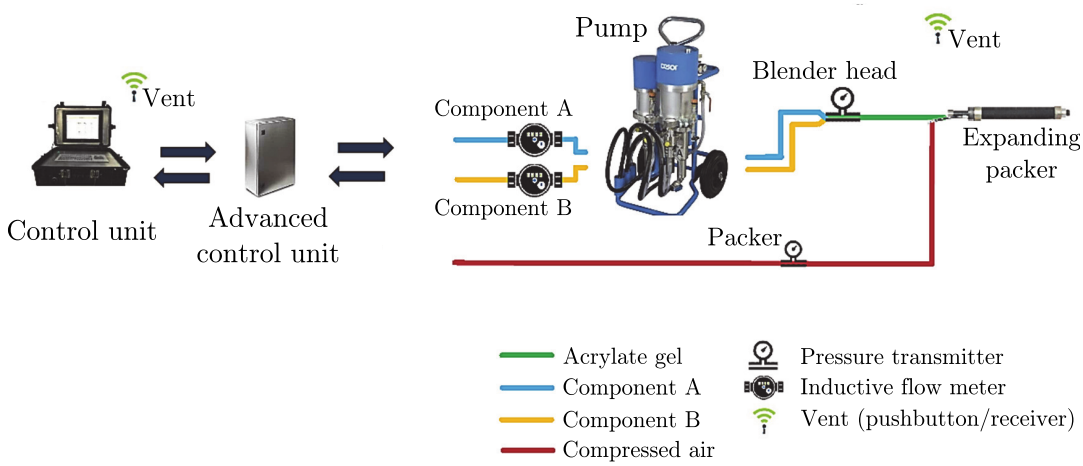


Figure 5.11: Digital system for measurement and control of AC injection grouting. The human-machine interface is realized via a laptop, connected with the advanced control unit. The two chemical components are pumped separately to the blender head, where they are mixed to form the injection agent. Each component's flow is logged with inductive flow meters. The agent is pumped through an expanding packer into the rock. The packer locks itself to the rock by being inflated with compressed air. The worker who operates the packer must vent the borehole before the injection process starts. He indicates the end of the venting process by pressing a button connected to the control unit. Figure adapted from Sabew et al. (2019b, p. 177, Abb. 7).

Figure 5.12: Production System. Depicted are the drilling unit (top, left), the truck mounted GEA container (middle) and a lifting platform (bottom). Blue canisters contain a chemical component for the grouting agent. Boreholes in the tunnel wall have been marked with colored spray paint.

5.1.4 Digitization of Injection Process

With over 500 injections per day being carried out by several AC and PU injection units, which again include up to three pumps, documentation by hand is not an option. Process parameters must be documented continuously from all units. They must be available without delay, 24 hours a day, and seven days per week, to enable decision-makers to provide instant feedback if changes of the planned procedures occur. The construction manager's decisions must be immediately available for the injection teams in the tunnel. For this, the interfaces must provide an exchange point for data to other IT systems, such as *Enterprise-Resource-Planning* (ERP) and BIM solutions, e. g., iTwo (RIB Software SE, 2019) or GBPlan.

These specifications are met by the eguana-System, which logs all building parameters and instantly mirrors them via the construction site's *Wireless Local Area Network* (WLAN) or public mobile network to the eguana server. The data is immediately available through the eguana WebPlatform eguana-SCALES. The data management system visualizes pressure- and flow rates of individual injections and provides a comment function for special events documentation. The system automatically compiles daily injection protocols. eguana SCALES provides an interactive visualization of the status of the construction site on several *Level of Detail*s (LODs), such as a construction site overview (map and sketch), the status of individual tunnel sections, Figure 5.14, and detailed information plots of single injections. Color codes provide information on the status of each injection. For feedback construction managers can upload data sets of injection specification to the system, which are immediately available for machine operators. The system further provides a documenta-

Figure 5.13: Screenshot of the visualization of processes with eguana-SCALES, eguana (2019). Each of the two diagrams shows the documentation of one pump's activities during the project. Depicted are five days during which pump GEA 2/P1 (top) performs seven and pump GEA 3/P1 (bottom) eight different types of processes.

tion of construction processes. Operators select from a list of around 20 items for standard processes, e. g., injection, machine movement or various maintenance and idle processes. Figure 5.13 shows a visualization of the process documentation by eguana-SCALES.

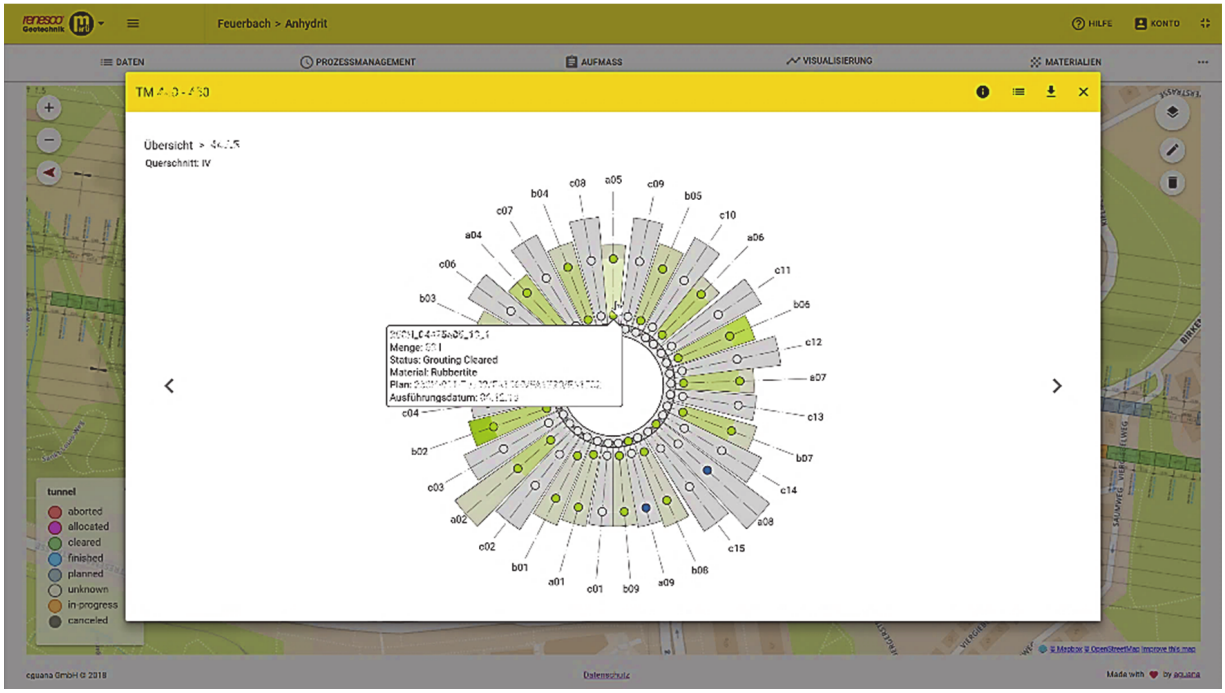


Figure 5.14: Visualization of tunnel section-cut with eguana-SCALES. Each of the boreholes is color coded. Meta information, such as the injected volume of grouting agent, can be retrieved with a click on the corresponding borehole. Figure adapted from Sabew et al. (2019b, p. 179, Abb. 10).

With this system, complete documentation of the injection pumps' activities throughout the entire project period is being performed and made available for external tools, i. e., via pre-defined interfaces, such as a csv-file for GBPlan. Figure 5.15 shows a zoom on the status map of eguana-Scales. The tool draws the tunnel's sections with their correct position on a street map. A section's status is given via color-coding. Additional information, such as its unique identifier or length, is provided as text directly on the map.

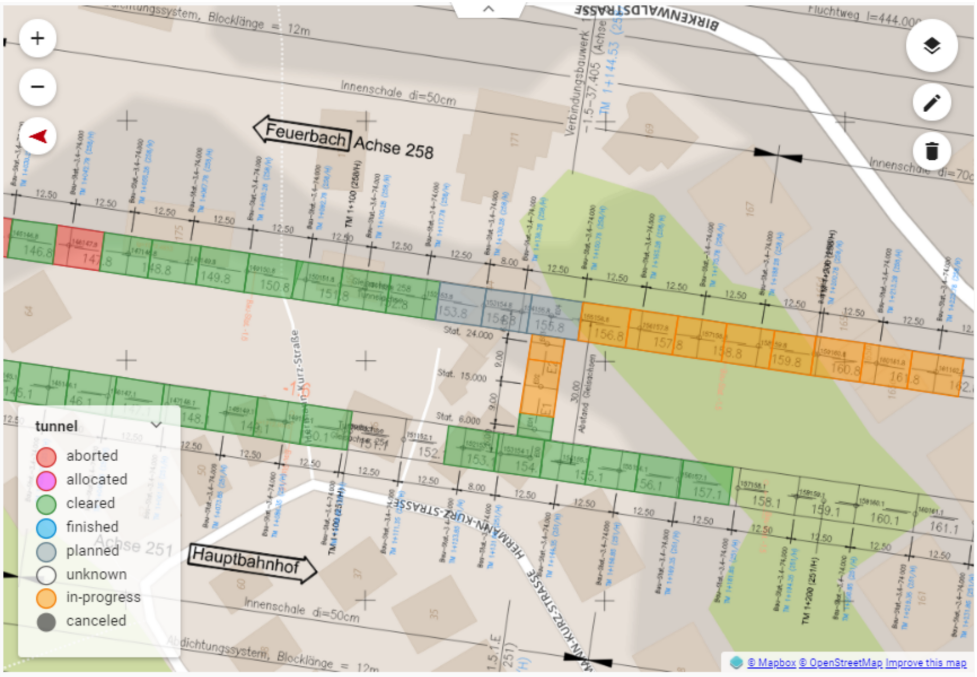


Figure 5.15: Screenshot of eguana Scales web front-end. Each section's location is provided via a zoomable street map, and its status is given in different colors. Additional information, such as the sections' unique identifiers and length, is provided as text directly on the map. Picture courtesy of eguana GmbH.

5.2 Selection of Input Data

The eguana cloud contains many different data sets reaching from start and end times of processes, over data collected from pressure valves and pumps to daily, handwritten logs. The items are all linked to a discrete process, which has been performed by injection pumps in the tunnel. From the vast amount of data, a set of nine items was selected as input for GBPlan. This data set is transferred from the eguana cloud to GBPlan via download of a *Comma Separated Value* (csv) file. The file is build as a $[n \times 9]$ table encoded in the *8-Bit Universal Coded Character Set Transformation Format* (UTF-8). Each row of the table gives a description of what one of the pumps in the production system did during a certain time interval. An excerpt of the file is shown in Figure 5.16. All n rows together provide a seamless documentation of all processes performed by all pumps in the production system. The columns of the table are labeled in German language and contain information on:

`start` (start) The start time of the process in the format [*dd.mm.yyyy hh:mm:ss*], e. g., $s =$ 26.09.2017 11:17:20.

`ende` (end) The end time of the process in the format [*dd.mm.yyyy hh:mm:ss*], e. g., $e =$ 26.09.2017 11:30:54.

`dauer` (duration) The duration of the process in hours as decimal number, e. g., duration $d = e - s = 13.56$.

```
"start"; "ende";"dauer";"tätigkeit";"maschine";"menge";"name";"kommentar";"status"
"2018-07-02 20:13:11";"2018-07-02 20:20:18";"0.119";"UP: Umsetzen IP";"GEA 3/P2";"";...
        ...""";"";""
"2018-07-02 20:20:18";"2018-07-02 20:22:25";"0.035";"IJ: Injektion";"GEA 3/P2";"10";...
        ..."251H_10350AL01_08_1";"Auslitern mit 10l";"I: Injektionskriterium erreicht"
"2018-07-02 20:21:42";"2018-07-02 20:27:40";"0.099";"IJ: Injektion";"GEA 1/P1";"4";...
        ..."251H_07060c08_01_1";"Bw - Bohrung feucht/wasserführend";...
        ..."I: Injektionskriterium erreicht"
"2018-07-02 20:22:25";"2018-07-02 20:27:32";"0.085";"UP: Umsetzen IP";"GEA 3/P2";...
        ...""";"";"";""
                                    ⋮
```

Figure 5.16: Excerpt of input csv-file downloaded from the eguana cloud. Each line documents one instance of a process with up to nine variables. The variables come as strings in UTF-8 and are separated by semicolon. Three dots "..." indicate a line break, which has been added to this figure for better readability.

tätigkeit (activity) The process type as string value. A total of 20 process types exists. These are summarized in Table 5.3.

maschine (machine) The machine which performs the process as a string value.

menge (amount) If a process involves the injection of grout material, the volume of the injected material is given in deciliter. If no flow of grout material has been detected, the value is empty.

name (name) The name or location identifier of an injected borehole. The location is only logged at discrete boreholes, i. e., no location information is logged when the pump is located between boreholes. Consequently, the value contains a string value for processes that perform an injection and is empty for the remaining process. A description of the location identifier string is provided in Table 5.1.

kommentar (comment) The comment column contains comments on injections. While some entries follow a standardized pattern, many have been entered manually. An example of those entries is: "Auslitern 10l" and "Auslitern mit 10l". Both inform that an injection has been performed as part of a maintenance process and that 10 liters of injection grout had been used during that process.

status (status) The final status of the injection. Three types of final statuses are logged:

- **A: Abbruch** (abort) Informs that the injection had been aborted.
- **I: Injektionskriterium erreicht** (injection criteria achieved) Informs that the injection criteria had been reached.
- **S: Störung Technik** (technical fault) Informs that a technical problem or malfunction had been identified during the injection process.

Table 5.1: Description of the input variable “name”, which serves as the unique location identifier of injections.

Character	1–4	6–10	11–13	15–16	18
Example	`258H`	`05500`	`a01`	`20`	`2`
Description	Name of tunnel tube	Position of section in tunnel tube [*dm*]; see Figure 5.10	Position of borehole in section; see Figure 5.10	Position of packer in borehole measured from borehole entry to packer [*dm*]	Number of previous injections +1

5.3 Data Cleaning

Data cleansing, i. e., the process to find and correct errors in a database, is crucial to produce sound computation results, Maletic and Marcus (2005, p. 22). Although the data set provided by the eguana cloud has already been cleansed in order to meet the requirements of the eguana SCALES software, further cleansing is required to make it usable for GBPlan as well. GBPlan follows the three-step approach outlined in Maletic and Marcus (2005, p. 26) as (1) define and determine error types, (2) search and identify error instances, and (3) correct the uncovered errors; see also Runkler (2015, pp. 2 sq.).

In GBPlan, the routines used for the data cleansing are implemented in the function `GBPreProc.preProcessingStruct` and executed when the eguana input is parsed from its raw data representation to the GBPlan internal `structData` data structure. Before the cleaning process starts an array list P_j of processes $p_{i,j} \in P_j$ is being computed for each pump j as

$$p_{i,j} = \{\text{process name}, \text{duration}, \text{start time}, \text{end time}, \text{location id}, \text{grout volume}\}\,. \quad (5.3)$$

Each list P_j contains only the processes $i = \{1, \ldots, n\}$ which are performed by pump j. The list is sorted according to the start date of the processes so that the process $p_{i,j}$ with the earliest start date can be found at position $i = 1$. The types of cases cleansed from the data set, and the correction process are explained in the following sections.

5.3.1 Merging of Split Processes

If two consecutive processes p_i and p_{i+1} are of the same type, i. e., their name is equal, then it is checked if the end time of the first process equals the start time of the second. If so the two processes are being merged to one process p_{merged}. p_{merged} has the start time of p_i, the end time of p_{i+1} and therefore the duration $d(p_{merged}) = d(p_i) + d(p_{i+1})$; see Figure 5.17.

If the processes are injections, then the grout volume g is summed up as well, i. e., $g(p_{merged}) = g(p_i) + g(p_{i+1})$. Then p_i and p_{i+1} are removed from the list of processes and p_{merged} is added to the same. Note that merging processes leaves a gap in the list of processes P. Given that p_{merged} is added to position i, then all processes $p_j \in P$ with $j > i + 1$ must be shifted by one entry, i. e., $\forall j > i + 1, j_{\text{new}} = j - 1$. After the shifting the last entry $p_n \in P$ is empty and must be removed.

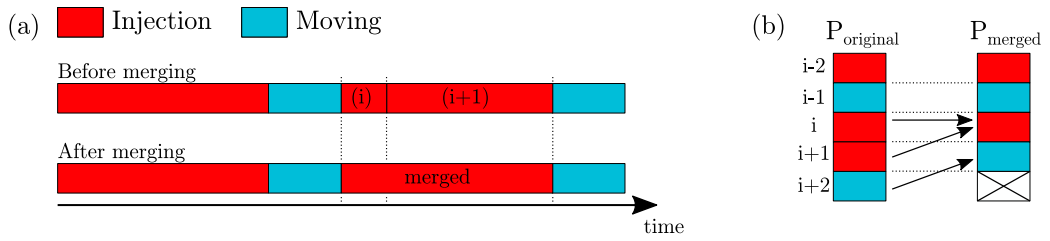


Figure 5.17: Merging of processes during the data cleansing process. Depicted is how the internal data structure, in particular arrays storing process information, must be adapted as result of the merging of two processes.

5.3.2 Time Gaps

The list of processes P_j contains many time gaps which the eguana SCALES system interprets as "`KS: Keine Schichtbelegung`", i. e., unspecific idle time. In GBPlan, concrete processes are assigned to these gaps, Figure 5.18 a. These are "`HOLY: HOLIday or sundaY`" for gaps due to weekends or public holidays, i. e., planned, recreational time. All other gaps are assigned to the process type "`UPT/GAP: Undocumented Process Time`". When adding a new process p_{new} to position i of P all processes $p_{j \geq i}$ have to be moved one position down the list, i. e., $\forall j \geq i, j_{\text{new}} = j + 1$, so that $p_i = p_{new}$, Figure 5.18 b.

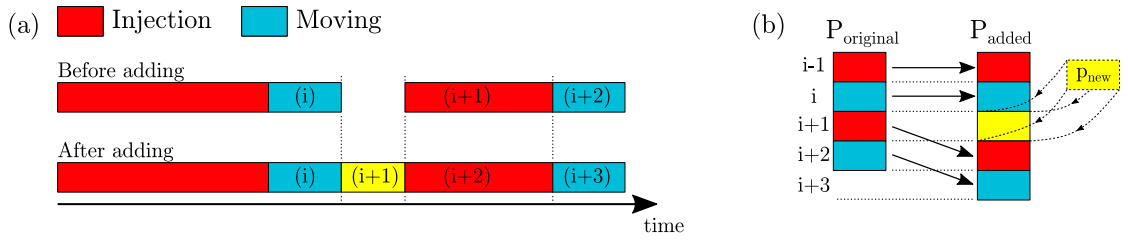


Figure 5.18: Adding of the new process p_{new} during the data cleansing procedure. When adding a process to the array of processes, all processes located after the additional process must be moved by one element.

Whether or not a gap is tagged as "`HOLY: HOLIday or sundaY`" depends on the day when it starts, when it ends, and a hardcoded list H of holiday time intervals $h \in H$ which is returned by the function `GBStatic.get_holidays`. Gaps $g_{i,j}$ which have their start g_s at the end time of process p_i and their end time g_e at the start date of process p_j are tagged "`HOLY: HOLIday or sundaY`" if

- g_s is during a Friday and g_e is during a Monday,
- g_s is during a Saturday and g_e is during a Monday,
- g_s is during a Sunday and g_e is during a Monday, or

or if the start or the end of the gap is during a time interval of H, such that

$$(g_s \cap H) \cup (g_e \cap H) \neq \varnothing. \tag{5.4}$$

5.3.3 Very Short Processes

If a process p_i has been logged with an unrealistic short duration $d_i < 1$ s, GBPlan informs the user by throwing a warning message. The warning is issued because injection or moving processes of less than a second seem not plausible, are most likely an error, and would bias the statistic if not removed or corrected. The rule applied by GBPlan is

$$d_i < 1 \text{ s} \begin{cases} \text{if true,} & \text{then GBPlan throws a warning to the user, and} \\ \text{if false,} & \text{then GBPlan does nothing.} \end{cases} \tag{5.5}$$

5.3.4 Adding and Correcting Location Tags

All injection processes are assigned to a borehole via the processes' location tag. This tagging includes those boreholes used in the calibration process described in section 5.3.5. In order to link the non-injection processes to a location, the following rule is applied: All processes which precede an injection are assigned to the borehole of that injection. The idea behind this rule comes from Lean Management, which divides processes into value-adding and non-value-adding processes. The injection is a value-adding process, and the processes preceding the injection, e. g., movement or maintenance, are not value-adding; however, they are necessary for the value creation. Because they are necessary to create value at the borehole, they are assigned to that borehole. Note that this rule did not exist throughout the construction work, which is why some non-injection processes are labeled according to a rule set that remains unknown. Therefore, location tags of non-injection processes are deleted from the data before their proper reassignment.

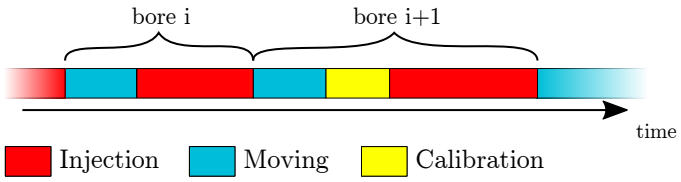


Figure 5.19: Assignment of the location identifier during the data cleansing process. The location identifier assignment follows the logic that all processes that happen before an injection occurs are support processes, i. e., they are required to perform that injection. Hence, they are associated with the borehole of that injection.

5.3.5 Identification and Labeling of Maintenance Processes

The calibration process "Auslitern" is logged as an injection process, i. e., with the process name "`IJ: Injektion`". Because GBPlan handles the system's calibration as an individual process, its process name is changed to "`WR: Wartung/AL-RM`". The challenge is to identify the calibration processes as the same, which is done as follows.

The calibration can be distinguished from the standard injection via its location string ID, which contains a string $s \in S$. Unfortunately, ID_i of the process p_i is entered into the system by hand, which is why many different strings $s \in S$ are used to denote calibration

$$S = \{\text{“}\texttt{AL}\text{”}, \text{“}\texttt{rm}\text{”}, \text{“}\texttt{tes}\text{”}, \text{“}\texttt{FA}\text{”}, \text{“}\texttt{fa}\text{”}, \text{“}\texttt{Fa}\text{”}, \text{“}\texttt{RM}\text{”}, \text{“}\texttt{Rm}\text{”}, \text{“}\texttt{LA}\text{”}, \text{“}\texttt{La}\text{”}\}\,. \tag{5.6}$$

GBPlan applies the rule

$$(ID_i \cap S) \neq \varnothing = \begin{cases} \text{true}, & \text{if } p_i \text{ is a calibration process} \\ \text{false}, & \text{if } p_i \text{ is not a calibration process} \end{cases} \tag{5.7}$$

to distinguish injections from calibrations.

5.3.6 Identification and Labeling of IDLE Processes

In the analyzed production system, most processes' occurrence and duration are either aleatory or epistemic, i. e., they follow a stochastic rule, which we know, albeit some parameters of that rule might be unknown. Then there are ontological processes that are outside of our horizon and therefore very hard if not impossible to estimate. Usually, the reason is that they originate from outside the production system. For instance, the drilling process has not been observed. If the drilling team is slow and does not produce enough boreholes, the injection units are idle. The waiting process resulting from such an event is logged as `StB: Stillstand Bohrungen`. Other examples are periods of downtime due to `StP: Stillstand Entscheidung Planung`, which is a standstill because a management decision or plan is missing. Or `GV: Gerätevorhaltung GEAc`, the provision of machinery, e. g., as a reserve. These processes have in common that they cause idle times for the machines. To be able to compare simulation results with measurements, ontological

processes are treated differently from the aleatory and epistemic process; see also section 6.1.1.7. To quickly identify the processes, they are grouped as 'excluded processes' and excluded from the standard way of forecasting. In a later step, the 'excluded' processes are included in the results in a way that allows a comparison of simulation results with measurements. The identifiers of the excluded processes are returned by the GBPlan function `GBStatic.get_stringsArray_excluded_Procs`, which are

- `GV: Gerätevorhaltung GEAc`,
- `GVA: Gerätevorhaltung GEAc-Abrechnung`,
- `KS: Keine Schichtbelegung`,
- `StB: Stillstand Bohrungen`,
- `StS: Stillstand Sonstiges`,
- `StP: Stillstand Entscheidung Planung`,
- `UPT/GAP: Undocumented Process Time`,
- `HOLY: HOLIday or sundaY`.

5.4 Transformation of Data

The primary data type used by GBPlan to store input data is the structured arrays or simply "structs". A struct groups related data using containers, which are called fields. Each field can contain any data, including arrays. The filed names of a struct can be accessed via dot notation such as in `struct.fieldname`. More details on structured arrays can be found in MathWorks (2019b). During the data import, two types of structs are being built. The first, `structData`, is centered on the machines' processes and build to provide information on what the smallest unit of the production system, the pump, is doing. The Second, `structBores`, is tube centered and provides information based on the smallest unit of a tube, the borehole.

5.4.1 The Data Structure: `strucData`

The `structData` focuses on the processes of a pump. One can think of it as a $m \times n \times q$ field f with m injection pumps and n types of processes per pump. Each field $f_{m,n}$ contains a list of q processes $p_q = \{\text{name}, \text{duration}, \text{start}, \text{end}, \text{location}, \text{grout}\}$. Note that q may differ from f to f depending on the number of processes stored in that field. Figure 5.20 shows principle sketch of `structData`. The elements of p correspond to the input data described in section 5.2, however, have been translated to English language according to Table 5.2.

```
structData:       [1x13] struct
  |--name:        [1x1]  string
  '--pArray:      [1x20] struct
     |--name:     [1x1]  string
     |--duration: [qx1]  duration
     |--start:    [qx1]  datetime
     |--end:      [qx1]  datetime
     |--location: [qx1]  string
     '--grout:    [qx1]  double
```

Figure 5.20: Design of the data structure `structData`. The example shows a `structData`, which comprises of 13 pumps. The processes logged for each pumps are stored in the sub-data structure `pArray` (the abbreviation for 'process array') with 20 different process types each. n processes of each process type per pumps are stored in `pArray`. The column to the right shows the respective data types. For details on data types see MathWorks (2019b).

Table 5.2: Name conversion from `rawData` to `structData.pArray`. A definition of the elements of the data structure `rawData` can be found in section 5.2.

`rawData`	`structData.pArray`
tätigkeit	name
dauer	duration
start	start
ende	end
name	location
menge	grout

5.4.2 The Data Structure: `strucBores`

The data structure `structBores` focuses on the boreholes in the tube. Each element of `structBores` stores the information for one borehole, i. e., the example given in Figure 5.21 models a tunnel tube with $i = 21{,}972$ boreholes. The field `location` contains the position of the bore in the tube, e. g., `03340c02` for the borehole located at position c02, compare Figure 5.10, in tunnel section 3340. The other fields are $1 \times m$ arrays and store data on the m processes associated with the borehole. Note that a different number of processes may be executed at different boreholes; hence m may vary between different boreholes i.

```
structBores:        [1x21972] struct
  |--location:      [1x1] string
  |--procName:      [1xm] string
  |--start:         [1xm] datetime
  |--end:           [1xm] datetime
  |--duration:      [1xm] duration
  |--ID:            [1xm] string
  |--machine:       [1xm] string
  '--grout:         [1xm] double
```

Figure 5.21: Design of the data structure `structBores`. The struct sets its focus on the tube. Sorted according to its position in the tube each element of `structBores` represents a borehole and stores data on processes connected to that borehole.

5.5 Data Mining

The project data set starts 26.09.2017 at 11:17 with the test injection `258H_MPtest01_0_0` by the injection pump GEA 1/P1. During the injection, the packer slipped from the borehole after 6.8 liters of grouting material had been injected. Two more trials were required to finish the injection of the borehole with success. The last process logged is the shifting of the injection nozzle of PU Station 1/P3 on 08.11.2019 05:23. During the roughly two years passed between the first and last process stored in the eguana database, over 300,000 process events were logged, about 150,000 injections performed, and over 1.78 million liters of AC and PU grouting material injected into the rock.

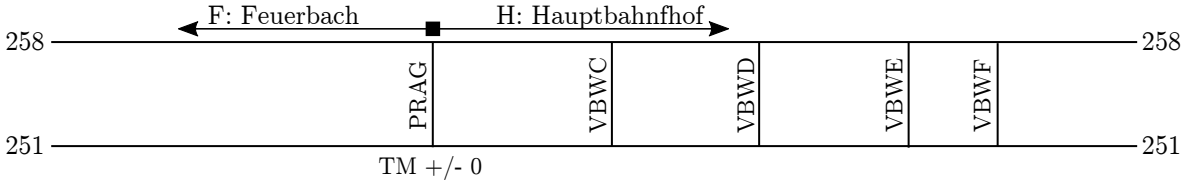
This section provides a detailed insight into the data used within the framework of this thesis and serves as a basis for the following forecast simulations of grout volume, project duration, and project cost.

5.5.1 Axes and Tunnel Tube Identifiers

The tunnel displayed in Figure 5.22 (a) consists of two parallel tubes along the axes 258 and 251. These axes' origin is set at the entry point for the construction works named 'Zwischenangriff Prag' or just PRAG. Starting from PRAG, each tube is divided into two parts. One leads in the direction of the train station Feuerbach (F), and the other, leading to the central railway station of Hauptbahnhof Stuttgart (H). These four primary tubes, 251H, 258H, 251F, and 258F, are interconnected with five connection buildings, "VBW" (abbreviation of the German term "Verbindungsbauwerk", which translates to "connection building") C to F. As a result, the data set allows the assignment of nine tubes, which together form the Feuerbach tunnel; see Figure 5.22 (b).

(a) Site sketch of Tunnel Feuerbach

(b) Nomenclature



VBW: Verbindungsbauwerk (Connection Building) C to F
TM: Tunnel meter
251/258: Tube axes

Figure 5.22: Two sketches of Tunnel Feuerbach, (a) as provided by the construction site, and (b) as a simplified model consisting of 9 interconnected tunnel tubes. The two axes, 251 and 258, are divided at the construction site's entry point (PRAG) in the northern F-tube (direction Train Station Feuerbach) and the southern H-tubes leading in the direction of the train station Hauptbahnhof. Five connection buildings (PRAG and VBW C–F) connect the two main tunnel tubes. Subfigure (a) adapted from Erichsen (2017).

5.5.2 Number of Instances and Total Durations of Processes

A total of 20 different process types has been logged since the start of the project. These are listed in Table 5.3. The table is sorted according to the total duration TD spend with each project type. It furthermore shows the number of instances n, i. e., how often a process of that type has been executed. From these values, the average duration $\bar{d} = TD/n$ is calculated. Although the injections are relatively short processes ($\bar{d}_{\text{injection}} = 0.33$ h, they take up a large part of the project time (about 25 percent). The three following items (SN 2-4) in the list are idle processes, during which no production occurs. HOLY, UPT/GAP, and GV sum up to about 45 percent of total project time. Items SN 5-6 support the injection, and their sum is roughly 14 percent of the total project duration. Item SN 7 (GVA) is, like HOLY, UPT/GAP and GV, and idle process and adds another 3.8 percent of idle time. From this, we see that the time spent on productive processes is relatively small. Hence, the hidden potential that can be tapped in through optimized utilization planning is considerable. Furthermore, we see that idle time tends to come in

big chunks, i. e., intervals of 22 to 49 hours (SN 2-4) while the processes which contribute to the project progress are relatively short (e. g., SN 1, 5 and 6).

The next section investigates how time is spent in the project, especially concerning productive and non-productive processes.

Table 5.3: Sum of duration and number of instances over all processes.

SN	Process	n	TD [h]	TD [%]	$\bar{d}$ [h]
1	IJ: Injektion	139,328	45,375	24.98	0.33
2	HOLY: HOLIday or sundaY	936	42,501	23.4	45.41
3	UPT/GAP: Undocumented Process Time	883	20,181	11.11	22.85
4	GV: Gerätevorhaltung GEAc	390	19,369	10.66	49.66
5	UP: Umsetzen IP	78,960	14,142	7.79	0.18
6	UPU: Umsetzen PU	61,969	9,575	5.27	0.15
7	GVA: Gerätevorhaltung GEAc-Abrechnung	196	6,868	3.78	35.04
8	SoA: Sonstige Arbeiten	1,285	4,923	2.71	3.83
9	StS: Stillstand Sonstiges	1,117	4,261	2.35	3.81
10	KS: Keine Schichtbelegung	431	3,793	2.09	8.8
11	UmA: Umsetzen Allgemein	3,053	3,533	1.95	1.16
12	StB: Stillstand Bohrungen	314	3,043	1.68	9.69
13	WR: Wartung/Reparatur	1,055	1,999	1.1	1.9
14	SPU: Stillstand PU	329	949	0.52	2.88
15	SKT: Störung / Kontrolle Technik	1,542	430	0.24	0.28
16	WR: Wartung/AL-RM	8,380	404	0.22	0.05
17	UT: Umsetzen Testphase	717	108	0.06	0.15
18	StP: Stillstand Entscheidung Planung	41	93	0.05	2.27
19	UPB: Umsetzen BV	263	47	0.03	0.18
20	UA: Unbekannte Aktivität	7	23	0.01	3.27

n: sum of instance, TD $= \sum_{i=1}^{n} d_i$: Total duration, $\bar{d}$: Average duration

5.5.3 Muda Analysis

Lean Construction is the transfer of the Lean Management philosophy to the construction industry. The authors of the bestsellers “The machine that changed the world” (Womack et al., 2007) and “Lean Thinking”, Womack and Jones (2013), describe the main principles with which the Lean Management approach pursues these goals as focusing on value for the customer, capturing the value stream, the flow principle, the pull principle and striving for perfection; see also Backhaus and Dahm (2020), Howell (1999), Ōno (2013), GLCI (2019), Womack and Jones (2013), and Koskela (1992). Through the consequent application of

these principles, one constantly improves in avoiding *Muda*, a lean management term coined by the Japanese engineer Taiichi Ōno, meaning waste, Ōno (2013). With this concept of waste, processes can be distinguished according their inherent types of Muda in 1.) value creating processes, 2.) necessary Muda and 3.) pure Muda. The muda-view comes with the advantage that instead of focusing on the improvement of the productive processes one strives to abandon waste or unproductive processes from the system. Because production systems tend to have a very high share of Muda the value creation can be influenced over-proportionally by adressing Muda. If for instance 95 percent of the processes of a production system were Muda and that share is reduced by just one percent the productive share rises from $(1 - 0.95) = 0.05$ to $1 - 0.95 \cdot 0.99 \approx 0.06$, i. e., the share of productivity rises by $(1 - 0.95 \cdot 0.99)/(1 - 0.95) = 1.19$ or 20 percent.

In the prevalent case the injection process is the only value creating process. If it were possible to reduce the work activity to this process it would be regarded as an optimum. There are, however, support processes which need to be performed, the necessary Mudas. For instance the process of moving the injection machine to the next injection hole and positioning it in accordance with the construction plan are examples for processes which are necessary but do not produce direct value for the customer. Finally, processes exist which neither produce value nor are they indirectly supporting the value creation. A prominent example is waiting time, which can be considered as pure waste, i. e., pure Muda.

At the level of discretization provided by the eguana data base, only the injection itself is a value creating process. The remaining processes can be divided in necessary waste and pure waste processes. Table 5.4 gives an overview of the categorization of the processes. When comparing the number of hours spend with each of the three types of Muda it can be seen that, based on the assumption that the injection process itself does not include further Muda, around 25 percent of the work time is spend with value creation. The remaining 75 percent belong to one of the two waste types, Figure 5.23.

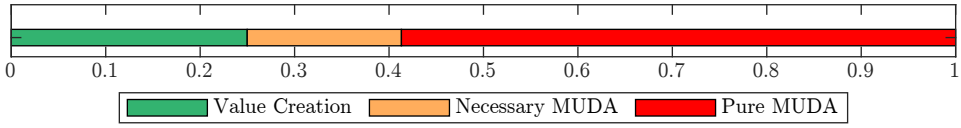


Figure 5.23: Overall share of each of the three types of Muda. (1) Value creation, (2) Necessary Muda and (3) Pure Muda.

Table 5.4: Mapping the processes to their types of Muda. Only one type of process is value creating. The remaining processes are either necessary or pure Muda.

		Muda Type		
SN	Process	1	2	3
1	IJ: Injektion	X		
2	HOLY: HOLIday or sundaY			X
3	UPT/GAP: Undocumented Process Time			X
4	GV: Gerätevorhaltung GEAc		X	
5	UP: Umsetzen IP		X	
6	UPU: Umsetzen PU			X
7	GVA: Gerätevorhaltung GEAc-Abrechnung			X
8	SoA: Sonstige Arbeiten			X
9	StS: Stillstand Sonstiges			X
10	KS: Keine Schichtbelegung			X
11	UmA: Umsetzen Allgemein		X	
12	StB: Stillstand Bohrungen			X
13	WR: Wartung/Reparatur			X
14	SPU: Stillstand PU			X
15	SKT: Störung / Kontrolle Technik			X
16	WR: Wartung/AL-RM			X
17	UT: Umsetzen Testphase			X
18	StP: Stillstand Entscheidung Planung			X
19	UPB: Umsetzen BV			X
20	UA: Unbekannte Aktivität			X

Muda types: (1) Value creation, (2) Necessary Muda, (3) Pure Muda

5.5.4 Distribution of Processes over a 24h Workday

A histogram of the number of active process instances per hour is shown in Figure 5.24 for six selected process groups. A drop in performance of the number of *Injection* processes can be observed around 06:00 and 18:00. The same is true for the *Movement of Tools and Equipment*, which comprises, in particular, in the movement of the injection nozzles between injections. The upcoming shift change can explain the performance dip at these times. Workers are preparing the injection units for the handover, which prevents them from doing injections and moving the equipment. Furthermore, the *Quality Assurance* measures, which are performed at the beginning of each of the two ten-hour shifts, bind resources that would otherwise be used to perform injections.

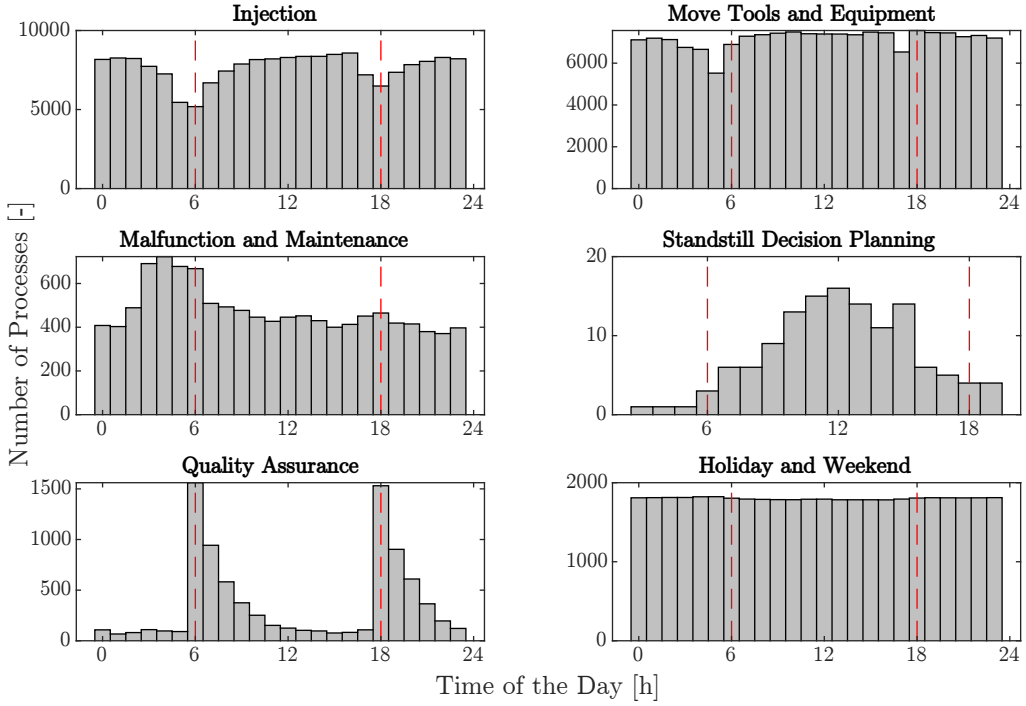


Figure 5.24: Time of day during which selected processes are being performed. The time of the shift change (6:00 and 18:00) is marked in dotted lines.

Most processes of the group *Malfunction and Maintenance* take place during the early hours around 3 o'clock. Furthermore, a slight peak can be observed around 13:00 and the late shift change at 18:00. These findings are in line with the results of Folkard and Tucker (2003), who investigate circadian performance rhythms in factories; see also Prinzhorn et al. (2017) and Seiwert (1984). Typically a performance minimum – and as such high rates of malfunctions due to errors – can be expected around 03:00, when worker's biorhythm leans rather towards rest than performance. The second dip, after 13:00, can be explained as a 'post-lunch' dip and reflects fatigue as a result of a heavy meal. Following Folkard and Tucker (2003, p. 96), the 13:00 performance dip could also be the result of a continually falling performance curve, which is superimposed by malfunction effects which decay from the 04:00 peak of the previous shift. Furthermore, the 06:00 and 18:00 peaks can be explained by a steep rise in errors after nine hours of on-duty, Folkard and Tucker (2003, p. 98).

The curve for *Standstill Decision Planning* starts to rise around the start of the day shift at 06:00, has its peak around noon, and declines from there until the end of the day shift at 18:00. Two effects can explain the shape. First, a standstill due to missing management decisions is more likely during the daytime, when management is on-site to order the standstill. It is unlikely that foremen would make such costly decisions without consulting their superiors. Second, the peak around noon could correlate with reporting flows and mark the time when crucial information from the night shift, together with a first report

from the day shift, reaches decision-makers. These then require time to make the decision, until which the injection unit is idle. Finally, a mundane explanation is management's lunchtime habits. With the mobile phone muted, decision-makers are not available to advise workers, who stay idle until the boss is back in his site container.

Finally, idle time due to *Holiday and Weekend* is evenly distributed over the day. This distribution is as expected because these processes comprise a whole day. Interesting, however, is a slightly higher value for the night shift. This increase reflects the urge of the construction site to work. As the night shift crosses the day-to-day boundary, it is a question of definition whether it counts as a holiday if it starts or ends during a holiday. Therefore, it can be assumed that night shifts start or end during holidays or weekends, while the day shift cannot use this effect.

As a whole, the processes' distribution is sound and reflects the circadian performance dynamics of the construction site. This is an indication that the measured data, on the one hand, are valid and, on the other hand, have been correctly imported and further processed.

5.5.5 Process Duration

In this section, we take a look at the duration of three important processes. The injection, the maintenance, and the shifting process are analyzed through their histogram. Figures 5.25, 5.26 and 5.27 show how often the measured duration of a process is within a small time interval. For this the possible durations $0 \leq d \leq \max d$ are sorted in $N = 100$ bins of equal width $w = \max d/100$. The number of events are the number of durations which fall into this bin. For instance in Figure 5.27 6^{th} injection, only two bins contain one duration each. Each of the figures show the curve of a continuous lognormal distribution, which has been fitted to the sample data using the MATLAB function `fitdist`, for more on data fitting see MathWorks (2019b), Johnson et al. (1995), and Bowman and Azzalini (1997).

All three processes' durations seem log-normally distributed. However, with shrinking sample size, e. g., as for the maintenance processes, Figure 5.26, the accuracy of the fit becomes more uncertain. For very small samples, as with the 6^{th} injection of a borehole, a fit is not possible at all; see Figure 5.26.

Albeit the process of moving the injection nozzle from one borehole to the other seems independent of the grout material, which is injected only after moving the nozzle, the mean duration $\bar{d}$ between AC and PU material differs, Table 5.5.

Table 5.5: Process duration statistics.

Pump	$\bar{d}$ [min.]	σ [min.]
GEA 1/p1	12.5	8.7
GEA 3/p1	12.5	9.0
PU Station 1/p1	9.8	14.8

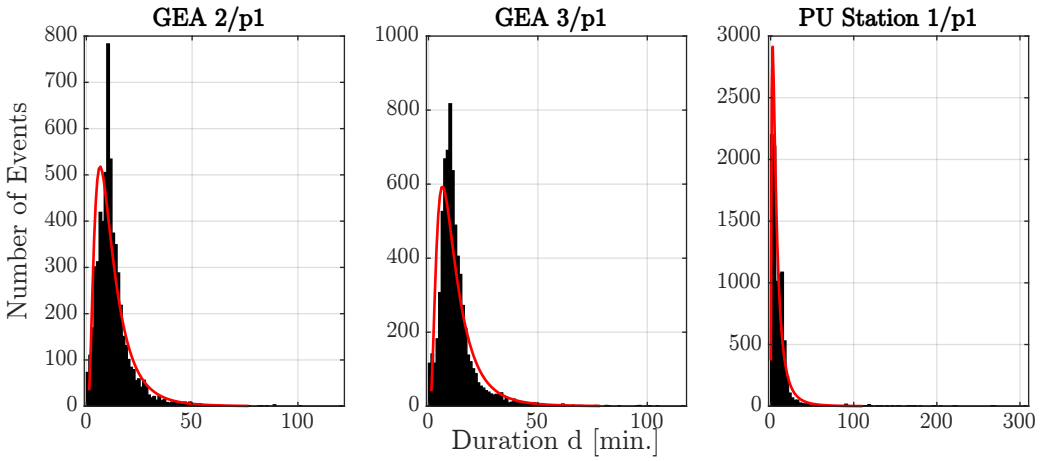


Figure 5.25: Histogram of the duration of the *shifting process* of three selected injection pumps in tube 258H.

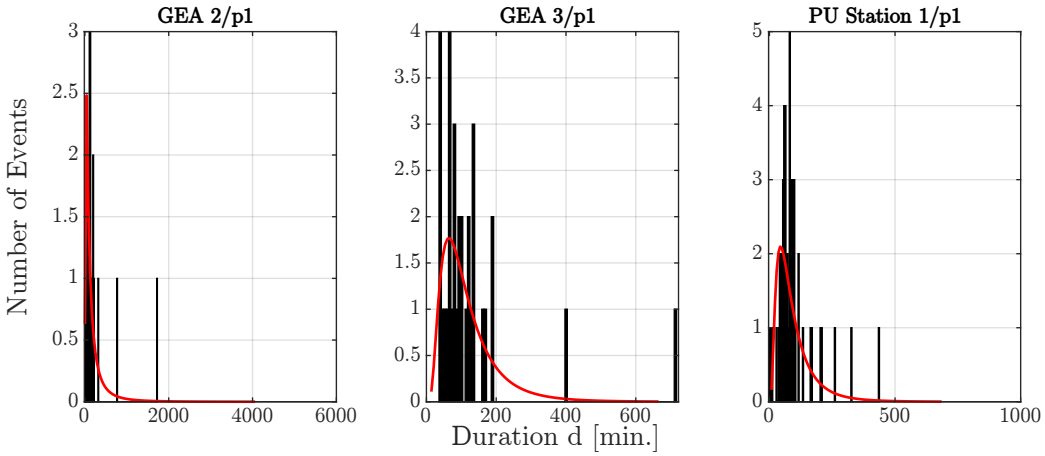


Figure 5.26: Histogram of the duration of the *maintenance process* of three selected injection pumps in tube 258H.

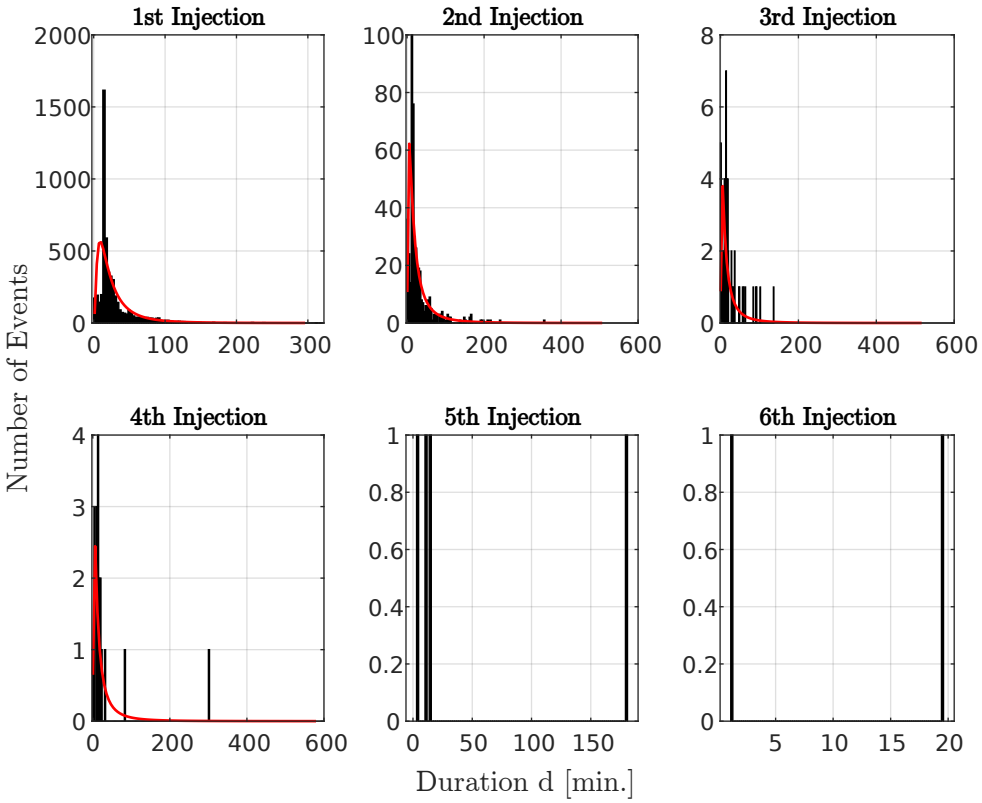


Figure 5.27: Histograms of the duration of the *injection process* of GEA2/p1 at the i^{th} injection in tube 258H.

5.5.6 Machine Location in Tunnel

In Figure 5.28 a plot of the time-location coordinates of injections in tube 251H are presented. Plot for all tunnel tubes can be found in Figure C.1-C.4. Areas that are not being injected (roughly 350–400 m, 1200–1300 m, and 1600–1700 m) are clearly visible. A PU injection is usually the final injection. Different injection units work together, e. g., at tube meter 350, GEA 1 is followed by GEA 2, and the final injection is produced by Mob. PU 1. Furthermore, the pilgrimage step procedure is visible. Figure 5.29 shows an enlarged part of Figure 5.28 in which the repeated passage through the tube can be observed. The slope of the lines indicates the operating speed. However, because the number of injections per section differs, the slope alone is insufficient to make a statement on the injection unit's overall performance. The time-process duration diagram, which is discussed in the following section 5.5.7 is better suited for such performance statements.

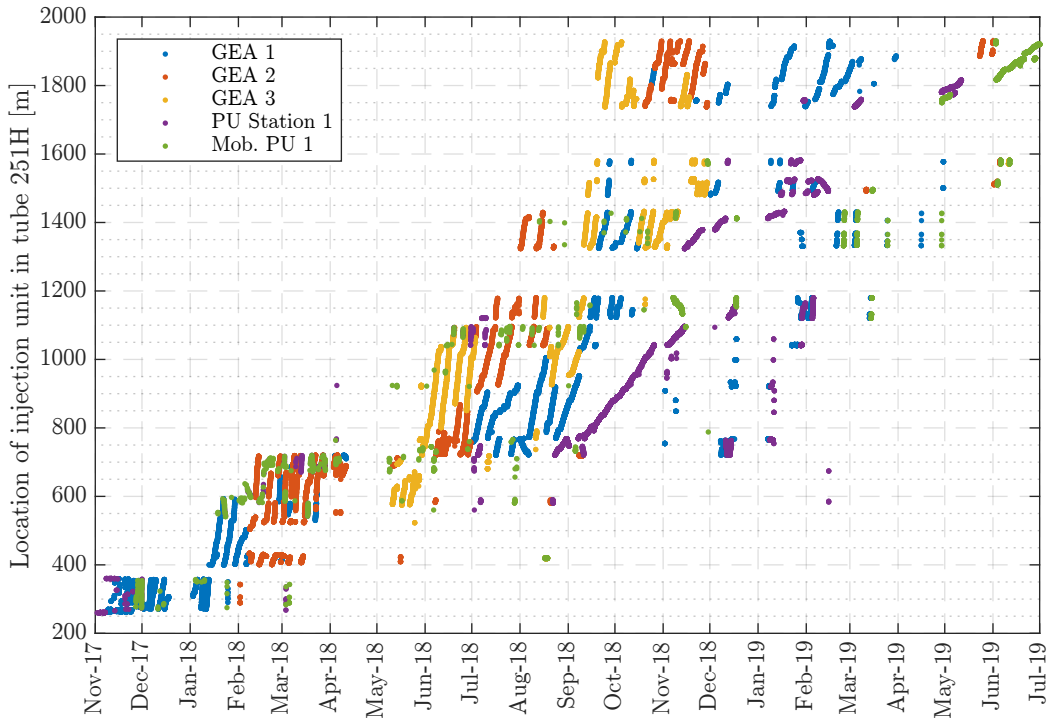


Figure 5.28: Location of injection units in tube 251H. While in some areas different AC units (GEA) support each other the final injection is performed by PU injection units.

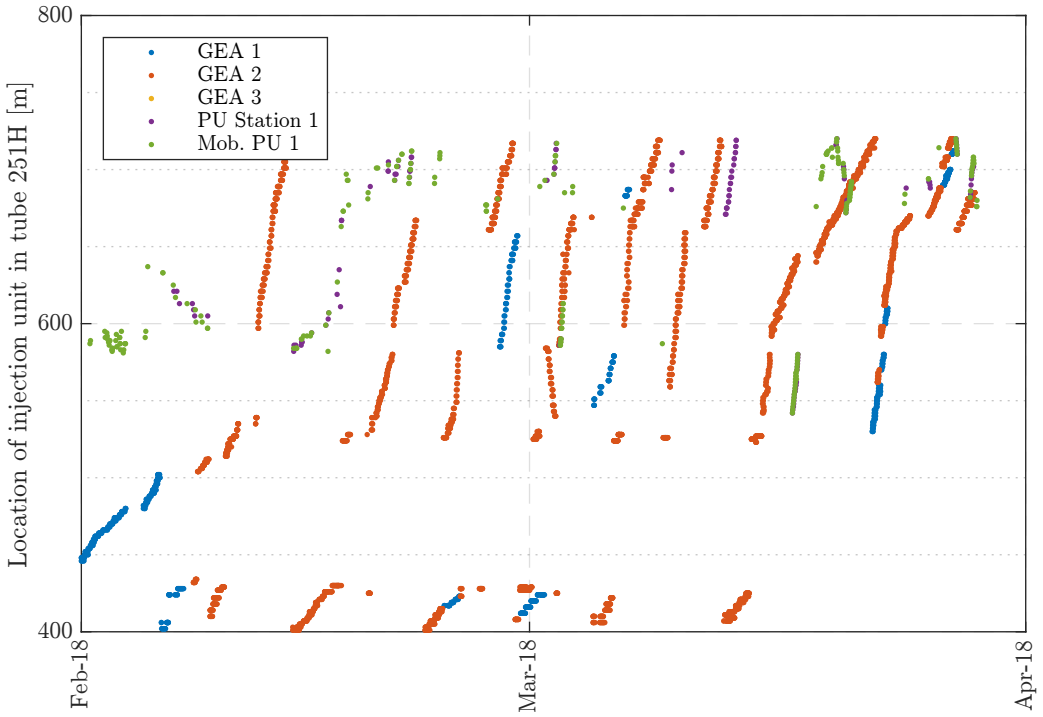


Figure 5.29: Zoom on plot location of injection units in tube 251H. The pilgrim step procedure is visible in the movement of the machines.

5.5.7 Number of Injections and Process Duration per Day

A typical performance factor for construction managers on injection construction sites is the number of injections per day. These are displayed in Figure 5.30, individually for each machine in red, green, and blue color. A black dashed line shows the sum of the three colored lines. Displayed is the moving average over eleven days.

The maximum number of machine injections seems to be limited to roughly 80-100 injections/day. With two injection units supporting each other, this leaves us with 160-200 injections/day per tube. From Figure 5.30 different project phases can be identified, namely the ramp-up and training phase (Nov 17-May 18), the production phase where the bulk of the injections is being performed (May 18-Jun 19), and the ramp-down phase at the end of the project (Jun 19-Nov 19).

Each phase follows a distinguished strategy: During the *ramp-up phase*, only one team works in the tunnel at a time. The machines in use are GEA 1. After the team has worked and learned with it for about four months, this unit is replaced with GEA 2. Typical for this phase is that maintenance times are higher than usual, Figure C.9. The parallel use of several machines characterizes the *productive phase*. Each unit's production rate is high, while time spent with maintenance is low; see Figure C.9. The third phase, *ramp-down*, shows a significantly lower output per unit and the machine park's shrinking size. GEA 1 is not active anymore, and unproductive time is high. Maintenance times at the end of

the project are higher than before; see Figure C.9. Furthermore, the time which the team spends each day with the moving process increases. As each injection is preceded by a moving process, the time spent moving the injection nozzle should decrease if the number of injections per day decreases. The contrary is the case, e. g., for machine GEA 3 when comparing the time of Feb 19 (productive phase) with Sept 19 (ramp-down phase) in Figure C.9 and Figure 5.30. The lower output seems to follow a strategy that tries to keep the staff on the construction site as long as possible, i. e., until the contractually stipulated deadline is reached. The reason is that, if a new project has not been acquired, project teams try to stick together as long as possible because the team requires trained workers for a follow-up project. If the project finished early, most foreign workers would leave the country and enter into contractual obligations with other construction projects. This assumption has been confirmed through interviews with construction managers.

Apart from a general insight into the construction site's performance, this section's main finding is that the significance of the discussed parameters is suitable for creating KPIs.

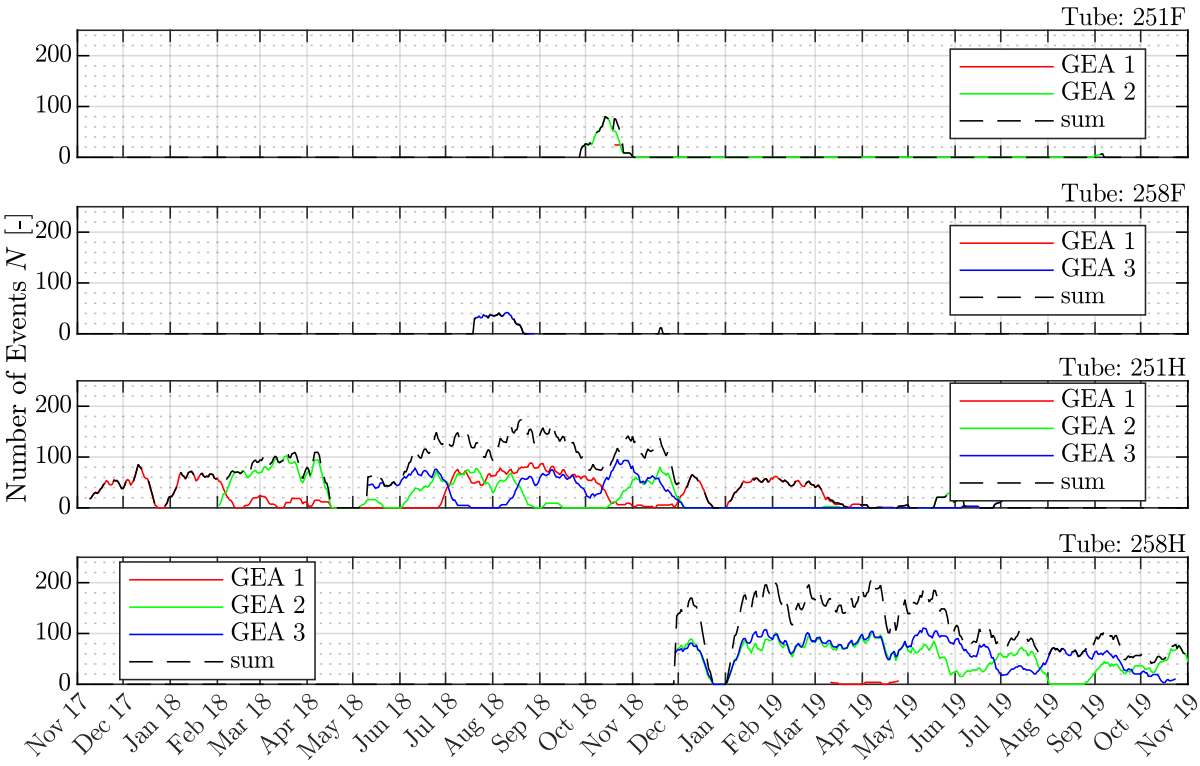


Figure 5.30: Number of AC injections per day in each of the four main tubes.

5.5.8 Number of Injections and Grout Volume per Borehole

The number of injections carried out per borehole depends on two factors: First, the planned number of injections specified by the client and second, the number of small, undocumented incidents such as easy to handle malfunctions.

Figure 5.31 and Figure 5.32 show the number of AC injections per day over the project time. After a test phase (Oct 17-May 18), the client decided to change the injection strategy, which lowered the planned AC injection from two to one injection per borehole. At the end of the project, starting in Jul 19, the number of injections per borehole is slightly rising. After consulting a construction manager from the construction site, two possible reasons for this behavior have been identified. First, a new, inexperienced team had been introduced to the site. As a result, the number of small mistakes increases due to the new workers' still low experience. A second explanation, which might complement the first, is an overzealous interpretation of injection criteria. With the project coming to an end and no following project in prospect, workers' incentive to finish fast is low. A lower injection quality, e. g., of 95 percent, might suffice during peak times when time is precious, and the team is uncertain whether it can finish within the agreed time frame. When the team already knows that deadlines will be met during the end phase, the team might opt for high quality and redo injections, which would have been counted as finished in rushed times; see Figure 5.32.

A geological reason for the change in the average number of injections is unlikely. Figure 5.33 and Figure 5.34 show the number of injection per tunnel meter. A correlation between tunnel meter and the number of injections does not seem to exist, except for areas which have been injected in the same project phase, e. g., boreholes in Figure 5.33 position 0-750 m, which have been injected during the ramp-up phase (Figure 5.31, Oct 17-May 18).

For each of the main tubes $t \in T = \{251\text{H}, 258\text{H}, 251\text{F}, 258\text{F}\}$ the mean value of the grout volume v and the share of boreholes which have been injected i times are plotted in two separate diagrams. Figure 5.35 shows the results for both, AC and PU injections. The general rule, that boreholes which are injected more often require more volume of the injected agent seems to apply with the exception of boreholes which are injected just one time; see Figure 5.35, 5.36 and 5.37. Overall most boreholes are injected two times, Figure 5.35. One time with AC, Figure 5.36 and one time with PU agent Figure 5.37. The share s_i of boreholes which are injected more than $i = 5$ times is, with $s_5 < 1$ %, small. Over 20 percent of boreholes in the tubes 258H, 251F, and 258F are injected only once. Because PU injections never precede AC injections, these 20 percent are AC injections.

The exact number of AC to PU injection combinations $c_{t,ac,pu} \in C_t$ per tube t is shown in EQ 5.8. The indices i (= ac) and j (= pu) of the matrix correspond to the number of injections minus one, i. e., $c_{251\text{H},1,3} = 114$ is the number of boreholes which have never been injected with AC and two times with PU grouting agent. For better readability EQ 5.9 shows the share of boreholes in percent for a reduced set of AC–PU combination $c_{t,\%,i,j} \in C_{t,\%}$ with $i, j \leq 4$. Values have been rounded to zero digits.

The data set contains no boreholes which have not been injected, $c_{t,1,1} = 0$. The boreholes which have been injected one time with AC and PU each predominate ($c_{t,2,2} \gg c_{t,ac\neq 2,pu\neq 2}$), and have a share of about 60-70 percent of the total number of injections. While boreholes with PU but without an AC injections can be found in the F-tubes ($c_{251\text{F}/258\text{F},\%,1,\cdot} \approx 26$ %) this is rarely the case in the H-tubes. The opposite is true for boreholes, which have been injected with AC agent but miss the PU injection. These predominate in the H-tubes ($c_{251\text{H}/258\text{H},\%,\cdot,1} \approx 18$ %).

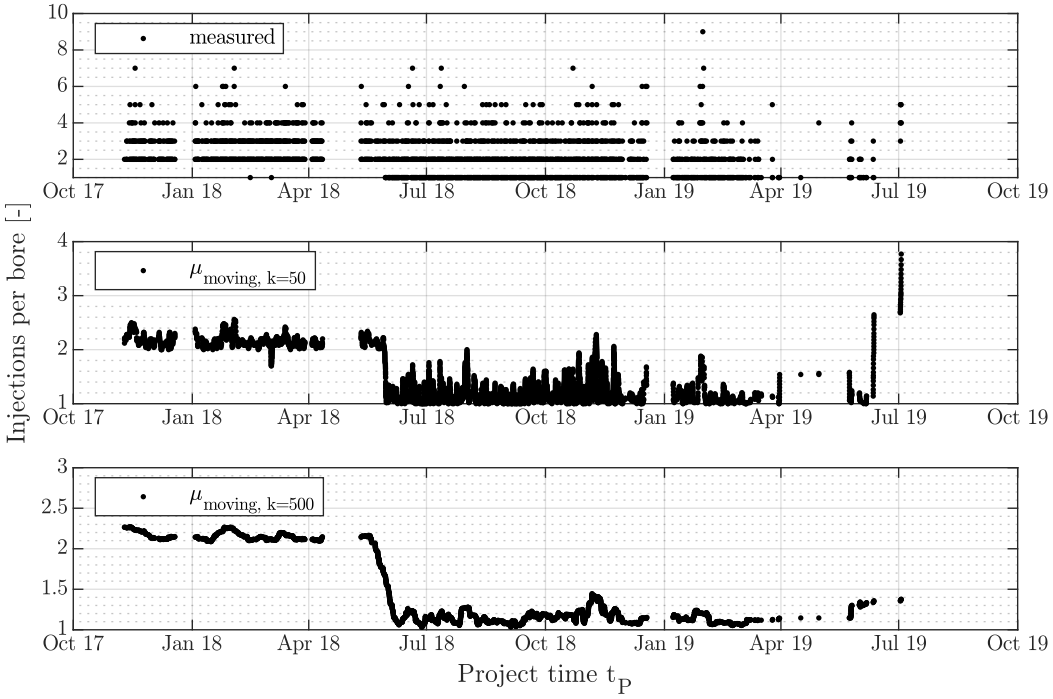


Figure 5.31: AC Injections per borehole over project time t_P in tunnel tube 251H. Each borehole is injected between one and ten times (top). The moving average over 50 boreholes takes values roughly between 2 and 2.5 until it drops in June 18 to roughly 1 to 2 injections per borehole (mid). With growing sample size k the moving average converges against the real average ($\mu_{real,258H} = 1.105$). At $k = 500$ samples the moving average $\mu_{moving,k=500}$ takes values within the limits $l \approx [2.0, 2.3]$ until it drops in Jun 18 to $l \approx [1.0, 1.5]$ (bottom).

However, these values are true for the data set as a whole. As shown in Figure 5.31 and 5.32, the ratio between boreholes which have been injected $i \in \{1, \cdots, 10\}$ times changes over time.

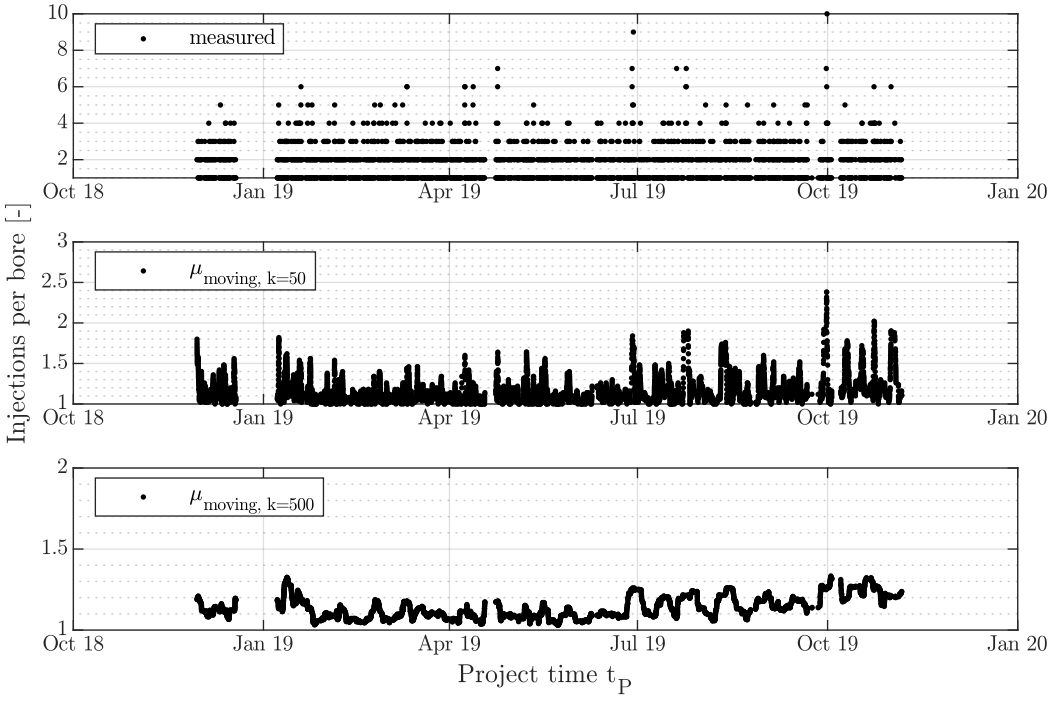


Figure 5.32: AC Injections per borehole over project time t_P in tunnel tube 258H. Each borehole is injection between one and ten times (top). The moving average over 50 boreholes takes values roughly between one and 2.5 injections per borehole (mid). With growing sample size k the moving average converges against the real average ($\mu_{real,258H} = 1.105$). At $k = 500$ samples the moving average $\mu_{moving,k=500}$ takes values within the limits $l \approx [1.0, 1.35]$ (bottom).

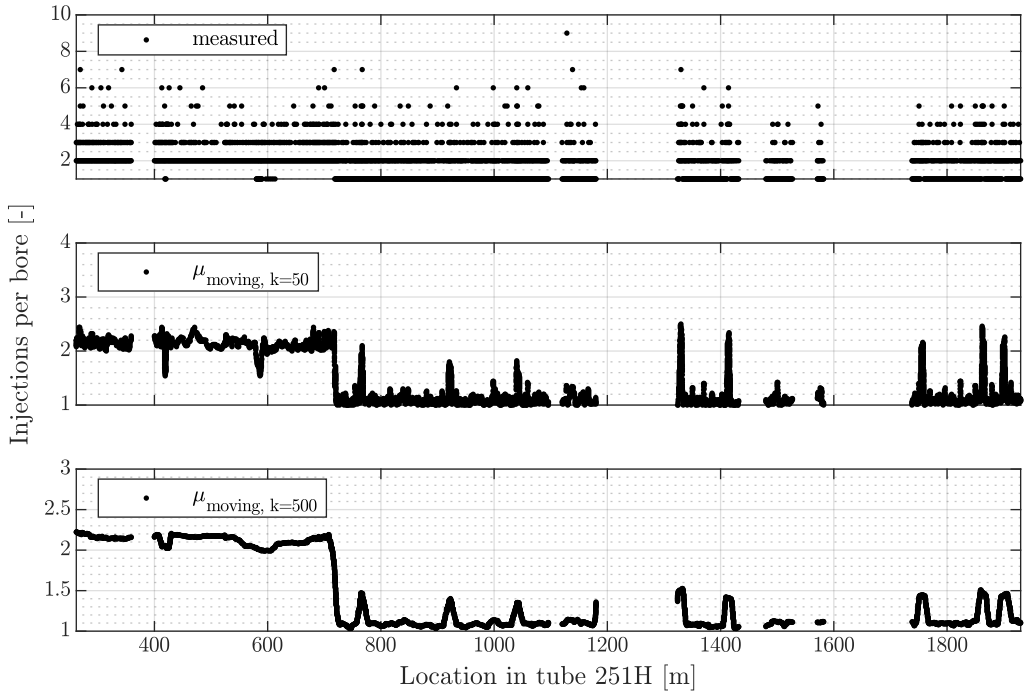


Figure 5.33: Number of AC injections per borehole over location in tunnel tube 251H. The middle and bottom plot show a location dependent change of the injection strategy.

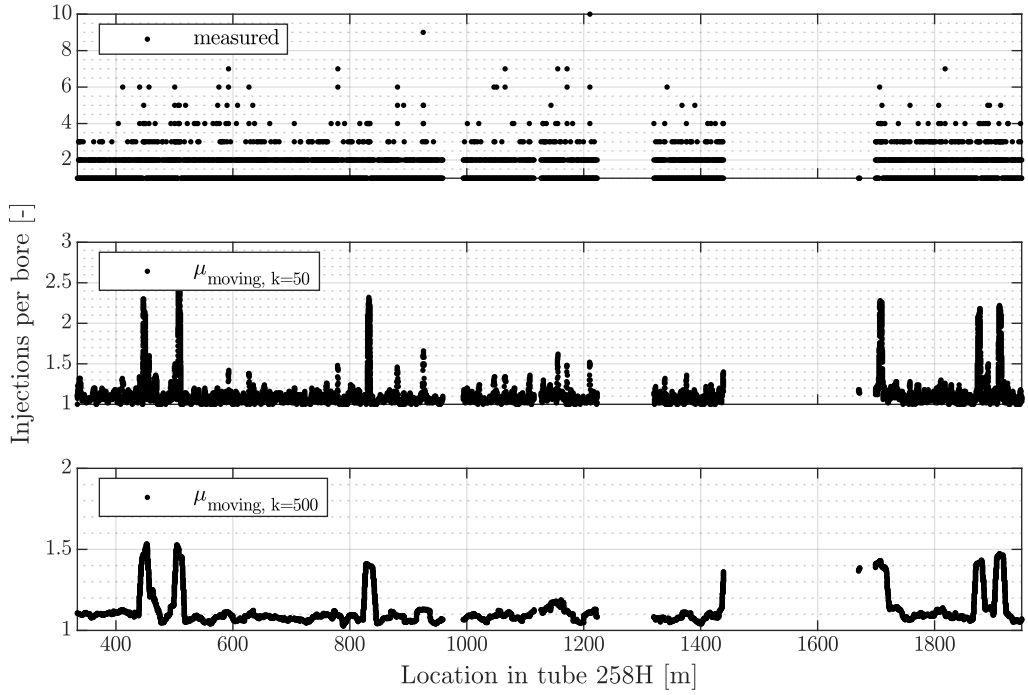


Figure 5.34: Number of AC injections per borehole over location in tunnel tube 258H.

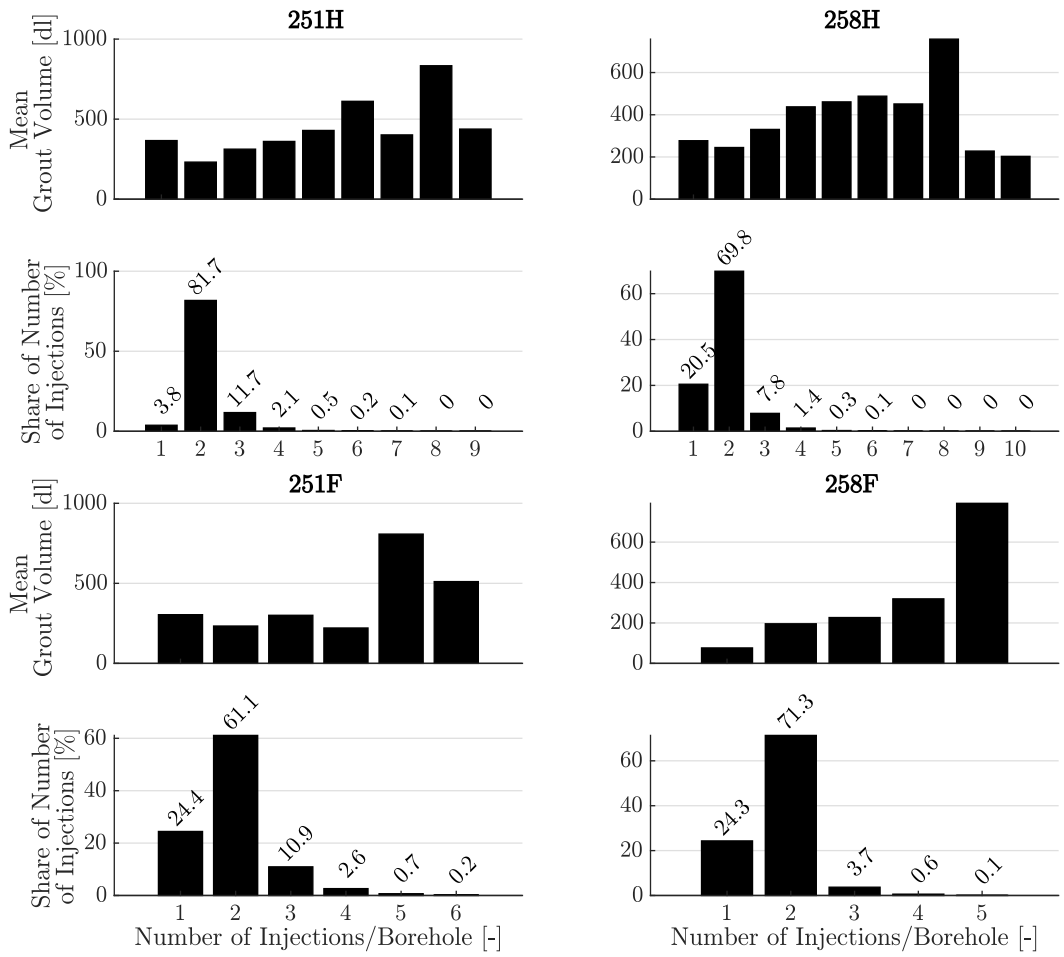


Figure 5.35: AC and PU grout volume versus number of injections per borehole.

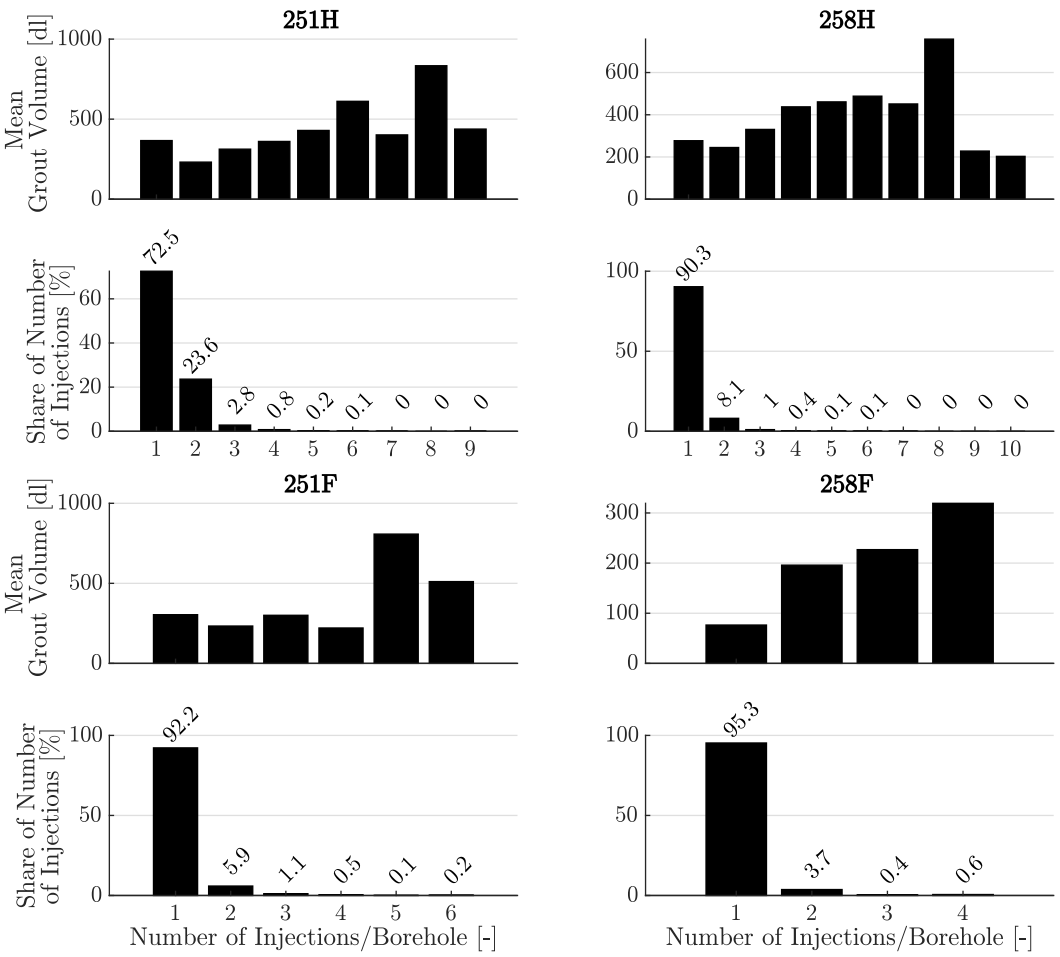


Figure 5.36: AC grout volume versus number of injections per borehole.

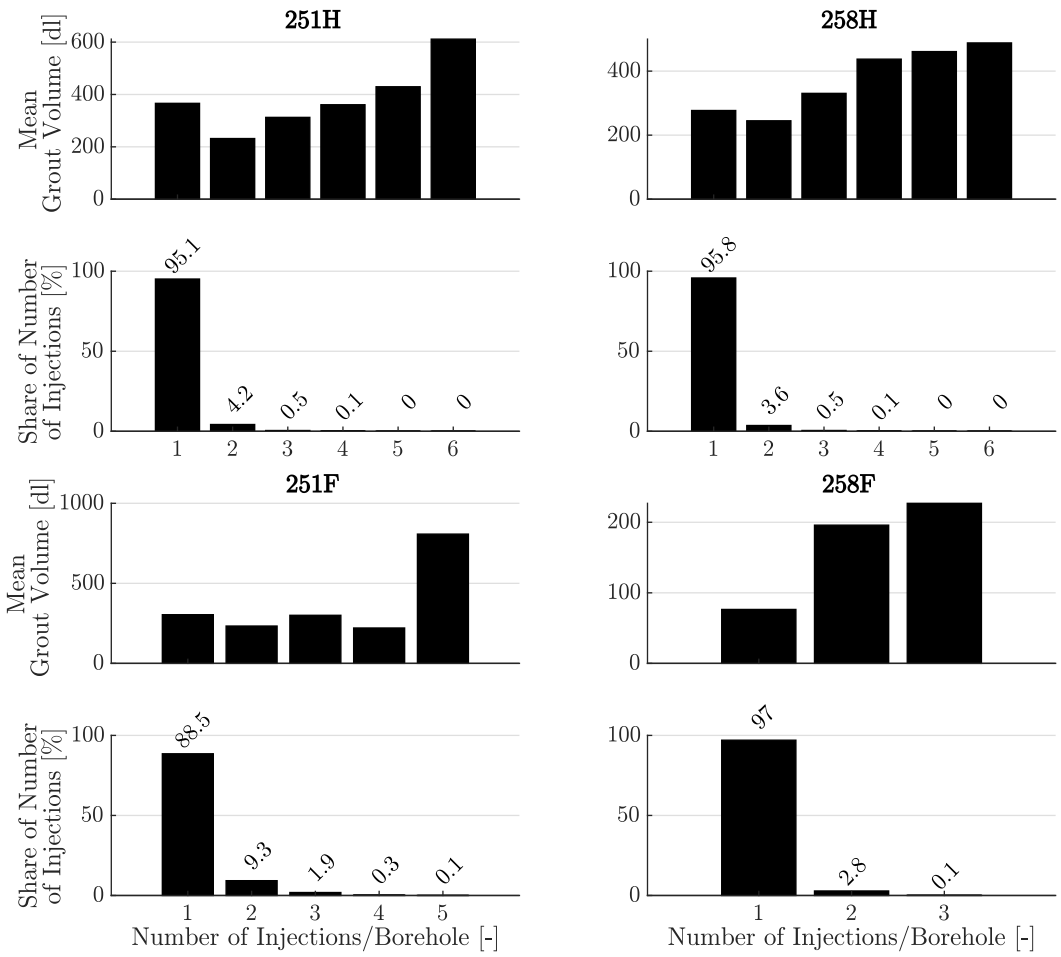


Figure 5.37: PU grout volume versus number of injections per borehole.

$$C_{251\mathrm{H}} = \begin{pmatrix} 0 & 889 & 114 & 20 & 6 & 1 & 1 \\ 214 & 19315 & 786 & 86 & 16 & 3 & 1 \\ 4433 & 2109 & 97 & 10 & 3 & 1 & 0 \\ 496 & 280 & 15 & 1 & 1 & 0 & 0 \\ 134 & 76 & 2 & 0 & 0 & 0 & 0 \\ 27 & 27 & 0 & 0 & 0 & 0 & 0 \\ 9 & 11 & 0 & 0 & 0 & 0 & 0 \\ 3 & 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \; C_{251\mathrm{F}} = \begin{pmatrix} 0 & 430 & 31 & 8 & 0 & 0 \\ 8 & 1060 & 122 & 24 & 5 & 1 \\ 4 & 65 & 8 & 1 & 0 & 0 \\ 1 & 11 & 3 & 0 & 0 & 0 \\ 4 & 2 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & 0 & 0 \end{pmatrix},$$

$$C_{258\mathrm{H}} = \begin{pmatrix} 0 & 1294 & 90 & 25 & 5 & 1 & 1 \\ 5919 & 23633 & 804 & 103 & 17 & 2 & 0 \\ 814 & 1792 & 113 & 16 & 1 & 0 & 1 \\ 126 & 215 & 10 & 0 & 0 & 0 & 0 \\ 61 & 62 & 3 & 0 & 0 & 0 & 0 \\ 8 & 13 & 1 & 0 & 0 & 0 & 0 \\ 13 & 8 & 0 & 0 & 0 & 0 & 0 \\ 4 & 4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \; C_{258\mathrm{F}} = \begin{pmatrix} 0 & 328 & 28 & 2 \\ 1 & 937 & 10 & 0 \\ 0 & 37 & 0 & 0 \\ 1 & 3 & 0 & 0 \\ 5 & 1 & 0 & 0 \end{pmatrix} \tag{5.8}$$

$$C_{251\mathrm{H},\%} = \begin{pmatrix} 0 & 3 & 0 & 0 \\ 1 & 66 & 3 & 0 \\ 15 & 7 & 0 & 0 \\ 2 & 1 & 0 & 0 \end{pmatrix}, \; C_{251\mathrm{F},\%} = \begin{pmatrix} 0 & 24 & 2 & 0 \\ 0 & 59 & 7 & 1 \\ 0 & 4 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix},$$

$$C_{258\mathrm{H},\%} = \begin{pmatrix} 0 & 4 & 0 & 0 \\ 17 & 67 & 2 & 0 \\ 2 & 5 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}, \; C_{258\mathrm{F},\%} = \begin{pmatrix} 0 & 24 & 2 & 0 \\ 0 & 69 & 1 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{5.9}$$

5.5.9 Grout Volume vs. Injection Time

The volume of grouting agent v_g injected per borehole depends on the number, size, and shape of cracks and cavities in the surrounding rock. Four general cases can be distinguished; see Figure 5.38.

(1) If the borehole is drilled through solid rock and does neither cross cracks nor cavities, then v_g is limited by the volume of the borehole

$$v_{g,max} = 100 \cdot \pi \cdot (0.51/2)^2 \approx 20 \text{ [l]}. \tag{5.10}$$

With a diameter of 51 mm and a maximum length of 10 m the maximum grout volume is 20 liter.

Depending on the packer's position during the injection and the number of times the borehole has been injected before $v_{g,max}$ can be spread over several injections. For instance, a typical case includes placing the packer in the middle of the borehole during the first injection and at the hole entrance for the second injection. This placing of packers effectively splits the maximum grout, exclusively used to fill the cavity occupied by the borehole, by two.

(2) If the borehole crosses cracks and cavities alike, the grouting agent will first flow into the bigger cavities. Once these have been filled, the system's pressure will rise, and smaller cracks are filled as well.

(3) If cavities exist that are only connected to the borehole with a small crack, then high pressure over a long time is required to finish the injection. This is because the grouting agent has to flow through a narrow crack, slowing down the process. At the same time, the total amount of the injection agent is high.

(4) If a voluminous cavity exists and is crossed by the borehole, then the grouting agent's flow is not hindered. High volumes with a relatively short injection period are the result.

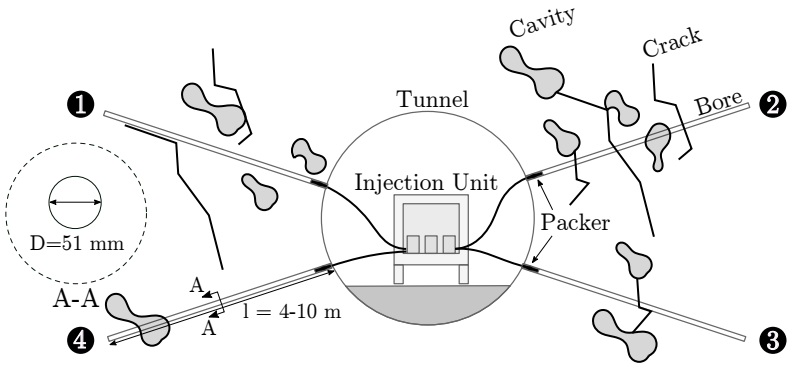


Figure 5.38: Sketch of four cases influencing the injection time and grout volume. The borehole crosses (1) neither cracks nor cavities, (2) both cracks and cavities, (3) only cracks which are connected to cavities and (4) only cavities but no cracks.

Figure 5.39 and 5.40 show a plot of the injection time t against the grout volume g for the tubes 251H and 258H. If a borehole has been injected n times, then t and g are the sum of all individual injections performed at that borehole, i. e.,

$$t = \sum_{i=1}^{n} t_i, \quad g = \sum_{i=1}^{n} g_i. \tag{5.11}$$

While the top subplot shows all measured injections, the bottom plot shows an enlarged section of the upper plot. This enlarged section contains approximately 95 Both figures show a remarkable accumulation of measuring points at a multiple of $g = 10$ l grout volume. The minimum duration, $\min t$, required for one injection is depended on the quantity of injected grout injected into the rock. It is limited by the maximum allowable injection pressure of 5 bar. This criterion allows for a maximum flow of $Q \approx 4$ l/min., i. e., $\min t(g) \approx \frac{1}{4} \cdot g + 1$, Figure 5.39 and 5.40. A line f to fit the data points has been calculated using the MATLAB function `fit` for linear polynomial curves, MathWorks (2019b). These are defined as

$$\begin{aligned} f_{251H}(g) &= 0.49 \cdot g + 25.54 \ , \\ f_{258H}(g) &= 0.70 \cdot g + 22.19 \ . \end{aligned} \tag{5.12}$$

Table 5.6 furthermore provides the mean $\hat{x}$, the median $\bar{x}$ and the standard deviation σ of the injection time and the grout volume separately for tube 251H and 258H. Both distributions are slightly skewed ($\hat{x} \neq \tilde{x}$) and have a variance of 17-21 minutes respectively 14-17 liters.

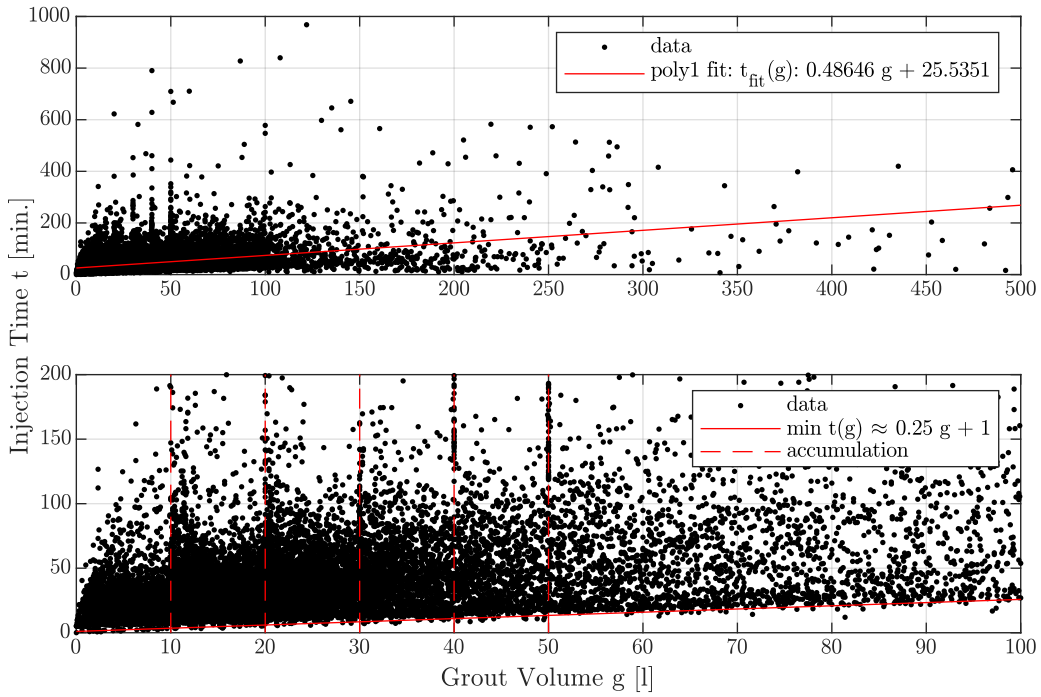


Figure 5.39: Tube 251H, Grout volume vs. injection time. The top diagram shows all measured points of the tube. A linear curve has been fitted to the data using the MATLAB function `fit`. The bottom diagram shows 95 percent of the measured points with respect to their grout volume. A linear curve shows the approximated minimum injection time per grout volume. An accumulation of points is visible at multiples of $g = 10$ l.

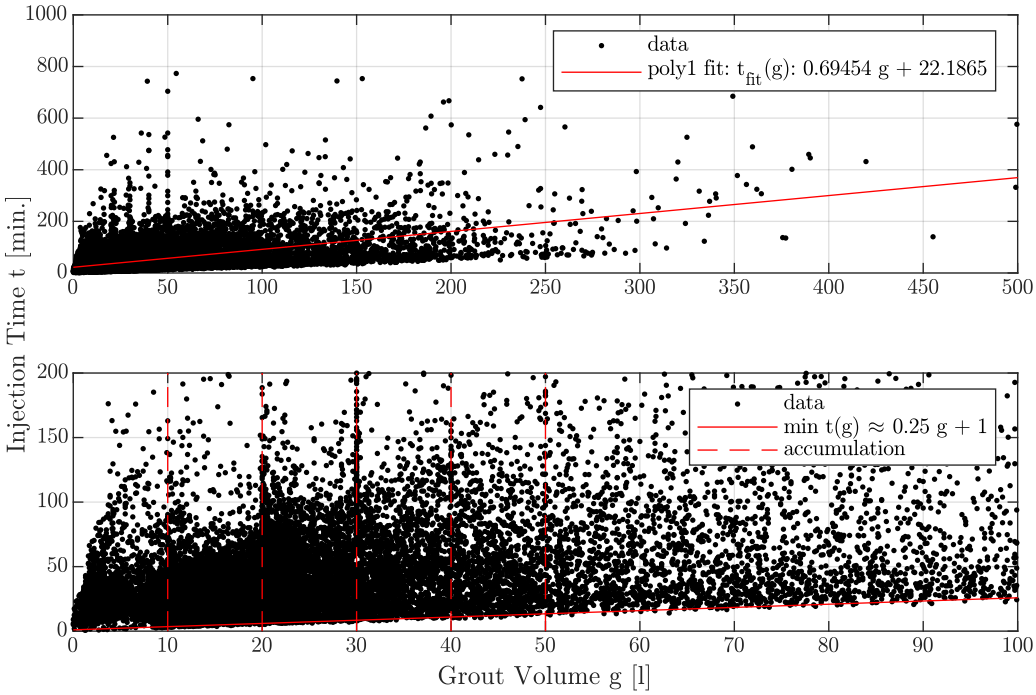


Figure 5.40: Tube 258H, Grout volume vs. injection time. The top diagram shows all measured points of the tube. A linear curve has been fitted to the data using the MATLAB function `fit`. The bottom diagram shows the injection time over the grout volume of 95 percent of the measured points. A linear curve shows the approximated minimum injection time per grout volume. An accumulation of points is visible at multiples of $g = 10\,\mathrm{l}$.

Table 5.6: Statistics: Injection Time and Grout Volume.

	Injection Time [min.]		Grout Volume [l]	
	251H	258H	251H	258H
x_{95}	95	111	78	95
$\hat{x}$	31	33	19	20
$\tilde{x}$	26	25	14	13
σ	17	21	14	17

5.6 Grout Volume Forecast

The storage capacity on construction sites is notoriously small. This is especially true in tunnel construction, where all building materials have to be transported through exactly the cavity that is being built. In the case of the surveyed construction site, the chemical components for the AC and PU injection agent have to be transported to the injection units. They are stored in canisters at the side of the tunnel, Figure 5.12, from where the workers have to manually lift them into the injection containers for further processing. If the injection unit moves before all canisters have been processes, transporting the unprocessed agent means additional work for the injection team. Furthermore, the canisters are blocking parts of the tunnel, and stored material binds cash. In the spirit of Lean Construction, it is, therefore, advantageous to avoid storage of injection agent wherever possible; on the prevention of waste see also Backhaus and Dahm (2020) and Womack and Jones (2013).

With the goal to improve the grout material allocation, this section presents three attempts to perform a forecast of the grout volume g. First, grout volumes from a finished tube are mapped to an unfinished tube using cross-correlation analysis, section 5.6.1. Then, samples of varying sizes are used to forecast future grout volumes within the same tube using a statistical model, section 5.6.2. Finally, an ANN is trained on the grout volume data of one tube to predict future grout volumes of the other tube-based on samples from that tube, section 5.6.3. At the end of this section, a short summary discussing the applicability of the results is given, section 5.6.4.

5.6.1 Forecast with Cross-Correlation Analysis

This section describes a model that uses cross-correlation analysis between the two main tubes of the tunnel to predict the grout volumes of one tube based on the grout volumes of the other.

5.6.1.1 Model and Results

The two tubes 251H and 258H, run parallel to each other. Due to their geometric proximity, it could be assumed that the soil properties P in the tube sections are also similar, i. e., $P_{251H} \leftrightarrow P_{258H}$. If such a relationship exists between the two tunnel tubes, this knowledge could improve the forecast of the expected grouting volume or injection times. One would simply sample the properties $p_1 \in P_1$ from a finished tube and use the sample to compute a forecast for the second, unfinished tube: $p_2(x_2) \approx p_1(x_1)$. However, it is unlikely that soil layers run perpendicular to the tunnel axis. Furthermore, the tunnel coordinates have been defined regardless of soil layers, see Figure 5.41. Hence, it can be expected that a transformation of tube coordinates would improve the result of the forecast, when mapping the soil properties of one tube to another, i. e.,

$$\begin{aligned} x_2 &= x_1 \cdot T, \\ p_2(x_1 \cdot T) &\approx p_1(x_1). \end{aligned} \tag{5.13}$$

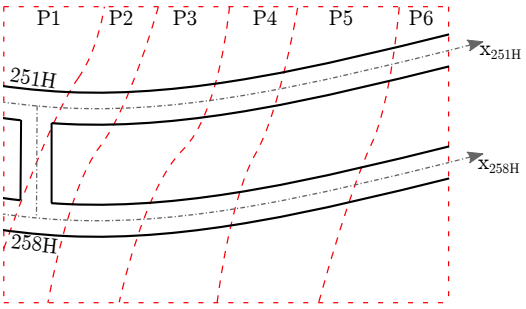


Figure 5.41: Sketch of hypothetical soil layers with individual properties P1 to P6. The layer's boundaries are skewed. In order to map $P_i(x_{251H}) \mapsto P_i(x_{258H})$ the x-coordinates must be transformed.

The only meta-parameter of the soil available through the GBPlan database is the injected grout volume. By following the logic of EQ 5.13, we assume that a linear relationship between the soil type and the volume of grouting material g injected into the rock exists. Other influences on g, for instance, the speed with which the TBM drilled through the rock, deviations in the shield pressure, or the influence of curves, are assumed to have an only minor influence on the grout volume.

Figure 5.42 (top, mid) shows a plot of the grout volume g which has been injected into the rock versus the position in the individual tube. Plotted are the average injection volumes per borehole of a tunnel tube ring-segment with a width of 1 m.

Areas that have not been injected likely do not contain Anhydrite rock material. For both tubes, these non-Anhydrite zones are more or less in the same area, e. g., at tunnel meter 1,300 and 1,600, respectively. The bottom plot shows the moving average over 11 injections for the two parallel tubes 251H and 258H. With the exception of several peaks (tunnel meter 600; 1,050; 1,500; 1,700-2,000) the lines seem to be close enough together to allow the application of EQ 5.13.

If we assume that $p(x_2) \mapsto p(x_1)$ via linear mapping of x_i, then a cross-correlation analysis, Box et al. (2008, pp. 474 sqq.), is suited to compute the shift s, such that

$$p_2(x_1 + s) \approx p_1(x_1). \tag{5.14}$$

The MATLAB function `crosscorr` is used to measure the similarity between the grout volumes per tunnel meter of one tube with a shifted or lagged version of the other as a function of the lag, MathWorks (2019b).

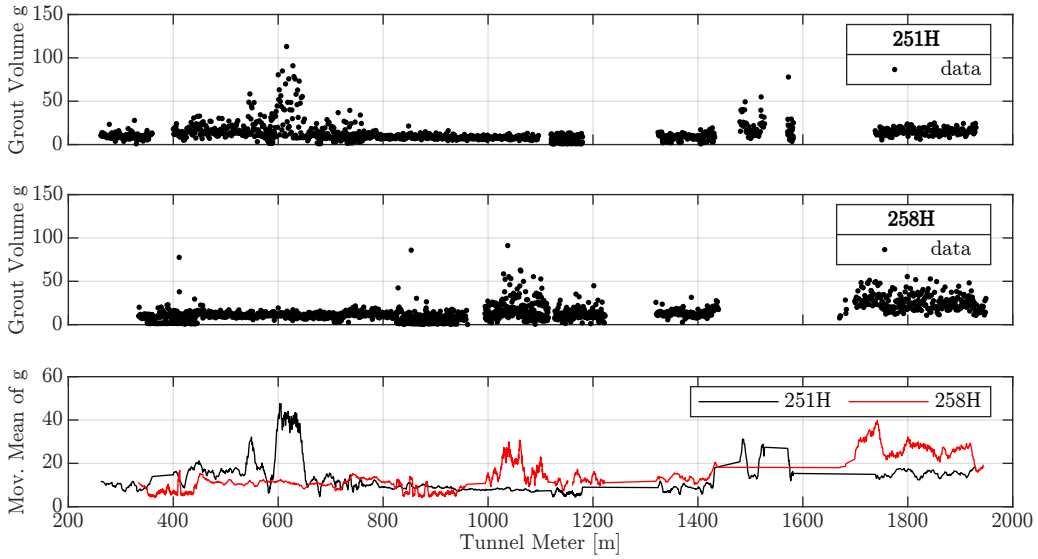


Figure 5.42: Grout volume per section (top, middle) and moving average grout volume per tunnel section (bottom) of tube 251H and 258H.

5.6.1.2 Discussion of Results

The plot in Figure 5.43 shows the *Sample Cross Correlation* (SCC), i. e., the similarity of the grout volume injected in the rock as a function of the lag (the longitudinal coordinate shift). The best value is reached at a lag of 425 m with a SCC of 0.34. Even though the result shows that a correlation between the injected grout volume of the two tubes exists, it is not strong enough to serve as a predictor for grout volume forecasts. Mean injection volumes from one tube cannot directly be transferred to a parallel tube.

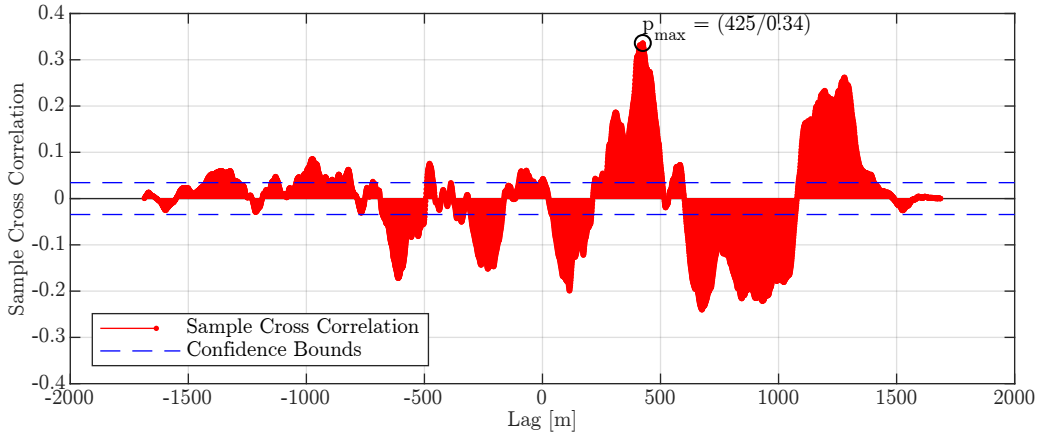


Figure 5.43: SCC between the grout volume of tube 251H and 258H. The SCC can be interpreted as the percentage of correlation between the two tubes. The plotted values are the SCC of the two tubes if they are shifted against each other by a number of meters defined by the lag. The confidence bounds indicate the value beyond which a correlation can be assumed.

5.6.2 Forecast with Stochastic Model

The previous section shows that the injected grout volumes cannot be easily transferred from one tunnel tube to a parallel tube. In this section, a method is presented which samples measured grout volumes from fully injected boreholes of a tube. It then uses a Markov chain to extrapolate the grout volume for the same tube's unfinished boreholes. The examples in this section show the forecast of the AC grouting volume. PU injections are being ignored. Some of the results presented in this section have already been published in Backhaus (2019). The inspiration to apply Markovian statistic to anticipate geologic characteristics comes from the works of Ioannou (1984), Kallen (2007), Lu et al. (2018), and Moavenzadeh et al. (1974).

5.6.2.1 Model and Results

GBPlan analysis the input data concerning the amount of injection agent injected at each injection stage. During the period under investigation, each borehole was injected up to ten times with AC grouting agent, EQ 5.8.

Decisive for the total grout volume V_{sum} is the number of boreholes d to be injected and the number of injections per borehole, also known as the injection level, $x \in X$. This is because the total volume of injection agent required to finish the injection of one borehole varies with the number of injections per borehole, Figure 5.36. During the period under investigation, each borehole was injected up to ten times with AC grouting agent, EQ 5.8.

The model for the stochastic forecast uses a sample S of already injected boreholes. That sample is used to calculate

a) the kernel K for a Markov chain, section 4.3, such that drawing k from K results in the same PDF as if drawing the number of injections per borehole x from S, i. e., $k \sim S$, and

b) the mean grout volume $\bar{v}_x$ for each injection level x, EQ 4.17.

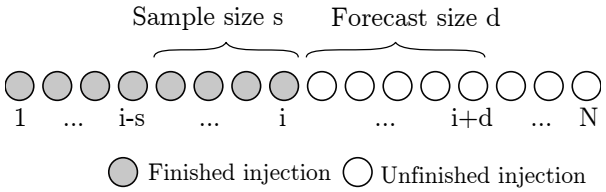


Figure 5.44: Computation of boreholes in sample and forecast.

The total remaining injection volume V_{forecast} is calculated by multiplying the injection levels of each borehole with their respective mean injection level volume $\bar{v}_x$. By adding the already injected grout volume V_{done} one gets the total expected grout volume V_{sum} for the tube

$$\begin{aligned} V_{\text{sum}} &= V_{\text{done}} + V_{\text{forecast}} \\ &= V_{\text{done}} + \sum_{p=1}^{d} \sum_{q=1}^{x_p} \bar{v}_q \end{aligned} \tag{5.15}$$

with

d : number of boreholes to be forecasted,

x_p : the injection level x of borehole p.

If $i \in \{1, \ldots, N\}$ denotes the number of the latest injected borehole and N the total number of boreholes in the tube, then the largest possible sample is $s_{i,\max} = i$ and the longest forecast duration $d_{i,\max} = N - i$, Figure 5.44. The sample and forecast duration are being varied to analyze the influence of the sample size s and forecast duration d on the forecast error. In this case the boreholes $b_i \in B$ of the tube which belong to the sample S or the forecast D depend on the sample size s and the forecast duration d, i. e.,

$$\begin{aligned} b_s &= \{b_j \in B \mid i - s < j \leq i\}, \\ b_d &= \{b_j \in B \mid i < j \leq i + d\}. \end{aligned} \tag{5.16}$$

5.6.2.2 GBPlan Implementation

The model's functionality has been implemented in the functions of the GBPlan package `GBGrouting_Markov`. `GBGrouting_Markov.run_compute_total_grout_volume` performs a simulation of the estimated total grout volume. Batch jobs for pre-defined combinations (s, d) of sample size s and forecast duration d are organized with the function `GBGrouting_Markov.run_batch_compute_grout_volume`. The visualization of the results in managed by `GBGrouting_Markov.plot_batch`.

5.6.2.3 Forecast Over Total Remaining Project Time with Maximum Sample

The development of the relative error δx of the forecast of the total grout volume after i fully injected boreholes is shown in Figure 5.45 and 5.46 for tube 251H and 258H. The sample used for the forecast contains all boreholes which have been injected up to the moment of the forecast, i. e., $b_s = \{b_j \in B \mid 1 \leq j \leq i\}$. Four curves are shown, which differ in the way the grout volume per borehole is being estimated. The curve *mean* uses the mean value, *trimmean* applies the 5 percent respectively 15 percent trimmed mean value and *median* the median grout volume for each injection level. The last three have been included to understand the influence of extreme grout volume values on the forecast.

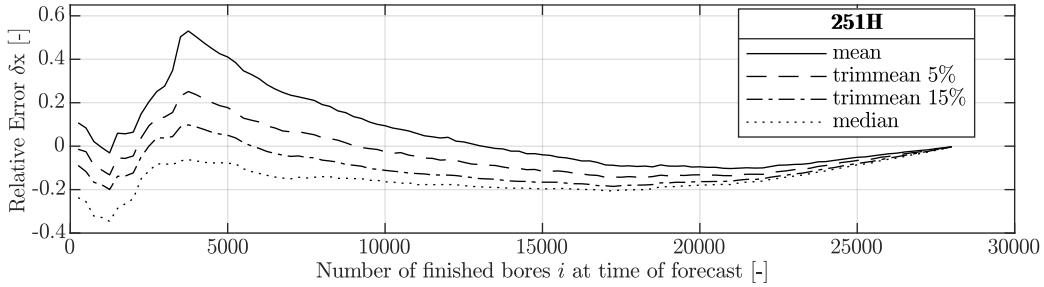


Figure 5.45: Relative error δx of total, remaining grout volume forecast at borehole i in tube 251H. The sample S contains all boreholes which have been injected up to the time of the forecast: $s_i \in S, i = \{1, \ldots, i\}$.

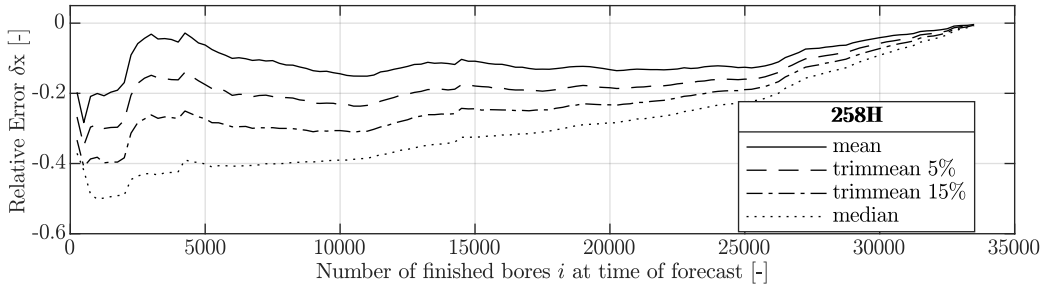


Figure 5.46: Relative error δx of total, remaining grout volume forecast at borehole i in tube 258H. The sample S contains all boreholes which have been injected up to the time of the forecast: $s_i \in S, i = \{1, \ldots, i\}$.

In tube 251H, the relative error δx of the mean curve is the highest during the early phase of the project (after 4,000 boreholes) with about 50 percent. From here on, the error shrinks until it reaches its minimum of approximately -10 percent after 20,000 boreholes. Then it continues to converge to zero. The four curves show the same behavior, however, with a relative error that seems scaled

$$\begin{aligned} r_{\text{mean}} > r_{5\%} > r_{15\%} > r_{\text{median}} \\ \text{with } r_{\text{curve}} \equiv |max(\delta_{\text{curve}}(i)) - min(\delta_{\text{curve}}(j))| \end{aligned} \tag{5.17}$$

and shifted along the ordinate

$$\delta_{\text{mean}}(i) > \delta_{5\%}(i) > \delta_{15\%}(i) > \delta_{\text{median}}(i). \tag{5.18}$$

In tube 258H all four curves underestimate the grout volume over the whole project period. The relative error δx for all four curves is negative with nearly -50 percent for the median curve early in the project. The smallest error is yielded by the mean curve, which only drops below -20 percent at the beginning of the project. Like in tube 251H, the curves seem shifted, EQ 5.18, with peaks early on during the project, after which they slowly converge towards zero.

In both diagrams, the error converges against zero, resulting from the shrinking extrapolation length and the correlated extrapolation error of the forecast. The vertical order of the lines, EQ 5.18, indicates, that outliers are foremost high, positive values ($> 2 \cdot \hat{x}_{\text{mean}}$), which is likely the reason for the scaling effect, EQ 5.17, as well.

The MAPE for each of the curves is listed in Table 5.7. For tube 251H the best results are provided with a trimmed mean of 5 percent ($\hat{x}_5 = 9$ %) and for tube 258H with a mean value ($\bar{x} = 12$ %). In average the mean curve yields the best results ($0.5 \cdot (\hat{x}_{251\text{H}} + \hat{x}_{258\text{H}}) = 11$ %).

Table 5.7: δ_{MAPE} of forecast of total, remaining grout volume for tube 251H and 258H.

Curve		251H	258H
mean	$\hat{x}$	12	10
trimmean 5%	$\hat{x}_5$	09	17
trimmean 15%	$\hat{x}_{15}$	11	22
median	$\tilde{x}$	15	29

values in percent

5.6.2.4 Forecast with Variation of Sample Size and Forecast Duration

It seems natural that a forecast over a short period yields better results than a forecast over a long period. Also, one might expect better results from a large sample than from a small sample. In tunnel construction, however, the genesis might change as the tunnel construction proceeds. That change may lead to the effect that large samples result in higher error values.

A parameter study has been performed to analyze the influence of the sample size s and the (expected) decline in accuracy with growing forecast duration d of the forecast function $f(s, d)$.

The changing parameters are s and d and the output is the MAPE: $\delta_{\text{MAPE}}(f(s, d))$. The analyzed values of s and d are $s, d \in P$ with

$$P = \{x \mid 250,500 \cdot i\} \text{ for } i = \left\{x \in \mathbb{Z}^{+}_{\neq 0} \mid 1 \leq i \leq 50\right\}, \tag{5.19}$$

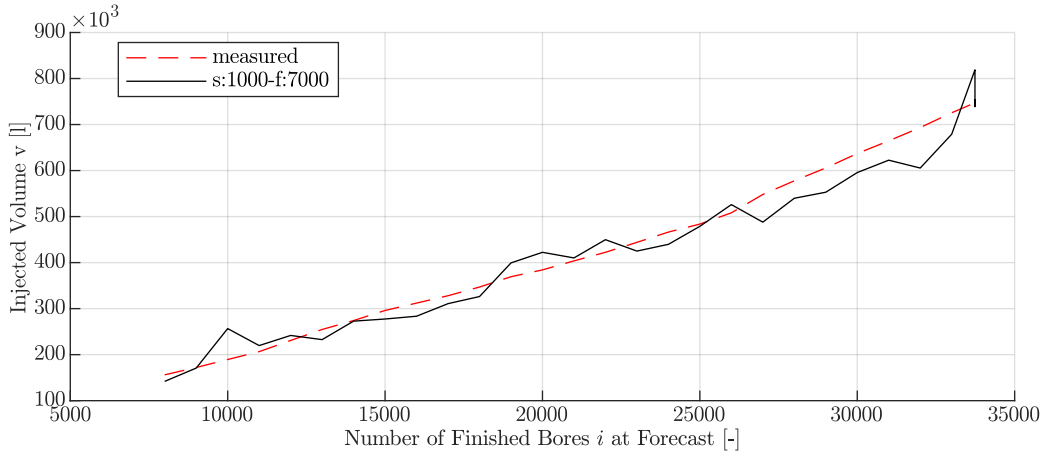


Figure 5.47: Example results of forecast $f(s = 1000, d = 7000)$ in tube 258H. The algorithm calculates a forecast all 1000 boreholes.

i. e., a total of $51^2 = 2601$ combinations are calculated every 1000 boreholes in each tube.

An example for one calculation of $f(s, d)$ in tube 258H is shown in Figure 5.47. The red dashed line shows the measured grout volume v, and the black line shows the simulated forecast. Both lines give the v up to the moment i, which has been forecasted. i is calculated as the point *from* which the forecast is made plus the forecast duration d. Consequently, the earliest forecast is at $i = 8000$ boreholes because the earliest forecast can be made after 1000 boreholes, when the minimum sample size for this scenario or the minimum interval of 1000 boreholes has been reached ($i_{min} = s + d = 1000 + 7000$). The MAPE is calculated for these two curves. For the example in Figure 5.47 it is

$$\delta_{\text{MAPE}}\left(f_{258\text{H}}(1000, 7000)\right) = 0.062 \approx 6\ \%.$$

A summary of all combinations (s, d) is visualized in two contour plots. One for tube 251H and one for 258H; see Figure 5.48 and 5.49. The MAPE in tube 251H is within a range of $\delta_{\text{MAPE}}(f(s, d)) \approx [1\ \%, 18\ \%]$. It shrinks from a global maximum at $(s, d) = (2000, 18000)$ to values under two percent for very small forecast durations ($d < 500$) and under four percent for very large samples over 24 thousand boreholes. The location of the global maximum is somewhat unexpected, however, still in an area (small s and large d) where the global maximum can be assumed. The decline to $\delta_{\text{MAPE}}(f(250, 25000)) \approx 14\ \%$ can be explained with a lucky composition of the samples $s = \{x \mid 1000 < x < 4000\}$.

The MAPE in tube 258H is smaller than in 251H and within the range of $\delta_{\text{MAPE}}(f(s, d)) \approx [1\ \%, 10\ \%]$. The highest error is measured for small samples with high forecast durations. However, some samples seem especially prone to higher errors. These are within the range of $s = \{x \mid 5000 < x < 17000\}$, where an increased sample size yields higher MAPE values.

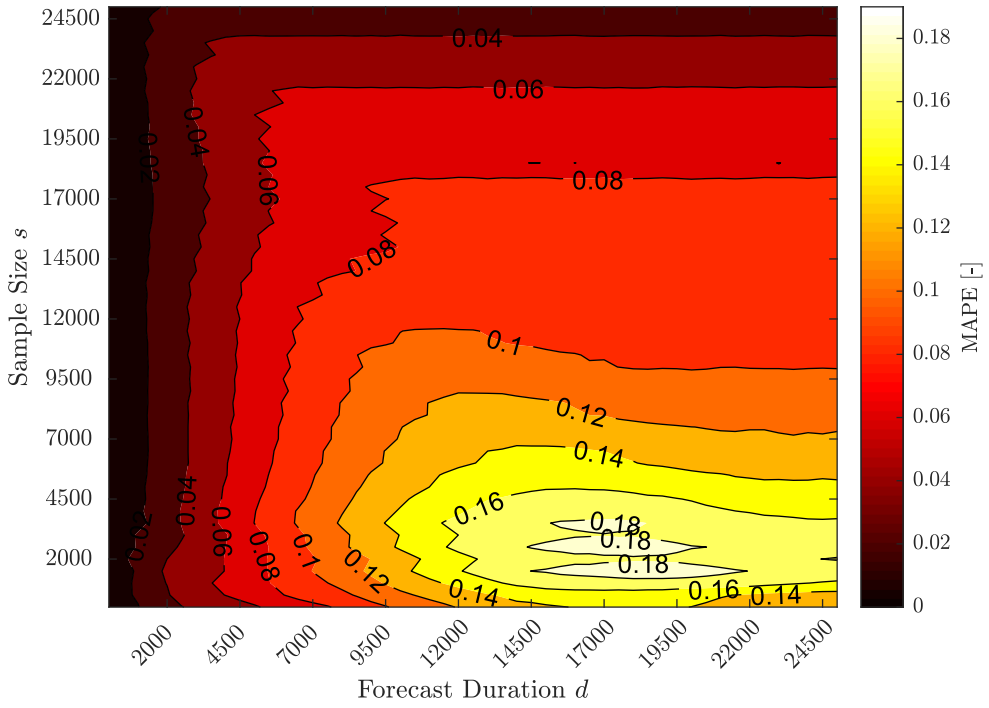


Figure 5.48: MAPE for tube 251H with varying sample size s and forecast duration d.

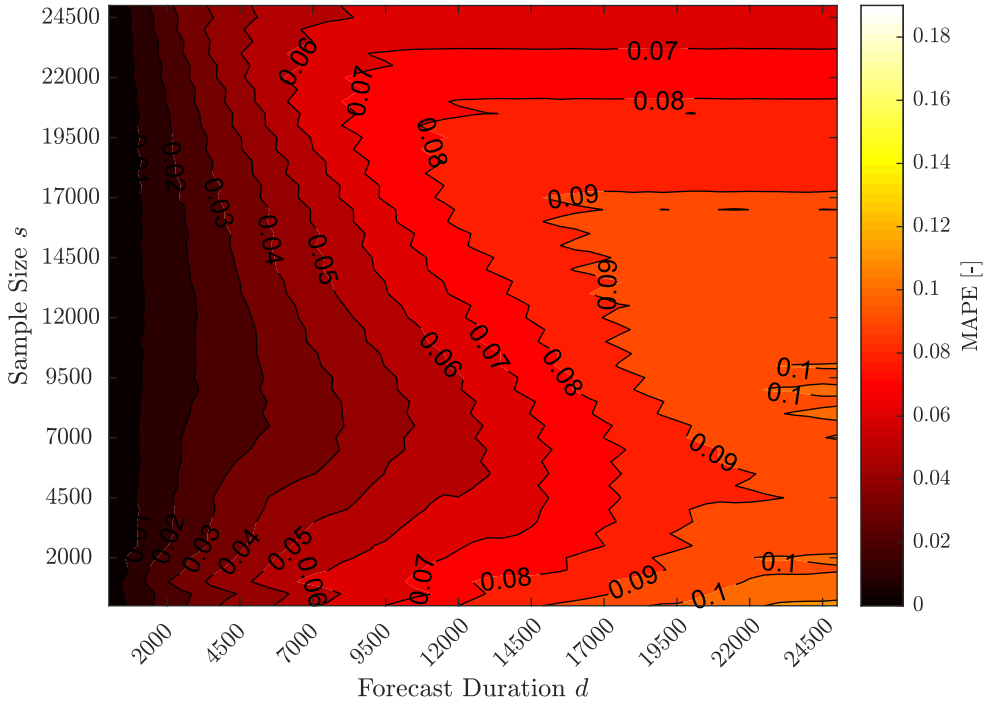


Figure 5.49: MAPE for tube 258H with varying sample size s and forecast duration d.

5.6.2.5 Discussion of Results

In order to get a better understanding of the results meaning for the daily management of the construction site, we add a time conversion factor, EQ 5.20. Under the assumption, that one injection units processes $b = 40$ boreholes per day the weekly forecast would include 240 boreholes, i. e., $f_{b2t} = 240 \left[\frac{\text{bores}}{\text{week}}\right]$.

$$t\ [\text{weeks}] = \frac{b\ [\text{bores}]}{f_{b2t}\left[\frac{\text{bores}}{\text{week}}\right]} \tag{5.20}$$

Regardless of the sample size a

- a weekly ($b = 240$) grout volume forecast with a $\delta_{\text{MAPE}} < 1$ %,
- a monthly ($b = 4 \cdot 240 = 960$) forecast with a $\delta_{\text{MAPE}} < 2$ %,
- a quarterly ($b = 4 \cdot 4 \cdot 240 = 3840$) forecast with a $\delta_{\text{MAPE}} < 6$ %

is possible in both tubes. For the forecast of the total remaining grout volume, a large sample seems to yield the best results, Figure 5.48 and 5.49. As large samples are not available during early forecasts, dynamic sampling can be used as fall back. These forecast yield a MAPE between 10 percent and 12 percent, Table 5.7.

5.6.3 Forecast with Artificial Neural Networks

This section presents a method to forecast the mean average of grout volumes per segment and the project's total grout volume. The model uses an ANN, which is trained with mean average grout volumes per segment of one tube. It then uses samples from the second tube to forecast future mean average grout volumes per segment and the total remaining grout volume.

Parts of this section have already been published in Backhaus (2021).

5.6.3.1 Model and Results

The distance between the center axis of the two tunnel tubes is only 20 to 30 meters. Despite their closeness, a direct transfer of grout volumes from one tube to the other is impossible; see section 5.6.1. However, suppose the general principle of the grout volumes' progression over the tube length in one tube is similar to that in the other tube. In that case, a function N exists, which predicts future grout volumes from a set of past grout volumes.

Let $v(a, b)$ be the discrete function with $b - a + 1$ points of the progression of grout volumes from tube meter a to tube meter b. Furthermore, let s be the number of sample points, d be the number of points to be forecasted, s and d be mean grout volumes per tube segment with a width of 1 m and i the segment from which a forecast is made, Figure 5.50. Then N

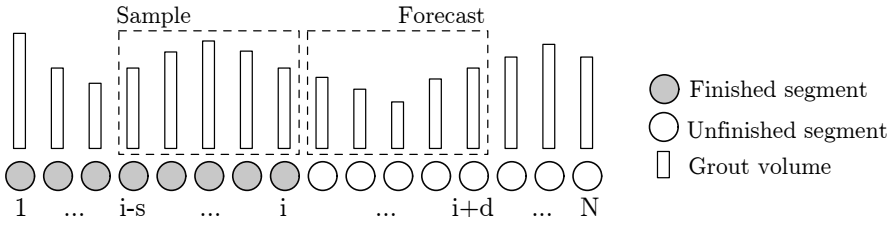


Figure 5.50: Sample and forecast size.

is the forecast function, which predicts the future trend of the grout volume curve $v(a, b)$ from the past progression of the grout volume curve

$$N(v(i-s, i)) = v(i+1, i+1+d). \tag{5.21}$$

If the rules which govern the shape of the grout volume progression are the same, then the knowledge of said rules would allow us to forecast, lets say the grout volume progression of the upcoming 50 tube meters from the grout volume progression of the past 50 meters. The challenger, however, is to identify these rules.

ANNs can learn the rules behind a process from historical data, without a priori knowledge about the process itself, section 4.6. The model uses a FNN, section 4.6.1, with s input nodes, d output nodes and $h_i \in H$ hidden layers with h_i being the number of nodes in the i^{th} hidden layer. The model is trained with the SCGA, section 4.6.2. The SCGA was chosen as a suitable training algorithm because, unlike the Jacobian training method (the MATLAB default method), it can be parallelized on CUDA-enabled *Graphics Processing Unit*s (GPUs)s, which comes with a significant speed benefit.

The MATLAB function `feedforwardnet` is used to build the FNN and the MATLAB function `train` with the option `trainscg` to train it

$$\begin{aligned} \text{FNN}_{\text{untrained}} &= \texttt{feedforwardnet}(s, d, H), \\ \text{FNN} &= \texttt{train}(E_t, O_t, \text{FNN}_{\text{untrained}}, \texttt{trainscg}). \end{aligned} \tag{5.22}$$

The training data for the model originates from tube 251H. Once adequately trained, the FNN is used to forecast future grout volumes from a sample of grout volumes in the same tube. The training data set is built by calculating the mean grout volume per tube segment. This curve is further smoothed by calculating the moving average over k segments. The resulting discrete curve of moving average grout volumes $v_t(i) \in V_t$ per tunnel segment i with n_{tot} points are used to build the set of training data. Let the number of input nodes be s, and the number of output nodes d. Then $T = \{E_t, O_t\}$ is the set with training data, with E_t being a $s \times n$ vector with the input values $e_{t,ij} \in E_t$ and O_t a $d \times n$ vector with the output values $o_{t,ij} \in O_t$ of grout volumes, EQ 5.23. Then the number of possible training elements per vector is $n = n_{tot} - s - d$.

$$\begin{aligned} E_t &= \{e_{t,ij} \in V \mid e_{t,ij} = v(i-1+j), 1 \leq i \leq n_{tot}, 1 \leq j \leq s\} \\ O_t &= \{o_{t,ij} \in V \mid e_{t,ij} = v(s+i-1+j), 1 \leq i \leq n_{tot}, 1 \leq j \leq d\} \end{aligned} \tag{5.23}$$

If the injected grout volume is zero ($o_{ij} = 0$), then the segment has intentionally not been injected. Data sets, which include such data points in their output set, are removed. This is because intentionally not injected grout volumes cannot be forecasted from prior injection volumes

$$\begin{aligned} E'_t &= E_t \setminus M, M = \left\{ e_{t,i\cdot} \in E_t, \forall i \mid \bigcup_{j=1}^{n} e_{t,ij} = 0 \right\}, \\ O'_t &= O_t \setminus M, M = \left\{ o_{t,i\cdot} \in O_t, \forall i \mid \bigcup_{j=1}^{n} e_{t,ij} = 0 \right\}. \end{aligned} \tag{5.24}$$

The model is used to calculate forecasts O_f in tube 258H from sample values E_f of the same tube. The indexing for O_f and E_f corresponds with each other

$$O_f = \mathrm{FNN}(E_f), \tag{5.25}$$

i. e., a forecast is made at each tube meter i for the upcoming d tube segments and the sample used for this forecast contains the last s boreholes including the current position i.

With this model, the distance which can be forecasted with one iteration depends on the number of output neurons of the ANN. One output value represents the moving average grout volume for one segment in the tunnel. Hence an ANN with 50 nodes in the output layer calculates a forecast over 50 meters.

If one wants to extend that distance either the number of nodes in the output layer is increased, i. e., a retraining of the FNN must be performed. Or the output of the first iteration k is used as input for the second iteration $k+1$, i. e., they are concatenated such that

$$E_f^{k+1} = O_f^k. \tag{5.26}$$

The ANN expects exactly s input nodes and computes exactly d output nodes. Hence, the concatenation in EQ 5.26 works only if $s = |E_f^{k+1}| = |O_f^k| = d$. A work around is required for the case that $|E_f^{k+1}| \neq |O_f^k|$. If $|O_f^k| > |E_f^{k+1}|$ only the last $|E_f^{k+1}|$ nodes of O_f^k are being used, and if $|O_f^k| < |E_f^{k+1}|$ then O_f^k must be supplemented by $n_{sup} = |E_f^{k+1}| - |O_f^k|$ values S, which stem either from measured results M or earlier forecast loops $p \leq k$. Hence,

$$E_f^{k+1} = \begin{cases} o_j \in O_{f,i}, \text{ with } j : |E_f^{k+1}| - |O_f^k| \leq j \leq |E_f^{k+1}| & \text{, if } |O_f^k| > |E_f^{k+1}|, \\ O_f^k & \text{, if } |O_f^k| = |E_f^{k+1}|, \\ S = M \cup O_f^{p \leq k} & \text{, if } |O_f^k| < |E_f^{k+1}|. \end{cases} \tag{5.27}$$

The number of hidden layers and their nodes is determined with a parameter study. For the first experiment the number of nodes in the hidden layers is selected according to Livshin (2019, p. 10) who recommends $h = 2 \cdot s + d$ nodes per hidden layer. Over 180 parameter sets are analyzed.

Neither the training method of the ANN nor the ANN itself takes boundaries for the output values into account. Therefore impossible values, such as negative injection volumes, might be calculated when forecasting with the ANN, EQ 5.25. To partly compensate for this

deficiency, a post-processing step is implemented directly after the evaluation of the ANN. During this step, the function P is applied to the forecasted results

$$O_f = P(\mathrm{FNN}(E_f)). \tag{5.28}$$

Within P, each output value is checked for grout volumes that are out of a common sense boundary and adjusted accordingly. The following two rules are applied in sequence:

R1 If no injection had been planned for the segment j, then the forecasted grout volume o_j is set to zero.

R2 If the forecasted grout volume o_j is smaller than 10 liter but not zero, it is set to the mean value of the last ten injections, EQ 5.29. The volume of 10 liters is the volume absorbed by the borehole if the packer is fixed in the middle of the borehole, EQ 5.10.

$$\begin{aligned} o_j &= \bar{o}_j, \forall o_j \in \{x \mid x \neq 0 \wedge x < 10\} \\ \text{for } \bar{o}_j &= \frac{1}{j-p} \sum_{i=p}^{j-1} o_i \text{ with } p = \begin{cases} j-11 & \text{, if } j-11 > 0 \\ 1 & \text{, if } j-11 \leq 0 \end{cases} \end{aligned} \tag{5.29}$$

Nomenclature of Artificial Neural Networks A nomenclature is introduced to provide an easy to read reference on the parameters that change between different FNNs. A network name is composed of four elements, namely: k, which is the width of the moving average used to smooth the input data; e, which is the number of input nodes of the network; o, which is the number of output nodes of the network; l, which is the number of hidden layers and the number of nodes on each of those hidden layers.

An example best describes this: The network k25_i100_o50_ffwd10x250 denotes a network for which the data set had been smoothed using the moving average with $k = 25$ segments. It has 100 nodes on its input layer and 50 nodes on its output layer. The hidden layers have 250 nodes each, and 10 of those hidden layers exist. The string *ffwd* signals, that a FNN had been used.

Training Data, Settings of Training Algorithm and Result Data Format In the framework of this study, the training data for the FNN has been extracted from tube 251H. All forecasts are performed for tube 258H.

With one exception, the MATLAB default parameters are being used for the `train` function: The maximum number of epochs e_{max} defines the number of iterations after which the algorithm stops to train the network. The default value 1,000 proved to be much too small, which results in the algorithm to stop too early. For this study, it has been set to $e_{max} = 100{,}000$ to assure that the training is performed long enough to reach a default convergence criterion.

Let the output of all forecasts for a tube be an $|O| \times |N|$ array $r_{ij} \in R$. $o_i \in O$ are the values of the output array. Hence, o_d denotes the forecasted moving average grout volume

in d meters (or segments) distance. N is the number of segments in the tube. Hence, r_{ij} denotes the forecast calculated at segment i for segment $i+j$; see Figure 5.50.

For the evaluation of the analysis results, the MAPE is calculated over three different sets of values. First the MAPE per forecasted meter, second the MAPE per tunnel meter or segment and third the MAPE over all forecasted points

$$\delta_{\text{MAPE}} := \begin{cases} \delta_{\text{MAPE}}(r(i,\cdot)) & \text{, for MAPE over all segments per forecasted meter } i, \\ \delta_{\text{MAPE}}(r(\cdot,j)) & \text{, for MAPE over all forecasted points per tunnel segment } j, \\ \delta_{\text{MAPE}}(r(\cdot,\cdot)) & \text{, for MAPE over all forecasted points.} \end{cases} \tag{5.30}$$

Because the training of ANNs with many nodes is very computation-intensive, two measures to minimize computation time have been implemented. First GBPlan makes use of the GPUs installed in the two computers used for this study; see Table C.6, which comes with a significant speed benefit when compared the performance CPU provide for the solution of the same problem. Furthermore, a *Job Distribution System* (JDS) has been implemented, which distributes jobs for parameter studies between the two systems. The main task of the JDS is to avoid that several systems train the same network. The JDS stores the input and result files for the calculation in a Microsoft OneCloud drive (Microsoft, 2020). When starting a calculation, a temporary lock-file with the job's name is created and stored in the same cloud folder. In order to avoid training the same network twice, the JDS only distributes jobs to GBPlan, if they meet two conditions. (1) There must be no temporary lock-file of that job in the cloud folder, and (2) if a result file for that network exists in the cloud folder, it must be older than a user-defined date. The reliability of the JDS depends on the network speed. If the synchronization of the lock-files takes too much time, the JDS fails, i. e., it cannot guarantee that the same network is not trained twice. For the given set-up, however, this has never been the case.

GBPlan Implementation The model's functionality has been implemented in the functions of the GBPlan package `GBGrouting_Neural`. `GBGrouting_Neural.run_batch` starts a batch job for parameter studies. The function trains the ANNs and saves the results to the system's hard drive. This training is done based on a set of input parameters that define the single jobs' details. This set is defined in `GBGrouting_Neural.define_nnset`. `GBGrouting_Neural.define_nnset` is further used by `GBGrouting_Neural.plot_batch` to load the trained ANNs, compute the forecast and visualize them in result plots.

5.6.3.2 Forecast of Moving Average Grout Volume per Segment

Influence of Sample Moving Average Width k on Forecast of Moving Average Grout Volume per Segment Smoothing the input data can significantly influence the ability of the network to produce fitting results. The models presented in this study use the moving average with a width of $k \in \{11, 25, 35\}$ as a smoothing function. Figure 5.51 shows the results of a forecast of the moving average grout volume per segment of three networks

with an equal configuration (i50_o50_ffwd10x250) but a different width k of the moving average function. With $o = 50$ the FNN performs a forecast for the next 50 m, i. e., 50 segments, per iteration. The graphs show the results of one iteration per tunnel segment of tube 258H. The top diagram gives the MAPE over the forecasted distance $o = d = 50$. The middle and bottom diagram shows the full (middle) and scaled (bottom) MAPE respectively, for all forecasts started along the tube, i. e., each data point shows the MAPE over 50 points.

The best results are achieved with the highest k-factor: $k = 35$. However, this good performance is expected; the largest error yields $k = 25$ and not, as one might suggest, $k = 11$. Table 5.8 shows the MAPE of all points of a selection of forecasts from a parameter study. It is in the range of roughly 160 to 166 percent for $k = 11$, 290 to 330 percent for $k = 25$ and 45 to 60 percent for $k = 35$. A summary of all results is listed in table C.1 and C.2.

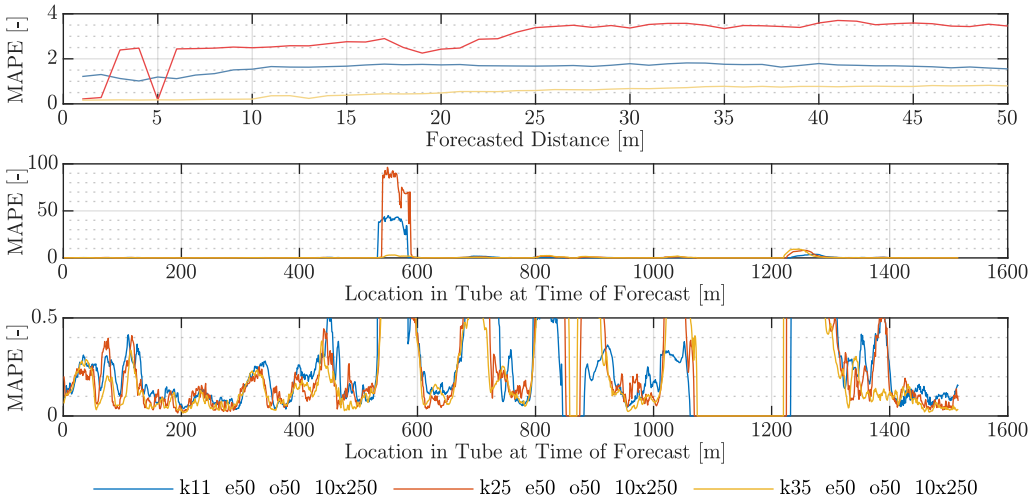


Figure 5.51: Influence of moving average width k. The results are calculated with network configuration k_i50_o50_ffwd10x250 with $k \in \{11, 25, 35\}$. While the MAPE over the forecasting distance shows only small changes (top), the MAPE per tube meter has a significant peak around tube loaction 500-600 m (middle). The zoom shows large variations of the MAPE over the whole forecasting period (bottom).

Table 5.8: Selected results of parameter study of forecast of mean average grout volume per segment. Shown is the MAPE of different ANNs. Varied parameters are the number of network nodes, hidden layers and moving average width.

Neural Network			k		
input	output	hidden layers	11	25	35
50	50	3x250	164	292	59
50	50	10x250	161	292	54
100	50	10x250	161	304	65
100	50	20x250	154	301	67
50	50	10x600	165	311	58
100	50	10x600	166	331	57
100	50	20x600	160	314	66

MAPE in percent

Influence of ANN Configuration on Forecast of Moving Average Grout Volume per Segment The network's configuration, i. e., the number of nodes on its input, output and hidden layers and the number of layers in total influence the results in a range of about 10-20 percent. For this study, the best results are achieved with the network configuration k35_i50_o50_ffws10x250 (Table 5.8) and yield a MAPE of all forecasted points of 54 percent. The MAPE of this configuration is plotted in Figure 5.51.

Accuracy of Moving Average Grout Volume Forecast Results over Tunnel Length The MAPE changes significantly over the tunnel length. While it reaches a value of nearly 10,000 percent for $k = 25$ around segment 500-600, Figure 5.51, middle, it shrinks to under 10 percent around segment 200, Figure 5.51, bottom. Figure 5.52 shows a histogram of all MAPEs of the tube segments. The histogram uses 200 bins with a width of approximately 4.7 percent. The 95%-quantile is at $q_{95} = 241$ %, the 90%-quantile is at $q_{90} = 143$ % and the median at $\tilde{x} = 12$ %, i. e., around 50 percent of the forecasted mean average grout volumes per tube segment are being forecasted with $\delta x_{\mathrm{MAPE}} \leq 12$ %.

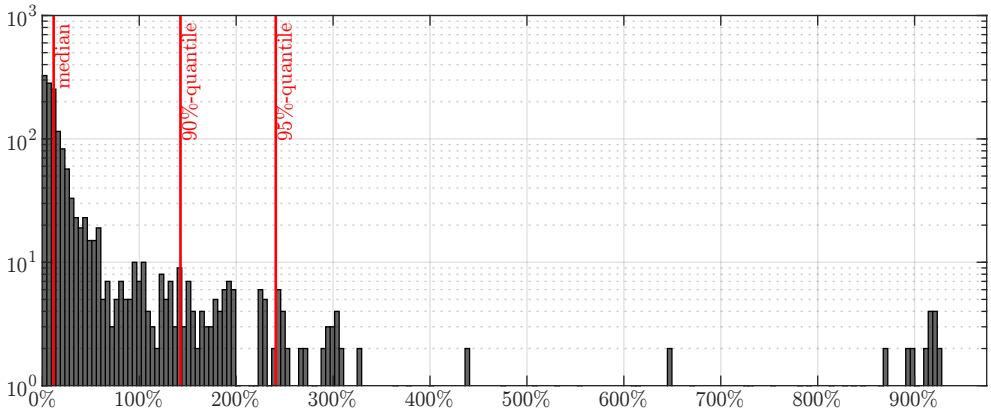


Figure 5.52: Histogram of MAPE values over all tube segments. Results of ANN configuration k35_i50_o50_ffwd10x250 were used.

5.6.3.3 Forecast of Total Grout Volume

The total grout volume is calculated as the sum of already injected grout volume and the forecasted remaining grout volume, section 5.6.3.1. Figure 5.53, 5.54 and 5.55 show exemplary results of forecasts with different configurations of the FNN. Depicted is the relative error δx for the forecast of total grout volumes. The abscissa denotes the location in the tube from which the forecast starts. Hence, the boreholes to the left of a value on the abscissa have already been finished. Those to the right are forecasted.

For all three diagrams, the δx decreases with an increasing location in the tube. This behavior is as expected because the extrapolation error necessarily shrinks with shrinking forecasting distance. The relative error alternates around the real value for the total grout volume ($\delta x = 0$). Interestingly the relative high MAPE for the prediction of mean average grout volumes per segment discussed in the section before, section 5.6.3.2, seem to cancel each other out. The total grout volume forecast results are more accurate than those of the mean average grout volume per segment. This becomes apparent, when comparing MAPE curves with the same FNN configuration but different smoothing function factor k. While the forecast of mean average grout volume per segment with $k = 25$ yield by far the highest MAPE, Table 5.8, the forecast of the total injection volume is on a par with those of the others ($k = \{11, 35\}$) and in some cases even falls below them, Table 5.9.

The best results were computed with a FNN configuration of k25_i50_o50_20x600 (11 percent), k25_i50_o50_10x250 (13 percent) and k11_i50_o50_3x150 (13 percent). A general trend that the quality of results increases with more extra nodes and layers seems to exist only among ANNs with a relatively small number of layers and nodes.

The MAPE of the predictions of total grout volumes is, in the best case, only 11 percent and varies between 11-20 percent for a wide range of network configurations. With the forecast of the total grout volume being dependent on the mean average grout volume per segment, these results are astonishingly good. There is a tendency of the model to

yield either a result with a very high or a very low MAPE for single forecasts. When combined to a MAPE for all forecasted points, these extremes level each other out to form the $\delta x_{\text{MAPE}} = [11\ \%, 20\ \%]$ pointed out above. For the practitioner, this behavior poses the challenge to estimate whether the current forecast of the expected grout volume at the end of the project is which of the two options: accurate or extreme. A possible way out of this dilemma is to average all previous forecast results, such that

$$\bar{v} = \frac{1}{i} \cdot \sum_{j=1}^{i} v_i \tag{5.31}$$

with

v_i : grout volume at forecast i.

While this method lets the result not converge against the real value, it cancels out intermediate extremes and allows for a more smooth allocation of building materials on the construction site; see dashed lines in Figure 5.53, 5.54 and 5.55.

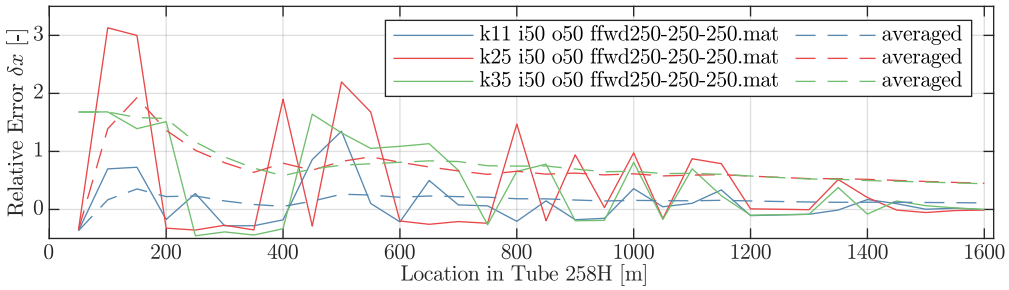


Figure 5.53: Relative error of forecast of total grout volume with the ANN configuration k_i50_o50_ffwd3x250 with $k = \{11, 25, 35\}$.

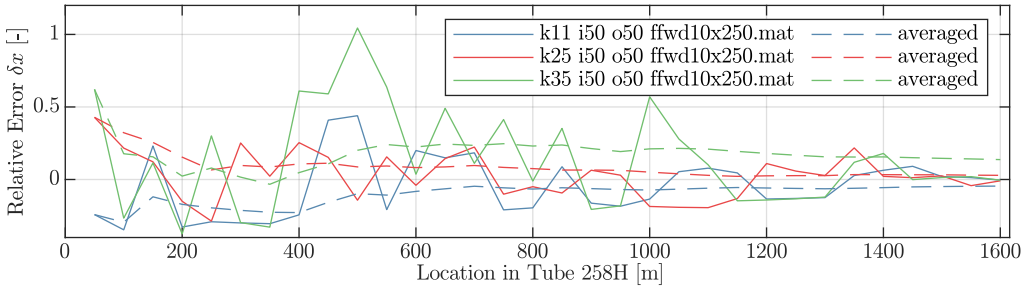


Figure 5.54: Relative error of forecast of total grout volume with the ANN configuration k_i50_o50_ffwd10x250 with $k = \{11, 25, 35\}$.

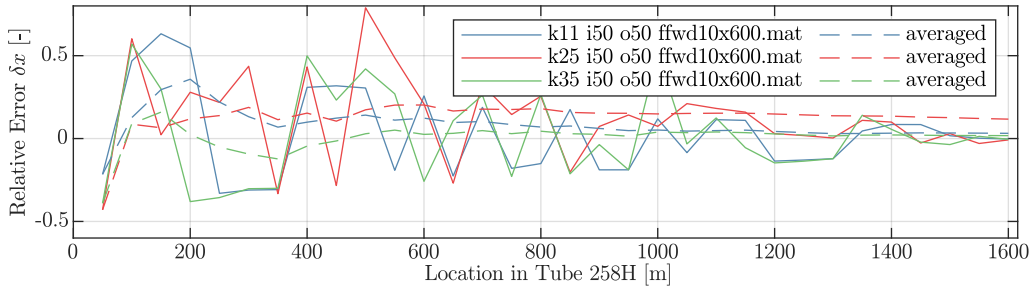


Figure 5.55: Relative error of forecast of total grout volume with the ANN configuration k_i50_o50_ffwd10x600 with $k = \{11, 25, 35\}$.

Table 5.9: Selected results of parameter study of total grout volume forecasts. Shown is the MAPE of different ANNs. Varied parameters are the number of network nodes, hidden layers and moving average width.

Neural Network			k		
input	output	hidden layers	11	25	35
50	50	3x150	13	39	52
50	50	3x250	26	66	61
50	50	10x150	18	16	15
50	50	10x250	18	13	28
50	50	10x450	20	27	42
50	50	10x600	21	22	22
50	50	20x150	17	19	16
50	50	20x450	19	19	25
50	50	20x600	16	11	16

MAPE in percent

5.6.3.4 Discussion of Results

With over 50 percent of forecasts of mean average grout volume per segment over a distance of 50 m having a MAPE smaller than 12 percent the applied forecasting method seems to work somewhat. However, the many predictions with high MAPEs indicate that the ANNs have not been sufficiently trained. With network configurations having only a minor influence on the overall results; see Table 5.8, the reasons for this may be an insufficiently small set of training data. It is suited to perform a 50 m forecast for many of the tunnel's sections; however, it dramatically fails in others.

5.6.4 Discussion of Results of the Presented Grout Volume Forecast Methods

Three methods to forecast grout volumes for the tunnel Feuerbach project have been presented. (1) with a cross-correlation, (2) by sampling and extrapolation of grout volume using statistical methods and (3) by training an ANN on the mean average grout volumes per segment of one tube and then using the trained ANN to forecast future grout volumes of the other tube. The first method did not produce sufficiently accurate results. With the second method, a forecast of total remaining grout volume with a MAPE of 10 to 12 percent was achieved. The third method predicted the total grout volume of the second tube with a MAPE of 11 percent; however, required a complete data set from the first tube as training data.

In total, method (2) and (3) seem to yield equally good results. This statement, however, is only valid for the MAPE including all forecasts. When comparing the relative error of intermediate results, ANNs tend to produce extreme grout volumes and require some post-processing to level these out. Added to this is the difficulty that identifying an adequate network layout is difficult in a priori analysis. Therefore, method (2), using traditional statistical methods, seems to be best fitted to perform the forecast. Nevertheless, the results yielded by the ANNs are astonishingly good, when keeping in mind that they were computed based on training data originating from a parallel tube. Further investigations should be performed to (a) collect more training data to produce better-trained networks and (b) analyze other types of networks regarding their ability to approximate the mean average grout volume function.

6 Project Duration Forecast

Without statistical methods, attempts to improve a process are hit or miss, with results that usually make matters worse.

Deming (1986)

This chapter describes models for the project time forecast implemented in GBPlan. It can be roughly divided into three sections. First, the single server model or *Single Server Production System* (SSPS) is being described in section 6.1. The SSPS is a core component of all coming models. It represents one injection pump, which is the smallest unit of the production system. In section 6.2 the SSPS is extended to several of those pumps. Such a multi-server model or *Multi Server Production System* (MSPS) can simulate a complete injection unit, e. g., three injection pumps in a container mounted on a truck. In the final section 6.3, simulation results for project time forecasts which have been produced with the MSPS are being discussed.

6.1 Time-Discrete Single Server Model (SSPS)

This section describes the SSPS as implemented in GBPlan. The model draws from experiences made in the stationery industry and serves as a basis for other, more advanced models used within the GBPlan framework. In the first section, a firm overview of the model's components is given. The sub-sections contain detailed descriptions of the functionality of each of these components. A special problem for evaluating the model's simulation results is to establish comparability with measurements from the construction site. This problem and its solution is being discussed in section 6.1.1.7. The second section deals with how samples are taken and processed for the simulation; see section 6.1.2. In the third section, a workaround for a case is described in which the sample does not contain enough information on all processes to perform a simulation; see section 6.1.3. The SSPS serves as the basis for the more developed MSPS, which is described in section 6.2.

6.1.1 Model

The simplest model provided by GBPlan for the project time forecasts is the SSPS. In this model, a single injection pump is injecting boreholes one after another. The SSPS assumes a production system set-up like in the stationery industry, where machines stay in place while the semi-finished parts are transported through the production line. In this

case, these semi-finished parts are the boreholes. Figure 6.1 shows a sketch of such a SSPS. When the machine, which represents one injection pump, is free then a borehole is taken from storage s_1, transported and loaded to the machine, processed and unloaded to storage s_2. The duration required for a borehole to move from s_1 to s_2 is called the tact duration $\delta t_{tact} = \delta t_{mov} + \delta t_p + \delta t_u$; see Little (1961) and Little and Graves (2008).

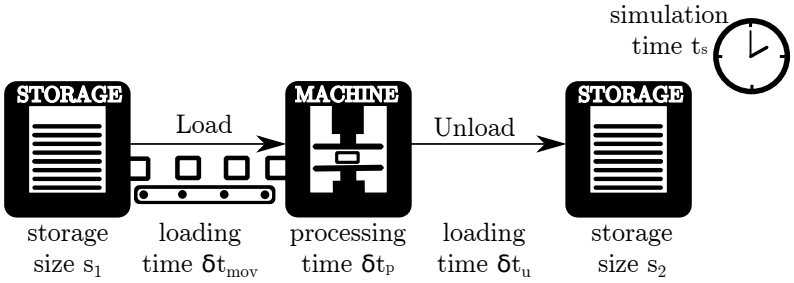


Figure 6.1: Single Server Model.

While each borehole has a planned number of injections not every borehole is injected as planned. Major and minor disturbances, or malfunctions, may result in lengthy repair and maintenance processes (major) or boreholes to be injected more often than planned (minor). The duration spent with disturbances δt_{dist} is the sum of the duration of major and minor disturbances, hence

$$\delta t_{dist} = \delta t_{maj} + \delta t_{min}. \tag{6.1}$$

In order to reduce downtime due to malfunctions, regular maintenance processes are being performed. Of these, the calibration process interferes the most with the production process. The duration required to calibrate the system is denoted as δt_{cal}. If many injections are being simulated, then the simulated project duration might include idle time intervals δt_{idle}, e. g., over public holidays. These idle times do not alter the tact duration δt_{tact}; however, they add to the total project duration, also called the total project time δt_{tot}. One-time step in the simulation Δt is the time that passes between the loading of two boreholes to the system. It is calculated as the sum of the tact duration δt_{tact}, the time to calibrate the machines δt_{cal}, the duration spend for handling disturbances δt_{dist} and the idle time δt_{idle}, i. e.,

$$\begin{aligned} \Delta t &= \delta t_{idle} + \delta t_{cal} + \delta t_{tact} + \delta t_{dist} \\ &= \delta t_{idle} + \delta t_{cal} + \delta t_{mov} + \delta t_p + \delta t_u + \delta t_{maj} + \delta t_{min}. \end{aligned} \tag{6.2}$$

With this the simulation time after the next time step $t_{s,i+1}$ can be calculated as

$$t_{s,i+1} = t_{s,i} + \Delta t. \tag{6.3}$$

The production system finishes its work once all n boreholes have been transferred from storage s_1 to storage s_2. The time required for this is calculated as

$$\delta t_{tot} = \sum_{i=1}^{n} \Delta t_i, \tag{6.4}$$

and the project time at project completion is obtained from

$$t_{s,tot} = t_0 + \delta t_{tot} \tag{6.5}$$

with

t_0 : Start time of the simulation, e. g., 25.01.1979 14:00.

Albeit formally correct, EQ 6.3 does not give insight into the mutual interrelations of the individual processes. The following sections give a detailed description of each of the processes to take those interrelations into account. In doing this, the parameters of EQ 6.3 are formalized, and the internal mechanics of GBPlan are further outlined.

6.1.1.1 Minor Disturbances

Planning determines the number of injections per borehole. This planning is relatively rigid and, with few exceptions, does not change. However, unplanned changes occur frequently due to minor disturbances, i. e., something goes wrong, and the injection process must be started anew. Albeit the reason for these disturbances can only be speculated about, it has been shown in section 5.5.8 that they tend to accumulate at some points, either in time or location, of the project.

The challenge is to guess how often a borehole is being injected without knowing the cause's nature, i. e., what exactly caused the minor disturbance that led to a new injection of the same borehole. GBPlan offers two modes of operation to handle this problem. First, the number of injections as planned, i. e., a fixed number of injections, is being performed for each borehole, regardless of contrary values in the sample. Second, the number of injections as per Markov chain, i. e., GBPlan guesses the number of injections per borehole by sampling the number of times boreholes have been injected in the past from a sample S. From this sample the kernel K of a Markov chain is built and converted into a stochastic vector v_s. Each element $v_{s,i} \in v_s = \{v_{s,1}, \ldots, v_{s,n}\}$ gives the probability that a borehole is being injected i times. Consequently $\sum_{i=1}^{|v_s|} v_{s,i} = 1$. The vector is further used to simulate the real distribution $v_s \sim R$ of several injections per borehole; see also section 4.3. This operation is implemented in `MCS_TimeForecast.get_xLevel_per_Bore_with_discreteTimeMarkovChain`. The function creates a $1 \times n$ vector X with n being the number of boreholes to be injected and x_i denoting the number of times the i^{th} borehole is being injected. Computing x before the simulation instead of calculating each item x_i when required brings the advantage that the problem can be easily vectorized, thus reducing overall computation time. In the example shown in EQ 6.6, the chance that a borehole is injected one time is 50 percent, two times 40 percent and three times 10 percent.

$$\text{Example: } S \Rightarrow K = \begin{Bmatrix} 0.5 & 0.5 & 0 \\ 0 & 0.8 & 0.2 \\ 0 & 0 & 1 \end{Bmatrix} \Rightarrow v_s = \begin{Bmatrix} 0.5 \\ 0.4 \\ 0.1 \end{Bmatrix} \Rightarrow X \tag{6.6}$$

When using this method to take minor disturbances into account, it becomes apparent that the disturbance itself has no duration and does not add to total project time. It is the additional number of injections, which extends the processing time. This is expressed in EQ 6.13.

6.1.1.2 Major Disturbances

Major disturbances are malfunctions that entail costly repairs and maintenance processes. These processes are being logged in the eguana cloud as separate items. The probability of their occurrence p_{maj} is estimated by counting the number of their occurrence n_{maj} and dividing them by the total number of injections n_{inj}, as in

$$\begin{aligned} p_{maj} &= \frac{n_{maj}}{n_{inj}} \\ &\text{with} \\ n_{maj} &= \text{number of major disturbances in the sample,} \\ n_{inj} &= \text{number of injections in the sample.} \end{aligned} \tag{6.7}$$

With the probability of occurrence per injection known, GBPlan draws a random number $r_0 \sim \mathcal{U}(0,1)$ for each injection. If this number is smaller or equal than p_{maj} the injection is followed by a repair and maintenance process. Otherwise, nothing happens. In the case of repair or maintenance, the process's time is computed with the processes' ICDF, which is extracted from the sample. These relationships are implemented by the function f_{maj}, which computes the duration of major disturbances. The function f_{maj} is evaluated for each injection i, which is expressed as

$$f_{maj,i} = \begin{cases} \text{ICDF}(r, S) & \text{, if } r_0 \leq p_{maj}, \\ 0 & \text{, otherwise.} \end{cases} \tag{6.8}$$

Because each borehole can be injected several times and a major disturbance might follow each injection, the time being spent with major disturbances δt_{maj} is calculated as the sum of all major disturbances. Furthermore, the calibration process, although not productive, is an injection and as such may be followed by a major disturbance

$$\delta t_{maj} = \begin{cases} \sum_{i=1}^{n} f_{maj,i} & \text{, if no calibration process occurs,} \\ \sum_{i=1}^{n+1} f_{maj,i} & \text{, if a calibration process occurs.} \end{cases} \tag{6.9}$$

The calibration processes is discussed in section 6.1.1.6.

6.1.1.3 Loading and Unloading

While unloading a machine is assumed to happen in an instant

$$\delta t_u = 0 \tag{6.10}$$

and is therefore excluded from the simulation the loading time δt_{mov} can be interpreted as the time it takes to move the injection nozzle from one borehole to the next one. It is calculated by the function f_{mov} in EQ 6.11. If the mean or median values are used to describe the movement process's duration, then f_{mov} takes the form of a constant value. Otherwise the ICDF is used, which depends on a sample S of past process times and a random value $r \sim \mathcal{U}(0,1)$.

$$\begin{aligned} &\delta t_{mov} = f_{mov} \\ &\text{with} \\ &f_{mov} = \begin{cases} \text{const.}, & \text{if } f_{mov} = \bar{S}, \\ \text{const.}, & \text{if } f_{mov} = \tilde{S}, \\ \text{not const.}, & \text{if } f_{mov} = \text{ICDF}(r, S). \end{cases} \end{aligned} \tag{6.11}$$

6.1.1.4 Injection Time

The duration of a single injection depends on the function $f_{inj,1}$. In GBPlan $f_{inj,1}$ can take the form of a constant value, i. e., the mean or median value of a set of sample boreholes S or an ICDF, which depends on a sample S and a random value $r \sim \mathcal{U}(0,1)$, i. e.,

$$f_{inj,1} = \begin{cases} \text{const.}, & \text{if } f_{inj,1} = \bar{S}, \\ \text{const.}, & \text{if } f_{inj,1} = \tilde{S}, \\ \text{not const.}, & \text{if } f_{inj,1} = \text{ICDF}(r, S). \end{cases} \tag{6.12}$$

6.1.1.5 Processing Time

The processing time $\delta t_{p,j}$ of the j^{th} borehole calculated as EQ 6.13, includes everything which happens between mounting the injection nozzle to the borehole until it is being removed. Hence, it represents the duration of one injection $f_{inj,i}$ and the duration of possible disturbances of the injection process. These disturbances are divided in the duration of major disturbances f_{maj}, section 6.1.1.2, and minor disturbances f_{min}, section 6.1.1.1. Between two injections of the same borehole, the injection nozzle is removed and reinserted into the borehole. This process is similar to the movement of the nozzle from one borehole to the next. Hence, the calculation of its duration is performed according to

$$\delta t_{p,j} = \sum_{i=1}^{X(j)} (f_{inj,i} + f_{maj,i} + f_{mov})$$

with

$$\begin{aligned} X &: \text{ see EQ 6.6,} \\ j &: \text{ the j}^{\text{th}} \text{ borehole in } X\text{, see section 6.1.1.1,} \\ f_{maj,i} &: \text{ see EQ 6.9,} \\ f_{mov,i} &: \text{ see EQ 6.11,} \\ f_{inj,i} &: \text{ see EQ 6.12.} \end{aligned} \tag{6.13}$$

6.1.1.6 Calibration Process

A calibration process is being started at the beginning of each of the 12 h shifts. During the calibration, a test injection is being performed during which the system parameters are being monitored. In case of a deviation between actual and target parameters, additional measures are being performed. An example of such a measure would be the replacement of the injection nozzle.

GBPlan logs the last point in time when the calibration process has been performed. Before each injection, it checks the duration $d_{cal} = t_s - t_{cal}$ between the last calibration t_{cal} and the current time t_s and performs a calibration process if that duration is greater than 12 h, i. e.,

$$\delta t_{cal} = \begin{cases} f_{cal}, & \text{if } d_{cal} \geq 12\,\text{h}, \\ 0, & \text{otherwise.} \end{cases} \tag{6.14}$$

In its general procedure, the calibration process is comparable to the standard injection process. The injection nozzle is mounted to a borehole, section 6.1.1.3, the calibration injection is being performed, and a failure of the system might lead to the calibration process to be repeated, section 6.1.1.2.

The duration the pump is occupied with the calibration process δt_{cal} is calculated via f_{cal}, EQ 6.15. The calibration requires that the injection nozzle is being moved. Furthermore, failure may occur during that process. These processes are handled exactly as if a standard injection is being performed and have been discussed above.

$$f_{cal} = \begin{cases} \text{const.}, & \text{if } f_{cal} = \bar{S}, \\ \text{const.}, & \text{if } f_{cal} = \tilde{S}, \\ \text{not const.}, & \text{if } f_{cal} = \text{ICDF}(r, S). \end{cases} \tag{6.15}$$

6.1.1.7 Idle Time

The handling of idle time δ_{idle} is a special case within the GBPlan framework. Following the definition in section 5.3.6, idle time is time spent with downtime, which cannot be forecasted based on the available data. An example is a management decision to put one of the injection units in stock, e. g., as a reserve if a future decision requires a rapid increase in the production capacity. However, if one would ignore these processes, the project's total duration would be shortened by the length of idle time. This simplification would render a comparison of the simulation results with measured results futile. Hence, the idle times have to be included a posteriori. However, adding the sum of idle process durations to the total project time works only for the one server system, i. e., a system with just one pump.

In preparation for the MSPS, the SSPS implements a functionality, which allows for interweaving idle time intervals of multiple injection units. This is required because the position of the idle time interval on the time-line defines which borehole an injection pump is processing. Systems composed of multiple injection pumps with different sets of idle time intervals would determine which borehole is being injected without taking idle time intervals into account. As boreholes differ – some must be injected only once while others require several injections, section 5.5.8 – this would alter the simulation result.

The idle time δ_{idle}, EQ 6.16, contained in the next time step is a function of the current simulation time t_s, the duration covered by all processes of the current time step δp and a set of predefined idle intervals $ii_i \in I$, where each idle interval ii_i is a tuple consisting of a start time ii_s and an end time ii_e.

$$\begin{aligned} \delta_{idle} &= f_{idle}(t_s, \delta p, I) \\ &\text{with} \\ \delta p^{(0)} &= \delta t_{cal} + \delta t_{tact} + \delta t_{dist} \\ ii &= [ii_s; ii_e] \\ ii_i &\in I \end{aligned} \tag{6.16}$$

After each injection of a borehole, GBPlan determines the process time interval $p^{(k)}$ as

$$\begin{aligned} p^{(k)} &= \left[p_s; p_e^{(k)}\right] \\ &\text{with} \\ p_s &= t_s, \\ p_e^{(k)} &= t_s + \delta p^{(k)}, \\ &\text{and} \\ k &: \text{the current iteration of } p. \end{aligned} \tag{6.17}$$

Then it is checked whether or not the interval intersects with any of the idle time intervals $ii_i \in I$. If the two intervals $p^{(k)}$ and ii_j intersect ($ii_j \cap p^{(k)} \neq \varnothing$), then the duration $\delta ii_j = ii_{e,j} - ii_{s,j}$ is added to $\delta p^{(k)}$, such that $\delta p^{(k+1)} = \delta p^{(k)} + \delta ii_j$. This procedure may result in a chain reaction. When extending the process time $p^{(k)}$ to $p^{(k+1)}$ the case can occur that $p^{(k+1)}$ intersects with yet another idle time interval ii_i which has not been regarded

during the iteration before ($ii_i \cap p^{(k)} = \varnothing \wedge ii_i \cap p^{(k+1)} \neq \varnothing$). An example for such a chain reaction is shown in Figure 6.2. To assure that also those idle time intervals ii_i are added to the total process time, which only intersect with p because the duration of an idle time interval δii_i has been added to p before, GBPlan performs several iterations until no more intersections can be found, i. e., $p_{fin} = p^{(i)}$ with $i = \left\{j \mid p^{(j)} \bigcap I = \varnothing\right\}$.

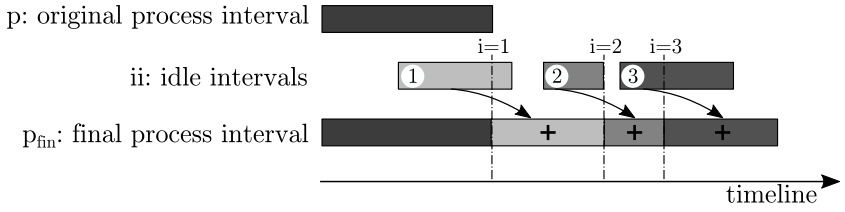


Figure 6.2: Iterative development of total idle time. When a process (p) intersects an idle time interval $ii_i \in I$, the duration of that idle time interval δii_i is added to the end of the total process duration $\delta p_{i+1} = \delta p_i + \delta ii_i$ and ii_i is removed from the set of idle time intervals $I_{i+1} = I_i \setminus ii_i$. This procedure is repeated until no idle time interval exists in I, which intersects with the last step of the process interval $p_i = p_{fin} :\Leftrightarrow p_i \bigcap I = \varnothing$.

While the decision whether or not two intervals intersect is intuitive for the average human, a discrete rule had to be defined and implemented to enable GBPlan to do the same. This rule relies on two arrays: I_s represents an ordered list of start dates and I_e represents an ordered list of end dates of idle intervals ii_i, such that

$$I_i = \{t \mid I_{s,i} \leq t \leq I_{e,i}\}\,. \tag{6.18}$$

When taking into account the definition in EQ 6.17, the set S that contains all intervals I_i which intersect with p can be calculated as

$$\begin{aligned} S = \{ii_i \in I, i \mid & ii_{s,i} > p_s \wedge ii_{s,i} < p_e \\ & \cup\ ii_{s,i} < p_s \wedge ii_{e,i} > p_s \\ & \cup\ ii_{e,i} > p_s \wedge ii_{e,i} < p_e \\ & \cup\ ii_{s,i} < p_e \wedge ii_{e,i} > p_e \\ & \cup\ ii_{s,i} > p_s \wedge ii_{e,i} < p_e \\ & \cup\ ((ii_{s,i} < p_s \wedge ii_{e,i} > p_s) \wedge (ii_{s,i} < p_e \wedge ii_{e,i} > p_e))\}. \end{aligned} \tag{6.19}$$

6.1.2 Sampling and Forecast

The procedure of sampling is illustrated in Figure 6.3. The variable $s \in S$ stands for a sampled process with all its characteristics. The variable $f \in F$ represents a process which has been forecasted based on S. GBPlan uses either a full sample S_{full}, which contains all processes that have been documented up to the point when the forecast is being performed, or a reduced sample S_s, where s denotes the number of fully processed boreholes included in the sample. These boreholes are always the boreholes, which have been processed last.

From these sets of samples, GBPlan computes the input parameters for the project time forecast simulation as described in the sections before and simulates one process $f \in F$ after another until the desired number of boreholes have been processed. GBPlan offers the option to either forecast all remaining boreholes (F_{full}) or a reduced number of f boreholes (F_f), e. g., to schedule the time required for a subsection of the tunnel tube with f boreholes. To improve readability Figure 6.3 has been reduced to sampling with ICDF; however, computation with mean or median values is possible for many processes as well; compare section 6.1.1.

The general procedure of the forecast is as follows: Based on already finished processes, a sample is taken (a). From the sample, the PDFs of the simulated processes are being calculated (b). The PDFs are transformed into ICDFs (c). By introducing a random variable $r \sim \mathcal{U}(0,1)$ the ICDFs are used to draw random process durations $d(r)$ (d); see also section 4.1.2. A forecast of the total, remaining project duration $\Delta t_{remaining}$ is calculated by summing up the so computed process forecast durations $d(f_j)$, such that

$$\Delta t_{remaining} = \sum_{j=1}^{m} d(f_j). \tag{6.20}$$

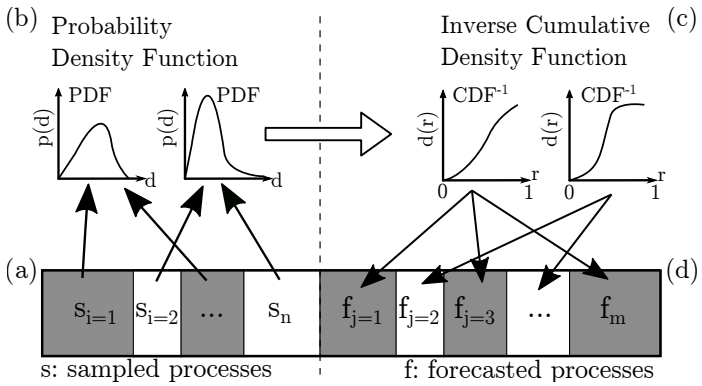


Figure 6.3: Computation of forecast based on sampling from ICDF. (a) A sample s of each process type is used to calculate (b) their individual PDF. (c) From the PDF the ICDF is calculated and further used to (d) draw random durations $d(f) \sim d(s)$.

6.1.3 Dealing with Insufficient Samples

During the simulation, each pump's statistical performance data is used to compute the durations d_i of its different processes p_i. A problem may arise when the sample size $s = |S|$ of a sample S is very small. In that case, it may happen that it does not contain enough information to compute the performance values for all processes, i. e., to build the ICDF or calculate a mean or median value. As a workaround, GBPlan calculates surrogate performance values if the required performance data cannot be retrieved from the sample.

The surrogate values are calculated based on ICDFs of other pumps which might be part of the sample. If such other pumps cannot be found in the sample, then the calculation fails, and GBPlan throws an error.

Let $F_{i,j}^{-1}$ be the ICDF of the process $j \in \{1, 2, \ldots, m\}$ of pump $i \in I = \{1, 2, \ldots, n\}$ with m being the number of processes and n being the number of pumps contained in the sample.

In the common case, i. e., when the $F_{i=a,j}^{-1}$ for an available pump a with $a \in A$ exists, the performance value p_a is randomly drawn from $F_a^{-1}(r)$ with $r \sim \mathcal{U}(0, 1)$ such that

$$p_a = F_i^{-1}(r). \tag{6.21}$$

If performance data for the pumps p_m with $m \in M$ is missing, the surrogate values are computed from the available pumps p_a with $a \in A = I \setminus M$ by drawing a random variable $r \sim \mathcal{U}(0, 1)$. r is used to draw performance values from all $F_{a,j}^{-1}(r)$. The mean of those performance values is then used as surrogate performance value, i. e.,

$$p_m = \frac{1}{|A|} \sum_{a=1}^{|A|} F_{a,j}^{-1}(r). \tag{6.22}$$

Note that in the latter case r is only drawn once regardless of the magnitude of A.

6.1.4 Discussion of Results

The model described in this chapter only represents a single pump. Because an injection unit is composed of at least two pumps, simulation results computed with the presented model could not be validated against reality. The following chapter describes how the single-server model as been improved to solve as multi-server model by implementing the mutual interaction of the injection unit's multiple pumps. Simulations based on live data from the construction site are being carried out and the results are being compared against reality.

6.2 Time-Discrete Multi-Server Model for AC Units (MSPS)

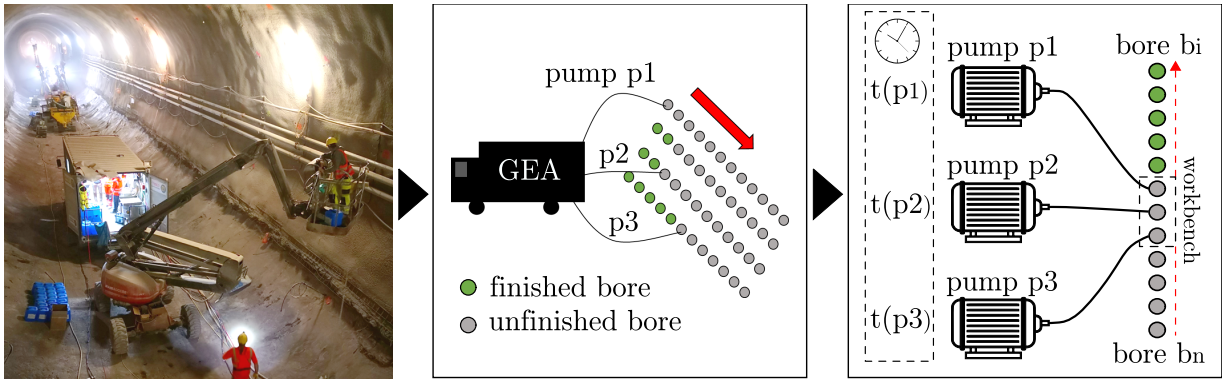
This section describes the MSPS and how it functions to predict the duration one GEA with several AC injection pumps, requires to finish the injections of a given number of boreholes. The model uses Markov chains, introduced in section 4.3; DES, presented in section 2.2.6; and MCS, described in section 2.2.3 for the prediction. It consists of several SSPSs, which are managed within the DES to work together in an integrated manner.

The DES simulates each sub-process, such as injection, translation of packers, and maintenance, separately. The sequence of the processes and the process duration are drawn from a random distribution based on random real-world samples. Exceptions are short or undocumented interruptions labeled Minor Disturbances; see also section 6.1.1.1. Their influence on the construction process is represented in the model by a Markov chain. Since the result of the DES depends on many randomly drawn parameters, a MCS is used to determine the duration which will most likely occur.

Some of the results presented in this section have already been published in Backhaus (2020).

6.2.1 Model

The MSPS models a complex real world scenario, Figure 6.4 a, in which several GEAs perform injections following the pilgrim procedure described in section 5.1.3. During this procedure, pumps have to target different boreholes, which are located all over the tunnel wall in a positional non-consecutive order, Figure 6.4 b. This order follows the logistics rule of *Earliest-Operation-Due-Date* (EODD); see Lödding (2016, p. 510). The simulation model simplifies the complex injection scenarios by assuming a consecutive processing order of the boreholes. Figure 6.4 c shows a sketch of this simulation model. The pumps $p_i \in P$ do not move but wait for the unprocessed boreholes $b_i \in B$ to arrive at their workbench as if they were semi-finished parts on a conveyor belt. The boreholes in B are sorted alphanumerically according to their position identifier, see Table 5.1, which is also the order of their processing in the simulation model, i. e., the *First In – First Out* (FIFO) logistics rule according to Kiran (1998, p. 684) and Lödding (2016, p. 509), is being applied.



(a) real world (b) intermediate model (c) simulation model

Figure 6.4: Model for work duration forecast of a single injection unit with three pumps. The real world scenario (a) is translated into an intermediate model (b) from which the simulation model (c) is being derived. The simulation model reduces the complex injection scenarios applied on the construction site to a simple FIFO logistic rule, according to which boreholes are injected one after another. This setup resembles a production line as can be found in the stationary industry.

Each pump on its own resembles a SSPS as described in the previous section 6.1. This means that, regardless of whether the injection pumps have the same performance parameters or not, the duration of the processing of one borehole may differ for each injection. Which injection pump works at which borehole is determined through the set of rules of the DES implemented in GBPlan. This rule can be roughly summarized as 'First come, first served', i. e., the first free injection pump processes the next unprocessed borehole.

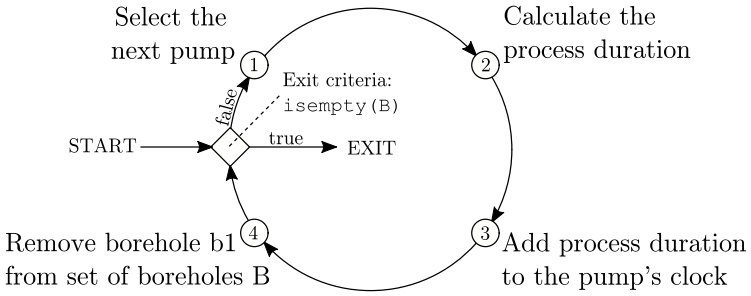


Figure 6.5: Computation cycle for MSPS project time forecast. (1) The first step after handing a set B of unprocessed boreholes to the algorithm is to select the next free pump p_i. (2) Then the time Δt required to process the first borehole b_1 in the set is calculated and (3) added to the injection pump's timer t_i. (4) Finally, b_1 is removed from the set of unprocessed boreholes B. If all boreholes have been processed ($B = \varnothing$) the cycle stops, otherwise it starts anew.

During each of the DES cycles k depicted in Figure 6.5 the algorithm goes through four steps. the injection pumps compete for the right to process boreholes. (1) First, according to EQ 6.23 it is determined which pump p_i is furthest behind in time and thus earns the right to process the next borehole b_1, (2) then the time Δt_1 for processing the next borehole is being computed using EQ 6.2, (3) added to the pumps clock $t_i \in T$ with EQ 6.24, and (4) finally the borehole b_1 is removed from the set of unfinished boreholes B as expressed by EQ 6.25.

$$p_{active} = \begin{cases} |\min(t_i)| = 1 : \{p_i \mid \min(t_i)\} \\ |\min(t_i)| > 1 : \{p_i \mid \min(t_i)\} \cap \{p_i \mid \min(i)\} \end{cases} \tag{6.23}$$

$$t_i^{(k+1)} = t_i^{(k)} + \Delta t_1{}^{(k)} \tag{6.24}$$

$$B^{(k+1)} = B^{(k)} \setminus b_1 \tag{6.25}$$

The four steps are being repeated until the set of unfinished boreholes is empty, i. e., $B = \varnothing$. With this procedure, it is only determined during simulation time which injection pump processes which borehole. If, by coincidence the time counter t_i of two or more pumps are the same, $t(p_i) = t(p_j)$, then the pump with the lowest array index i is chosen, EQ 6.23.

With this procedure, a problem may arise with regard to the integration of idle time intervals described in section 6.1.1.7. Unfortunate chaining of idle time intervals, as in Figure 6.2, may result in too long project durations. This case may arise if the last process performed by a pump intersects with an idle-time interval. The result can be a chain of idle time intervals when the last idle time interval's end date would be used to compute the total project duration. This yields wrong results because the construction project finishes once the last borehole has been processed and not after a subsequent idle time interval has been finished. Therefore a post-processing step is performed, during which it is checked whether or not the last process carried out by one of the pumps had been an idle process. If so, this process is removed from the simulation result.

6.2.2 Integration of Monte-Carlo Method

So far, the algorithm calculates a single project time forecast. Although the result of such a forecast is often accurate enough to provide meaningful assistance for construction managers, see section 6.3.1, an unfortunate drawing of random numbers during the calculation of process durations with ICDFs may result in overly short or long project time forecasts. This problem is solved by the integration of the MCM. Because MCSs are notoriously costly, GBPlan uses the multi-processor architecture of the computation hardware used for this study. Information on hardware and software specifications are presented in Table C.6. GBPlan offers the option to distribute each of the many MCS's iterations to a separate processor on the host system. On the code side this is implemented with the `parfor` structure provided by the MATLAB Parallel Computing Toolbox described in MathWorks (2020).

6.2.3 Wording for Sample Size and Forecast Duration

Even though the model computes forecast durations based on samples of processes, the unit for the sample size s and forecast duration d is given in numbers of boreholes. The logic behind this is that each process is associated with a borehole. Hence, if the sample size is s boreholes, these include all processes associated with the s boreholes, such as movement, injection, or maintenance processes. Concerning the forecast duration d, the number of boreholes shall be forecasted. In other words, the result of the forecast is the duration required to process d boreholes.

6.2.4 GBPlan Implementation

The functionality of the model has been implemented in the functions of the GBPlan package `MCS_Multi_Timeforecast`. With `MCS_Multi_Timeforecast.run_MCSTFC_basic` users are provided with an editable start script. If MATLAB licenses must be shared with other users and a functioning queuing system is not in place, GBPlan offers a solution with `MCS_Multi_Timeforecast.waitForLicence`. The function checks the system at regular intervals for free licenses and starts the simulation once such a license is available.

6.3 Discussion of Results

Five different functions f_{method} or methods to compute performance data from the samples are compared with each other. The functions differ in that they use different approaches to approximate (a) future process durations and (b) the number of injections per borehole, i. e., minor disturbances per borehole. Which approach is used can be derived from the methods' name listed in Table 6.1, in which *mean* indicates that the mean process duration is used to forecast future process durations, see EQ 4.17; *median* indicates that the median process duration is used to forecast future process durations, see EQ 4.16; *cdf* indicates

that the ICDF is used to forecast future process durations, see EQ 4.6; *2Inj* indicates the assumption that each borehole is being injected exactly two times, and *markov* indicates that a Markov chain is used to approximate the number of injections per borehole.

The method mean/2Inj is used as a benchmark because it is closest to the time forecasting method commonly used on the construction site. Construction managers use self-designed spreadsheets to calculate project forecasts. These are usually implemented in MS EXCEL and not designed to include a detailed analysis of the many different process types. Instead, mean processing times for injections $\bar{t}_{\text{inj}}$ are computed and then multiplied with the planned number of injections per boreholes n_{inj} and the number of boreholes d of the next section to estimate the required construction time dt, i. e.,

$$dt = \bar{t}_{\text{inj}} \cdot n_{\text{inj}} \cdot b. \tag{6.26}$$

In section 6.3.1, the convergence behavior of the performed MCSs is discussed. The following section 6.3.2 analyzes the quality of total project time forecasts. The focus of the subsequent section 6.3.3 lies on partial time forecasts, i. e., forecasts over a shorter period. Furthermore, the influence of a reduced set of samples is investigated. Finally, section 6.3.4 connects the discussion of total and partial forecasts to make a statement on the validity and applicability of the presented method.

6.3.1 Convergence of Monte-Carlo Simulation

To minimize computation time and to understand the accuracy of the project time forecast results as a function of the number of MCS iterations i, a convergence study is being performed. The sample size s and forecast duration d are being varied, just as for the investigation of the grout volume forecasts, Figure 5.44. Also, MCSs for different periods with varying start dates calculate the total remaining project duration d.

An example plot of the development of MCS results with growing number of MCS iterations i is shown in Figure 6.6. The plot shows the mean value $\mu(i)$ and standard deviation $\sigma(i)$ of all project duration forecasts, which have been performed up to the i^{th} iteration of the MCS. Note that the sample used for the example produces very poor prediction results, i. e., the forecasted end time is $\approx$ 26.06.19, while real end date is 15.05.19. As this section deals with convergence, the discussion on the quality of prediction results is done in the next section.

The set $D^{(i)}$ in EQ 6.27 contains all forecast durations up to the i^{th} iteration. At the end of each iteration, its project duration forecast result is added to D, see EQ 6.28. With this

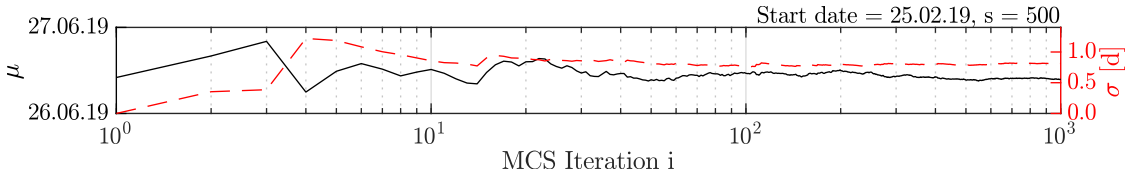


Figure 6.6: Exemplary plot. Project duration forecast of a MCS with sample size $s = 500$ boreholes and 1,000 iterations. Simulated are all AC units in tunnel tube 258H from 25.02.2019 to 15.05.2019. The abscissa shows the number of MCS iterations. The ordinate shows the mean value of the forecasted end date. The mean value $\mu(i)$ (black) and the standard deviation $\sigma(i)$ (red, dashed) of the calculated project completion dates $D^{(i)} = \{d_1, d_2, \ldots, d_i\}$ are shown.

the development of the mean value $\mu(i)$ and the standard deviation $\sigma(i)$ can be plotted as a function of the MCS iteration i, see EQ 6.29 and EQ 6.30.

$$D^{(i)} = \{d_1, d_2, \ldots, d_i\} \tag{6.27}$$

$$D^{(i+1)} = D^{(i)} \cap d_{i+1} = \{d_1, d_2, \ldots, d_i, d_{i+1}\} \tag{6.28}$$

$$\mu(i) = \mu\left(D^{(i)}\right) = \frac{1}{i}\sum_{j=1}^{i} d_j \tag{6.29}$$

$$\sigma(i) = \sigma\left(D^{(i)}\right) = \sqrt{\sum_{j=1}^{i}\left[d_j - \mu(i)\right]^2} \tag{6.30}$$

The example in Figure 6.6 uses a sample of $s = 500$ boreholes of tube 258H and covers a measured project interval from 25.02.2019 to 15.05.2019. Note that the abscissa is logarithmic. The convergence plots of all MCSs performed for this analysis can be found in the appendix, Figure C.11 to C.16.

From the development of MCS results, it can be observed that: (1) the mean value μ changes until around 100 MCS iterations have been performed and stays relatively constant for $i \geq 100$, (2) the same is true for the standard deviation σ, and (3) the standard deviation is in a range between zero and two days.

For a better understanding of the behavior of the distribution of project time forecast results, Figure 6.7 shows three selected histograms of the results of a MCS with 1,000 iterations each. Each bar is calculated according to EQ 6.31 and represents the relative number of observations, i. e., the sum of the areas covered by all bars is less than or equal to 1. The start day varies between the diagrams. In all three cases, the difference between mean value μ and the 95%-quantile q_{95} is about one to three days. Furthermore, the overall spread is quite small. It reaches from three days, Figure 6.7 (top, bottom), to 13 days, Figure 6.7 (mid). The reason for the high spread of the middle histogram is that the calculation draws its sample at 28.01.19, i. e., shortly after a phase of disturbances in the building process. Such disturbances can be identified at points, where the acclivity of the production progress curve changes, e. g., in Figure C.47, bottom.

Thus the sample is not representative of the future process and results in an unrealistic high remaining project duration d.

$$P(d) = \frac{c_i}{N \cdot w_i} \tag{6.31}$$

with

c_i : number of elements in bin

w_i : width of bin

N : total number of elements in input data set

To conclude, about 100 MCS iterations seems sufficient to yield good results. This does not necessarily mean that after 100 iterations, the results of the simulation are correct. However, their alteration has faded enough to ensure that the influence of unfavorably high or low drawings of process times are attenuated.

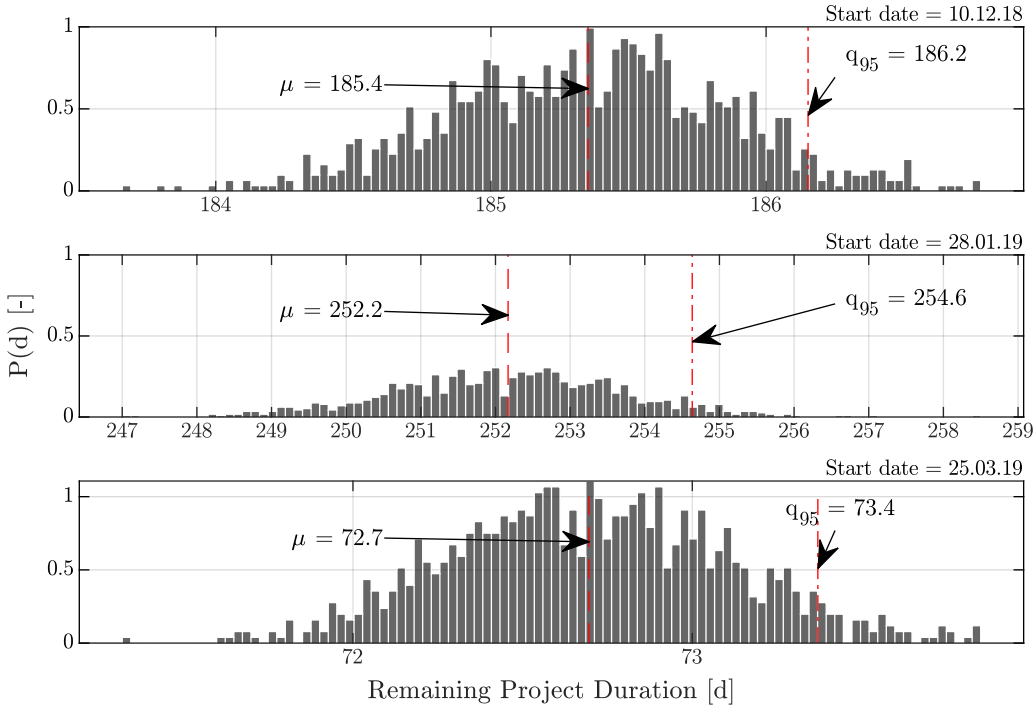


Figure 6.7: Histograms of selected MCS results with varying start dates and a sample size of $s = 500$. In all three cases the mean value μ and the 95%-quantile q_{95} differ roughly by one day, which is acceptable with respect to the forecast duration of 175 to 186 days. For the calculation of $P(d)$ see EQ 6.31.

6.3.2 Total Project Time Forecast

A total project time forecast with maximum sample size is being performed for the tube 251H and 258H. Three AC injection units with three pumps each are being simulated. In

intervals δf of one week, a forecast for the total remaining project duration is calculated. This calculation is done by determining the number of boreholes d to be injected and then simulating the duration $f(d)$ required to process these boreholes. With $f(d)$ and the current date t_c the completion date $t_{\text{fin}} = t_c + f(d)$ can be calculated. Note that the forecasted remaining project duration $f(d)$ likely decreases as the number of not injected boreholes decreases with increasing t_c.

For the sample size s, the maximum number of available samples is being used. Hence, with increasing t_c, the sample size s increases as well. Note that, during long idle intervals, e. g., if the construction site is closed over the Christmas holidays, the increase of t_c might lead to an unchanged sample size s.

The results of the total project duration forecast at different points in time of the project t_p are shown in Figure 6.8 for tube 251H and in Figure 6.9 for tube 258H, respectively. The top diagram shows the completion date forecasted at $t_c = t_p$ and the middle diagram the relative error in percent of that forecast. The bottom picture shows the number of boreholes that have been finished at project time t_p and allows a rough guess of the project development in general. For instance, the interval in tube 251H from April 2018 to June 2018 shows little progress.

With nearly no intersections, the curves can be sorted on an ordinal scale. The order, however, differs between the two tubes, i. e.,

$$\begin{aligned} &\left\{\delta x_{\text{cdf/markov}} \cup \delta x_{\text{mean/markov}} \cup \delta x_{\text{mean/2Inj}}\right\} > \left\{\delta x_{\text{median/markov}} \cup \delta x_{\text{median/2Inj}}\right\} \quad \text{(tube 251H)}, \\ &\delta x_{\text{mean/2Inj}} > \left\{\delta x_{\text{cdf/markov}} \cup \delta x_{\text{median/2Inj}}\right\} > \delta x_{\text{mean/markov}} > \delta x_{\text{median/markov}} \quad \text{(tube 258H)}. \end{aligned} \tag{6.32}$$

The MAPE of the of the results per curve or method, is given in Tabe 6.1. The smallest MAPE in tube 258H is computed with median/markov and the highest with the mean/2Inj method. This result seems to confirm findings in Backhaus (2020), where forecasts from the same project and with the same methods but reduced sample size s and reduced forecast duration d are being presented. The contrary is true, however, for tube 251H. Here the overall best results are yielded by the mean/2Inj curve, which is contrary to the findings in Backhaus (2020) where results computed with the cdf/markov method are about two to five times more accurate than those yielded with the mean/2Inj.

The complete set of all project time forecasts can be found in the appendix section C.4.

Table 6.1: MAPE of forecast of total project duration for tube 251H and 258H. The mean value is calculated as $\bar{x} = 0.5 \cdot (\delta_{\text{MAPE,251H}} + \delta_{\text{MAPE,258H}})$.

Method	$\delta_{\text{MAPE,251H}}$	$\delta_{\text{MAPE,258H}}$	$\bar{x}$
mean/2Inj	12.3	30.9	21.6
mean/markov	16.5	7.9	12.2
median/2Inj	15.2	13.4	14.3
median/markov	20.0	5.5	12.8
cdf/markov	16.7	14.8	16.1

values in percent

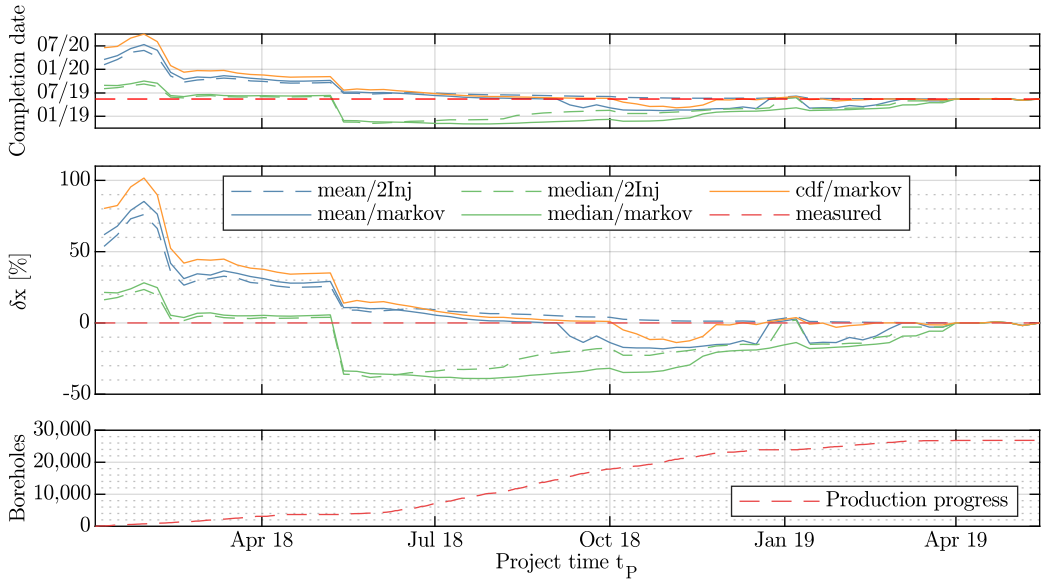


Figure 6.8: Total project time forecast with five different forecasting methods in tube 251H. Shown is the forecasted project end when forecasted at the project time t_p (top), the relative error δx of that forecast (middle) and the number of finished injections over the project period (bottom). The forecast uses the maximum possible sample size s at the time of the forecast t_p.

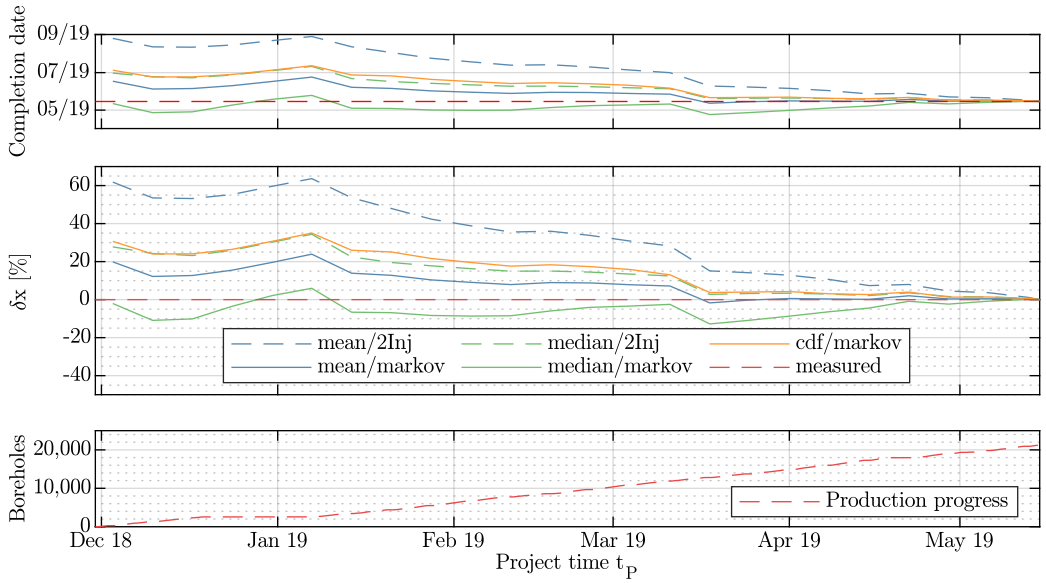


Figure 6.9: Total project time forecast with five different forecasting methods in tube 258H. Shown is the forecasted project end when forecasted at the project time t_p (top), the relative error δx of that forecast (middle) and the number of finished injections over the project period (bottom). The forecast uses the maximum possible sample size s at the time of the forecast t_p.

6.3.3 Partial Project Time Forecast

In this section, the results of project time forecasts with reduced sample size $s \in S$ and forecast duration $d \in D$ are presented, see EQ 6.33. A forecast for each of the 25 possible combinations (s, d) is calculated for each week of the project. Three AC injection units with three pumps each are being simulated. The sample always consists of the s boreholes, which have been processed last. If the number of processed boreholes at the time of the forecast is smaller than s, i. e., there are not enough finished boreholes to produce the required s samples, then no forecast is computed. The time forecast is computed for the next d boreholes. Therefore the forecast calculates the duration which the system requires to finish the next d boreholes based on the performance values collected during the processing of the last s boreholes.

$$\begin{aligned} S &= \{500, 1000, 1500, 3000, 4000\} \\ D &= \{500, 1000, 1500, 3000, 4000\} \end{aligned} \tag{6.33}$$

An example of a result plot produced with the mean/markov method for $S = \{500, 1500, 4000\}$ and $D = 1500$ is shown in Figure 6.10. Depicted is the forecasted number of processed boreholes at project time t_p (top) and the corresponding relative error δt (bottom) for the measured number of processed boreholes at project time t_p (red, dashed line). The interval between the individual forecasts is $dt = 1$ week.

An accumulation of higher relative errors is visible shortly after times of idle intervals, such as from April 2018 to July 2018 and after the Christmas holidays in January 2019.

These are artifacts that result from the way idle time is handled in the simulation. This is best explained by an example: Curve $s/f = 500/1{,}500$ (circle) forecasts two points around April 2018 with approximately 3,000 finished boreholes, i. e., $p_{\text{circle},1} = (\text{April 2018}; 3{,}000)$ and three points around $p_{\text{circle},2} = (\text{July 2018}; 2{,}000)$. Obviously, the difference of 1,000 boreholes does not mean that the project regressed and turned processed into un-processed boreholes between April and July. The forecast for $p_{\text{circle},2}$ had been made very early in the project, when 500 $(= 2{,}000 - f)$ boreholes had been processed. The performance values from this sample of 500 boreholes led the simulation to conclude that the processing time intersects with the idle time interval after April 2018. This intersection then leads to unfortunate chaining of idle time intervals, an effect described in section 6.1.1.7, and with this to a high forecasted processing duration for $f = 1{,}500$. The example further shows the prediction interval of one week, with the forecasted points being distanced approximately one week from each other during forecasting periods with relatively constant performance parameters, e. g., during the interval from July 2018 to October 2018.

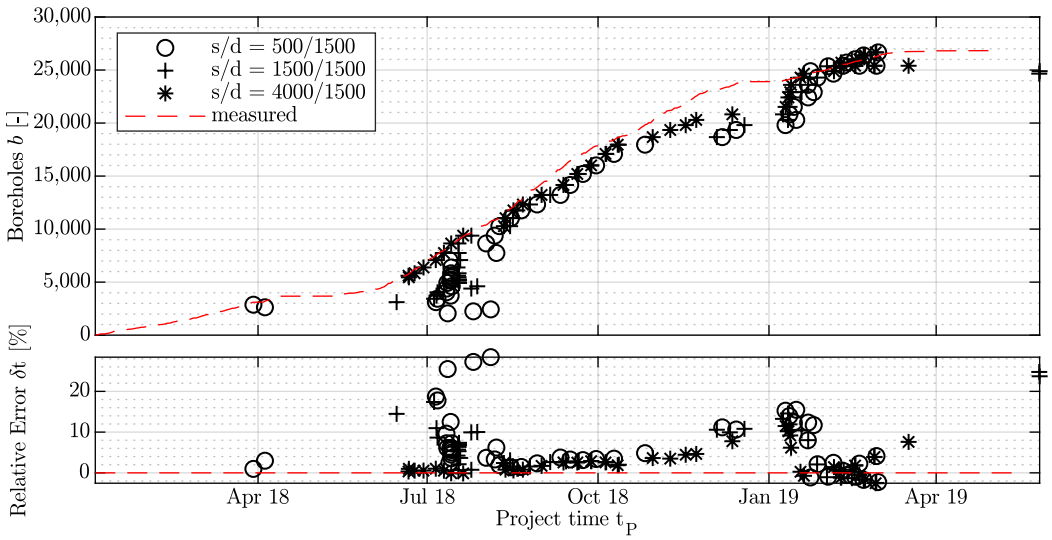


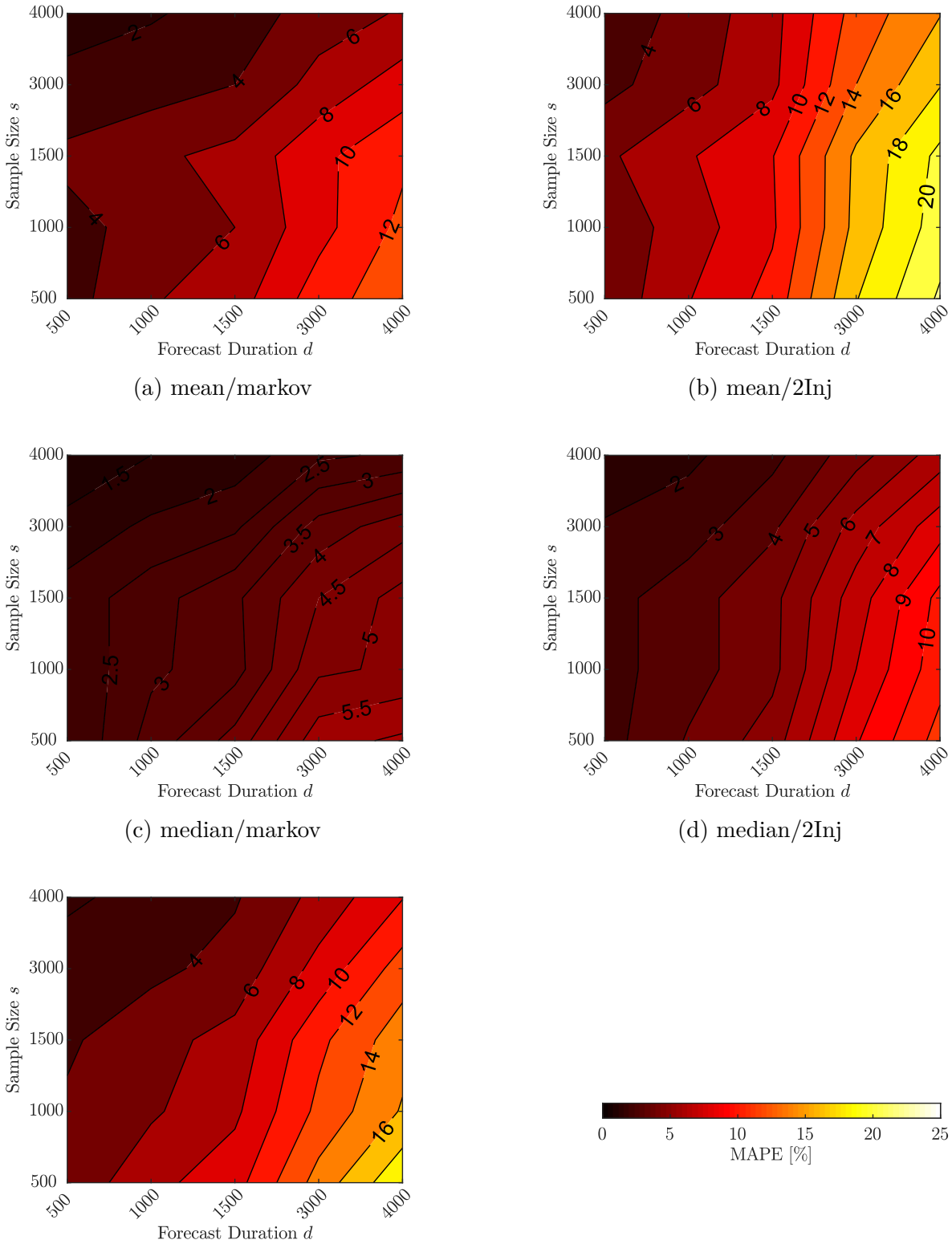
Figure 6.10: Partial project time forecast with mean/markov method in tube 251H for $S = \{500; 1{,}500; 4{,}000\}$ with $D = 1{,}500$. Shown is the forecasted number of processed boreholes at project time t_p (top) and the relative error δt between measured and forecasted project time (bottom). The red dashed line gives the measured or real processed boreholes per project time t_p. Note that the forecast is calculated for and not at t_p and that the time interval between forecasts is $dt = 1$ week.

In order to avoid the comparison of 25 curves for each of the five methods, a contour plot of each method's MAPE with all s/d-combinations is given in Figure 6.11 for tube 251H and in Figure 6.12 for tube 258H. A selection of partial project time forecasts curves can be found in the appendix section C.5. The forecast results for tube 251H are overall more accurate than those performed in 258H. The contour plots show the intuitive result that forecasts over a short duration or with large samples are more accurate than those over

long durations or with small samples. The contour lines are not equidistant, i. e., the rise of the MAPE is not linear from $\delta_{\text{MAPE}}(s, d) = \delta_{\text{MAPE}}(\text{large}, \text{small})$ to $\delta_{\text{MAPE}}(\text{small}, \text{large})$. The five methods' accuracy differs with the median/markov method yielding the most accurate results with $\delta_{\text{MAPE}} < 10\ \%$ in both tubes. The highest MAPE is computed by the mean/2Inj method and reaches $\delta_{\text{MAPE}}(4{,}000; 4{,}000) \approx 50\ \%$. In all cases, the approximation with Markov chains yields better results than the estimation with the 2Inj method.

The cdf/markov method, albeit still better than the benchmark method, yields higher MAPEs than the other forecast functions. The higher MAPE together with the need to calculate the ICDF and then draw random values from it, makes it the least effective method of the four non-benchmark methods.

If we assume that the simulated system can process around 1,000 boreholes per week, a weekly forecast with $\delta_{\text{MAPE, weekly}} < 4\ \%$ and a monthly forecast with $\delta_{\text{MAPE, monthly}} < 10\ \%$ is possible when using the median/markov method.



(a) mean/markov

(b) mean/2Inj

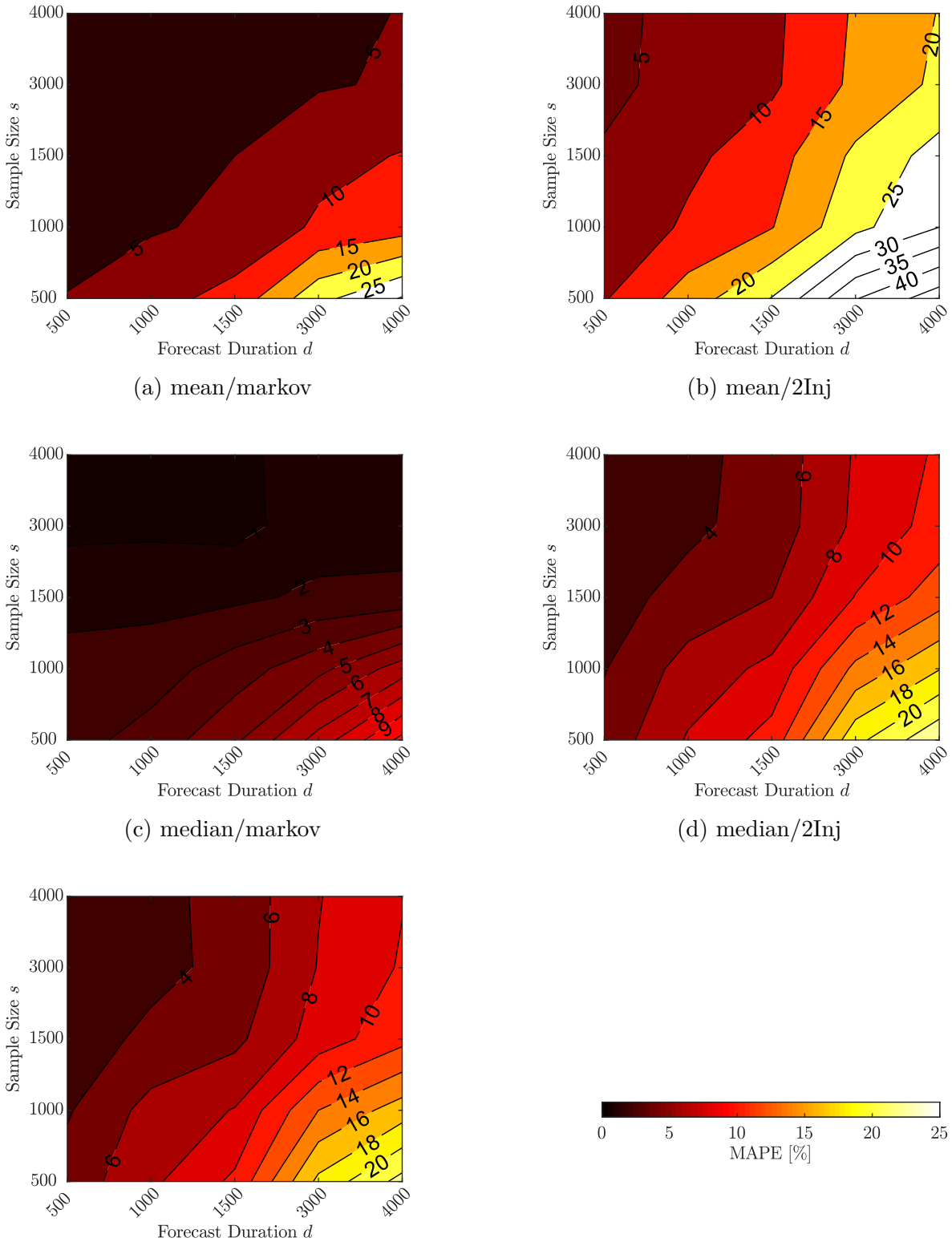
(c) median/markov

(d) median/2Inj

(e) cdf/markov

(f) Legend

Figure 6.11: Contour plot of project duration forecast in tube 251H with varying sample size s and forecast duration f. White color denotes out of scale values.



(a) mean/markov

(b) mean/2Inj

(c) median/markov

(d) median/2Inj

(e) cdf/markov

(f) Legend

Figure 6.12: Contour plot of project duration forecast in tube 258H with varying sample size s and forecast duration f. White color denotes out of scale values.

6.3.4 Summary of Results

Compared with the benchmark method (mean/2Inj), the presented alternatives' results represent an improvement for the MAPE of forecasted results. This is the case for both total and partial project time forecasts. In general, forecasts based on an approximation of the number of injections per borehole with MCs are superior to those based on the straightforward assumption that each borehole is injected two times.

While the mean/markov method is best suited for total project time forecasts, it yields a slightly higher MAPE for partial project time forecasts. It can be argued that the mean value, if known, yields the best results. However, as the real distribution of process parameters remains unknown, the sampled median provides a better approximation of the real mean value than the sampled mean. This holds for small samples. With a rising number of samples, the influence of outliers on the mean value is diminished – an effect which the median provides for small samples, such as those used for the partial project time forecast simulations.

Concerning the quality, the comparison of measured data with simulation results validates the model. Construction managers should use the median/markov method for their weekly or monthly forecasts and only switch to mean/markov if sampling from a large set of process measurements is possible. This guideline will be considered in the next chapter, which deals with the Pareto optimization of the machinery for total project time and cost.

7 Pareto Optimization Under Consideration of Collaboration of Injection Units

Any variation which is not inherited is unimportant for us. But the number and diversity of inheritable deviations of structure [...] is endless.

Darwin (1859)

Two specific goals in construction management are to minimize the construction time and the construction cost. These two goals, however, may contradict each other. With a rising number of machines on the construction site, the system's overall productivity increases, as well as cost per hour. If the time saved by using an additional machine means that the construction time is reduced to such an extent that overall costs are saved, then employing an additional unit is worthwhile. Whether or not this is the case may depend on the extent to which the production units influence each other. For example, if the output of machine B depends on the input it receives from machine A, an unbalanced increase in the number of machines A and B can lead to an accumulation of material between machines or too little material being supplied to machine B. An example is presented in Figure 7.1. The problem becomes even more complicated if production machines are added to the construction site throughout the project. While this strategy leads to shifting the origin of costs further to the end of the project and a possible payment on account, it also carries the risk that the machines are used too late, which may lead to contractually agreed deadlines exceeded and thus to penalties.

The first part of this chapter describes the *Multi Server Provduction System 2* (MSPS2), a model designed to calculate Pareto optimal solutions for different combinations and starting times of AC and PU injection units. The MSPS2 builds on the SSPS and the MSPS which were presented in chapter 6 and is implemented in the `GBOptimization` package of the GBPlan framework. The MSPS2 implements a cost model to estimate the total cost of the project based on the number of machines at work and the time which these machines require to finish the processing of all boreholes. This cost model is described in the second part of this chapter. The chapter closes with section 7.3, in which the results of the Pareto optimization with regards to the optimal number of machines and the optimal machine deployment strategy are presented and discussed.

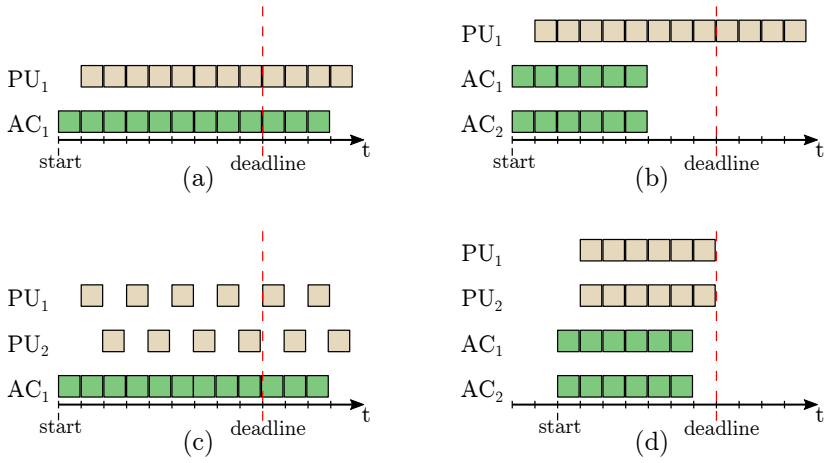


Figure 7.1: Interdependencies between production machines influence the extent to which a ramp-up of production units leads to shorter project durations. (a) When injecting PU only if all AC injections of that borehole have been finished, then working in parallel, albeit with a minimum offset, is possible. (b) If project time forecasts show that the production with that setup would exceed the deadline, then a ramp-up might be considered. If not appropriately managed, a ramp-up of machines may increase the production speed of a sub-trade. If, for instance, the AC injection capacity is doubled, PU units may not be able to handle the resulting overproduction. (c) If PU capacity is doubled, the resulting underproduction of AC injections can lead to regular idle periods. When changing from setup (a) to setup (b) or (c), the total project time does not change, and the deadline cannot be met; i. e., more machines lead to higher costs while the overall project duration remains the same. Only if the interdependencies of production units are taken into account, the overall project time decreases with an increase in machines. This is is shown in sub-figure (d).

7.1 Time-Discrete Model for AC & PU Units (MSPS2)

The MSPS2 can be understood as MSPS model with additional features, Figure 7.2. After an introduction to the general structure of the MSPS2 in section 7.1.1, those features are being described. Scenarios, section 7.1.2, are structures, which allow the easy customization of simulations. In addition to adapting the time forecast model of the MSPS to allow the integration and mutual interaction of both AC and PU units, section 7.1.3, the MSPS2 includes a cost function to approximate the project cost, section 7.1.4. Together with extensive source code optimization performed while porting the MSPS to the MSPS2, section 7.1.5, this allows for fast time-cost multi objective optimization for the project. However, because the MATLAB function `gamultiobj`, which is used for the optimization, does not accept mixed-integer problems, parts of that function had to be rewritten. The details on the changes are being described in section 7.1.6.

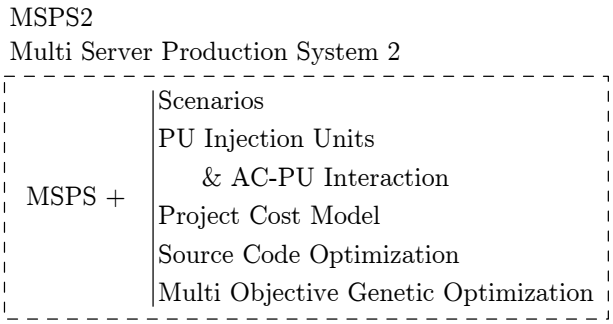


Figure 7.2: The MSPS2 is an advanced version of the MSPS. In addition to the MSPS functionality it adds features, such as customizable simulation scenarios, a project cost model, an additional type of injection unit and rules that define the interaction between injection units of different types. It furthermore implements the option to perform time-cost multi objective optimization analysis on the project. The core functions of the MSPS have been re-designed to provide lower cost of computation through source code optimization.

7.1.1 General Structure

GBPlan divides the `GBOptimization` package in two main modules: The `model` package and the `controller` package, Figure 7.3. Visualization of results is organized via static functions and are bundled in `GBPHDplots.GBOptimization`.

The `model` package provides foremost frameworks for two types of structures. These are the `project`, which can experience a variety of `scenarios`. A `project` defines the tool and machine setup and implements a function to calculate its own project cost; see section 7.1.4. A `scenario` provides the simulation with information about how the project is being managed. This information includes machine performance and tunnel parameters, such as the number of boreholes to be injected; see section 7.1.2.

The `controller` package contains functions to organize the data. In particular, it implements the MSPS2 advanced rules. With `controller.simulate` it organizes the interaction of AC and PU injection units, see section 7.1.3, and with `controller.optimization` the correct initialization and execution of the multi objective optimization is being managed.

7.1.2 Scenarios

Scenarios are created with the GBPlan package `GBOptimization.model.scenario`. They are data structures, which provide the simulation with information on what it is supposed to simulate. GBPlan offers two, customizable scenario models: `varpumps` and `varstart`.

The scenario `varpumps` stores the scenario name, the number of AC and PU injection units, the start date of the project, the number of workdays per week, statistical performance values such as the mean or median duration of an injection or the probability of a maintenance process taking place and the number of required AC and PU injections per borehole.

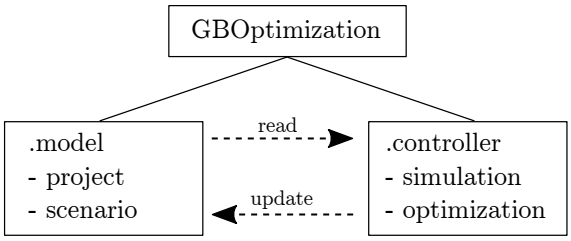


Figure 7.3: The MSPS2 is implemented in the GBPlan package `GBOptimization`, which consists of two sub-packages. `model` contains functions to manage the models, which define the data structure of the simulation. `controller` implements functions to update the model, for instance during the simulation of one project scenario or the execution of an optimization.

The latter is stored in two $1 \times n$ double arrays: B_{ac} and B_{pu}, with n being the number of boreholes in the tunnel and $b_j \in B$ the number of AC or PU injections, which must be performed on the j^{th} borehole. The statistical performance data is retrieved from a sample of real performance data. Because a scenario may define more injection units than those used on the construction site, the case may occur that the sample does not contain enough data for all the scenario units. In such a case the missing values are calculated as the median of the existing values, i. e., if the set D_s contains the durations of all injections $u_i,\ i = \{1, \ldots, n\}$ of all AC units in the sample, then the injection duration $d_j,\ j > n$ of an additional AC injection unit u_j is calculated as $d_j = \tilde{D}_s$.

In addition to the named parameters, `varstart` contains information on the earliest week after project start during which an injection pump begins to work. This information is stored in an $1 \times m$ array S, with $s_j \in S, j = \{1, \ldots, a\}$ for AC pumps and $s_j \in S, j = \{a+1, \ldots, m\}$ for PU pumps. In this array, S a zero ($s_j = 0$) indicates that the injection pump does not exist at all. The scenario allows applying a rule, which defines a minimum time interval between two injection units' starting time. The underlying idea is that, in many cases, it would be inefficient to start with all injection units at once. Not only would they be in each other's way, but exceed the training capacity of the construction site. Therefore, a stepped ramp-up makes sense.

7.1.3 AC-PU Interaction

The MSPS2 adds PU injection pumps to the MSPS. PU grout is never injected before an AC grout injection, see EQ 5.9. GBPlan models the MSPS2 as a production chain with two work cells, where the output of the first serves as the input for the second, Figure 7.4. In this model, the boreholes in each storage have a fixed order and carry the information on how often they have to be injected with AC and PU grout. The tact time δt is the time it takes a borehole to move from storage s_1 to storage s_3. If the AC units process the boreholes faster than the PU units, then boreholes accumulate at storage s_2 and waiting time δt_s is generated. Analogous to EQ 6.2 the duration of one time step Δt of the simulation

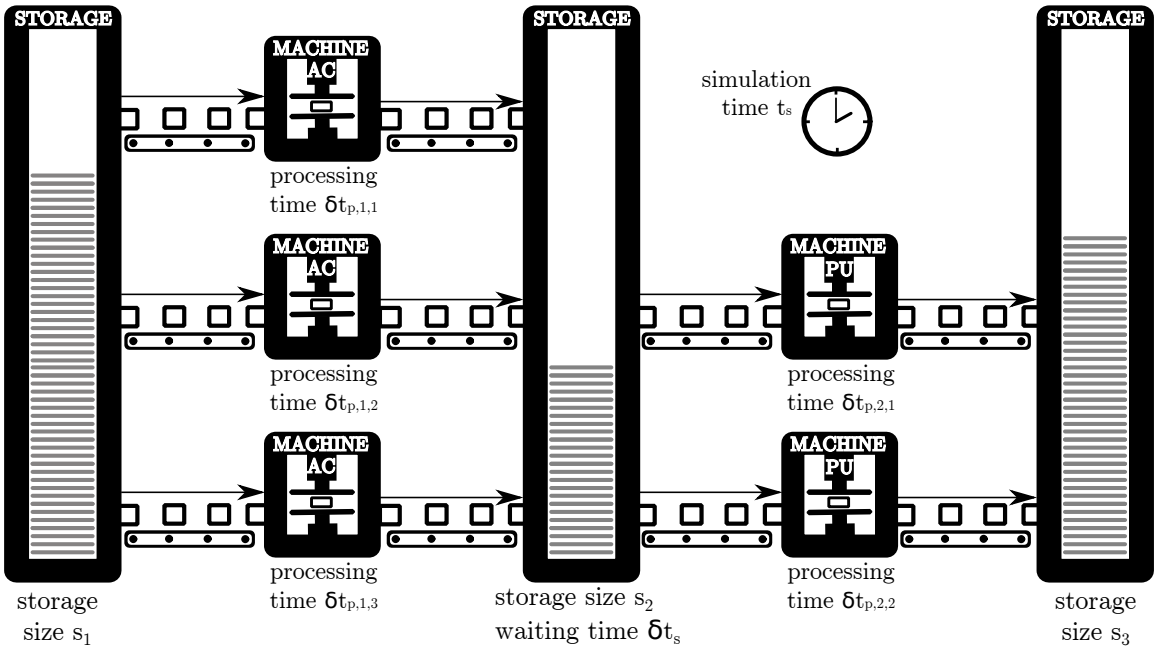


Figure 7.4: Sketch of Multi-Server Production System 2. The boreholes take the role of semi-finished parts moving along a construction line with two work cells. The first work cell performs the AC and the second the PU injection. The project is finished when all boreholes have been transferred from storage hopper 1 to storage hopper 3. The storage hopper 2 serves as a buffer if more AC injections are being carried out than the PU injection pumps can handle.

is calculated by simply adding up all the process time intervals the borehole experiences during its journey trough the production system, i. e.,

$$\begin{aligned}\Delta t &= \delta t_{idle} + \delta t_{cal,\mathrm{AC}} + \delta t_{tact,\mathrm{AC}} + \delta t_{dist,\mathrm{AC}} + \delta t_s + \delta t_{cal,\mathrm{PU}} + \delta t_{tact,\mathrm{PU}} + \delta t_{dist,\mathrm{PU}} \\ &= \delta t_{idle,\mathrm{AC}} + \delta t_{cal,\mathrm{AC}} + \delta t_{mov,\mathrm{AC}} + \delta t_{p,\mathrm{AC}} + \delta t_{u,AC} + \delta t_{maj,\mathrm{AC}} + \delta t_{min,\mathrm{AC}} \\ &\quad + \delta t_s \\ &\quad + \delta t_{idle,\mathrm{PU}} + \delta t_{cal,\mathrm{PU}} + \delta t_{mov,\mathrm{PU}} + \delta t_{p,\mathrm{PU}} + \delta t_{u,PU} + \delta t_{maj,\mathrm{PU}} + \delta t_{min,\mathrm{PU}}.\end{aligned} \tag{7.1}$$

In its implementation, the simulation uses the fact that an AC injection is never followed by an PU injection to divide the calculation into two steps. After an initialization phase, during which the algorithm loads the required data from the scenario S, the first step is to loop over the array B_{AC}, which contains the number of injections per borehole. It uses the information from B_{AC} and S to compute each borehole's injection duration, one after another. A general sketch of that procedure is given in Figure 7.5. In doing this it continuously updates the internal clock of each pump (see EQ 6.24), is oblivious to the following PU injections, and behaves just like the MSPS. It further writes the finishing date of each borehole to the array $T_{\mathrm{fin,AC}} = \{t_{\mathrm{fin,AC},1}, t_{\mathrm{fin,AC},j}, \ldots, t_{\mathrm{fin,AC},n}\}$.

During the second step the algorithm loops over all boreholes $b_{\mathrm{PU},j} \in B_{\mathrm{PU}}$. Instead of merely computing the injection duration one after another, it uses the finishing dates

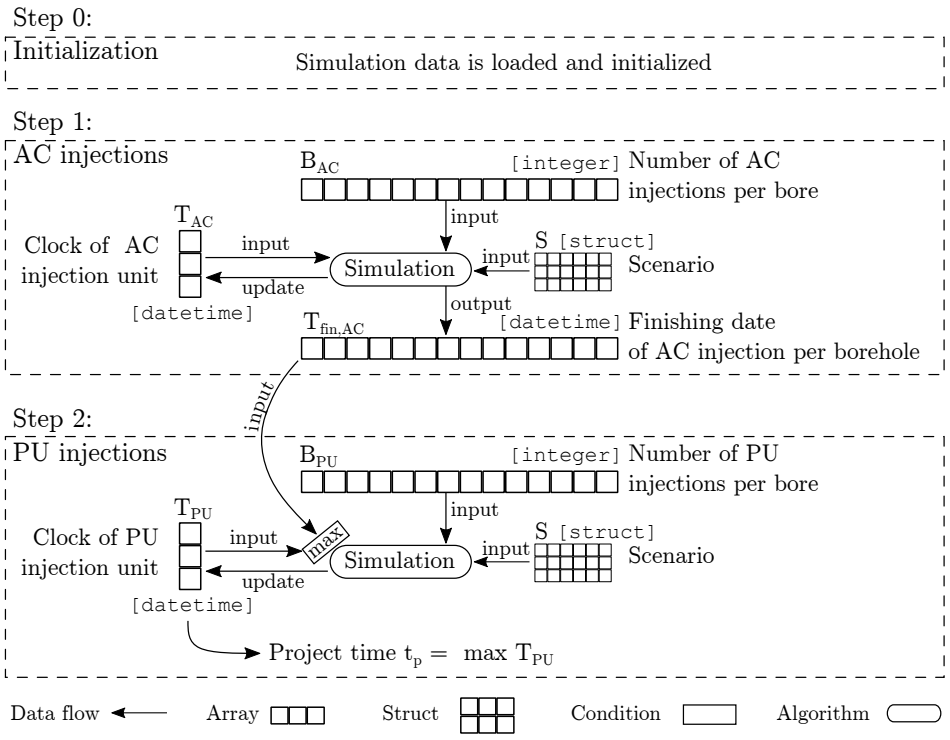


Figure 7.5: The MSPS2 computes the total project duration in a multi-step approach. After an initialization phase, the AC injections are being simulated. The algorithm stores the end of AC processing for each borehole before it moves on to simulate the PU injections. PU units are not allowed to overtake the preceding AC units. Hence, during each iteration of the simulation, the algorithm checks whether or not a conflict between the end time of AC processing and the start time of PU processing occurs. If so, the start time of the PU injection is adapted to ensure that PU units stay behind the AC units.

stored in $T_{\text{fin,AC}}$ as the earliest starting date for the injections. This changes the update function for the pumps' clock from EQ 6.24 to

$$t_{i,\text{PU}}^{(k+1)} = \max\left(t_{i,\text{PU}}^{(k)}, t_{\text{fin,AC},1}\right) + \Delta t_{1,\text{PU}}^{(k)}. \tag{7.2}$$

Off course, analogous to EQ 6.25, $T_{\text{fin,AC}}$ and B_{PU} must be updated at the end of each iteration step k for the model to stay consistent, therefore

$$\begin{aligned} T_{\text{fin,AC}}^{(k+1)} &= T_{\text{fin,AC}}^{(k)} \setminus t_{\text{fin,AC},1}, \text{ and} \\ B_{\text{PU}}^{(k+1)} &= B_{\text{PU}}^{(k)} \setminus b_{\text{PU},1}. \end{aligned} \tag{7.3}$$

Once the end condition $B_{\text{PU}}^{(k)} = \varnothing$ has been reached the project end t_{end} is retrieved from the clock of PU units T_{PU} as $t_{\text{end}} = \max t_{\text{PU},j}$.

7.1.4 Cost Function

The cost function f_c calculates the total project cost c. It is part of the GBPlan package `GBOptimization.model.project` and therefore has access to all project-related data. Note that for the MSPS2 a `project` serves as a container for a data structure, which defines the number and kind of machines (`GBOptimization.model.machine`) of a project. It must not be confused with the far more complex data structure for projects, which is used in connection with the *Multi Server Production System 3* (MSPS3) described in the next chapter.

The input parameters for the cost function f_c are the injection units M with their individual number of pumps and the duration intervals D during which the injection units are active. While M is known from the moment of the project start, D is the result of the project time calculation.

$$c = f_c(M, D) \tag{7.4}$$

The cost function implements the GBPlan cost model, which is described in detail in section 7.2.

7.1.5 Source Code Optimization

A major design choice made during the implementation of the MSPS was to produce easy to read and easy to debug source code. While this choice fostered the understanding and thus the development of the model, it also resulted in a relatively high computation cost. However, a common challenge in computational variation and optimization is to reduce the cost of computation, i. e., the reduction of the time the system needs to compute one step in the simulation. Hence, a significant re-design of the source code stemming from the MSPS has been performed. In particular, the following five changes were implemented:

1. Structuring of the model in the `controller` and the `model` package, to maintain clarity with the rising complexity of the source code.
2. Sacrificing clarity of source code in favor of computation speed by replacing complex data types, in particular the MATLAB data type `datetime`, with `double` values, i. e., 64 bit floating-point numbers.
3. Reduction of read and write access to notoriously slow `struct` variables by writing the stored values to primitive data types, such as `double` and `char`, during an initialization phase at the beginning of a function.
4. Simplification of randomized maintenance processes to avoid the time-intensive drawing of random numbers. If $0 < p < 1$ denotes the probability of a maintenance process to take place per injection, it is assumed that for a total of x injections, the process is executed $n = p \cdot x$ times. To avoid that several pumps with the same p value always experience maintenance together, i. e., that maintenance processes accumulate in regular intervals, the algorithm distributes these processes evenly throughout the project.

5. The integration of idle times into the simulated process sequence, section 6.1.1.7, is extremely computation cost-intensive. Furthermore, it is a feature which allows a posteriori analysis of the project and does not support fictional scenarios. Because the MSPS2 is designed for optimization, i. e., the comparison of many fictional scenarios, this feature has been removed.

7.1.6 Genetic Multi Objective Optimization

GBPlan uses the MATLAB function `gamultiobj` for the multi-objective optimization of the project. Because the MATLAB version, see Table C.6, used for the implementation of GBPlan does not allow the solving of mixed-integer multi-objective optimization problems in its implementation of `gamultiobj`, this feature is provided by GBPlan. `gamultiobj` accepts function handles as input parameters to overload methods, which define how the initial population is created, how mutation is being handled, and how crossover operations are being performed. In addition to providing integer optimization features, overloading these methods allows improving computation time and accuracy by integrating knowledge on the optimization problem into the otherwise generic optimization functions. The redefining functions are nested in `GBOptimization.controller.optimization` and described in detail in the following sections, i. e., the initialization of the population in section 7.1.6.1, the crossover of two genomes in section 7.1.6.2 and the mutation of a genome in section 7.1.6.3. A brief introduction to genetic optimization theory has already been given in section 4.5. Furthermore, a post-selection process is introduced in section 7.1.6.4 to shorten computation time. A validity check for the parameter set (the genome) is performed before evaluating the objective functions. If the genome fails the check, then the simulation does not start. Instead, the genome is associated with a very high project time and cost to guarantee that it will not be selected during the next optimization cycle.

7.1.6.1 Initial Population

The initial population is created in a way to guarantee that individuals with extreme genomes are included. Hence, the initial population differs depending on the simulation scenario.

For the scenario `varpumps` the function determines the maximum number of AC units $a_{\max}$ and PU units $p_{\max}$. It then includes extreme tuples (genomes) $g_i = (a, p)$, which gives the number of AC and PU units of the i^{th} individual of the starting population $g \in G$. The following tuples are included in each start population:

$$g_1 = (1, p_{\max}),\ g_2 = (a_{\max}, 1),\ g_3 = (1, 1),\ g_4 = (a_{\max}, p_{\max}). \tag{7.5}$$

The same strategy is pursued in scenario `varstart`. The number of AC and PU machines is limited to four injection units with three pumps each. This leads to an array of earliest starting dates or genomes $g = (a_1, a_2, a_3, a_4, p_1, p_2, p_3, p_4)$ with a_i being the earliest week after project start during which the AC unit is allowed to work and p_i being the earliest week after project start during which the PU unit is allowed to work. A zero indicates that

the injection unit does not exist, so that $g_{\text{example}} = (0, 0, 0, 1, 0, 0, 0, 1)$ stands for a scenario in which only one AC unit and one PU unit are active and have their earliest starting date during the first week after project start. If $s_{\max}$ is the highest, possible value for the earliest starting date, a_i and p_i can take the value

$$a_i, p_i = \left\{ b \in \mathbb{Z}_0^+ \mid 0 \leq b \leq s_{\max} \right\}. \tag{7.6}$$

With this the extreme genomes, which are included in the initial population are

$$\begin{aligned} g_1 &= (0, 0, 0, 1, 0, 0, 0, 1), \\ g_2 &= (1, 1, 1, 1, 1, 1, 1, 1), \\ g_3 &= (s_{\max}, s_{\max}, s_{\max}, s_{\max}, s_{\max}, s_{\max}, s_{\max}, s_{\max}), \\ g_4 &= (0, 0, 0, s_{\max}, 0, 0, 0, s_{\max}). \end{aligned} \tag{7.7}$$

Furthermore, some average genomes are included. These are

$$\begin{aligned} g_5 = (0, 0, 0, m, 0, 0, 0, m) \text{ and } g_6 = (m, \ldots, m) \text{ with } m = \left\lceil \frac{1}{4} \cdot s_{\max} \right\rceil, \\ g_7 = (0, 0, 0, m, 0, 0, 0, m) \text{ and } g_8 = (m, \ldots, m) \text{ with } m = \left\lceil \frac{1}{2} \cdot s_{\max} \right\rceil, \\ g_9 = (0, 0, 0, m, 0, 0, 0, m) \text{ and } g_{10} = (m, \ldots, m) \text{ with } m = \left\lceil \frac{3}{4} \cdot s_{\max} \right\rceil. \end{aligned} \tag{7.8}$$

7.1.6.2 Crossover

Roulette-Selection (section 4.10) is used to determine which of three implemented crossover functions, SC, TC or UC are being executed. The probability of a SC is 10 percent, of a TC 20 percent and of an UC 70 percent. The two genomes used for the crossover are $g_p, g_q \in G$, with G being the set of genomes or individuals of the population. For the crossover pairs (g_p, g_q), neighboring genomes are chosen, i. e.,

$$\begin{aligned} p &= k, \\ q &= \text{mod}(k, n) + 1. \end{aligned} \tag{7.9}$$

Off course, both genomes contain the same number of n genes.

For the SC the position c_i, EQ 7.10, at which the genome is cut, is selected randomly with the MATLAB function `randi`, i. e.,

$$c_i, c_j = \left\{ i, j \in \mathbb{Z}_{\neq 0}^+ \mid 1 \leq i, j \leq n \right\}. \tag{7.10}$$

With this the genomes $\dot{g_p}$ and $\dot{g_q}$ after the crossover are calculates as per

$$\begin{aligned} \dot{g_p} &= \{ g_{p,1}, \ldots, g_{p,c_1}, g_{q,c_1+1}, \ldots, g_{q,n} \}, \\ \dot{g_q} &= \{ g_{q,1}, \ldots, g_{q,c_1}, g_{p,c_1+1}, \ldots, g_{p,n} \}. \end{aligned} \tag{7.11}$$

For the TC, two positions c_i and c_j are drawn with the same function, EQ 7.10. However, the algorithm for TC demands that the condition $c_i \neq c_j$ is fulfilled. In those cases, in which this condition is not met, c_i and c_j are changed to valid values according to Figure 7.6.

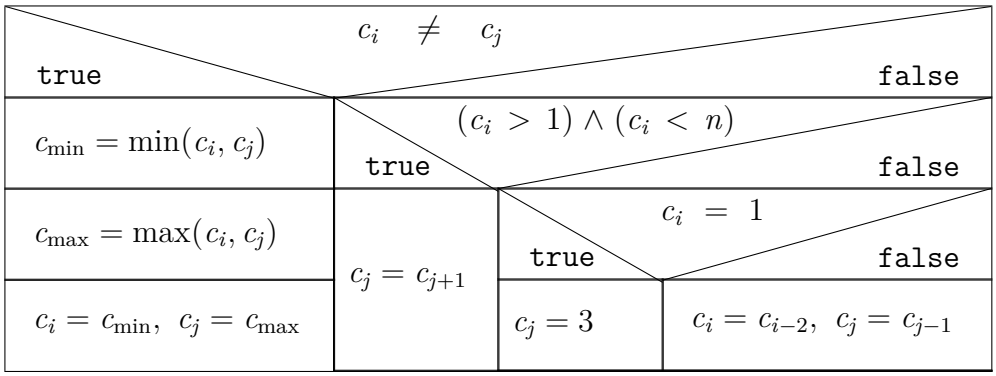


Figure 7.6: Nassi-Shneiderman chart of cut line selection for Two-Point Crossover. Two randomly drawn integers c_i, c_j, EQ 7.10, define the positions at which the genome is being cut.

With this the genomes after the crossover $\dot{g_p}$ and $\dot{g_q}$ are calculated as per

$$\begin{aligned} \dot{g_p} &= \left\{ g_{p,1}, \ldots, g_{p,c_i}, g_{q,c_i+1}, \ldots, g_{q,c_j}, g_{p,c_j+1}, \ldots, g_{p,n} \right\}, \\ \dot{g_q} &= \left\{ g_{q,1}, \ldots, g_{q,c_i}, g_{p,c_i+1}, \ldots, g_{p,c_j}, g_{q,c_j+1}, \ldots, g_{q,n} \right\}. \end{aligned} \tag{7.12}$$

For the UC an $1 \times n$ Boolean vector $b_i \in B$ with $b_i \sim \mathcal{U}\{\text{true}, \text{false}\}$ is created. The genes $g_{p,i}$ and $g_{q,i}$ are swapped, if $b_i = \text{true}$.

7.1.6.3 Mutation

In accordance with Haupt and Haupt (2004, p. 60), the mutation rate (section 4.5.2) is set to $r_m = 0.2$, i. e., 20 percent of the genes of the latest population are being randomly changed. If n_p is the population size and n_g the genes per genome, then the total number of genes which are mutated per cycle is $n_m = \lfloor r_m \cdot n_p \cdot n_g \rfloor$. The algorithm retrieves the $n_p \times n_g$ array p from `gamultiobj`. p contains the genes of all individuals of the latest population. Each row of p represents one individual or genome g_i. The algorithm then creates two $1 \times n_m$ arrays r and c, which contain the row and column indices of the genes which are to be mutated in p. Finally, the method loops over these arrays and replaces the corresponding elements in p with a random value of R, i. e.,

$$\begin{aligned} p_{a,b} = R \text{ with} \quad & a = r_i,\ b = c_i \\ \text{for} \quad & i = \{1, 2, \ldots, n_m\} \end{aligned} \tag{7.13}$$

Note that R is a random variable following the discrete uniform distribution, such that $R \sim \mathcal{U}\{b_l, b_u\}$ with b_l and b_u being the upper lower boundaries of the values being allowed for the gene.

7.1.6.4 Post-Selection

The function chain of selection-crossover-mutation, which computes the next population, might result in genomes, which are either not unique in the sense that the scenario they are representing is not unique, or they have a structure that identifies the scenario as unfit for further processing. Unfit genomes are labeled *not valid* to reduce computation time. This label tells the algorithm to return a default value instead of computing their objective functions. To distinguish the process from the selection of best fit individuals and because it takes place after that selection, it is named *Post-selection*; see Figure 7.7.

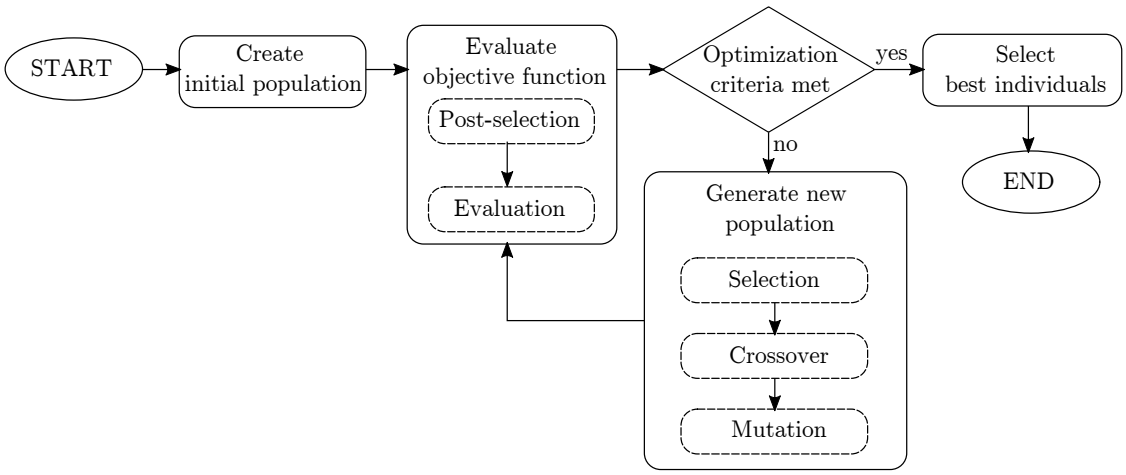


Figure 7.7: Flow chart of the genetic optimization algorithm including a post-selection step. The post-selection is carried out before genome combinations are evaluated through the computation-intensive objective functions. If the genome combination is identified as not fit to serve as offspring of the last generation, it is labeled *not valid*. The label tells the evaluation function that a simulation must not be performed. Instead, unrealistic high values for project cost and project duration are associated with that combination. Hence, the genomes labeled as *not valid* become subject to the unlabeled genomes in the coming selection phase. Hence, they will not serve as parents for the next generation. Furthermore, by skipping the evaluation with objective functions, the cost of computation is being reduced.

An example of `varstart` scenarios shall illustrate several exemplary cases in which genomes are not valid. In case 1, two genomes are not unique, even though their genomes look different. For instance, if the scenario allows a maximum of two AC and two PU units, the two genomes $g_1 = [0, 1, 0, 5]$ and $g_2 = [1, 0, 5, 0]$ represent the same setting and are therefore not unique. In both cases, one AC injection unit starts in week one, and one PU injection unit starts in week five of the project. In case 2 the genome $g_3 = [1, 1, 0, 3]$ violates the boundary condition, that a time interval of $w = 2$ weeks shall exist between the start time of two injection units. In case 3 the genomes $g_4 = [0, 0, 1, 3]$ and $g_5 = [2, 6, 0, 0]$ do not contain enough injection units to inject all boreholes of the construction site, because either AC or PU injection units are missing.

The formal rules behind the labeling process are expressed in EQ 7.14, where the result of $v_k = \{\text{true}, \text{false}\}$ denotes whether the k^{th} genome of the population has been labeled *valid* (true) or *not valid* (false).

$$v_k = v_1 \wedge v_2 \wedge v_3 \wedge v_4 = \begin{cases} \text{true} \Rightarrow \text{genome is valid} \\ \text{false} \Rightarrow \text{genome is not valid} \end{cases} \tag{7.14}$$

with

$$v_1 = 0 < \left(\sum_{i=1}^{a} g_i \right), \qquad v_2 = \bigwedge_{i=1}^{a-1} \left((g_i = 0) \vee (g_i \leq g_{i+1} + w) \right),$$
$$v_3 = 0 < \left(\sum_{i=a+1}^{a+b} g_i \right), \quad v_4 = \bigwedge_{i=a+1}^{a+b-1} \left((g_i = 0) \vee (g_i \leq g_{i+1} + w) \right),$$

and

a: Maximum number of AC units,
b: Maximum number of PU units,
g_i: Gene i of genome k,
w: Minimum time interval between start of two units.

If a genome has been identified as *not valid* the result of the project duration function f_d and project cost function f_c are set to infinity, i. e., if

$$v = \begin{cases} \text{true, then } 0 < f_c, f_d < \infty, \\ \text{false, then } f_c = f_d = \infty. \end{cases} \tag{7.15}$$

During the next selection phase, genomes with infinity as result for their objective functions loose against those with realistic values. Hence, their genes will not be carried to the next generation. This procedure is significantly less costly than solving the objective functions and thus speeds up the optimization process.

7.1.7 GBPlan Implementation

The MSPS2 is implemented in the GBPlan package `GBOptimization`, which is organized in a model and a controller package. `GBOptimization.controller` contains the functionality for the simulation itself, while `GBOptimization.model` is responsible for the creation of simulation scenarios and provides a model of the project, which is required for the cost calculation described in section 7.2.

7.2 Project Cost Model

The cost model distinguishes between non-recurring costs and time-dependent costs. While the non-recurring costs depend on the number and type of injection units, the time-dependent costs additionally require a time interval as input.

The cost model estimates the number of supporting machines, section 7.2.1, and workers, section 7.2.2, based on the number and type of injection units on the construction site. It furthermore uses these values to calculate wage cost, section 7.2.3, and the one-time cost to buy the machines, section 7.2.4. The time-dependent costs from machines' operation are divided into costs for machines that work and those that are incurred if the machines are only kept in stock, section 7.2.5.

Moreover, costs, which belong to the contract management and finance perspective, are taken into account by the cost model. These are costs incurred due to delay, such as daily costs contracted after passing a deadline, section 7.2.7. Furthermore, cost of opportunity, such as costs incurred because the project binds money that cannot be used to earn interest from an alternative investment, section 7.2.8; see Braeley and Myers (2003, pp. 17 sq.) and Weygandt et al. (2008, pp. 1137 sq.). Including said cost positions is vital for the Pareto optimization because they provide boundaries against early or late spending.

The cost estimate roughly follows the structure proposed by Leimböck et al. (2015, pp. 20 sqq.) and Thomas (2019, pp. 9 sqq.) but does not take the costs of supporting materials, section 7.2.6, such as construction timber, into account. Furthermore, equipment provided by the contractor, such as the team-, office- and washroom containers have been excluded from the model. In terms of wages, the costs of non-scalable staff positions outside the tunnel are excluded from the cost function. These positions include the project and construction manager, as well as supporting functions such as the secretariat.

7.2.1 Size of Machine Park

The total machine costs depend on the type and number of injection units m and the number of pumps installed on each injection unit p_i. It is assumed, that:

- Standard AC and PU injection units are installed in a container mounted on a truck (item 2 and 3 in Table 7.1), Figure 7.8.
- Mob. PU units do not include a container. The the whole system is mounted to a pick-up truck (item 6 in Table 7.1), Figure 7.9.
- A minimum of one lifting platform (item 1 in Table 7.1) is required to make an injection unit operational.
- A single lifting platform can support a maximum of two pumps.
- A maximum of three lifting platforms can be assigned to the work area of one injection unit.
- A maximum of five injection pumps can be installed in an injection unit.

With this the number of lifting platforms l_i for injection unit i can be calculated with EQ 7.16 and the number of total lifting platforms in the production system l with EQ 7.17; see also Figure 7.13.

$$l_i(p_i) = \left\lceil \frac{1}{2} \cdot p_i \right\rceil \tag{7.16}$$

$$l = \sum_{i=1}^{m} \left\lceil \frac{1}{2} \cdot p_i \right\rceil \tag{7.17}$$

with

p_i = number of pumps of injection unit i

m = total number of injection unit

Figure 7.8: GEA, an AC injection unit. Visible are the truck and the container mounted on the truck. One of the two chemical components are stored in blue canisters and visible in the bottom right of the picture. In the background one of the lifting platforms raises to allow a worker the access to a borehole in the upper part of the tunnel.

7.2.2 Number of Workers

The number of required workers w_{tot} on the construction site is the sum of workers operating the injection units w_{inj} plus the number of workers supporting at the maintenance site w_{supp}

$$w_{tot} = w_{inj} + w_{supp}. \tag{7.18}$$

The number of workers in each work area is determined as follows, Figure 7.13:

Figure 7.9: The injection unit “Mob. PU” is mounted directly to a pick-up truck. Foreman and equipment are directly exposed to the tunnel.

Figure 7.10: Workers of the maintenance team unload injection heads, packer and hoses for further cleaning from the transport pick-up truck.

Figure 7.11: Three pumps installed in an AC injection unit.

Figure 7.12: Two pumps installed in a PU injection unit.

Workers at injection unit w_{inj} : Each injection unit requires at least one worker able to operate the unit's electronics. This type of worker is called *team leader*. The team leader is mostly occupied with operating the injection container's electronics, documentation, and refill of the tanks with injection materials. If need be, he can support his team in placing packers in boreholes. Each additional worker at the injection unit is called *packer worker*. Packer workers place inflatable packers in the boreholes. It is assumed that a minimum of two workers – one team leader and one packer worker – must run an injection unit. Furthermore, it is assumed that the maximum number of injection nozzles one packer worker can operate is two. With this, the number of workers w_{inj} per injection unit is calculated as per

$$w_{inj,i}(p_i) = \left\lceil 1 + \frac{p}{2} \right\rceil \tag{7.19}$$

with

$w_{inj,i}$: number of workers at injection unit i,

p_i : number of pumps at injection unit i.

Workers supporting at the maintenance station w_{sup} : The injection teams are supported by workers from the maintenance station. The maintenance station is located inside the tunnel, about 200 m from the tunnel's entrance. Their task is to clean the injection equipment; a routine maintenance process, which has to be performed after the injection unit's system indicates that the injection material's denaturation has led to deviating system parameters. During maintenance, the injection nozzle is removed and transported to the maintenance station. Based on interview data with construction managers, about 1.5 to 2 workers are continuously working on cleaning the nozzles and moving them from and to the injection units in the tunnel, Figure 7.10. Within this simulation, the number of supporting workers w_{sup} is calculated based on the number of injection pumps active in the whole production system. For each pump $\frac{2}{3}$ workers are supporting in the maintenance station, EQ 7.20.

$$w_{sup}(\widehat{p}) = \left\lceil \widehat{p} \cdot \frac{2}{3} \right\rceil \tag{7.20}$$

with

w_{sup} : number of workers at maintenance station

$\widehat{p}$: total number of pumps in production system

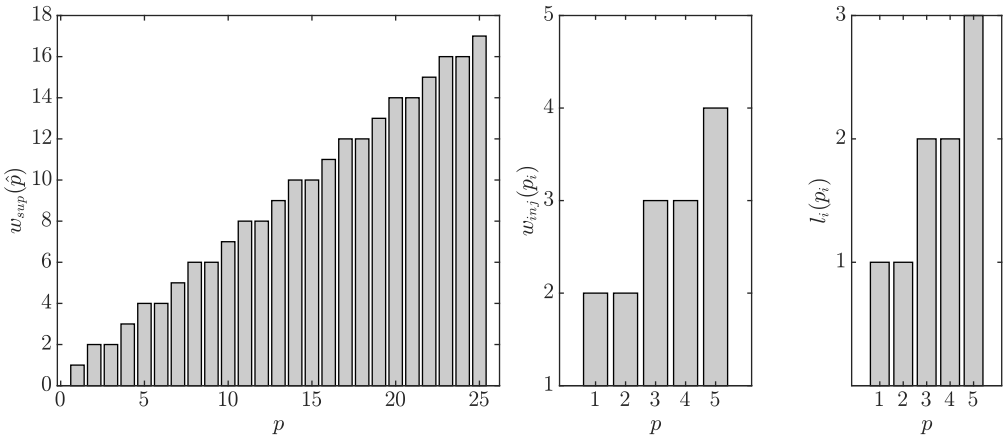


Figure 7.13: Number of resources per injection pump. $w_{sup}(\hat{p})$ is the number of workers supporting the injection units from the maintenance station and dependent on the total number of pumps $\hat{p}$ in the production system. $w_{inj}(p_i)$ is the number of workers required at each injection unit i and dependent on the number of pumps p_i at each injection unit i. $l_i(p_i)$ is the number of lifting platforms supporting injection unit i with p_i pumps.

Hence, the total number of workers w_{tot} in the production system can be calculated with

$$w_{tot} = \left\lceil \widehat{p} \cdot \frac{2}{3} \right\rceil + \sum_{i=1}^{m} \left\lceil 1 + \frac{p_i}{2} \right\rceil, \tag{7.21}$$

$$\widehat{p} = \sum_{i=1}^{m} p_i$$

with

w : number of workers in production system,

p_i : number of pumps of injection unit i,

$\widehat{p}$: total number of pumps in production system,

m : number of injection units in production system.

7.2.3 Wage Costs

Wage costs comprise wages for industrial labor in the construction industry regulated within the collective agreement of the *Bundesrahmentarifvertrag* (BRTV) and wages for foremen, which are usually not regulated by collective agreements. These costs are supplemented by performance bonuses, premium wages, above-standard payments, time bonuses for overtime, working on Sundays, public holidays and overtime, difficulty bonuses, and the employer's allowance for capital forming benefits.

These costs are bundled into a uniform medium wage for all workers and are approximately 50 €/h (gross). The construction site operates 24 h and six days per week. During standby

duty, e. g., because the client has ordered a temporary stop of the injection activities, the team is paid an average of 16 h per day. These values have been approximated based on an interview with one of the construction managers. GBPlan calculates the wage costs c_{wage} of the construction site as

$$c_{wage} = 50\ €/\text{h} \cdot \sum_{i=1}^{m} \left(\int_{t_s}^{t_e} a_i(t) + \frac{16}{24} \cdot (1 - a_i(t))\, dt \right) \tag{7.22}$$

with

t_s, t_e : start and end time of project,

$a_i(t)$: activity function of each worker i,

m : total number of workers,

$$a_i(t) = \begin{cases} 0 : \text{if worker } i \text{ is on standby at time } t, \\ 1 : \text{if worker } i \text{ is working at time } t. \end{cases}$$

7.2.4 Machine Costs

Most of the machine costs result from acquisition costs and costs for maintenance and repair, Thomas (2019, p. 510). The machine types in use and their *Original Value* (OV), a factor f_{RC} for *Repair Costs* (RCs) and *Engine Power* (EP) are summarized in Table 7.1. Note that the AC and PU injection systems are mounted on a truck (item 1 in Table 7.1), while the Mob. PU system is installed on a pick-up truck (item 6 in Table 7.1). A significant cost factor results from the injection unit's electronic equipment, Figure 7.14 and 7.15, which is expensive in both OV and RC (item 5 in Table 7.1).

Figure 7.14: Inside of GEA. A foreman explains the electronic equipment to a PhD student. A scale for quality monitoring of one of the chemical components is visible in the background.

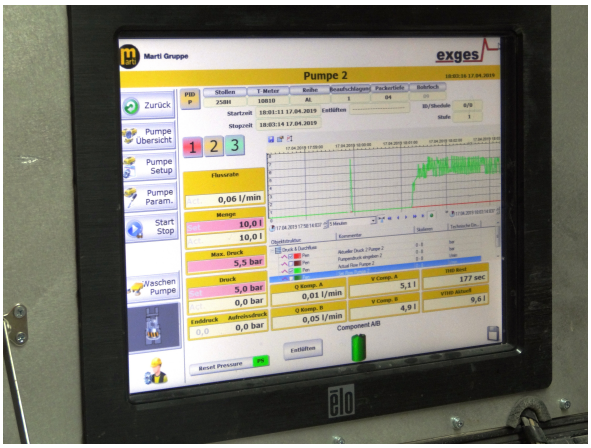


Figure 7.15: Optional monitoring screen for one of the three pumps installed in the GEA. Beside other values the flow rate (green) and pressure (red) are plottet to a diagram.

It is assumed that the machine's imputed depreciation takes place over the total duration of the project. Furthermore, the acquisition costs are assumed to be the OV and are not being adapted to the producer price index.

Table 7.1: Fixed cost parameters for machines working on the construction site. The table shows the relevant cost parameters for the machines used for the cost function of the simulation. The values have been determined through an interview with a construction manager of the construction site and by consulting external sources. In order to honor the construction site managers' wish for confidentiality, values reflect the mean of a realistic price range rather than exact values.

Item	Description	OV [€]	f_{RC} [%]	EP [kW]	Source
1	Lifting Platform Manitou 180 ATJ	114,500.00	1.4	33.0	BGL (2015, C.7.15.140) and MANITOU (2020)
2	Truck, 12 t	70,100.00	2.2	170.0	BGL (2015, P.2.00.0120)
3	Container Transport System for Truck	2,550.00	2.2	-/-	BGL (2015, P.2.00.***-AH)
4	Desoi Injection Pump	10,000.00	2.2	1.0	Interview with construction manager, f_{RC} & EP acc. BGL (2015, T.1.00)
5	Electronic equipment of injection units	120,000.00	4.0	1.5	Interview with construction manager, f_{RC} & EP best guess
6	Pick-Up Truck	43,100.00	1.9	140.0	BGL (2015, P.0.11.0110)

OV = Original Value, f_{RC} = Monthly Repair Cost Factor, EP = Engine Power

Costs for acquisition c_a is calculated as the sum of OV of each machine item, Table 7.1, used on the construction site and calculated as

$$c_a = \sum_{i=1}^{m} OV_i \tag{7.23}$$

with

m = Number of machine items.

Costs for repair are being calculated according to BGL (2015, p. 22), which assumes that 60 percent of the repair costs originate from wages. Non-wage labor cost are included via a non-wage labor cost factor f_{nwlc}. Following Stiepelmann et al. (2017) and Paulsen (2019), f_{nwlc} is approximated as 86 percent of the wage cost.

$$RC_i = \mathrm{OV}_i \cdot f_{RC,i} \cdot f_{u,i} \cdot (1 + 0.6 \cdot f_{nwlc,i}) \tag{7.24}$$

with

$f_{nwlc} = 0.86$ (Factor for non-wage labor cost)

$f_u = \{0.5, 1\}$ (Factor for intensity of utilization)

Furthermore, a utilization factor f_u is being introduced. The factor reflects the fact that most of the time the trucks stand motionless in the tunnel while the rest of the equipment is used according to their design specification. It is assumed that this has an influence on the repair costs. The unmodified OV is being used as a basis for the computation of RC.

Table 7.2: Hourly repair cost parameters for the machines on the construction site.

Item	Description	OV [€]	f_{RC}[%]	f_u [-]	RC $[\frac{e}{\text{month}}]$	RC $[\frac{e}{h}]$
1	Lifting Platform Manitou 180 ATJ	114,500.00	1,4	1	2,430.15	3.39
2	Truck, 12 t	70,100.00	2,2	0.5	1,168.99	1.63
3	Container Transport System for Truck	2,550.00	2.2	1	85.05	0.12
4	Desoi Injection Pump	10,000.00	2,2	1	333.52	0.47
5	Electronic equipment of injection units	120,000.00	4.0	1	7,276.8	10.16
6	Pick-Up Truck	43,100.00	1.9	0.5	620.73	0.87

OV = Original Value, f_{MRC} = Repair Cost Factor, f_u = Utilization Factor
RC = Repair Cost

It is assumed that RCs are only incurred if the machines are being in operation. Hence, the RC for the entire machinery can be calculated according to EQ 7.25.

7.2.5 Material and Energy Costs

Material costs include costs for building materials (such as concrete, bricks, mortar, injection grout, or precast concrete part) and supporting materials (such as nails and wire), Thomas (2019, p. 510). Leimböck et al. (2015, p. 21) supplements these with the cost of operating materials (such as petrol or electricity). In the analyzed project, operating materials are provided by the client, who does not distinguish energy spent between trades and did provide insight into financial data for this study. It is assumed that injection material volume does not differ between different machine setups because 100 boreholes will require the same amount of injection grout regardless if one or one hundred pumps are being used for the injection. Therefore the cost for building material has been excluded from the cost function.

Energy cost, however, might differ because it makes a difference if a setup uses one or two lifting platforms. In order to provide energy cost values for the simulation, these costs are being guessed. Table 7.1 summarizes the made assumptions and the sources upon which the assumption are based. The energy cost c_{energy} are calculated as per EQ 7.26.

$$c_{repair,total} = \sum_{i=1}^{n} \left(RC_i \cdot \int_{t=t_s}^{t_e} a_i(t)\,dt \right) \tag{7.25}$$

$$c_{energy,total} = \sum_{i=1}^{n} \left(c_i \cdot \int_{t=t_s}^{t_e} a_i(t)\,dt \right) \tag{7.26}$$

with

t : project time,
i : machine index,
n : max. number of machines,
t_s, t_e : start and end time of project,
c_i : cost factor of machine i,
$a_i(t)$: activity function of machine i,
and

$$a_i(t) = \begin{cases} 0 : \text{if machine } i \text{ is not active at time } t, \\ 1 : \text{if machine } i \text{ is active at time } t. \end{cases}$$

The cost factor c_i, EQ 7.27 and 7.28, is assumed to be constant over the project and based on energy unit prices c_u as per Table 7.3. The fuel consumption is calculated according to BGL (2015, p. 15) as approximately 0.84 l/kWh with additional cost for lubricants of 10 percent of fuel cost.

$$c_{i,diesel} = \mathrm{EP}_i \cdot c_u \cdot 1.1 \cdot 0.84 \tag{7.27}$$

$$c_{i,electricity} = \mathrm{EP}_i \cdot c_u \cdot 1.1 \tag{7.28}$$

Table 7.3: Cost of energy.

Type	Cost c_u	Unit	Source
Diesel	1.27	[€/l]	infoRoad (2020)
Electricity	0.28	[€/kWh]	Price comparison of 3 German energy providers by telephone call on 02.04.2020

7.2.6 Support Material

These are materials that are required to perform the work but do not become part of the construction. Costs for support material have not been documented, are assumed to be low compared to other cost factors, and as such negligible. Therefore, they are excluded from the present study.

7.2.7 Delay Costs

*Delay Cost*s (DCs) c_d are incurred as penalty payments if the real project end t_e exceeds the contractually agreed project deadline t_d, Figure 7.17. In GBPlan, c_d is implemented as a fixed factor f_d, which gives the daily penalty the contractor has to pay in the case of a project time overrun. With this, c_d is calculated as

$$c_d = \begin{cases} 0, & \text{if } t_d \geq t_e, \\ f_d \cdot (t_e - t_d), & \text{otherwise.} \end{cases} \tag{7.29}$$

7.2.8 Costs of Opportunity

Costs of Opportunity (COOP) are the costs incurred because money had been spent, for instance, for the production of one injected borehole, instead of using it to earn interest. These costs are also called alternative costs because it is money lost due to missing potential gain from an alternative investment. For the calculation of the COOP, an investment in government bonds is commonly assumed to be the alternative investment because these have a secure yield with a low chance of default. A good introduction to COOP is given by Braeley and Myers (2003, pp. 17 sq.) and Weygandt et al. (2008, pp. 1137 sq.). For the simulation period, the interest earned on German government bonds was about 0.45 percent, Figure 7.16. The effect of COOP in the simulation is, that the overall cost performance

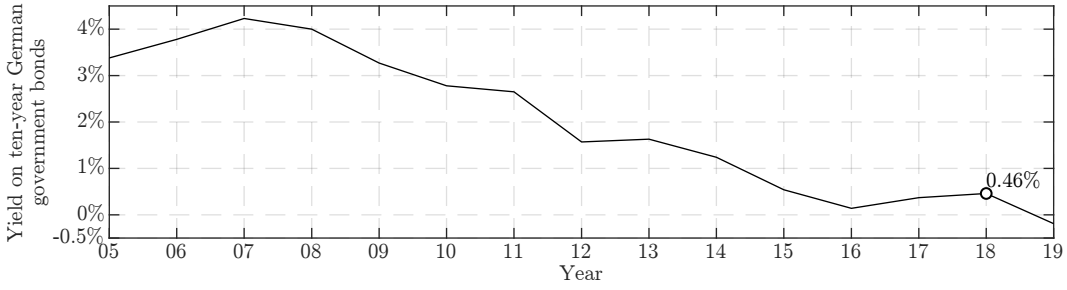


Figure 7.16: Yield in ten-year German government bonds from 2005 to 2019. The simulation results presented hereafter use a value of 0.45 percent for the calculation of the COOP. Figure adapted from Banque centrale du Luxembourg (2020).

is improved, if investments are low or if the time between spending and earning is short. With the assumption that the invoice is being paid either one month after the official project deadline t_d or, in case of project time overrun, one month after the real project

end t_e, a late project start decreases COOP; see Figure 7.17. If t_i is the time of spending of c_i, e. g., to buy machinery or pay workers, then COOP c_o is calculated as

$$c_o = \sum_{i=1}^{n} c_i \cdot f_o \cdot (max(t_d, t_e) + \delta t_p - t_i) \tag{7.30}$$

with

f_o : Interest rate of alternative investment, e. g., 0.45 percent per year,

δt_p : payment period.

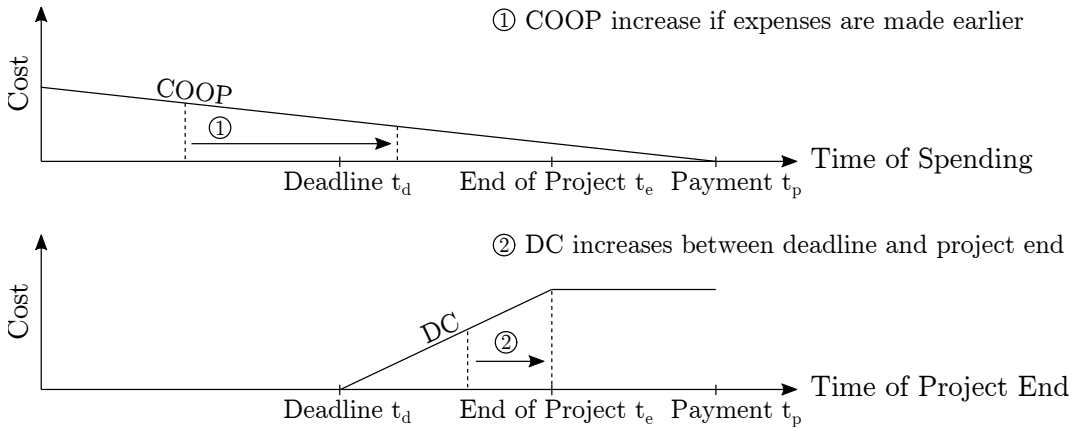


Figure 7.17: Principle sketch of COOP and DC development over time. COOP result from missing earnings from alternative investments. Early spending leads to increased opportunity costs, as alternative investments cannot be made over a more extended period. DCs are generated if the project end t_e follows the project deadline t_d, i. e., $t_e > t_d$. After the end of the project t_d, the DC stay constant until payday t_p.

7.2.9 GBPlan Implementation

The cost model has been implemented as part of GBPlan's model of construction projects in the function `GBOptimization.project.calculate_project_cost`. The package `GBOptimization.pr` is a simplified version of the more complex `GBModel.project`, which models the Tunnel Feuerbach project and is described in more detail in section 8.1.2.

7.3 Discussion of Results

In the following the results of the simulation of `varpumps` and `varstart` scenarios are being discussed. In the former scenario, the number of AC and PU injection pumps is being optimized. The latter is used to optimize the number and start dates of AC and PU injection units with three pumps each.

The simulated scenarios are built from a sample with 1,500 boreholes which have been injected between 21.09.18 and 18.02.19 in tube 251H, Table 7.4. Based on this sample, GBPlan simulates the injection of 67,497 boreholes. That is the total number of boreholes in the four primary tubes 251H/F and 258H/F.

For better comparability of simulation results, the total number of injections are kept constant, i. e., the number of AC and PU injections per borehole is drawn once. Without this measure, the results would differ between two simulations with the same input parameters since the number of injections per borehole is determined by chance during runtime.

Table 7.4: Size of sample and simulated scenario.

Item	Sample	Simulated Scenario	Unit
Number of boreholes	1,500	67,497	boreholes
Origin	251H	251H/F, 258H/F	[-]
Number of injections*	1,787	179,867**	injections
Time interval	21.09.18 - 18.02.19	01.01.18 - end	dd.mm.yy

* excl. maintenance injections, **constant for better comparability.

The actual number of injection units employed on the construction site were three AC and two PU injection units, i. e., $9 \times$ AC and $6 \times$ PU pumps. However, the digital documentation suggests that a fourth AC unit was kept in stock to step in if the productivity of the active units had been underestimated. In the following, the former combination is called the *working solution* and the latter the *backup solution.*

7.3.1 Optimization of the Number of Injection Units

A decision to be made during the Tunnel Feuerbach project's planning is the number of injection pumps to be employed. These directly influence the total number of workers and auxiliary machines, such as lifting platforms, on the constructions site. A parameter analysis is being performed to compute the number of machines on the time-cost Pareto frontier. This analysis shall provide an answer to whether or not the chosen number of injection units is also the best in terms of a time-cost optimum. The parameters are the number of AC injection pumps n_{AC} and PU injection pumps n_{PU}. These are being varied within the range $n_{\mathrm{AC}}, n_{\mathrm{PU}} = [1, 15]$. The GBPlan scenario `varpumps`, section 7.1.2, is used to manage the variables. With the optimized source code, see section 7.1.5, the MSPS2 needs roughly 1.5 minutes for the computation of the $n = 15 \cdot 15 = 225$ possible combinations.

The algorithm assumes that the maximum number of injection pumps per injection unit is three. For instance, if five injection pumps are in use, two injection units are deployed: One with three injection pumps and one with just two injection pumps. It is further assumed that all machines start to work at the same time. Hence, the features delay costs, section 7.2.7, and costs of opportunity, section 7.2.8, of the cost model are not being used.

The results of the parameter study are shown in Figure 7.18. The top picture shows all calculated solutions. The bottom picture shows an enlargement of the area in which the solutions around the Pareto optima are situated. In both diagrams, the abscissa shows the project duration in days and the ordinate shows the costs in millions of Euro. Each solution is labeled with a two-digit code. The first value represents the number of AC and the second value the number of PU pumps used for the solution. For instance, (6,3) represents a solution with six AC and three PU pumps.

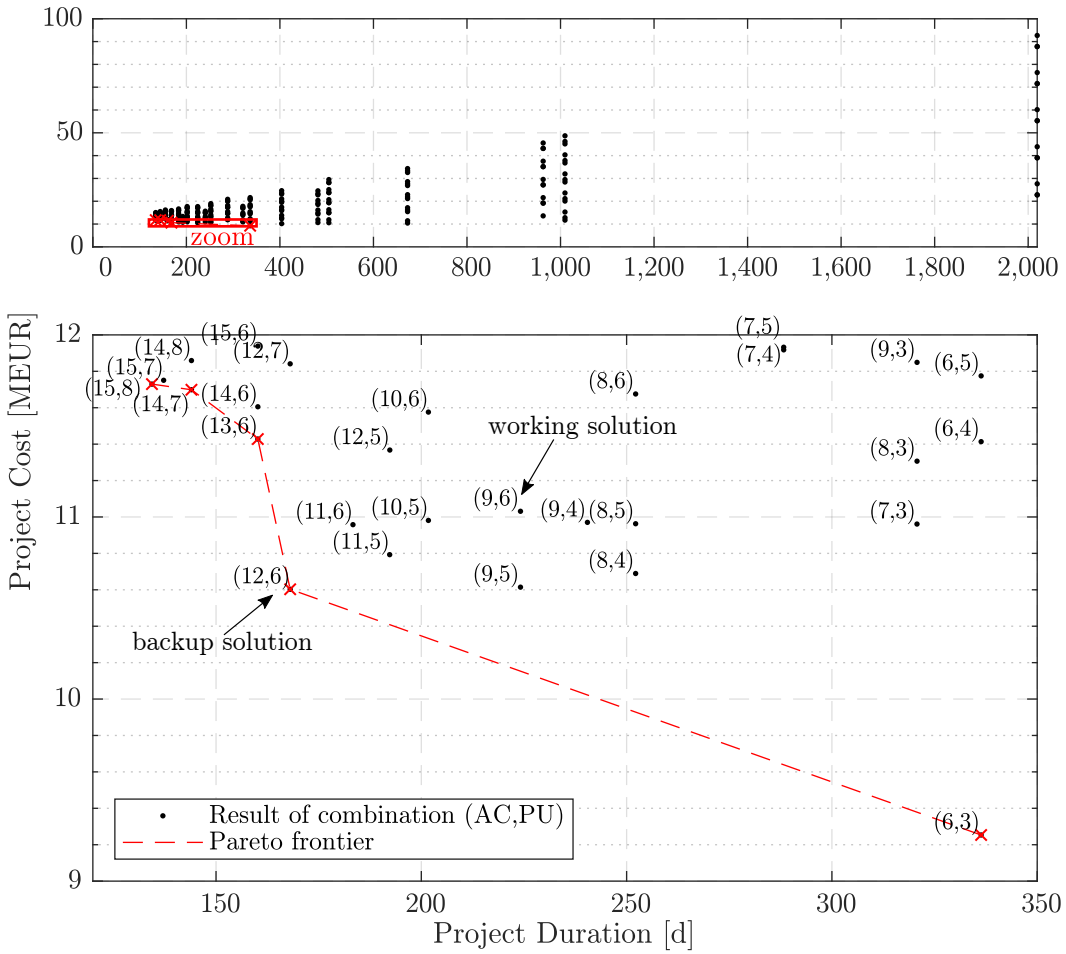


Figure 7.18: Complete set of results of variation analysis results with `varpumps` senario (top) and Pareto frontier (bottom) for different combinations $c_j = (i_{\mathrm{AC}}, i_{\mathrm{PU}})$ of injection pumps, with $i_{\mathrm{AC}}, i_{\mathrm{PU}} \in \{k \mid 1 \leq k \leq 15\}$. The working solution (9,6) chosen by the construction site is not part of the Pareto frontier. The backup solution (12,6) is part of the Pareto frontier. For both solutions, cheaper or faster solutions exist. Note that in a `varpumps` scenario all injection pumps start to work at the same time.

In the following, a selection of result points is being discussed. The combinations (6,3), (12,6), (13,6), (14,7) and (15,8) are on the Pareto frontier. It seems, that a ratio of $n_{\mathrm{AC}}/n_{\mathrm{PU}} \approx$ 1.8-2.2 yields optimum results. The best of all combinations in terms of the lowest project cost is (6,3). Straight above, i. e., with a higher cost but same project duration, are solutions with more PU pumps. Here the number of AC units determines the speed at which the system processes boreholes. This pattern is repeated for the remaining result points, above which either solutions with a fixed number of AC, or PU pumps are located.

The combination (9,6), which has been selected for the real construction site, is not part of the Pareto frontier. According to the simulation results, one AC injection pump less, i. e., (6,5), would have improved the cost of the construction site by roughly €400,000 while maintaining the same project duration. This would have meant equipping one of the two PU units with only two pumps and removing a performance buffer from the system. When comparing (9,6) with other solutions having three injection pumps per injection unit, it appears to be a compromise between (12,6) and (6,3). It exploits each injection unit to its limit, while at the same time introducing an important time buffer to the system. Compared to the working solution, the backup solution includes one additional AC unit. The corresponding combination in the simulation (12,6) lies on the Pareto frontier. Albeit cost-wise equal to (9,5), that combination would have improved the project duration by 56 calendars ($\approx$ 48 workdays) if it had been used throughout the whole project.

An interesting effect can be observed when comparing the used solution (9,6) with the backup solution (12,6). Even though more machines are in use for the backup solution, the overall cost is lower. This decrease in cost happens because the additional machines allow for faster processing of boreholes. The resulting, shorter project duration is reflected in the lower total cost of the backup solution.

The next fastest solution is (13,6), however, for a price. One more AC pump in the system would decrease the project duration by merely eight days when compared with (12,6), but increases its cost by about €800,000. The reason for the steep increase in cost is that, by changing the number of pumps from 12 to 13, an additional injection unit must be employed; i. e., together with the additional pumps, costs for the truck, a container, and costly electronics incur. The example shows how sensitive the simulation is to the cost model's assumptions, section 7.2. The fastest solution on the Pareto frontier is also the most expensive. Solution (15,8) finishes after 134 days at the cost of €11,750,000.

In summary, it can be stated that the simulated results are sound. They are comprehensible in themselves and close to the solution selected through expert assessments. On the construction site, a combination, which is a compromise between two solutions on the Pareto frontier, is being used. Without any sample data available and relying only on expert opinion for the machines' dimensioning, this choice is surprisingly close to the optimum as calculated by GBPlan. The same is true for the backup solution, which even lies on the computed Pareto frontier. One may argue that removing one injection pump from the working solution or using the backup solution (12,6) from the very beginning could have saved cost. When considering the project's strict schedule, the unknown building ground, and the foreseeable interruptions of work due to the client's unforeseeable decisions, it

would have been irresponsible to decide on a solution that eliminates any time buffer. Furthermore, the assumption that all machines start to work during the same week is only a very imprecise approximation of reality. In the following section, the results of simulations with varying start days of injection units are being discussed.

7.3.2 Optimization of Injection Unit Deployment Date

In the previous section, all possible combinations of the number of AC and PU injection pumps were analyzed to find the Pareto optimal number of pumps. The assumed boundary condition was that all injection pumps start to work simultaneously, i. e., they work from project start until all boreholes have been processed. In this section `varstart` scenario, section 7.1.2, and genetic multi-objective optimization, section 7.1.6, is used to find Pareto optimal start dates for the individual injection units.

A challenge is that the number of parameter combinations exceeds the available computing capacity. Assuming that an injection unit can either not start at all or that it can start each week between the first and the 40^{th} week of the project and that up to five injection units with three pumps each can be employed, the number of possible combinations would be $n = (10 \cdot 3)^{40+1} > 3.6 \cdot 10^{60}$. Considering a computation time of $dt \approx 0.35$, EQ 7.31, seconds per solution, see section 7.3.1, the total computation time would exceed 10^{54} years, i. e., it is impossible to calculate all solutions in the given time.

$$dt = 1.5 \left[\frac{\text{min}}{\text{combinations}}\right] \cdot 60 \left[\frac{\text{s}}{\text{min}}\right] \cdot \frac{1}{255} \left[\frac{\text{combinations}}{\text{solution}}\right] \approx 0.35 \left[\frac{\text{s}}{\text{solution}}\right] \tag{7.31}$$

Therefore, genetic multi-objective optimization, see section 4.5, is used to find a solution on a reduced number of combinations. The reduction of the set of combinations is made by allowing only three injection pumps per unit, resulting in $n = 10^{41}$ combinations, and by applying the post-selection process described in section 7.1.6.4.

With few exceptions, the default settings of `gamultiobj` are used. These are listed in appendix C.7, Table C.5. The settings have been adjusted to guarantee that the algorithm stops because the average relative change in the best fitness function between the current and last generations is below a set tolerance threshold. With these settings, GBPlan finishes the optimization in about 160 minutes when run on hardware setup (2) GBT described in Table C.6.

Three different `[varstart]` scenarios are being analyzed:

Scenario 1 assumes that an injection unit can start during any week of the project,

Scenario 2 assumes that a minimum of 2 weeks must pass between the deployment of two injection units of the same type (AC or PU) and

Scenario 3 assumes that the project's injection strategy changes after half of all boreholes have been injected.

The `varstart` scenario contains parameters for DC c_d and COOP c_o. These ensure that the project starts as late as possible, without letting delay costs render the project uneconomic.

The default values used for the results presented in this section are $c_d = 15{,}000$ €/day and an interest rate of $f_o = 0.5$ %. The project deadline is set to $\delta t_{\text{dead}} = 400$ days after project starts. The project's payday is assumed to be 30 days after the deadline or the simulated project's end, whichever is later.

GBPlan provides standardized result plots for the results of the optimization, e. g., Figure 7.19. Each plot shows the results of the last population of an optimization run. The abscissa shows the total project duration in days, and the ordinate shows the project costs in a million Euros. The Pareto frontier is plotted as a red dashed line. The parameter configuration of each of the Pareto frontier solutions is shown in the table below the result plot. Column i refers to the number above the result-point in the result plot. Column *ac/pu* shows the number of AC and PU injection units used for solution i and the column *combinations* provides the start week of each injection unit as number of weeks after project start, see section 7.1.2. For instance, the combination [0.0.0.8.8][0.0.0.0.10] means that two AC injection units started in week eight and one PU injection unit in week 10 after the project start. Consequently, no boreholes were processed, and no money was spent until week eight of the project.

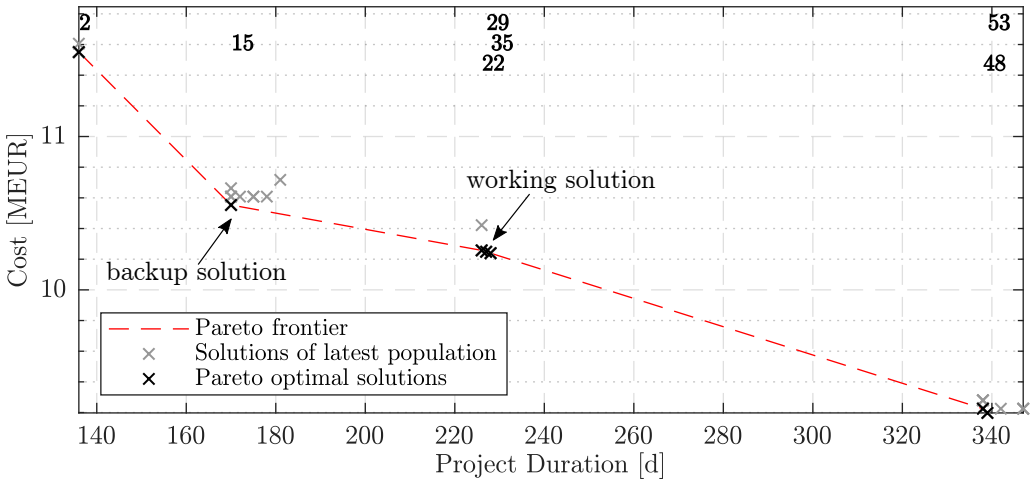
7.3.2.1 Scenario 1: New Units Can Start Each Week

The first scenario has a Pareto frontier consisting of the ac/pu combinations (5,3), (4,2), (3,2) and (2,1). The bandwidth of the ac/pu ratio is 1.5-2. Both, the working solutions $i = \{22, 29, 35\}$ with the combination (3,2) and the backup solution $i = 15$ with the combination (4,2) are part of the Pareto frontier. While the injection units of the backup solution all start to work during the same week of the project ($\approx$ 30-31), the injection units of the working solution start to work as (3,1) in week 20 and change to (3,2) around week 35-36. When comparing the working solution in Figure 7.19 with the results of Figure 7.18, the positive effect of varying start times of injection units becomes apparent. If one PU unit started 15 weeks after the other units, nearly €800,000 could be saved while keeping a constant project duration of about 230 days. Like the working solution, the fastest combination on the Pareto frontier ($i = 2$), did not appear in Figure 7.18 and experienced a significant cost improvement by starting as (5,2) and changing to (5,3) after ten weeks of the project have passed.

As expected the algorithm finds solutions which have their end date close to the project deadline, which is calculated as

$$\begin{aligned} &t_e(i) = t_s + t_d \cdot f_1 \\ &\text{with} \\ &\quad t_d : \text{Project duration in days,} \\ &\quad t_s : \text{Earliest starting week in weeks,} \\ &\quad f_1 : \text{Days per calender week.} \end{aligned} \tag{7.32}$$

The solutions found by the optimization algorithm are good, however, not perfect. This is illustrated in Figure 7.20, where the start week of all injection units for the combination



i	ac/pu	combination	i	ac/pu	combination
2	5/3	[30.30.30.30.30][0.0.30.30.40]	48	2/1	[0.0.0.8.8][0.0.0.0.9]
15	4/2	[0.31.31.31.31][0.0.0.32.32]	53	2/1	[0.0.0.8.8][0.0.0.0.10]
22	3/2	[0.0.20.20.20][0.0.0.20.35]			
29	3/2	[0.0.20.20.20][0.0.0.20.36]			
35	3/2	[0.0.20.20.21][0.0.0.21.35]			

Figure 7.19: Scenario 1: Injection units can start to work during each week of the project. Both, the working solution and backup solution are part of the Pareto frontier.

[0.0.20.20.20][0.0.0.20.35] has been shifted by δt weeks. The ordinate shows the change in cost δc compared to the cost c at $\delta t = 0$,

$$\begin{aligned} \delta c(\delta t) &= c_0 - c_1 \\ &\text{with} \\ c_0 &= c([0,0,20,20,20][0,0,0,20,35]), \\ c_1 &= c([0,0,20+\delta t_s, 20+\delta t_s, 20+\delta t_s][0,0,0,20+\delta t_s, 35+\delta t_s]), \\ \delta t_s &= \{i \in \mathbb{Z} \mid -5 \leq i \leq 5\}\,. \end{aligned} \tag{7.33}$$

Starting from $\delta t_s = 0$ the cost increase with shrinking δt_s. This behavior can be attributed to the influence of the cost of opportunity. With a fixed payday and increased time distance to that payday, the COOP increases. The same effect is also responsible for the decreasing cost from $\delta t_s = 0$ to $\delta t_s = 4$. After $\delta t_s = 4$ a sharp increase in cost is visible. The reason is the overrun of the deadline for $\delta t = 5$. If the optimization algorithm had identified the perfect solution, it should have found a solution such as $\delta t = 4$ or better. When considering the total project cost of $c_0 =$ €10,250,000, the improvement of $\delta c(4) = 1{,}000$ is only marginal.

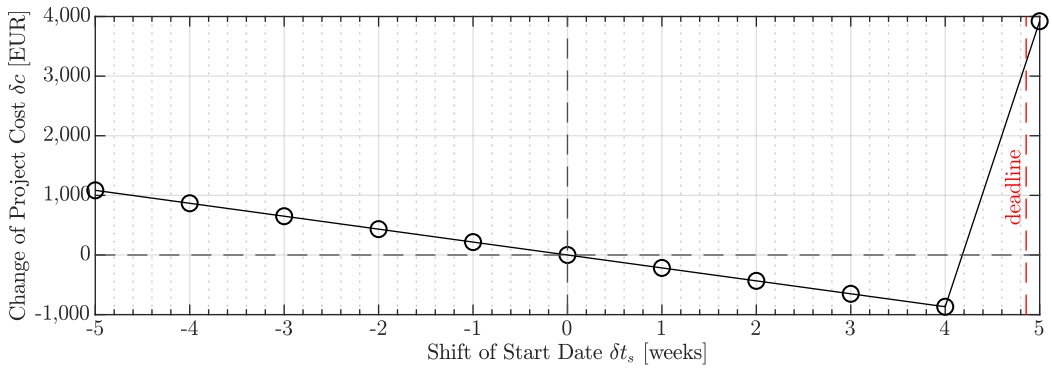


Figure 7.20: Scenario 1: Influence of shifting the start week by δt_s on the project cost c. The combination of start weeks of the injection pumps as function of the time shift is $c(\delta t_s) = [0, 0, 20 + \delta t_s, 20 + \delta t_s, 20 + \delta t_s][0, 0, 0, 20 + \delta t_s, 35 + \delta t_s]$.

7.3.2.2 Scenario 2: Two Weeks Offset Between Unit-Employment

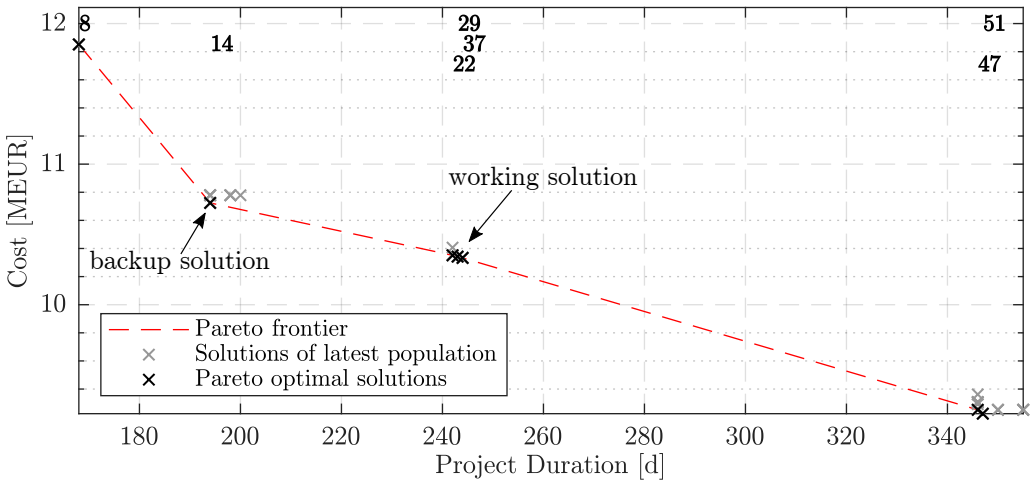
The second scenario includes an additional boundary condition. The time distance between the deployment of two AC or two PU units must be at least $\delta t_{\text{unit}} = 2$ weeks. The result plot of the optimization is shown in Figure 7.21. Compared to the solutions of scenario 1, Figure 7.19, the ac/pu combinations situated on the Pareto frontier are the same. However, both costs and project duration increase as s result of the additional boundary condition; see Table 7.5. The cost difference is highest for the ac/pu combination (5,3) with scenario 2 being €301,000 more expensive and 32 days longer than scenario 1. The smallest impact on project duration and cost is yielded by combination (2,1) with only €28,000 additional costs and a project duration prolonged by eight days. For the working solution (3,2), the costs increase by €94,000 with 16 additionally days. The backup solution (4,2) yields additional €171,000 and 24 days, respectively.

Scenario 2 assumes that the $\delta t = 2$ weeks are being used to provide training to workers and identify and solve problems, which can be expected regularly when new technology is used for the first time. Hence, on top of increased cost and project time resulting from the mere shifting of the injection units' start time, additional costs may need to be considered. These may result from slower processing times during the training phase, downtime due to maintenance, the higher workload for the supervisors who conduct the training, higher rework ratios, and additional quality checks.

Overall the simulation results are sound and meet the assumptions of experts on the construction site.

Table 7.5: Change of project duration and costs as result of two week distance between the deployment of two injection units of the same type.

	Scenario 1			Scenario 2			Difference	
ac/pu [–]	i [–]	Duration [d]	Cost [MEUR]	i [–]	Duration [d]	Cost [MEUR]	Duration [d]	Cost [TEUR]
5/3	2	136	11.551	8	168	11.852	32	301
4/2	15	170	10.553	14	194	10.724	24	171
3/2	22	226	10.257	22	242	10.351	16	94
3/2	29	227	10.248	29	243	10.342	16	94
3/2	35	228	10.239	37	244	10.333	16	94
2/1	48	338	9.225	47	346	9.252	8	27
2/1	53	339	9.197	51	347	9.225	8	28



i	ac/pu	combination	i	ac/pu	combination
8	5/3	[1.3.5.7.9][0.0.5.9.11]	47	2/1	[0.0.0.7.9][0.0.0.0.9]
14	4/2	[0.4.6.8.10][0.0.0.6.10]	51	2/1	[0.0.0.7.9][0.0.0.0.10]
22	3/2	[0.0.6.8.10][0.0.0.10.21]			
29	3/2	[0.0.6.8.10][0.0.0.10.22]			
37	3/2	[0.0.6.8.11][0.0.0.11.21]			

Figure 7.21: Scenario 2: Injection units of the same type have a minimum time distance of $\delta t = 2$ weeks. Time and cost increase compared to scenario 1. The solutions of the working solution and the backup solution are part of the Pareto frontier.

7.3.2.3 Scenario 3: Mitigation of Detrimental Effects after Changes to Injection Strategy

The third scenario analyses the hypothetical case that the client changes the injection strategy after 50 percent of all boreholes have been processed. It is assumed that starting with borehole 33,749, one additional AC injection must be applied per borehole. Boreholes, which have no planned AC injection are not affected. The new scenario 3 can be easily created in GBPlan by altering the `varstart` scenario 2 discussed in the previous section. The corresponding lines of code are shown in Figure 7.22.

```
% Define from which boreholes to which borehole the
%   changes are made.
from = ceil(length(s.injections_ac)/2);
to   = length(s.injections_ac);
% Apply changes to the varstart scenario s
s.injections_ac(from:to) = s.injections_ac(from:to) + ...
     (sc.injections_ac(from:to) ~= 0);
```

Figure 7.22: Example: Modification of a scenario. The number of the last 50 percent of AC injections per borehole is increased by 1.

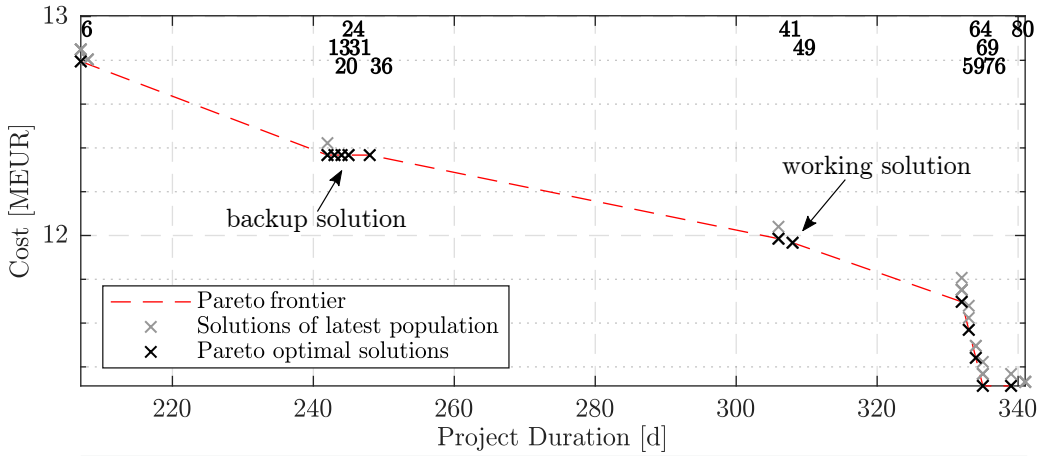
When simulated with the working solution from scenario 2, i. e., [0,0,6,8,10][0,0,0,10,21], the project duration increases to 306 days and the resulting costs are €12,874,000. While an increase in both cost and duration can be expected with rising work volume, the question arises whether a better solution exists. Figure 7.23 shows the results of a GBPlan optimization of scenario 3. For better comparability with the working solution of scenario 2 the results are listed in Table 7.6. The columns to the right show the difference of scenario solution i to the working solution of scenario 2. It can be seen that mere shifting of the start dates of a (3,2) combination is not sufficient to avoid overspending. A minimum of €889,000 of extra costs can be expected when sticking to that combination of ac/pu units. A solution available on the construction site would be the backup solution (4,2). If implemented early enough, i. e., during project week 4, that solution would not only cut production time by 64 days but also reduce overspending to €507,000. Even superior would be a solution, which demands the acquisition and manning of a fifth AC and a third PU injection unit. When implemented during the first project week, a (5,2) combination would be able to reduce overspending to €79,000. These numbers, however, are only correct for the case that the injection units can work undisturbed. In particular, it must be assured that downtime due to a slower, preceding drilling jumbo can be avoided.

With an assumed start date on 01.01.2018, the production system reaches the 33,749$^{\text{th}}$ borehole on 02.06.2018 04:18, i. e., about 152 days after project start. Suppose the decision that the injection strategy has to change is made based on local geological conditions. In that case, the information about the change likely reaches the contractor on short notice. Assuming a notice period of two weeks, that would be day 138 after the first pump started to work. Until then the working solution (3,2) with [0,0,6,8,10][0,0,0,10,21] would be in

Table 7.6: Mitigation solutions. Shown is a selection of solutions to mitigate the impact on the project duration and cost after changes to the injection scenario. The changes include one additional AC injection per borehole and are implemented after 50 percent of all boreholes have been processed. The combination $i = w$ shows the impact of the working solution's changes if no mitigation measures were performed. This solution serves as a reference to benchmark the other solutions presented in this table. The row of combination $i = w_{s2}$ shows the unaltered scenario results, i. e., if the additional AC injections had not been ordered. Solution $i = b$ represents what happened if, with a notice period of two weeks, the construction side would change from the working to backup solution. The rows with $i \in \mathbb{Z}$ show the results of an optimization that has been run with the new boundary conditions. These can also be found in Figure 7.23.

		Scenario 3		Difference*		Comment
ac/pu	i	Duration	Cost	Duration	Cost	
[−]	[−]	[d]	[MEUR]	[d]	[MEUR]	
3/2	w	306	12,874	0	0	working, scenario 3, reference
3/2	w_{s2}	226	10,257	−80	−2.617	working, scenario 2
4/2	b	272	12,981	−34	0.107	backup, scenario 3, notice period
5/2	6	207	12,795	−99	−0.079	see Figure 7.23
4/2	13	242	12,367	−64	−0.507	see Figure 7.23
4/2	20	243	12,367	−63	−0.507	see Figure 7.23
4/2	24	244	12,367	−62	−0.507	see Figure 7.23
4/2	31	245	12,367	−61	−0.507	see Figure 7.23
4/2	36	248	12,367	−58	−0.507	see Figure 7.23
3/2	41	306	11,985	0	−0.889	see Figure 7.23
3/2	49	308	11,967	2	−0.907	see Figure 7.23
3/1	59	332	11,697	26	−1.177	see Figure 7.23
3/1	64	333	11,569	27	−1.305	see Figure 7.23
3/1	69	334	11,441	28	−1.433	see Figure 7.23
3/1	76	335	11,313	29	−1.561	see Figure 7.23
3/1	80	339	11,313	33	−1.561	see Figure 7.23

*Difference to the working solution w in scenario 2.



i	ac/pu	combination	i	ac/pu	combination
6	5/2	[1.3.5.7.9][0.0.0.4.9]	49	3/2	[0.0.6.8.11][0.0.0.11.37]
13	4/2	[0.4.6.8.10][0.0.0.10.17]	59	3/1	[0.0.6.8.12][0.0.0.0.7]
20	4/2	[0.4.6.8.10][0.0.0.10.18]	64	3/1	[0.0.6.8.15][0.0.0.0.7]
24	4/2	[0.4.6.8.11][0.0.0.11.17]	69	3/1	[0.0.6.8.18][0.0.0.0.7]
31	4/2	[0.4.6.9.11][0.0.0.11.17]	76	3/1	[0.0.6.8.21][0.0.0.0.7]
36	4/2	[0.4.6.9.12][0.0.0.9.20]	80	3/1	[0.0.6.8.22][0.0.0.0.8]
41	3/2	[0.0.6.8.10][0.0.0.10.37]			

Figure 7.23: Scenario 3. Injection units of the same type have a minimum of two weeks time distance between their start days. After 50 percent of the boreholes have been processed, the injection strategy changes, and one additional injection must be performed for all remaining boreholes injected with AC grout.

place. If one would change to the backup solution (4,2) during the coming week 27, EQ 7.34, the corresponding start dates were [0,6,8,10,21][0,0,0,10,27].

$$\begin{aligned} t &= \min\left([0,0,6,8,10][0,0,0,10,21]\right) && \text{(weeks until work starts)} \\ &\quad + \lceil 138/7 \rceil && \text{(week of notice after start of work)} \\ &\quad + 1 && \text{(changes made in upcoming week)} \\ &= 27 \end{aligned} \tag{7.34}$$

GBPlan calculates project costs of €12,981,400 and a project duration of 272 days for that combination. Hence, the backup solution is suited to mitigate the increased project duration; however, not the increase in cost. Overall, the results computed by GBPlan for scenario 3 are sound. With regards to the working and the backup solution, they are close to assumptions made on the construction site and suitable for supporting decision-makers in the development of mitigation solutions during construction.

8 Pareto Optimization of Project Time and Cost in Confined Working Environment

Too many cooks spoil the broth.

Proverb, Camebridge Dictionary (2020)

This chapter presents the MSPS3, a further development of the MSPS2 discussed in the previous chapter. The model uses the SSPS, see chapter 6.1, for the simulation of the injection of a single borehole. However, instead of conveying boreholes past the injection pumps, as in Figure 6.1, the MSPS3 allows the injection units to move from borehole to borehole. This is possible because the MSPS3 features a model of the tunnel tubes. This model uses a simple graph, see section 4.4, to describe the connections between the ring segments of the tunnel. Furthermore, the *Mobile Polyurethane Injection Unit* (MobPU) is being introduced as a third type of injection unit. Instead of being installed in a bulky container, the MobPU's injection pumps are directly mounted on a pick-up truck; see Figure 7.9. These trucks are agile enough to maneuver past the other injection units. Because passing other units in the narrow tunnel environment is not always possible for some injection units, the time required for additional shunting prolongs the duration required to process all boreholes of the tunnel. This effect may limit the effectiveness of additional injection units. Hence, using MobPU units may improve productivity.

The next section, section 8.1, provides a detailed description of the MSPS3, its sub-models, and how these are related to each other. Section 8.2 goes beyond that description and details how the sub-models interact with each other, for instance, to perform a path search, identify obstructing objects or minimize computation time. The last section discusses the results of a variation analysis conducted with the MSPS3, section 8.3.

8.1 Time- and Location Discrete Model (MSPS3)

The MSPS3 adds a location component to the MSPS2. Instead of waiting to be fed with boreholes to be processed, the injection units move through the tunnel in search of the next borehole to be injected. The location component adds significant complexity to the model, which is why a graphical output – the GBView – has been implemented. The View components work in compound of a *Model View Controller* (MVC), which is

described in detail in the following section. Sections on the individual sub-models follow. These are implemented on the basis of MATLAB data structures (`struct`) and model the project, section 8.1.2; the tunnel based on tunnel segments, section 8.1.3,); machines, i. e., individual injection units, their position, and pump configuration, section 8.1.4; and time, section 8.1.5.

8.1.1 General Design Pattern: The Model-View-Controller

GBPlan has been implemented with the intent to be further developed in the future. This goal requires a source code which can be easily understood, altered and maintained. As a solution for this problem GBPlan makes use of standard design patters. In the case of the simulation with MSPS3 a simplified version of the MVC design pattern (Eilebrecht and Gernot, 2013, pp. 92 sqq.; Goll, 2014, pp. 377 sqq.) is being used, Figure 8.1.

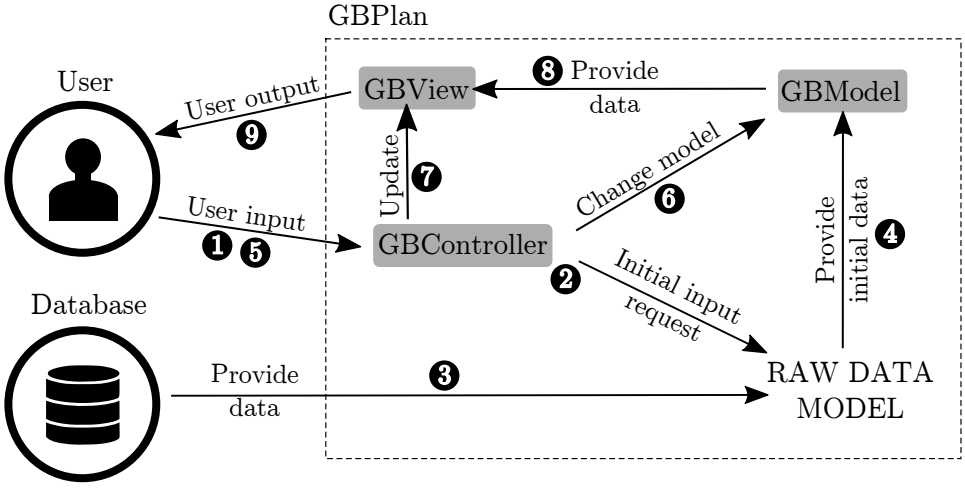


Figure 8.1: Implementation of the Model-View-Controller design pattern. (1) In order to build the model the user issues a request to the controller. (2) The controller creates an empty raw data model and (3) calls the model's initialization functions. These access the database and process the data contained in that database to fit the raw models data structure. (4) The raw data model serves as input for the simulation model. (5) Once the simulation model has been initialized the user orders the controller to start the simulation. (6) By calling the update functions of the model the controller changes its state to the next calculation step. (7) If a visualization of the results during the simulation is desired, the controller calls the view's update function, which (8) uses the current model's data to draw a figure of the current state of the simulation. (9) This visualization serves as output for the user.

The MVC distributes responsibility of handling and processing data to three different roles:

- The *model* includes the core functionality of the program and encapsulates the data and its processing. In GBPlan the model represents the state of the project at a given time. The project model contains a representation of the tunnel, the machines

in and outside of the tunnel, and a clock to keep track of the project time. A detailed description of the project model's structure is given in section 8.1.2. During the initialization phase a temporary intermediate model, the *raw data model*, manages the raw data input from the database, i. e., it works as raw data interface. It ensures that the model receives data which has been cleansed from errors and has the desired data format.

- The *view* manages the visualization of the data stored in the model. In GBPlan the *model* is handed to the *view* by the *controller*. Figure 8.2 provides an example of the view's output which shows a simplified sketch of the tunnel. Features of the view include realistic tunnel curvature in the analyzed area, labeling of the tunnel tubes, the time of the current time step, the percentage of finished AC and PU boreholes, tunnel coordinates as used in the original blueprints of the tunnel and color coding of the current status of each ring segment, and the current position of each machine in the tunnel. Segments without boreholes are transparent. The segment colors distinguish between
  - *black* segments in which no borehole has been processed,
  - *red* segments in which the work has started, e. g., all AC injections have been finished but some PU injection still remain to be processed, and
  - *green* segments in which all boreholes have been fully processed.

  The current position of each injection unit is plotted every δi_t tick (= computation steps), with a default of $\delta i_t = 100$. GBPlan offers the option to create animations of the simulated project development in the *Graphics Interchange Format* (GIF). The animations shows all computed view outputs after another, thus creating a stop motion animation.
- The *controller* is responsible for changing the model and updating the view. This is done by invoking the function `GBController.model.next()`, which computes the model's next state increment. The function `GBController.run_one_simulation()` is designed as entry point for new users and starts the simulation of one scenario for the Tunnel Feuerbach project.

On the codes side, the MVC is distributed among three packages: `GBView`, `GBController` and `GBModel`. As entry point for new users `GBController.run_one_simulation` is recommended.

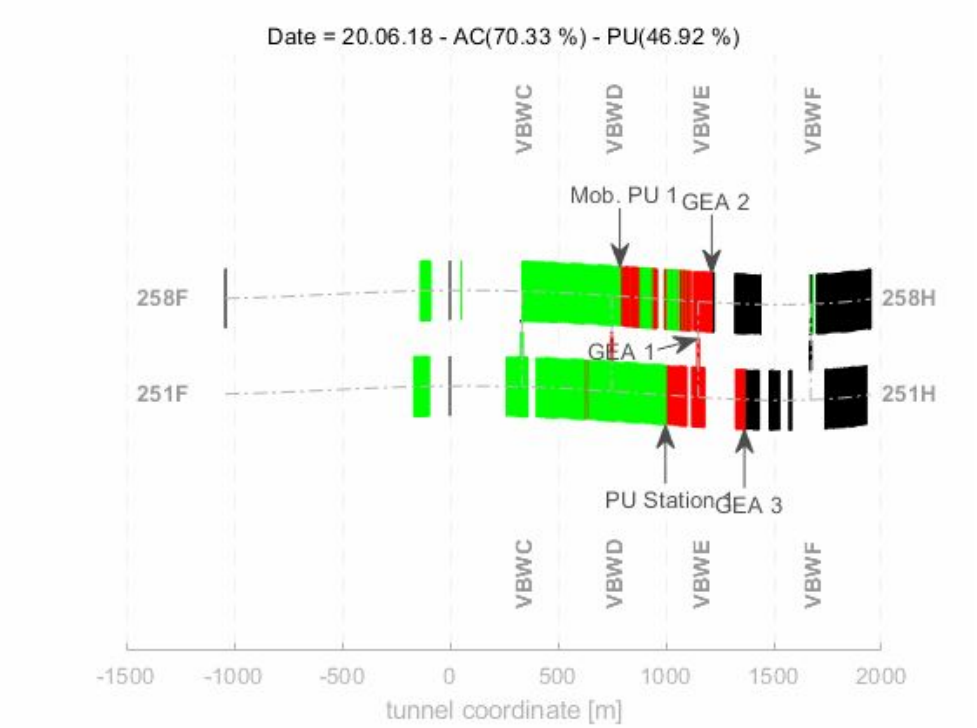


Figure 8.2: Visualization of one time step. Green represents segments in which all AC and PU injections are finished. Red segments contain both processed and unprocessed boreholes, and black segments contain only unprocessed boreholes. The current position of each injection unit is indicated with arrows. The header shows the time of the current time step and the percentage of processed AC and PU boreholes.

8.1.2 Model of Project

The project model is depicted in Figure 8.3. It serves as container for the simulation and encapsulates the simulation parameters. These are:

`project.`

`name` The name of the project.

`status` The current status of the project. The status can have the following values: -1 = *not started*, 0 = *in work* or 1 = *finished*.

`tunnel` A model of the simulation relevant physical properties of the tunnel. The tunnel model is described in detail in section 8.1.3.

`machine` A model of the injection units in the project, i. e., the AC, PU and MobPU injection units and their pumps. The model for machines is described in detail in section 8.1.4.

`clock` A model of a clock which keeps the time for each of the pumps. The time model is described in detail in section 8.1.5.

`kpi` A model of pre-measured KPIs. These are being used as input for the performance calculation of the machines. A detailed description of the model is given in section 8.1.6.

`idle` Idle times are common during projects. If they are known prior to project start they can be integrated into the model of the project with this parameter.

8.1.3 Model of Tunnel

At its core the tunnel is modeled as simple, weighted, directed graph, see section 4.4, which is made of segments (nodes) and connections (edges). Figure 8.4 shows that graph model. Its data structure in GBPlan consists of the following elements:

`tunnel.`

`name` A string value which serves as unique identifier for the tunnel.

`segment` An array containing the data of all tunnel segments. A `segment` is a data structure on its own and described in section 8.1.3.1. Among others, segments contain the information on how they are connected to other segments. This information is stored in a separate data structure called `connection` and is described in detail in section 8.1.3.2.

`tube` An index which creates a relation between a tube's name (e. g., 251H) and an integer reference number stored in `segment.tube`. Albeit harder to understand for the human user, referencing with integer values instead of string values lowers the cost of computation.

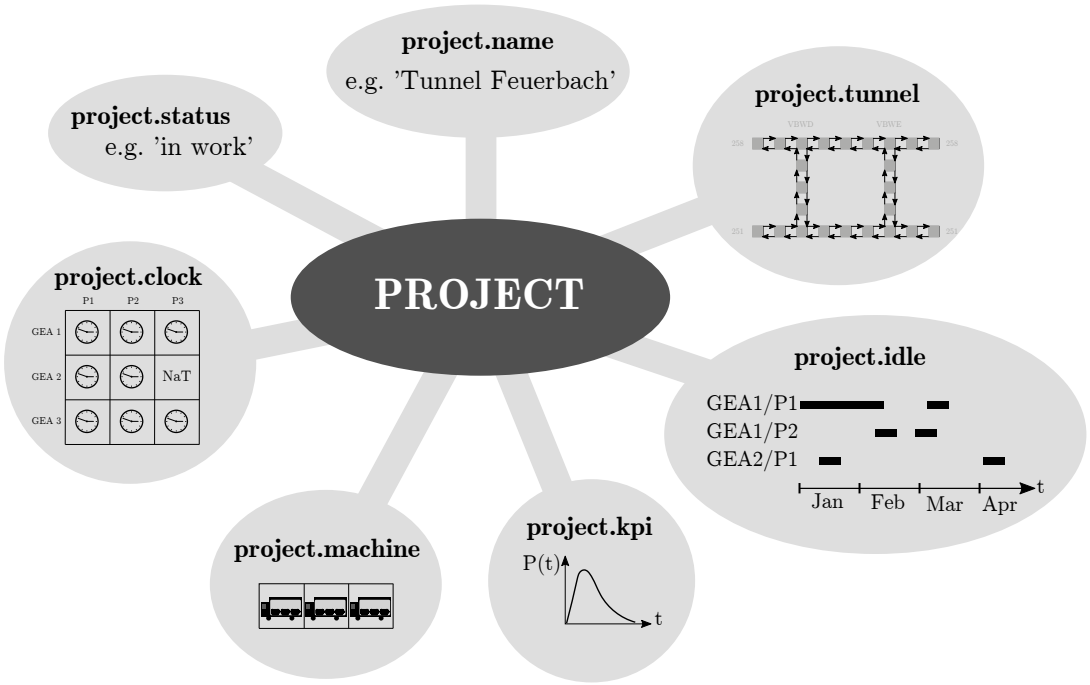


Figure 8.3: Model of project. The data structure of a project bundles the simulation's data. This is, in particular, a graph-model of the tunnel, the work equipment in the form of injection machines and their corresponding KPIs, a clock to keep track of the project time and an idle list with pre-defined downtime of the injection pumps. Furthermore, the project contains information on its current status and a unique identifier to distinguish it from other projects.

8.1.3.1 Segments and Nodes

The segments of the tunnel are organized in the array `tunnel.segment(idx)` and contain information on the local tunnel's state of processing. Furthermore, each segment serves as a node in the graph which represents the tunnel. A segment consists of the following parameters:

`tunnel.segment.`

`tube` The tube in which a segment is located coded as integer value; see description of the parameter `tunnel.tube` above.

`position` The position of the segment in the tunnel tube. The position is measured in tunnel meter as used in the blueprints of the project.

`total_AC_bores` The number of AC injections which have to be performed in the segment.

`total_PU_bores` The number of PU injections which have to be performed in the segment.

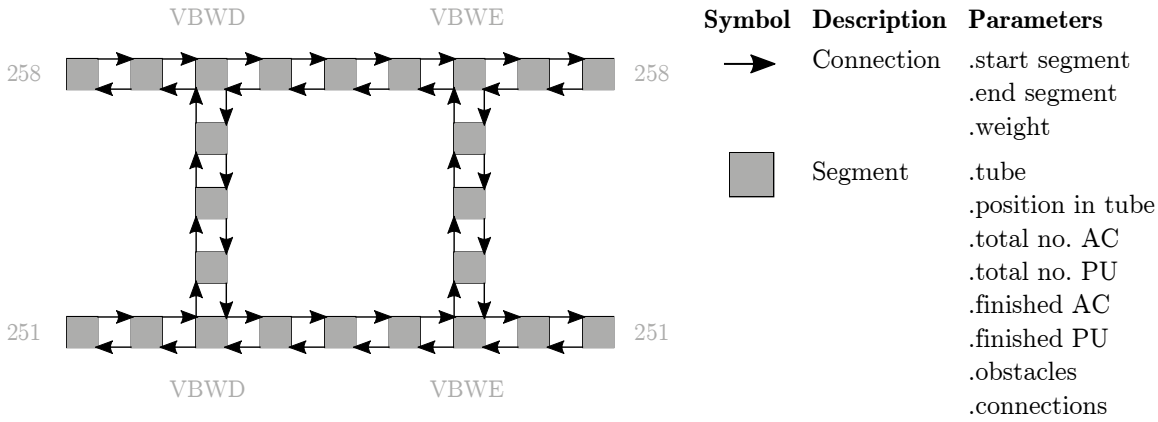


Figure 8.4: Graph model of the tunnel. Depicted is a sketch of a section of the tunnel's graph model. The ring segments of the tunnel are the graph's nodes (grey squares). The neighborhood relations between the segments are modeled with connections (arrows). Each connection contains a weight that represents the distance between its start and end segment.

`fin_AC_bores` The number of AC injections which have already been finished in the segment.

`fin_PU_bores` The number of PU injections which have already been finished in the segment.

`occupied` A Boolean value which indicates whether or not the segment is blocked by an obstacle such as a machine.

`connection` An array with outgoing connections to other segments. The `connection` indicates to which segment a machine can move without crossing another segment of the tunnel and how long the distance between these segments is. The model of a `connection` is described in section 8.1.3.2.

8.1.3.2 Connections, Edges and Positions in the Tunnel

The data structure `connection` creates an unidirectional, weighted relation between two segments. As such, it resembles an edge in a simple weighted directed graph. A `connection` is modeled as a 1×3 integer array. The first value *idx* points to the segment `project.tunnel.segment(idx)` from which the edge starts, the second points to the segment where the edge ends, and the third value contains the distance which a machine has to travel from the start to the end segment without crossing any other segment. This is best explained by an example which is given in Figure 8.5.

Modifying connections in tunnels with many segments can be cumbersome. GBPlan provides two functions to support users. These are part of the `GBModel.tunnel` package:

- `.add_connection()` adds a connection from a segment to another segment and
- `.modify_all_connections()` implements two functionalites:
  1. to flip the direction of all connections $c_{s,i} = [a, b, c]$ in a tunnel, such that $c_{s,i} = [b, a, c]$, with $s \in S$ being the index of the segment and i being the index of the connection of segment s.
  2. to make all connections undirected. This is done by adding a connection $c_{s,n+1} = [b, a, c]$ to the connection array of segment s, if that array contains a connection $c_{s,i} = [a, b, c]$ but not a connection $c_{s,i} = [b, a, c]$.

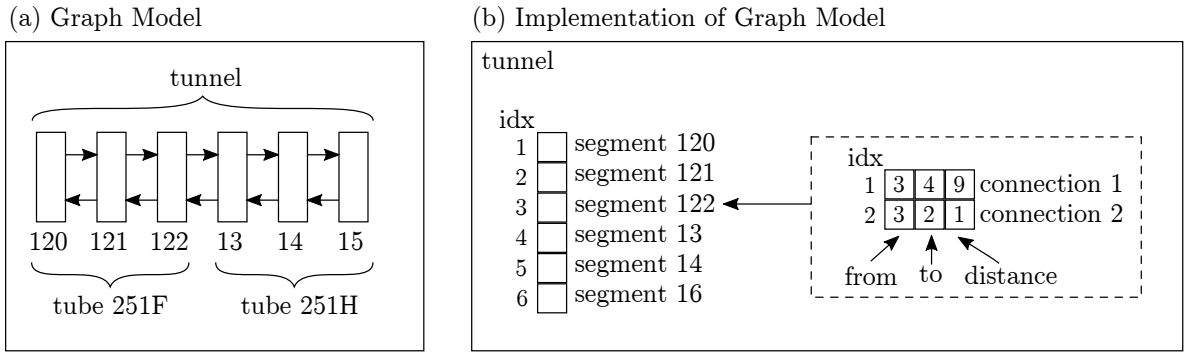


Figure 8.5: Example of a segment's connection array. A principle sketch of the graph is shown in sub-figure (a). Sub-figure (b) shows how that graph is modeled in GBPlan. The `tunnel` data structure contains an array with `segments`. Each segment has an array with connections. In the portrayed example, the `connection` array of segment 122 is shown. It defines the connections from segment 122 to segment 13 and from segment 122 to segment 121. Because segment 122 has the index $idx = 3$ and the segment 13 the index $idx = 4$ in the `segment` array of the tunnel, the connection between these two segments is a vector $c = [3, 4, d]$. d is the distance between the two segments. In the given example the distance is $d = 9$, i. e., 9 meter. Most segments have a distance of $d = 1$. A value $d > 1$ is common between transitions from one tunnel tube to another, as it is the case when moving from segment 122 to segment 13.

8.1.3.3 Tunnel Specific Functions

Apart from functions to support in the initial building process of the tunnel model, such as adding segments and connections between segments, the `GBModel.tunnel` package provides functions to modify the tunnel during the runtime of the simulation and to receive simulation relevant information from the tunnel model. Of the latter, two functions are noteworthy. For one thing the function

- `.get_closest_unfinished_segment()` and for another thing
- `.set_segments_around_machines_2_occupied()`.

Combined, the two functions calculate the next free, unfinished segment starting from a given segment. How this works in detail is described in section 8.2.2 and 8.2.3.

8.1.4 Model of Machine

A machine represents an AC, PU, or MobPU injection unit, Figure 8.6. The model is implemented in `GBModel.machine` and described by its parameters:

`machine.`

`name` A unique string value which identifies the machine. An example from the project Tunnel Feuerbach is "GEA 1" for the first AC injection unit.

`type` A Boolean which indicates whether the unit injects AC (`type` = true) or PU (`type` = false) material.

`pos` The segment at which the machine is located. `machine.pos` is an integer value referencing to the segment's array element index *idx* in `tunnel.segment(`*idx*`)`; see Figure 8.5.

`obstacle` A Boolean value which indicates whether or not the machine can be obstructed by obstacles. Obstacles can be other machines. In the simulation most machines are to big to pass each other in the narrow tube of the tunnel. An exception is the injection unit "Mob. PU 1", a PU unit which is mounted to a pickup truck; see Figure 7.9.

`hose_length` A double value containing the length of the hose the injection unit is eqipped with. The hose length determines the maximum size of the working area. That is the area the injection unit can reach without moving.

`pump.` An array which contains data structures to model the pumps the machine is equipped with. These contain two parameters:

`name` A string value which serves as unique identifier of the pump. The value corresponds to the performance data stored in `project.kpi`.

`lastAuslitern` A value in the MATLAB datetime format which logs the last time the pump underwent a maintenance process of the type "AL: Auslitern" (= calibration).

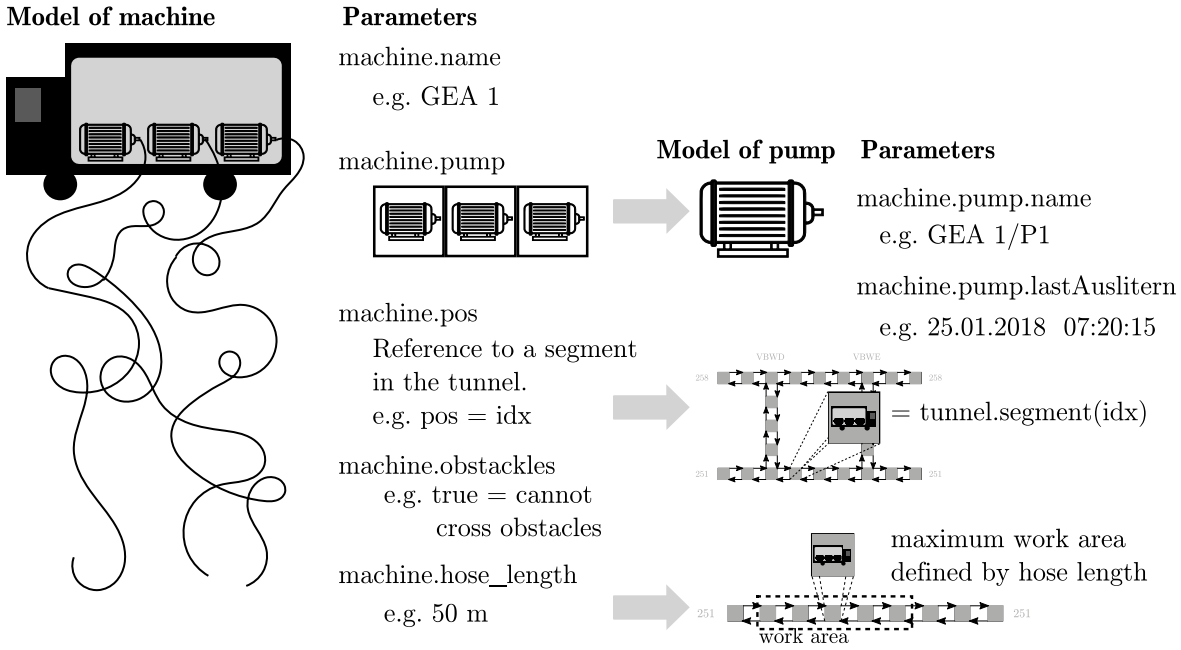


Figure 8.6: Model of an injection unit (machine). The data structure of an injection unit consists of a unique name, its position in the tunnel data structure's segment array, a Boolean value (obstacles) to indicate whether or not the machine is small enough to pass obstacles in the tunnel, the hose length which defines the maximum work area, and an array with data structures for pumps. The latter consist of two parameters. First a unique identifier (name) and second a datetime value to track maintenance processes.

8.1.5 Model of Time

The data structure `clock` keeps track of the last time a pump has finished a process, Figure 8.7. It does this for all pumps of the simulation. The `clock` is modeled as a $m \times n$ datetime array C with m being the number of machines and n being the maximum number of pumps per machine. The element $c_{m,n}$ therefore contains the end time of the last process which pump n of machine m has finished. Because machines can have a different number of pumps, some elements of C are empty. This is indicated by *Not a Time* (NaT), MATLAB's *Not a Number* (NaN) equivalent for datetime values.

8.1.6 Model of KPIs

The data structure `project.kpi` contains statistical performance data for each of the pumps in the system. The performance values are being stored in $m \times n$ arrays with the index idx. The reference between a pump and the index $1 \leq idx \leq n$ is created with the array `project.kpi.pump`, which contains the pump's unique identifier (name) as string value. With this, n is equal to the number of pumps in the simulation. The value m denotes

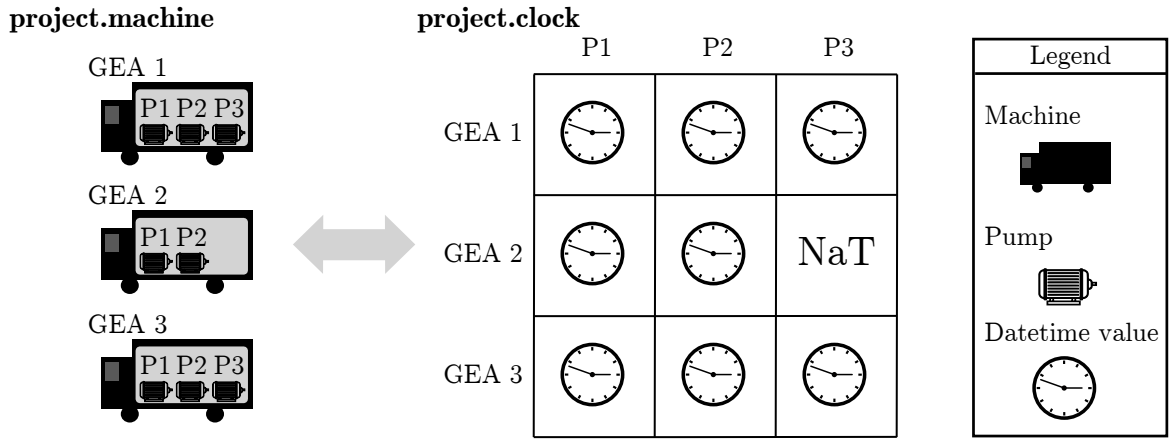


Figure 8.7: Model of a project clock. The clock keeps track of the end datetime of the last process of each pump. If a pump has been switched off or does not exist its datetime value in `project.clock` is NaT, the MATLAB datetime equivalent of NaN.

the maximum number of injections of the same type (AC or PU) per borehole. The values stored are:

`project.kpi.`

`pump` A $1 \times n$ array containing the unique identifiers of the pumps of the construction site. The index *idx* of the array corresponds to the other arrays of `project.kpi` to create a reference between pump name and the statistical performance value's position in the individual array.

`cdf_small_maintenance` A $1 \times m$ array containing the support points of the discrete ICDF which is used to draw the number of injections required for one borehole.

`injection_mean` A $m \times n$ array containing the mean duration of an injection of pump n. Note that values are the same for all $1 \leq j \leq m$.

`injection_median` A $m \times n$ array containing the median duration of an injection of pump n. Note that values are the same for all $1 \leq j \leq m$.

`injection_mean_perLevel` A $m \times n$ array containing the mean duration of the m^{th} injection of pump n.

`auslitern_mean` A $1 \times n$ array containing the mean duration of the quality process.

`auslitern_median` A $1 \times n$ array containing the median duration of the quality process.

`umsetzen_mean` A $1 \times n$ array containing the mean duration of the moving process, i. e., the time required to move the injection nozzle from one borehole to another.

`umsetzen_median` A $1 \times n$ array containing the median duration of the moving process, i. e., the time required to move the injection nozzle from one borehole to another.

`wartungStoerung_mean` A $1 \times n$ array containing the mean duration of the maintenance process.

`wartungStoerung_median` A $1 \times n$ array containing the median duration of the maintenance process.

`propabilityOfWartungPerInjection` A $1 \times n$ array containing a value $0 \leq p \leq 1$ for the probability per injection that a maintenance process takes place.

8.2 Algorithms for the Time- and Location Discrete Model

This section describes the algorithms implemented in `GBController` to run the simulation. First, the general program sequence of the simulation is explained, section 8.2.1. Then the path search, section 8.2.2, and handling of situations in which no free path can be found, section 8.2.3, is discussed. In section 8.2.4 the application of the cost function is explained. Methods whose functionality have already been outlined are not included in this section. These are in particular

- the computation of the duration of the injection of one borehole, section 6.1.1, which is implemented in `GBController.model.simulate_bore_injection_duration` and
- the visualization of results, Figure 8.2, with the `GBView` package, section 8.1.1.

8.2.1 Basic Program Sequence

The basic program sequence is shown in Figure 8.8. During the first step, the project p_0 is being initialized with the functions of the GBPlan package `GBModel.project`. A function to create a project scenario customized to fit the Tunnel Feuerbach project, is provided with `GBModel.project.create_project_feuerbach()`.

The function `GBController.model.simulate_project()` is executed to start the simulation of the project. It returns the project itself, the project duration, its cost, and the time spend with idle time due to shunting processes. By evoking `GBController.model.next()`, the function uses the current project state p_i to compute the next state p_{i+1} of the project, i. e.,

$$p_{i+1} = \texttt{GBController.model.next}(p_i). \tag{8.1}$$

For a complete project simulation, GBPlan executes `GBController.model.next()` in a loop until all boreholes of the tunnel have been injected. Once all boreholes have been injected, the project's status changes to `project.status` $= 1$ and the loop breaks. If the user requires a visual output of the simulation during runtime, then GBPlan creates a view of the simulation with `GBView.draw_tunnel_Feuerbach(`p`)` prior to the first execution of the `.next()` function. The algorithm uses MATLAB function handles, see MathWorks (2019b), to update the single graphical elements of the view every 100^{th} execution of the `next()` function.

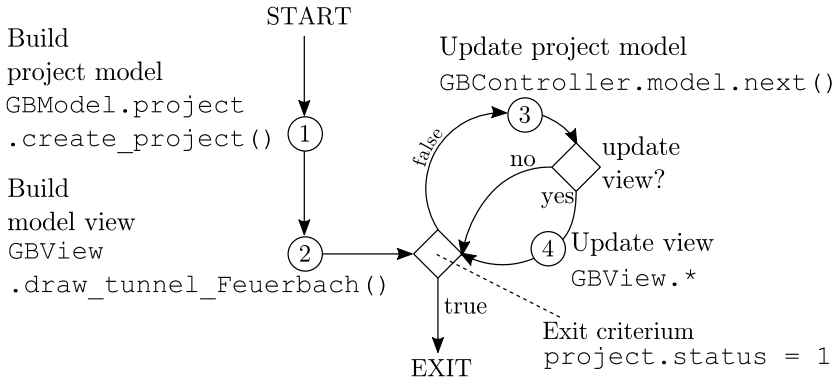


Figure 8.8: Basic program sequence for the GBPlan simulation of the MSPS3.

A principle sketch of the workings of the `next()` function is given in Figure 8.9. In general, the function changes the model in one of three ways. It identifies the next injection pump to act and (1) performs an injection with that pump, (2) moves the injection unit to another segment if the unit is not in a segment with unprocessed boreholes, or (3) lets the injection unit wait for some time if it cannot reach any unprocessed borehole. This means that the `next()` function first identifies the pump that is furthest back in time. It then checks whether or not the pump has access to an unprocessed borehole. If so, the number of finished injections of that segment is incremented by 1, and the injection process time is calculated. After adding that injection time to the internal clock of the pump, the algorithm ends. If the pump does not have access to an unprocessed borehole, then the algorithm computes the shortest path to the next segment with an unprocessed borehole. In most cases, that path leads to a neighboring segment. The machine is moved to that segment, the time which has passed is computed, added to the pump's internal clock, and the algorithm ends. In some cases, a path exists but is blocked, e. g., by another machine. If shunting is not allowed for the pump's injection unit, it cannot reach the segment of the unprocessed borehole. In this case, there is nothing to do for the pump but to wait. Therefore, the process time of waiting is added to the pump's internal clock, and the algorithm ends.

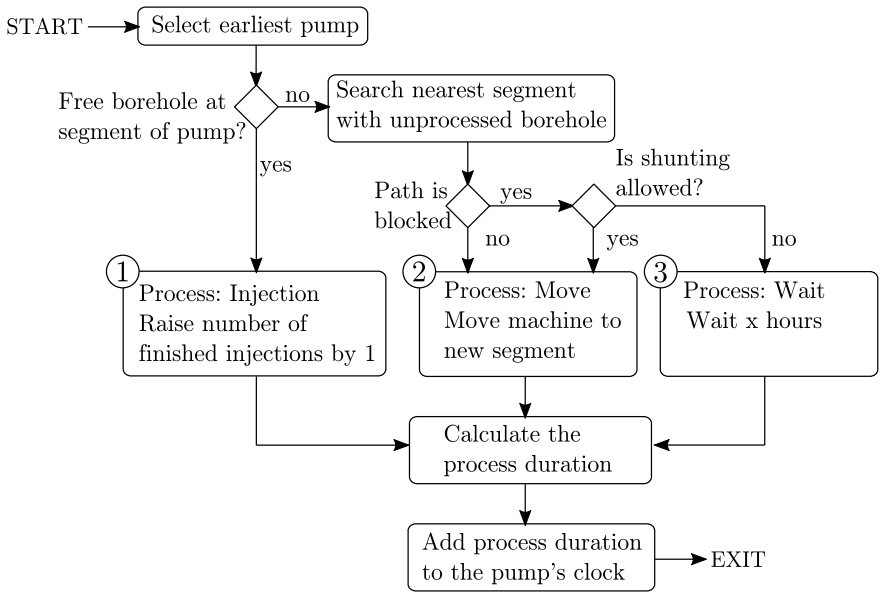


Figure 8.9: Principle sketch of program flow in function `GBController.model.next()`. After the next pump to act has been selected it is checked whether or not an unprocessed borehole is located at the pump's position. If yes, the borehole is injected, (1). If not, the nearest segment with unprocessed boreholes is searched. Suppose the path to that segment is not blocked, the machine moves to that location, (2). The machine also moves if the path is obstructed, but shunting is allowed for the machine. Only if the path is obstructed and shunting is not allowed, the machine has to wait, (3). Regardless of which of the three possible actions (injection, moving, waiting) were performed, the duration of that action is computed and added to the machine's clock.

8.2.2 Path Search and Traffic Rules

When a machine has finished all injections of section s in which it is located, it has to move to the next free section. In most cases, this section is a direct neighbor $n^1(s)$ of the current section s. This assumption allows a simplification of the path search problem. Instead of 1.) computing the transitive hull to 2.) find the shortest paths to all segments with unprocessed boreholes, 3.) using these paths to select the segment closest to the current position, and then 4.) moving the machine along that path, a proximity search is used. The proximity search is performed because the structure of the tunnel network has only a few nodes with more than two outgoing edges and because most of the time, the next segment is just adjacent to the current segment. Hence, in most cases, it is not required to check all segments' connectivity and distance to the current segment. Albeit a benchmark has not been performed, a proximity search is expected to provide results at a lower cost of computation than building the transitive hull. The search algorithm is described in section 4.4.3.3.

In GBPlan, a modified deep search, section 4.4.3.2, is performed to compute the next free segment to which the machine will move. A segment is called *free* if two conditions apply.

1. It must contain at least one not injected borehole which can be injected by the injection machine for which the path search is being performed, i. e., a PU machine can only inject PU boreholes and
2. it must not be occupied, i. e., blocked by an obstacle such as another machine.

A segment is called the *next free* segment if it is the closest free segment of all free segments. Five traffic rules are applied in the course of the route search. These are summarized in Figure 8.10, a–e.

(a) In the standard case, the next free segment is the segment closest to the start segment and free, i. e., unfinished and not blocked.

(b) If the path to the next free segment E is blocked, then the standard injection units, which are too big to pass obstacles, ignore that segment. They move to the next free segment e, which does not require them to cross a blocked segment X. In this case, the path's end segment is the segment e. In some cases, injection units, such as MobPU, are small enough to pass obstacles. They behave as if the obstacle X does not exist but do not choose the obstacle's segment itself as next free segment. In this case, the end segment is the segment E.

(c) In some cases, a next free segment cannot be found because the path to all free segments is blocked by an obstacle X. In this case, there is no next free segment, and the machine does not move. Its start segment S equals to its end segment e. Small machines, such a MobPU, can pass the obstacle X. In this case, the end segment is the segment E.

(d) The path search algorithm uses the distance w between two segments to calculate the path's length. In this case, it chooses the end segment e, which is closest to S.

(e) If the distance $w_1 = w_2$, then several segments would qualify as the end segment e. In this case, a pseudo-random segment is chosen from the set of eligible end segments. The selection method is called pseudo-random because it is difficult for the user to anticipate which segment will become the end segment during runtime. During the path search, the algorithm searches segments with small element index *idx* in `tunnel.segment(`*idx*`)` first, see section 8.1.3. If it finds more than one shortest path, it chooses the path found earlier during the algorithm's execution. Consequently, paths whose segments in `tunnel.project` have small indices are preferred over those with high indices.

The function `GBModel.tunnel.get_closest_unfinished_segment()` implements the path search. It recursively loops through the segments of the tunnel in search of segments with unprocessed boreholes. In doing this, it follows a deep search logic as described in section 4.4.3.2 and applies the before mentioned traffic rules.

8.2.3 Obstruction and Shifting

With too many injection units in the tunnel, the case may occur that injection units block each other's paths. This blocking leads to downtime, as injection units cannot reach any

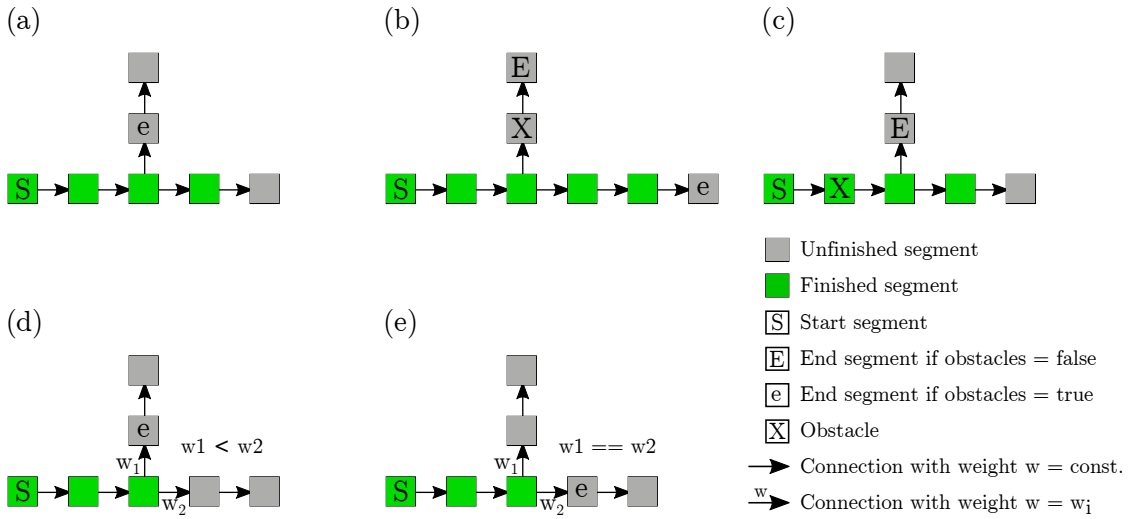


Figure 8.10: Five traffic rules for injection units. (a) Standard case. (b) Blocked segments can only be passed by small injection units such as the MobPU. (c) In some cases, a free segment can not be reached, and the injection unit has to wait, i. e., it does not move. (d) The distance between segments is used to determine which end segment is closest to the start segment. (e) If the distance of two eligible end-segments e to the start segment S are equal, then the end segment of the path is chosen pseudo-randomly. The decision which segment becomes the end segment is based on the element index of the segments in `tunnel.segment`, see section 8.1.3.

unprocessed boreholes, Figure 8.11. In the first example, Figure 8.11 (a), segments with unprocessed AC or PU boreholes exist in the tunnel. However, these segments cannot be reached by the respective injection units because they block each other's path. In the second example, Figure 8.11 (b), three injection units are in the tunnel, of which two are at segments with unprocessed boreholes. These two injection units can work. However, there is no free path to a segment with unprocessed AC boreholes for the third injection unit. In particular, to the end of a project, when the number of unprocessed segments decreases, situations in which injection units are obstructed may occur. In real life, the construction manager rearranges the units in a way that all units can work. However, this rearrangement cannot be done daily because other trades may be in the tunnel blocking efforts to shunt injection units. The construction site has implemented a rule of thumb, according to which the units can be shunted once per week. GBPlan deals with this rule by allowing units to pass obstacles once per week. This passing of obstacles is implemented by setting `machine.obstacle = true`, see section 8.1.4, during the day at which shunting is allowed if all paths to next free segments are blocked. Hence, on "shunting day" units may shunt but prefer not to if possible.

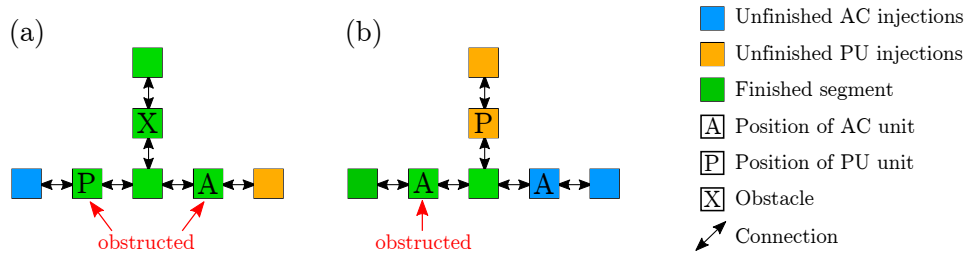


Figure 8.11: Mutual obstruction of injection units. (a) Both AC and PU injection units would be able to work. However, they cannot reach the tunnel section in which the boreholes are to be injected with their respective type of injection grout. The reason is that they are blocking each other's path. (b) While one AC and one PU injection unit can operate, the AC unit on the left is blocked. Although there are unfinished AC injection boreholes in the tunnel, the path of the left AC unit is blocked by the AC unit on the right.

8.2.4 Calculation of Project Cost

The cost model is implemented in `GBModel.project.calculate_project_cost()`. It is in line with the cost model already used for the MSPS2; see section 7.2. The MSPS3 assumes that all injection units start simultaneously and excludes costs of opportunity and delay cost.

8.3 Discussion of Results

The results of a variation analysis simulation are presented in this section. The varying parameters are the number of AC, PU and MobPU injection units. The former two include a container mounted on a truck, effectively preventing the machine from passing others. The latter is mounted on a pick-up truck. The smaller vehicle increases its maneuverability and allows it to pass machines, blocking its path in narrow tunnel tubes.

8.3.1 Simulation Scenario

The simulation is carried out on a tunnel layout that includes all tubes used by injection units during the Tunnel Feuerbach project. In addition to the primary tubes 251H/F and 258H/F, this layout includes the connection buildings VBWC to VBWF and the entry point *Zwischenangriff Prag*. Apart from allowing access to boreholes in the connection tubes themselves, these connection buildings serve as shunting facilities. A sketch of the tunnel layout is shown in Figure 8.2. Note that the entry point *Zwischenangriff Prag*, see Figure 5.22, serves as a connection between the four primary tubes 251H/F and 258H/F tubes but is not depicted in the `GBView` visualization, because it does not contain any boreholes to be injected.

The starting positions of the machines are $idx = 1$ to $idx = n$ in `tunnel.segment(`*idx*`)` with n being the number of machines in the tunnel. These segments are all located in tube 251F, with `.segment(1)` being connected to the entry segment *Zwischenagriff Prag*, i. e., from here the machines can navigate to each of the four primary tubes. Even though the start positions are not realistic – a tunnel segment's width is 0.5 to 1 m, which is too small for an injection unit – the traffic rules, section 8.2.2, will move the machines to a realistic position during the first loops of the simulation. To arrange the injection units in this way results in some idle time due to machines' shunting. However, this idle time is assumed to be negligible compared to the total idle time accumulated during the simulation.

During the simulation's final computation steps, when most of the segments have been processed, mutual obstruction of paths increases. Some machines are being removed to the end of the simulation, to avoid the artificial increase of idle time due to shunting. For instance, this is the case when all machines try to inject boreholes in the last tunnel tube containing unprocessed boreholes. Two conditions have been defined to signal that the simulation may remove injection units. (1) Once all AC injections have been finished, all AC injection are being removed. Hence, idle AC machines cannot block the tubes. (2) When only 400 segments with unprocessed boreholes are left, all but one AC and one PU injection unit are removed from the simulation. The value of 400 segments has been chosen by trial and error. If a MobPU unit is part of the machine park, that unit is chosen over PU units to remain in the tunnel. In order to save computation time these condition are checked every 50^{th} execution of the `next()` function.

The sample of performance data used to determine process durations for the simulation comprises all data points collected between 01.01.2018 and 31.12.2018. The start script of the simulation is stored in `GBController.run_parall`.

8.3.2 Computation Time

The computation time of the MSPS3 is very high compared to the MSPS2. Without graphical output, it takes GBPlan about 15 Minutes on hardware setup (2) GBT and 25 Minutes on hardware setup (1) BELMART to finish a single project simulation. Details on hard- and software are listed in Table C.6. The time required to assemble and store graphical output further increases the execution duration. The reason for the long computation time is, in particular, the path search algorithm, section 8.2.2, which has to check a high number of segments during each execution. A solution could be to simplify the tunnel graph during runtime by uniting sub-graphs of finished segments to a single substitute node, Figure 8.12. Instead of searching the sub-graph with its many edges, the algorithm would have to check only the substitute node, hence saving computation time. Such a solution has already been described for other graph-based problems by Wagner and Willhalm (2007) and Holzer et al. (2008).

Another solution would be to shift the path search from the *Central Processing Unit* (CPU) to the GPU. A path search algorithm optimized for solving large graphs on a GPU has been presented by Kumar et al. (2011). The algorithm accepts weighted graphs and promises a significant speed gain.

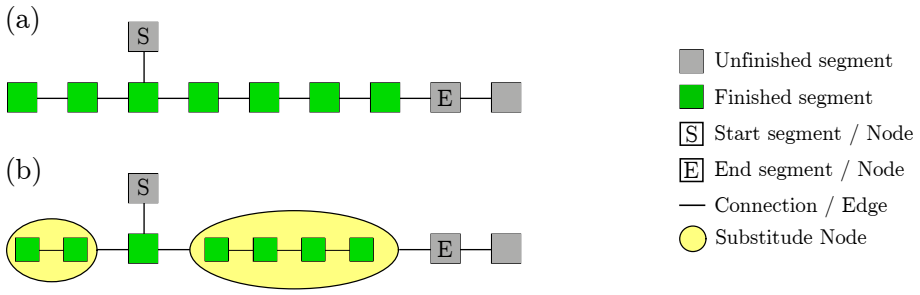


Figure 8.12: Example. Substitute nodes can be used to reduce the computation time of a path search in graphs. The finished nodes in (a) are united in sub-graphs. Substitute nodes then replace these to form a graph with a reduced number of nodes, (b). In this way, the number of nodes checked during the path search, starting at node S and ending at node E, is reduced from eight nodes in (a) to four nodes in (b). Note that the length of edges to and from the substitute nodes must be adjusted to mirror the sub-graph length.

With the given computation time, performing an optimization as discussed in section 7.3.2 seems not practical. A feasible approach is a variation analysis with pre-selected combinations of machines. Three types of injection units can be simulated: AC, PU and MobPU injection units. Assuming that

- none to five units of each type with three injection pumps each can be used on the construction site, and
- that there is no difference in the performance of injection units of the same type

the resulting number of possible combinations $c = 6^3 = 216$ take between 54 and 90 hours to be simulated. This value can be reduced by removing combinations, which are not practical. In order to inject all boreholes of the tunnel, at least one AC and one PU or MobPU unit have to be on the construction site. Furthermore, it can be assumed that not more than five units injecting PU grout are required, i. e., the sum of PU and MobPU units must be smaller or equal to five. With these assumptions, the number of combinations can be reduced to $c = 100$, i. e., a simulation time of 25 to 42 hours.

8.3.3 Structure of Result Diagrams and Tables

The result of each of the simulation's machine combinations is given in the three dimensions: project duration, project cost, and total shunt duration. The project duration is the time that has passed between the first and the last injection. The project cost is the results of the cost function, section 8.2.4. The shunting duration is the sum of the idle time spent by all injection pumps due to waiting for shunting and shunting itself. Hence, the case may occur that the shunting duration is longer than the project duration.

Three result diagrams compare the three dimensions with each other: Figure 8.13, Figure 8.15 and Figure 8.17. The diagrams show the solutions in the respective dimensions for all simulated combinations of AC, PU and MobPU injection units. The Pareto frontier

of the result is plotted as a dashed line. For better readability, labels are only given for result points on the Pareto frontier. Each label shows the number of units per unit type as: (AC, PU, MobPU). Red, dotted boxes mark areas for which an enlarged plot is being provided in Figure 8.14, Figure 8.16 and Figure 8.18. These figures show the Pareto frontier and label all result points with their corresponding combination of injection units. While this sections provides results mostly as diagrams, the exact result values can be found in appendix C.6, Table C.3 and Table C.4.

8.3.4 Project Duration-Cost Analysis

The simulation results are shown in Figure 8.13. The Pareto optimal solutions include the combinations (5,0,5), (4,1,4), (4,0,4), (4,0,3), (3,1,2), (3,0,3), (3,0,2), (2,0,2) and (1,0,1). It is noticeable that there are hardly any solutions that include PU units. The duration optimum is yielded by the combination (5,0,5) with only 260 days and a cost of 24.3 MEUR. The cost optimum is at combination (1,0,1) with 11.5 MEUR and 700 days. Neither the working solution (3,1,1) nor the backup solution (4,1,1) are part of the Pareto frontier. These are, however, close to the optimum, Figure 8.14 (top). With a cost of 17.7 MEUR and a duration of 401 days, the working solution (3,1,1) is roughly 0.7 MEUR more expensive and about seven weeks slower than solution (3,0,2), the closest combination on the Pareto frontier.

In Figure 8.13, it is noticeable that solutions can be divided into two clusters. The first cluster includes solutions with a duration $d < 460$ days (Figure 8.14, top) and the second (Figure 8.14, bottom) those with $d > 600$ days. The two clusters can be described with statistical analysis of the respective ratio of the number of AC units n_{AC} to the number of PU grout injecting units $n_{\mathrm{PU+Mob.PU}}$, EQ 8.2. The results are listed in Table 8.1.

$$x_i = \frac{n_{\mathrm{i,AC}}}{n_{\mathrm{i,PU}} + n_{\mathrm{i,MobPU}}},\ x \in X$$
$$\text{with} \tag{8.2}$$
$$n_{i,t} : \text{Number of injection units of type } t \text{ in combination } i$$

Despite the higher number of points in cluster 1, its bandwidth is more narrow than that of cluster 2. The distribution of cluster 2 is skewed ($\bar{X}_2 \neq \tilde{X}_2$) and has a standard deviation which is about three times larger than that of cluster 1. This can be interpreted as cluster 2 having in general too many units doing AC or PU injections. The uneven distribution of units results in higher project durations due to higher idle times. This interpretation is supported by Figure 8.15, which shows idle times due to shunting on the ordinate. Combinations located in cluster 2 have in general higher shunting durations than those in cluster 1.

Figure 8.14 (bottom) shows a zoom on cluster 2. It features many combinations, with the number of one type of injection unit being at least two times higher than the other. The inefficiency increases if that ratio increases, e. g., when changing from (3,0,1) to (4,0,1) the cost increase while the project duration stays constant. The duration stays constant because the PU injection unit is a MobPU and therefore not impeded in its movement

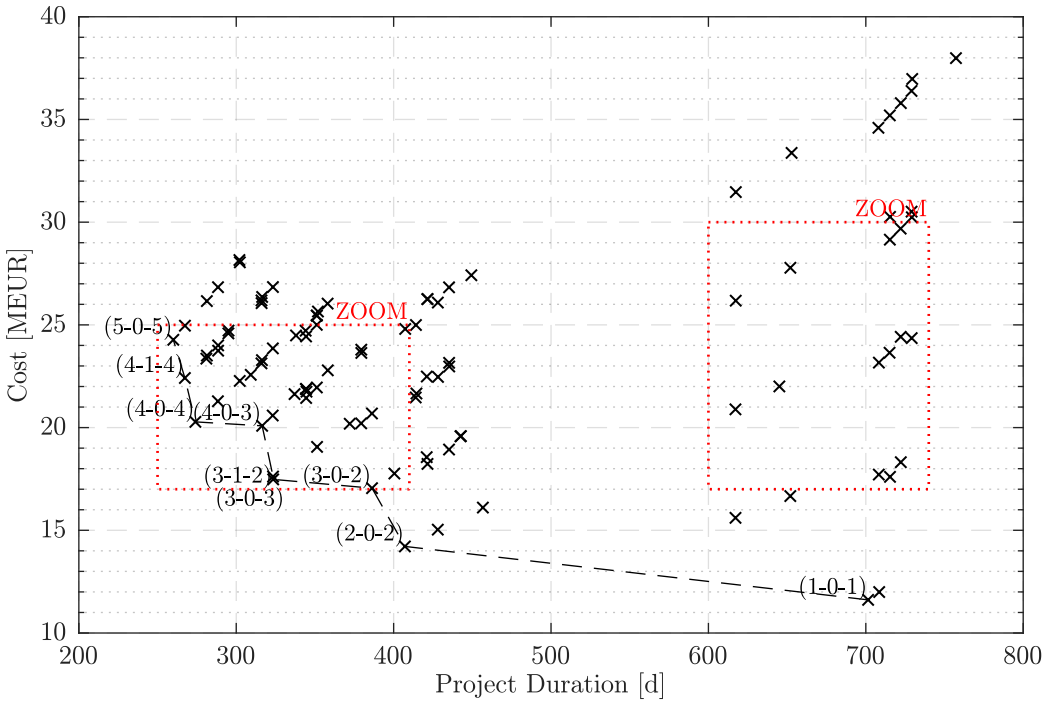


Figure 8.13: Pareto optimal solutions in the dimensions *Project Duration* and *Project Cost* for different combinations of injection units. The values of the points' labels represent the number of each machine type, i. e., (AC-PU-MobPU). An enlarged view of the zoom boxes' content is shown in Figure 8.14.

when the number of AC machines increase. This is different when comparing (3,1,0) with (4,1,0). Here, the project duration increases because the additional AC unit impedes the movement of the single PU unit. The time required for shunting is nearly twice as high for (4,1,0) than for (3,1,0); see Table C.3.

The positive effect of increased mobility can also be observed in Figure 8.14 (top). To the top right (slower and more expensive) of combinations with MobPU units one finds costlier and slower solutions in which one of the agile MobPU units has been exchanged with a bulky PU unit, e. g., from (4,1,3) to (4,2,2) in Figure 8.14 (top).

Neither the working nor the backup solution is part of the Pareto frontier. However, the working solution (3,1,1) is with a duration of ≈ 400 days and total cost of €17.800.000 close to that optimum. By replacing the PU unit of the working solution with a MobPU unit, the project duration could have been reduced by roughly 14 days, and the project cost by about €700.000.

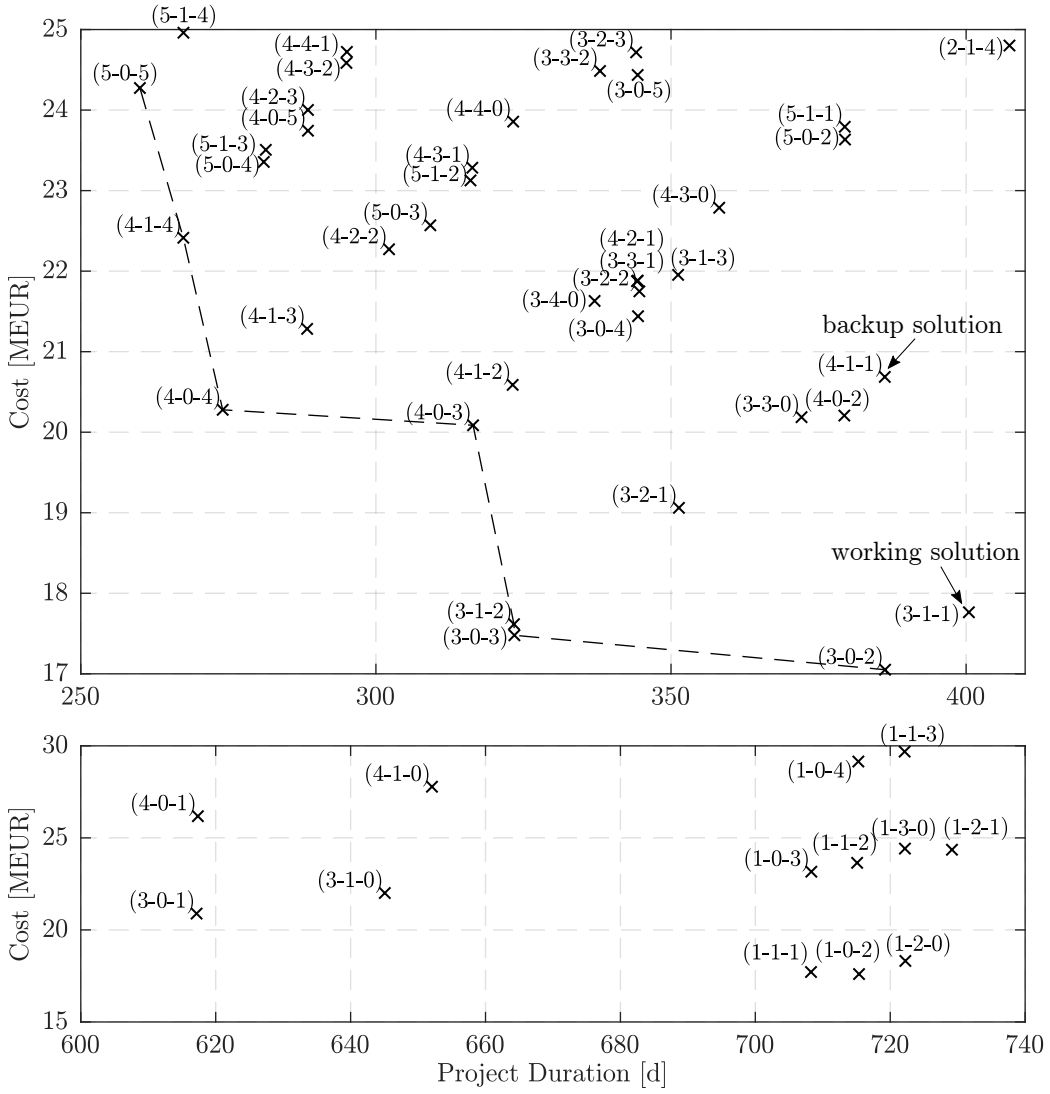


Figure 8.14: Zoomed areas of Figure 8.13. Pareto optimal solutions in the dimensions *Project Duration* and *Project Cost* for different combinations of injection units. Values of the labels represent the number of each machine type, i. e., (AC-PU-MobPU).

Table 8.1: Statistic analysis of results. The results of simulations with the MSPS3 can be divided in two clusters according to the ratio of AC to PU+MobPU injection units. $x \in X$ with $x_i = \frac{n_{\mathrm{i,AC}}}{n_{\mathrm{i,PU}}+n_{\mathrm{i,MobPU}}}$.

Function [−]	Cluster 1 $d < 460$ [d]	Cluster 2 $d > 600$ [d]
$\lvert X\rvert$	72	28
$\max(X)$	2.4	5.0
$\min(X)$	0.4	0.2
$\bar{X}$	1.0	1.3
$\tilde{X}$	1.0	0.3
$\sigma(X)$	0.5	1.6

8.3.5 Project Duration-Shunt Duration Analysis

In the same way, as in section 8.3.4, the result points can be divided into two clusters. The results of cluster 2 (project duration $d > 600$ days) have higher shunting durations. The reason is that the injection units are blocking each other's paths. The shunting duration of the working solution is close to the Pareto frontier, i. e., the frontier is about 120 days below the solution (3,1,1). Even though the project duration of the backup solution (4,1,1) is about 14 days shorter, its shunting duration is, with nearly 400 days, significantly higher, i. e., much time is spent unproductively.

When comparing the combinations on the Pareto frontier with those of the Pareto frontier of the project duration-cost diagram, Figure 8.13, it is noticeable that the two curves share many combinations or have combinations which are very close to each other. These are (5,0,5), (4,0,4), (4,0,3), (3,1,2) (PU swapped with MobPU), (3,0,3), (3,0,2) (close), and (2,0,2) (close). This effect can be attributed to the effect idle time has on overall cost, i. e., higher idle time rates suggest a higher cost.

How a constant number of one type of injection unit can block further improvement can be seen in Figure 8.16 (bottom). The combinations (2,0,1), (3,0,1), (4,0,1) and (5,0,1) have a constant number of one MobPU unit. That unit is crucial for the overall production speed. Regardless of additional AC injection units, the project duration does not improve.

As in the project cost-duration diagram, the improved mobility of the MobPU units increases the overall performance, as can be seen when comparing (5,0,4) with (5,2,2) or (5,3,1) with (5,4,0). In both cases swapping a PU unit with a MobPU unit leads to decreased idle time, lower cost, and faster project execution; see also Table 8.2.

Interestingly situations exist, in which sacrificing productivity may improve the overall cost and duration of the project. This is the case with the combinations (4,0,4) and (4,1,3), Table 8.2. As expected, (4,0,4) is both cheaper and has a lower project duration. However, the time spent on shunting is higher. This effect results from the nature of

Table 8.2: MSPS3 Results, Quick Reference. The complete list of results can be found in Table C.3 and C.4.

Label [–]	Duration [d]	Cost [MEUR]	Shunting [d]	Comment [–]
(1,0,1)	701.5	11.6	83.0	
(3,1,1)	400.5	17.8	210.0	working solution
(4,0,4)	274.0	20.3	259.8	
(4,1,1)	386.2	20.7	395.8	backup solution
(4,1,3)	288.3	21.3	251.5	
(5,0,4)	281.0	23.4	285.0	
(5,2,2)	316.1	26.1	299.0	
(5,3,1)	316.0	26.2	334.5	
(5,4,0)	323.3	26.8	359.8	

the simulation algorithm, which blindly applies the *traffic rules* outlined in section 8.2.2, instead of performing a planning forecast and optimization before moving an injection unit. The result is that solution (4,1,3) has a lower shunting duration because of its AC units coincidental move along paths which are less obstructed than those computed for combination (4,0,4). Even though the difference is, with only eight days shunting time, small, the example shows the simulation method's limits.

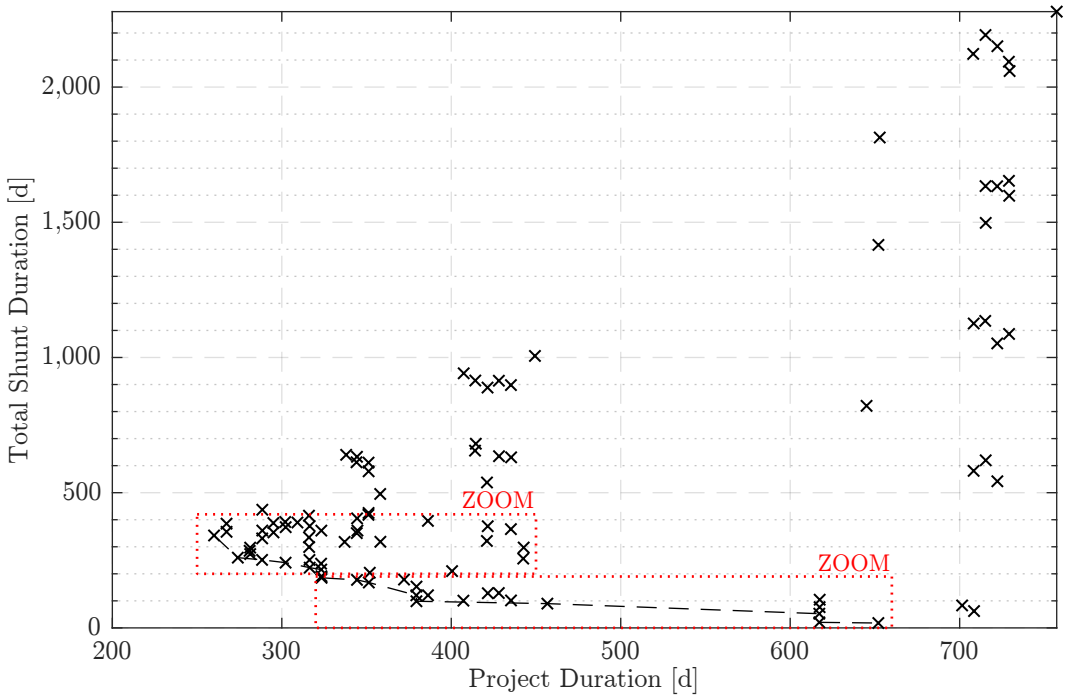


Figure 8.15: Pareto optimal solutions in the dimensions *Project Duration* and *Total Shunting Time* for different combinations of injection units. An enlarged view of the zoom boxes' content is shown in Figure 8.16.

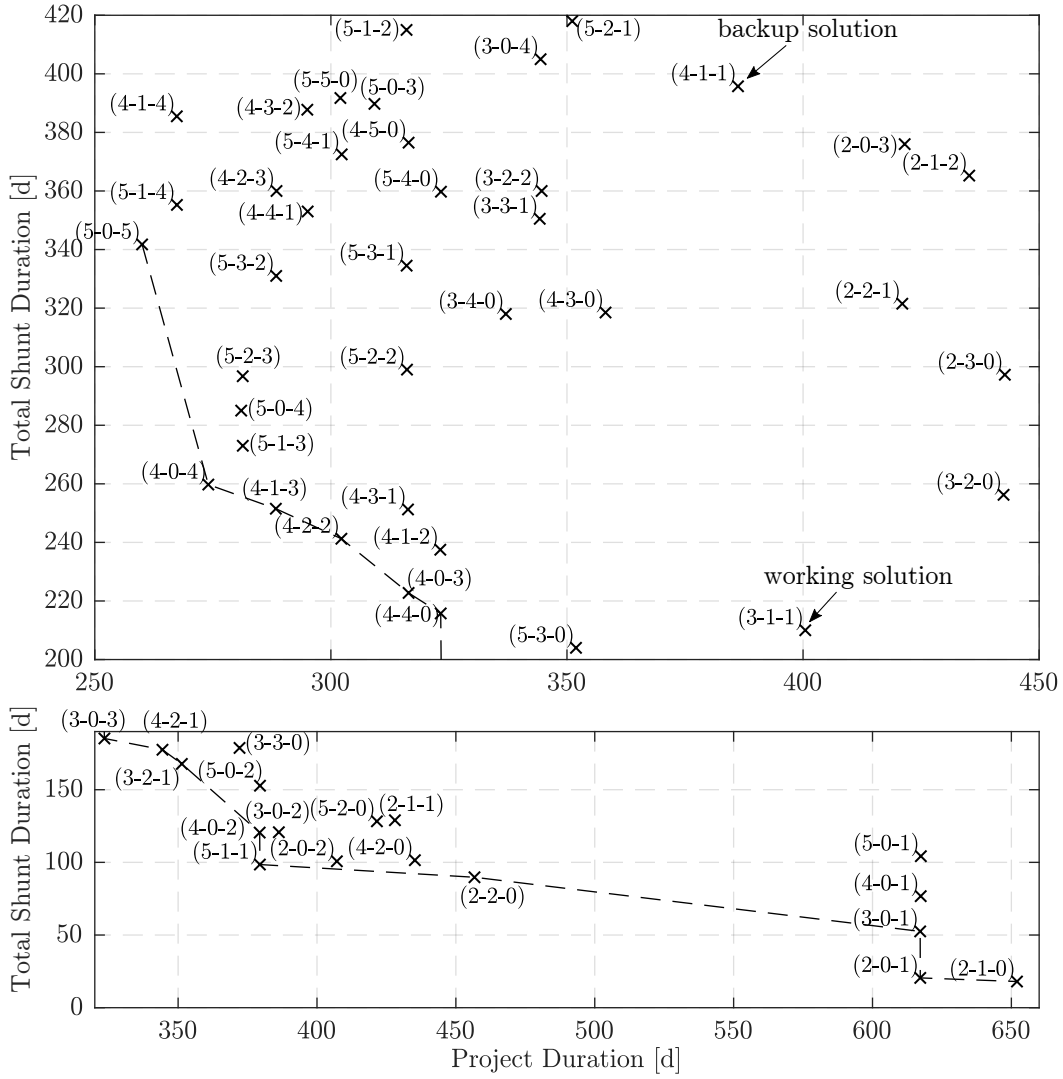


Figure 8.16: Zoomed areas of Figure 8.15. Pareto optimal solutions in the dimensions *Project Duration* and *Total Shunt Duration* for different combinations of injection units. The values of the points' labels represent the number of each machine type, i. e., (AC-PU-MobPU).

8.3.6 Project Cost-Shunt Duration Analysis

The Pareto frontier develops along combinations with only a few units, i. e., (1,0,1), (1,1,0), (2,0,1) and (2,1,0), Figure 8.18 (middle). This behavior is expected because fewer units result in a reduced chance for units to block each other. Furthermore, they have a lower upper limit for the total shunting time than combinations with many units. This limit is determined by the project duration multiplied by the number of injection pumps.

The diagram in Figure 8.17 confirms the relation between cost and shunting time, which has been assumed in section 8.3.5, i. e., higher idle time due to shunting correlates with higher total project cost. For higher shunting durations $d_s > 1{,}000$, the relation between cost and shunting duration develops linearly.

A lower boundary for the shunting duration exists. In the interval $d_s = [1, 83]$ it follows the Pareto frontier and after that, starting at combination (1,0,1), rises with an acclivity of $m = 15.44$ TEUR, EQ 8.3. In Figure 8.17 that lower boundary is plotted as dashed-dotted line.

$$c(d_s) = \begin{cases} \text{Pareto frontier} & \text{, if } d_s \leq 83 \text{ [d]}, \\ 15.44 \cdot d_s \left[\frac{TEUR}{d}\right] & \text{, otherwise.} \end{cases} \tag{8.3}$$

Hence, one day of accumulated idle time due to obstructed paths results in a minimum of 15.5 TEUR additional cost. It is likely, however, that the additional cost is higher because practical combinations, such as the working and the backup solutions, have shunting durations of $d_s < 400$ days. In this area, costs diverge most from the minimum boundary given by EQ 8.3. Furthermore, a minimum shunting duration of $d_s = 83$ days exists in the simulation. This minimum must not be confused with the real minimum in the sense of optimization of routes of the injection units. It results from the applied, static traffic rules described in section 8.2.2.

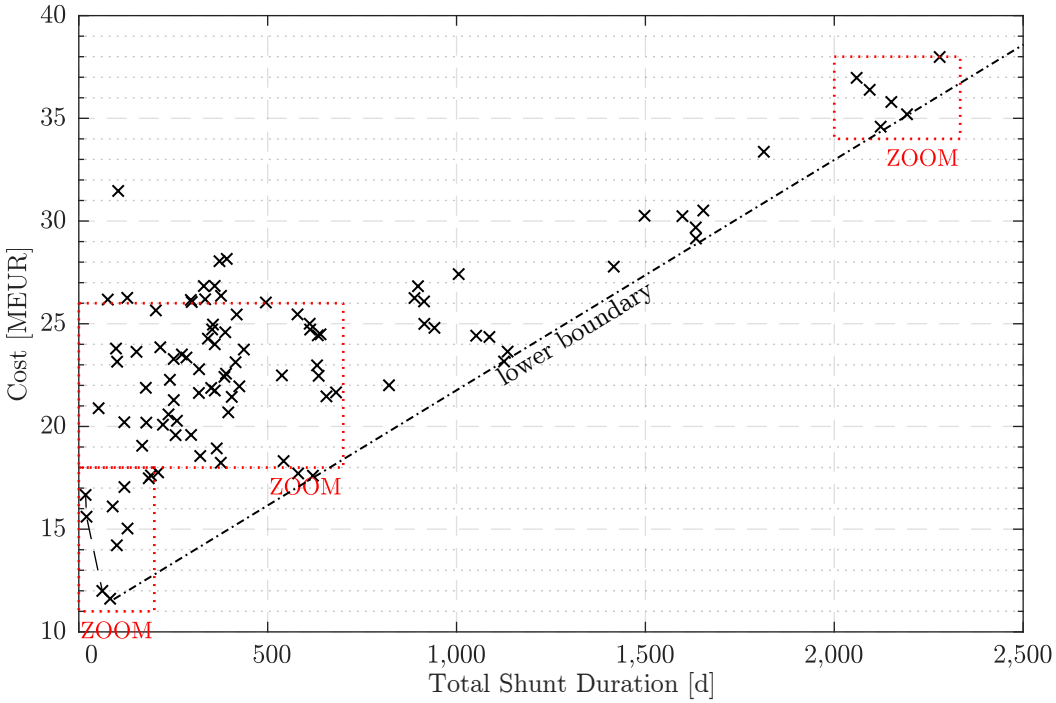


Figure 8.17: Pareto optimal solutions in the dimensions *Total Shunting Time* and *Project Cost* for different combinations of injection units. An enlarged view of the zoom boxes' content is shown in Figure 8.18.

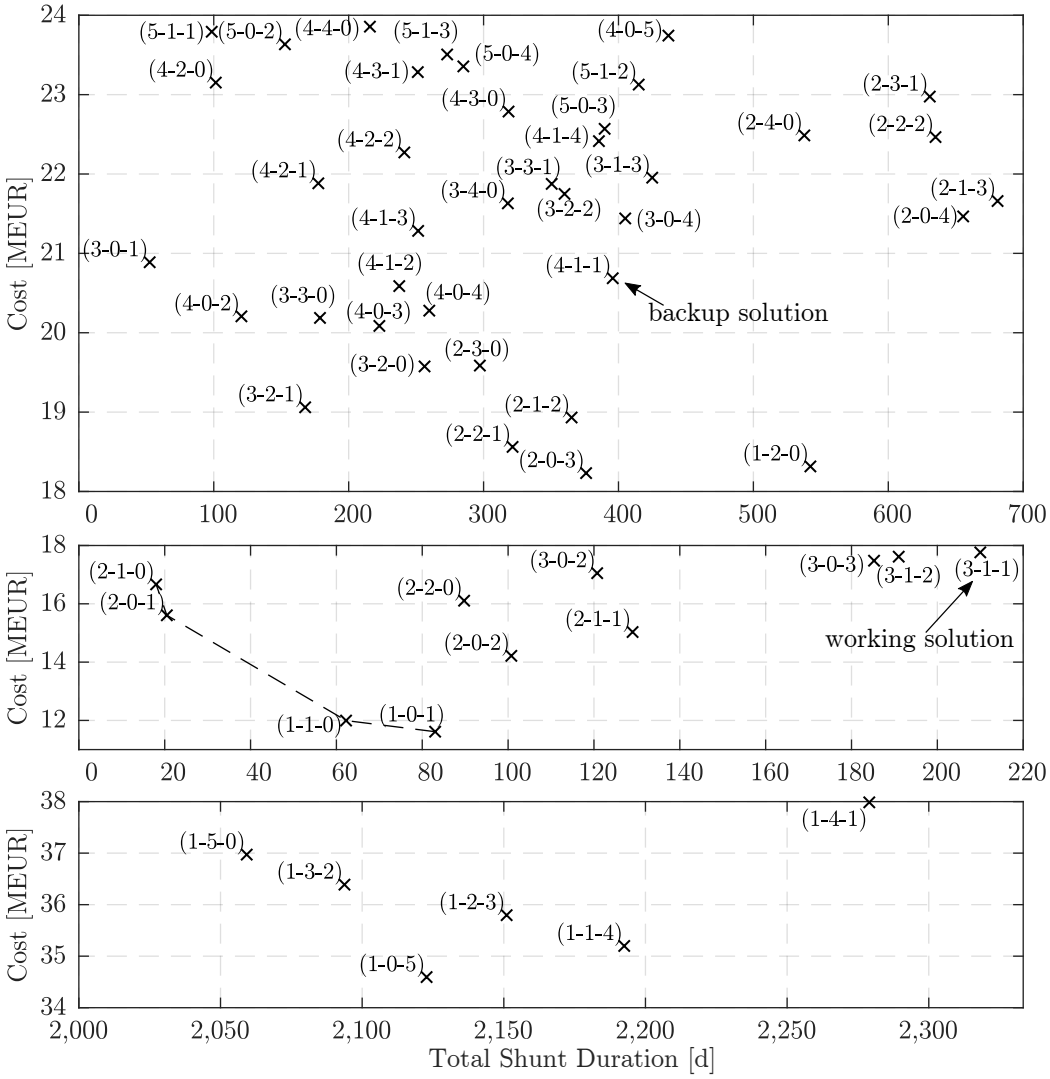


Figure 8.18: Zoomed areas of Figure 8.17. Pareto optimal solutions in the dimensions *Total Shunting Time* and *Project Cost* for different combinations of injection units. The values of the points' labels represent the number of each machine type, i. e., (AC-PU-MobPU).

8.3.7 Summary

The simulation results are particularly good with regard to the simple traffic model. The mutual interaction of the injection machines is represented conclusively and comprehensibly. Essential relations between the project cost, project duration, and the total shunting duration are sound. The working and the backup solution are not on the Pareto frontier; however, close to it. By replacing a PU with a MobPU unit, cost reduction of €700.000 and 14 days project time seem possible.

It can be expected, that a variation analysis of start dates, like for the MSPS2, would allow the identification of solutions with even lower project cost and duration. Such a variation, however, requires a high number of parameter iterations. Many iterations are, with regards to the high cost of computation of the MSPS3, not practical. Suggestions to solve this problem are given in section 8.3.2 and include the implementation of improvements of the graph's structure during simulation runtime and shifting parts of the computation from the CPU to the GPU.

The simulation results are suited for a qualitative analysis of the construction process. How far they are suitable to provide quantitative insight remains to be seen. Real project costs remain confidential, the entire shunting time was not logged, and extended downtime occurs due to external, unverifiable decisions. Hence, comparing simulation results with real project data is impossible. This comparison, together with the improvement of the computation speed, and further research regarding applicable traffic rules, should be the next step to improve the model.

9 Summary and Outlook

Now is no time to think of what you do not have. Think of what you can do with what there is.

Hemingway (1952)

This chapter contains two sections. In the first, a summary of the most important research results of this thesis is presented and linked to the objectives defined in chapter 3. In the second section, an outlook in the form of a summary of open questions and unsolved problems is given.

9.1 Summary

A comprehensive literature review on the topic of construction management simulation for large injection projects in tunneling is presented in chapter 2. The identified research gaps are pointed out in section 2.5 and used to define the objectives of this thesis in chapter 3. These objectives were all achieved. After a summary of the thesis' methodological framework in chapter 4, a knowledge discovery process from the literature is used to structure an analysis of the construction site under investigation in chapter 5. The analysis includes a detailed description of the construction site, the collected raw data on construction processes, and its processing into input data for GBPlan – a *Single Purpose Simulation* (SPS) developed within this thesis's framework. During the subsequent data mining process, a statistical analysis of the data is conducted in section 5.5. The main findings of the statistical analysis are:

- A ranking of the construction processes' influence on the construction site performance has been established in section 5.5.2. The ranking reveals that most of the time is spent with the injection process, directly followed by holidays and waiting time. The latter is a consequence of long idle periods of machines on the construction site.
- A MUDA analysis has been conducted in section 5.5.3. It reveals that at least 75 percent of the time spent on the construction site can be associated with one of two types of MUDA.
- An analysis over a 24-hour workday in section 5.5.4 shows the rhythm according to which the construction site functions. For instance, the number of machine handling processes has its minimum at shift changes, malfunctions and maintenance processes

have a peak around three o'clock in the morning, standstills due to decision planning processes tend to happen during management's lunchtime, and quality assurance is conducted as planned, i. e., shortly after each shift change.

- An analysis of the shifting, maintenance, and injection process in section 5.5.5 shows that the duration of these processes are log-normal distributed.
- The visualization of time-location data of injection processes in section 5.5.6 supports the general understanding of the construction site. However, it is not suited to make statements on the performance of the construction site.
- The visualization of the number of injections per day over time in section 5.5.7 in comparison to the total duration spent with a process per day allows making statements on what can be called life-cycle phases of the construction site. A ramp-up phase, a production phase, and a ramp-down phase can be distinguished. Furthermore, the results imply that significant *Key Performance Indicator*s (KPIs) can be calculated from these parameters.
- Between 60 and 70 percent of all boreholes are injected one time with *Acrylate* (AC) and *Polyurethane* (PU) agent each. The analysis in section 5.5.8 suggests that the reason for the change of the injection strategy during the first year of the project has not been made based on geological conditions in the tunnel.
- An analysis of the relation between grout volume and injection time in section 5.5.9 shows that 95 percent of all boreholes are injected with less than 100 liters. Furthermore, the minimum injection time is limited by the maximum flow of grout material, injections are often stopped at multiples of 10 liters, and the acclivity of linear curves fitted to injection duration over grout volume diagrams differs between tunnel tubes. The latter would render forecasts based on a transfer of parameters from one tube to another futile.

The implemented functions are suitable to support managers in gaining knowledge of their construction project. They provide information, which is crucial for decision-making. For instance, suitable KPIs can be identified and analyzed from the available data. Furthermore, the information is automatically extracted from the already existing data, i. e., without any additional workload for construction site personnel. **Herewith objective 1 is achieved.**

A prediction of injected grout volumes is carried out in section 5.6. Three different methods are evaluated.

Method 1: Cross-correlation The first method uses cross-correlation and takes advantage of the fact that the two tunnel tubes are distanced close together. The assumption is that conditions in one tube are mirrored by those of the other, i. e., boreholes in the same area behave similarly. That assumption is partly right, as a correlation between the injection volumes of the two tubes exists. That correlation is, however, not strong enough to be used as a reliable predictor.

Method 2: Markov Chain and mean values The second method uses *Markov Chain*s (MCs) in combination with mean values of injection volumes. It predicts future injection grout volumes based on sampled volumes from the latest injections. The *Mean Absolute Percentage Error* (MAPE) of the forecasts of the total grout volume per

tunnel tube is 12 percent in tube 251H and 10 percent in tube 258H. Furthermore, the influence of the sample size, and the forecast duration on the prediction accuracy has been analyzed. As expected, the MAPE increases with increasing forecast duration and increasing sample size. In general, it can be stated that the model predicts grout volumes with an accuracy of $\delta_{\mathrm{MAPE}} < 1$ % for the next week, $\delta_{\mathrm{MAPE}} < 2$ % for the next month, and $\delta_{\mathrm{MAPE}} < 6$ % for the next quarter.

Method 3: *Artificial Neural Network* The third method uses sampled data from one tube to train a *Feedforward Neural Network* (FNN) and then uses that FNN to predict future injection grout volumes in the other tube. For short forecast distances not greater than 50 m, the method can forecast the grout volume with an accuracy of $\delta_{\mathrm{MAPE}} < 12$ % in roughly half of the predicted cases. Even though the results seem promising, the model is not good enough to serve as a reliable predictor. Furthermore, the forecasting distance, for which somewhat acceptable results are produced, is inferior to Method 2.

From the three tested forecasting methods, Method 2 is superior. It is therefore adapted to serve as a forecasting method for the following prediction models. **Herewith objective 2 is achieved.**

In chapter 6, a project time forecast model named *Multi Server Production System* (MSPS) has been developed and tested. The MSPS uses *Discrete Event Simulation* (DES) and *Monte-Carlo Simulation* (MCS) to simulate a single injection unit consisting of several injection pumps. Samples from past processes serve as input for the forecasting method, predicting the time required to process future boreholes. The model benefits from the experience gained during the development of the grout volume forecast model, from which it inherits MCs. The MSPS uses these to estimate the occurrence of disturbances that may lead to multiple injections of the same borehole. Furthermore, the model includes disturbances, which result in downtime, the movement process of injection nozzles to the next borehole, the injection process, and the calibration process.

The simulation results had to be made comparable with measured data to validate the modal. For this, a method to interweave erratic idle periods from the measured data into the simulation has been developed. Furthermore, a method dealing with the case that the totality of available process samples is too small to be used in the simulation has been implemented.

Five different sample strategies are analyzed. These differ in the way samples are used. For samples of process durations, the options are (1) sampling with *Cumulative Distribution Function* (CDF), (2) mean values, or (3) median values. For a sampling of occurrence of disturbances that may lead to additional injections of boreholes, the options are (1) sampling with MC or (2) the assumption that each borehole is being injected with the planned number of injections. The method which comes closest to that currently used by practitioners is used as a benchmark. This benchmark method combines mean value sampling and the assumption of two injections per borehole for the prediction.

A convergence analysis is performed. It shows that only a small number of about 100 iterations are required to let the simulation converge towards a solution, which is stable enough to be used for the forecast.

In a first experiment, the methods' accuracy to forecast the total processing duration required for one tunnel tube is assessed. Here, the best results are yielded by combining mean value sampling and MC. The average MAPE of this combination is just 12.2 percent.

In a second experiment, the methods' accuracy to forecast the processing duration of a limited number of boreholes based on a limited number of samples is being assessed. The method, which performs best uses a combination of median value and MC. It provides weekly forecasts with a MAPE smaller than 4 percent and monthly forecasts with a MAPE smaller than 10 percent.

In both experiments, results are superior to those yielded by the benchmark method. The results show that the MSPS represents a clear improvement compared to the method currently used by practitioners. **Herewith objective 3 is achieved.**

In chapter 7 a project time-cost forecast model named *Multi Server Provduction System 2* (MSPS2) is presented and tested. Like the MSPS, the MSPS2 uses DES and MCS. It simulates a production system consisting of a variable number of AC and PU injection units with a variable number of injection pumps for each unit. Furthermore, the model takes the order in which AC and PU injections must be performed into account. This leads to the boundary condition that PU units cannot overtake AC units, even if the former require less time per injection.

The MSPS2 is used to calculate the project's time-cost Pareto frontier. The variable parameters are the number of injection units or pumps, and each injection unit's start date. The approximation of the Pareto frontier is performed with *Genetic Algorithm* (GA). For this purpose, the MATLAB function `gamultiobj` is adapted to handle mixed-integer optimization problems.

The cost model implements *Costs of Opportunity* (COOP) and a *Delay Cost* (DC) in addition to more traditional cost factors, such as energy cost over time or one-time payments to buy machines. With these, the simulation calculates machine deployment dates, which take place as late as possible but do not result in excessive overtime cost.

In a first numerical experiment, the time-cost Pareto frontier for a varying number of pumps, which share the same starting date, is calculated. The results are compared to the working and backup solution used on the construction site. While the backup solution lies on the calculated Pareto frontier, the working solution does not. It is, however, close to that ideal front.

In a second numerical experiment, the time-cost Pareto frontier for a varying number of injection units with three pumps is calculated. Three optimization scenarios are analyzed.

Scenario 1 models a situation in which a new injection unit can start to work each Monday of the project. The Pareto frontier calculated for this scenario includes the working and the backup solution. The solutions identified by GBPlan do not include the best possible solution. However, with a cost difference to the optimal solution of around €1,000 and a forecasted total project cost of €10,250,000, the margin to the perfect solution is negligibly small.

Scenario 2 models a situation in which a minimum of two weeks must pass between the deployment of two injection units of the same type (AC or PU). This minimum

duration takes the time into account, which managers need to train workers and to get the highly customized injection units running. The results are sound, not only in themselves but also in comparison to those of Scenario 1. Furthermore, they align with project and site managers' decisions for the real construction site.

Scenario 3 assumes that the project's injection strategy changes after half of all boreholes have been processed. The simulation results show that significant optimization potential exists, depending on when information about the changed injection strategy reaches the construction site. If no changes to the execution planning are made, the change of the injection strategy results in overspending of €2,617,000 and 80 days delay of the anticipated finishing date. Depending on whether cost or deadlines are the priority, the simulation offers different mitigation solutions. One solution would require to deploy one additional machine to decrease the overtime from 80 days to one month and at the same time eliminate overspending. Another solution would change the machines' deployment dates and eliminate overtime while decreasing overspending to €889,000.

Comparing solutions on and close to the Pareto frontier provides valuable insight into the cost and project duration development between machine and deployment strategy configurations. The optimization algorithm identifies near-optimal solutions for the number of machines and their deployment dates. If the injection strategy is changed during the project, the simulation finds mitigation solutions, significantly reducing overtime and overspending. **Herewith objective 4 is achieved.**

In chapter 8 a project time-cost forecast model under consideration of confined work environment named *Multi Server Production System 3* (MSPS3) is presented and tested. The MSPS3 combines the forecasting functionality for borehole processing durations of the MSPS2 with Graph Theory. The latter is used to create a basic tunnel course model, in which each of the two tunnel tubes and their connection buildings is divided into one meter long sections. Each of the sections is represented by a node in a graph.

Furthermore, rules for the movement of injection units in the tunnel, i. e., the graph, are combined with a search algorithm to find the shortest path of the injection unit's current position to the next unprocessed borehole. These rules allow injection units to block each other's path. For the MSPS3, the third type of injection unit is being introduced. While AC and PU units are mounted on trucks and may block each other, the *Mobile Polyurethane Injection Unit* (MobPU) units are mounted on smaller pick-up trucks. These may overtake bigger units in the narrow tunnel. A shunting mechanism is implemented to avoid those units which block each other end up sitting idle for the rest of the project. This mechanism follows the same rules as the real construction site, i. e., once per week, some of the units move from tunnel tubes to the connection buildings to let other units pass.

The implementation of the MSPS3 in GBPlan includes a visualization, which allows observing the movement of the units and each tunnel segment's current status, i. e., not started, in work, finished, both during the calculation and a posteriori. For the latter, a *Graphics Interchange Format* (GIF) output file is created.

A side effect of the more complex model is a significant increase in the cost of computation. It is so high that the application of GA to find near-optimal solutions for start dates

for each injection unit is impractical. Instead, a parameter study is being performed to find the best combinations of machines, assuming that all machines are being deployed simultaneously.

Three Pareto frontiers, namely cost-project time, cost-shunt time, and project time-shunt time, are computed in a numerical experiment. The variable parameters are the number of machines. The results are promising. In particular, the qualitative relation between the three analyzed dimensions is sound. According to the results, the most efficient ratio of AC to PU injecting units is 1, i. e., those combinations are closest to the Pareto frontier. In most cases, the performance of MobPU units are superior to the less agile PU units. This superiority means that trading PU for MobPU units improves the construction site's time-cost performance. Neither the working solution of AC, PU, and MobPU units used on the construction site nor the backups solution is part of the Pareto frontier. However, both are part of a cluster of combinations, which minimize the project duration. Of these, the working solution is close to the cost optimum. However, simulation results show that the opportunity for improvement exists. By replacing a PU with a MobPU unit, cost reduction of €700.000 and 14 days project time would be possible. In comparison to the results of the MSPS2, it is apparent that the boundary conditions dictated by the narrow tunnel environment have a significant influence on the performance of discrete machine combinations. **Herewith objective 5 is achieved.**

All five objectives of the study have been archived. However, in the course of achieving those objectives, further questions arose. These are summarized in the following section.

9.2 Outlook

The results are promising. Significant improvement of forecast results compared to manual forecasts are possible, and the presented optimization methods show good results. However, many questions that came up during this study remain to be answered by future scientists and practitioners. Some may be solved based on the collected data; others require additional data collection and, therefore, managers of injection projects who agree to share their digitized process data.

Large injection projects such as that investigated in this study are a rare opportunity. Therefore, it should be examined how the presented methods can be transferred to smaller projects. This question has partly been answered in chapter 6. The methodology applied in that chapter guides to answer that question.

The research project started well after the start of the investigated injection project. It would have been interesting to understand how forecasts and recommendations computed by GBPlan would have influenced the course of the project. Therefore, either GBPlan or its methodological framework should be applied to a project starting during the planning phase. Since no digitized project data can be available in the planning phase, the existing data from the Feuerbach project can be used and modified to calculate initial forecasts and recommendations for the number and dimensioning of the machines.

The presented methodology supports a multi-machine setup. Future research projects should make use of this by including additional information to improve simulation results. In particular, preceding trades should be integrated into the simulation. For instance, if process durations and location of drilling jumbos are known, an improvement of forecast results should be possible. This effect can be expected to grow when additional upstream trades of the value stream are included. Here the universities have an essential role to play. Among the stakeholders, who are often defined by their opposing financial interests, Universities may take a neutral position. If research results are released after the project's financial claims have been solved, such a neutral position will allow them to collect data from different building companies.

Apart from the apparent advantage of integrating upstream process durations and machine locations, using additional machine parameters may improve the accuracy of prediction results. For instance, the machine parameters already collected from drilling jumbos or *Tunnel Boring Machine* (TBM) may give insight into the surrounding rock's porosity. Such information could be used as input for an *Artificial Neural Network* (ANN), which then computes the expected grout volumes.

Concerning the GBPlan software, the implementation of an easy to use *Graphical User Interface* (GUI) would allow practitioners without programming or simulation experience to use the tool. If provided with the ability to edit the tunnel geometry and add traffic rules, such a GUI would allow managers to quickly modify and test scenarios. Usability would also be increased by porting the GBPlan source code to a language, which does not require licensing. Also, execution of GBPlan or its methodology on a cloud server and delivery of results via web-solution, as already practiced by the eguana-SCALES software, could be a solution. Furthermore, decreasing the cost of computation of the MSPS3 model would increase acceptance upon users. Some approaches suitable for this purpose are presented in section 8.3.2.

Finally, this study deals with a small number of selected methods. Namely DES, MCS, MCs and Graph Theory. Future research should investigate how the combination of these methods performs against that of others. For instance, continuous methods are already successful used in the wind energy sector for a posteriori analysis during adjudication procedures to determine eligibility and amount of variation claims or delay costs. Such methods have the potential to improve forecast and optimization results further.

Bibliography

Collections

Abraham, Ajith, Lakhmi Jain, and Robert Goldberg, eds. (2005). *Evolutionary Multiobjective Optimization*. Theoretical Advances and Applications. London, UK: Springer, 2005.

Agarwal, Pragya and Andr Skupin, eds. (2008). *Self-Organising Maps*. Chichester, UK: John Wiley & Sons, Ltd, 2008.

Armstrong, J. Scott, ed. (2001). *Principles of Forecasting:* A Handbook for Researchers and Practicioneers. Vol. 1. New York, NJ: Springer Science, 2001.

Banks, Jerry, ed. (1998a). *Handbook of Simulation*. New York, NY: John Wiley & Sons, Inc., 1998.

– (1998b). "Principles of Simulation". In: *Handbook of Simulation*. Ed. by Jerry Banks. New York, NY: John Wiley & Sons, Inc., 1998. Chap. 1, pp. 3–30.

Berkhahn, Volker (2007). "Überblick zum Themenbereich Netzwerkgerechte Prozessmodellierung. Grundlagen, Methoden, Anwendungen und Perspektiven zur vernetzten Ingenieurkooperation". German. In: *Vernetz-kooperative Planungsprozesse im Konstruktiven Ingenieurbau*. Ed. by Uwe Rüppel. Berlin: Springer, 2007, pp. 21–30.

Conrads, Alena et al. (2016). "Prozesssimulation für die Leistungsermittlung und -planung beim maschinellen Tunnelbau". German. In: *Taschenbuch für den Tunnelbau 2017: Kompendium der Tunnelbautechnologie Planungshilfe für den Tunnelbau*. Ed. by Deutsche Gesellschaft für Geotechnik e.V. Weinheim, Germany: Wiley-VCH, 2016, pp. 166–198.

Conrads, Alena et al. (2018b). "Simulation of production and logistic processes in mechanized tunneling: Simulation based maintenance and availability analysis". In: *SFB 837 Brochure 2018*. 2018, pp. 49–50.

Grabe, Jürgen, ed. (2018). *Digitale Infrastruktur und Geotechnik 2018*. German. Vol. 42. Veröffentlichungen des Instituts für Geotechnik und Baubtrieb der TU Hamburg. Institut für Geotechnik und Baubetrieb, Technische Universität Hamburg, 2018.

Han, Sangwon, SangHyun Lee, and Moonseo Park (2014). "Dynamic Project Management: An Application of System Dynamics in Construction Engineering and Management". In: *Intelligent Systems, Control and Automation: Science and Engineering*. Springer Netherlands, 2014, pp. 219–231.

Jodehl, Annika, Markus König, and Markus Thewes (2019). "Process simulation in mechanized tunneling: From forecasting to real-time process controlling with continuous model updates with process data". In: *SFB 837 Brochure 2019*. 2019, pp. 53–54.

Joines, Jeffrey A. and Stephen D. Roberts (1998). “Object-Oriented Simulation”. In: *Handbook of Simulation*. Ed. by Jerry Banks. New York, NY: John Wiley & Sons, Inc., 1998. Chap. 11, pp. 397–427.

Józefowska, Joanna and Jan Weglarz, eds. (2006). *Perspectives in Modern Project Scheduling*. 1st ed. Vol. 92. International Series in Operations Research & Management Science (ISOR). Boston, MA: Springer, 2006.

Kamimura, Ryotaro (2011). “Information-Theoretic Approach to InterpretInternal Representations of Self-Organizing Maps”. Applications and Novel Algorithm Design. In: *Self Organizing Maps*. Ed. by Josphat Igadwa Mwasiagi. 2011, pp. 3–32.

Kiran, Ali S. (1998). “Simulation and Scheduling”. In: *Handbook of Simulation*. Ed. by Jerry Banks. New York, NY: John Wiley & Sons, Inc., 1998. Chap. 21, pp. 677–720.

Little, John D. C. and Stephen C. Graves (2008). “Little’s Law”. In: *International Series in Operations Research & Management Science*. Vol. 115: *Building Intuition: Insights From Basic Operations Management Models and Principles*. Ed. by Dilip Chhajed and Timothy J. Lowe. New York, NY: Springer Science + Business Media, 2008. Chap. 4, pp. 81–100.

Lu, Le et al., eds. (2017). *Deep Learning and Convolutional Neural Networks for Medical Image Computing*. Cham, Switzerland: Springer International Publishing, 2017.

Maletic, Jonathan I. and Andrian Marcus (2005). “Data Cleansing”. In: *Data Mining and Knowledge Discovery Handbook*. Ed. by Oded Maimon and Lior Rokach. New York, NY: Springer Science & Business Media, 2005, pp. 21–35.

Neumann, John von (1951). “Various techniques used in connection with random digits”. In: *Monte Carlo Method*. Ed. by A. S. Householder, G. E. Forsythe, and H. H. Germond. Vol. 12. National Bureau of Standards Applied Mathematics Series. Washington, DC: US Government Printing Office, 1951. Chap. 13, pp. 36–38.

Pidd, Michael, ed. (1989). *Computer modelling for discrete simulation*. Chichester, UK: Wiley, 1989.

Rüppel, Uwe, ed. (2007). *Vernetz-kooperative Planungsprozesse im Konstruktiven Ingenieurbau. Grundlagen, Methoden, Anwendungen und Perspektiven zur vernetzten Ingenieurkooperation*. German. Berlin: Springer, 2007.

Schriber, Thomas J. and Daniel T. Brunner (1998). “How Discrete-Event Simulation Software Works”. In: *Handbook of Simulation*. Ed. by Jerry Banks. New York, NY: John Wiley & Sons, Inc., 1998. Chap. 24, pp. 765–811.

Seiwert, Lothar J. (1984). “Beachten Sie die Leistungskurve”. German. In: *Das 1 × 1 des Zeitmanagement*. Springer Berlin Heidelberg, 1984, pp. 32–35.

Sokolov, Boris, Dmitry Ivanov, and Alexandre Dolgui, eds. (2020). *Scheduling in Industry 4.0 and Cloud Manufacturing*. Cham, Switzerland: Springer International Publishing, 2020.

Uhl, Sebastian (2011). “Rahmenbedingungen und Herausforderungen des Logistimanagements in der Bauwirtschaft”. German. In: *Digitale Baustelle – innovativer Planen, effizienter Ausführen. Werkzeuge und Methoden für das Bauen im 21. Jahrhundert*. Ed. by Willibald Günthner and André Borrmann. 1st ed. Heidelberg: Springer Nature, 2011. Chap. 5.1.1, pp. 208–210.

Yagiz, S. et al. (2010). “Factors influencing performance of hard rock boring machines”. In: *Rock Engineering in Difficult Ground Conditions – Soft Rocks and Karst*. Ed. by I. Vrkljan. London, UK: Taylor & Francis, 2010, pp. 696–700.

Monographs

Aarts, Korst (1989). *Simulated Annealing Boltzmann Machines*. Chichester, UK: John Wiley & Sons, 1989. 288 pp.

Alexandre, Victor (2020). *Eléments de praxéologie*. French. Paris, France: Editions L'Harmattan, 2020.

Allen, Theodore T. (2011). *Introduction to Discrete Event Simulation and Agent-based Modeling. Voting Systems, Health Care, Military, and Manufacturing*. 1st ed. London: Springer, 2011.

Anastassiou, George A. (2011). *Intelligent Systems: Approximation by Artificial Neural Networks*. Berlin, Heidelberg: Springer, 2011.

Bala, Bilash Kanti, Fatimah Mohamed Arshad, and Kusairi Mohd Noh (2017). *System Dynamics*. Singapore: Springer Singapore, 2017.

Baldwin, Andrew and David Bordoli (2014). *A Handbook for Construction Planning and Scheduling*. Ed. by Andrew Baldwin and David Bordoli. Chichester, UK: John Wiley & Sons, Ltd, 2014.

Banks, Jerry and John S. Cason (1984). *Discrete-Event System Simulation*. Englewood Cliffs, NJ: Prentice-Hall, 1984.

Banzhaf, Wolfgang (1998). *Genetic programming : an introduction*. on the automatic evolution of computer programs and its applications. San Francisco, CA: Morgan Kaufmann, 1998.

Barton, Nick R. (2000). *TBM Tunnelling in Jointed and Faulted Rock*. Rotterdam: Balkema, 2000.

Bättig, Daniel (2015). *Angewandte Datenanalyse. Der Bayes'sche Weg*. German. 2nd ed. Berlin, Germany: Springer Spektrum 2015, 2015.

Baumgarten, Bernd (1996). *Petri-Netze: Grundlagen und Anwendungen*. German. Heidelberg: Spektrum Akademischer Verlag, 1996.

BGL (2015). *Baugeräteliste 2015. Technisch-wirtschaftilche Baumaschinendaten*. German. Ed. by Hauptverband der Deutschen Bauindustrie e.V. 1st ed. Gütersloh: Bauverlag, 2015.

Bowman, Adrian W. and Adelchi Azzalini (1997). *Applied Smoothing Techniques for Data Analysis*. The Kernel Approach with S-Plus Illustrations. Oxford: Clarendon Applied Mathematics Research eXpress, 1997.

Box, George E. P., Gwilym M. Jenkins, and Gregory C. Reinsel (2008). *Time Series Analysis*. 4th ed. Hoboken, NJ: Wiley, 2008.

Braeley, Richard A. and Stewart C. Myers (2003). *Principles of Corporate Finance*. 7th ed. New York, NY: McGraw-Hill, 2003.

Brameier, Makrus and Wolfgang Banzhaf (2007). *Linear Genetic Programming*. 2007: Springer, 2007.

Caroll, Lewis (1865). *Alice's Adventures in Wonderland*. Reprint by CreateSpace Independent Publishing Platform, Scotts Valley, CA, 2020. London, UK: Macmillan, 1865.

Choi, Byoung Kyu and Donghun Kang (2013). *Modeling and Simulation of Discrete-Event Systems*. 1st ed. Hoboken, NY: John Wiley & Sons, 2013.

Conforti, Michele, Gérard Cornuéjols, and Giacomo Zambelli (2014). *Integer Programming*. Cham, Switzerland: Springer International Publishing, 2014.

Darwin, Charles (1859). *On the Origin of Species by Means of Natural Selection, or the Preservation of Favoured Races in the Struggle for Life*. London, UK: John Murray, 1859. 612 pp.

Deb, Kalyanmoy (2009). *Multi-Objective optimization using evolutionary algorithms*. Chichester, UK: John Wiley & Sons, 2009. 544 pp.

Deming, William Edwards (1986). *Out of the crisis*. 2nd ed. Cambridge, MA: Cambridge University Press, 1986.

Diestel, Reinhard (2017). *Graph Theory*. 5th ed. Berlin: Springer, 2017.

Domschke, Wolfgang et al. (2015). *Einführung in Operations Research*. German. 9th ed. Lehrbuch. Berlin: Springer, 2015.

Dori, Dov (2016a). *Model-Based Systems Engineering with OPM and SysML*. New Yokr, NY: Springer New York, 2016.

Dörn, Sebastian (2017). *Programmieren für Ingenieure und Naturwissenschaftler. Algorithmen und Programmiertechniken*. German. 1st ed. Berlin: Springer, 2017.

Dunn, William L. and J. Kenneth Shultis (2012). *Exploring Monte Carlo Methods*. 1st ed. Amsterdam, The Netherlands: Elsevier, 2012.

Eilebrecht, Karl and Starke Gernot (2013). *Patterns kompakt. Entwurfsmuster für effektive Software-Entwicklung*. German. 4th ed. Berlin: Springer Vieweg, 2013.

Epstein, Joshua M. and Robert L. Axtell (1996). *Growing Artificial Societies*. Camebridge, MA: MIT Press Ltd, 1996. 224 pp.

Fletcher, R. (2000). *Practical Methods of Optimization*. Chichester, UK: John Wiley & Sons, Ltd, 2000.

Floudas, Christodoulos A. (1995). *Nonlinear Mixed-integer Optimization. Fundamentals and Applications*. New York, NY: Oxford University Press, 1995.

Fogel, Lawrence J., A. J. Owens, and M. J. Walsh (1966). *Artificial Intelligence Through Simulated Evolution*. New York, NJ: John Wiley & Sons, 1966.

Forbes, Catherine et al. (2011). *Statistical Distributions*. 4th ed. Hoboken, NJ: John Wiley, 2011.

Forbes, Lincoln H. and Syed M. Ahmed (2011). *Modern Construction. Lean Project Delivery and Integrated Pratices*. Ed. by Adedeji B. Badiru. 1st ed. Industrial Innovation. Boca Raton, FL: CRC press, 2011.

Forrester, Jay W. (1961). *Industrial Dynamics*. Cambridge, MA: MIT Press, 1961.

Gilbert, Nigel and Klaus G. Troitzsch (2005). *Simulation for the Social Scientist*. 2nd ed. Berkshire, UK: Open University Press, 2005.

Girmscheid, Gerhard and Thorsten A. Busch (2008). *Projektrisikomanagement in der Bauwirtschaft*. German. 1st ed. Berlin, Germany: Bauwerk, 2008.

Goh, Chi-Keong and Kay Chen Tan (2009). *Evolutionary Multi-objective Optimization in Uncertain Environments*. Berline, Germany: Springer Berlin Heidelberg, 2009.

Goldberg, David E. (1989). *Genetic algorithms in search, optimization, and machine learning*. Reading, MA: Addison-Wesley, 1989.

Goll, Joachim (2014). *Architektur- und Entwurfsmuster der Softwaretechnik*. Mit lauffähigen Beispielen in Java. German. 2nd ed. Wiesbaden: Springer, 2014.

Hagan, Martin T. et al. (1996). *Neural Network Design*. 2nd ed. Boston, MA: PWS Publishing, 1996.

Haidar, Ali D. (2016). *Construction Program Management – Decision Making and Optimization Techniques*. 1st ed. Cham, Swizerland: Springer, 2016.

Halpin, D. W. and R. Woodhead (1976). *Design of construction and process operations*. New York, NY: John Wiley & Sons, 1976.

Halpin, Daniel W. and L. S. Riggs (1992). *Planning and analysis of construction operations*. New York, NY: Wiley Wiley-Interscience, 1992.

Haupt, Randy L. and Sue E. Haupt (2004). *Practical Genetic Algorithms*. 2nd ed. Hoboken, NJ: Wiley-Interscience, 2004.

Hemingway, Ernest (1952). *The Old Man and the Sea*. New York, NY: Charles Scribner's Sons, 1952.

Hemphill, Gary B. (2012). *Practical Tunnel Construction*. Hoboken, NJ: John Wiley & Sons, Inc., 2012.

Herbert, Frank (1990). *Dune*. 25th Anniversary Edition. New York, NY: Penguin, 1990. 896 pp.

Hertie School Of Governance (2016). *The Governance Report 2016 (Hertie Governance Report)*. 1st ed. Berlin: Oxford University Press, 2016.

Holland, John H. (1992). *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence (Complex Adaptive Systems)*. Bradford Book edition. Detroit, MI: MIT Press, 1992.

Holt, Jon and Simon Perry (2008). *SysML for Systems Engineering*. London, UK: Institution of Engineering and Technology, 2008.

Johnson, Norman L., Samuel Kotz, and N. Balakrishnan (1995). *Continuous Univariate Distributions*. Hoboken, NJ: Wiley-Interscience, 1995.

Kaplan, Robert S. and David P. Norton (2004). *Strategy Maps. Converting Intangible Assets into Tangible Outcomes*. 1st ed. Boston, MA: Harvard Business School Press, 2004.

Karatkevich, Andrei (2007). *Dynamic Analysis of Petri Net-Based Discrete Systems*. German. 1st ed. Lecture Notes in Control and Information Scieence 356. Berlin, Germany: Springer Berlin Heidelberg, 2007.

Kienitz, Jörg and Daniel Wetterau (2012). *Financial Modelling. Theory, Implementation and Practice (with Matlab source)*. Chichester, UK: Wiley, 2012.

Kim, Phil (2017). *MATLAB Deep Learning*. New York, NY: Springer Science+Business, 2017.

Köster, Dieter (2007). *Marketing und Prozessgestaltung am Baumarkt*. German. 1st ed. Wiesbaden: Deutscher Universitäts-Verlag, 2007.

Krumke, Sven Oliver and Hartmut Noltemeier (2005). *Graphentheoretische Konzepte und Algorithmen*. German. Ed. by Bernd Becker et al. 1st ed. Leitfäden der Informatik. Wiesbaden: Teubner, 2005.

Law, Averill M. (2007). *Simulation Modeling & Analysis*. 4th ed. New York, NJ: McGraw-Hill, 2007.

Leimböck, Egon, Ulf Rüdiger Klaus, and Oliver Hölkermann (2015). *Baukalkulation und Projektcontrolling. unter Berücksichtigung der KLR Bau und der VOB*. German. 13th ed. Wiesbaden: Springer & Vieweg, 2015.

Lindfield, George and John Penny (2012). *Numerical Methods: Using MATLAB*. 1st ed. Oxford, UK: Academic Press, 2012.

Livshin, Igor (2019). *Artificial Neural Networks with Java*. New York, NY: Springer Science+Business, 2019.

Lödding, Herman (2016). *Verfahren der Fertigungssteuerung*. German. 3rd ed. Berlin, Germany: Springer Berlin Heidelberg, 2016.

Lynne Hamill, Nigel Gilbert (2016). *Agent-Based Modelling in Economics*. Chichester, UK: John Wiley & Sons Inc, 2016. 256 pp.

Mandic, Danilo P. and Jonathon A. Chambers (2001). *Recurrent Neural Networks for Prediction*. Chicheter, UK: John Wiley & Sons, 2001. 308 pp.

Michelucci, Umberto (2019). *Advanced Applied Deep Learning*. New York, NY: Apress, 2019.

Monahan, John F. (2011). *Numerical Methods of Statistics*. 2nd ed. New York, NY: Cambridge University Press, 2011.

Muro, Mark et al. (2017). *Digitalization and the American Workforce*. Washington, DC: Brookings Metropolitan Policy Programm, 2017.

Noosten, Dirk (2013). *Netzplantechnik. Grundlagen und Anwendungen im Bauprojektmanagement*. German. 1st ed. Wiesbaden: Springer Science + Business Media, 2013.

Nyamsi, Eric A. (2020). *IT-Lösungen auf Basis von SysML und UML*. Anwendungsentwicklung mit Eclipse UML Designer und Eclipse Papyrus. German. Wiesbaden, Germany: Springer Fachmedien Wiesbaden, 2020.

O'Brien, James J. (1993). *CPM in Construction Management*. 4th ed. New York, NY: McGraw-Hill, 1993.

Ōno, Taiichi (2013). *Das Toyota-Produktionssystem*. German. 3rd ed. (Original Japanese language version published 1978 by Diamond Inc., Tokyo, Japan). Frankfurt/Main, Germany: Campus, 2013.

Pahl, Peter Jan and Rudolf Damrath (2000). *Mathematische Grundlagen der Ingenieurinformatik*. German. 1st ed. Berlin: Springer, 2000.

Pai, G. A. Vijayalakshmi (2018). *Metaheuristics for Portfolio Optimization. An introduction using MATLAB*. Ed. by Nicolas Monmarché and Patrick Siarry. 1st ed. Vol. 11. Metaheuristics Set. London, UK: Wiley, 2018.

Patan, Krzysztof (2008). *Artificial Neural Networks for the Modelling and Fault Diagnosis of Technical Processes*. Berlin, Germany: Springer Berlin Heidelberg, 2008.

– (2019). *Robust and Fault-Tolerant Control*. Cham, Switzerland: Springer International Publishing, 2019.

Pritsker, Alan B. (1984). *Introduction to simulation and SLAM II*. 2nd ed. New York, NY: Halsted Press, John Wiley & Sons, 1984.

– (1989). *SLAM II network models for decision support*. Engelwood Cliffs, NJ: Prentice Hall, 1989.

Rasmussen, Carl Edward and Christopher K. I. Williams (2005). *Gaussian Processes for Machine Learning*. Cambridge, MA: MIT Press, 2005.

Rechenberg, Ingo (1973). *Evolutionstrategie: Optimierung Technisher Systeme nach Prinzipien des Biologischen Evolution*. German. Stuttgart, Germany: problemata frommann-holboog, 1973.

Reisig, Wolfgang (2010). *Petrinetze*. German. Wiesbaden, Germany: Vieweg Teubner Verlag, 2010.

Robert, Christian P. and George Casella (2004). *Monte Carlo Statistical Methods*. 2nd ed. New York, NY: Springer, 2004.

Runkler, Thomas A. (2015). *Data Mining. Modelle und Algorithmen intelligenter Datenanalyse*. German. Ed. by Wolfgang Bibel, Rudolf Kruse, and Bernhard Nebel. 2nd ed. Computational Intelligence. Lehrbuch. Wiesbaden, Germany: Springer Vieweg, 2015.

Runzheimer, Bodo, Thomas Cleff, and Wolfgang Schäfer (2005). *Operations Research 1. Lineare Planungsrechnung und Netzplantechnik*. German. 8th ed. Wiesbaden: Gabler Verlag, 2005.

Schwefel, Hans-Paul (1981). *Numerical optimization of computer models*. Chichester, UK: Wiley, 1981.

Singh, Jagman (1993). *Heavy Construction. Planning, Equipment and Methods*. 1st ed. Rotterdam, Netherlands: A.A. Balkema, 1993.

Solnon, Christine (2013). *Ant Colony Optimization and Constraint Programming*. Ed. by Christine Solnon and Narendra Jussien. London, UK: John Wiley & Sons, Inc., 2013.

Thomas, Monika, ed. (2019). *Vergabe- und Vertragshandbuch für die Baumaßnahmen des Bundes*. German. (Ausgabe 2017 - Stand 2019). Berlin, Germany: Abteilung "Bauwesen, Bauwirtschaft und Bundesbauten" im Bundesministerium für Umwelt, Naturschutz, Bau und Reaktorsicherheit, 2019.

Turau, Volker (2009). *Algorithmische Graphentheorie*. German. 3rd ed. München: Oldenbourg, 2009.

Werbos, Paul John (1994). *The roots of backpropagation: from ordered derivatives to neural networks and political forecasting*. New York, NY: Wiley, 1994.

Weygandt, Jerry J., Donald E. Kieso, and Paul D. Kimmel (2008). *Accounting Principles*. 7th ed. Hoboken, NJ: John Wiley & Sons, Inc., 2008.

Wittke, Walter (2014). *Rock Mechanics Based on an Anisotropic Jointed Rock Model (AJRM)*. Wiley-VCH Verlag GmbH, 2014.

Womack, James P. and Daniel T. Jones (2013). *Lean Thinking. Ballast abwerfen, Unternehmensgewinne steigern*. German. 3rd ed. (Originally published in 1996 by Simon & Schuster, New York, NY). Frankfurt am Main, Germany: Campus, 2013.

Womack, James P., Daniel T. Jones, and Daniel Ross (2007). *The machine that changed the world. How lean production revolutionized the global car wars*. New Ed. (Originally published in 1990 by Rawson Asociates, New York, NY). London, UK: Simon & Schuster, 2007.

Yang, Xin-She (2014). *Nature-Inspired Optimization Algorithms*. 1st ed. Amsterdam, Netherlands: Elsevier, 2014.

Zimmermann, Jürgen, Christoph Stark, and Julia Rieck (2006). *Projektplanung. Modelle, Methoden, Management*. German. 1st ed. Berlin: Springer, 2006.

Thesis

Ashton, Andrew R. (1989). “Computer aided analysis and design of mine transportation systems”. PhD thesis. University of Nottingham, 1989.

Bruland, Amund (2000). “Hard Rock Tunnel Boring”. Advance Rate and Cutter Wear. Vol. 3. PhD thesis. Trondheim, Norway: Norwegian University of Science and Technology, 2000.

Chang, D. Y.-M. (1986). “RESQUE: A Resource Based Simulation System for Construction Process Planning”. PhD thesis. University of Michigan, 1986.

Conrads, Alena (2020). “Maintenance of cutting tools in mechanised tunnelling”. PhD thesis. Bochum, Germany: Ruhr-Universität Bochum, 2020.

De Jong, Kenneth Alan (1975). “An analysis of the behavior of a class of genetic adaptive systems”. Published as Technical Report No: 185. PhD thesis. University of Michigan, 1975.

Demmler, Markus (2009). “Risikomanagement im internationalen Tunnelbau unter Anwendung der Vertragsform FIDIC-Red-Book -”. German. PhD thesis. Kaiserslautern: Technische Universität Darmstadt, 2009.

Dori, Gergö (2016b). “Simulation-based methods for float time determination and schedule optimization for construction projects”. Dissertation. München, Germany: Technische Universität München, 2016.

Fischer, Philipp (2016). “Concoluted Networks to Relate Images”. PhD thesis. Albert-Ludwigs-Univrsität Freiburg im Breisgau, 2016.

Franz, Volker (1989). “Planung und Steuerung komplexer Bauprozesse durch Simulation mit modifizierten höheren Petri-Netzen”. German. PhD thesis. Universität Kassel, 1989.

Freiboth, Axel (2006). “Ermittlung der Entschädigung bei Bauablaufstörungen”. German. Schriftreihe des Institut für Bauwirtschaft und Baubetrieb, Heft 43. PhD thesis. Braunschweig, Germany: Fakultät Architektur, Bauingenieurwesen und Umweltwissenschaften, Technische Universität Varolo-Wilhelmina zu Braunschweig, 2006.

Ioannou, Photios G. (1984). “The economic value of geologic exploration as a risk reduction strategy”. PhD thesis. Department of Civil Engineering, Massachusetts Institute of Technology, 1984.

Kallen, Maarten-Jan (2007). “Markov processes for maintenance optimization of civil infra-structure in the Netherlands”. PhD thesis. Delft, Netherlands: Delft University of Technology, 2007.

Kinzler, Steffen (2011). “Zur Parameteridentifikation, Entwurfs- und Strukturoptimierung in der Geotechnik mittels numerischer Verfahren”. German. Erschienen in: Veröffentlichungen des Instituts für Geotechnik und Baubetrieb 23. PhD thesis. Hamburg: Hamburg Techn. Univ. Hamburg-Harburg, Inst. für Geotechnik und Baubetrieb, 2011.

Liu, Liang-Yun (1991). “COOPS: Construction object-oriented process simulation system”. PhD thesis. Ann Arbort, MI: University of Michigan, 1991.

Martínez, Julio César (1996). “STROBOSCOPE. State and Resource Based Simulation of Construction Processes”. PhD thesis. Arbor, MI: University of Michigan, 1996.

Odeh, Abdalla Mohammad (1992). “CIPROS: Knowledge-Based Construction Integrated Project and Process Planning Simulation System”. PhD thesis. Jordan University of Science and Technology, 1992.

Petri, Carl Adam (1962). “Kommunikation mit Automaten”. German. PhD thesis. Technische Hochschule Darmstadt, 1962.

Rabetge, Christian (1990). “Ansätze einer expliziten Berücksichtigung der Datenunschärfe beim Zeitaspekt in der Betriebswirtschaftslehre”. German. PhD thesis. Universität Göttingen, 1990.

Rahm, Tobias (2016). “Simulation-Based Evaluation of Disturbances of Production and Logistic Processes in Mechanized Tunneling Operations”. PhD thesis. Bochum, Department of Civil and Environmental Engineering, Ruhr-Universiät Bochum, 2016.

Rostami, Jamal (1997). “Development of a force estimation model for rock fragmenttion with disc cutters through theoretical modeling and physical measurement of crushed zone pressure”. PhD thesis. Golden, CO: Colorado School of Mines, 1997.

Ruwanpura, Janaka Y. (2001). “Special purpose simulation for tunnel construction operations”. PhD thesis. University of Alberta, 2001.

Weigl, Werner (1993). “Leistungsprognosen beim Schildvortrieb durch Simulation”. German. PhD thesis. Technische Universität München, 1993.

Werbos, Paul John (1974). “Beyond Regression: New Tools for Predition and Analysis in the Behavioral Sciences”. PhD thesis. Harvard Univesity, 1974.

Yagiz, Saffet (2002). “Development of rock fracture and brittleness indices to quantify the effects of rock mass features and toughness in the CSM model basic penetration for hard rock tunneling machines. PhD Dissertation, T-5605, Colorado School of Mines, USA”. English. T-5605. PhD thesis. Golden, CO: Colorado School of Mines, 2002.

Reports

Amstad, Christian and Kalman Kovári (2001). *Untertagbau in quellfähigem Fels*. German. Tech. rep. (Forschungsauftrag 52/94, 5408.01). Zurich, Switzerland: Institut für Geotechnik, Professur für Untertagbau, Eidgenössische Technische Hochschule Zürich, 2001. 238 pp.

Backhaus, Jan Onne (2018b). *Übungsvorlesung Bauprojektmanagement, Teil 6: Netzplantechnik*. German. Presentation. Institut für Geotechnik und Baubetrieb, Technische Universität Hamburg, 2018.

Bennett, Robert D. (1981). *Tunnel Cost-Estimating Methods*. Tech. rep. Vicksburg, MO: Geotechnical Laboratory, U. S. Army Engineering Waterways Experiment Station, Vicksburg, MO, 1981. 247 pp.

BGH (2019a). *Urteil des VII. Zivilsenats vom 21.11.2019 - VII ZR 10/19 -*. German. Leitsatzentscheidung. (guiding principle decision). Bundesgerichtshof, 2019.

– (2019b). *Urteil des VII. Zivilsenats vom 8.8.2019 - VII ZR 34/18 -*. German. Leitsatzentscheidung mit Berichtigungsbeschluss. (guiding principle decision with rectifying order). Bundesgerichtshof, 2019.

BMVI (2015a). *Reformkommission Bau von Großprojekten. Komplexität beherrschen - kostengerecht, termintreu und effizient*. German. Final Report. Berlin, Germany: Bundesministerium für Verkehr und digitale Infrastruktur, 2015.

– (2015b). *Stufenplan Digitales Planen und Bauen. Einführung moderner, IT-gestützter Prozesse und Technologien bei Planung, Bau und Betrieb von Bauwerken*. German.

Progress Report. Berlin, Germany: Bundesministerium für Verkehr und digitale Infrastruktur, 2015.

BMVI (2017). *Umsetzung des Stufenplans Digitales Planen und Bauen. Erster Fortschrittsbericht*. German. Progress Report. Berlin, Germany: Bundesministerium für Verkehr und digitale Infrastruktur, 2017.

Destatis (2019). *Volkswirtschaftliche Gesamtrechnung. Inlandsproduktberechnung Lange Reihen ab 1970*. German. Report. Fachserie 18 Reihe 1.5. Wiesbaden: Statistisches Bundesamt, 2019.

Dozzi, S. P. and S. M. AbouRizk (1993). *Productivity in construction*. Tech. rep. NRCC-37001. Ottawa, Canada: Institute for Research on Construction, National Research Council, Ottawa, 1993. 54 pp.

Erichsen, Claus (2017). *Fernbahn 1-gleisig Achse 251 und 258 Injektionen Anhydrit Querschnittstypen*. German. Construction Plan. Weinheim, Germany: Prof. Dr.-Ing. W. Wittke Beratende Ingenieure für Grundbau und Felsbau GmbH, 2017.

GLCI (2019). *Lean Construction Begriffe und Methoden*. German. Tech. rep. Karlsruhe, Germany: German Lean Construction Institute, 2019, p. 113.

Halpin, D. W. (1990). *MicroCYCLONE. Version 2.7 user's manual*. Tech. rep. Division of Construction Engineering and Management, Purdue University, IN, 1990.

Halpin, Daniel W. (1983). *Resource analysis using microcomputers*. Research rep. Grant No. CEE -8208271. Georgia Institute of Technology, 1983.

Hibbard, R. R. and Pietrzak (1972). *Computer simulation of hard rock tunneling*. Volume I – Analysis. Tech. rep. CR-2-190, AD-763 563, prepared for Advanced Research Project Agency Washington, D. C. General Research Corporation, 1972.

Ioannou, Photios G. (1989). *UM-CYCLONE, Discrete Event Simulation System, Reference Manual*. Tech. rep. Version March 19, 1990. UMCEE Report No. 89-11. Department of Civil Engineering, University of Michigan, 1989.

Kalk, Anthony (1980). *Insight: Interactive Simulation of Construction Operations Using Graphical Techniques*. Tech. rep. Technical Report 238, National Science Foundation (U.S.) PB81 235863. Stanford University, Department of Civil Engineering, 1980.

Koskela, Lauri (1992). *Application of the new production philosophy to construction*. Technical Report 72. Stanford, CA: Center for integrated facility engineering (CIFE), Stanford University, 1992.

Moavenzadeh, Fred et al. (1974). *Tunnel Cost Model: A stochastic simulation model of hard rock tunneling. Volume 1. Summary Report*. Tech. rep. R74-22. Cambridge, MA: Department of Civil engineering, MIT, Cambridge, MA, 1974.

NEDO (1988). *Faster building for commerce*. Tech. rep. National Economic Development Office, London, UK, 1988.

Nelson, P. P., A. A. Yousof, and P. E. Laughton (1994). *Tunnel Boring Machine project data. Bases and construction simulation. Geotechnical Engineering*. Tech. rep. Report GR94-4. Geotechnical Engineering Center, Department Civil Engineering, The University of Texas at Austin, 1994.

Ozdemir, Levent, R. J. Miller, and F. D. Wang (1978). *Mechanical tunnel boring prediction and machine design*. Tech. rep. NSF APR73-07776-A03. Golden, CO: Colorado School of Mines, 1978.

Petri, Carl Adam (1966). *Communication with automata*. Tech. rep. No. RADC-TR-65-377. Griffiss Air Force Base, New York, 1966.

Pritsker, Alan B. (1966). *GERT: Graphical Evaluation and Review Technique*. Memorandum. RM-4973-NASA, Contract No. NASr-21. Santa Monica, CA: National Aeronautics and Space Administration, 1966.

Reading (2016). *IT Training: Microsoft Access 2013™*. An Essential Guide (Level 1). Manual. University of Reading, 2016.

Rumelhart, David E., Geoffrey E. Hinton, and Ronald J. Williams (1985). *Learning internal representations by error propagation*. Tech. rep. ICS Report 8506. Institute of Cognitive Science, University of California, San Diego, 1985.

Sadri, Kambiz et al. (2013). *Betriebliche Prozesssimulation für maschinelle Tunnelvortriebe SFB 837-C3*. German. Presentation. Lehrstuhl für Tunnelbau, Leitungsbau und Baubetrieb, Ruhr Universität Bochum, 2013.

SFB (2015). *SFB 837 Brochure 2015*. Interaction Modeling in Mechanized Tunneling. Tech. rep. Bochum, Germany: Ruhr University Bochum, 2015.

– (2018). *SFB 837 Brochure 2018*. Interaction Modeling in Mechanized Tunneling. Tech. rep. Bochum, Germany: Ruhr University Bochum, 2018.

– (2019). *SFB 837 Brochure 2019*. Interaction Modeling in Mechanized Tunneling. Tech. rep. Bochum, Germany: Ruhr University Bochum, 2019.

Stiepelmann, Heiko, Peter Kraus, and Heinricht Weitz (2017). *Bauwirtschaft im Zahlenbild*. Ausgabe 2017. Haupterverband der Deutschen Bauindustrie e.V., 2017.

Wheby, F. T. and E. M. Cikanek (1973). *A computer program for estimating costs of tunneling*. Research rep. FRA-ORD&D 74-16, prepared for Department of Transportation, Federal Railroad Administration, Washington, D. C. Harza Engineering Company, 1973.

Conference Articles

AbouRizk, Simaan and Yasser Mohamed (2002). "Optimal construction project planning". In: *Proceedings of the 2002 Winter Simulation Conference (WSC)*. WSC02: Winter Simulation Conference 2002 (San Diego, CA). Ed. by E. Yücesan et al. IEEE, 2002, pp. 1704–1708.

AbouRizk, Simaan et al. (2010). "Developing complex distributed simulation for industrial plant construction using High Level Architecture". In: *Proceedings of the 2010 Winter Simulation Conference (WSC)*. WSC09: Winter Simulation Conference 2009 (Baltimore, MD). Ed. by B. Johansson et al. IEEE, 2010, pp. 3177–3188.

AbouRizk, Simaan M. and Stephen Hague (2009). "An overview of the COSYE environment for construction simulation". In: *Proceedings of the 2009 Winter Simulation Conference (WSC)*. 2009 Winter Simulation Conference (WSC) (Austin, TX). Ed. by M. D. Rossetti et al. IEEE, 2009, pp. 2624–2634.

AbouRizk, Simaan M. et al. (1999). "Special purpose simulation template for utility tunnel construction". In: *Proceedings of the I999 Winter Simulation Conference (WSC). Simulation - A Bridge to the Future*. WSC99: Winter Simulation Conference 1999 (Phoenix, AZ, Dec. 5, 1999). Ed. by P. A. Farrington, D. T. Nembhard H. B. Sturrick, and G. W. Evans. IEEE, 1999, pp. 948–955.

Agrama, Fatma A. (2015). “Versatile multi-objective genetic optimization for non-identical multi-storey building projects”. In: *2015 International Conference on Industrial Engineering and Operations Management (IEOM)*. IEEE, 2015.

Al-Allaf, Omaima N. Ahmad (2012). “Cascade-Forward vs. Function Fitting Neural Network for Improving Image Quality and Learning Time in Image Compression System”. In: *Proceedings of the World Congress on Engineering 2012 Vol II* (London, UK). 2012.

Backhaus, Jan Onne (2018a). “Digitale Optimierung der Bauplanung”. German. In: *Tagungsband zum 29. BBB - Assistententreffen – Fachkongress der wissenschaftlichen Mitarbeiter der Bereiche Bauwirtschaft | Baubetrieb | Bauverfahrenstechnik*. Beiträge zum 29. BBB - Assistententreffen vom 06. bis 08. Juni 2018 in Braunschweig (Braunschweig, Germany). Zentrum für Bau- und Infrastrukturmanagement, TU Braunschweig, 2018, pp. 23–33.

– (2019). “Vorhersage des Injektionsvolumens einer Tunnelbaustelle unter Verwendung von Markov Ketten”. German. In: *Tagungsband zum 30. BBB-Assistententreffen 2019 in Karlsruhe*. Fachkongress der wissenschaftlichen Mitarbeiter Bauwirtschaft, Baubetrieb, Bauverfahrenstechnik (Karlsruhe, Germany). Ed. by Shervin Haghsheno, Kunibert Lennerts, and Sascha Gentes. Karlsruhe, Germany: KIT Scientific Publishing, 2019, pp. 6–22.

Backhaus, Jan Onne and Jürgen Grabe (2018). “Numerische basierte Prozessanalyse”. German. In: *Workshop Digitale Infrastruktur und Geotechnik* (Hamburg, Germany). Ed. by Jürgen Grabe. Vol. 42. Veröffentlichungen des Institutes Geotechnik und Baubetrieb. Institut für Geotechnik und Baubetrieb, Technische Universität Hamburg. 2018, pp. 247–264.

Balbo, Gianfanco and Giovanni Chiola (1989). “Stochastic petri net simulation”. In: *Proceedings of the 1989 Winter Simulation Conference (WSC)*. WSC ’89: Proceedings of the 1989 Winter Simulation Conference (New York, NY). Ed. by E. A. MacNair, K. J. Musselman, and J. Heidelberger. ACM Press, 1989, pp. 266–276.

Balci, Osman (1988). “The implementation of four conceptual frameworks for simulation modeling in high-level languages”. In: *Proceedings of the 1988 Winter Simulation Conference (WSC)*. WSC ’88: Proceedings of the 1988 Winter Simulation Conference (Washington, DC). Ed. by M. Abrams, P. Haigh, and J. Comfort. ACM Press, 1988, pp. 287–295.

Bamford, W. E. (1984). “Rock test indices are being successfully correlated with tunnel boring machine performance”. In: *State of the art in underground development and construction*. 5th Australian Tunnelling Conference (Sydney, Australia). Barton, A. C. T. : Institution of Engineers, 1984, pp. 218–221.

Al-Battaineh, Hussien T. et al. (2006). “Productivity Simulation during the Planning Phase of the Glencoe Tunnel in Calgary, Canada: A Case Study”. In: *Proceedings of the 2006 Winter Simulation Conference (WSC)* (Miami, FL). 2006, pp. 2087–2092.

Berkhahn, V. et al. (2005). “Process Modelling in Civil Engineering based on Hierarchical Petri Nets”. In: *Proceedings of the 22th International Conference on Information Technology for Construction CIB-W78* (Dresden, Germany). Ed. by Raimar Scherer, Peter Katranuschkov, and Sven-Eric Schapke. Institute for Construction Informatics, Technische Universität Dresden, 2005.

Chang, David Y. and Md. Saiful Hoque (1989). “A Knowledge-Based Simulation System for Construction Process Planning”. In: *Proceedings of the 6th International Symposium*

on Automation and Robotics in Construction (ISARC). International Association for Automation and Robotics in Construction (IAARC), 1989, pp. 348–355.

Dang, Trung Thanh, Britta Schoesser, and Markus Thewes (2013). "Process simulation of microtunnelling perations for pro-ductivity assessment depending on ground conditions". In: *Proceedings of the Third International Conference on Computational Methods in Tunneling and Subsurface Engineering*. Conference on Computational Methods in Tunneling and Subsurface Engineering (EURO:TUN 2013) (Bochum, Germany). Ed. by Giinther Meschke et al. Ruhr Universität Bochum, 2013, pp. 311–322.

Fernando, S. et al. (2003). "A Review of Simulation Applications for Varying Demands in Tunneling". In: *Construction Research Congress*. ASCE Construction Research Congress 2003 (Honolulu, HI). Ed. by Keith R. Molenaar and Paul S. Chinowsky. ASCE, 2003, pp. 80–90.

Hajjar, Dany and Simaan M. AbouRizk (1996). "Building a special purposes simulation tool for earth moving operations". In: *Proceedings of the 1996 Winter Simulation Conference (WSC)*. WSC96: Winter Simulation Conference 1996 (Coronado, CA). Ed. by J. M. Charnes et al. ACM Press, 1996, pp. 1313–1320.

– (1999). "Simphony: An environment for building special purpose construction simulation tools". In: *Proceedings of the 1999 Winter Simulation Conference (WSC). A bridge to the future*. WSC99: Winter Simulation Conference 1999 (Phoenix, AZ). ACM Press, 1999, pp. 998–1006.

Halpin, Daniel W., Henyi Jen, and Jungwuk Kim (2003). "A construction process simulation web service". In: *Proceedings of the 2003 Winter Simulation Conference (WSC)*. WSC03: Winter Simulation Conference 2003 (New Orleans, LA). Ed. by S. Chick et al. ACM Press, 2003, pp. 1503–1509.

Han, Sangwon, Moonseo Park, and Feniosky Peña-Mora (2005). "Comparative Study of Discrete-Event Simulation and System Dynamics for Construction Process Planning". In: *Construction Research Congress 2005* (San Diego, CA). Ed. by iris D. Tommelein. ASCE, 2005.

Hollermann, Sebastian, Jürgen Melzner, and Hans-Joachim Bargstädt (2012). "Concept for scheduling of bridge construction processes based on distributed simulation". In: *Proceedings of the 19th EG-ICE International Workshop on Intelligent Computing in Engineering 2012 (ICE12)*. 19th EG-ICE International Workshop on Intelligent Computing in Engineering 2012 (ICE12) (Munich, Germany). Ed. by A. Borrmann. 2012.

Horenburg, T., J. Wimmer, and W. A. Günthner (2012). "Resource Allocation in Construction Scheduling based on Multi-Agent Negotiation". In: *Proceedings of the 14th International Conference on Computing in Civil and Building Engineering*. 14th International Conference on Computing in Civil and Building Engineering (ICCCVE 2012) (Moscow, Russia). 2012.

Howell, Gregory A. (1999). "What Is Lean Construction". In: *Proceedings of the 7th Annual Conference of the International Group for Lean Construction*. 7th Annual Conference of the International Group for Lean Construction (Berkeley, CA). 1999, pp. 33–38.

Huang, Rong-Yau, A. M. Grigoriadis, and D. W. Halpin (1994). "Simulation of cable-stayed bridges using DISCO". In: *Proceedings of the 1994 Winter Simulation Conference (WSC)*. WSC '94: Proceedings of the 1994 Winter Simulation Conference (Lake Buena Vista, FL). Ed. by J. D. Tew et al. IEEE, 1994, pp. 1130–1136.

Ioannou, P. G. and V. Likhitruangsilp (2005). “Simulation of Multiple-drift Tunnel Construction with Limited Resources”. In: *Proceedings of the 2005 Winter Simulation Conference (WSC)*. WSC05: Winter Simulation Conference 2005. Ed. by M. E. Kuhl et al. IEEE, 2005, pp. 1483–1491.

Kennedy, James and Russell Eberhart (1995). “Particle Swarm Optimization”. In: *Proceedings of the IEEE International Conference on Neural Networks* (Perth, Australia). IEEE. 1995, pp. 1942–1945.

Kooragamage, Ruvinde et al. (2013). “Using agent-based simulation to manage logistics for earthmoving operations in construction”. In: *Proceedings of CIB World Building Congress*. CIB World Building Congress (Kuala Lumpur, Malaysia). 2013.

Kugler, Martin and Volkhard Franz (2007). “Entwurf eines multiagentenbasierten Referenzmodells für Simulationen im Hochbau”. German. In: *Schriftreihe Bauwirtschaft - III Tagungen und Berichte 4*. IBW Workshop Simulation in der Bauwirtschaft (Kassel, Germany, Sept. 13, 2007). Ed. by Volkhardt Franz. 2007, pp. 69–83.

Kumar, Sumit, Alok Misra, and Raghvendra Singh Tomar (2011). “A modified parallel approach to Single Source Shortest Path Problem for massively dense graphs using CUDA”. In: *2011 2nd International Conference on Computer and Communication Technology (ICCCT-2011)*. IEEE, 2011, pp. 635–639.

Likhitruangsilp, Veerasak and Photios G. Ioannou (2003). “Stochastic Evaluation of Tunneling Performance Using Discrete-Event Simulation”. In: *Construction Research Congress*. ASCE, 2003.

– (2004). “Risk-Sensitive Decision Support System for Tunnel Construction”. In: *Geotechnical Engineering for Transportation Projects*. ASCE, 2004, pp. 1508–1515.

– (2005). “Economic Assessment of Site Exploration Programs Using Stochastic Dynamic Programming”. In: *Construction Research Congress 2005* (San Diego, CA). Ed. by iris D. Tommelein. ASCE, 2005.

Liu, Liang-Yun (1995). “Simulating construction operations of precast-concrete parking structures”. In: *Proceedings of the 1995 Winter Simulation Conference (WSC)*. WSC95: Winter Simulation Conference 1995 (Piscataway, NJ). Ed. by C. Alexopoulos et al. ACM Press, 1995, pp. 1004–1008.

Liu, Liang-Yun and Photios G. Ioannou (1992). “Graphical object-oriented discrete-event simulation system”. In: *Proceedings of the 1992 Winter Simulation Conference (WSC)* (Arlington, VA). Ed. by Robert C. Crain. ACM Press, 1992, pp. 1285–1291.

Lu, H., E. Kim, and M. Gutierrez (2018). “A Markovian Rock Mass Quality Q-based Prediction Model for Tunneling”. In: *52nd US Rock Mechanics / Geomechanics Symposium*. 52nd US Rock Mechanics / Geomechanics Symposium (Seattle, WA). Seattle, WA: American Rock Mechanics Association (ARMA), 2018.

Martínez, Julio César (2001). “EZStrobe-general-purpose simulation system based on activity cycle diagrams”. In: *Proceedings of the 2001 Winter Simulation Conference (WSC)*. WSC01: Winter Simulation Conference 2001 (Arlington, VA). Ed. by Brett A. Peters et al. IEEE, 2001, pp. 1556–1564.

Marzok, Mohamed, Moatassem Abdallah, and Moheeb El-Said (2008). “Tunnel_Sim: Decision support tool for planning tunnel construction using computer simulation”. In: *Proceedings of the 2008 Winter Simulation Conference (WSC)*. WSC08: Winter Simu-

lation Conference 2008 (Miami, FL). Ed. by S. J. Mason et al. IEEE, 2008, pp. 2504–2511.

Mattern, Hannah et al. (2016). "Simulation-Based Analysis of Maintenance Strategies for Mechanized Tunneling Projects". In: *Proceedings of the WTC 2016*. World Tunnel Congress 2016 (San Francisco, CA). 2016, pp. 1555–1566.

Moghani, Elmira et al. (2011). "Analyzing transit tunnel construction strategies using discrete event simulation". German. In: *Proceedings of the 2011 Winter Simulation Conference (WSC)*. WSC11: Winter Simulation Conference 2011 (Phoenix, AZ). Ed. by S. Jain et al. IEEE, 2011, pp. 3500–3510.

Moore, Philip and Hai V. Pham (2012). "Predicting Intelligence Using Hybrid Artificial Neural Networks in Context-Aware Tunneling Systems under Risk and Uncertain Geological Environment". In: *Proceedings of the 6th International Conference on Complex, Intelligent, and Software Intensive Systems*. 6th International Conference on Complex, Intelligent, and Software Intensive Systems (CISIS-2012) (Palermo, Italy). Ed. by Leonard Barolli et al. IEEE, 2012, pp. 989–994.

Oren, Tuncer I. et al. (2000). "Agent-directed simulation – challenges to meet defense and civilian requirements". In: *Proceedings of the 2000 Winter Simulation Conference (WSC)*. WSC '00: Proceedings of the 2000 Winter Simulation Conference (Orlando, FL). Ed. by J. A. Joines et al. IEEE, 2000, pp. 1757–1762.

Rahm, Tobias et al. (2012). "Advancement simulation of tunnel boring machines". In: *Proceedings of the 2012 Winter Simulation Conference (WSC)*. Ed. by C. Laroque et al. IEEE, 2012, pp. 727–738.

Rahm, Tobias et al. (2013). "Uncertainty modeling and simulation of tool wear in mechanized tunneling". In: *Proceedings of the 2013 Winter Simulations Conference (WSC)*. WSC13: Winter Simulation Conference 2013. Simulation: Making Decisions in a Complex World. Ed. by R. Pasupathy et al. IEEE, 2013, pp. 3121–3132.

RazaviAlvi, SeyedReza and Simaan AbouRizk (2014). "An Integrated Simulation Model for Site Layout Planning of Tunneling Projects". In: *Proceedings of the International Conference on Modeling and Applied Simulation*. The 13th International Conference on Modeling and Applied Simulation (MAS2014) (Bordeaux, France). Ed. by A. Bruzzone et al. 2014, pp. 46–53.

Ruwanpura, Janaka Y. and Simaan M. AbouRizk (2001). "Design, development and application of soil transition algorithms for tunneling using special purpose simulation". In: *Proceedings of the 2001 Winter Simulation Conference (WSC)*. WSC01: Winter Simulation Conference 2001 (Arlington, VA). Ed. by Brett A. Peters et al. IEEE, 2001, pp. 1512–1520.

Sabew, Sewerin, Andreas Heinzmann, and Philipp Maroschek (2019a). "Automation and digitization of chemical grouting for sealing rock mass". In: *Proceedings of the 9th Nordic Grouting Symposium 2019 (ngs2019)* (Helsinki, Finland). 2019, p. 9.

– (2019b). "Herausforderungen bei der Ausführung von chemischen Injektionen zum Abdichten von anhydritführendem Gebirge. Aktuelle Entwicklungen in Bohrtechnik und Injektionsverfahren". German. In: *Beiträge zum 34. Christian Veder Kolloquium*. 34. Christian Veder Kolloquium (Graz, Austria, Apr. 25, 2019). Ed. by Roman Marte, Helmut F. Schweiger, and Franz Tschuchnigg. 34 vols. 2. NAWI Graz Geocenter. Graz, Austria: NAWI Graz Geocenter, 2019, pp. 171–184.

Sawhney, Anil (1997). “Petri net based simulations of construction schedules”. In: *Proceedings of the 1997 Winter Simulation conference (WSC)*. WSC97: Winter Simulation Conference 1997 (Atlanta, GA). Ed. by Sigrún Andradóttir et al. IEEE, 1997, pp. 1111–1118.

Sawhney, Anil and Amarneethi Vamadevan (2000). “Petri Net-Based Scheduling of a Bridge Project”. In: *Construction Congress VI*. ASCE, 2000, pp. 107–114.

Scheffer, Markus and Ruben Duhme (2018). “Construction Process Simulation in Tunnel Construction – A Prerequisite for Automation”. In: *Proceedings of the 35th International Symposium on Automation and Robotics in construction (ISARC 2018)*. 35th Internatinal Symposium on Automation and Robotics in construction (ISARC 2018) (Berlin, Germany, July 25, 2018). Ed. by Jochen Teizer and Markus König. Berlin, Germany: The International Association for Automation and Robotics in Construction, 2018.

Scheffer, Markus, Tobias Rahm, and Markus König (2014a). “Simulation-based analysis of surface jobsite logistics in mechanized tunneling”. In: *Computing in Civil and Building Engineering (2014)*. 2014 International Conference on Computing in Civil and Building Engineering (Orlando, FL). 23-25 June. ASCE, 2014, pp. 705–712.

Scheffer, Markus et al. (2014b). “Jobsite Logistics Simulation in Mechanized Tunneling”. In: *Proceedings of the 2014 Winter Simulation Conference (WSC)*. WSC14: Winter Simulation Conference 2014 (Savannah, GA). Ed. by A. Tolk et al. 2014, pp. 1843–1854.

Seitz, Karlotta and Jürgen Grabe (2016). “Optimization of geotechnical structures for states of serviceability and ultimate loads”. In: *Insights and Innovations in Structural Engineering, Mechanics and Computation*. 6th International Conference on Structural Engineering, Mechanics and Computation (SEMC 2016) (Cape Town, Sout Africa). Ed. by A. Zingoni. Taylor & Francis Group, 2016, pp. 2048–2053.

– (2018). “Einsatzmöglichkeiten der multikriteriellen Optimierung im digitalen Bauen.” German. In: *Proceedings of Digitale Infrastruktur und Geotechnik 2018*. Digitale Infrastruktur und Geotechnik 2018 (DIG2018) (Hamburg, Germany). Ed. by Jürgen Grabe. Institut für Geotechnik und Baubetrieb, Technische Universität Hamburg. 2018, pp. 169–181.

Simões, Marcelo and Taehong Kim (2006). “Fuzzy Modeling Approaches for the Prediction of Machine Utilization in Hard Rock Tunnel Boring Machines”. In: *Conference Record of the 2006 IEEE Industry Applications Conference Forty-First IAS Annual Meeting*. IEEE, 2006, pp. 947–954.

Tam, C. M. and Arthur W. T. Leung (2010). “Using construction process simulation to assess productivity of laying water mains in Hong Kong”. In: *Proceedings of the CIB W78 2010*. 27th W78 Conference (Cairo, Egypt). 2010.

Tesfaye, Erimas et al. (2015). “A Simulation of Project Completion Probability Using Different Probability Distribution Functions”. In: *Afro-European Conference for Industrial Advancement*. Ed. by Ajith Abraham, Pavel Krömer, and Vaclav Snasel. Cham: Springer International Publishing, 2015, pp. 133–145.

Tommelein, Iris D., Robert I. Carr, and Abdalla M. Odeh (1994). “Knowledge-based assembly of simulation networks using construction designs, plans, and methods”. In: *Proceedings of the 1994 Winter Simulation Conference (WSC)*. WSC ’94: Proceedings of the 1994 Winter Simulation Conference (Lake Buena Vista, FL). Ed. by J. D. Tew et al. IEEE, 1994, pp. 1145–1152.

Touran, Ali (1992). "Facilitating simulation model development for construction engineers". In: *Proceedings of the 1992 Winter Simulation Conference (WSC)*. WSC92: Winter Simulation Conference 1992 (Arlington, VA). Ed. by Robert C. Crain. ACM Press, 1992, pp. 1278–1284.

Vapnik, Vladimir, Steven E. Golowich, and Alex Smola (1996). "Support Vector Method for Function Approximation, Regression Estimation, and Signal Processing". In: *Proceedings of the 1996 Conference on Advances in Neural Information Processing Systems 9*. Advances in Neural Information Processing Systems 9 (Denver, CO). Ed. by Michael C. Mozer, Michael I. Jordan, and Thomas Petsche. MIT Press, 1996, pp. 281–287.

Wagner, Dorothea and Thomas Willhalm (2007). "Speed-up Techniques for Shortest-Path Computations". In: *STACS 2007*. 24th Annual Symposium on theoretical Aspects of Computer Science (Aachen, Germany, Feb. 22, 2007). Ed. by Wolfgang Thomas and Pascal Weil. 2007, pp. 23–36.

Werner, Michael and Simaan AbouRizk (2015). "Simulation case study: Modelling distinct breakdown events for a Tunnel Boring Machine excavation". In: *2015 Winter Simulation Conference (WSC)*. WSC15: Winter Simulation Conference 2015 (Savannah, GA). Ed. by L. Yilmaz et al. IEEE, 2015, pp. 3234–3245.

Werner, Michael, Wenying Ji, and Simaan AbouRizk (2018). "Improving tunneling simulation using Bayesian updating and hidden M arkov chains". In: *2018 Winter Simulation Conference (WSC)*. WSC18: Winter Simulation Conference 2018 (Gothenburg, Sweden). Ed. by M. Rabe et al. IEEE, 2018, pp. 3930–3939.

Xie, Hua, Siri Fernando, and Simaan AbouRizk (2011). "Integrating realtime project progress input into a construction simulation model". In: *Proceedings of the 2011 Winter Simulation Conference (WSC)*. WSC11: Winter Simulation Conference 2011 (Phoenix, AZ). Ed. by S. Jain et al. IEEE, 2011, pp. 3448–3459.

Zahran, Hany and Khaled Nassar (2013). "Modeling pipeline projects using computer simulation". In: *2013 Winter Simulations Conference (WSC)*. WSC13: Winter Simulation Conference 2013. Simulation: Making Decisions in a Complex World (Washington, D.C. Dec. 8, 2013). Ed. by R. Pasupathy et al. Washington, D. C.: IEEE, 2013, pp. 3269–3281.

Zhang, Limao et al. (2015). "Updating geological conditions using Bayes theorem and Markov chain". In: *Proceedings of the 2015 Winter Simulation Conference (WSC)*. WSC15: Winter Simulation Conference 2015. Ed. by L. Yilmaz et al. IEEE, 2015, pp. 3367–3378.

Zhang, Yang et al. (2010). "3D CAD modeling and visualization of the tunnel construction process in a distributed simulation environment". In: *Proceedings of the 2010 Winter Simulation Conference*. WSC10: Winter Simulation Conference 2010 (Baltimore, MD). Ed. by B. Johansson et al. IEEE, 2010, pp. 3189–3200.

Zhou, Fangyi, Simaan M. AbouRizk, and Siri Fernando (2008). "A simulation template for modeling tunnel shaft construction". In: *Proceedings of the 2008 Winter Simulation Conference (WSC)*. WSC08: Winter Simulation Conference 2008 (Miami, FL). Ed. by S. J. Mason et al. IEEE, 2008, pp. 2455–2461.

Journal Articles

Abdallah, Moatassem and Mohamed Marzouk (2013). "Planning of tunneling projects using computer simulation and fuzzy decision making". In: *Journal of Civil Engineering and Management* 19.4 (2013), pp. 591–607.

Abdollahisharif, J. and E. Bakhtavar (2018). "Using geostatistical simulation to determine optimal grout injection pressure in dam foundation based on geomechanical characteristics". In: *Bulletin of Engineering Geology and the Environment* 78.4 (2018), pp. 2253–2266.

AbouRizk, S. et al. (2014). "Simphony: a next generation simulation modelling environment for the construction domain". In: *Journal of Simulation* 10.3 (2014), pp. 207–215.

AbouRizk, Simaan M., Antonio A. Gonzales-Quevedo, and Daniel W. Halpin (1990). "Applications of Variance Reduction Techniques in Construction Simulation". In: *Computer-Aided Civil and Infrastructure Engineering* 5.4 (1990), pp. 299–306.

Agrama, Fatma Abd El-Mohye (2011). "Linear projects scheduling using spreadsheets features". In: *Alexandria Engineering Journal* 50 (2011), pp. 179–185.

Alanjari, Pejman, SeyedReza RazaviAlavi, and Simaan AbouRizk (2015). "Hybrid Genetic Algorithm-Simulation Optimization Method for Proactively Planning Layout of Material Yard Laydown". In: *Journal of Construction Engineering and Management* 141.10 (2015), p. 06015001.

Alkoc, Ediz and Fuat Erbatur (1998). "Simulation in concreting operations: a comparison of models and resource combinations". In: *Engineering, Construction and Architectural Management* 5.2 (1998), pp. 159–173.

Azadivar, Farhad and John Jian Wang (2000). "Facility layout optimization using simulation and genetic algorithms". In: *International Journal of Production Research* 38.17 (2000), pp. 4369–4383.

Backhaus, Jan Onne (2020). "Bauzeitenvorhersage von Injektionen im Tunnelbau". German. In: *Bautechnik* 97 (2020), pp. 1–11.

– (2021). "Einsatz von neuronalen Netzen zur Vorhersage des Materialvolumens von Injektionsbaustellen im Tunnelbau". German. In: *Bautechnik* 98 (2021).

Backhaus, Jan Onne and Markus Dahm (2020). "Einblick in den Stand der Implementierung von Lean Construction Ansätzen in ausgewählten deutschen Bauunternehmen – Ergebnisse einer qualitativen Studie". German. In: *Bauingenieur* 95.2 (2020), pp. 64–72.

Barron, Andrew R. (1993). "Universal approximation bounds for superpositions of a sigmoidal function". In: *IEEE Transactions on Information Theory* 39.3 (1993), pp. 930–945.

Barton, Nick Ryland (1999). "TBM performance in rock using QTBM". In: *Tunnels & Tunnelling International* September (1999), pp. 30–34.

Al-Bataineh, Mohammed, Simaan AbouRizk, and Holly Parkis (2013). "Using Simulation to Plan Tunnel Construction". In: *Journal of Construction Engineering and Management* 139.5 (2013), pp. 564–571.

Bayes, Thomas (1763). "An Essay towards Solving a Problem in the Doctrine of Chances". In: *Philosophical Transactions* 53 (1763), pp. 370–418.

Bektas, Esra (2013). "Knowledge Sharing Strategies for Large Complex Building Projects." In: *A+BE : Architecture and the Built Environment* 4 (2013), pp. 1–332.

Benardos, A. G. and D. C. Kaliampakos (2004). "Modelling TBM performance with artificial neural networks". In: *Tunnelling and Underground Space Technology* 19.6 (2004), pp. 597–605.

Beran, V. and E. Hromada (2008). "Dynamic simulations in cost and time estimation of the construction process". In: *Acta Polytechnica* 48.1 (2008), pp. 30–35.

Bernstein, Serge N. (1927). "Sur l'Extension du Théoréme Limite du Calcul des Probabilités aux Sommes de quantités dépendentes". French. In: *Mathematische Annalen* 97 (1927), pp. 1–59.

Bruland, Amund (1998). "Prediction Model for Performance and Cost". In: *Norwegian TBM Tunneling* 11 (1998), pp. 29–34.

Bundesanzeiger (2013). "Verordnung über die Honorare für Architekten- und Ingenieurleistungen. (Honorarordnung für Architekten und Ingenieure–HOAI)". German. In: *Bundesgesetzblatt* 1.37 (2013). Ed. by Bundesanzeiger Verlag, pp. 2276–2374.

– (2016). "Bekanntmachung der Vergabe- und Vertragsordnung für Bauleistungen. Teil A (VOB/A) Teil B (VOB/B) vom 7. Januar 2016". German. In: *Bundesanzeiger* (2016). Ed. by Bundesministerium der Justiz und für Verbraucherschutz, pp. 16–71.

– (2020). "Erste Verordnung zur Änderung der Honorarordnung für Architekten und Ingenieure. (Honorarordnung für Architekten und Ingenieure–HOAI)". German. In: *Bundesgesetzblatt* 1.58 (2020). Ed. by Bundesanzeiger Verlag, pp. 2636–2642.

Cao, Feilong, Tingfan Xie, and Zongben Xu (2008). "The estimate for approximation error of neural networks: A constructive approach". In: *Neurocomputing* 71.4-6 (2008), pp. 626–630.

Censor, Yair (1977). "Pareto Optimality in Multiobjective Problems". In: *Applied Mathematics and Optimization* 4 (1977), pp. 41–59.

Chehayeb, Nader N. and Simaan M. AbouRizk (1998). "Simulation-Based Scheduling with Continuous Activity Relationships". In: *Journal of Construction Engineering and Management* 124.2 (1998), pp. 107–115.

Chen, Tianping and Hong Chen (1995). "Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems". In: *IEEE Transactions on Neural Networks* 6.4 (1995), pp. 911–917.

Cheng, F. F. et al. (2011). "A Petri net simulation model for virtual construction of earthmoving operations". In: *Automation in Construction* 20.2 (2011), pp. 181–188.

Cheng, Feifei et al. (2013). "Modeling resource management in the building design process by information constraint Petri nets". In: *Automation in Construction* 29 (2013), pp. 92–99.

Christodoulou, Symeon, Georgios Ellinas, and Pooyan Aslani (2009). "Entropy-based scheduling of resource-constrained construction projects". In: *Automation in Construction* 18.7 (2009), pp. 919–928.

Chua, D. K. H., W. T. Chan, and K. Govindan (1997). "A time-cost trade-of model with resource consideration using genetic algorithm". In: *Civil Engineering Systems* 14.4 (1997), pp. 291–311.

Chui, Charles K. and Xin Li (1992). "Approximation by ridge functions and neural networks with one hidden layer". In: *Journal of Approximation Theory* 70.2 (1992), pp. 131–141.

Chung, Tae Hwan, Yasser Mohamed, and Simaan AbouRizk (2006). "Bayesian Updating Application into Simulation in the North Edmonton Sanitary Trunk Tunnel Project". In: *Journal of Construction Engineering and Management* 132.8 (2006), pp. 882–894.

Clemmens, John P. and Jack H. Willenbrock (1978). "The SCRAPESIM Computer Simulation". In: *Journal of computer simulation* 104.4 (1978), pp. 419–435.

Conrads, Alena et al. (2017). "Assessing maintenance strategies for cutting tool replacements in mechanized tunneling using process simulation". In: *Journal of Simulation* 11.1 (2017), pp. 51–61.

Conrads, Alena et al. (2018a). "Robustness evaluation of cutting tool maintenance planning for soft ground tunneling projects". In: *Underground Space* 3.1 (2018), pp. 72–85.

Cooper, Mary W. (1981). "A Survey of Methods for Pure Nonlinear Integer Programming". In: *Management Science* 27.3 (1981), pp. 353–361.

Cortes, Corinna and Vladimir Vapnik (1995). "Support-Vector Networks". In: *Machine Learning* 20.3 (1995), pp. 273–297.

Costa, Ana Laura, Rita L. Sousa, and Herbert H. Einstein (2018). "Probabilistic 3D alignment optimization of underground transport infrastructure integrating GIS-based subsurface characterization". In: *Tunnelling and Underground Space Technology* 72 (2018), pp. 233–241.

Cybenko, G. (1989). "Approximation by Superpositions of a Sigmoidal Function". In: *Mathematics of Control Signals and System* 2 (1989), pp. 303–314.

Dabbas, Majed A. A. and Daniel W. Halpin (1982). "Integrated Project and Process Management". In: *Journal of the Construction Division* (1982), pp. 361–374.

Dalkey, Norman and Olaf Helmer (1963). "An Experimental Application of the DELPHI Method to the Use of Experts". In: *Management Science* 9.3 (1963), pp. 458–467.

Damci, Atilla, David Arditi, and Gul Polat (2013). "Resource Leveling in Line-of-Balance Scheduling". In: *Computer-Aided Civil and Infrastructure Engineering* 28.9 (2013), pp. 679–692.

Dang, Trung Thanh et al. (2018). "Evaluation of productivities influenced by disturbances and different soil compositions in microtunnelling using process simulation". In: *Tunnelling and Underground Space Technology* 76 (2018), pp. 10–20.

Deep, Kusum and Manoj Thakur (2007a). "A new crossover operator for real coded genetic algorithms". In: *Applied Mathematics and Computation* 188.1 (2007), pp. 895–911.

– (2007b). "A new mutation operator for real coded genetic algorithms". In: *Applied Mathematics and Computation* 193.1 (2007), pp. 211–230.

Deep, Kusum et al. (2009). "A real coded genetic algorithm for solving integer and mixed integer optimization problems". In: *Applied Mathematics and Computation* 212.2 (2009), pp. 505–518.

Devi, A. A. Cindrela and K. Ananthanarayanan (2015). "Development of Resource-Driven Scheduling Model for Mass Housing Construction Projects". In: *International Journal of Engineering and Technology* 7.5 (2015), pp. 419–423.

Dijkstra, Edsger W. (1959). "A Note on Two Problems in Connexion with Graphs". In: *Numerische Matehmatik* 1 (1959), pp. 269–271.

DuBois, Didier and Henri Prade (1989). "Processing fuzzy temporal knowledge". In: *IEEE Transactions on Systems, Man, and Cybernetics* 19.4 (1989), pp. 729–744.

Ebrahimy, Yasser et al. (2011a). “Simphony Supply Chain Simulator: a simulation toolkit to model the supply chain of construction projects”. In: *SIMULATION* 87.8 (2011), pp. 657–667.

Ebrahimy, Yasser et al. (2011b). “Simulation modeling and sensitivity analysis of a tunneling construction project’s supply chain”. In: *Engineering, Construction and Architectural Management* 18.5 (2011), pp. 462–480.

Einstein, H. H. (2004). “Decision Aids for Tunneling: Update”. In: *Transportation Research Record: Journal of the Transportation Research Board* 1892.1 (2004), pp. 199–207.

Einstein, Herbert H. (2001). “The Decision Aids for Tunneling (DAT) - a brief Review”. In: *Magazine of Korean Tunnelling and Underground Space Association* 3.3 (2001), pp. 37–49.

Einstein, Herbert H. et al. (1999). “Decision Aids for Tunneling”. In: *Journal of the Transportation Research Board* 1656.1 (1999), pp. 6–13.

Euler, Leonhard (1741). “Solutio problematis ad geometriam situs pertinentis”. Latin. In: *Commentarii academiae scientiarum Petropolitanae* 8 (1741), pp. 128–140.

Fahihi, Vahid, Kenneth F. Reinschmidt, and Julian H. Kang (2014). “Construction scheduling using Genetic Algorithm based on Building”. In: *Expert systems with Applications* 41.16 (16 2014), pp. 7565–7578.

Feng, Chung-Wei, Liang Liu, and Scott A. Burns (1997). “Using Genetic Algorithms to Solve Construction Time-Cost Trade-Off Problems”. In: *Journal of Computing in Civil Engineering* 11.3 (1997), pp. 184–189.

– (2000). “Stochastic construction time-cost trade-off analysis”. In: *Journal of computing in civil engineering* (2000), pp. 117–126.

Ferrari, Silvia and Robert F. Stengel (2005). “Smooth Function Approximation Using Neural Networks”. In: *IEEE Transactions on Neural Networks* 16.1 (2005), pp. 24–38.

Folkard, Simon and Philip Tucker (2003). “Shift work, safety and productivity”. In: *Occupational Medicine* 53.2 (2003), pp. 95–101.

Forrester, Jay W. (1958). “Industrial Dynamics: A Major Breakthrough for Decision Makers”. In: *Harvard Business Review* 36.4 (July-August 1958), pp. 37–66.

– (2003). “Dynamic models of economic systems and industrial organizations”. In: *System Dynamics Review* 19.4 (2003), pp. 329–345.

Funahashi, Ken-Ichi (1989). “On the approximate realization of continuous mappings by neural networks”. In: *Neural Networks* 2.3 (1989), pp. 183–192.

Ganjalipour, Kamal and Masoud Esmailzadeh (2019). “Numerical modeling for evaluating grout curtain depth and providing a new approach for calculating the efficiency based on installation position of piezometers around sealing system”. In: *Modeling Earth Systems and Environment* 5.4 (2019), pp. 1587–1605.

Geem, Zong Woo, Joong Hoon Kim, and G. V. Loganathan (2001). “A New Heuristic Optimization Algorithm: Harmony Search”. In: *SIMULATION* 76.2 (2001), pp. 60–68.

Gehring, K. (1995). “Leistungs- und Verschleissprognosen im maschinelle”. In: *Tunnelbau Felsbau* 13.6 (1995), pp. 439–448.

Ghasemi, Ebrahim, Saffet Yagiz, and Mohammad Ataei (2013). “Predicting penetration rate of hard rock tunnel boring machine using fuzzy logic”. In: *Bulletin of Engineering Geology and the Environment* 73.1 (2013), pp. 23–35.

Gilkinson, Norman and Brian Dangerfield (2013). “Some results from a system dynamics model of construction sector competitiveness”. In: *Mathematical and Computer Modelling* 57.9-10 (2013), pp. 2032–2043.

Gneiting, Tilmann (2011). “Making and Evaluating Point Forecasts”. In: *Journal of the American Statistical Association* 106.494 (2011), pp. 746–762.

Gong, Q. M. and J. Zhao (2009). “Development of a rock mass characteristics model for TBM penetration rate prediction”. In: *International Journal of Rock Mechanics and Mining Sciences* 46.1 (2009), pp. 8–18.

Gonzalez-Quevedo, Antonio A. et al. (1993). “Comparison of Two Simulation Methodologies in Construction”. In: *Journal of Construction Engineering and Management* 119.3 (1993), pp. 573–589.

Grima, M. Alvarez, P. A. Bruines, and P. N. W. Verhoef (2000). “Modeling tunnel boring machine performance by neuro-fuzzy methods”. In: *Tunnelling and Underground Space Technology* 15.3 (2000), pp. 259–269.

Grossmann, Ignacio E. (2002). “Review of Nonlinear Mixed-Integer and DisjunctiveProgramming Techniques”. In: *Optimization and Engineering* 3 (2002), pp. 277–252.

Haas, Christoph and Herbert H. Einstein (2002). “Updating the Decision Aids for Tunneling”. In: *Journal of Construction Engineering and Management* 128.1 (2002), pp. 40–48.

Hahm, Nahmwoo and Bum Il Hong (2004). “An approximation by neural networkswith a fixed weight”. In: *Computers & Mathematics with Applications* 47.12 (2004), pp. 1897–1903.

Hajjar, Dany, Simaan AbouRizk, and Jianfei Xu (1998). “Construction site dewatering analysis using a special purpose simulation-based framework”. In: *Canadian Journal of Civil Engineering* 25.5 (1998), pp. 819–828.

Hajjar, Dany and Simaan M. AbouRizk (1998). “Modeling and Analysis of Aggregate Production Operations”. In: *Journal of Construction Engineering and Management* 124.5 (1998), pp. 390–401.

Halpin, D. W. (1977). “CYCLONE Method for modeling job site processes”. In: *Journal of the Construction Division* 103.3 (1977), pp. 489–499.

Hartmann, Veronika, Tom Lahmer, and Kay Smarsly (2017). “Project scheduling under uncertainty and resource constraints”. In: *PAMM* 17.1 (2017), pp. 839–840.

Hassan, Marwa M. and Stan Gruber (2008). “Simulation of Concrete Paving Operations on Interstate-74”. In: *Journal of Construction Engineering and Management* 134.1 (2008), pp. 2–9.

Hassanpour, J., J. Rostami, and J. Zhao (2011). “A new hard rock TBM performance prediction model for project planning”. In: *Tunnelling and Underground Space Technology* 26.5 (2011), pp. 595–603.

Hastings, W. K. (1970). “Monte Carlo sampling methods using Markov chains and their applications”. In: *Biometrika* 57.1 (1970), pp. 97–109.

Hegazy, Tarek (1999a). “Optimization of construction time-cost trade-off analysis using genetic algorithms”. In: *Canadian Journal of Civil Engineering* 26.6 (1999), pp. 685–697.

– (1999b). “Optimization of Resource Allocation and Leveling Using Genetic Algorithms”. In: *Journal of Construction Engineering and Management* 125.3 (1999), pp. 167–175.

Herroelen, Willy and Roel Leus (2005). "Project scheduling under uncertainty: Survey and research potentials". In: *European Journal of Operational Research* 165.2 (2005), pp. 289–306.

Holmboe, Michael, Susanna Wold, and Torbjörn Petterson (2011). "Effects of the injection grout Silica sol on bentonite". In: *Physics and Chemistry of the Earth, Parts A/B/C* 36.17-18 (2011), pp. 1580–1589.

Holzer, Martin, Frank Schulz, and Dorothea Wagner (2008). "Engineering multilevel overlay graphs for shortest-path queries". In: *Journal of Experimental Algorithmics* 13 (2008), pp. 1–26.

Hornik, Kurt, Maxwell Stinchcombe, and Halbert White (1989). "Multilayer feedforward networks are universal approximators". In: *Neural Networks* 2.5 (1989), pp. 359–366.

– (1990). "Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks". In: *Neural Networks* 3.5 (1990), pp. 551–560.

Huang, Rong-Yau and Daniel W. Halpin (1994). "Visual Construction Operation Simulation: The DISCO Approach". In: *Computer-Aided Civil and Infrastructure Engineering* 9.3 (1994), pp. 175–184.

Javad, Gholamnejad and Tayarani Narges (2010). "Application of artificial neural networks to the prediction of tunnel boring machine penetration rate". In: *Mining Science and Technology (China)* 20.5 (2010), pp. 727–733.

Jiang, Zhongming, Dongping Fang, and Mengchun Zhang (2015). "Understanding the Causation of Construction Workers' Unsafe Behaviors Based on System Dynamics Modeling". In: *Journal of Management in Engineering* 31.6 (2015), p. 04014099.

Jorne, Fernando, Fernando M. A. Henriques, and Luis G. Baltazar (2014). "Injection capacity of hydraulic lime grouts in different porous media". In: *Materials and Structures* 48.7 (2014), pp. 2211–2233.

Kalantary, F., H. Ardalan, and N. Nariman-Zadeh (2009). "An investigation on the Su–NSPT correlation using GMDH type neural networks and genetic algorithms". In: *Engineering Geology* 104.1-2 (2009), pp. 144–155.

Kamat, Vineet R. and Julio C. Martinez (2003). "Validating Complex Construction Simulation Models Using 3D Visualization". In: *Systems Analysis Modelling Simulation* 43.4 (2003), pp. 455–467.

Kaplan, Robert S. and David P. Norton (1992). "The Balanced Scorecard". In: *Harvard Business Review* Reprint 92105 (January-February 1992), pp. 70–79.

– (1996). "Linking the Balanced Scorecard to Strategy". In: *California Management Review* 39.1 (1996), pp. 53–79.

– (2000). "Having Trouble with Your Strategy? Then Map IT." In: *Harvard Business Review* (September-October 2000), pp. 167–176.

Karam, Karim S., Jad S. Karam, and Herbert H. Einstein (2007). "Decision Analysis Applied to Tunnel Exploration Planning. I: Principles and Case Study". In: *Journal of Construction Engineering and Management* 133.5 (2007), pp. 344–353.

Kavanagh, Donncha P. (1985). "Siren: A Repetitive Construction Simulation Model". In: *Journal of Construction Engineering and Management* 111.3 (1985), pp. 308–323.

Kim, Hongjo et al. (2018a). "Analyzing context and productivity of tunnel earthmoving processes using imaging and simulation". In: *Automation in Construction* 92 (2018), pp. 188–198.

Kim, Jung In, Martin Fischer, and Calvin Kam (2018b). "Generation and evaluation of excavation schedules for hard rock tunnels in preconstruction and construction". In: *Automation in Construction* 96 (2018), pp. 378–397.

Kim, K. J. and G. Edward Gibson (2003). "Interactive simulation modeling for heavy construction operations". In: *Automation in Construction* 12.1 (2003), pp. 97–109.

Kim, Kyoungmin and Kyong Ju Kim (2010). "Multi-agent-based simulation system for construction operations with congested flows". In: *Automation in Construction* 19.7 (2010), pp. 867–874.

Kim, Kyungki, John Walewski, and Yong K. Cho (2016). "Multiobjective Construction Schedule Optimization Using Modified Niched Pareto Genetic Algorithm". In: *Journal of Management in Engineering* 32.2 (2016), p. 04015038.

Kirkpatrick, S., C. D. Gelatt, and M. P. Vecchi (1983). "Optimization by Simulated Annealing". In: *Science* 220.4598 (1983), pp. 671–680.

Knotts, Gary, Moshe Dror, and Bruce C. Hartman (2000). "Agent-based project scheduling". In: *IIE Transactions* 32.5 (2000), pp. 387–401.

König, Markus et al. (2014). "Prozesssimulation von maschinellen Tunnelvortrieben. Verfügbarkeitsanalysen der Leistungsprozesse unter Berücksichtigung von Stillständen". German. In: *Bauingenieur* 89.11 (2014), pp. 467–477.

Koo, Bonsang, Martin Fischer, and John Kunz (2007). "A formal identification and resequencing process for developing sequencing alternatives in CPM schedules". In: *Automation in Construction* 17.1 (2007), pp. 75–89.

Koopialipoor, Mohammadreza et al. (2018). "Predicting tunnel boring machine performance through a new model based on the group method of data handling". In: *Bulletin of Engineering Geology and the Environment* 78.5 (2018), pp. 3799–3813.

Krumbein, W. C. and Michael F. Dacey (1969). "Markov Chains and Embedded Markov Chains in Geology". In: *Mathematical Geology* 1 (1 1969), pp. 79–96.

Lau, Sze-Chun, Ming Lu, and Samuel T. Ariaratnam (2010). "Applying radial basis function neural networks to estimate next-cycle production rates in tunnelling construction". In: *Tunnelling and Underground Space Technology* 25.4 (2010), pp. 357–365.

Laufer, A. and R. L. Tucker (1987). "Is construction project planning really doing its job? A critical examination of focus, role and process". In: *Construction Management and Economics* 5.3 (1987), pp. 243–266.

Lee, Sangyoub, Daniel W. Halpin, and Hoon Chang (2006). "Quantifying effects of accidents by fuzzy-logic- and simulation-based analysis". In: *Canadian Journal of Civil Engineering* 33.3 (2006), pp. 219–226.

Leon, Hany et al. (2018). "System Dynamics Approach for Forecasting Performance of Construction Projects". In: *Journal of Management in Engineering* 34.1 (2018), p. 04017049.

Leshno, Moshe et al. (1993). "Multilayer feedforward networks with a nonpolynomial activation function can approximate any function". In: *Neural Networks* 6.6 (1993), pp. 861–867.

Leu, Sou-Sen and Tri Joko Wahyu Adi (2011). "Probabilistic prediction of tunnel geology using a Hybrid Neural-HMM". In: *Engineering Applications of Artificial Intelligence* 24.4 (2011), pp. 658–665.

Levitt, Raymond E., Nabil A. Kartam, and John C. Kunz (1988). "Artificial Intelligence Techniques for Generating Construction Project Plans". In: *Journal of Construction Engineering and Management* 114.3 (1988), pp. 329–343.

Lienhart, Christoph et al. (2019). "Feuerbach and Bad Cannstatt Tunnels – special requirements of the inner-city Location and difficult geotechnical Conditions". German. In: *Geomechanics and Tunnelling* 12.5 (2019), pp. 456–466.

Little, John D. C. (1961). "A Proof for the Queuing Formula: $L = \lambda W$". In: *Operations Research* 9.3 (1961), pp. 383–387.

Liu, Donghai, Yunqing Zhou, and Kai Jiao (2010). "TBM construction process simulation and performance optimization". In: *Transactions of Tianjin University* 16.3 (2010), pp. 194–202.

Liu, Donghai et al. (2015). "Schedule Risk Analysis for TBM Tunneling Based on Adaptive CYCLONE Simulation in a Geologic Uncertainty–Aware Context". In: *Journal of Computing in Civil Engineering* 29.6 (2015), p. 04014103.

Liu, Mingqiang et al. (2019). "System dynamics modeling for construction management reserach: Critical review and future trends". In: *Journal of Civil Engineering and Management* 25.8 (2019), pp. 730–741.

Lluch, Jose and Daniel W. Halpin (1982). "Construction operations and microcomputers". In: *Journal of Construction Engineering and Management* 108.1 (1982), pp. 129–145.

Love, P. E. D. et al. (2002). "Using systems dynamics to better understand change and rework in construction project management systems". In: *International Journal of Project Management* 20.6 (2002), pp. 425–436.

Low Sui, PhengChong Joo Chuan (2001). "Just-in-time management in precast concrete construction: a survey of the readiness of main contractors in Singapore". In: *Integrated Manufaturing Systems* 12 (6 2001), pp. 416–429.

Lu, Ming (2003). "Simplified Discrete-Event Simulation Approach for Construction Simulation". In: *Journal of Construction Engineering and Management* 129.5 (2003), pp. 537–546.

Lu, Ming and Hoi-Ching Lam (2008). "Critical Path Scheduling under Resource Calendar Constraints". In: *Journal of Construction Engineering and Management* 134.1 (2008), pp. 25–31.

Lu, Ming and Heng Li (2003). "Resource-Activity Critical-Path Method for Construction Planning". In: *Journal of Construction Engineering and Management* 129.4 (2003), pp. 412–420.

Lutz, James D., Daniel W. Halpin, and James R. Wilson (1994). "Simulation of Learning Development in Repetitive Construction". In: *Journal of Construction Engineering and Management* 120.4 (1994), pp. 753–773.

MacCrimmon, Kenneth R. and Charles A. Ryavec (1964). "An Analytical Study of the PERT Assumptions". In: *Operations Research* 12.1 (1964), pp. 16–37.

Mahdevari, Satar et al. (2014). "A support vector regression model for predicting tunnel boring machine penetration rates". In: *International Journal of Rock Mechanics and Mining Sciences* 72 (2014), pp. 214–229.

Mahmoodzadeh, Arsalan et al. (2020). "Decision-making in tunneling using artificial intelligence tools". In: *Tunnelling and Underground Space Technology* 103 (2020), p. 103514.

Maiorov, Vitaly and Ron S. Meir (1998). "Approximation bounds for smooth functions in $C(\mathbb{R}^d)$ by neural and mixture networks". In: *IEEE Transactions on Neural Networks* 9.5 (1998), pp. 969–978.

Maji, V. B. and G. V. Theja (2017). "A New Performance Prediction Model for Rock TBMs". In: *Indian Geotechnical Journal* 47.3 (2017), pp. 364–372.

Makovoz, Y. (1998). "Uniform Approximation by Neural Networks". In: *Journal of Approximation Theory* 95.2 (1998), pp. 215–228.

Marchand, Hugues et al. (2002). "Cutting planes in integer and mixed integer programming". In: *Discrete Applied Mathematics* 123.1-3 (2002), pp. 397–446.

Markov, A. A. (1906). "Rasprostranenie zakona bol'shih chisel na velichiny, zavisyaschie drug ot druga [Extension of the Law of Large Numbers to Dependent Quantitites]". Russian. In: *Izv. Fiz. Matem.* 15.94 (1906). Obsch. Kazan Univ., (2nd Ser.), pp. 135–156.

Martinez, Julio C. and Photios G. Ioannou (1999). "General-Purpose Systems for Effective Construction Simulation". In: *Journal of Construction Engineering and Management* 125.4 (1999), pp. 265–276.

Maruvanchery, Varun, Shao Zhe, and Tiong Lee Kong Robert (2020). "Early construction cost and time risk assessment and evaluation of large-scale underground cavern construction projects in Singapore". In: *Underground Space* 5.1 (2020), pp. 53–70.

Maryani, Anny, Sritomo Wignjosoebroto, and Sri Gunani Partiwi (2015). "A System Dynamics Approach for Modeling Construction Accidents". In: *Procedia Manufacturing* 4 (2015), pp. 392–401.

Marzouk, Mohamed, Moatassem Abdallah, and Moheeb El-Said (2010). "Modeling Microtunneling Projects using Computer Simulation". In: *Journal of Construction Engineering and Management* 136.6 (2010), pp. 670–682.

McCahill, Dennis F. and Leonhard E. Bernold (1993). "Resource-Oriented Modeling and Simulation in Construction". In: *Journal of Construction Engineering and Management* 119.3 (1993), pp. 590–606.

Metropolis, Nicholas et al. (1953). "Equation of State Calculations by Fast Computing Machines". In: *The Journal of Chemical Physics* 21.6 (1953), pp. 1087–1092.

Mhasjar, H. N. and C. A. Micchelli (1992). "Approximation by superposition of a sigmoidal function". In: *Advances in applied mathematics* 13 (1992), pp. 350–373.

Mhaskar, H. N. and C. A. Micchelli (1995). "Degree of Approximation by Neural and Translation Networks with a Single Hidden Layer". In: *Advances in Applied Mathematics* 16.2 (1995), pp. 151–183.

Mikulakova, Eva et al. (2010). "Knowledge-based schedule generation and evaluation". In: *Advanced Engineering Informatics* 24.4 (2010), pp. 389–403.

Min, S. Y. et al. (2003). "Application of Decision Aids for Tunneling (DAT) to a Drill & Blast Tunnel". In: *KSCE Journal of Civil Engineering* 7.5 (2003), pp. 619–628.

Min, S. Y. et al. (2008). "Design and construction of a road tunnel in Korea including application of the Decision Aids for Tunneling – A case study". In: *Tunnelling and Underground Space Technology* 23.2 (2008), pp. 91–102.

Min, Sangyoon and Herbert H. Einstein (2016). "Resource scheduling and planning for tunneling with a new resource model of the Decision Aids for Tunneling (DAT)". In: *Tunnelling and Underground Space Technology* 51 (2016), pp. 212–225.

Mirjalili, Seyedali, Seyed Mohammad Mirjalili, and Andrew Lewis (2014). "Grey Wolf Optimizer". In: *Advances in Engineering Software* 69 (2014), pp. 46–61.

Mohamed, Sherif and Thanwadee Chinda (2011). "System dynamics modelling of construction safety culture". In: *Engineering, Construction and Architectural Management* 18.3 (2011), pp. 266–281.

Mohamed, Y. and S. M. AbouRizk (2005). "Framework for Building Intelligent Simulation Models of Construction Operations". In: *Journal of Computing in Civil Engineering* 19.3 (2005), pp. 277–291.

Møller, Martin Fodslette (1993). "A scaled conjugate gradient algorithm for fast supervised learning". In: *Neural Networks* 6.4 (1993), pp. 525–533.

Monghasemi, Shahryar et al. (2015). "A novel multi criteria decision making model for optimizing time–cost–quality trade-off problems in construction projects". In: *Expert Systems with Applications* 42.6 (2015), pp. 3089–3104.

Moon, HyounSeok et al. (2014). "Development of a schedule-workspace interference management system simultaneously considering the overlap level of parallel schedules and workspaces". In: *Automation in Construction* 39 (2014), pp. 93–105.

Moret, Yvonne and Herbert H. Einstein (2016). "Construction Cost and Duration Uncertainty Model: Application to High-Speed Rail Line Project". In: *Journal of Construction Engineering and Management* 142.10 (2016), p. 05016010.

Naghadehi, Masoud Zare et al. (2016). "The probabilistic time and cost risk analysis of a challenging part of an urban tunneling project". In: *Tunnelling and Underground Space Technology* 58 (2016), pp. 11–29.

Najafi, Ali and Robert Tiong (2015). "Simulation as A Tool for Productivity Modeling of Precast Concrete Installation". In: *Current World Environment* 10 (Special Issue 2015), pp. 177–183.

Nasir, Muhammad Kamran and Bonaventura H. W. Hadikusumo (2019). "System Dynamics Model of Contractual Relationships between Owner and Contractor in Construction Projects". In: *Journal of Management in Engineering* 35.1 (2019), p. 04018052.

Nasirzadeh, Farnad and Pouya Nojedehi (2013). "Dynamic modeling of labor productivity in construction projects". In: *International Journal of Project Management* 31.6 (2013), pp. 903–911.

Nasirzadeh, Farnad et al. (2008). "Integrating system dynamics and fuzzy logic modelling for construction risk management". In: *Construction Management and Economics* 26.11 (2008), pp. 1197–1212.

Nido, Alberto A., Craig J. Knies, and Dulcy M. Abraham (1999). "Role of operation simulation in the analysis and improvement of microtunnelling projects". In: *Tunnelling and Underground Space Technology* 14 (1999), pp. 1–19.

Nübel, Konrad et al. (2016). "Produktionsplanung und Produktionssteuerung im Spezialtiefbau Prozessorientierter Ablauf von Bauprojekten im Spezialteifbau". German. In: *Bauingenieur. Die richtungsweisende Zeitschrift um Bauinenieurwesen*. VDI-Bautechnik (2015/2016 2016), pp. 101–107.

Oloufa, Amr A. (1993). "Modeling Operational Activities in Object-Oriented Simulation". In: *Journal of Computing in Civil Engineering* 7.1 (1993), pp. 94–106.

– (1994). "A User-Oriented Approach to Construction Simulation of Buildings". In: *Computer-Aided Civil and Infrastructure Engineering* 9.6 (1994), pp. 425–433.

Oloufa, Amr A., Masaaki Ikeda, and Tang-Hung Nguyen (1998). "Resource-based simulation libraries for construction". In: *Automation in Construction* 7.4 (1998), pp. 315–326.

Oraee, Kazem and Bahram Salehi (2011). "Assessing prediction models of advance rate in tunnel boring machines—a case study in Iran". In: *Arabian Journal of Geosciences* 6.2 (2011), pp. 481–489.

Padovnik, Andreja and Violeta Bokan-Bosiljkov (2020). "Effect of Ultralight Filler on the Properties of Hydrated Lime Injection Grout for the Consolidation of Detached Historic Decorative Plasters". In: *Materials* 13.15 (2020), p. 3360.

Paez, Omar et al. (2005). "Moving from Lean Manufacturing to Lean Construction: Toward a Common Sociotechnological Framework". In: *Human Factors and Ergonomics in Manufacturing* 15.2 (2005), pp. 223–245.

Pal, Mahesh and Surinder Deswal (2010). "Modelling pile capacity using Gaussian process regression". In: *Computers and Geotechnics* 37.7-8 (2010), pp. 942–947.

Paltrinieri, Erika et al. (2016). "Probabilistic simulations of TBM tunnelling in highly fractured and faulted rocks". In: *Tunnelling and Underground Space Technology* 57 (2016), pp. 183–194.

Paulson, Boyd C., Weng Tat Chan, and Charlie C. Koo (1987). "Construction Operations Simulation by Microcomputer". In: *Journal of Construction Engineering and Management* 113.2 (1987), pp. 302–314.

Paulson, Boyd C. et al. (1983). "Simulation and Analysis of Construction Operations". In: *Journal of Technical Topics in Civil Engineering* (1983), pp. 89–104.

Petroutsatou, K. et al. (2012). "Early Cost Estimating of Road Tunnel Construction Using Neural Networks". In: *Journal of Construction Engineering and Management* 138.6 (2012), pp. 679–687.

Pilcher, R. and I. Flood (1984). "The use of simulation models in construction". In: *Proceedings of the Institution of Civil Engineers* 76.3 (1984), pp. 635–652.

Poshdar, Mani et al. (2018). "A multi-objective probabilistic-based method to determine optimum allocation of time buffer in construction schedules". In: *Automation in Construction* 92 (2018), pp. 46–58.

Prinzhorn, Henrik, Philip Rochow, and Peter Nyhuis (2017). "Grundlagen einer Methode für eine qualitätsorientierte Belegungsplanung". German. In: *Logistics Journal* 1 (2017), pp. 1–7.

Pucker, T. and J. Grabe (2011). "Structural optimization in geotechnical engineering: basics and application". German. In: *Acta Geotechnica* 6.1 (2011), pp. 41–49.

Rahm, Tobias et al. (2015). "Evaluation of Disturbances in Mechanized Tunneling Using Process Simulation". In: *Computer-Aided Civil and Infrastructure Engineering* 31.3 (2015), pp. 176–192.

– (2016). "Evaluation of Disturbances in Mechanized Tunneling Using Process Simulation". In: *Computer-Aided Civil and Inrastructure Engineering* 31 (2016), pp. 176–192.

RazaviAlvi, SeyedReza and Simaan AbouRizk (2015). "A hybrid simulation approach for quantitatively analyzing the impact of facility size on construction projects". In: *Automation in Construction* 60 (2015), pp. 39–48.

Restuccia, L. et al. (2017). "An investigation of the beneficial effects of adding carbon nanotubes to standard injection grout". In: *Fatigue & Fracture of Engineering Materials & Structures* 41.1 (2017), pp. 119–128.

Riggs, Leland S. (1987). "Graphic Input to Cyclone". In: *Journal of Computing in Civil Engineering* 1.3 (1987), pp. 175–182.

Ritter, S., H. H. Einstein, and R. Galler (2013). "Planning the handling of tunnel excavation material – A process of decision making under uncertainty". In: *Tunnelling and Underground Space Technology* 33 (2013), pp. 193–201.

Rogalska, Magdalena, Wojciech Bożejko, and Zdzisław Hejducki (2008). "Time/cost optimization using hybrid evolutionary algorithm in construction project scheduling". In: *Automation in Construction* 18.1 (2008), pp. 24–31.

Roxborough, Frank F. and Huw R. Phillips (1975). "Rock excavation by disc cutter". In: *International Journal of Rock Mechanics and Mining Sciences & Geomechanics Abstracts* 12.12 (1975), pp. 361–366.

Rumelhart, David E., Geoffrey E. Hinton, and Ronald J. Williams (1986). "Learning representations by back-propagating errors". In: *Nature* 323.6088 (1986), pp. 533–536.

Ruwanpura, Janaka Y., Simaan M. AbouRizk, and M. Allouche (2004). "Analytical methods to reduce uncertainty in tunnel construction projects". In: *Canadian Journal of Civil Engineering* 31.2 (2004), pp. 345–360.

Ruwanpura, Janaka Y. et al. (2001). "Special purpose simulation templates for tunnel construction operations". In: *Canadian Journal of Civil Engineering* 28.2 (2001), pp. 222–237.

Savitzky, Abraham. and M. J. E. Golay (1964). "Smoothing and Differentiation of Data by Simplified Least Squares Procedures." In: *Analytical Chemistry* 36.8 (1964), pp. 1627–1639.

Sawhney, Anil and Simaan M. AbouRizk (1996). "Computerized Tool for Hierarchical Simulation Modeling". In: *Journal of Computing in Civil Engineering* 10.2 (1996), pp. 115–124.

Scheffer, Markus et al. (2016). "Simulation-Based Analysis of Integrated Production and Jobsite Logistics in Mechanized Tunneling". In: *Journal of Computing in Civil Engineering* 30.5 (2016), pp. C4016002.1–14.

Seitz, K.-F., J. Grabe, and T. Köhne (2018). "A three-dimensional topology optimization model for tooth-root morphology". In: *Computer Methods in Biomechanics and Biomedical Engineering* 21.2 (2018), pp. 177–185.

Seitz, Karlotta-Franziska, Tim Pucker, and Jürgen Grabe (2016). "Topologieoptimierung in der Geotechnik: Anwendung auf Gründungsstrukturen und Validierung". German. In: *Geotechnik* 39.1 (2016), pp. 18–28.

Senior, Bolivar A. and Daniel W. Halpin (1998). "Simplified Simulation System for Construction Projects". In: *Journal of Construction Engineering and Management* 124.1 (1998), pp. 72–81.

Senouci, Ahmed and Hassan R. Al-Derham (2008). "Genetic algorithm-based multi-objective model for scheduling of linear construction projects". In: *Advances in Engineering Software* 39.12 (2008), pp. 1023–1028.

Senouci, Ahmed B. and Neil N. Eldin (2004). "Use of Genetic Algorithms in Resource Scheduling of Construction Projects." In: *Journal of Construction Engineering & Management* 130.6 (2004), pp. 869–877.

Senouci, Ahmed B. and Saleh A. Mubarak (2016). "Multiobjective optimization model for scheduling of construction projects under extreme weather". In: *Journal of Civil Engineering and Management* 22.3 (2016), pp. 373–381.

Shahin, A. et al. (2013). "Simulation modeling of weather-sensitive tunnelling construction activities subject to cold weather". In: *Canadian Journal of Civil Engineering* 41.1 (2013), pp. 48–55.

Shao, Zhe et al. (2015). "DAT modifications and its application in large-scale cavern construction". In: *Tunnelling and Underground Space Technology* 50 (2015), pp. 209–217.

Shi, Jingsheng and Simaan M. AbouRizk (1997). "Resource-Based Modeling for Construction Simulation". In: *Journal of Construction Engineering and Management* 123.1 (1997), pp. 26–33.

Shi, Jonathan Jingsheng (1999). "Activity-Based Construction (ABC) Modeling and Simulation Method". In: *Journal of Construction Engineering and Management* 125.5 (1999), pp. 354–360.

Shields, Rob (2012). "Cultural Topology: The Seven Bridges of Königsburg, 1736". In: *Theory, Culture & Society* 29.4-5 (2012), pp. 43–57.

Sinfield, Joseph V. and Herbert H. Einstein (1996). "Evaluation of tunneling technology using the "decision aids for tunneling"". In: *Tunnelling and Underground Space Technology* 11.4 (1996), pp. 491–504.

Snyder, J. Riley et al. (2017). "Agent-based modelling and construction – reconstructing antiquity's largest infrastructure project". In: *Construction Management and Economics* 36.6 (2017), pp. 313–327.

Sonmez, Rifat and Önder Halis Bettemir (2012). "A hybrid genetic algorithm for the discrete time–cost trade-off problem". In: *Expert Systems with Applications* 39.13 (2012), pp. 11428–11434.

Špačková, Olga, Jiří Šejnoha, and Daniel Straub (2013). "Probabilistic assessment of tunnel construction performance based on data". In: *Tunnelling and Underground Space Technology* 37 (2013), pp. 62–78.

Špačková, Olga and Daniel Straub (2012). "Dynamic Bayesian Network for Probabilistic Modeling of Tunnel Excavation Processes". In: *Computer-Aided Civil and Infrastructure Engineering* 28.1 (2012), pp. 1–21.

Squazzoni, Flaminio (2010). "The Impact of Agent-Based Models in the Social Sciences after 15 Years of Incursion". In: *History of Economic Ideas* (2010), pp. 195–233.

Storn, Rainer and Kenneth Price (1997). "Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces". In: *Journal of Global Optimization* 11.4 (1997), pp. 341–359.

Suzuki, Shin (1998). "Constructive function-approximation by three-layer artificial neural networks". In: *Neural Networks* 11.6 (1998), pp. 1049–1058.

Szczesny, Kamil and Markus König (2015). "Reactive scheduling based on actual logistics data by applying simulation-based optimization". In: *Visualization in Engineering* 3.1 (2015), pp. 1–16.

Tanner, John P. (1985). "The learning curve". In: *Production Engineering* 32 (1985), pp. 72–78.

Thang, Hong, Jonathan J. Shi, and Chi-Ming Tam (2002). "Application of simulation related techniques to construction operations". In: *Engineering, Construction and Architectural Management* 9.5/6 (2002), pp. 433–445.

Touran, Ali and Toshiyuki Asai (1987). "Simulation of Tunneling Operations". In: *Journal of Construction Engineering and Management* 113.4 (1987), pp. 554–568.

Vargas, Juan P., Jair C. Koppe, and Sebastián Pérez (2014). "Monte Carlo simulation as a tool for tunneling planning". In: *Tunnelling and Underground Space Technology* 40 (2014), pp. 203–209.

Vargas, Juan P. et al. (2015). "Planning Tunnel Construction Using Markov Chain Monte Carlo (MCMC)". In: *Mathematical Problems in Engineering* 2015 (2015). Article ID 797953, p. 8.

Wakefield, Ron R. and Glenn A. Sears (1997). "Petri Nets for Simulation and Modeling of Construction Systems". In: *Journal of Construction Engineering and Management* 123.2 (1997), pp. 105–112.

Wales, R. J. and S. M. AbouRizk (1996). "An integrated simulation model for construction". In: *Simulation Practice and Theory 3* 3.6 (1996), pp. 401–420.

Wallerang, Elmar (2015). "Der Bau soll von der produzierenden Industrie lernen". German. In: *VDI Nachrichten* 1-2 (2015), p. 27.

Walski, T. M. (1980). "Water Resources Planning Cost Estimating Tools". In: *IFAC Proceedings Volumes* 13.3 (1980), pp. 513–519.

Wan, Jiangping and Yaqiong Liu (2014). "A System Dynamics Model for Risk Analysis during Project Construction Process". In: *Open Journal of Social Sciences* 02.06 (2014), pp. 451–454.

Werra, D. de and A. Hertz (1989). "Tabu search techniques". In: *Operations-Research-Spektrum* 11.3 (1989), pp. 131–141.

Willis, R.J. (1985). "Critical path analysis and resource constrained project scheduling — Theory and practice". In: *European Journal of Operational Research* 21.2 (1985), pp. 149–155.

Wittke, Walter et al. (2017). "AJRM as basis for design and construction of more than 70 km of tunnels of the Railway Project Stuttgart-Ulm". In: *Geomechanics and Tunnelling* 10.2 (2017), pp. 204–211.

Wu, Lingzi, Wenying Ji, and Simaan M. AbouRizk (2020). "Bayesian Inference with Markov Chain Monte Carlo – Based Numerical Approach for Input Model Updating". In: *Journal of Computing in Civil Engineering* 34.1 (2020).

Xu, Xiaoxiao and Patrick X. W. Zou (2020). "System dynamics analytical modeling approach for construction project management research: A critical review and future directions". In: *Frontiers of Engineering Management* (2020).

Yagiz, S. et al. (2009). "Application of two non-linear prediction tools to the estimation of tunnel boring machine performance". In: *Engineering Applications of Artificial Intelligence* 22.4-5 (2009), pp. 808–814.

Yagiz, Saffet and Halil Karahan (2011). "Prediction of hard rock TBM penetration rate using particle swarm optimization". In: *International Journal of Rock Mechanics and Mining Sciences* 48.3 (2011), pp. 427–433.

Yagiz, Saffet and Halil Karahan (2015). “Application of various optimization techniques and comparison of their performances for predicting TBM penetration rate in rock mass”. In: *International Journal of Rock Mechanics and Mining Sciences* 80 (2015), pp. 308–315.

Yamín, René A. and David J. Harmelink (2001). “Comparison of Linear Scheduling Model (LSM) and Critical Path Method (CPM)”. In: *Journal of Construction Engineering and Management* 127 (September/October 2001), pp. 374–381.

Yildiz, Acelya Ecem, Irem Dikmen, and M. Talat Birgonul (2020). “Using System Dynamics for Strategic Performance Management in Construction”. In: *Journal of Management in Engineering* 36.2 (2020).

Zhang, Hong, Heng Li, and Ming Lu (2008). “Modeling Time-Constraints in Construction Operations through Simulation”. In: *Journal of Construction Engineering and Management* 134.7 (2008), pp. 545–554.

Zhang, Hong, C. M. Tam, and Heng Li (2005). “Activity Object-Oriented Simulation Strategy for Modeling Construction Operations”. In: *Journal of Computing in Civil Engineering* 19.3 (2005), pp. 313–322.

Zhang, Jun et al. (2017). “Earth Dam Construction Simulation Considering Stochastic Rainfall Impact”. In: *Computer-Aided Civil and Infrastructure Engineering* 33.6 (2017), pp. 459–480.

Zhou, Jian et al. (2019). “Forecasting of TBM advance rate in hard rock condition based on artificial neural network and genetic programming techniques”. In: *Bulletin of Engineering Geology and the Environment* 79.4 (2019), pp. 2069–2084.

Zongben, Xu and Cao Feilong (2004). “The essential order of approximation for neural networks”. In: *Science in China Series F* 47.1 (2004), p. 97.

Norms and Standards

DIN 69900 (2009-01). *Projektmanagement – Netzplantechnik; Beschreibungen und Begriffe*. German. Ed. by Deutsches Institut für Normung e.V. 2009-01.

IEEE 1516 (2010). *IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA)– Framework and Rules*. Ed. by Institute of Electrical and Electronics Engineers. 2010.

TPH Bausysteme, ed. (2019a). *Technisches Datenblatt: PUR-O-STOP FS-L*. German. Technical specification. Stand 05.12.2019. TPH Bausysteme GmbH. Norderstedt, Germany: TPH Bausysteme GmbH, 2019.

– ed. (2019b). *Technisches Datenblatt: RUBBERTITE*. German. Technical specification. Stand 04.03.2019. TPH Bausysteme GmbH. Norderstedt, Germany: TPH Bausysteme GmbH, 2019.

VDI 3633 Blatt 8 (2007). *Simulation von Logistik-, Materialfluss- und Produktionssystemen - Maschinennahe Simulation*. German. Verein Deutscher Ingenieure. 2007.

Internet

Aegerter & Bosshardt, ed. (2019). *Company Website*. German. Aegerter & Bosshardt AG. 2019. URL: http://www.aebo.ch/ (visited on Nov. 10, 2019).

AnyLogic (2020). *AnyLogic: Simulation Modeling Software Tools & Solutions for Business*. Website. The AnyLogic Company. 2020. URL: https://www.anylogic.com/ (visited on Sept. 15, 2020).

Banque centrale du Luxembourg, ed. (2020). *Entwicklung der Rendite zehnjähriger Staatsanleihen Deutschlands in den Jahren von 1995 bis 2019*. German. quoted from de.statista.com. Banque centrale du Luxembourg. 2020. URL: https://de.statista.com/statistik/daten/studie/200193/umfrage/entwicklung-der-rendite-zehnjaehriger-staatsanleihen-in-deutschland/ (visited on Apr. 10, 2020).

Bloomberg, Jason (2018). *Digitization, Digitalization, And Digital Transformation: Confuse Them At Your Peril*. Ed. by Forbes. (published on Apr. 29, 2018, 08:42am EST). Forbes Media LLC. 2018. URL: https://www.forbes.com/sites/jasonbloomberg/2018/04/29/digitization-digitalization-and-digital-transformation-confuse-them-at-your-peril/ (visited on Mar. 14, 2020).

Camebridge Dictionary, ed. (2020). *Too many cooks spoil the broth*. 2020. URL: https://dictionary.cambridge.org/dictionary/english/too-many-cooks-spoil-the-broth (visited on Nov. 23, 2020).

Caterpillar, ed. (2020). *Caterpillar*. Company Website. Caterpillar Inc. 2020. URL: https://www.caterpillar.com/ (visited on Oct. 8, 2020).

DB Projekt Stuttgart-Ulm, ed. (2019). *Bauleistung Stuttgart 21 Tunnel Feuerbach*. German. 2019. URL: http://www.bahnprojekt-stuttgart-ulm.de/baustelle/vortrieb-und-aushub-fuer-tunnelbau/s21-vortrieb-und-aushub/downloadmedium/tunnel-feuerbach-vortrieb-und-aushub-tunnelbau-grafik/downloadParameter/download/Medium// (visited on Feb. 22, 2020).

eguana, ed. (2019). *Company Website*. German. eguana GmbH. 2019. URL: https://eguana.at/ (visited on Oct. 31, 2019).

Eiben, A. E. and M. Schoenauer (2020). *Evolutionary Computing*. 2020. URL: https://arxiv.org/pdf/cs/0511004.pdf (visited on Apr. 12, 2020).

Ellgaard, Holger (2009). *Svenska: Citybanan, Södermalmstunnelns arbetstunnel, visning den 2009-09-20*. Creative Commons Attribution 3.0 Unported, https://creativecommons.org/licenses/by/3.0/deed.en. 2009. URL: https://commons.wikimedia.org/wiki/File:S%C3%B6dermalmstunneln_2009a.jpg (visited on June 22, 2020).

Facebook (2020). *Internet Search Engine*. Social networking service Publisher. Facebook Inc. 2020. URL: https://www.facebook.com/ (visited on Nov. 12, 2020).

Forrester, Jay W. (1989). *The Beginning of System Dynamics*. Banquet Talk at the international meeting of the System Dynamics Society Stuttgart, GermanyJuly 13, 1989. System Dynamics Society. 1989. URL: http://web.mit.edu/sysdyn/sd-intro/D-4165-1.pdf (visited on Sept. 19, 2020).

Gartner, Inc., ed. (2020). *Information Technology Gartner Glossary*. Gartner Inc. 2020. URL: https://www.gartner.com/en/information-technology/glossary (visited on Mar. 14, 2020).

Google (2020). *Internet Search Engine*. German. Internet Search Engine. Alphabet Inc. 2020. URL: https://www.google.de/ (visited on Nov. 12, 2020).

IBM (2020). *The IBM PC's debut: At the beginning*. IBM Corporation. 2020. URL: https://www.ibm.com/ibm/history/exhibits/pc25/pc25_intro.html (visited on Sept. 7, 2020).

ieee Systems (2020). *Stella*. Website. ieee systems inc. 2020. URL: https://www.iseesystems.com/store/products/stella-architect.aspx (visited on Sept. 19, 2020).

infoRoad, ed. (2020). *Historische Preisentwicklung*. German. Website. infoRoad GmbH. 2020. URL: https://www.clever-tanken.de (visited on Mar. 4, 2020).

Ingber, Lester (1995). *Adaptive simulated annealing (ASA): Lessons learned*. 1995. URL: https://www.ingber.com/asa96_lessons.ps.gz (visited on Apr. 12, 2020).

Intel (1990). *8086 16-Bit HMOS Microprocessor 8086/8086-2/8086-1*. Data Sheet. Intel. 1990. URL: http://datasheets.chipdb.org/Intel/x86/808x/datashts/8086/231455-006.pdf (visited on Oct. 8, 2020).

Ioannou, Photios G. (2020). *Photios G. Ioannou*. 2020. URL: http://www.ioannou.org/ (visited on Sept. 2, 2020).

Kleiner, Art (2009). *Jay Forrester's Shock to the System*. Online Article. MIT Sloan Management Review. 2009. URL: https://sloanreview.mit.edu/article/jay-forrester-shock-to-the-system/ (visited on Sept. 19, 2020).

MANITOU, ed. (2020). *Manitou 180 ATJ Gelenk-Teleskopbühne*. German. MANITOU BF SA. 2020. URL: https://www.hkl-baumaschinen.de/manitou-180-atj-gelenk-teleskopbuehne-mieten (visited on Mar. 4, 2020).

MathWorks (2019a). *gamultiobj Algorithm*. 2019. URL: https://de.mathworks.com/help/releases/R2019a/gads/gamultiobj-algorithm.html (visited on Oct. 19, 2020).

– ed. (2019b). *MATLAB R2019a Documentation*. The Language of Technical Computing. MathWorks, Inc. 2019. URL: https://de.mathworks.com/help/releases/R2019a/index.html (visited on Mar. 26, 2020).

– (2019c). *What Is Multiobjective Optimization?* 2019. URL: https://de.mathworks.com/help/releases/R2019a/gads/what-is-multiobjective-optimization.html (visited on Oct. 19, 2020).

– (2020). *Get Started with Parallel Computing Toolbox. Perform parallel computations on multicore computers, GPUs, and computer clusters*. 2020. URL: https://de.mathworks.com/help/parallel-computing/getting-started-with-parallel-computing-toolbox.html (visited on July 1, 2020).

Microsoft, ed. (2020). *OneDrive*. German. 2020. URL: https://www.microsoft.com/de-de/microsoft-365/onedrive/online-cloud-storage (visited on July 1, 2020).

Paulsen, Enno (2019). *Eckdaten zur Baupreiskalkulation*. German. Ed. by Bauindustrieverband NRW e.V. Bauindustrieverband NRW e.V. 2019. URL: https://www.bauindustrie-nrw.de/service/bauwirtschaftsdaten/ (visited on Mar. 5, 2020).

Powersim Software (2020). *Powersim software*. Website. Powersim Software AS. 2020. URL: https://powersim.com/ (visited on Sept. 19, 2020).

Purdue (2020). *Simulation in Construction using CYCLONE & MicroCYCLONE*. Ed. by Purdue University. Purdue University, West Lafayette, IN. 2020. URL: https://engineering.purdue.edu/CEM/people/Personal/Halpin/Sim/index_html (visited on Aug. 25, 2020).

Reed, Matthew (2020). *The TRS-80 Model II*. Website. 2020. URL: http://www.trs-80.org/model-2/ (visited on Sept. 7, 2020).

Renesco, ed. (2019). *Company Website*. German. Renesco GmbH. 2019. URL: https://www.renesco.com/ (visited on Oct. 31, 2019).

RIB Software SE, ed. (2019). *iTWO für externe Dienstleister der DB AG*. German. 2019. URL: https://www.rib-software.com/loesungen/itwo-db/ (visited on Nov. 17, 2019).

Ritchie, Dennis M. (2020). *The development of the C language*. 2020. URL: https://www.bell-labs.com/usr/dmr/www/chist.html (visited on Sept. 2, 2020).

Spicer, Dag (2000). *Control Data 6600: The Supercomputer Arrives*. 2000. URL: https://web.archive.org/web/20190707174250/http://www.drdobbs.com/control-data-6600-the-supercomputer-arri/184404102 (visited on Sept. 26, 2020).

Swaine, Michael R. and Paul A. Freiberger (2020). *UNIVAC computer*. Website. Encyclopaedia Britannica (Online). 2020. URL: https://www.britannica.com/technology/UNIVAC (visited on Oct. 6, 2020).

SysML (2020). *SysML Open Source Project - What is SysML? Who created SysML?* Object Management Group. 2020. URL: https://sysml.org/ (visited on Sept. 15, 2020).

TechPowerUp (2020a). *NVIDIA GeForce GTX 1660 SUPER*. Ed. by W1zzard. 2020. URL: https://www.techpowerup.com/gpu-specs/geforce-gtx-1660-super.c3458 (visited on July 1, 2020).

– (2020b). *NVIDIA Quadro P2000*. Ed. by W1zzard. 2020. URL: https://www.techpowerup.com/gpu-specs/quadro-p2000.c2931 (visited on July 1, 2020).

The Economist (2017). *The world's most valuable resource is no longer oil, but data*. Article appeared in the Leaders section of The Economist print edition under the headline "The world's most valuable resource". The Economist Group Limited. 2017. URL: https://www.economist.com/leaders/2017/05/06/the-worlds-most-valuable-resource-is-no-longer-oil-but-data (visited on May 6, 2017).

VDI (2010). *Neue Formen der Zusammenarbeit und der Vertragsgestaltung bei Planung und Ausführung großer Bauprojekte*. German. Ed. by Reinhold Jesorsky and Dirk Manske. Verein Deutscher Ingenieure. 2010. URL: https://www.vdi.de/uploads/media/10-07_Partnerschaft_am_Bau_03.pdf (visited on Nov. 26, 2016).

Ventana Systems (2020). *Vensim*. Website. Ventana Systems Inc. 2020. URL: https://vensim.com/ (visited on Sept. 19, 2020).

WBI, ed. (2019). *Company Website*. German. WBI GmbH. 2019. URL: https://www.wbionline.de/ (visited on Oct. 31, 2019).

Weisstein, Eric W. (2020a). *Convolution*. MathWorld–A Wolfram Web Resource. 2020. URL: https://mathworld.wolfram.com/Convolution.html (visited on Oct. 23, 2020).

– (2020b). *Median*. MathWorld–A Wolfram Web Resource. 2020. URL: https://mathworld.wolfram.com/Median.html (visited on July 6, 2020).

– (2020c). *Relative Error*. MathWorld–A Wolfram Web Resource. 2020. URL: https://mathworld.wolfram.com/RelativeError.html (visited on May 31, 2020).

– (2020d). *Statistical Median*. MathWorld–A Wolfram Web Resource. 2020. URL: https://mathworld.wolfram.com/StatisticalMedian.html (visited on July 6, 2020).

Weisstein, Eric W. (2020e). *Wolfram MathWord. the web's most extensive mathematics resource*. MathWorld–A Wolfram Web Resource. 2020. URL: https://mathworld.wolfram.com/ (visited on Oct. 27, 2020).

Westman, George (2017). *Digital transformation: MIT's Westerman shares new lessons*. Ed. by Ginny Hamilton. 2017. URL: https://enterprisersproject.com/article/2017/5/digital-transformation-mits-westerman-shares-new-lessons (visited on Nov. 12, 2020).

Weyhrich, Steven (2020). *The Apple II Plus*. Website. 2020. URL: https://apple2history.org/history/ah06/ (visited on Sept. 7, 2020).

Workyard, ed. (2020). *Company Website*. 2020. URL: https://www.workyard.com/ (visited on Mar. 14, 2020).

zafaco GmbH (2020). *Willkommen bei der Breitbandmessung der Bundesnetzagentur*. German. 2020. URL: https://breitbandmessung.de/ (visited on July 1, 2020).

Zub, Russel (1981). *A computer program (VEHSIM) for vehicle fuel economy and performance simulation (automobiles and light trucks)*. Volume I: Description and Analysis. Version Final Report. Report No. DOT-HS-806-037. 1981. URL: https://archive.org/details/computerprogramv00zubr/page/n1/mode/2up (visited on Sept. 16, 2020).

Appendix A Notation

Nomenclature

This sections provides a short description of variables and operators used in this document.

Generic Operators

$\bigwedge_{i=1}^{n} x$	Arbitrary AND-operator, i. e., $x_1 \wedge x_2 \wedge \cdots \wedge x_n = \bigwedge_{i=1}^{n} x_i$
$\forall x, y$	For all x applies y
$\lceil x \rceil$	Ceil operator, i. e., next highest integer larger than x
$\lfloor x \rfloor$	Floor operator, i. e., next smallest integer smaller than x
$\longrightarrow$	Chemical reaction
$\sum_{i=a}^{b} x_i$	Sum of x_i from $i = a$ to $i = b$
$A \Rightarrow B$	From A follows B
$n \to \infty$	The value a approaches infinity

Set Theory

$\varnothing$	Empty set
$\mathbb{N}$	Set of all natural numbers
$\mathbb{R}$	Set of all real numbers
$\mathbb{Z}$	Set of all integer numbers
$A \cap B$	Intersection of set A and set B
$A \cup B$	Union of set A and set B
$a \in A$	a is an element of the set A
$A \mapsto B$	The set A can be mapped to the set B
$a \notin A$	a is not an element of the set A
$A \setminus B$	Complement, also difference, of set B in set A

$A \subseteq B$	A is a subset of B
$A \times B$	Cross product of A and B

Deep Learning

v_i	Weighted sum of the output node i
α	Learning rate ($0 < \alpha \leq 1$)
β	Momentum decay rate
δ	Delta rule, i. e., contributes of x_j to e_i
φ	Activation function
φ'	Derivative of the activation function φ
b	Bias
d_i	Correct output at node i
e_i	Error of output at node i
l_i	Layer number i
m_t	Momentum at time step t
t	Iteration step
w_{ij}	Weight between node i and node j
x	Input to neuron
x_j	Output from input node j
y	Output from neuron
y_i	Calculated output at node i

Graph Theory

$A \mapsto B$	Set A can be mapped to the set B
A	Adjacency matrix
a_{ij}	Boolean adjacency relation $a \in A$ from v_i to v_j
E	Set of edges or connections
e	Edge or connections
$E_1 \sqcup E_2$	Joining of set of edges E_1 with E_2

G	Graph
n	Node (same as vertex v)
$n_i \to n_j$	A sequence of nodes in which n_j is the successor of n_i
P	Probability matrix
p_{ij}	Probability $p \in P$ to change from v_i and v_j
R	Relation matrix
R^m	The $m^{(\text{th})}$ power of R
R_t	Transitive hull of relation matrix R
S	Sub-graph, such that $S \subseteq G$
V	Set of nodes or vertices
v	Vertex (same as node n)
W	Weight matrix
w_{ij}	Weight $w \in W$ of edge e_{ij}

Numerical Optimization

d_i	Distance of solution i to all its neighbor solutions
$f(x)$	Objective or goal function
G	Genome
g	Gene
X	Parameter set
x	Parameter
x^*	Parameter solution on Pareto frontier

Statistics, Monte Carlo Method, Markov Chains

$\bar{x}$	Mean value of sample
$\bar{x}_{p\%}$	Trimmed sample mean with p being the trimmed range in percent
$\cdot$	Any other element in the given set
Δx	Absolute Error
δx	Relative Error Error

δ_{MAPE}	Mean Absolute Percentage Error
δ_{MSE}	Mean Square Error
δ_{NMSE}	Normalized Mean Square Error
δ_{NRMSE}	Normalized Root Mean Square Error
$\langle x \rangle$	Mean value of a population
$\mathcal{U}(a, b)$	Uniform, continuous distribution within the boundaries a and b
$\mathcal{U}\{a, b\}$	Uniform, discrete distribution within the boundaries a and b
$\mathcal{X}$	Sample space
μ	Mean value
$\pi(\theta)$	Prior distribution function
$\pi^{(i)}$	Probability that x takes the value x_i after n transitions
ρ	Uniformly distributed random number
σ	Standard deviation of a population
σ^2	Variance of a population
τ_y	Number of transitions to reach y from x
$\mathrm{Prob}(x)$	Probability that x is true
$f(x \mid \theta)$	Density of x conditional on θ
K	Transition kernel / matrix
$K(x \to u)$	Transition probability of transition from x to u
N	Size of population or sample
$P(X \mid Y)$	Conditional probability of the likelihood that X is true given that Y is true
P_{xy}	Probability of transition from x to y
s	Standard deviation of a sample
s^2	Variance of a sample
U	Unit rectangular CDF
$X \sim f$	X is distributed with density f

Appendix B List of Abbreviations

ABC	*Activity-Based Construction*
ABM	*Agent Based Model*
AC	*Acrylate*
AI	*Artificial Intelligence*
ANN	*Artificial Neural Network*
AOA	*activity-on-arrow diagram*
AON	*activity-on-node diagram*
BIM	*Building Information Modeling*
BMVI	*Bundesministerium für Verkehr und Digitale Infrastruktur*
BP	*Back-Propagation*
BRTV	*Bundesrahmentarifvertrag*
CAD	*Computer-aided Design*
CCTV	*Closed Circuit Television*
CDF	*Cumulative Distribution Function*
CLCPM	*Constraint-Loaded Critical Path Method*
CLT	*Central Limit Theorem*
cma	*Central Moving Average*
COOP	*Costs of Opportunity*
CPN	*Colored Petri Net*
CPM	*Critical Path Method*
CPU	*Central Processing Unit*
CRC	*Collaborative Research Center*
CSD	*Construction Site Dewatering*
CSM	*Colorado School of Mines*
csv	*Comma Separated Value*
CUDA	*Compute Unified Device Architecture*
CYCLONE	*Cyclic Operation Network*
DAT	*Decision Aids for Tunneling*
DC	*Delay Cost*
DCV	*Dynamic Construction Visualizer*
DEM	*Discrete Event Model*
DES	*Discrete Event Simulation*
DNA	*Deoxyribonucleic Acid*
EODD	*Earliest-Operation-Due-Date*
EP	*Engine Power*
ERP	*Enterprise-Resource-Planning*
FNN	*Feedforward Neural Network*

FIDIC	*Fédération Internationale des Ingénieurs Conseils*
FIFO	*First In – First Out*
GA	*Genetic Algorithm*
GPR	*Gaussian Process Regression*
GPS	*General Purpose Simulation*
GEA	*Geräteeinheit Acrylat*
GERT	*Graphical Evaluation and Review Technique*
GIF	*Graphics Interchange Format*
GIS	*Geographic Information System*
GVA	*Gross Value Added*
GMDH	*Group Method of Data Handling*
GPU	*Graphics Processing Unit*
GUI	*Graphical User Interface*
HLA	*High Level Architecture*
HMM	*Hidden Markov Model*
HOAI	*Verordnung über die Honorare für Architekten und Ingenieure*
ICDF	*Inverse Cumulative Distribution Function*
ICONS	*Interactive Construction Simulation*
IT	*Information Technology*
JDS	*Job Distribution System*
KDP	*Knowledge Discovery Process*
KMOS	*Knowledge-Embedded Modularized Simulation System*
KPI	*Key Performance Indicator*
LLN	*Law of Large Numbers*
LOB	*Line of Balance*
LOD	*Level of Detail*
LP	*Linear Programming*
MAPE	*Mean Absolute Percentage Error*
MC	*Markov Chain*
MCM	*Monte-Carlo Method*
MCS	*Monte-Carlo Simulation*
MCMCS	*Markov-Chain Monte-Carlo Simulation*
MicroCYCLONE	*Microcomputer Cyclic Operation Network*
MiSAS	*Microtunnelling: Statistics, Analysis and Simulation*
MOGA	*Multi Objective Genetic Algorithm*
MPM	*Metra Potential Method*
MobPU	*Mobile Polyurethane Injection Unit*
MS	*Microsoft*
MSE	*Mean Square Error*
MSPS	*Multi Server Production System*
MSPS2	*Multi Server Provduction System 2*
MSPS3	*Multi Server Production System 3*
MVC	*Model View Controller*
NaN	*Not a Number*
NaT	*Not a Time*

NEDO	*National Economic Development Office, London, UK*
NMSE	*Normalized Mean Square Error*
NRMSE	*Normalized Root Mean Square Error*
NTNU	*Norwegian University of Science*
OECD	*Organisation for Economic Co-operation and Development*
OO	*Object-Oriented*
OS	*Operating System*
OV	*Original Value*
PDF	*Probability Density Function*
PERT	*Program Evaluation and Review Technique*
PICASSO	*Project Integrated Cyclic Analysis of Serial System Operations*
PN	*Petri Net*
PM	*Project Management*
PU	*Polyurethane*
RACPM	*resource-activity critical-path method*
RAM	*Risk Analysis & Mitigation*
RC	*Repair Cost*
RESQUE	*Resource based Queuing Network Simulation System*
SA	*Simulated Annealing*
SCGA	*Scaled Conjugated Gradient Algorithm*
SC	*Single-Point Crossover*
SCC	*Sample Cross Correlation*
SD	*System Dynamics*
SDESA	*Simplified Discrete Event Simulation Approach*
SFB	*Sonderforschungsbereich*
SimCon	*Simulation-based project Control*
SN	*Sequence Number*
SPS	*Single Purpose Simulation*
SSPS	*Single Server Production System*
STEPS	*Structured Environment for Process Simulation*
SUT	*Single Purpose Simulation for Utility Tunneling*
SVR	*Support Vector Regression*
SysML	*System and Modeling Language*
TC	*Two-Point Crossover*
TCM	*Tunnel Cost Model*
TBM	*Tunnel Boring Machine*
TM	*performance/cost Tunnel Model*
TPN	*Timed Petri Net*
TSC/FMA	*Transportation Systems Center/Foster Miller Associates*
UC	*Uniform Crossover*
UK	*United Kingdom*
USA	*United States of America*
UTF-8	*8-Bit Universal Coded Character Set Transformation Format*
UNIVAC I	*Universal Automatic Computer I*
VBA	*Visual Basics for Applications*

VOB *Vergabe- und Vertragsordnung für Bauleistungen*
WLAN *Wireless Local Area Network*

Appendix C Tables and Figures

C.1 Status Diagrams of Construction Site

The diagrams shown in this chapter supplement those given in chapter 5.

C.1.1 Position of Injection Units in Time-Location Diagram

In this section, a number of time-location diagrams are presented. These diagrams show the location of each injection unit at different points in time. The diagrams were created based on the timestamp of injections, which are again linked to a borehole's location identifier. Diagrams have been limited to the four primary tubes to account for the limited space in this work; i. e., those for connection buildings are not included.

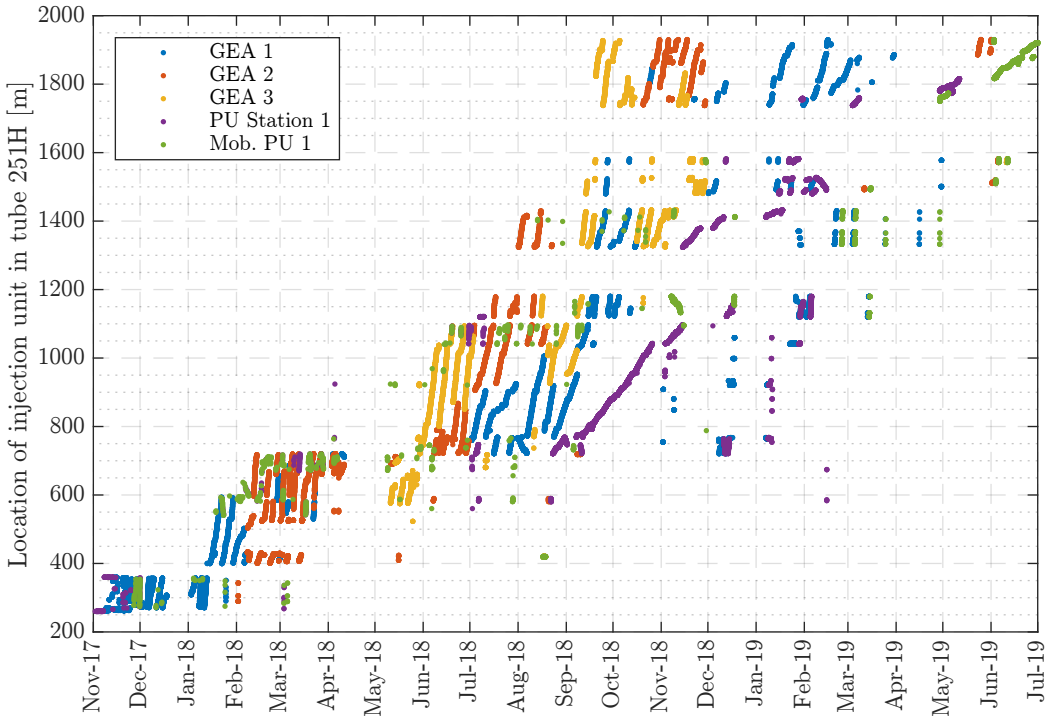


Figure C.1: Location of injection units in tube 251H.

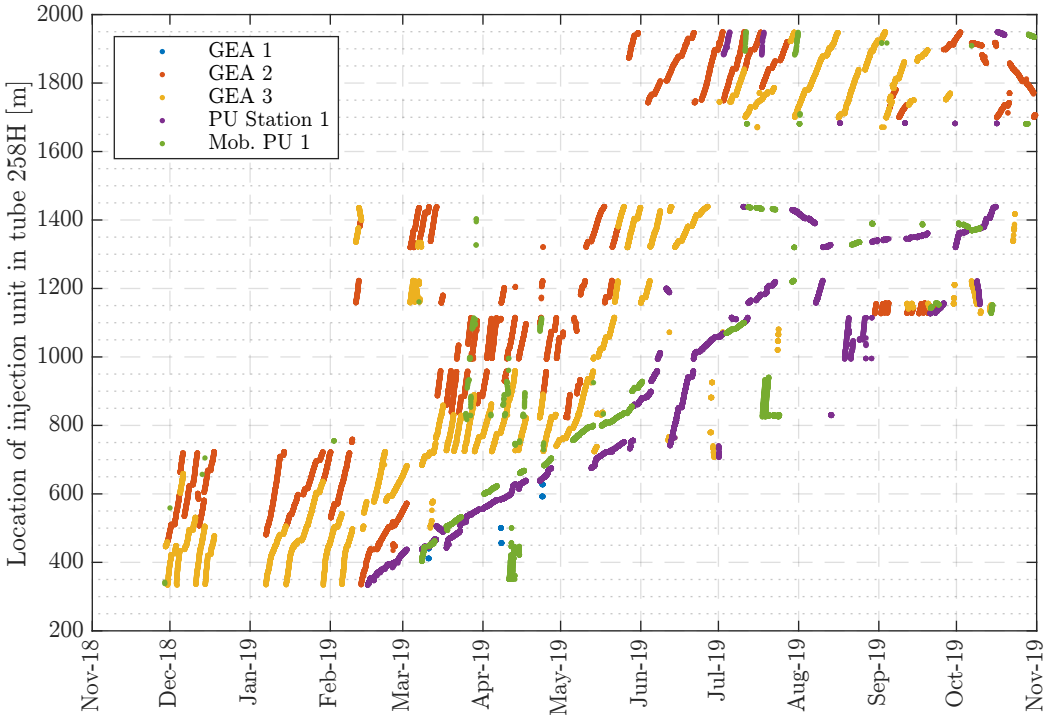


Figure C.2: Location of injection units in tube 258H.

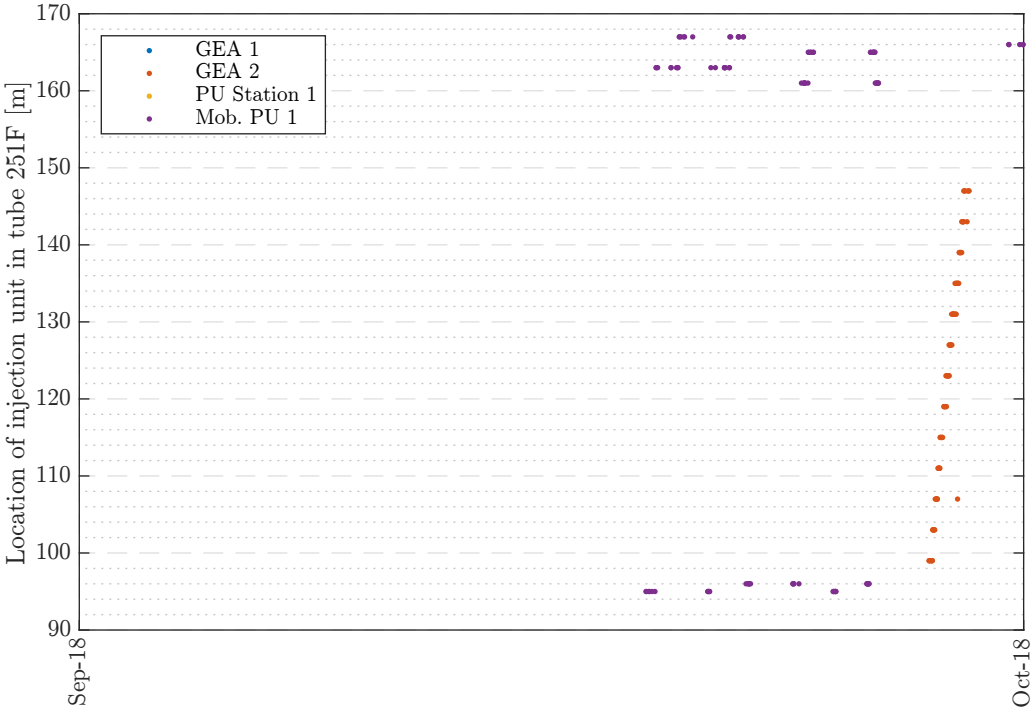


Figure C.3: Location of injection units in tube 251F.

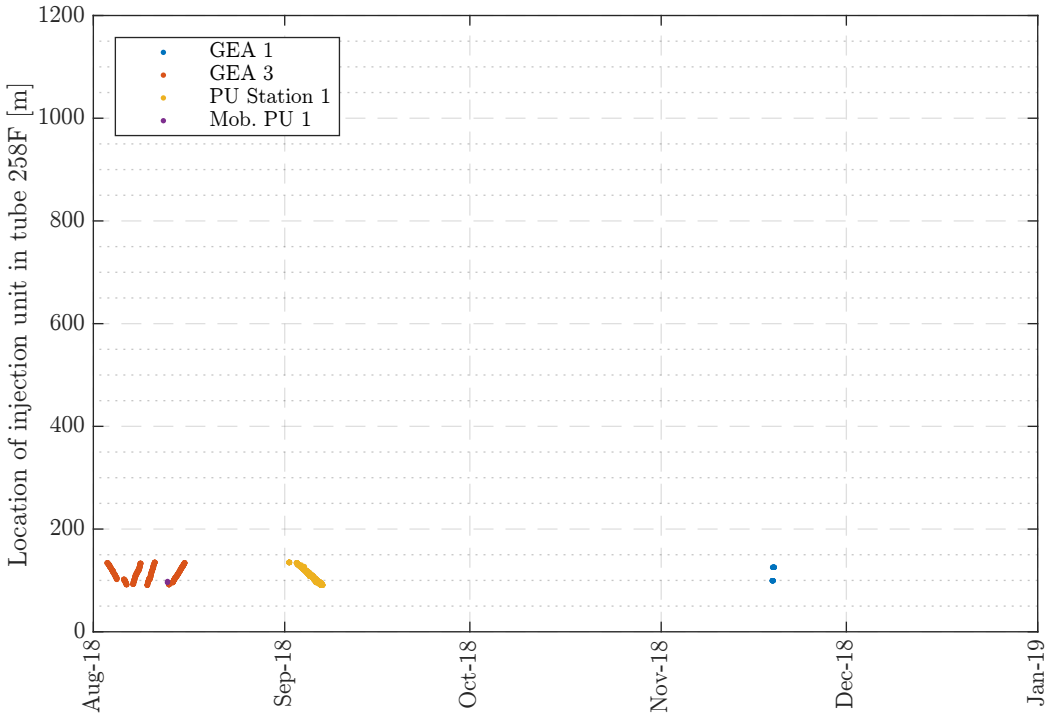


Figure C.4: Location of injection units in tube 258F.

C.1.2 Duration of Different Process Types per Day

This section contains diagrams of the *Central Moving Average* (cma) of the measured processes over the course of the project. The cma is defined by Savitzky and Golay (1964) as

$$\mathrm{cma}_i = \frac{1}{2k+1} \sum_{n=i-k}^{i+k} S_n. \tag{C.1}$$

The value k controls the number of days to be included in the calculation of the cma. For the presented diagrams it is $k = 5$, i. e., data from eleven days are used to calculate the cma. S_n denotes the sum of the duration of all processes on day n.

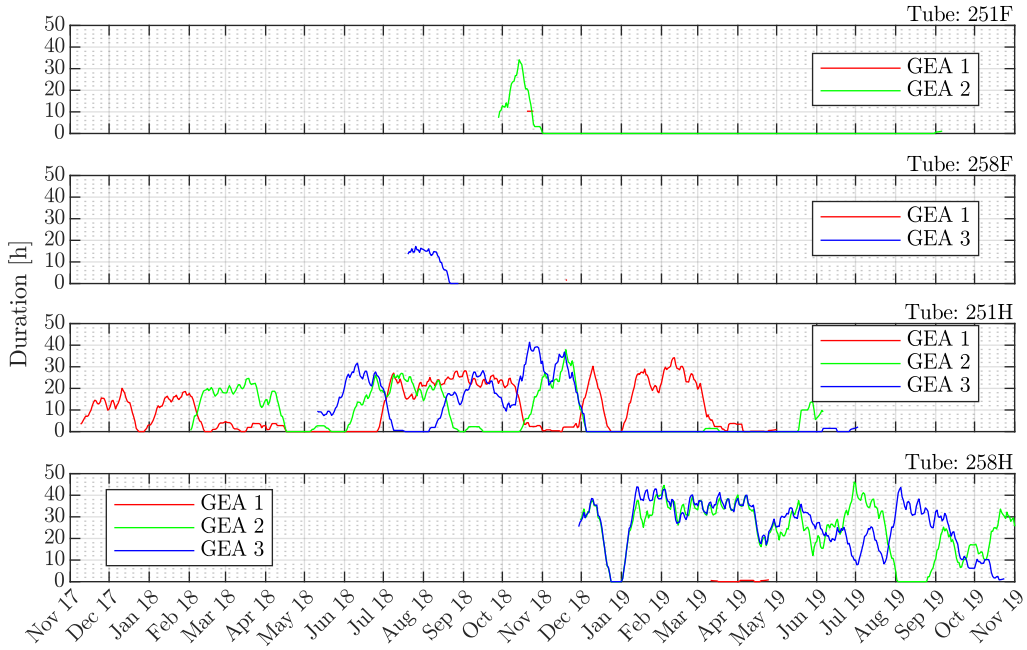


Figure C.5: Injection process duration per day. Depicted is the cma over an interval of eleven days.

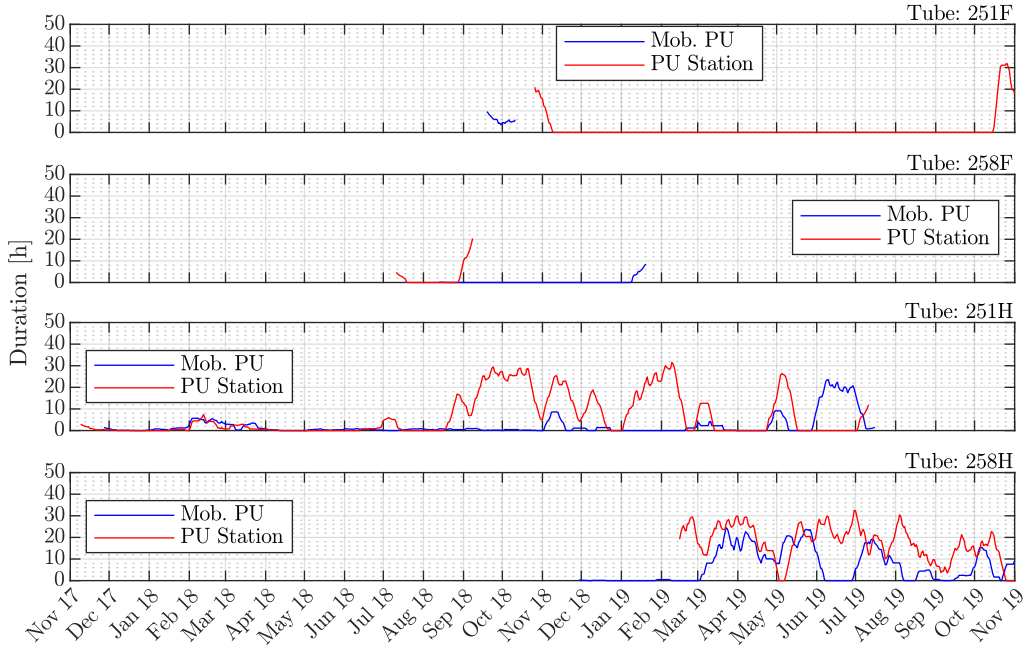


Figure C.6: Injection process duration per day. Depicted is the cma over an interval of eleven days.

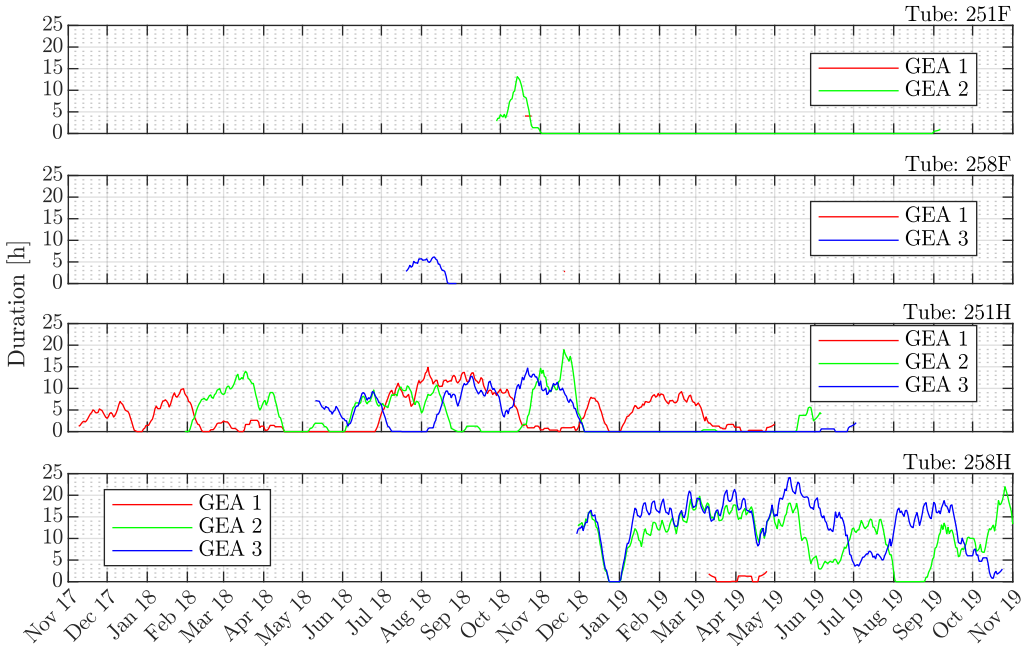


Figure C.7: Moving process duration per day. Depicted is the cma over an interval of eleven days.

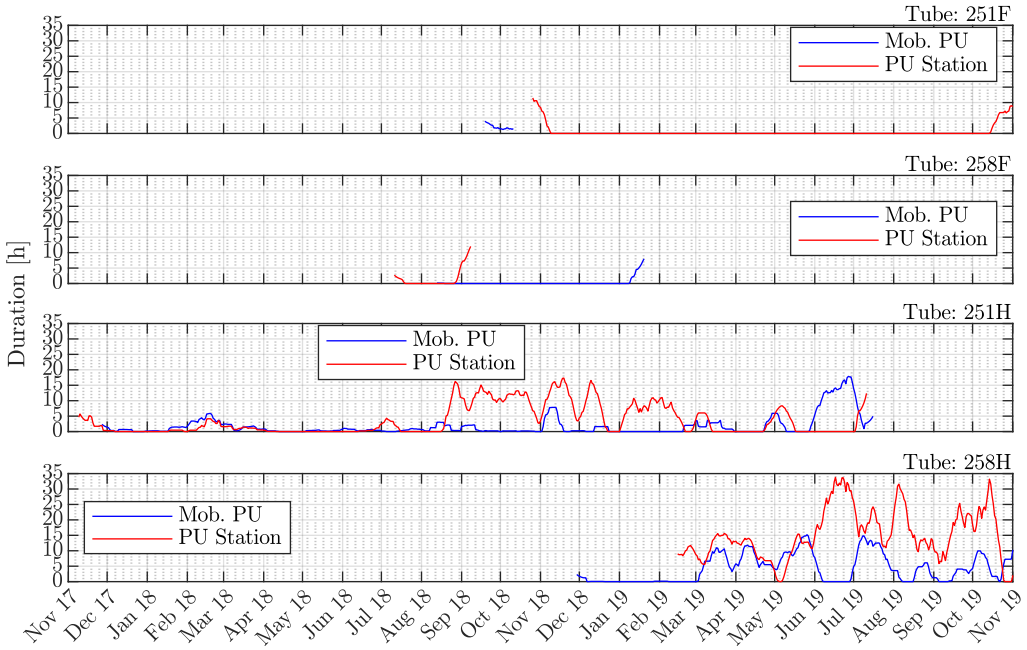


Figure C.8: Moving process duration per day. Depicted is the cma over an interval of eleven days.

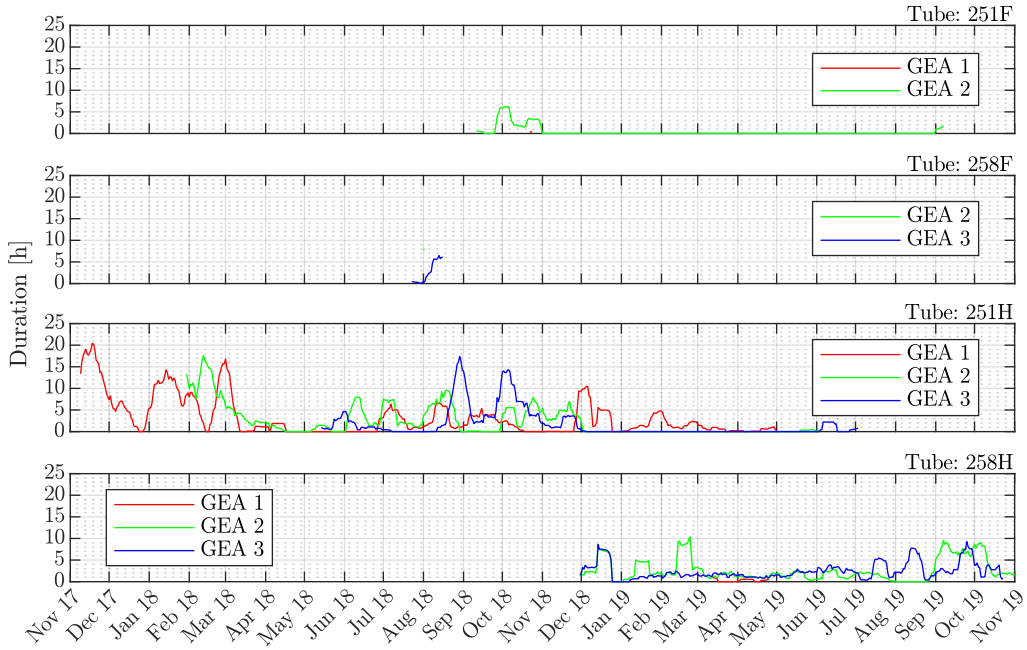


Figure C.9: Maintenance and malfunction process duration per day. Depicted is the cma over an interval of eleven days.

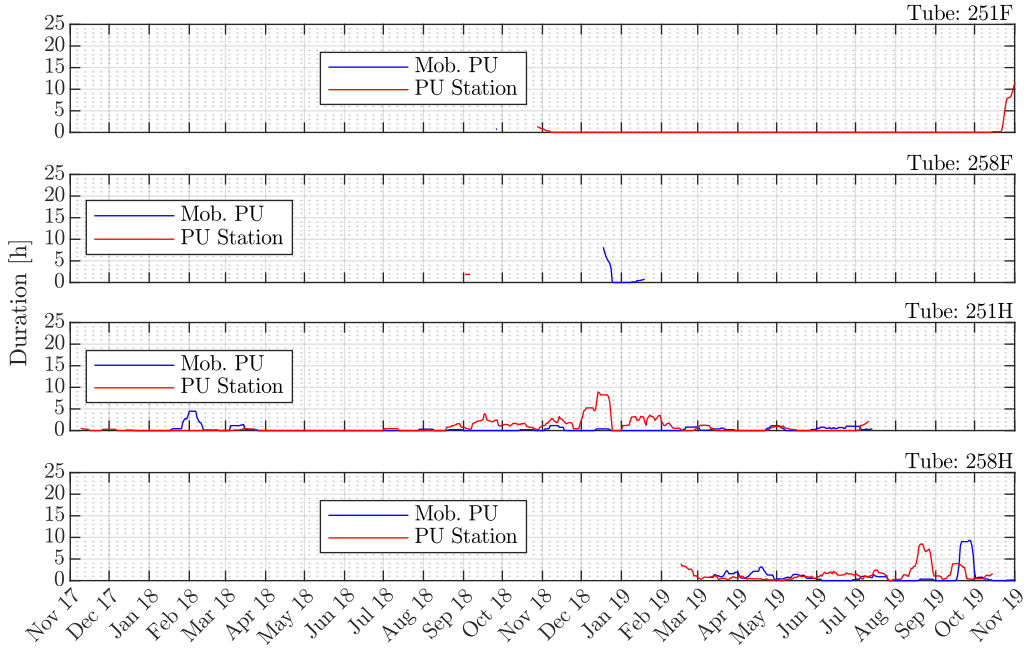


Figure C.10: Maintenance and malfunction process duration per day. Depicted is the cma over an interval of eleven days.

C.2 Result Tables of Artificial Neural Network Grout Volume Forecast

The result tables show in this chapter supplement those presented in section 5.6.3 on the grout volume forecast by using ANNs.

The tables show the MAPEs achieved with different configurations of FNNs. The variable names follow the notation defined in section 5.6.3, i. e.,

- δ_{MAPE}: Mean Absolute Percentage Error,
- k: Number of elements included in the calculation of the moving average,
- i: Number of nodes on the input layer,
- o: Number of nodes on the output layer,
- h: Number of hidden layers, and
- H: Number of nodes on the hidden layers. The symbol "×" signals that the following number denotes the number of nodes on each hidden layer. Several discrete values, such as "50 10 25" mean, that the number of nodes on the hidden layers differs. From left to right, each value gives the number of nodes on the first, second, etc. layer.

Table C.1: δx_{MAPE} of forecast of moving average grout volume per segment (1 of 2)

δ_{MAPE} [%]	k	i	o	h	H	δ_{MAPE} [%]	k	i	o	h	H
54	35	50	50	10	×250	121	11	200	100	6	×600
54	35	50	50	3	×250	123	25	50	25	-	100 50
55	35	50	50	20	×450	123	25	10	10	1	×10
55	35	50	50	3	×150	160	11	200	50	6	×600
55	35	50	50	5	×250	160	25	50	25	1	×100
55	35	50	50	15	×600	160	25	50	25	3	×150
56	35	50	50	10	×600	160	11	50	400	6	×600
56	35	50	50	10	×450	161	25	50	25	-	50 100 25
56	35	50	50	20	×150	161	25	50	25	-	50 25
56	35	50	50	20	×600	161	11	400	100	6	×600
57	35	50	50	5	×600	161	25	50	25	-	50 100 50
57	35	50	50	20	×250	161	11	500	50	6	×600
57	35	50	50	10	×150	161	25	50	25	3	×100
57	35	100	50	10	×600	162	25	50	25	-	50 50 25
58	35	100	50	5	×600	162	25	50	25	4	×100
58	35	100	50	15	×600	162	25	50	25	3	×50
58	35	100	50	10	×250	162	11	500	100	6	×600
58	35	100	50	6	×600	162	25	50	25	4	×150
58	35	100	50	20	×600	162	25	50	50	6	×50
58	35	100	50	20	×250	162	25	50	50	4	×50
58	35	100	50	5	×250	162	25	50	50	1	×300
58	35	100	50	6	×500	163	25	100	50	1	×150
59	35	150	50	6	×500	163	25	50	50	6	×600
59	35	150	50	6	×600	164	25	50	50	10	×250
59	11	100	50	20	×600	164	25	50	50	3	×150
59	11	50	50	20	×150	165	25	50	50	7	×600
63	11	50	50	20	×600	165	25	50	50	1	×150
63	11	50	50	10	×250	165	25	50	50	3	×300
64	11	100	50	10	×250	165	25	50	50	2	×150
64	11	50	50	10	×150	165	25	50	50	5	×50
65	11	50	50	3	×150	165	25	50	50	1	×600
65	11	50	50	20	×250	166	25	50	50	5	×600
65	11	50	50	6	×600	166	25	100	50	1	×300
65	11	50	50	10	×450	174	25	50	50	5	×250
65	11	50	50	20	×450	174	25	50	50	4	×150
65	11	50	50	3	×250	177	25	50	50	1	×50
66	11	50	50	10	×600	177	25	100	50	7	×150
66	11	100	50	20	×250	179	25	50	50	20	×450
67	11	100	50	10	×600	179	25	100	50	7	×50
67	11	300	50	6	×600	182	25	50	50	4	×300
68	11	50	100	6	×600	182	25	50	50	2	×300
68	11	400	50	6	×600	188	25	50	50	10	×450
73	25	10	10	2	×10	188	25	100	50	20	×250
73	11	300	100	6	×600	195	25	50	50	3	×600
121	11	50	200	6	×600						

Table C.2: δx_{MAPE} of forecast of moving average grout volume per segment (2 of 2)

δ_{MAPE} [%]	k	i	o	h	H	δ_{MAPE} [%]	k	i	o	h	H
195	25	50	50	2	×600	292	25	100	50	4	×600
202	25	50	50	7	×50	292	25	200	25	6	×300
202	25	50	50	7	×300	292	25	100	50	20	×300
214	25	50	50	6	×150	292	25	300	100	6	×600
214	25	50	50	5	×150	294	25	100	50	40	×300
214	25	50	50	20	×600	294	25	100	50	10	×600
214	25	50	50	3	×50	294	25	100	50	50	×300
216	25	50	50	3	×250	294	25	50	100	6	×600
216	25	100	50	10	×250	296	25	100	50	5	×300
222	25	100	50	3	×150	296	25	100	50	7	×300
222	25	50	50	5	×300	296	25	100	50	6	×300
226	25	50	50	4	×600	296	25	100	50	5	×250
226	25	50	50	6	×500	296	25	100	50	14	×150
228	25	100	50	14	×300	296	25	100	50	4	×300
228	25	50	50	20	×150	296	25	100	50	25	×150
231	25	50	50	2	×10	296	25	100	50	6	×600
231	25	50	50	7	×150	296	25	100	50	5	×600
233	25	50	50	20	×250	296	25	100	50	7	×600
233	25	50	50	10	×150	297	25	100	100	6	×100
237	25	100	50	2	×150	297	25	100	100	6	×500
237	25	50	50	15	×600	297	25	200	50	7	×1000
237	25	100	50	10	×300	297	25	400	100	6	×600
237	25	50	50	10	×600	297	25	150	50	6	×500
242	25	50	50	2	×50	297	25	100	50	30	×300
242	25	50	50	6	×300	298	25	150	50	6	×600
246	25	100	50	4	×150	298	25	200	100	6	×600
246	25	100	50	20	×600	298	25	150	150	6	×150
247	25	100	50	12	×150	298	25	50	200	6	×600
247	25	100	50	2	×600	299	25	50	400	6	×600
248	25	100	50	1	×600	299	25	500	50	6	×600
248	25	100	50	20	×150	299	25	150	150	6	×500
251	25	400	50	6	×600	299	25	100	50	40	×150
251	25	100	50	8	×150	299	25	100	50	150	×150
252	25	100	50	10	×150	299	25	100	50	50	×150
252	25	100	50	30	×150	300	25	100	50	200	×300
257	25	100	50	6	×150	300	25	100	50	100	×150
257	25	100	50	5	×150	301	25	100	50	60	×300
271	25	100	50	15	×600	301	25	100	50	100	×400
271	25	100	50	8	×300	301	25	100	50	150	×400
285	25	100	50	3	×600	301	25	100	50	150	×300
285	25	100	50	12	×300	301	25	100	50	60	×150
285	25	100	50	3	×300	301	25	100	50	200	×150
285	25	100	50	3	×250	301	25	100	50	100	×300
288	25	100	50	2	×300	301	25	500	100	6	×600
288	25	100	50	6	×500						

C.3 MSPS1 Result Histograms of Monte-Carlo Simulation

This section shows a summary of MCS result plots. For the meaning of variables and symbols, and the interpretation of the results refer to section 6.3.1.

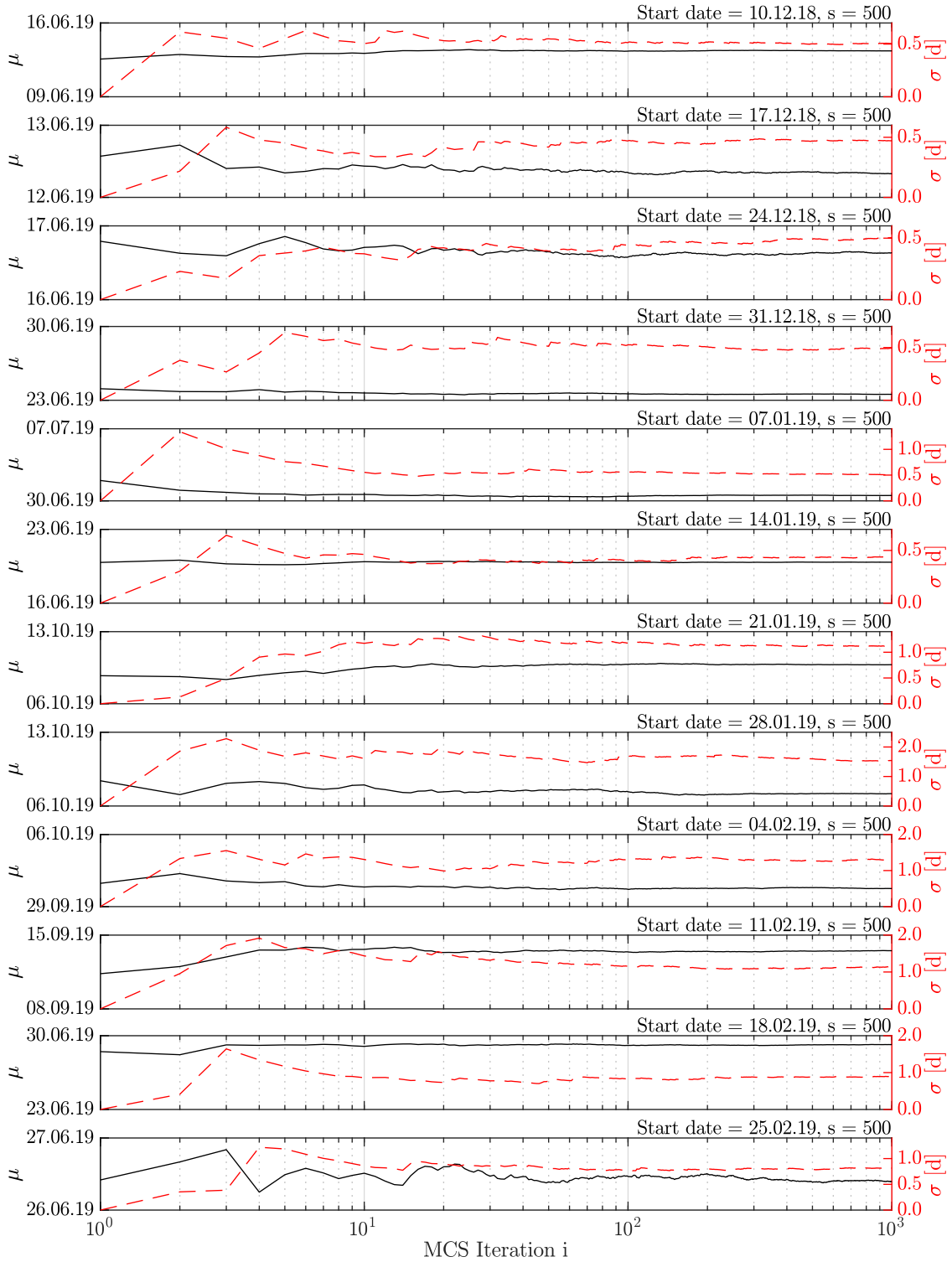


Figure C.11: Project Time Forecast of a MCS with sample size $s = 500$ boreholes and 1,000 iterations (part 1). Simulated are all AC units in tunnel tube 258H from 01.01.2018 to 15.05.2019. The mean value $\mu_i(p_i)$ (black) and the standard deviation $\sigma_i(p_i)$ (red, dotted) of the calculated project completion dates $p_i = [p_1, p_2, \ldots, p_i]$ are shown.

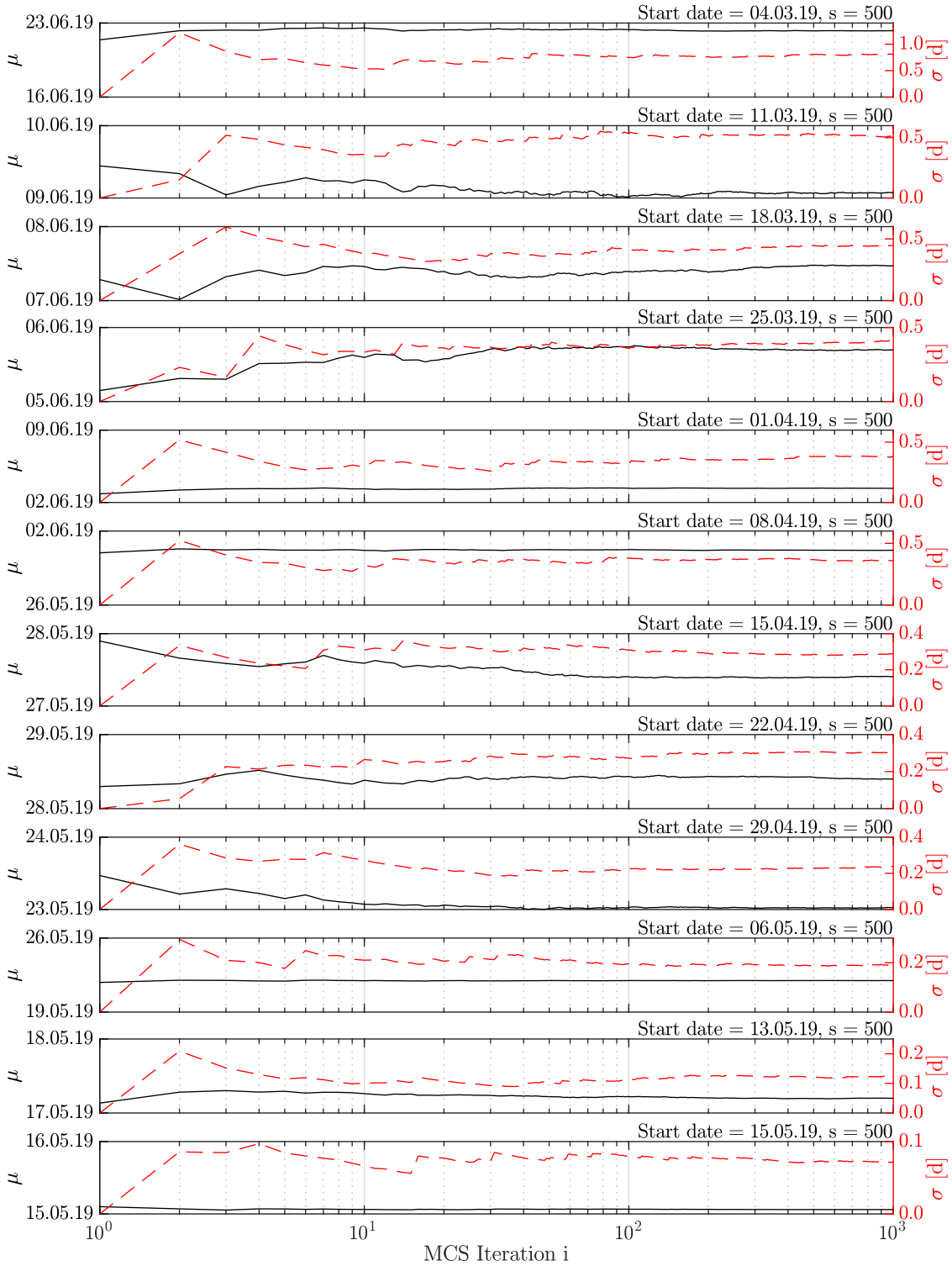


Figure C.12: Project Time Forecast of a MCS with sample size $s = 500$ boreholes and 1,000 iterations (part 2). Simulated are all AC units in tunnel tube 258H from 01.01.2018 to 15.05.2019. The mean value $\mu_i(p_i)$ (black) and the standard deviation $\sigma_i(p_i)$ (red, dotted) of the calculated project completion dates $p_i = [p_1, p_2, \ldots, p_i]$ are shown.

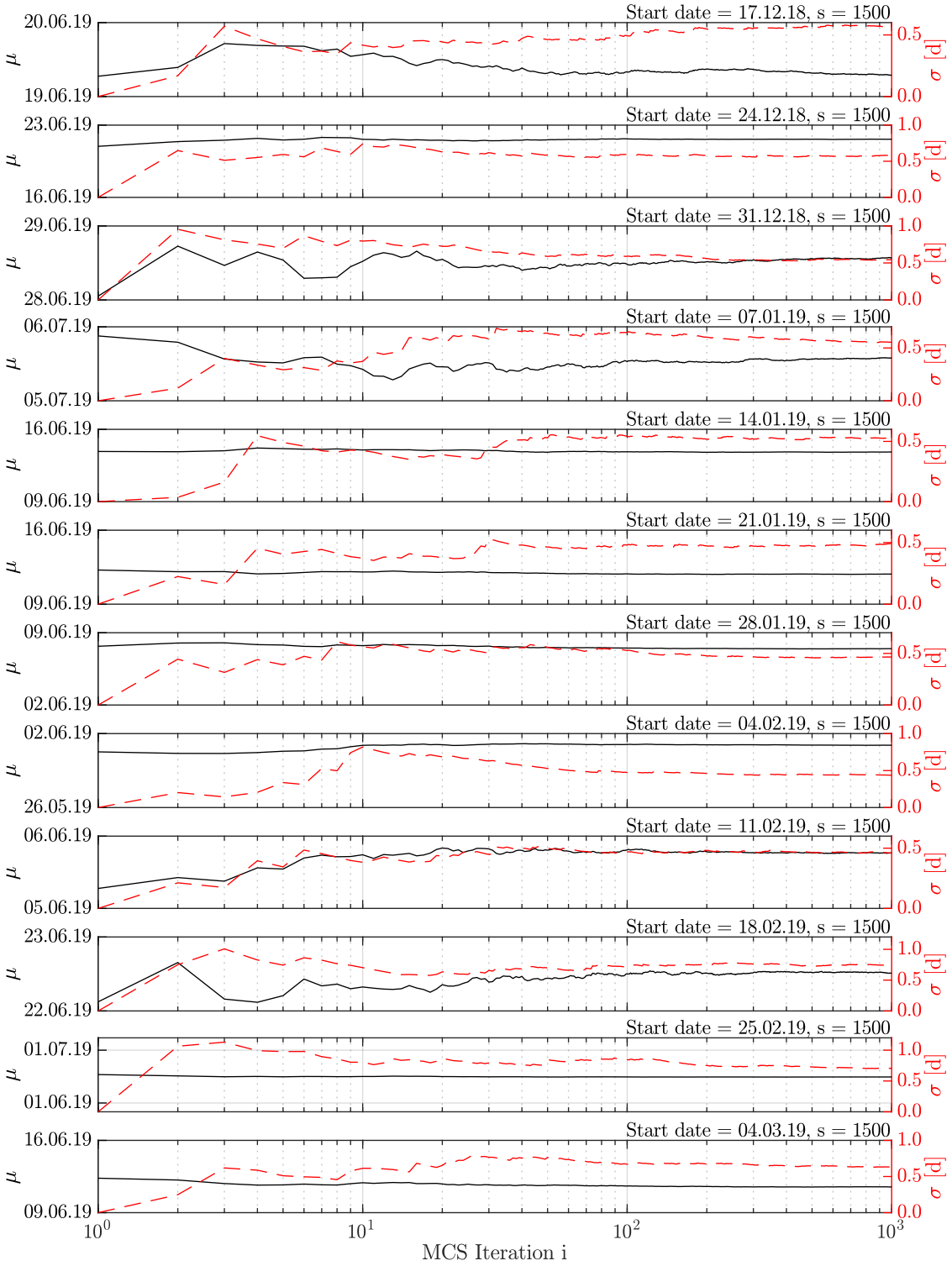


Figure C.13: Project Time Forecast of a MCS with sample size s = 1,500 boreholes and 1,000 iterations (part 1). Simulated are all AC units in tunnel tube 258H from 01.01.2018 to 15.05.2019. The mean value $\mu_i(p_i)$ (black) and the standard deviation $\sigma_i(p_i)$ (red, dotted) of the calculated project completion dates $p_i = [p_1, p_2, \ldots, p_i]$ are shown.

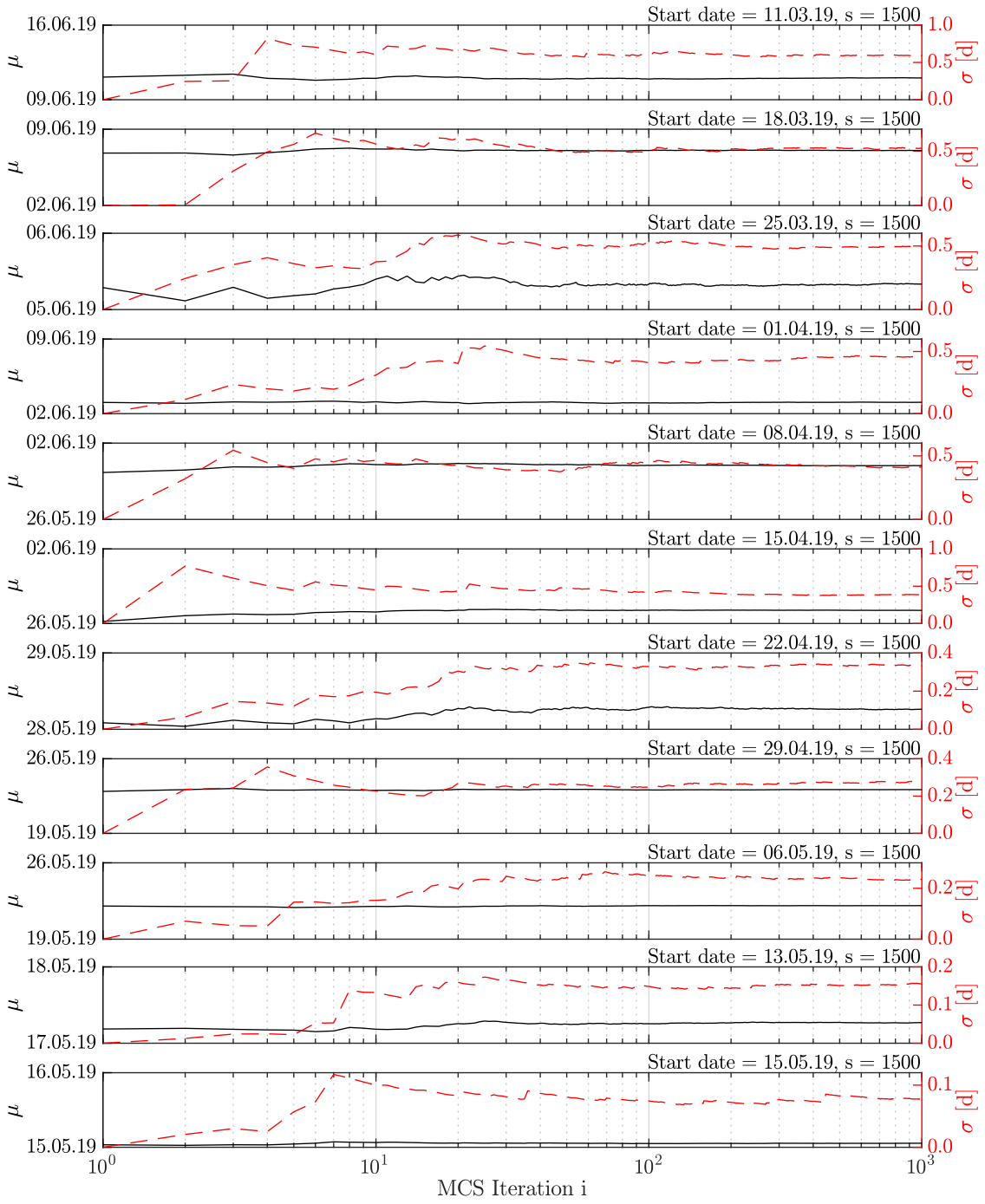


Figure C.14: Project Time Forecast of a MCS with sample size s = 1,500 boreholes and 1,000 iterations (part 2). Simulated are all AC units in tunnel tube 258H from 01.01.2018 to 15.05.2019. The mean value $\mu_i(p_i)$ (black) and the standard deviation $\sigma_i(p_i)$ (red, dotted) of the calculated project completion dates $p_i = [p_1, p_2, \ldots, p_i]$ are shown.

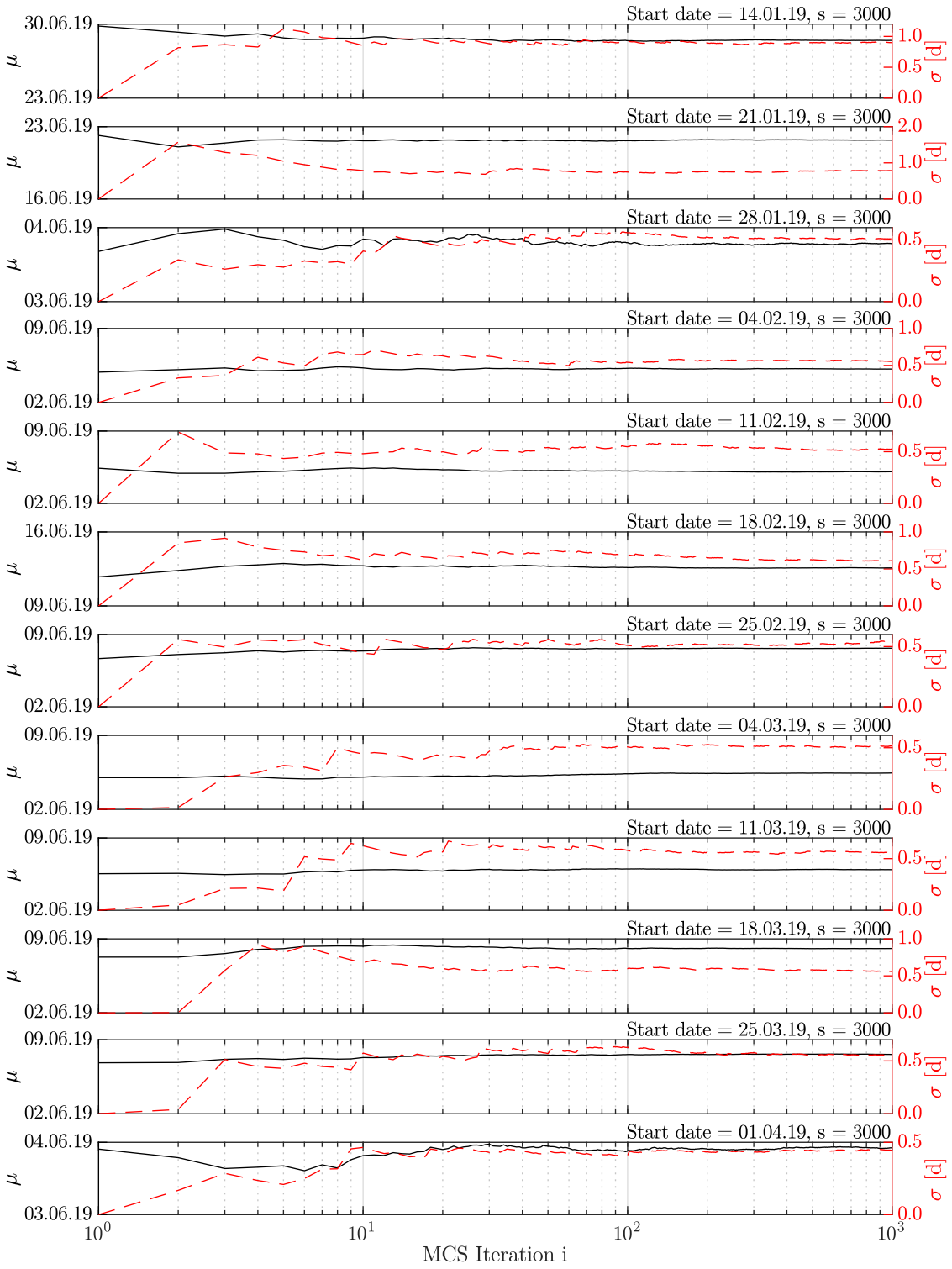


Figure C.15: Project Time Forecast of a MCS with sample size s = 3,000 boreholes and 1,000 iterations (part 1). Simulated are all AC units in tunnel tube 258H from 01.01.2018 to 15.05.2019. The mean value $\mu_i(p_i)$ (black) and the standard deviation $\sigma_i(p_i)$ (red, dotted) of the calculated project completion dates $p_i = [p_1, p_2, \ldots, p_i]$ are shown.

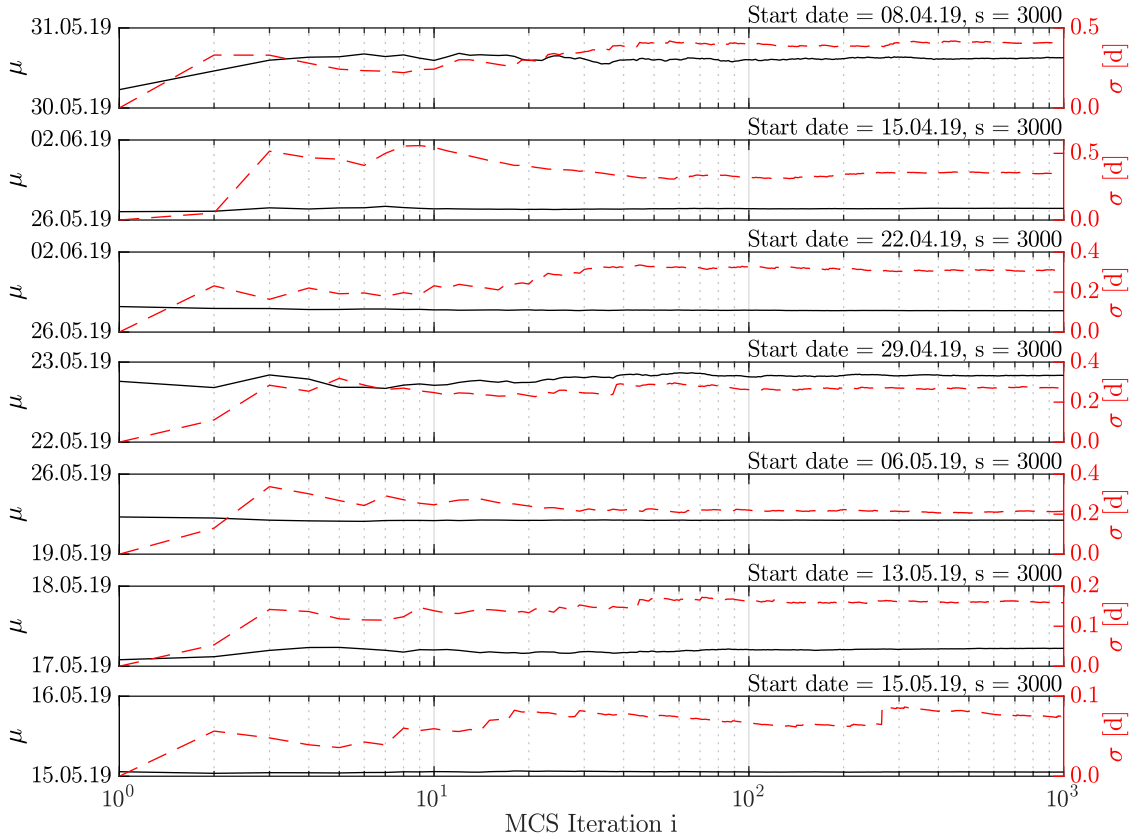


Figure C.16: Project Time Forecast of a MCS with sample size s = 3,000 boreholes and 1,000 iterations (part 2). Simulated are all AC units in tunnel tube 258H from 01.01.2018 to 15.05.2019. The mean value $\mu_i(p_i)$ (black) and the standard deviation $\sigma_i(p_i)$ (red, dotted) of the calculated project completion dates $p_i = [p_1, p_2, \ldots, p_i]$ are shown.

C.4 MSPS1 Result Diagrams of Total Project Time Forecast

The result diagrams presented in this section supplement those shown in section 6.3.3. Each sub-section shows the result diagrams of a different combination of AC injection units (GEA 1, GEA 2, and GEA 3) and either the 251H or 258H tunnel tube. Lines in black show the forecast. The red dashed line represents the measured values. For better readability, the scale of the abscissa of plots showing the deviation in percent has been cut off if values exceed 50 percent.

C.4.1 Total Project Time Forecast for Tube 251H, GEA1

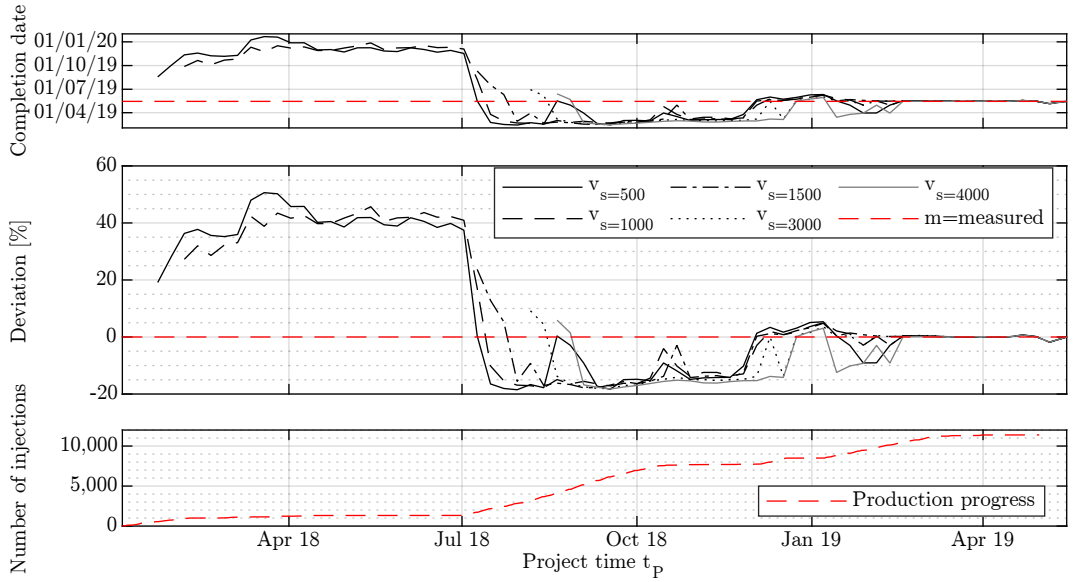


Figure C.17: Results, tunnel: 251H, machine: GEA1, timeframe: 01.01.18-15.05.19, method: cdf/markov.

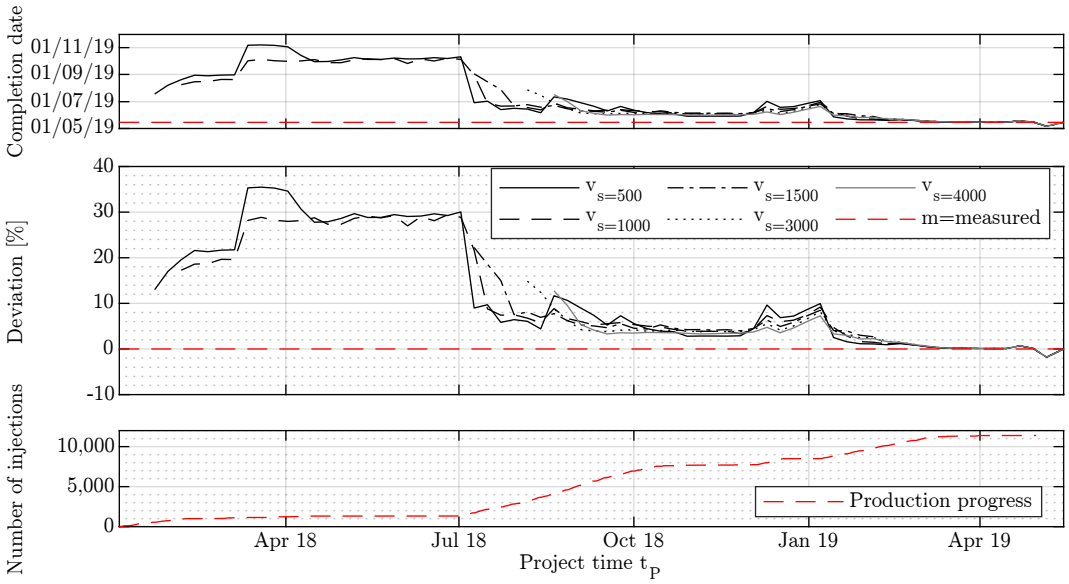


Figure C.18: Results, tunnel: 251H, machine: GEA1, timeframe: 01.01.18-15.05.19, method: mean/2Inj.

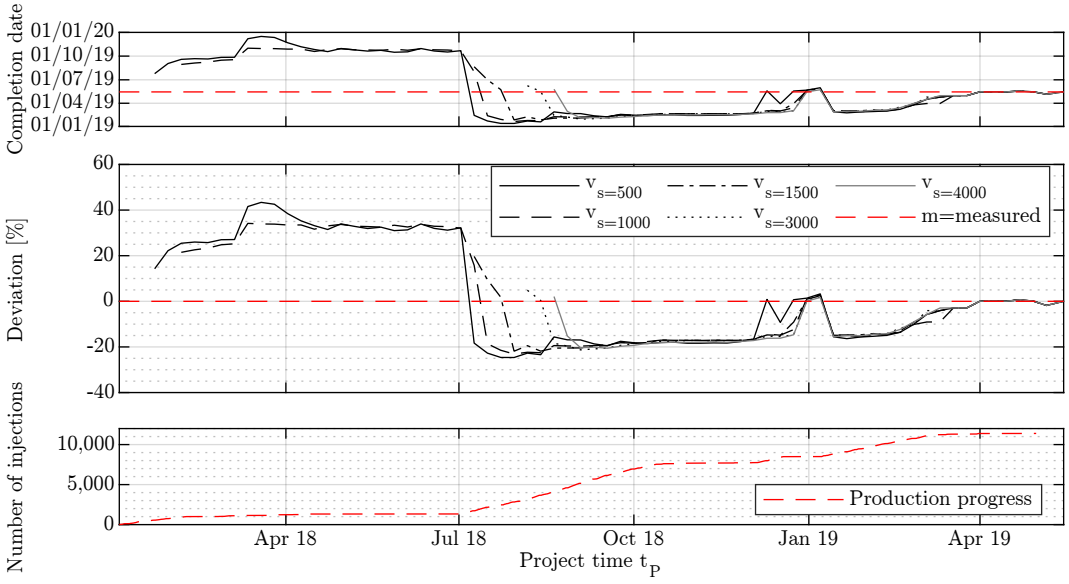


Figure C.19: Results, tunnel: 251H, machine: GEA1, timeframe: 01.01.18-15.05.19, method: mean/markov.

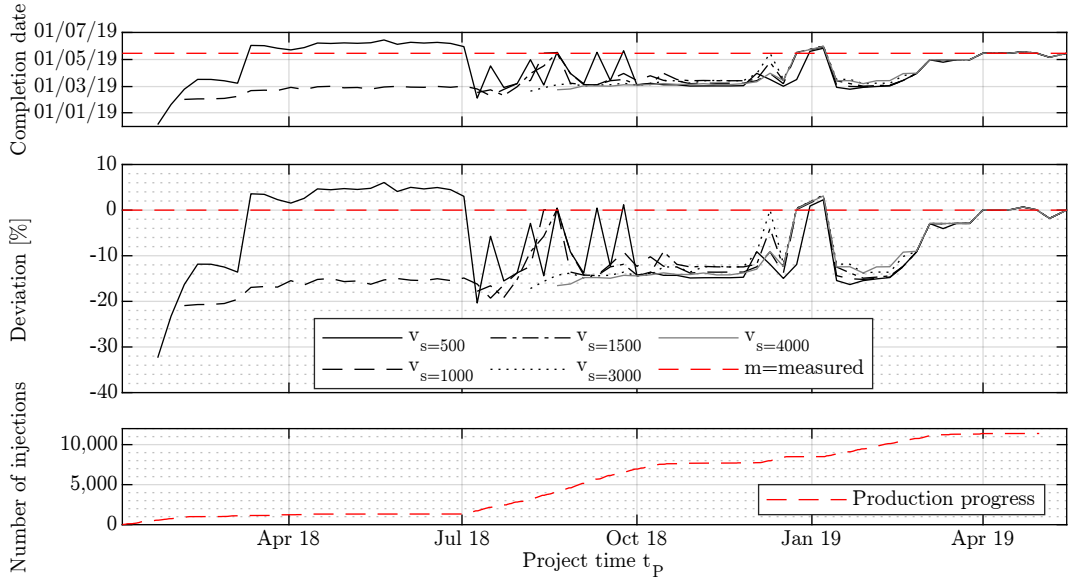


Figure C.20: Results, tunnel: 251H, machine: GEA1, timeframe: 01.01.18-15.05.19, method: median/2Inj.

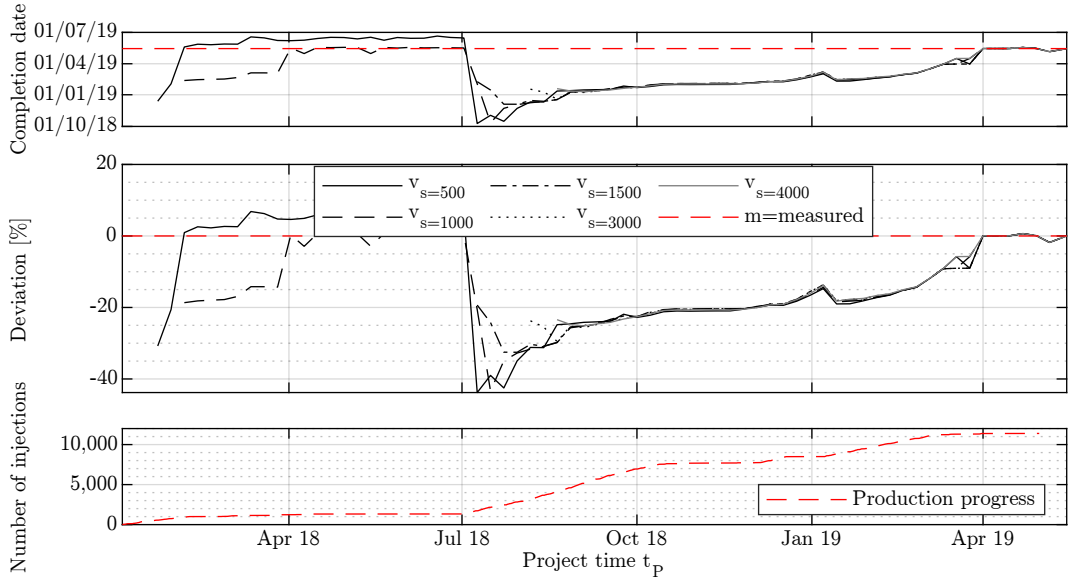


Figure C.21: Results, tunnel: 251H, machine: GEA1, timeframe: 01.01.18-15.05.19, method: median/markov.

C.4.2 Total Project Time Forecast for Tube 251H, GEA2

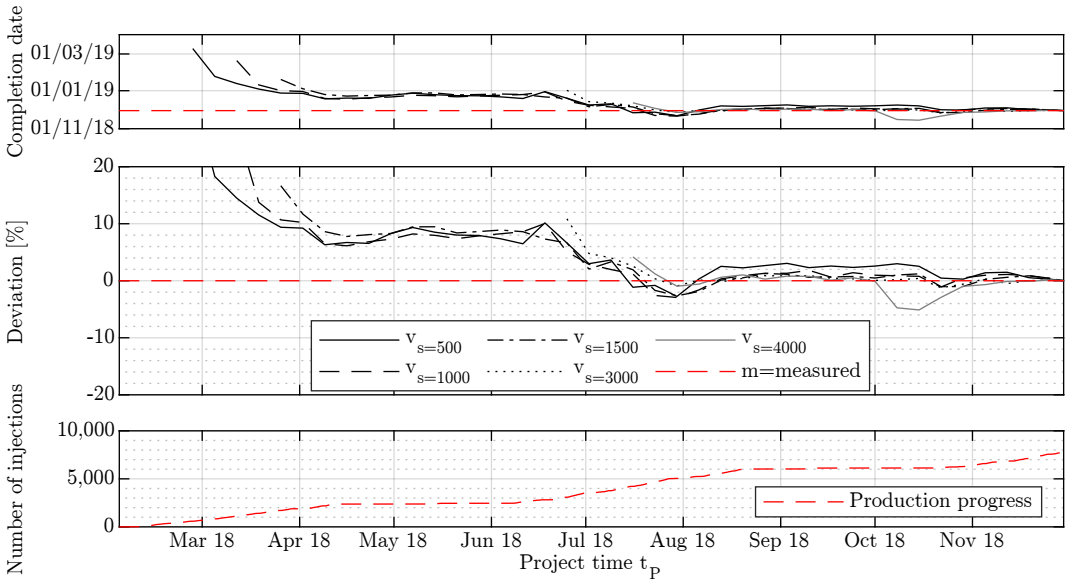


Figure C.22: Results, tunnel: 251H, machine: GEA2, timeframe: 01.01.18-30.11.18, method: cdf/markov.

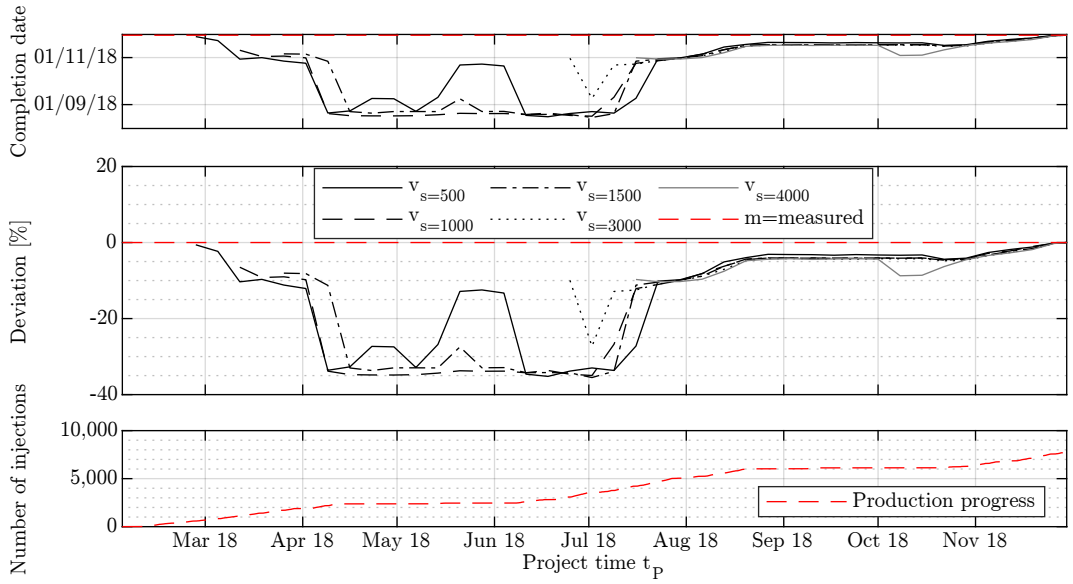


Figure C.23: Results, tunnel: 251H, machine: GEA2, timeframe: 01.01.18-30.11.18, method: mean/2Inj.

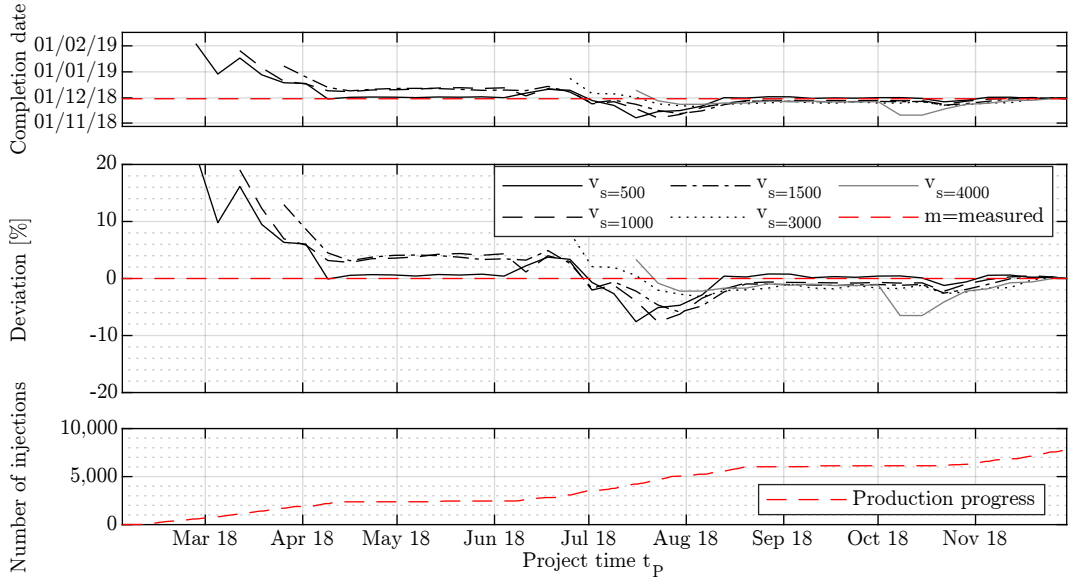


Figure C.24: Results, tunnel: 251H, machine: GEA2, timeframe: 01.01.18-30.11.18, method: mean/markov.

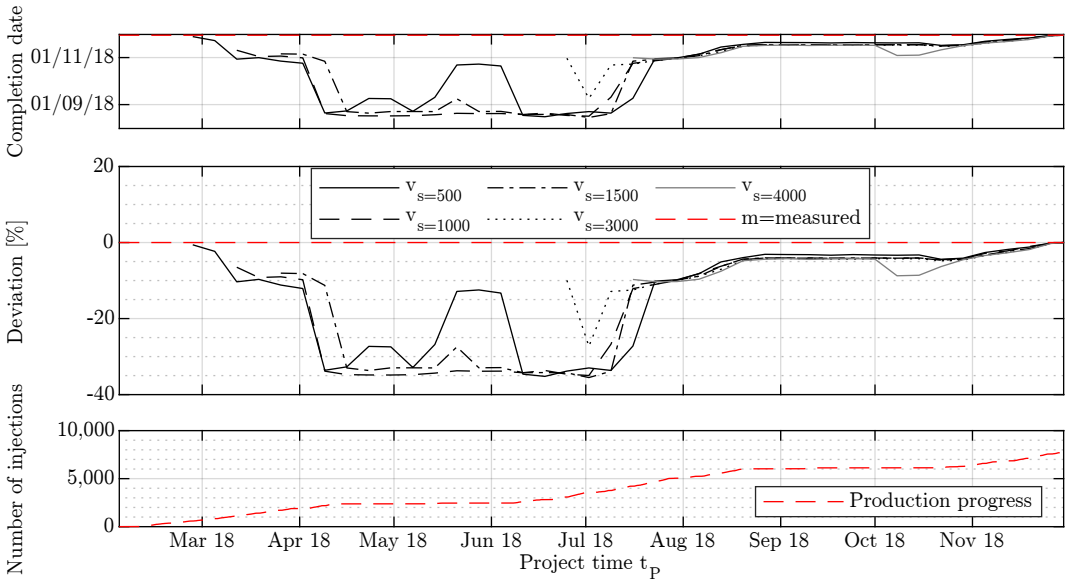


Figure C.25: Results, tunnel: 251H, machine: GEA2, timeframe: 01.01.18-30.11.18, method: median/2Inj.

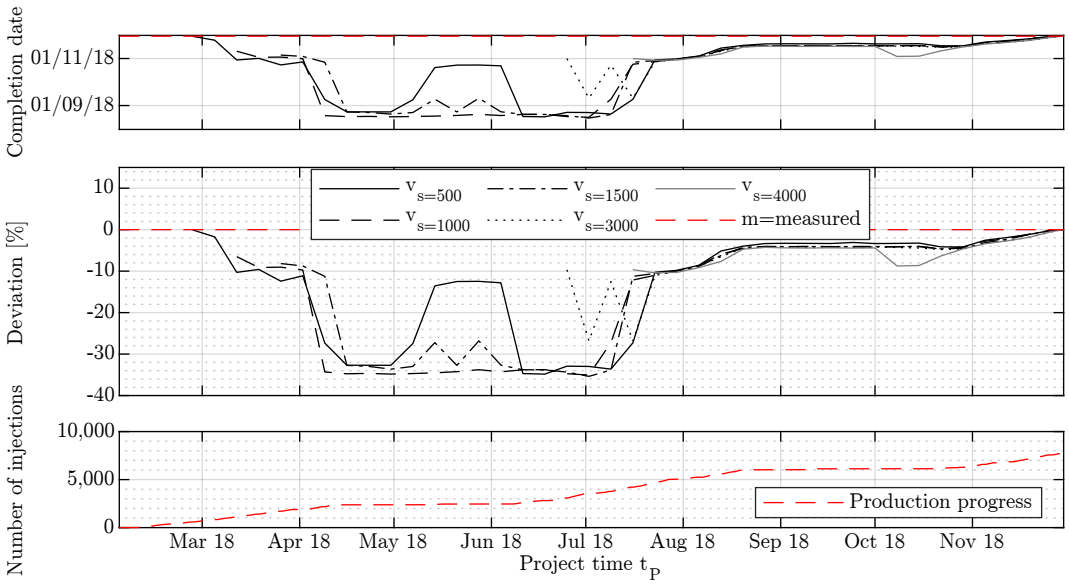


Figure C.26: Results, tunnel: 251H, machine: GEA2, timeframe: 01.01.18-30.11.18, method: median/markov.

C.4.3 Total Project Time Forecast for Tube 251H, GEA3

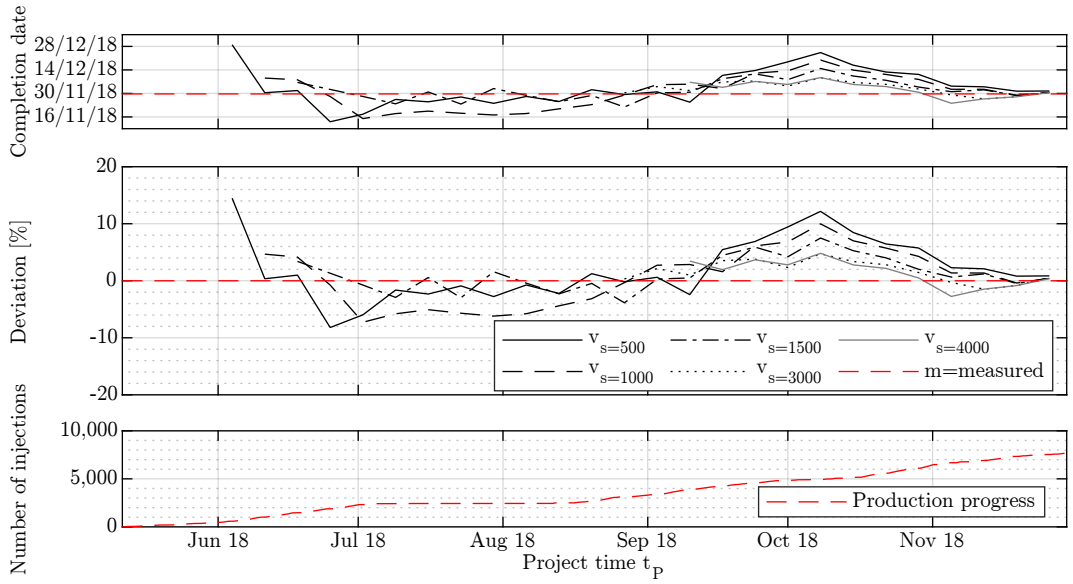


Figure C.27: Results, tunnel: 251H, machine: GEA3, timeframe: 01.01.18-30.11.18, method: cdf/markov.

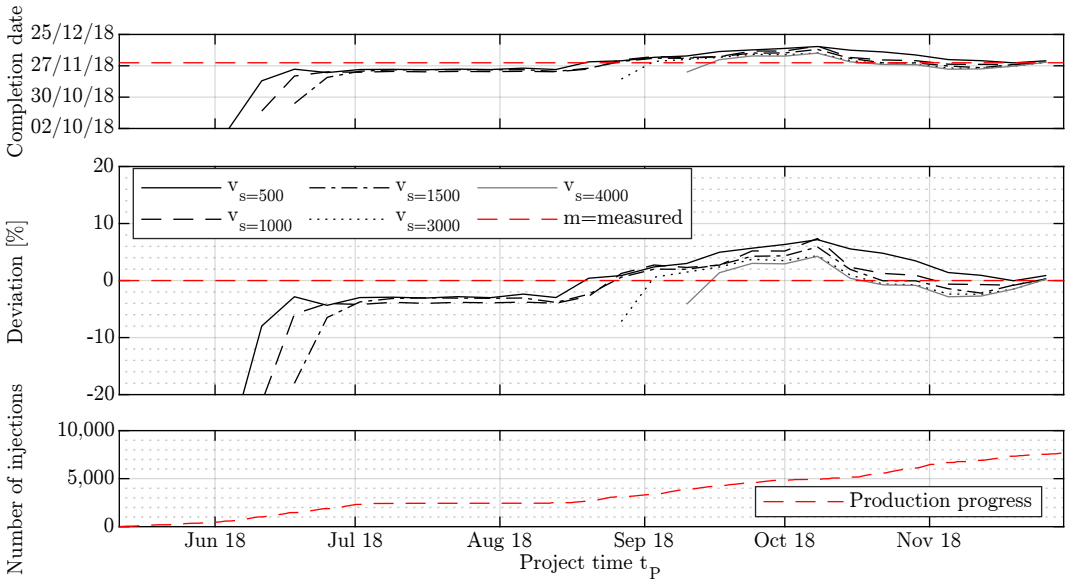


Figure C.28: Results, tunnel: 251H, machine: GEA3, timeframe: 01.01.18-30.11.18, method: mean/2Inj.

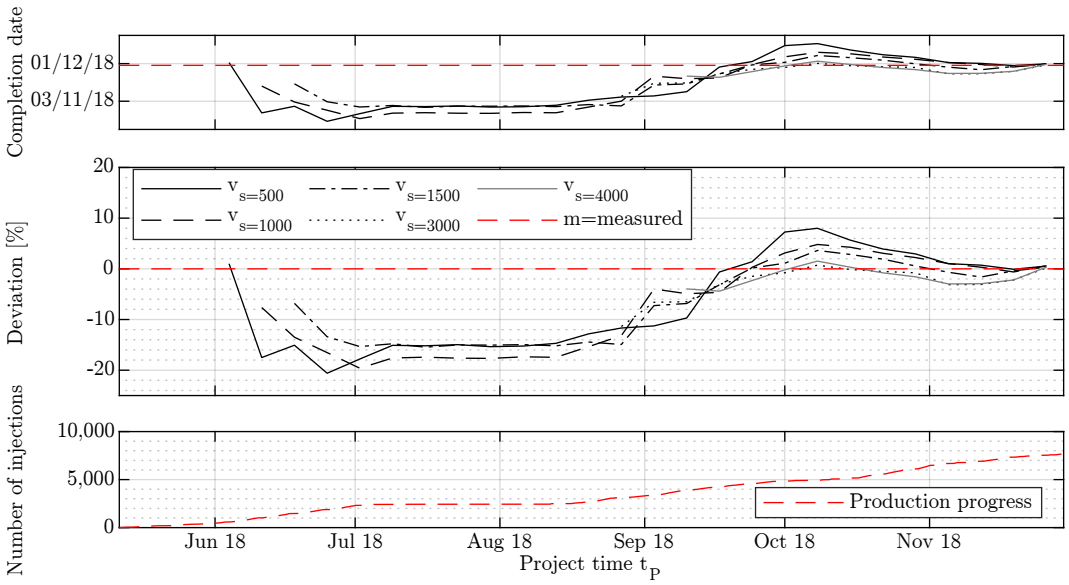


Figure C.29: Results, tunnel: 251H, machine: GEA3, timeframe: 01.01.18-30.11.18, method: mean/markov.

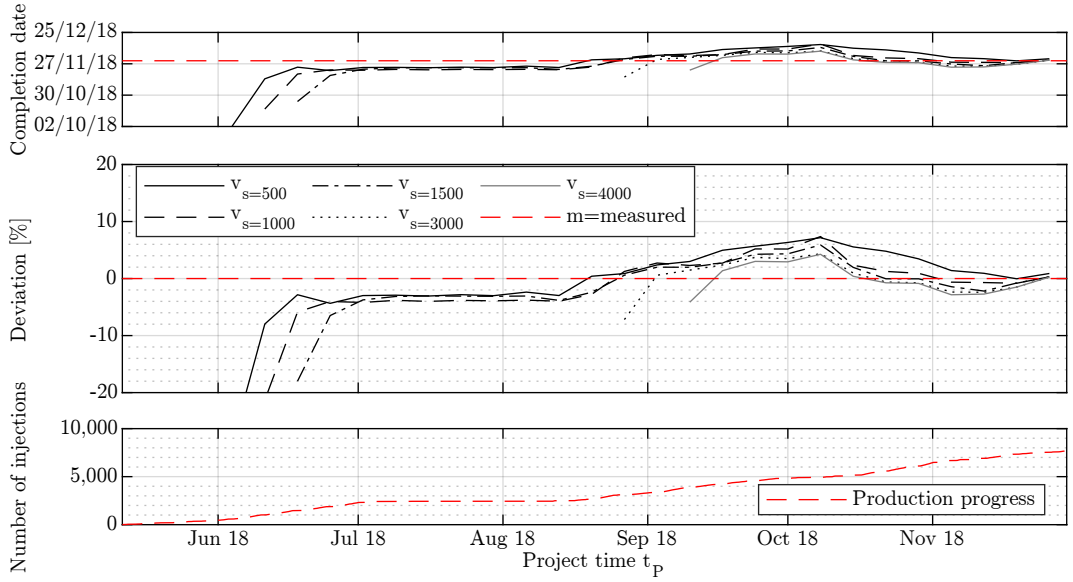


Figure C.30: Results, tunnel: 251H, machine: GEA3, timeframe: 01.01.18-30.11.18, method: median/2Inj.

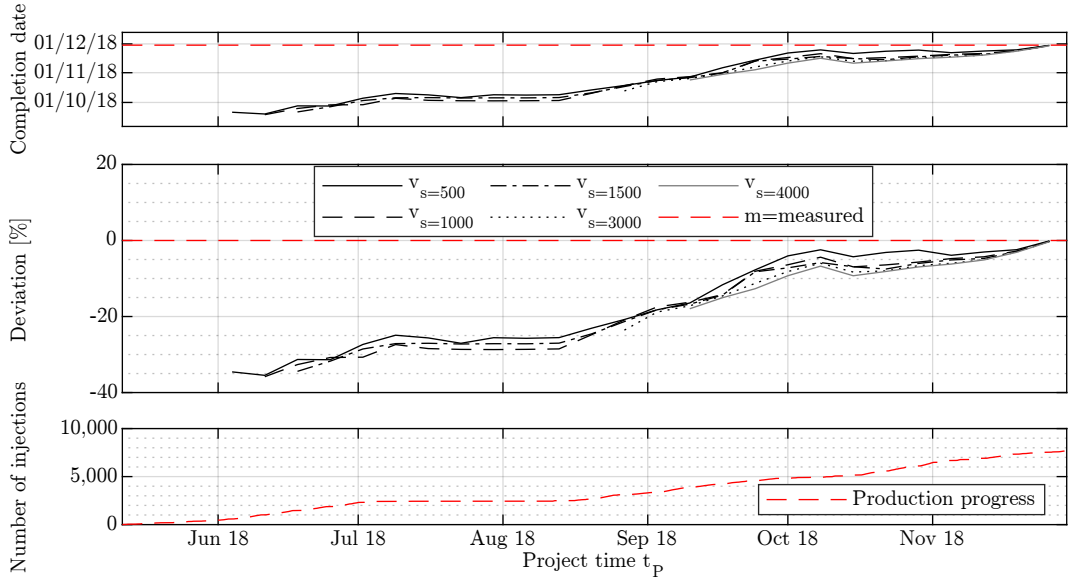


Figure C.31: Results, tunnel: 251H, machine: GEA3, timeframe: 01.01.18-30.11.18, method: median/markov.

C.4.4 Total Project Time Forecast for Tube 251H, GEA1-3

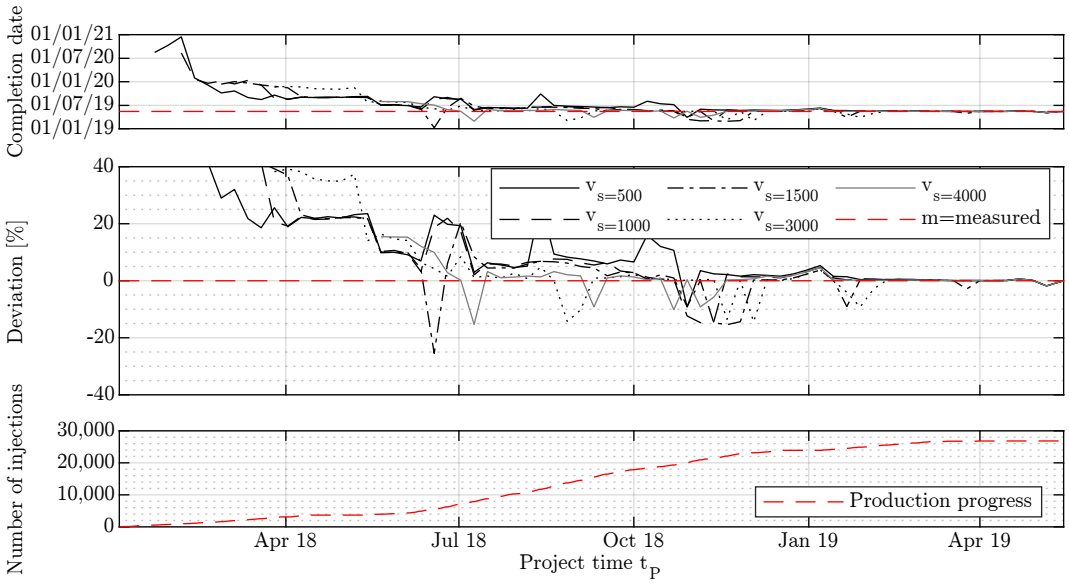


Figure C.32: Results, tunnel: 251H, machine: GEA1-3, timeframe: 01.01.18-15.05.19, method: cdf/markov.

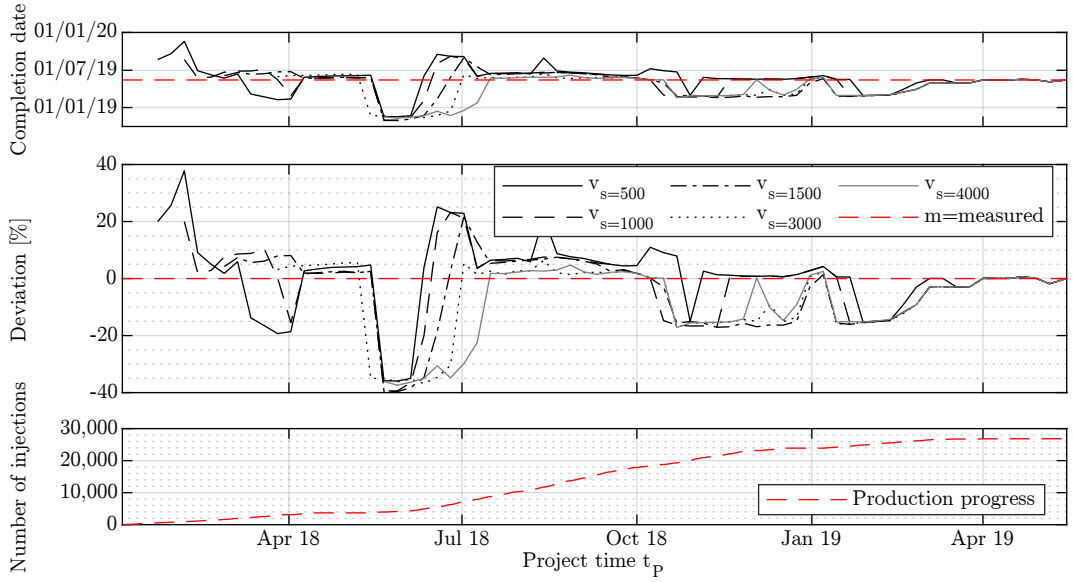


Figure C.33: Results, tunnel: 251H, machine: GEA1-3, timeframe: 01.01.18-15.05.19, method: mean/2Inj.

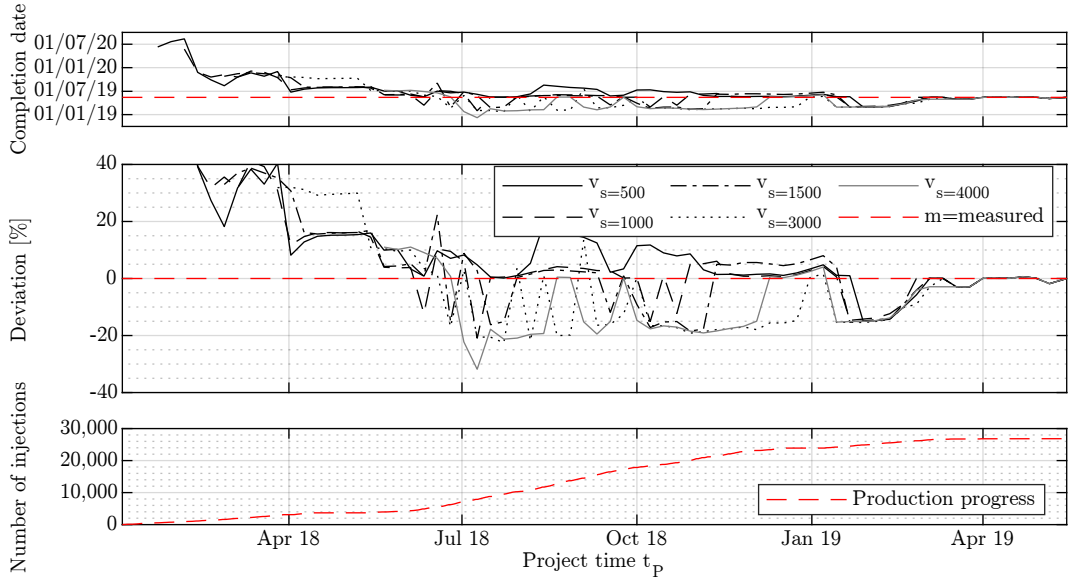


Figure C.34: Results, tunnel: 251H, machine: GEA1-3, timeframe: 01.01.18-15.05.19, method: mean/markov.

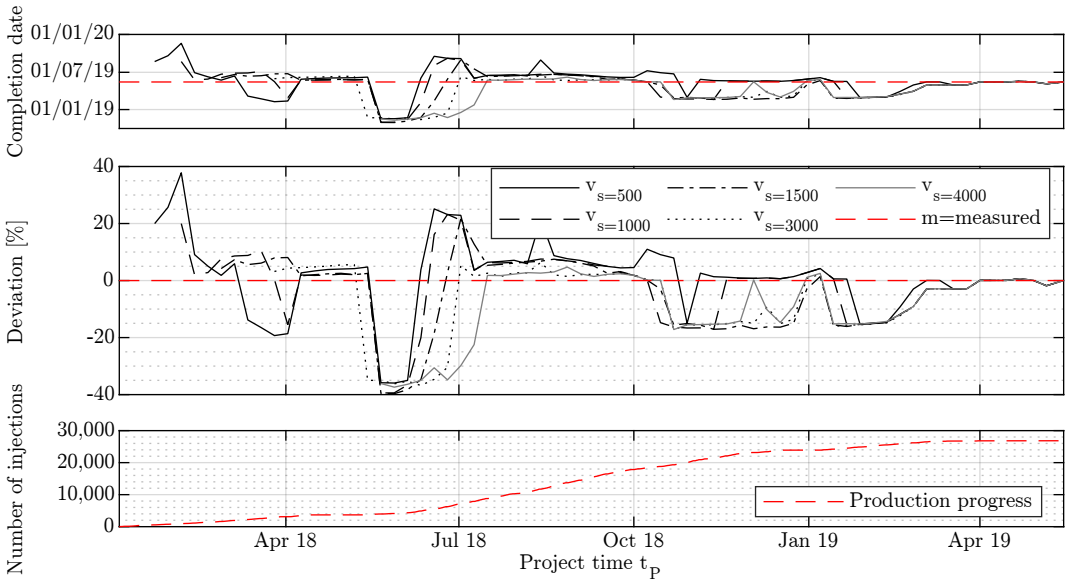


Figure C.35: Results, tunnel: 251H, machine: GEA1-3, timeframe: 01.01.18-15.05.19, method: median/2Inj.

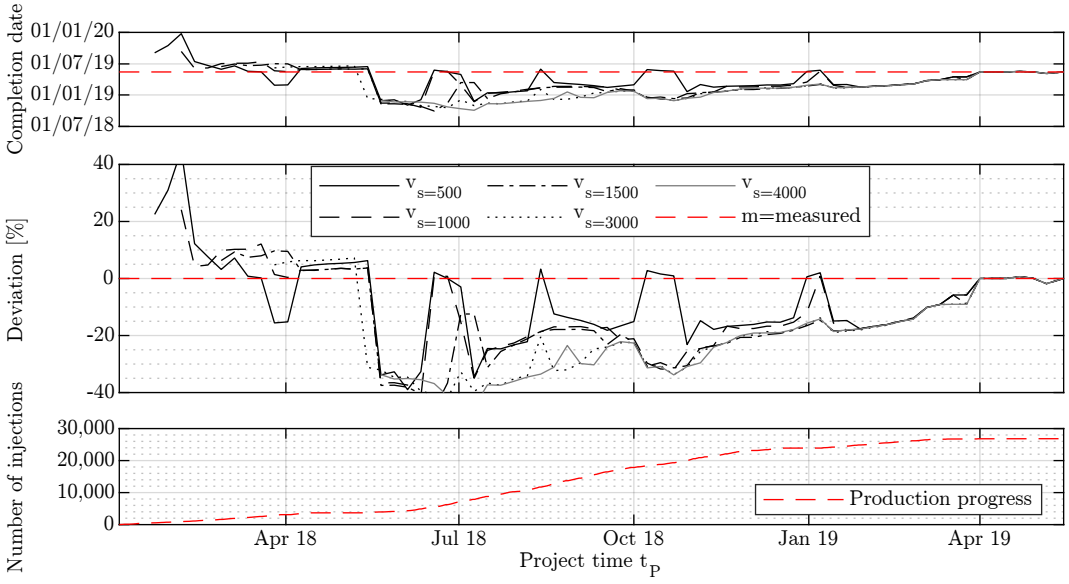


Figure C.36: Results, tunnel: 251H, machine: GEA1-3, timeframe: 01.01.18-15.05.19, method: median/markov.

C.4.5 Total Project Time Forecast for Tube 258H, GEA2

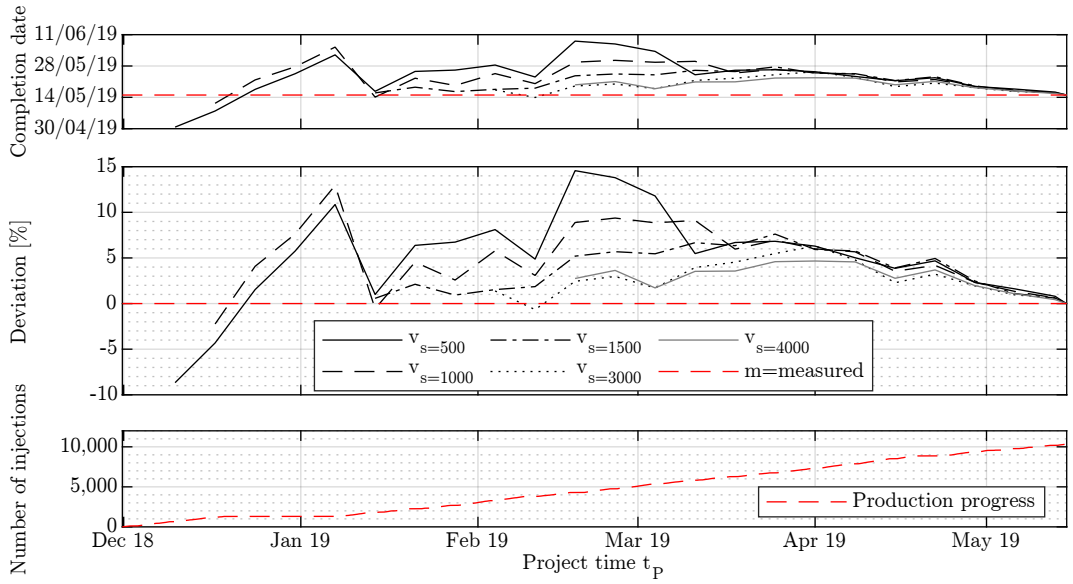


Figure C.37: Results, tunnel: 258H, machine: GEA2, timeframe: 01.01.18-15.05.19, method: cdf/markov.

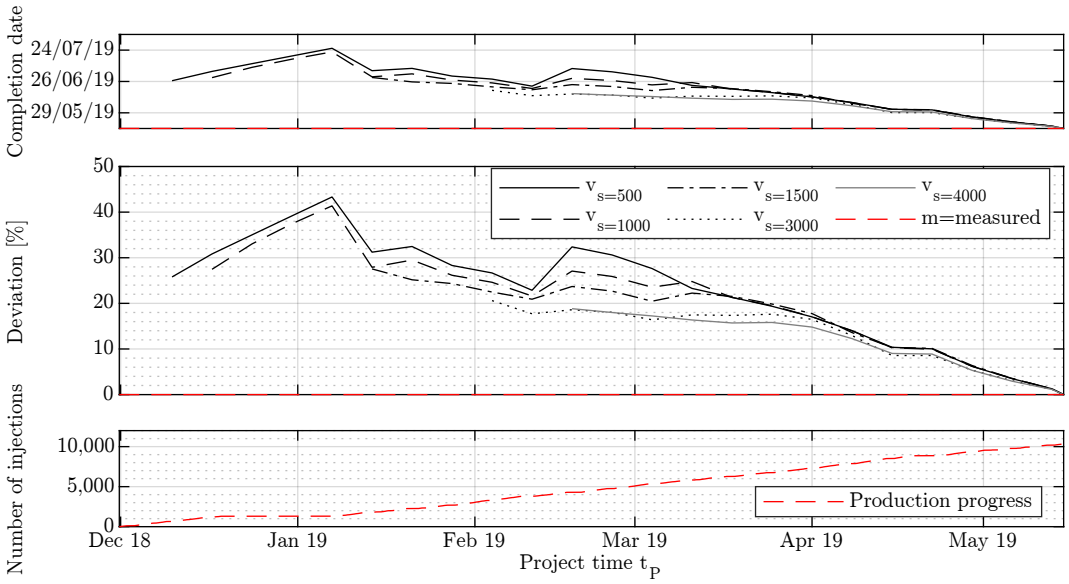


Figure C.38: Results, tunnel: 258H, machine: GEA2, timeframe: 01.01.18-15.05.19, method: mean/2Inj.

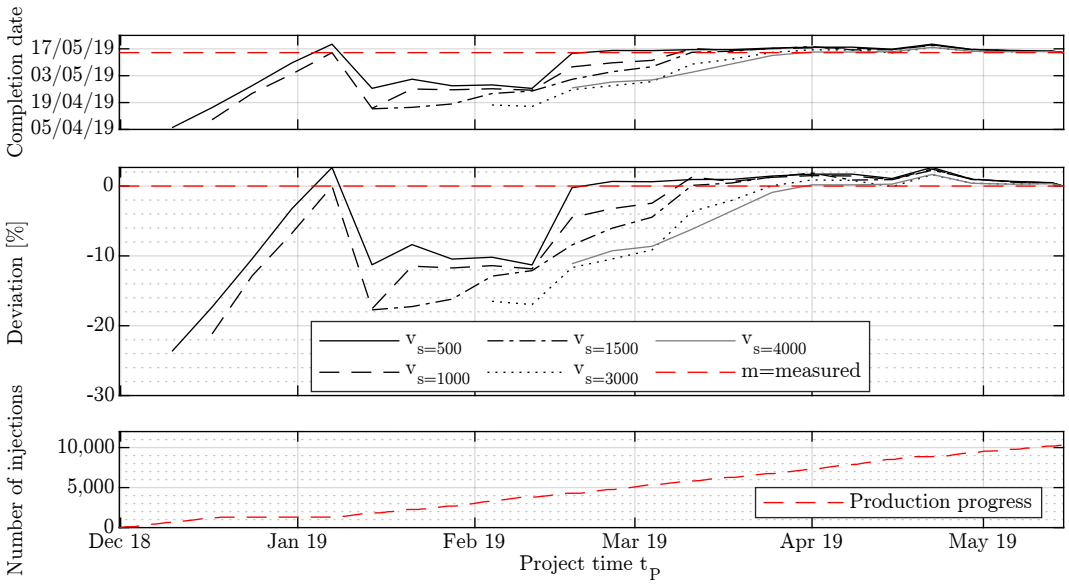


Figure C.39: Results, tunnel: 258H, machine: GEA2, timeframe: 01.01.18-15.05.19, method: mean/markov.

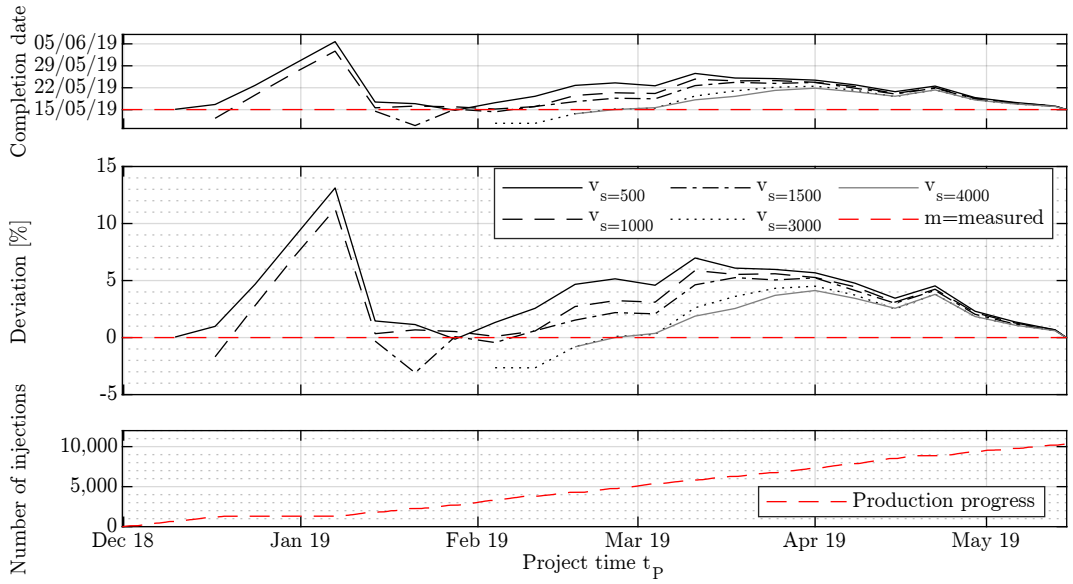


Figure C.40: Results, tunnel: 258H, machine: GEA2, timeframe: 01.01.18-15.05.19, method: median/2Inj.

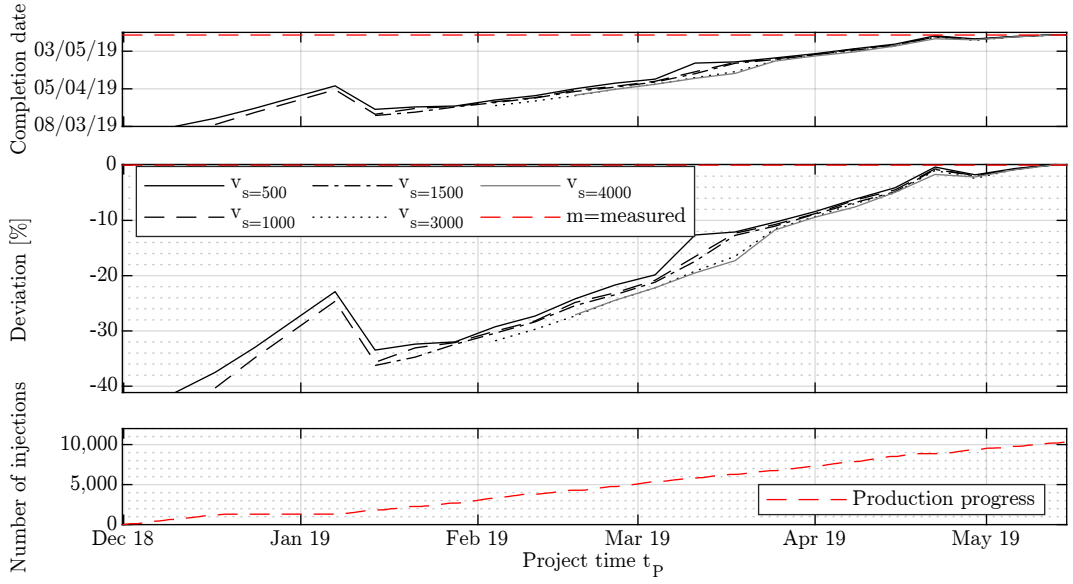


Figure C.41: Results, tunnel: 258H, machine: GEA2, timeframe: 01.01.18-15.05.19, method: median/markov.

C.4.6 Total Project Time Forecast for Tube 258H, GEA3

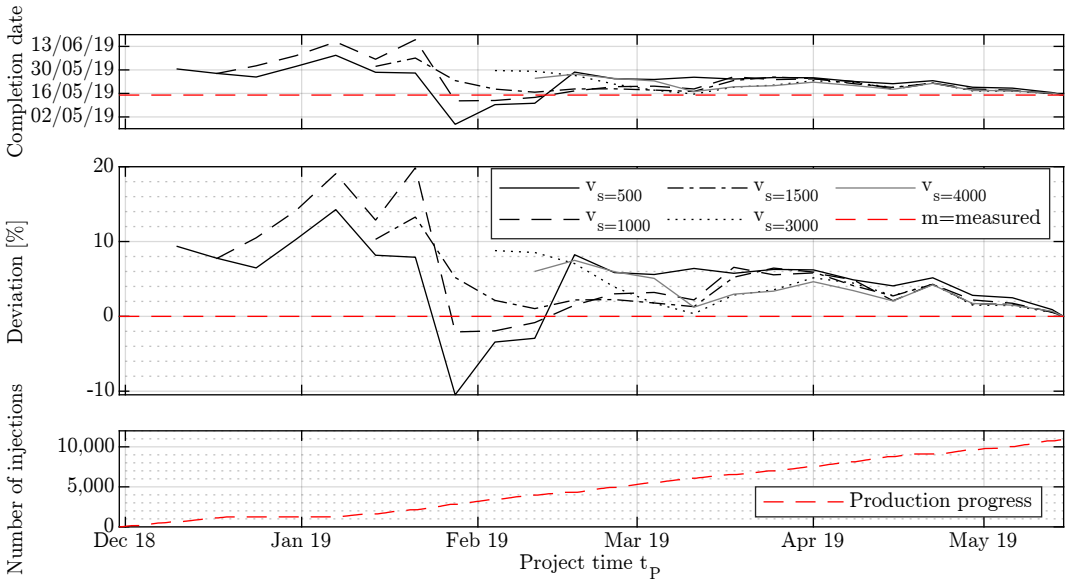


Figure C.42: Results, tunnel: 258H, machine: GEA3, timeframe: 01.01.18-15.05.19, method: cdf/markov.

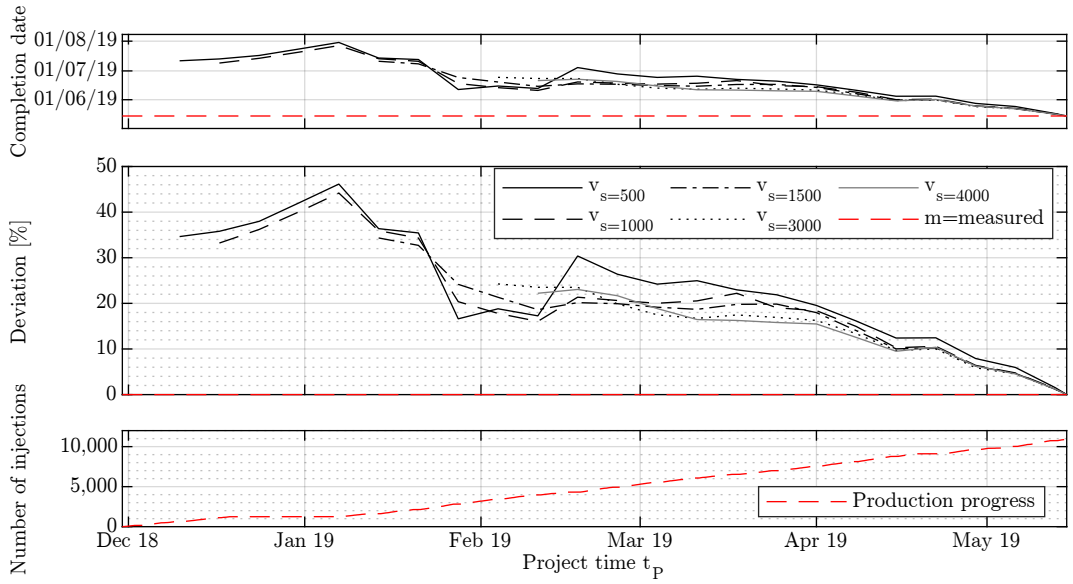


Figure C.43: Results, tunnel: 258H, machine: GEA3, timeframe: 01.01.18-15.05.19, method: mean/2Inj.

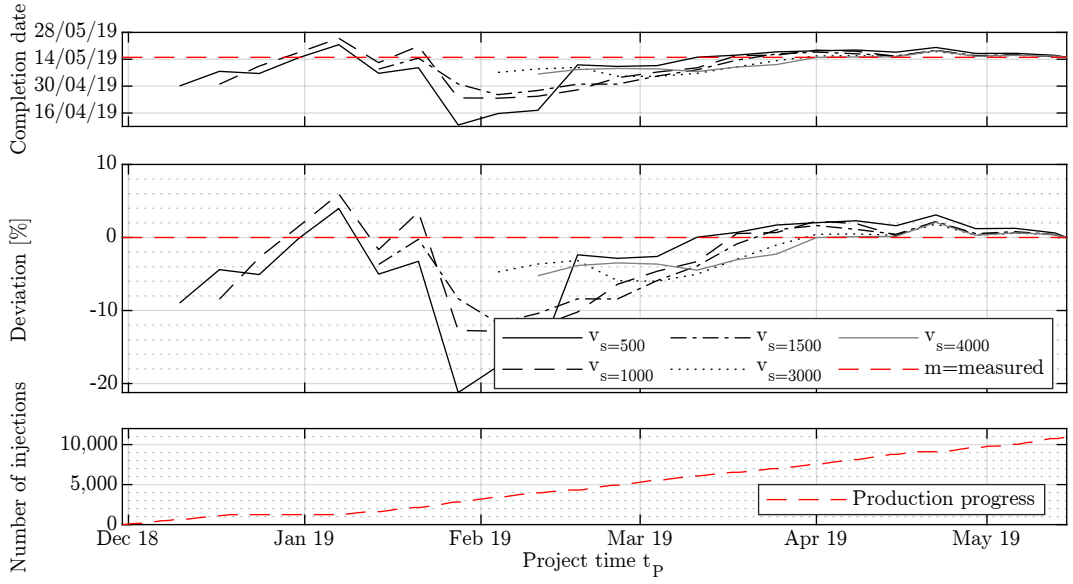


Figure C.44: Results, tunnel: 258H, machine: GEA3, timeframe: 01.01.18-15.05.19, method: mean/markov.

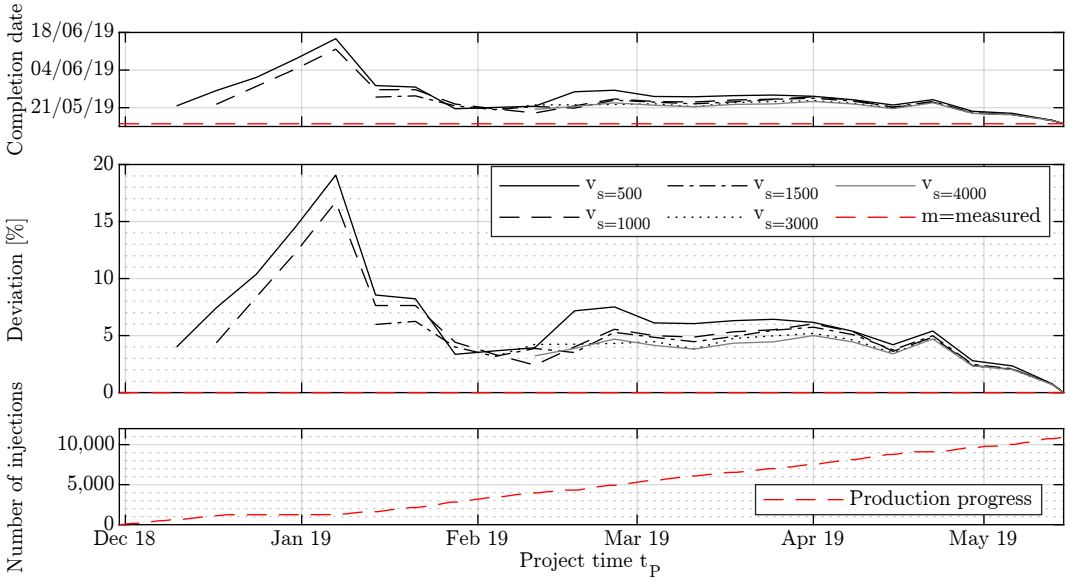


Figure C.45: Results, tunnel: 258H, machine: GEA3, timeframe: 01.01.18-15.05.19, method: median/2Inj.

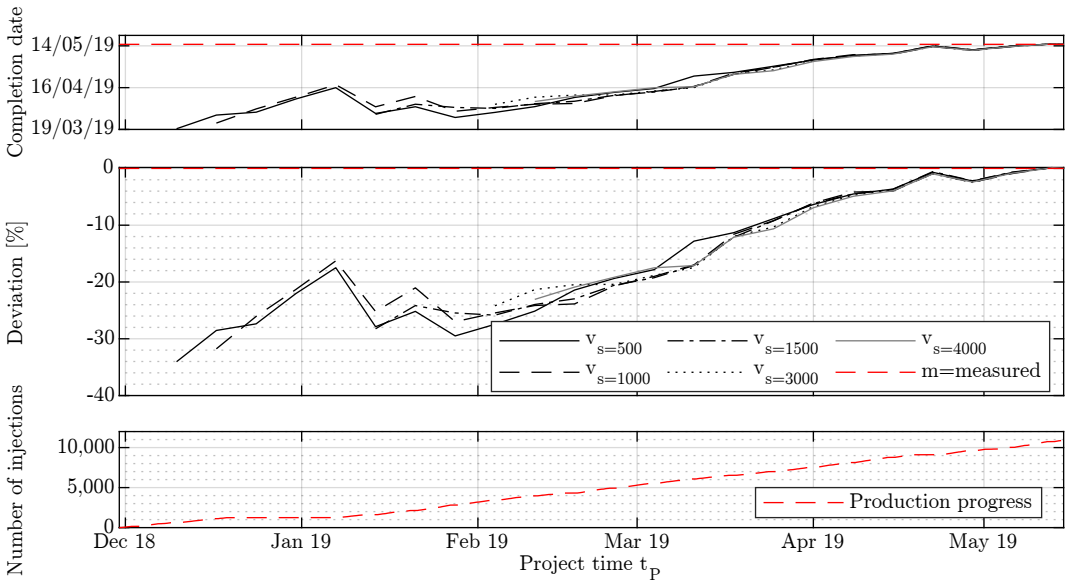


Figure C.46: Results, tunnel: 258H, machine: GEA3, timeframe: 01.01.18-15.05.19, method: median/markov.

C.4.7 Total Project Time Forecast for Tube 258H, GEA1-3

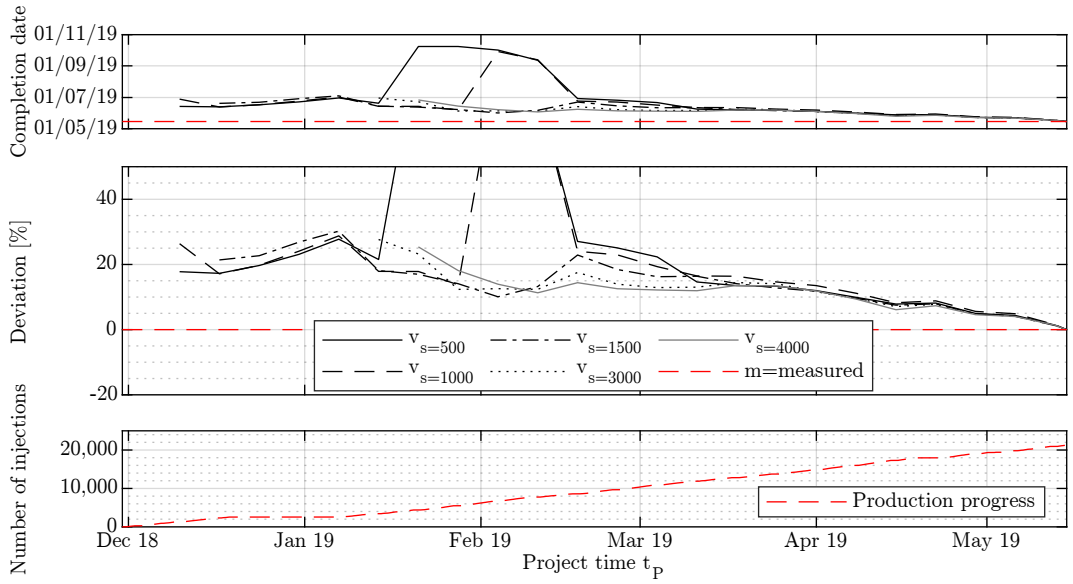


Figure C.47: Results, tunnel: 258H, machine: GEA1-3, timeframe: 01.01.18-15.05.19, method: cdf/markov.

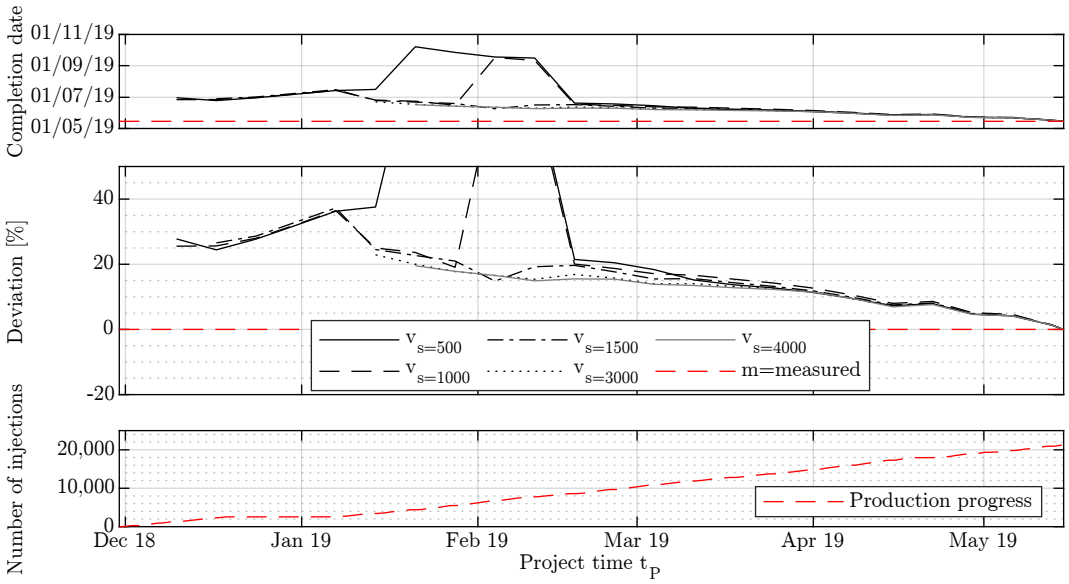


Figure C.48: Results, tunnel: 258H, machine: GEA1-3, timeframe: 01.01.18-15.05.19, method: mean/2Inj.

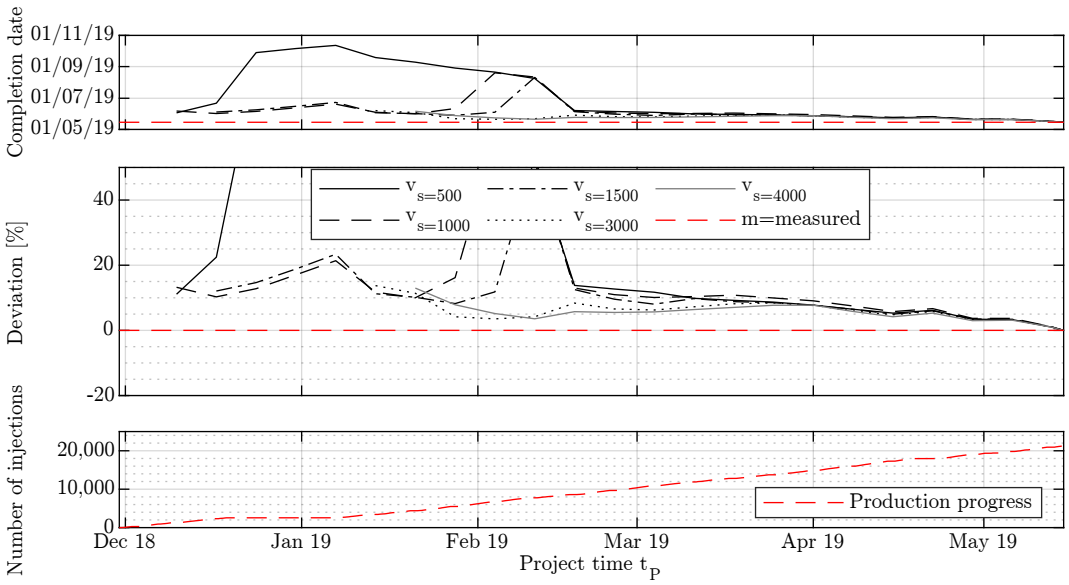


Figure C.49: Results, tunnel: 258H, machine: GEA1-3, timeframe: 01.01.18-15.05.19, method: mean/markov.

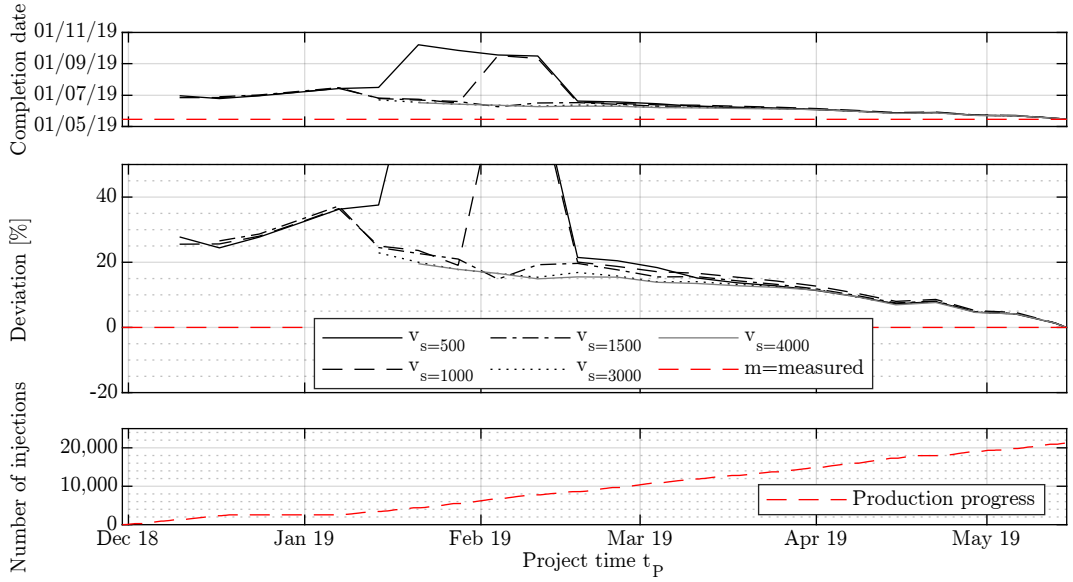


Figure C.50: Results, tunnel: 258H, machine: GEA1-3, timeframe: 01.01.18-15.05.19, method: median/2Inj.

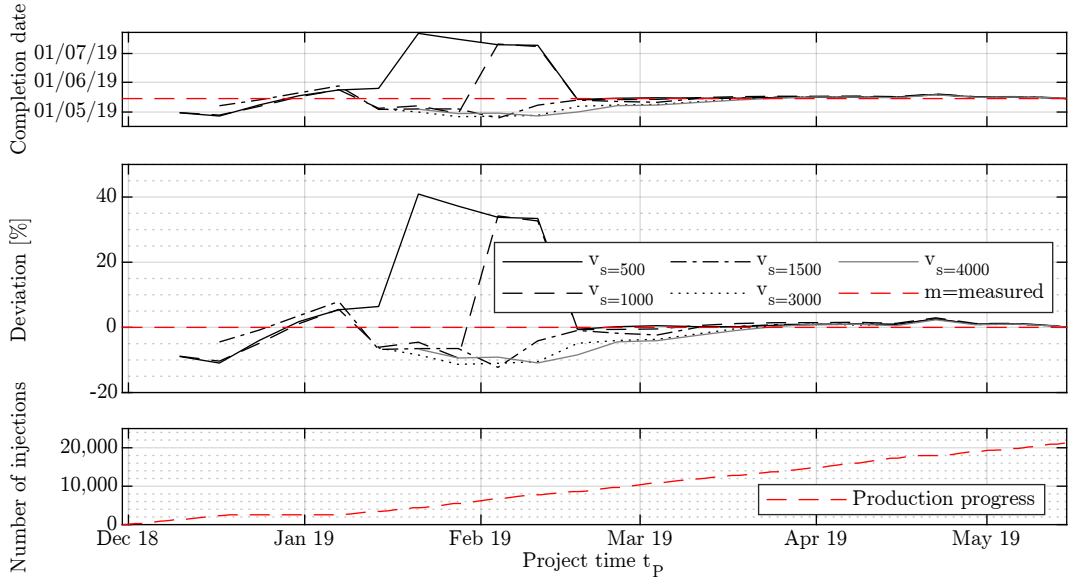


Figure C.51: Results, tunnel: 258H, machine: GEA1-3, timeframe: 01.01.18-15.05.19, method: median/markov.

C.5 MSPS1 Result Diagrams of Partial Project Time Forecast

The result diagrams presented in this section supplement those presented in section 6.3.3. The following sub-sections differ in the sample size s used for the forecast. Points in black show the forecast. The red dashed line represents the measured values.

C.5.1 Partial Time Forecast, Sample Size $s = 500$, Tube 258H, GEA1-3

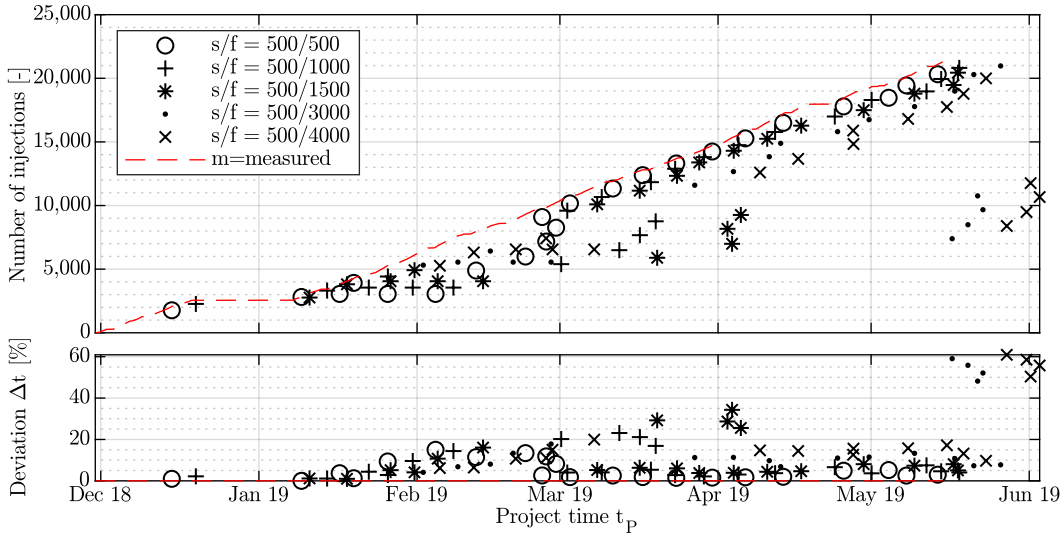


Figure C.52: Results, partial project time forecast, tunnel: 258H, machine: GEA1-3, method: cdf/markov, sample size $s = 500$.

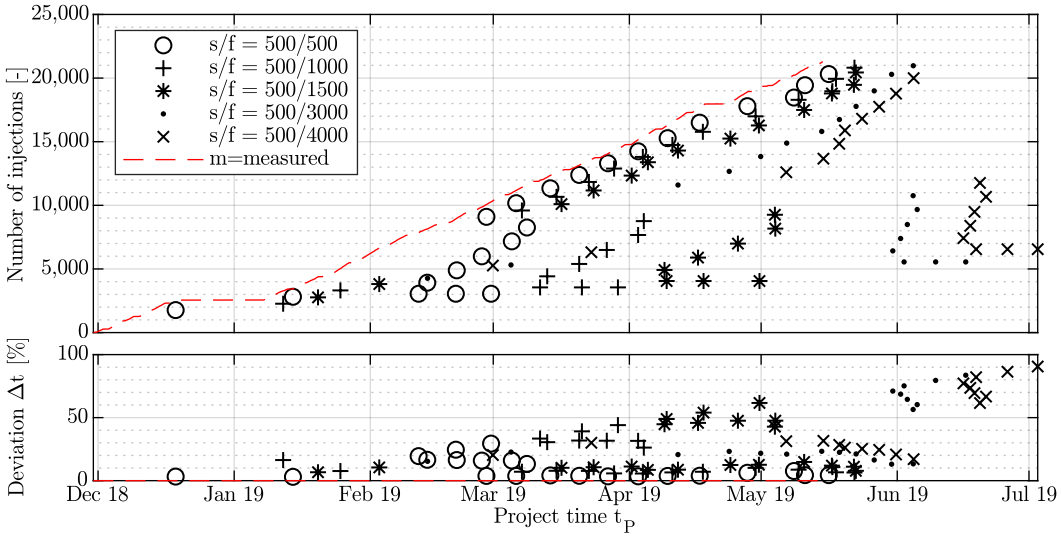


Figure C.53: Results, partial project time forecast, tunnel: 258H, machine: GEA1-3, method: mean/2Inj, sample size $s = 500$.

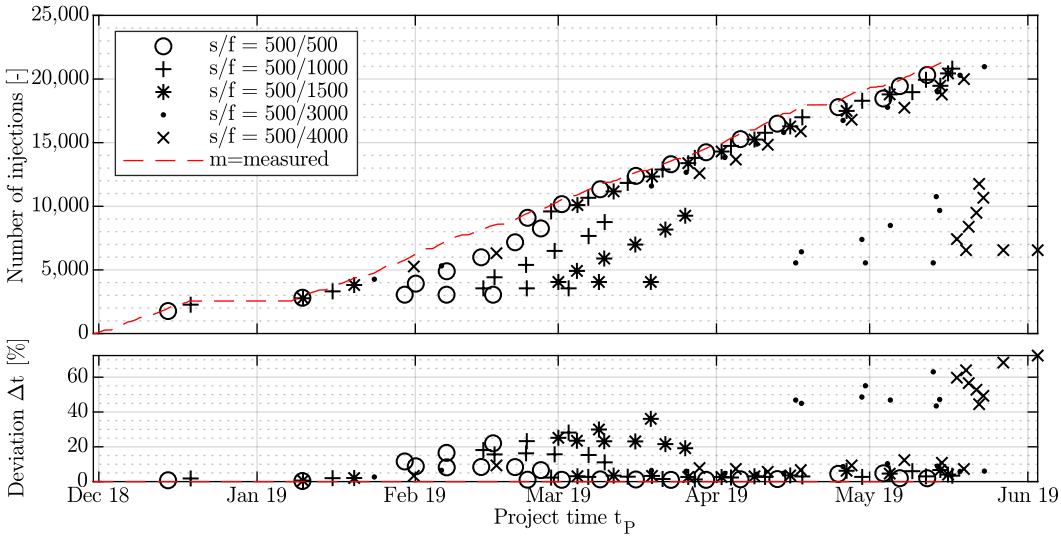


Figure C.54: Results, partial project time forecast, tunnel: 258H, machine: GEA1-3, method: mean/markov, sample size $s = 500$.

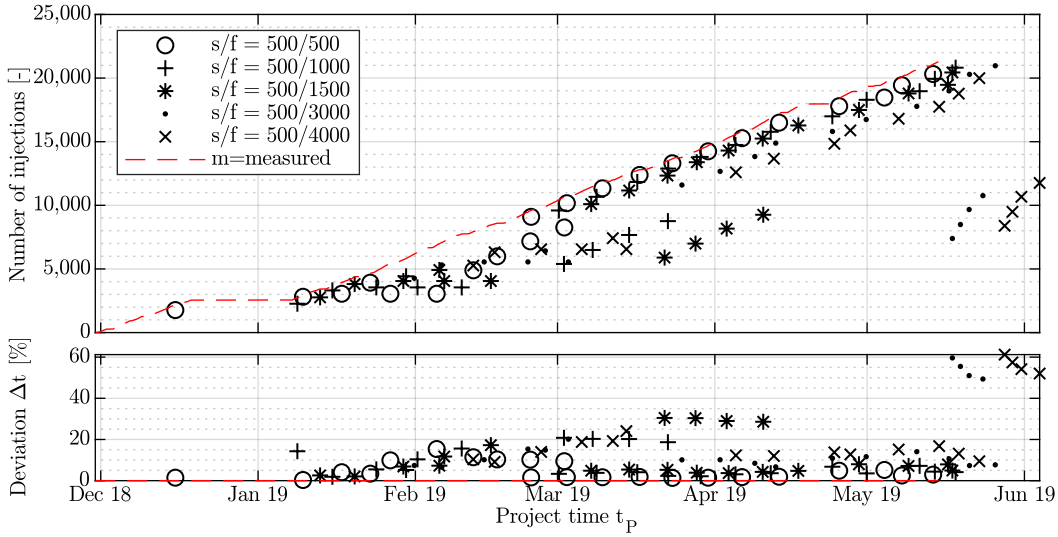


Figure C.55: Results, partial project time forecast, tunnel: 258H, machine: GEA1-3, method: median/2Inj, sample size $s = 500$.

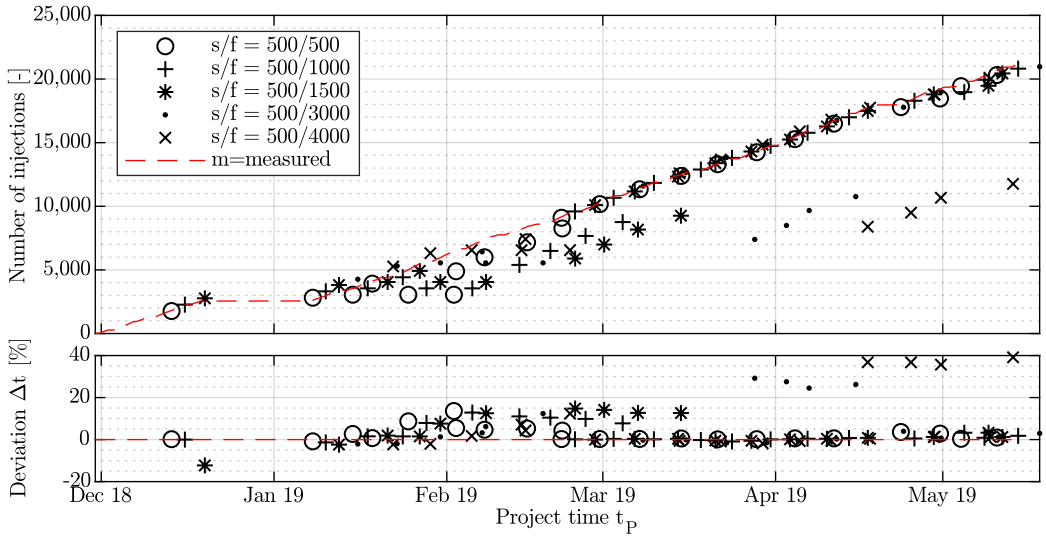


Figure C.56: Results, partial project time forecast, tunnel: 258H, machine: GEA1-3, method: median/markov, sample size $s = 500$.

C.5.2 Partial Time Forecast, Sample Size $s = 1000$, Tube 258H, GEA1-3

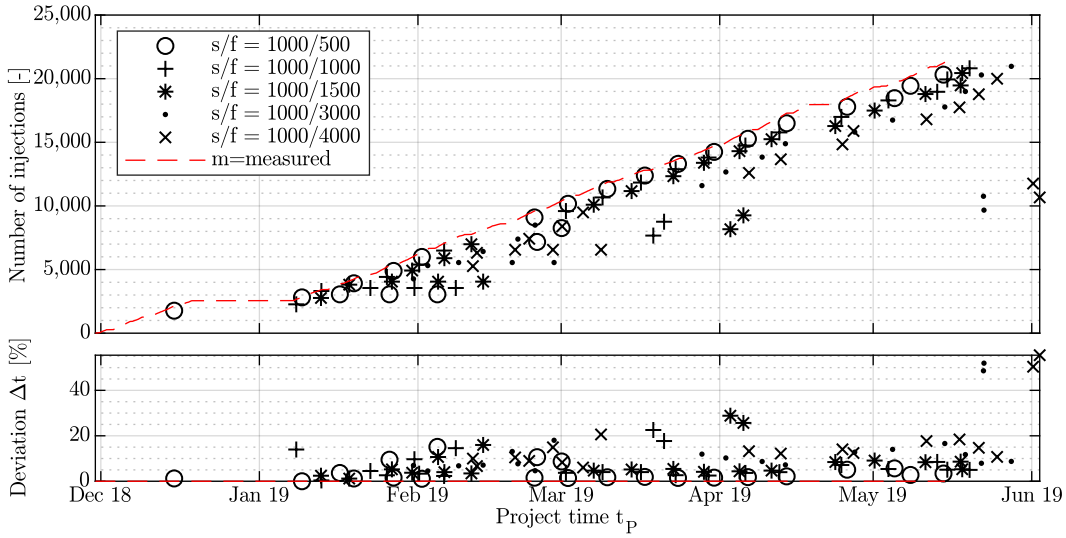


Figure C.57: Results, partial project time forecast, tunnel: 258H, machine: GEA1-3, method: cdf/markov, sample size $s = 1000$.

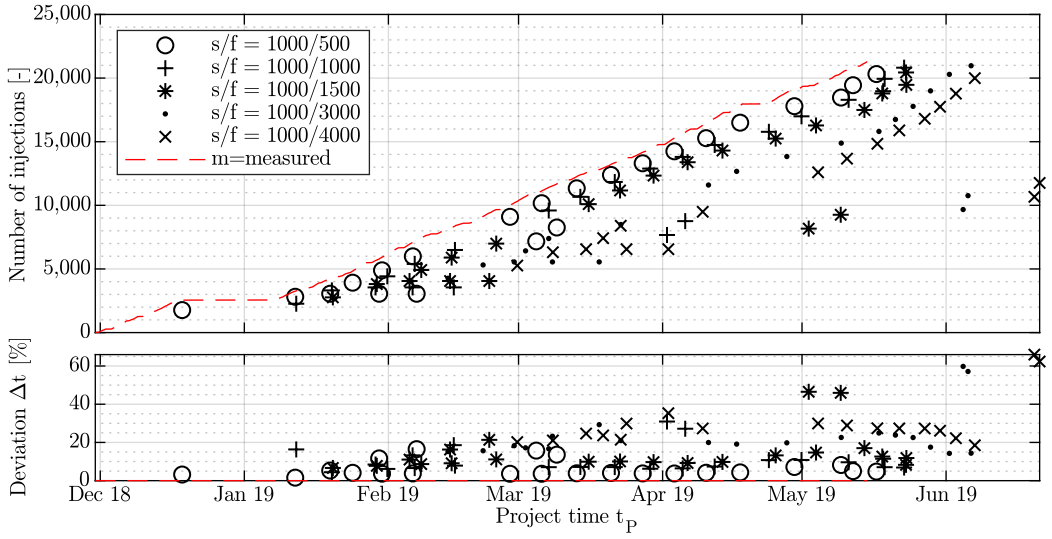


Figure C.58: Results, partial project time forecast, tunnel: 258H, machine: GEA1-3, method: mean/2Inj, sample size $s = 1000$.

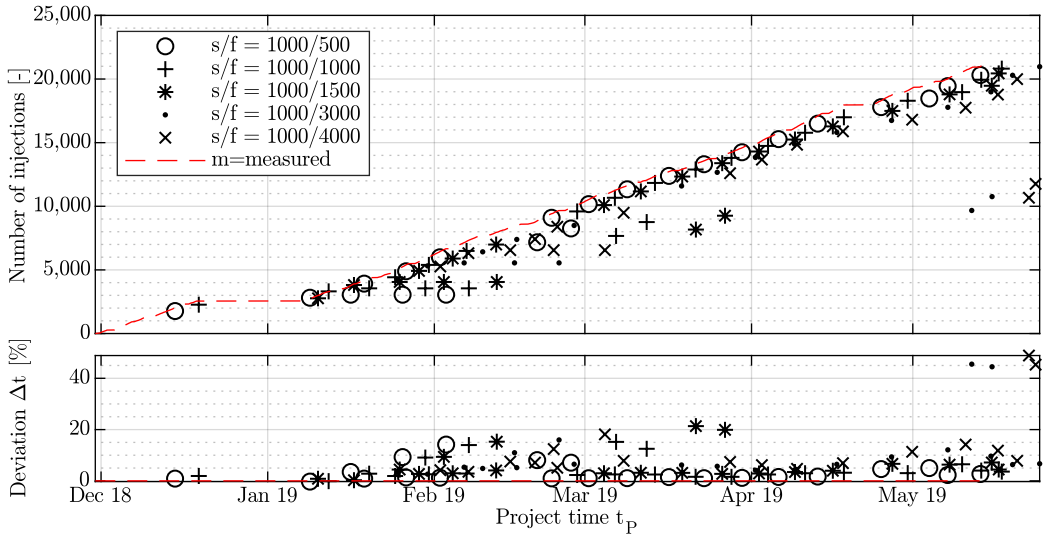


Figure C.59: Results, partial project time forecast, tunnel: 258H, machine: GEA1-3, method: mean/markov, sample size $s = 1000$.

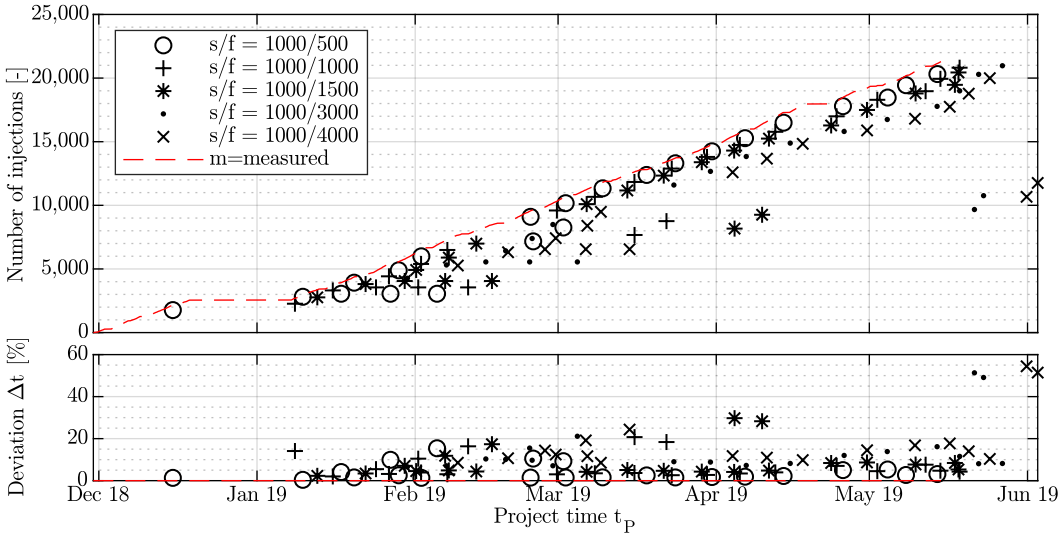


Figure C.60: Results, partial project time forecast, tunnel: 258H, machine: GEA1-3, method: median/2Inj, sample size $s = 1000$.

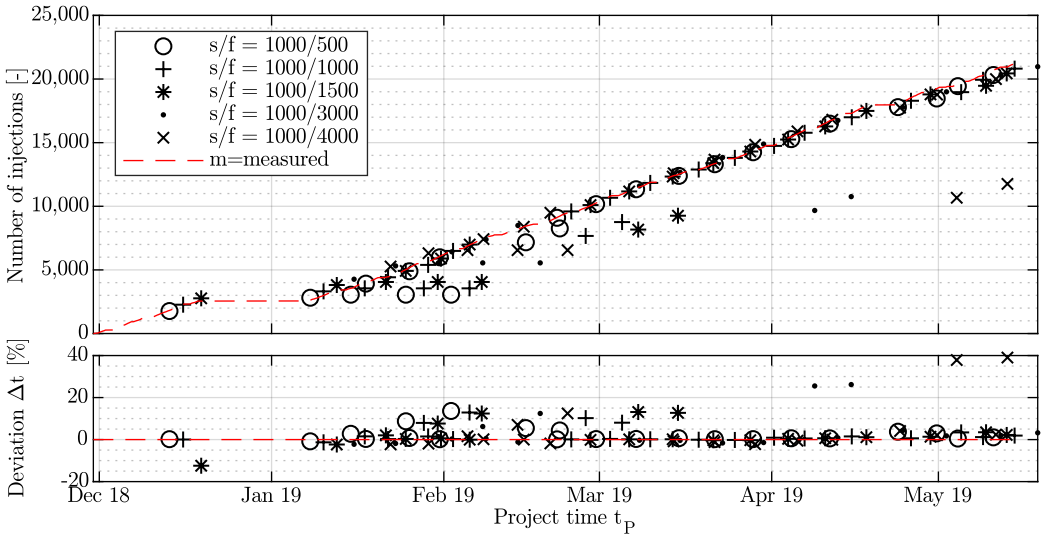


Figure C.61: Results, partial project time forecast, tunnel: 258H, machine: GEA1-3, method: median/markov, sample size $s = 1000$.

C.5.3 Partial Time Forecast, Sample Size $s = 1500$, Tube 258H, GEA1-3

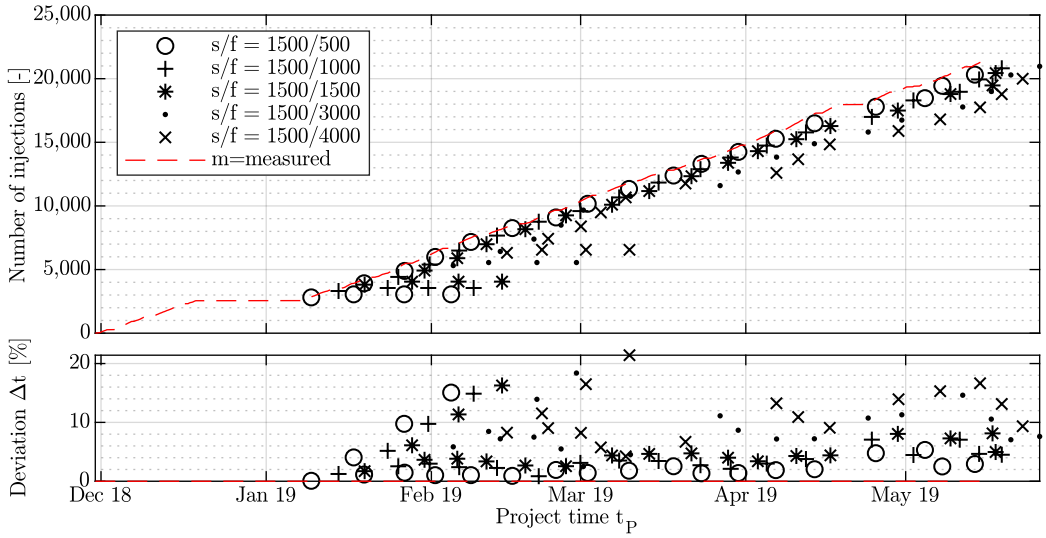


Figure C.62: Results, partial project time forecast, tunnel: 258H, machine: GEA1-3, method: cdf/markov, sample size $s = 1500$.

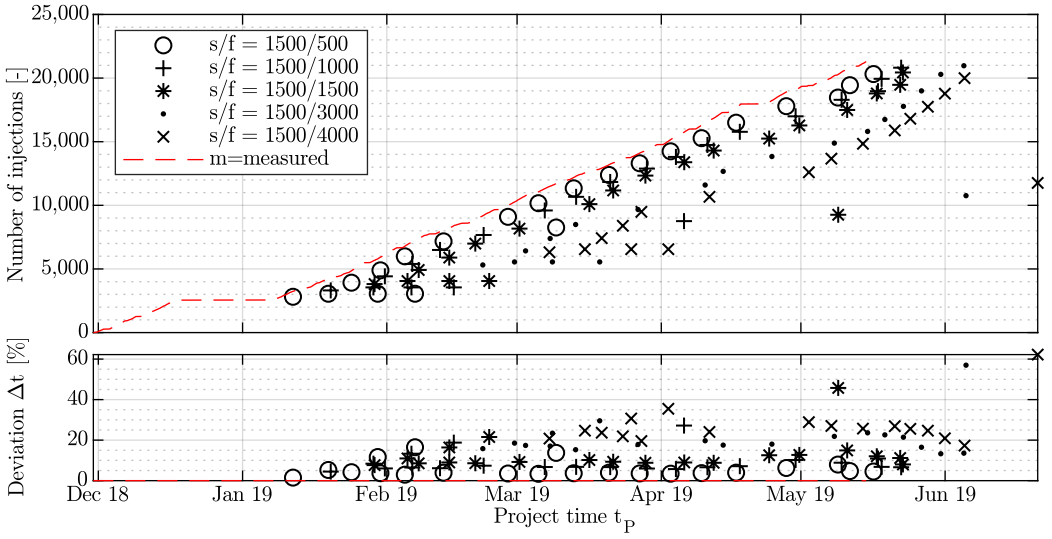


Figure C.63: Results, partial project time forecast, tunnel: 258H, machine: GEA1-3, method: mean/2Inj, sample size $s = 1500$.

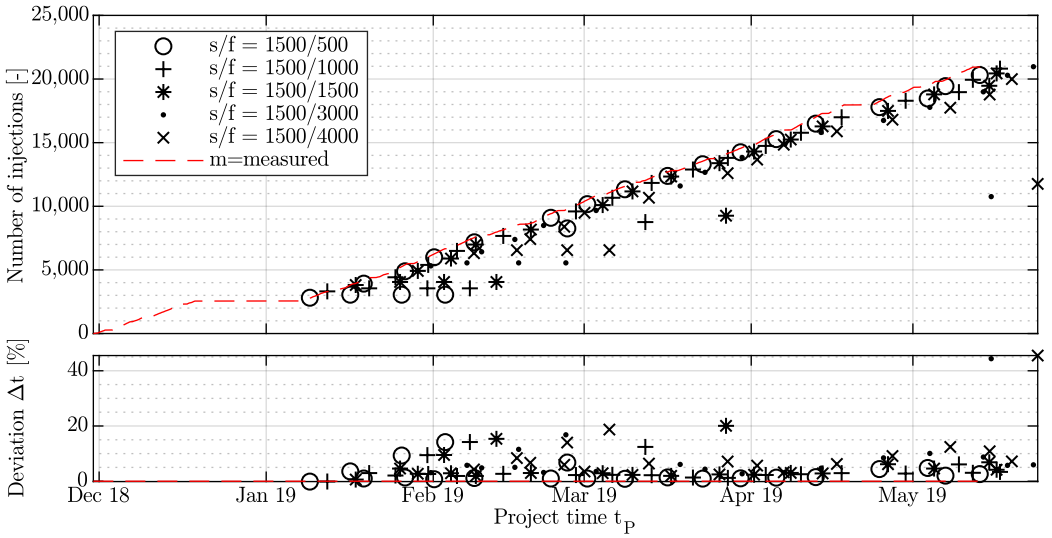


Figure C.64: Results, partial project time forecast, tunnel: 258H, machine: GEA1-3, method: mean/markov, sample size $s = 1500$.

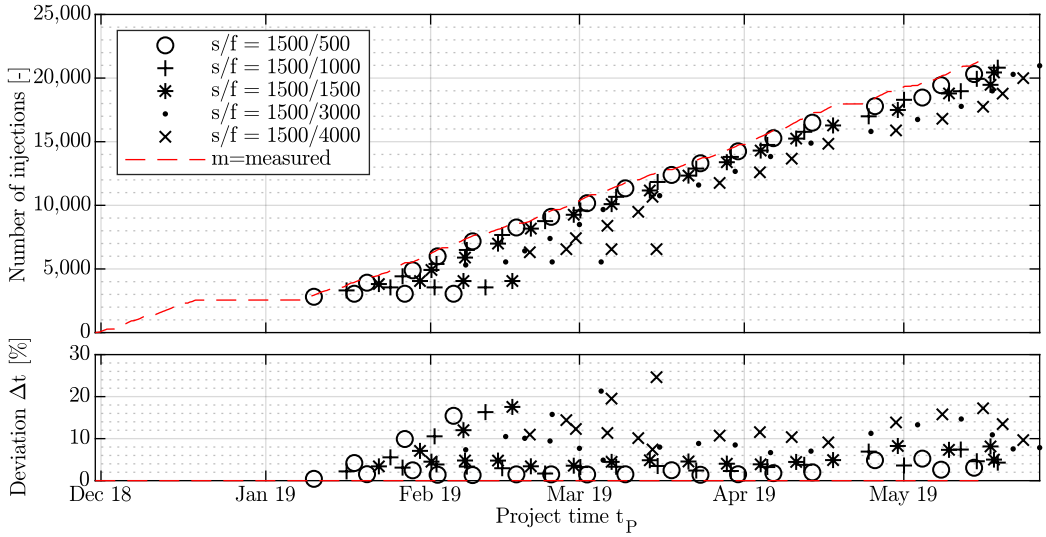


Figure C.65: Results, partial project time forecast, tunnel: 258H, machine: GEA1-3, method: median/2Inj, sample size $s = 1500$.

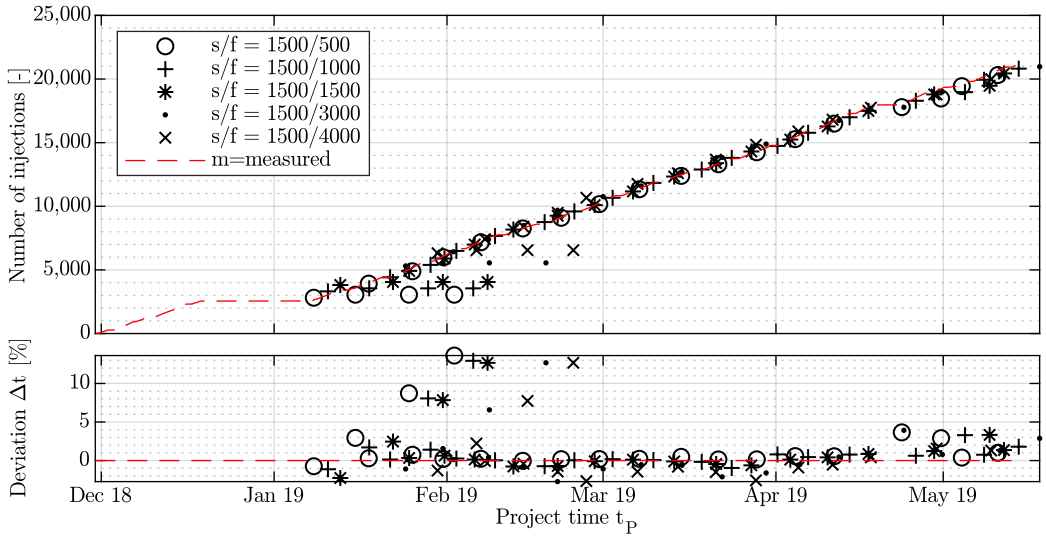


Figure C.66: Results, partial project time forecast, tunnel: 258H, machine: GEA1-3, method: median/markov, sample size $s = 1500$.

C.5.4 Partial Time Forecast, Sample Size $s = 3000$, Tube 258H, GEA1-3

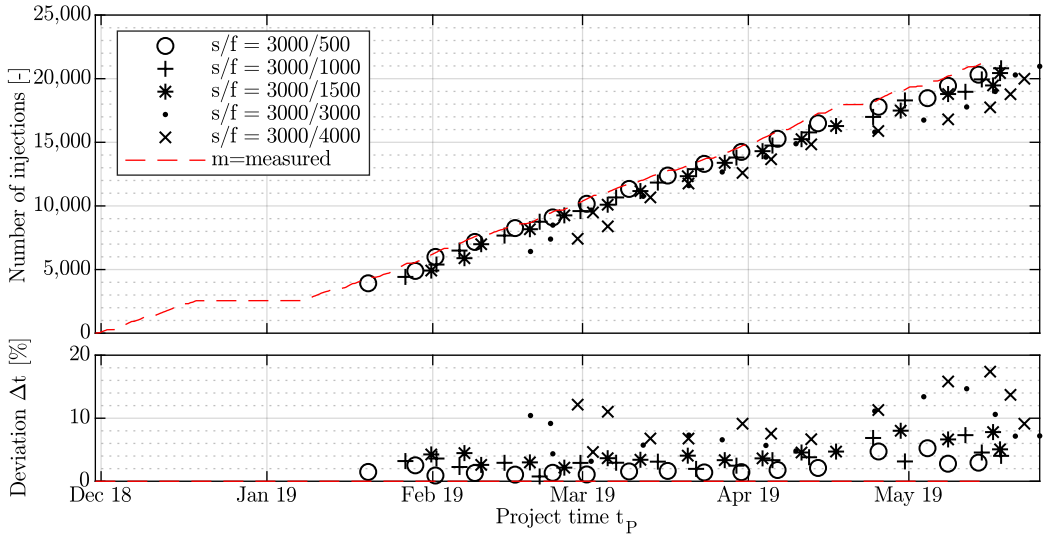


Figure C.67: Results, partial project time forecast, tunnel: 258H, machine: GEA1-3, method: cdf/markov, sample size $s = 3000$.

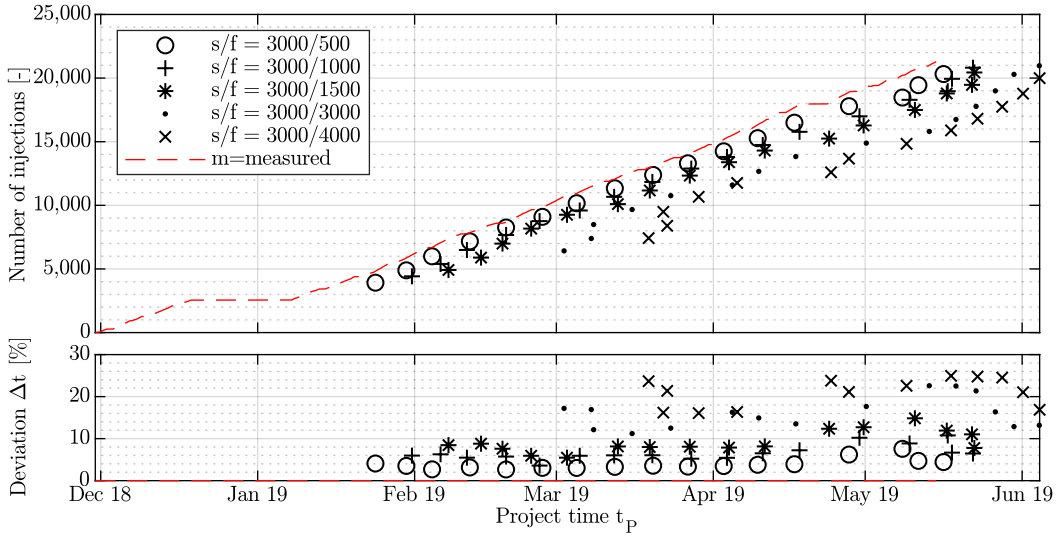


Figure C.68: Results, partial project time forecast, tunnel: 258H, machine: GEA1-3, method: mean/2Inj, sample size $s = 3000$.

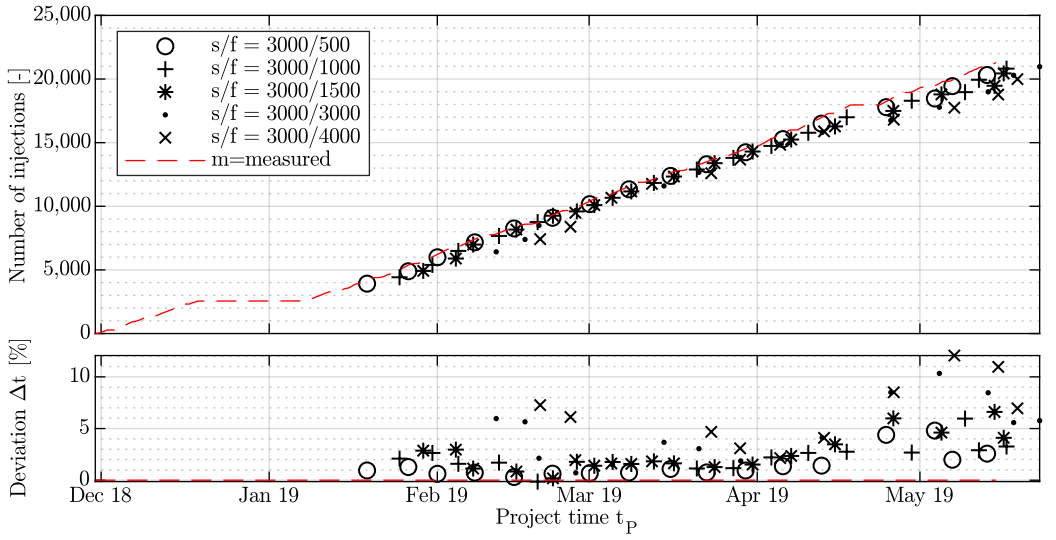


Figure C.69: Results, partial project time forecast, tunnel: 258H, machine: GEA1-3, method: mean/markov, sample size $s = 3000$.

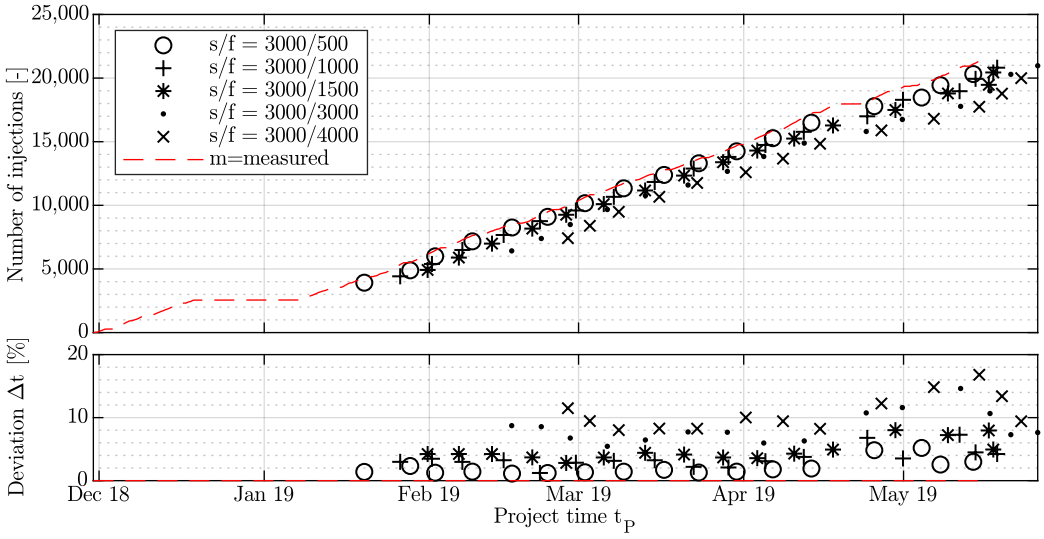


Figure C.70: Results, partial project time forecast, tunnel: 258H, machine: GEA1-3, method: median/2Inj, sample size $s = 3000$.

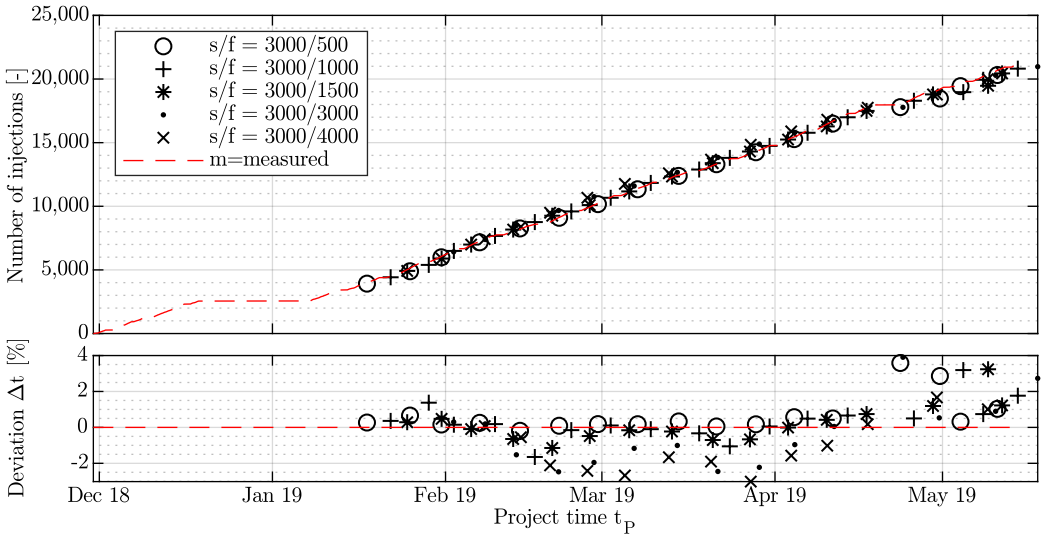


Figure C.71: Results, partial project time forecast, tunnel: 258H, machine: GEA1-3, method: median/markov, sample size $s = 3000$.

C.5.5 Partial Time Forecast, Sample Size $s = 4000$, 258H, GEA1-3

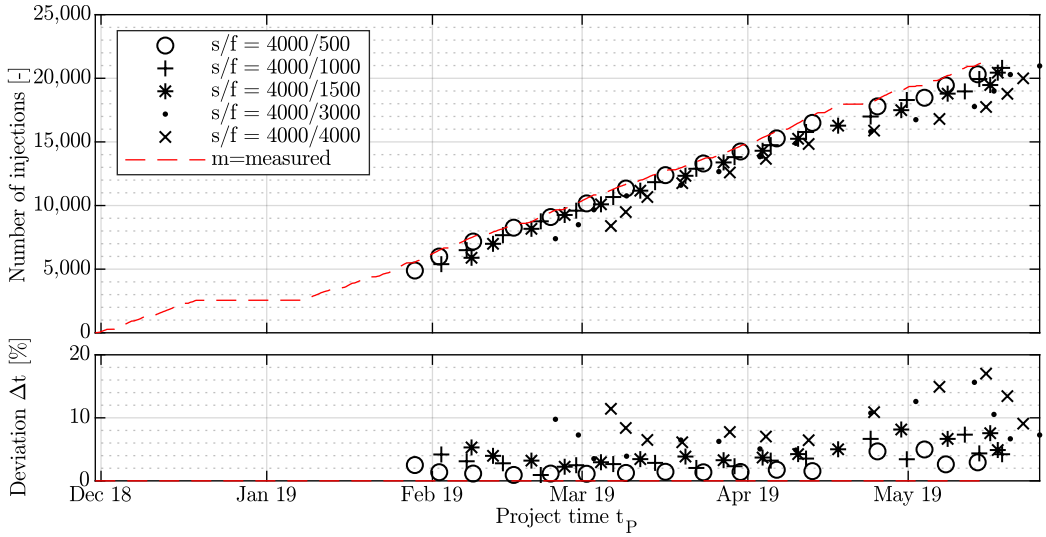


Figure C.72: Results, partial project time forecast, tunnel: 258H, machine: GEA1-3, method: cdf/markov, sample size $s = 4000$.

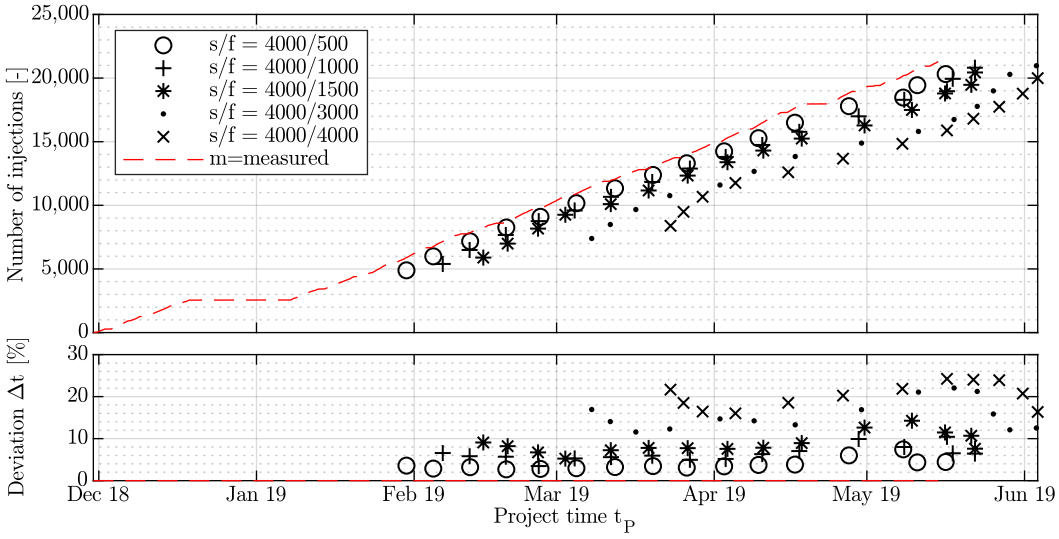


Figure C.73: Results, partial project time forecast, tunnel: 258H, machine: GEA1-3, method: mean/2Inj, sample size $s = 4000$.

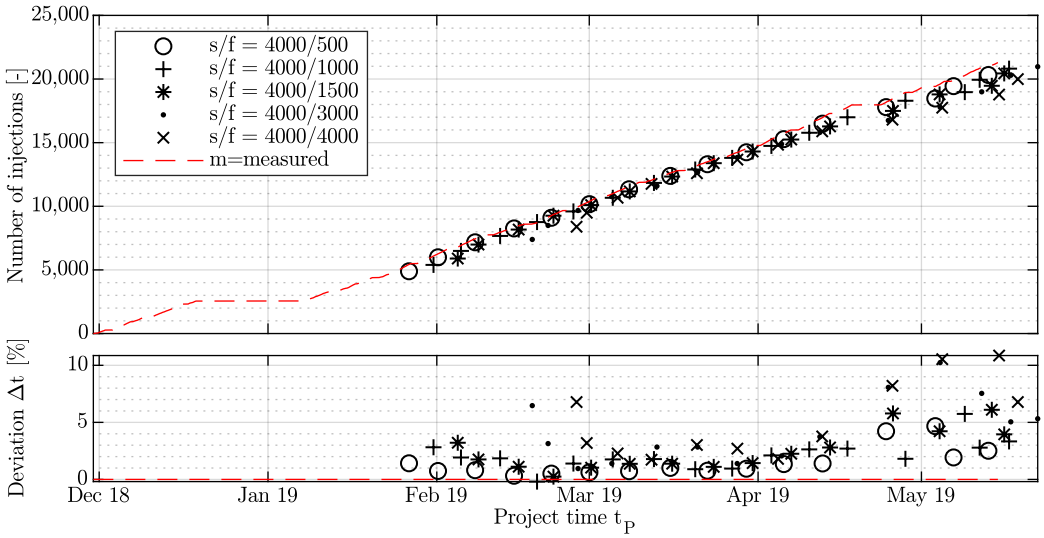


Figure C.74: Results, partial project time forecast, tunnel: 258H, machine: GEA1-3, method: mean/markov, sample size $s = 4000$.

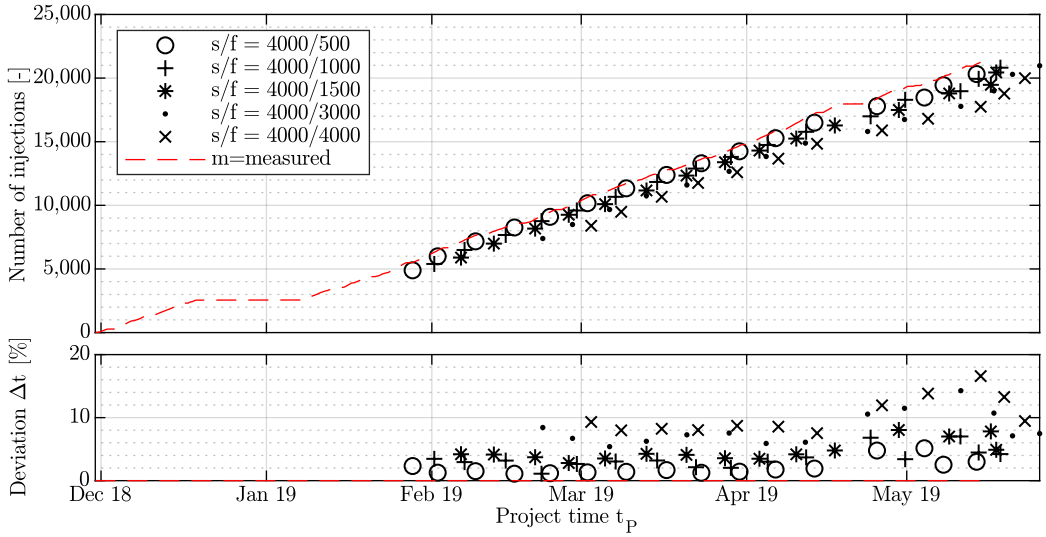


Figure C.75: Results, partial project time forecast, tunnel: 258H, machine: GEA1-3, method: median/2Inj, sample size $s = 4000$.

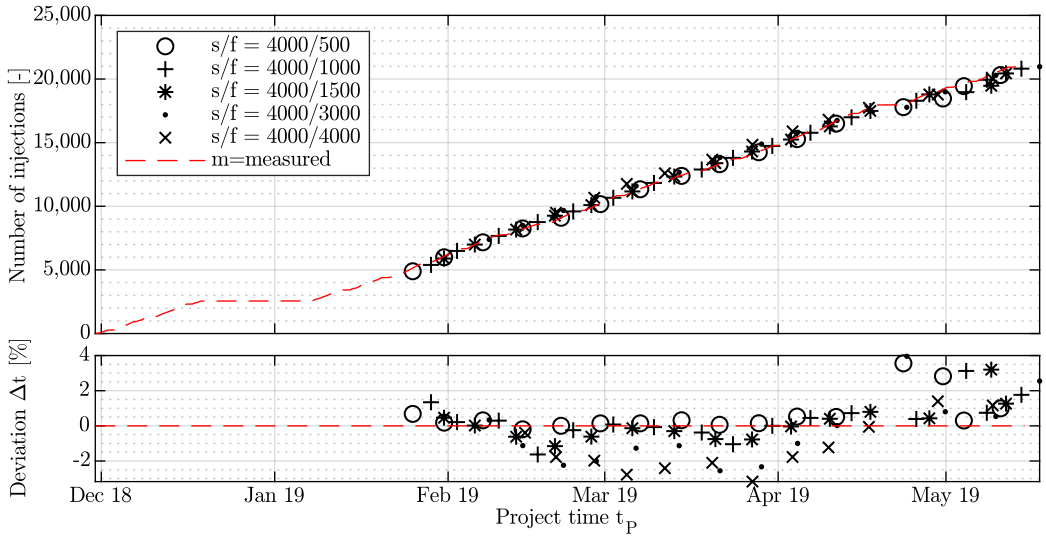


Figure C.76: Results, partial project time forecast, tunnel: 258H, machine: GEA1-3, method: median/markov, sample size $s = 4000$.

C.6 MSPS3 Result Tables

This section shows the result tables used to create the result diagrams for the forecasts calculated with the MSPS3. As such it supplements section 8.3. Four values per simulation result are shown. These are

Label: The value represent the number of injection units of deployed on the construction site. The first value represents AC injection units, the second PU injection units, and the third value MobPU injection units.

Duration: The value gives the forecast result for the total duration of the project in days.

Cost: The value gives the forecast result for the total cost of the project in a million euros.

Shunting: The value gives the forecast result for all injection units' total duration spent with shunting or being idle because their path had been blocked.

Table C.3: MSPS3 Results (1/2)

Label [−]	Duration [d]	Cost [MEUR]	Shunting [d]	Label [−]	Duration [d]	Cost [MEUR]	Shunting [d]
(1-0-1)	701.5	11.6	83.0	(2-3-1)	435.3	23.0	630.8
(1-0-2)	715.4	17.6	619.8	(2-3-2)	449.4	27.4	1005.8
(1-0-3)	708.3	23.2	1125.8	(2-4-0)	421.1	22.5	537.8
(1-0-4)	715.3	29.1	1633.8	(2-4-1)	435.2	26.8	898.0
(1-0-5)	708.0	34.6	2122.8	(2-5-0)	421.4	26.3	888.8
(1-1-0)	708.5	12.0	62.3	(3-0-1)	617.2	20.9	52.5
(1-1-1)	708.3	17.7	580.8	(3-0-2)	386.3	17.1	120.8
(1-1-2)	715.1	23.6	1135.3	(3-0-3)	323.5	17.5	185.3
(1-1-3)	722.1	29.7	1633.3	(3-0-4)	344.4	21.4	405.0
(1-1-4)	715.2	35.2	2192.5	(3-0-5)	344.4	24.4	632.8
(1-2-0)	722.2	18.3	542.3	(3-1-0)	645.1	22.0	821.3
(1-2-1)	729.2	24.4	1087.0	(3-1-1)	400.5	17.8	210.0
(1-2-2)	729.2	30.2	1598.0	(3-1-2)	323.4	17.6	191.0
(1-2-3)	722.2	35.8	2151.0	(3-1-3)	351.2	22.0	425.0
(1-3-0)	722.2	24.4	1052.0	(3-1-4)	351.2	25.0	611.3
(1-3-1)	729.1	30.5	1653.3	(3-2-0)	442.5	19.6	256.3
(1-3-2)	729.0	36.4	2093.8	(3-2-1)	351.3	19.1	167.8
(1-4-0)	715.4	30.3	1497.8	(3-2-2)	344.6	21.7	360.0
(1-4-1)	757.3	38.0	2279.0	(3-2-3)	344.1	24.7	612.5
(1-5-0)	729.5	37.0	2059.3	(3-3-0)	372.1	20.2	178.8
(2-0-1)	617.2	15.6	20.5	(3-3-1)	344.2	21.9	350.5
(2-0-2)	407.2	14.2	100.8	(3-3-2)	338.0	24.5	640.0
(2-0-3)	421.5	18.2	376.0	(3-4-0)	337.1	21.6	318.0
(2-0-4)	414.0	21.5	655.5	(3-4-1)	351.2	25.5	579.0
(2-0-5)	414.2	25.0	914.8	(3-5-0)	358.1	26.0	495.3
(2-1-0)	652.0	16.7	18.0	(4-0-1)	617.4	26.2	76.8
(2-1-1)	428.1	15.0	129.0	(4-0-2)	379.4	20.2	120.5
(2-1-2)	435.2	18.9	365.3	(4-0-3)	316.5	20.1	222.8
(2-1-3)	414.5	21.7	681.0	(4-0-4)	274.0	20.3	259.8
(2-1-4)	407.3	24.8	941.5	(4-0-5)	288.5	23.7	437.0
(2-2-0)	456.7	16.1	89.8	(4-1-0)	652.1	27.8	1416.3
(2-2-1)	421.0	18.6	321.5	(4-1-1)	386.2	20.7	395.8
(2-2-2)	428.2	22.5	635.0	(4-1-2)	323.2	20.6	237.5
(2-2-3)	428.1	26.1	914.0	(4-1-3)	288.3	21.3	251.5
(2-3-0)	442.7	19.6	297.3	(4-1-4)	267.4	22.4	385.5

Table C.4: MSPS3 Results (2/2)

Label [−]	Duration [d]	Cost [MEUR]	Shunting [d]
(4-2-0)	435.2	23.2	101.5
(4-2-1)	344.4	21.9	177.5
(4-2-2)	302.2	22.3	241.3
(4-2-3)	288.5	24	360
(4-3-0)	358.2	22.8	318.5
(4-3-1)	316.3	23.3	251.3
(4-3-2)	295	24.6	387.8
(4-4-0)	323.3	23.9	215.8
(4-4-1)	295.1	24.7	353
(4-5-0)	316.5	26.4	376.5
(5-0-1)	617.4	31.5	104.3
(5-0-2)	379.5	23.6	152.8
(5-0-3)	309.2	22.6	389.8
(5-0-4)	281	23.4	285
(5-0-5)	260	24.3	341.8
(5-1-0)	652.8	33.4	1813.5
(5-1-1)	379.4	23.8	98.5
(5-1-2)	316	23.1	415
(5-1-3)	281.3	23.5	273
(5-1-4)	267.4	25	355.3
(5-2-0)	421.6	26.3	128.3
(5-2-1)	351.1	25.5	418
(5-2-2)	316.1	26.1	299
(5-2-3)	281.3	26.2	296.8
(5-3-0)	351.9	25.7	204
(5-3-1)	316	26.2	334.5
(5-3-2)	288.4	26.8	331
(5-4-0)	323.3	26.8	359.8
(5-4-1)	302.3	28	372.5
(5-5-0)	302	28.2	391.8

C.7 Parameter Settings of MATLAB Optimization Algorithm

Table C.5 shows the default parameter settings for the MATLAB function `gamultiobj` as used for the presented calculations. For a more detailed description of the parameters see MathWorks (2019b).

Table C.5: Parameter settings for optimization with `gamultiobj`.

Parameter	Value	Description*
`UseParallel`	`true`	Calculation of fitness is distributed to multiple processors.
`CreationFcn`	function handle	Handle to own creation function for the creation of a new generation, see section 7.1.6.1.
`CrossoverFcn`	function handle	Handle to own crossover function for the crossovers during the computation of the new generation, see section 7.1.6.2.
`MutationFcn`	function handle	Handle to own mutation function for the mutation during the computation of the new generation, see section 7.1.6.3.
`PopulationSize`	200	Number of individuals created per generation.
`PopInitRange`	$\begin{bmatrix} l_j & \dots & l_n \\ u_j & \dots & u_n \end{bmatrix}$	Range of values for genes in the initial population, e. g., $l_j = 0$ and $u_j = 40, 1 \leq j \leq 10$ for the scenario described in section 7.3.2.1.
`Generations`	3,000	Stop criterion. Maximum number of generations to be computed.
`FunctionTolerance`	$0.5 \cdot 10^{-4}$	Stop criterion. The algorithm stops when the geometric average of the relative change in value of the spread over `StallGenLimit` generations is less than `FunctionTolerance`.
`StallGenLimit`	2,000	see `FunctionTolerance`.
`MaxTime`	18,000	Stop criterion. Maximum computation time in seconds. 18,000 [s] = 5 [h].

* see also MathWorks (2019b)

C.8 Hardware and Software Specifications

Table C.6 shows the specifications of Hard- and Software used for the calculations presented in this thesis.

Table C.6: Hardware and Software. The table lists the hardware and software used in connection with GBPlan. Two set-ups were used. Set-up 1 'BELMART' was located in a home office in the city of Hamburg, while set-up 2 'GBT' was located at the Institute of Geotechnical Engineering and Construction Management, TUHH. Software snippets from external sources are referenced in the GBPlan source code and are located in the GBPlan folder `+Ext.*`.

Item	Specification*	
General		
Name	(1) BELMART	(2) GBT
Location	Hamburg-Altona	Hamburg-Harburg, TUHH
Personal Computer		
CPU	i7-2600, 3.40 GHz	Xeon W-2135 CPU, 3.70GHz
CPU cores	8x	12x
RAM	24 GB	16 GB
Mainboard	Gigabyte P67A-UD4-B3	Dell Precision 5820 Tower
Graphic Card		
Model	NVIDIA GeForce GTX 1660 SUPER	NVIDIA Quadro P2000
GPU	TU116	GP106
Memory Size / Type	6 GB / GDDR6	5 GB / GDDR5
Cores	1,408	1,024
Base Clock	1,530 MHz	1,076 MHz
CUDA	7.5	6.1
Network Connection		
Upload (spec/real)	50 / 47 Mbit/s	** / 707 Mbit/s
Download (spec/real)	10 / 9,4 Mbit/s	** / 523 Mbit/s
Software		
OS	Windows 10 Pro 64-bit (10.0, Build 18362)	
Program language	MATLAB Version: 9.6.0.1335978 (R2019a), Update 8 o Curve Fitting Toolbox, Version 3.5.9 o Deep Learning Toolbox, Version 12.1 o Econometrics Toolbox, Version 5.2 o Global Optimization Toolbox, Version 4.1 o Parallel Computing Toolbox, Version 7.0 o Statistics and Machine Learning Toolbox, Version 11.5	

* The values were measured with DxDiag.exe and zafaco GmbH (2020) and complemented with information from TechPowerUp (2020a,b), ** Specification of contractual values not available

Appendix D Company Presentations

This thesis would not have been possible without the support of the two companies Renesco GmbH and eguana GmbH. For that reason the following two sections provide a short presentation of these two companies.

D.1 Renesco GmbH

"Renesco is a leading international company, specialized in the field of structural waterproofing and injections/grouting services. Established in 1965, Renesco is part of the Marti Group in Switzerland, an owner-operated family business that specializes in underground construction. Marti is a multi-disciplined organization with services that range from building transportation tunnels as a contractor of record to performing specialty underground services on a subcontract basis through its wholly-owned Renesco Group subsidiary.

Renesco provides all kind of specialty services for sealing operations in underground structures against pressurized and non-pressurized water, as well as injection/grouting works. Main areas of application include national and international large-scale projects in tunnelling and infrastructure, as well as in the hydraulic, environmental and rehabilitation sectors.

Renesco has become an international leader in tunnel waterproofing, geomembrane application and specialty tunnel grouting services. The company currently has operating several divisions in different countries and [...] ongoing projects in many more." (Renesco, 2019)

Key areas of expertise:

- Thermoplastic welding of all kind of geomembranes, like PVC, TPO/TPE, FPO, PEHD, etc.
- Application/Installation of sheet waterproofing, loose-laid, fully bonded, post-applied or pre-applied
- Application of liquid and spray-applied waterproofing and coatings
- Fully automatically installation in tunnelling via hot-melt applied sheet waterproofing
- Application of frost and waterproofing protection (PE-foams, PELD, PEHD)
- Setting of anchor bolts including specialised drilling services and the installation of concrete elements
- All kind of chemical (acrylate, polyurethane, silicate, etc.) and cementitious grouting/injection works

- Compensation, permeation and contact grouting, ground stabilization
- Curtain injection, injections into hard and unconsolidated rock
- Specialised drilling services
- Controlling, monitoring and data management

Figure D.1: Company Logo of Renesco GmbH. Source Renesco (2019).

D.2 eguana GmbH

eguana GmbH, Figure D.2, is a young company founded in 2015. Today eguana employs eleven employees and caters its IT-service to well known companies such as Züblin Spezialtiefbau GesmbH, Renesco GmbH, PORR AG and Implenia AG.

Figure D.2: Company Logo of eguana GmbH. Source eguana (2019).

eguana's goal is to develop software to make work in the construction industry easier, more efficient and more productive. The company excels in automated digitization of construction site data and the consecutive post processing of the data captured in its database. Users can access the database through a customizable software solution called eguana SCALES. eguana SCALES runs in the user's browser and processes the data collected from different sensors into easy to understand, customizable evaluations and analysis reports. The company supports its customers in the instrumentation of machines and is able to provide solutions tailored to the customers needs. So far eguana GmbH has provided such individual solutions for injection grouting in tunneling, monitoring of drilling and jetting and the tracing of drainage and water level indicators, eguana (2019).

Appendix E Pre-Publications

Pre-publications of parts of the dissertation are indicated below.

- Jan Onne Backhaus (2021). "Einsatz von neuronalen Netzen zur Vorhersage des Materialvolumens von Injektionsbaustellen im Tunnelbau". German. In: *Bautechnik* 98 (2021)
- Jan Onne Backhaus (2020). "Bauzeitenvorhersage von Injektionen im Tunnelbau". German. In: *Bautechnik* 97 (2020), pp. 1–11
- Jan Onne Backhaus and Markus Dahm (2020). "Einblick in den Stand der Implementierung von Lean Construction Ansätzen in ausgewählten deutschen Bauunternehmen – Ergebnisse einer qualitativen Studie". German. In: *Bauingenieur* 95.2 (2020), pp. 64–72
- Jan Onne Backhaus (2019). "Vorhersage des Injektionsvolumens einer Tunnelbaustelle unter Verwendung von Markov Ketten". German. In: *Tagungsband zum 30. BBB-Assistententreffen 2019 in Karlsruhe*. Fachkongress der wissenschaftlichen Mitarbeiter Bauwirtschaft, Baubetrieb, Bauverfahrenstechnik (Karlsruhe, Germany). Ed. by Shervin Haghsheno et al. Karlsruhe, Germany: KIT Scientific Publishing, July 2019, pp. 6–22
- Jan Onne Backhaus and Jürgen Grabe (2018). "Numerische basierte Prozessanalyse". German. In: *Workshop Digitale Infrastruktur und Geotechnik* (Hamburg, Germany). Ed. by Jürgen Grabe. Vol. 42. Veröffentlichungen des Institutes Geotechnik und Baubetrieb. Institut für Geotechnik und Baubetrieb, Technische Universität Hamburg. Mar. 2018, pp. 247–264
- Jan Onne Backhaus (2018a). "Digitale Optimierung der Bauplanung". German. In: *Tagungsband zum 29. BBB - Assistententreffen – Fachkongress der wissenschaftlichen Mitarbeiter der Bereiche Bauwirtschaft | Baubetrieb | Bauverfahrenstechnik*. Beiträge zum 29. BBB - Assistententreffen vom 06. bis 08. Juni 2018 in Braunschweig (Braunschweig, Germany). Zentrum für Bau- und Infrastrukturmanagement, TU Braunschweig, 2018, pp. 23–33