RIGOROUS ERROR BOUNDS FOR THE OPTIMAL VALUE IN SEMIDEFINITE PROGRAMMING

CHRISTIAN JANSSON[†] AND CHRISTIAN KEIL[†]

Abstract. A wide variety of problems in global optimization, combinatorial optimization as well as systems and control theory can be solved by using linear and semidefinite programming. Sometimes, due to the use of floating point arithmetic in combination with ill-conditioning and degeneracy, erroneous results may be produced. The purpose of this article is to show how rigorous error bounds for the optimal value can be computed by carefully postprocessing the output of a linear or semidefinite programming solver. It turns out that in many cases the computational costs for postprocessing are small compared to the effort required by the solver. Numerical results are presented including problems from the SDPLIB and the NETLIB LP library; these libraries contain many ill-conditioned and real life problems.

Key words. semidefinite programming, linear programming, interval arithmetic, rigorous error bounds, sensitivity analysis, SDPLIB, NETLIB lp library.

AMS subject classifications. 90C22, 65G30, 65N15

1. Introduction.

We consider the *(primal) semidefinite program* in block diagonal form

$$p^* := \min \sum_{j=1}^{n} \langle C_j, X_j \rangle \quad \text{s.t.} \quad \begin{array}{l} \sum_{j=1}^{n} \langle A_{ij}, X_j \rangle = b_i \quad \text{for } i = 1, \dots, m, \\ X_j \succeq 0 \quad \text{for } j = 1, \dots, n, \end{array} \tag{1.1}$$

where C_j, A_{ij}, and X_j are symmetric $s_j \times s_j$ matrices, $b \in \mathbf{R}^m$, and

$$\langle C, X \rangle = \text{ trace } (C^T X) \tag{1.2}$$

denotes the *inner product* for the set of symmetric matrices. Moreover, $\succeq$ is the *Löwner partial order*, that is $X \succeq Y$ iff $X - Y$ is positive semidefinite. In the case $n = 1$ we suppress the index j, and write shortly C, X, A_i, and s for the dimension.

If $s_j = 1$ for $j = 1, \dots, n$ (i.e. C_j, A_{ij}, and X_j are real numbers), then (1.1) defines the standard linear programming problem. Hence, semidefinite programming is an extension of linear programming.

The *Lagrangian dual* of (1.1) is

$$d^* := \max b^T y \quad \text{s.t.} \quad \sum_{i=1}^{m} y_i A_{ij} \preceq C_j \quad \text{for } j = 1, \dots, n, \tag{1.3}$$

where $y \in \mathbf{R}^m$. The constraints $\sum_{i=1}^{m} y_i A_{ij} \preceq C_j$ are called *linear matrix inequalities (LMI)*.

The duality theory is similar to linear programming, but more subtle. The programs satisfy the *weak duality* condition

$$d^* \leq p^*, \tag{1.4}$$

but strong duality requires in contrast to linear programming additional conditions (see Ramana, Tunçel, and Wolkowicz [25] and Vandenberghe and Boyd [31]).

THEOREM 1.1 (Duality Theorem).

[†]Inst. of Computer Science III, Technical University Hamburg–Harburg, Schwarzenbergstraße 95, 21071 Hamburg, Germany.

a) If (1.1) is strictly feasible (i.e. there exist feasible positive definite matrices X_j for $j = 1, \dots, n$) and p^* is finite, then $p^* = d^*$ and the dual supremum is attained.

b) If (1.3) is strictly feasible (i.e. there exists some $y \in \mathbf{R}^m$ such that $C_j - \sum_{i=1}^m y_i A_{ij}$ are positive definite for $j = 1, \dots, n$) and d^* is finite, then $p^* = d^*$, and the primal infimum is attained.

In general, one problem may have optimal solutions and its dual is infeasible, or the duality gap may be positive at optimality. The strict feasibility assumptions in Theorem 1.1 are called *Slater constrained qualifications*.

Semidefinite programming and LMI-methods are evidenced by many applications and a number of survey papers (see for example Skelton and Iwasaki [29], Balakrishnan and Feron [2], and Vandenberghe and Boyd [31]). Applications include global optimization problems, optimal state space realizations, robust controller design, integer programming problems, as well as eigenvalue problems in the form of minimizing the largest, or minimizing the sum of the first few largest eigenvalues of a symmetric matrix X subject to linear constraints on X.

Semidefinite programs can be solved in polynomial time if an a priori bound for the size of their solution is known (see M. Grötschel, L. Lovász, and A. Schrijver [7]). This is a consequence of the ellipsoid method for convex programming. The ellipsoid method has not proven practical, and interior point methods turned out to be the method of choice in semidefinite programming.

Conventionally, algorithms assume that the input data are given exactly, and they use floating point arithmetic for computing an approximate solution. Occasionally, wrong results may be produced, not solely but especially for ill-conditioned problems. Examples where commercial solvers fail for linear optimization problems can be found in Neumaier and Shcherbina [23], and in [10]. It cannot be answered how frequently such failures occur. Ill-conditioning is, however, frequently observed. In a recent paper by Ordóñez and Freund [24] it is stated that 71% of the lp-instances in the NETLIB Linear Programming Library [19] are ill-posed, i.e. the problems have an infinite condition number. The condition number is defined as the scale-invariant reciprocal of the smallest data perturbation that will render the perturbed data instance either primal or dual infeasible. It is set to ∞ if the distance to primal or dual infeasibility is 0, and in this case the problem is called ill-posed.

As pointed out in Neumaier and Shcherbina [23], ill-conditioning is also likely to take place in combinatorial optimization when branch-and-cut procedures sequentially generate linear or semidefinite programming relaxations. Therefore, the computation of rigorous error bounds, which takes account of all rounding errors and of small errors in the input data, can be valuable in practice.

The primary purpose of this paper is to show that by properly postprocessing the output of a semidefinite or linear solver, rigorous error bounds for the optimal value can be obtained. Moreover, existence of optimal solutions can be proved, or a certificate of infeasibility can be given. The input data are allowed to vary within small intervals. Our numerical experience with the NETLIB LP library and other problems demonstrates that, roughly spoken, rigorous lower and upper error bounds for the optimal value are computed even for ill-conditioned and degenerate problems. The quality of the error bounds depends on the quality of the computed approximations and the distances to dual and primal infeasibility. It is typical that either no finite rigorous bounds or distant bounds are computed if the solver gives bad approximations.

The presented results can be viewed as a further development of similar methods for linear programming (Neumaier and Shcherbina [23], and [10]) and convex programming [9].

The paper is organized as follows. Section 2 contains notation, and in §3 an algorithm for computing a rigorous lower bound of the global minimum value is considered. Then, in §4, a rigorous upper bound of the optimal value together with a certificate of existence of optimal solutions is presented. In §5 we show how these rigorous bounds can be used for obtaining certificates of infeasibility. Section 6 contains numerical results. Finally, in §7 some conclusions are given.

2. Notation, interval arithmetic.

Throughout this paper we use the following notation. $\mathbf{R}$, $\mathbf{R}^n$, $\mathbf{R}^n_+$, and $\mathbf{R}^{m\times n}$ denote the sets of real numbers, real vectors, real nonnegative vectors, and real $m \times n$ matrices, respectively. Comparisons $\leq$, absolute value $|\cdot|$, min, max, inf and sup are used entrywise for vectors and matrices.

For a symmetric matrix A the eigenvalues are sorted non-increasingly, $\lambda_{\max}(A) = \lambda_1(A) \geq \lambda_2(A) \geq \ldots \geq \lambda_{\min}(A)$.

For $\mu \in \mathbf{R}$ the operator

$$s\text{vec}(A,\mu) := (A_{11}, \mu A_{21}, \ldots, \mu A_{n1}, A_{22}, \mu A_{32}, \ldots, \mu A_{n\,n-1}, A_{nn})^T, \tag{2.1}$$

transforms symmetric $n \times n$ matrices into $(n+1)n/2$ vectors with the property that the inner product of two symmetric matrices A, B is

$$\langle A, B\rangle = s\text{vec}(A,2)^T s\text{vec}(B,1) = s\text{vec}(A,\sqrt{2})^T s\text{vec}(B,\sqrt{2}), \tag{2.2}$$

and $s\text{vec}(A,\sqrt{2})$ is the customary $s$vec operator. We prefer the first representation of the inner product, since this avoids conversion errors of the input data of semidefinite programs in its vector representation form. The inverse operator of $s$vec is denoted by $s\text{mat}(a,\mu)$.

For block matrices with blocks A_j for $j = 1, \ldots, n$ we define the concatenated vector

$$s\text{vec}((A_j),\mu) := (s\text{vec}(A_1,\mu); \ldots; s\text{vec}(A_n,\mu)). \tag{2.3}$$

We require only some elementary facts about interval arithmetic, which are described here. There are a number of textbooks on interval arithmetic and self-validating methods that can be highly recommended to readers. These include Alefeld and Herzberger [1], Moore [18], and Neumaier [20], [21].

If $\mathbf{V}$ is one of the spaces $\mathbf{R}$, $\mathbf{R}^n$, $\mathbf{R}^{m\times n}$, and $\underline{v}, \overline{v} \in \mathbf{V}$, then the box

$$\mathbf{v} := [\underline{v}, \overline{v}] := \{v \in \mathbf{V} : \ \underline{v} \leq v \leq \overline{v}\} \tag{2.4}$$

is called an *interval quantity* in $\mathbf{IV}$ with *lower bound* $\underline{v}$ and *upper bound* $\overline{v}$. In particular, $\mathbf{IR}$, $\mathbf{IR}^n$, and $\mathbf{IR}^{m\times n}$ denote the set of real intervals $\mathbf{a} = [\underline{a}, \overline{a}]$, the set of real interval vectors $\mathbf{x} = [\underline{x}, \overline{x}]$, and the set of real interval matrices $\mathbf{A} = [\underline{A}, \overline{A}]$, respectively. The real operations $A \circ B$ with $\circ \in \{+, -, \cdot, /\}$ between real numbers, real vectors and real matrices can be generalized to *interval operations*. The result $\mathbf{A} \circ \mathbf{B}$ of an interval operation is defined as the interval hull of all possible real results, that is

$$\mathbf{A} \circ \mathbf{B} := \cap\{\mathbf{C} \in \mathbf{IV} : \ A \circ B \in \mathbf{C} \quad \text{for all} \quad A \in \mathbf{A}, B \in \mathbf{B}\}. \tag{2.5}$$

All interval operations can be easily executed by working appropriately with the lower and upper bounds of the interval quantities. For example, in the simple case of addition, we obtain

$$\mathbf{A} + \mathbf{B} = [\underline{A} + \underline{B}, \overline{A} + \overline{B}]. \tag{2.6}$$

Interval multiplications and divisions require a distinction of cases. For interval quantities $\mathbf{A}, \mathbf{B} \in \mathbf{IV}$ we define

$$\begin{aligned}
\mathrm{mid}\mathbf{A} &:= (\underline{A} + \overline{A})/2 \quad \text{as the } \textit{midpoint}, && (2.7)\\
\mathrm{rad}\mathbf{A} &:= (\overline{A} - \underline{A})/2 \quad \text{as the } \textit{radius,} && (2.8)\\
|\mathbf{A}| &:= \sup\{|A| :\ A \in \mathbf{A}\} \quad \text{as the } \textit{absolute value}, && (2.9)\\
\mathbf{A}^{+} &:= \max\{0, \overline{A}\}, && (2.10)\\
\mathbf{A}^{-} &:= \min\{0, \underline{A}\}. && (2.11)
\end{aligned}$$

Moreover, the comparison in $\mathbf{IV}$ is defined by

$$\mathbf{A} \leq \mathbf{B} \quad \text{iff} \quad \overline{A} \leq \underline{B},$$

and other relations are defined analogously. Real quantities v are embedded in the interval quantities by identifying $v = \mathbf{v} = [v, v]$.

We call $\mathbf{A} \in \mathbf{IR}^{n\times n}$ *symmetric*, if $\mathbf{A}_{ij} = \mathbf{A}_{ji}$ for all i, j, and $\mathbf{A}$ is called positive semidefinite if all $A \in \mathbf{A}$ have this property.

For linear systems of equations with inexact input data, the aim frequently is to compute an interval vector $\mathbf{x} \in \mathbf{IR}^{n}$ containing the *solution set*

$$\Sigma(\mathbf{A}, \mathbf{b}) := \{x \in \mathbf{R}^{n} :\ Ax = b \text{ for some } A \in \mathbf{A},\ b \in \mathbf{b}\}, \tag{2.12}$$

where $\mathbf{A} \in \mathbf{IR}^{n\times n}$, and $\mathbf{b} \in \mathbf{IR}^{n}$. This is an NP-hard problem, but there are several methods that compute enclosures $\mathbf{x}$. A precise description of such methods, required assumptions, and approximation properties can be found for example in Neumaier [20]. Roughly speaking, it turns out that for interval matrices with $\|I - R\mathbf{A}\| < 1$ (R is an approximate inverse of the midpoint $\mathrm{mid}\mathbf{A}$) there are several methods which compute an enclosure $\mathbf{x}$ with $O(n^3)$ operations. The radius $\mathrm{rad}\mathbf{x}$ decreases linearly with decreasing radii $\mathrm{rad}\mathbf{A}$ and $\mathrm{rad}\mathbf{b}$. For the computation of enclosures in the case of large-scale linear systems the reader is referred to Rump [26].

In interval arithmetic several methods for computing rigorous bounds for all or some eigenvalues of interval matrices were developed. Some important references are Floudas [5], Mayer [17], Neumaier [22], and Rump [26, 27].

3. Rigorous lower bound. In many applications some or all input data are uncertain. We model these uncertainties by intervals. In the case of semidefinite programming we assume that symmetric interval matrices $\mathbf{C}_j, \mathbf{A}_{ij} \in \mathbf{IR}^{s_j\times s_j}$, $i = 1, \ldots, m$, $j = 1, \ldots, n$, and an interval vector $\mathbf{b} \in \mathbf{IR}^{m}$ are given. This yields a family of semidefinite programs (1.1), where the input data $P = (A, b, C)$ are allowed to vary within interval bounds $\mathbf{P} := (\mathbf{A}, \mathbf{b}, \mathbf{C})$.

In order to indicate the dependency on the input data, we sometimes write $p^*(P)$, $d^*(P)$, $X^*(P)$, etc.

First, we state a lemma proving a lower bound for the inner product of two symmetric matrices.

LEMMA 3.1. *Let D, X be symmetric matrices of dimension s that satisfy*

$$\underline{d} \leq \lambda_{\min}(D), \quad 0 \leq \lambda_{\min}(X), \quad \textit{and} \quad \lambda_{\max}(X) \leq \overline{x}. \tag{3.1}$$

Then

$$\langle D, X \rangle \geq s \cdot \underline{d}^{-} \cdot \overline{x}. \tag{3.2}$$

Proof. Let D have the eigenvalue decomposition

$$D = Q\Lambda(D)Q^T, \quad QQ^T = I,$$

where $\Lambda(D)$ is the diagonal matrix with eigenvalues of D on the diagonal. Then

$$\begin{aligned} \langle D, X \rangle &= \text{trace}(Q\Lambda(D)Q^T X) \\ &= \text{trace}(\Lambda(D)Q^T XQ) \\ &= \sum_{k=1}^{s} \lambda_k(D)Q(:,k)^T XQ(:,k). \end{aligned}$$

Because of (3.1), we have $0 \leq Q(:,k)^T XQ(:,k) \leq \overline{x}$ yielding

$$\langle D, X \rangle \geq \sum_{k=1}^{s} \lambda_k(D)^{-} \cdot \overline{x} \geq s \cdot \underline{d}^{-} \cdot \overline{x}. \qquad \square$$

We are now ready to prove a rigorous lower bound for the optimal value p^*.

THEOREM 3.2. *Let $\mathbf{P}$ define a family of semidefinite programs (1.1) with input data $P \in \mathbf{P}$, let $\tilde{y} \in \mathbf{R}^m$, set*

$$\mathbf{D}_j := \mathbf{C}_j - \sum_{i=1}^{m} \tilde{y}_i \mathbf{A}_{ij} \quad \textit{for } j = 1, \ldots, n, \tag{3.3}$$

and suppose that

$$\underline{d}_j \leq \lambda_{\min}(\mathbf{D}_j) \quad \textit{for } j = 1, \ldots, n. \tag{3.4}$$

Assume further that upper bounds for the maximal eigenvalues of the primal feasible solution of (1.1)

$$\lambda_{\max}(X_j) \leq \overline{x}_j, \quad \textit{for } j = 1, \ldots, n \tag{3.5}$$

are known, where $\overline{x}_j$ may be infinite. If

$$\underline{d}_j \geq 0 \quad \textit{for } \overline{x}_j = +\infty, \tag{3.6}$$

then for every $P \in \mathbf{P}$ the inequality

$$p^*(P) \geq \inf\{\mathbf{b}^T \tilde{y} + \sum_{j=1}^{n} s_j \cdot \underline{d}_j^{-} \cdot \overline{x}_j\} \tag{3.7}$$

is satisfied, and the right hand side of (3.7) is finite. Moreover, for every $P \in \mathbf{P}$ and every j with $\underline{d}_j \geq 0$ the LMI

$$\sum_{i=1}^{m} y_i A_{ij} - C_j \preceq 0$$

is feasible with $y := \tilde{y}$.

Proof. Let $P = (A, b, C) \in \mathbf{P}$ be chosen fixed, and let $X_j = X_j(P)$ be primal feasible for P and $j = 1, \ldots, n$. Let

$$D_j = C_j - \sum_{i=1}^{n} \tilde{y}_i A_{ij} \quad \text{for } j = 1, \ldots, n,$$

then

$$\sum_{j=1}^{n} \langle C_j, X_j \rangle = \sum_{j=1}^{n} \langle D_j + \sum_{i=1}^{n} \tilde{y}_i A_{ij}, X \rangle = b^T \tilde{y} + \sum_{j=1}^{n} \langle D_j, X_j \rangle.$$

Since $D_j \in \mathbf{D}_j$, Lemma 3.1 implies

$$\sum_{j=1}^{n} \langle D_j, X_j \rangle \geq \sum_{j=1}^{n} s_j \cdot \underline{d}_j^- \cdot \overline{x}_j,$$

which proves the inequality (3.7), and the assumption (3.6) yields a finite right hand side. The last statement is an immediate consequence of $D_j \in \mathbf{D}_j$ and $\lambda_{\min}(D_j) \geq \underline{d}_j \geq 0$. □

In order to judge the quality of the lower bound (3.7), we assume that

i) exact input data $P = \mathbf{P}$ are given,

ii) $D = \mathbf{D}$ is computed exactly, and

iii) the Slater constrained qualifications are fulfilled.

Moreover, let $\tilde{y}$ be the optimal solution of the dual problem (1.2), and let $\underline{d}_j = \lambda_{\min}(D)$ for $j = 1, \ldots, n$. Then $\underline{d}_j \geq 0$ for $j = 1, \ldots, n$, and

$$p^*(P) = d^*(P) = b^T \tilde{y} = \inf\{b^T \tilde{y} + \sum_{j=1}^{n} s_j \cdot \underline{d}_j^- \cdot \overline{x}_j\}.$$

Hence, no overestimation occurs, and it follows that the quality of this lower bound mainly depends on the quality of the $\underline{d}_j$ and on the computed approximation $\tilde{y}$.

An immediate consequence is the following error bound for linear programming problems

$$p^* := \min c^T x \quad \text{s.t. } Ax = b, x \geq 0, \tag{3.8}$$

which is proved in [10], and in [27] for finite bounds $\overline{x}_j$. The input data are $A \in \mathbf{R}^{m \times n}$, $b \in \mathbf{R}^m$, $c \in \mathbf{R}^n$ and $P = (A, b, c) \in \mathbf{R}^{m \times n + m + n}$.

COROLLARY 3.1. *Let $\mathbf{P} = (\mathbf{A}, \mathbf{b}, \mathbf{c}) \in \mathbf{IR}^{m \times n + m + n}$, $\tilde{y} \in \mathbf{R}^m$, and let*

$$\mathbf{d} := \mathbf{c} - \mathbf{A}^T \tilde{y}. \tag{3.9}$$

Assume further that upper bounds for the primal feasible solutions of (3.8)

$$x_j \leq \overline{x}_j \quad \text{for } j = 1, \ldots, n$$

are known, which may be infinite. If

$$\mathbf{d}_j \geq 0 \quad \text{for} \quad \overline{x}_j = +\infty, \tag{3.10}$$

then for every $P \in \mathbf{P}$ the optimal value $p^*(P)$ satisfies the inequality

$$p^*(P) \geq \inf\{\mathbf{b}^T\tilde{y} + \sum_{j=1}^{n} \mathbf{d}_j^- \cdot \overline{x}_j\}. \tag{3.11}$$

Proof. Apply Theorem 3.2 to the semidefinite program where the symmetric matrices A_{ij}, C_j and X_j are one-dimensional. ∎

Next, we describe an algorithm for computing a lower bound of the optimal value, which is based on Theorem 3.2. We assume that an approximate dual optimal solution $\tilde{y} \in \mathbf{R}^m$ of the midpoint problem $\operatorname{mid}\mathbf{P}$ is known. If condition (3.6) is fulfilled, the only work is to compute the right hand side of (3.7). Otherwise, the idea is to perturb all constraints which violate condition (3.6); that is, we solve a perturbed midpoint problem $P = (\operatorname{mid}\mathbf{A}, \operatorname{mid}\mathbf{b}, C(\varepsilon))$ with

$$C_j(\varepsilon) = \operatorname{mid}\mathbf{C}_j - \varepsilon_j I,\ \varepsilon_j = \begin{cases} > 0 & \text{if } \underline{d}_j < 0 \text{ and } \overline{x}_j = +\infty \\ 0 & \text{otherwise.} \end{cases} \tag{3.12}$$

Then the dual optimal solution $y(\varepsilon)$ satisfies the constraints

$$\operatorname{mid}\mathbf{C}_j - \sum_{i=1}^{m} y_i(\varepsilon)\operatorname{mid}\mathbf{A}_{ij} \succeq \varepsilon_j I.$$

Hence, the minimal eigenvalues of the new defect

$$\mathbf{D}_j(\varepsilon) := \mathbf{C}_j - \sum_{i=1}^{m} y_i(\varepsilon)\mathbf{A}_{ij}$$

will increase. Choosing ε_j very large may imply dual infeasibility, choosing $\varepsilon_j > 0$ too small may not be sufficient for satisfying (3.6). Our current trade off is to solve repeatedly perturbed programs until either condition (3.6) is satisfied, or the dual is infeasible. The details are given in Algorithm 3.1.

The algorithm terminates during the first iteration in step 3 if all simple bounds $\overline{x}_j$ are finite or all $\underline{d}_j$ are nonnegative. In this case the computational costs are $O(m \cdot \sum_{j=1}^n s_j^2)$ for computing the $\mathbf{D}_j$'s, the lower bounds $\underline{d}_j$ require $O(\sum_{j=1}^n s_j^3)$ operations, and the bound $\underline{p}^*$ needs $O(m{+}n)$ operations. Hence the costs are negligible compared to the costs for approximately solving a semidefinite program.

In other cases, however, the computational costs may increase because perturbed semidefinite programs must be solved until either the semidefinite programming solver indicates dual infeasibility of the perturbed problem or the maximal number of iterations $l_{\max}$ is reached.

Several modifications of this algorithm are possible and may yield improvements. Here we have considered a simple choice of perturbations: In each step we add to ε_j the negative defects $-\underline{d}_j$ multiplied by a factor 2^{k_j}, where k_j counts the number of iterations that violated the inequality $\underline{d}_j \geq 0$.

In applications we recommend to use infinite bounds $\overline{x}_j$ instead of unreasonable large bounds, because otherwise the sum in (3.7) may yield an unnecessary overestimation.

If the upper bounds $\overline{x}_j = +\infty$ for $j = 1, \ldots, n$, and Algorithm 3.1 delivers a finite lower bound $\underline{p}^*$, then the lower eigenvalue bounds $\underline{d}_j$ must be nonnegative. Since the computation of these eigenvalue bounds introduces some small overestimation, the termination in step 3 in fact proves strict dual feasibility. Hence, the distance to dual infeasibility is greater than zero.

ALGORITHM 3.1. *Rigorous lower bound*

```
given: real or interval input data $\mathbf{P} = (\mathbf{A}, \mathbf{b}, \mathbf{c})$,
       upper bounds $\overline{x}_j$ for $j = 1, \ldots, n$,
       approximate dual optimal solution $\tilde{y}$ for mid $\mathbf{P}$,
       $\underline{p}^* := -\infty$,
       maximal numbers of iterations $l_{\max}$,
       $\varepsilon := 0$, $k := 0$, $l := 0$.
while perturbed problem $P(\varepsilon)$ is dual feasible and $l \leq l_{\max}$
    1. Compute $\mathbf{D}_j = \mathbf{C}_j - \sum_{i=1}^{m} \tilde{y}_i \mathbf{A}_{ij},\ j = 1, \ldots, n$.
    2. Compute rigorous lower bounds $\underline{d}_j \leq \lambda_{\min}(\mathbf{D}_j)$, for $j = 1, \ldots, n$.
    3. If $\underline{d}_j \geq 0$ for every $j$ with $\overline{x}_j = +\infty$ then compute
           $\underline{p}^* = \inf\{\mathbf{b}^T \tilde{y} + \sum_{j=1}^{n} s_j \cdot \underline{d}_j^- \cdot \overline{x}_j\}$,
       STOP.
    4. Compute for $j = 1, \ldots, n$
           $k_j := \begin{cases} k_j + 1 & \text{if } \underline{d}_j < 0 \text{ and } \overline{x}_j = +\infty \\ k_j & \text{otherwise,} \end{cases}$
           $\varepsilon_j := \begin{cases} -2^{k_j} \underline{d}_j + \varepsilon_j & \text{if } \underline{d}_j < 0 \text{ and } \overline{x}_j = +\infty \\ \varepsilon_j & \text{otherwise.} \end{cases}$
    5. Solve the perturbed midpoint problem $P(\varepsilon) = (\operatorname{mid}\mathbf{A}, \operatorname{mid}\mathbf{b}, C(\varepsilon))$, where
       $C_j(\varepsilon) = \operatorname{mid}\mathbf{C} - \varepsilon_j I$ for $j = 1, \ldots, n$, and set $\tilde{y} := \tilde{y}(\varepsilon)$ (approximate dual
       optimal solution).
    6. $l := l + 1$.
end
```

4. Rigorous upper bound.

In this section we investigate the computation of a rigorous upper bound for the optimal value of a semidefinite program together with a certificate of existence of primal feasible solutions. The basic idea is to compute interval matrices $\mathbf{X}_j$ for $j = 1, \ldots, n$ that contain for every semidefinite program $P \in \mathbf{P}$ a primal feasible solution. The desirable characteristics of the matrices $\mathbf{X}_j$ are given in the next theorem.

THEOREM 4.1. *Let $\mathbf{P}$ define a family of semidefinite programs (1.1), and suppose that there exist interval matrices $\mathbf{X}_j$ for $j = 1, \ldots, n$, such that*

$$\begin{gathered} \forall\, b \in \mathbf{b},\ A_{ij} \in \mathbf{A}_{ij},\ i = 1, \ldots, m,\ j = 1, \ldots, n \\ \exists \textit{ symmetric } X_j \in \mathbf{X}_j :\ \sum_{j=1}^{n} \langle A_{ij}, X_j \rangle = b_i, \end{gathered} \tag{4.1}$$

and for $j = 1, \ldots, n$

$$X_j \succeq 0 \textit{ for all symmetric } X_j \in \mathbf{X}_j. \tag{4.2}$$

Then, the optimal value is bounded from above by

$$p^*(P) \leq \sup\{\sum_{j=1}^{n}\langle \mathbf{C}_j, \mathbf{X}_j\rangle\} \tag{4.3}$$

Moreover, if all symmetric $X_j \in \mathbf{X}_j$ are positive definite and $p^*(P)$ is bounded from below, then $p^*(P) = d^*(P)$ for every $P \in \mathbf{P}$ (no duality gap), and the dual supremum is attained.

Proof. Let $P \in \mathbf{P}$ be a fixed chosen problem. Then the conditions (4.1) and (4.2) imply that there exists a primal feasible solution $X_j = X_j(P)$ for $j = 1, \dots, n$. Hence, $\sum_{j=1}^{n}\langle C_j, X_j\rangle \geq p^*(P)$, and the inclusion property (2.5) yields (4.3). The Strong Duality Theorem together with (4.1) and (4.2) shows the existence of a dual optimal solution, and that there is no duality gap. □

In the following, we describe an algorithm for computing this rigorous upper bound. This algorithm must find appropriate interval matrices $\mathbf{X}_j$, and verify the conditions (4.1) and (4.2). We discuss these items below.

To make sure that the upper bound (4.3) is close to the optimal value, the interval matrices $\mathbf{X}_j$ must be close to optimality. The complementary slackness relations may yield rank-deficient matrices that are not positive definite. Therefore, we solve the slightly perturbed midpoint problem

$$\begin{aligned} \min \sum_{j=1}^{n} \langle C_j, X_j\rangle \quad \text{s.t.} \quad \sum_{j=1}^{n} \langle A_{ij}, X_j\rangle &= b_i \quad \text{for } i = 1, \dots, m, \\ X_j &\succeq \varepsilon_j \cdot I, \quad \text{for } j = 1, \dots, n, \end{aligned} \tag{4.4}$$

where ε_j is positive and the input data $(A, b, c) = \operatorname{mid} \mathbf{P}$. Then for small ε_j the optimal solution $(X_j(\varepsilon_j))$ is positive definite and close to the optimal solution of the midpoint problem. This solution is used below to construct appropriate interval matrices $(\mathbf{X}_j)$.

The semidefinite program (1.1) can be written in the equivalent vector representation form

$$\min c^T x \quad \text{s.t.} \quad A^{\text{mat}} x = b,\ X_j \succeq 0,\ \text{for } j = 1, \dots, n, \tag{4.5}$$

where

$$c := s\text{vec}((C_j), 2), \tag{4.6}$$
$$x := s\text{vec}((X_j), 1), \tag{4.7}$$

and the i-th row of the $m \times \sum_{j=1}^{n} \frac{s_j(s_j+1)}{2}$ matrix A^{mat} is defined by

$$A^{\text{mat}}(i, :) = s\text{vec}((A_{ij})_{j=1}^{n}, 2). \tag{4.8}$$

If interval input data $\mathbf{P}$ are given, then we denote by $\mathbf{A}^{\text{mat}}$, $\mathbf{b}$, and $\mathbf{c}$ the corresponding interval quantities. Thus condition (4.1) is equivalent to

$$\forall\, b \in \mathbf{b},\ \forall\, A^{\text{mat}} \in \mathbf{A}^{\text{mat}}\ \exists\, x \in \mathbf{x}:\ A^{\text{mat}} x = b, \tag{4.9}$$

which is an underdetermined system of linear equations with interval input data. Given an approximate optimal solution $(X_j(\varepsilon_j))_{j=1}^{n}$, it is straight forward to solve such a system.

We start by assuming that the $m \times m$ submatrix $\operatorname{mid} \mathbf{A}_I^{\mathrm{mat}}$ with the m columns $\operatorname{mid} \mathbf{A}^{\mathrm{mat}}(:, \beta_i)$ is nonsingular. Let $I = \{\beta_1, \ldots, \beta_m\}$, let N denote all indices of columns of $\operatorname{mid} \mathbf{A}^{\mathrm{mat}}$ which are not in I, let $\mathbf{A}_N^{\mathrm{mat}}$ be the matrix with columns corresponding to the indices of N, and let $\tilde{x} = s\mathrm{vec}((X_j(\varepsilon_j)), 1)$. Now we fix the variables $\tilde{x}_N$, and compute with some verification method for interval linear systems an enclosure $\mathbf{x}_I$ of the solution set

$$\Sigma_I := \{x_I \in \mathbf{R}^m : \; A_I^{\mathrm{mat}} x_I = b - \sum_{\gamma \in N} A_N^{\mathrm{mat}} \tilde{x}_N, \; A \in \mathbf{A}^{\mathrm{mat}}, \; b \in \mathbf{b}\}. \tag{4.10}$$

Then $\mathbf{x} := (\mathbf{x}_I; \tilde{x}_N)$ fulfills (4.9), and therefore $(\mathbf{X}_j) := s\mathrm{mat}(\mathbf{x}, 1)$ satisfies condition (4.1). Condition (4.2) must be verified by some method for computing a rigorous lower bound for the smallest eigenvalue of a symmetric interval matrix.

Algorithm 4.1 contains the details for computing a rigorous upper bound for the optimal value and for proving existence of primal feasible solutions.

If Algorithm 4.1 delivers a finite upper bound $\overline{p}^*$, then the lower eigenvalue bounds $\underline{\lambda}_j$ must be nonnegative. Since the computation of these eigenvalue bounds introduces some small overestimation, the termination in step 3 in fact proves strict primal feasibility. Hence, the distance to primal infeasibility is greater than zero.

Krawzcyk [15] was the first who solved non degenerate interval linear programming problems by using the technique of fixing appropriate variables (the nonbasic variables) and solving a remaining quadratic interval linear system for the basic variables. In [8] this technique was used to compute enclosures of all optimal vertices in the case of degeneration. Hansen used this technique in order to prove existence of a feasible point for nonlinear equations within a bounded box. It was further modified and investigated numerically by Kearfott [11], [12], and is also described in his book [13]. Corresponding algorithms are implemented in his software package GlobSol.

5. Certificate of Infeasibility.

In branch and bound algorithms a subproblem is discarded if the local nonlinear solver detects infeasibility. It is not a rare phenomenon that sometimes local solvers do not find feasible solutions of a subproblem, although they exist (see for example the comments for use of SDPT3 [30]). A consequence is that the global minimum solutions may be cut off.

To avoid this disadvantage we can apply the rigorous lower bounds to a phase I problem. In the literature there are several variations of the phase I method. It is common, however, that the auxiliary objective function describes the infeasibility in the sense that the problem has no feasible solutions, provided the optimal value is greater than zero. The latter property can be verified by the algorithms of the previous section.

Another approach for verifying infeasibility for linear programs in the case of bounded variables is described in Neumaier and Shcherbina [23]. It is based on the observation that the dual of an infeasible problem is unbounded or infeasible, and in many cases solvers compute a ray exposing this. This information can be used for a certificate of infeasibility based on the Farkas lemma.

6. Numerical results.

In this section, we present some numerical experiments. The results for the following semidefinite problems were obtained by using MATLAB [16], the interval toolbox INTLAB [28], and the semidefinite solver SDPT3 (version 3.02) [30].

ALGORITHM 4.1. *Rigorous upper bound, certificate of feasibility*

```
given: real or interval input data $\mathbf{P} = (\mathbf{A}, \mathbf{b}, \mathbf{c})$,
       approximate primal optimal solution $(\tilde{X}_j)_{j=1}^n$ of the midpoint problem,
       $\overline{p}^* := \infty$,
       maximal number of iterations $l_{\max}$, $\varepsilon := 0$, $k := 0$, $l := 0$.
Choose an index set $I$ such that the submatrix mid $\mathbf{A}^{\text{mat}}(:, I)$ is (at least numerically)
nonsingular (for example, by performing an lu decomposition on mid $\mathbf{A}^{\text{mat}}$).

if there is no nonsingular submatrix then STOP.

while perturbed problem $P(\varepsilon)$ is primal feasible and $l \le l_{\max}$
    1. Compute an enclosure $\mathbf{x}_I$ of the solution set $\Sigma_I$, and set $\mathbf{x} := (\mathbf{x}_I; \tilde{x}_N)$.
    2. Set $(\mathbf{X}_j) = s\text{mat}(\mathbf{x}, 1)$, and compute rigorous bounds
           $\underline{\lambda}_j \le \lambda_{\min}(\mathbf{X}_j) \quad \text{for} \quad j = 1, \dots, n.$
    3. if $\underline{\lambda}_j \ge 0$ for $j = 1, \dots, n$ then compute
           $\overline{p}^* = \sup\{\mathbf{c}^T \mathbf{x}\},$
       STOP.
    4. Compute for $j = 1, \dots, n$
           $k_j := \begin{cases} k_j + 1 & \text{if } \underline{\lambda}_j < 0 \\ k_j & \text{otherwise,} \end{cases}$
           $\varepsilon_j := \begin{cases} -2^{k_j}\underline{\lambda}_j + \varepsilon_j & \text{if } \underline{\lambda}_j < 0 \\ \varepsilon_j & \text{otherwise.} \end{cases}$
    5. Solve the perturbed problem (4.4), set $\tilde{X}_j := \tilde{X}_j(\varepsilon)$ for $j = 1, \dots, n$ (approx-
       imate primal optimal solution), and set $\tilde{x} := s\text{vec}((\tilde{X}_j), 1)$.
    6. $l := l + 1$.
end
```

First we consider a semidefinite program of small size

$$\min \quad \langle \begin{pmatrix} 0 & \frac{1}{2} & 0 \\ \frac{1}{2} & \delta & 0 \\ 0 & 0 & \delta \end{pmatrix}, X \rangle$$

$$\text{s.t.} \quad \langle \begin{pmatrix} 0 & -\frac{1}{2} & 0 \\ -\frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, X \rangle \quad = 1,$$

$$\langle \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, X\rangle \quad = \varepsilon,$$

$$\langle \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}, X\rangle \quad = 0,$$

$$\langle \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}, X\rangle \quad = 0,$$

$$X \succeq 0.$$

The Lagrangian dual is

$$\begin{aligned} d^* = \max y_1 + \varepsilon y_2 \quad \text{s.t.} \quad Y \quad &:= \quad C - \sum_{i=1}^{4} A_{i1} y_i \\ &= \quad \begin{pmatrix} -y_2 & \frac{1+y_1}{2} & -y_3 \\ \frac{1+y_1}{2} & \delta & -y_4 \\ -y_3 & -y_4 & \delta \end{pmatrix} \succeq 0. \end{aligned}$$

The linear constraints of the primal problem imply

$$X = \begin{pmatrix} \varepsilon & -1 & 0 \\ -1 & X_{22} & 0 \\ 0 & 0 & X_{33} \end{pmatrix},$$

and X is positive semidefinite iff $X_{22} \geq 0$, $X_{33} \geq 0$, and $\varepsilon \cdot X_{22} - (-1)^2 \geq 0$. Hence, for $\varepsilon \leq 0$, the problem is primal infeasible and $p^* = +\infty$. The dual problem is infeasible for $\delta < 0$ with $d^* = -\infty$.

For $\varepsilon = 0$ and $\delta = 0$ we obtain a duality gap with $p^* = +\infty$ and $d^* = -1$, and the problem is ill-posed. For $\varepsilon > 0$ and $\delta > 0$ the Slater constrained qualifications are satisfied and the optimal value $p^* = d^* = -1 + \delta/\varepsilon$.

Numerical results for different values ε and δ are summarized in Table 6.1. The termination code $tc = 0$ in SDPT3 means normal termination without warning, whereas $tc = -7$ indicates primal infeasibility.

We see that SDPT3 is not backward stable, since in five cases $\tilde{p}^* < \tilde{d}^*$, violating the weak duality. Nevertheless, the rigorous bounds $\overline{p}^*$ and $\underline{p}^*$ overestimate the optimal value only slightly, and this overestimation depends on the quality of the computed approximations. The bounds are infinite if the problem is infeasible or very ill-conditioned. For larger values $\varepsilon > 0$ and $\delta > 0$ the approximations and the rigorous bounds are almost identical, and are not displayed here.

Next, we consider some random problems that are generated by the routine *randsdp* available in SDPT3. For fixed $n = 2$ with dimensions $s_1 = s_2 = 50$ and

Table 6.1
Approximations $\tilde{p}^*$, $\tilde{d}^*$ and rigorous bounds $\overline{p}^*$, $\underline{p}^*$

ε	δ	$\tilde{p}^*$	$\tilde{d}^*$	tc	$\overline{p}^*$	$\underline{p}^*$
0	0	-1.0004	-0.99355	0	∞	$-\infty$
10^{-8}	10^{-8}	-0.99184	-0.98372	0	∞	-0.98373
10^{-6}	10^{-10}	-1.0007	-1.0027	0	-0.99965	-1.0061
10^{-4}	10^{-3}	8.9004	8.9990	0	9.2586	8.9990
-10^{-4}	10^{-3}	28.228	142.86	0	∞	142.86
10^{-4}	-10^{-4}	-5.9323	-1.0361	-7	-5.9324	$-\infty$

Table 6.2
Accuracy for the random problems

m	$\overline{p}^*$	$\mu(\tilde{p}^*, \tilde{d}^*)$	$\mu(\overline{p}^*, \underline{p}_1^*)$	$\mu(\overline{p}^*, \underline{p}_2^*)$
10	$-6.2681e+002$	$4.3247e-007$	$5.9976e-004$	$5.9976e-004$
30	$8.0343e+003$	$1.3344e-007$	$9.8268e-005$	$9.8268e-005$
50	$4.9363e+003$	$7.3853e-008$	$7.3835e-008$	$7.3835e-008$
100	$1.2226e+004$	$1.0397e-009$	$1.0379e-009$	$1.0379e-009$
200	$3.3755e+003$	$5.0769e-009$	$1.5861e-007$	$1.5861e-007$
500	$2.5818e+004$	$5.6273e-009$	$6.8369e-007$	$6.8369e-007$
1000	$7.0016e+004$	$6.8072e-009$	$6.8744e-009$	$6.8744e-009$

varying m the results are displayed in Tables 6.2 and 6.3. All A_{ij}'s and C_j's are dense symmetric matrices, and the number of variables (coefficients of X_j's) is equal to 2550. The accuracy is measured by

$$\mu(a, b) := \frac{|a - b|}{\max\{1.0, (|a| + |b|)/2\}}.$$

For these problems we have computed the rigorous upper bound $\overline{p}^*$, and two rigorous lower bounds $\underline{p}_1^*$ and $\underline{p}_2^*$. For the first lower bound it is assumed that $\overline{x}_1 = \overline{x}_2 = +\infty$, and for the second one we used $\overline{x}_1 = \overline{x}_2 = 10^5$. Table 6.2 shows the accuracy, and in Table 6.3 the performance is given. By $t_{\tilde{p}^*}$, $t_{\overline{p}^*}$, $t_{\underline{p}_1^*}$, and $t_{\underline{p}_2^*}$ we denote the times in seconds for computing the corresponding quantities. In all cases the existence of optimal solutions is rigorously verified.

We see that in the cases $m = 10$, $m = 30$, $m = 200$, and $m = 500$, the rigorous accuracy $\mu(\overline{p}^*, \underline{p}_1^*)$ is inferior compared to the approximate accuracy $\mu(\tilde{p}^*, \tilde{d}^*)$, whereas surprisingly in the other cases both accuracies are almost equal. At a first glance one would suspect that this accuracy is lost due to the worst case analysis done in our approach. But looking more deeply into the code we found the following reason. The algorithm SDPT3 [30] is stopped in the case of solvability if a sufficient accurate solution has been obtained, especially for the primal solution $\tilde{x}$ the inequality

$$\|A^{\text{mat}}\tilde{x} - b\| / \max\{1, \|b\|\} \leq 10^{-8} \tag{6.1}$$

must be satisfied. The corresponding matrix $\tilde{X} := s\text{mat}(\tilde{x}, 1)$ is close to rank-deficiency, and the magnitude of its smallest eigenvalue is frequently around $+10^{-9}$ or $+10^{-10}$. On the other hand, the inequality (6.1) only takes account of the defect $A^{\text{mat}}\tilde{x} - b$. Hence, it can happen that for the exact solution $\hat{x}$ of $A^{\text{mat}}x = b$ that is closest to the computed approximation $\tilde{x}$, the norm $\|\hat{x} - \tilde{x}\| \gg 10^{-8}$ and the smallest eigenvalue of $\hat{X} = s\text{mat}(\hat{x}, 1)$ is negative. In other words, the computational approximation $\hat{X}$ is not sufficiently close to optimality. The consequence is that in Algorithm 4.1 some perturbed problems must be solved, which decreases the rigorous accuracy

TABLE 6.3
Performance comparison for the random problems

m	$t_{\tilde{p}^*}$	$t_{\overline{p}^*}$	$t_{\underline{p}_1^*}$	$t_{\underline{p}_2^*}$
10	1.78	5.25	0.45	0.16
30	3.80	6.89	0.36	0.38
50	5.55	5.64	0.53	0.52
100	13.72	18.19	1.05	1.05
200	41.52	87.52	1.94	1.89
500	209.98	449.23	4.73	4.73
1000	728.81	1421.03	11.81	9.97

TABLE 6.4
Accuracy for the SDPLIB problems

problem	$\overline{p}^*$	$\underline{p}^*$	$\mu(\tilde{p}^*, \tilde{d}^*)$	$\mu(\overline{p}^*, \underline{p}^*)$
arch2	$-6.71509e-001$	$-6.71515e-001$	$2.69574e-007$	$6.52201e-006$
arch8	$-7.05698e+000$	$-7.05698e+000$	$1.84683e-008$	$4.75031e-007$
control1	$-1.77730e+001$	$-1.77846e+001$	$7.27895e-008$	$6.54242e-004$
control4	$-1.97482e+001$	$-1.97942e+001$	$7.31969e-007$	$2.32978e-003$
control10	$-3.49547e+001$	$-3.85331e+001$	$2.77620e-006$	$9.73881e-002$
control11	$-2.51222e+001$	$-3.19587e+001$	$6.47652e-006$	$2.39539e-001$
mcp100	$-2.26157e+002$	$-2.26157e+002$	$5.76094e-009$	$1.62170e-008$
mcp250-1	$-3.17264e+002$	$-3.17264e+002$	$4.76695e-010$	$8.71740e-009$
theta3	$-4.21670e+001$	$-4.21670e+001$	$7.93078e-010$	$3.96012e-007$
theta4	$-5.03212e+001$	$-5.03212e+001$	$1.07772e-008$	$7.55169e-007$
theta5	$-5.72320e+001$	$-5.72323e+001$	$5.00122e-008$	$5.08575e-006$
truss2	$1.23382e+002$	$1.23380e+002$	$1.56295e-007$	$9.64838e-006$
truss5	$1.32636e+002$	$1.32636e+002$	$5.09856e-010$	$4.81005e-006$
truss8	$1.33130e+002$	$1.33115e+002$	$5.36216e-006$	$1.13718e-004$

but increases the computational work. Notice, that for computing the upper bound we set $\hat{x}_N = \tilde{x}_N$, and thus we try to compute the closest exact solution.

Following, we describe the numerical results on some problems from the SDPLIB collection of Borchers [4]. We emphasize that with our current implementation we cannot solve the largest problems. At the moment, for problems with more than 3500 equations and 50000 variables the algorithm runs out of memory; for example *theta5* with 3028 equations and about 30000 variables can be solved rigorously, but *theta6* cannot. We guess that in future releases the range of applicability can be extended. Results are given in Tables 6.4 and 6.5. For all these problems existence of primal and dual optimal solutions and the Slater constrained qualifications could be verified.

We see that, as in the case of random problems, sometimes the rigorous accuracy is worse than the approximate accuracy. It is typical that the computational time for the upper bound $t_{\overline{p}^*}$ is larger than the time $t_{\tilde{p}^*}$ needed for the approximate solution, whereas $t_{\underline{p}^*}$ is in many cases only a fraction of $t_{\tilde{p}^*}$.

There are a number of problems in the SDPLIB which are ill-posed, for example the graph partitioning problems *gpp*. There, the aim is to find a partition of the node set of a weighted undirected graph, such that the cardinality of the partition is equal, and the cut is minimal with respect to the given weight. A semidefinite programming relaxation of this problem is

$$\begin{array}{rrcll} p^* := & \min\langle C, X\rangle & & & \\ & \text{s.t.}\langle E_{ii}, X\rangle & = & \frac{1}{4}, & \text{for } i = 1, \ldots, n \\ & \langle E, X\rangle & = & 0, & \\ & X & \succeq & 0 & \end{array}$$

Table 6.5
Performance comparison for the SDPLIB problems

problem	$t_{\tilde{p}^*}$	$t_{\overline{p}^*}$	$t_{\underline{p}^*}$
arch2	11.08	16.09	1.00
arch8	11.22	17.55	13.47
control1	1.14	1.72	0.31
control4	10.91	13.03	0.97
control10	398.78	514.30	290.49
control11	630.70	813.75	434.25
mcp100	1.61	4.80	0.30
mcp250-1	5.64	12.81	0.75
theta3	13.67	37.81	11.75
theta4	46.44	150.67	37.25
theta5	143.75	547.34	91.81
truss2	8.94	19.95	3.37
truss5	12.84	17.14	10.84
truss8	15.50	26.63	26.77

Table 6.6
Accuracy for some SDPLIB problems

problem	$\overline{p}^*$	$\underline{p}^*$	$\mu(\tilde{p}^*, \tilde{d}^*)$	$\mu(\tilde{p}^*, \underline{p}^*)$
gpp100	$+\infty$	$4.49435e+001$	$7.00054e-008$	$-6.79464e-008$
gpp124-1	$+\infty$	$7.34307e+000$	$3.44803e-007$	$-3.22380e-007$
gpp124-4	$+\infty$	$4.18988e+002$	$7.30945e-008$	$-7.17255e-008$
gpp250-1	$+\infty$	$1.54449e+001$	$7.12482e-008$	$1.52073e-008$
qap5	$+\infty$	$4.36000e+002$	$1.17714e-009$	$1.17716e-009$
qap6	$+\infty$	$3.81404e+002$	$9.41390e-005$	$-9.41390e-005$
qap7	$+\infty$	$4.24790e+002$	$7.24794e-005$	$-7.24794e-005$
qap8	$+\infty$	$7.56865e+002$	$1.21964e-004$	$-1.21964e-004$
qap9	$+\infty$	$1.40988e+003$	$4.54030e-005$	$-4.54030e-005$
hinf1	$+\infty$	$-2.03281e+000$	$1.00564e-004$	$-1.00564e-004$
hinf4	$+\infty$	$-2.74768e+002$	$1.35848e-005$	$-1.35848e-005$
hinf7	$+\infty$	$-3.90827e+002$	$4.39663e-005$	$4.39663e-005$
hinf10	$+\infty$	$-1.08863e+002$	$1.38839e-003$	$-1.38839e-003$
hinf11	$+\infty$	$-6.59384e+001$	$1.15421e-003$	$-1.15421e-003$
hinf12	$+\infty$	$-7.54028e-001$	$6.65981e-001$	$-6.99339e-001$
hinf15	$+\infty$	$-2.60852e+001$	$8.32685e-002$	$-8.32685e-002$

where E_{ii} denotes the $n \times n$ matrix with $E_{ii}(i,i) = 1$, and all other coefficients are equal to zero, E denotes the $n \times n$ matrix with all coefficients equal to one, and n is the number of vertices. Because the inner product of two positive semidefinite matrices is nonnegative, the perturbed equation

$$\langle E, X \rangle = -\varepsilon$$

can never be fulfilled for small positive ε. Hence, the distance to primal infeasibility is zero, i.e. the problem is ill-posed. Since Algorithm 4.1 allows to verify only positive definiteness, the upper bound $\overline{p}^* = +\infty$ is computed for these problems. Tables 6.6 and 6.7 display the results of some problems with $\overline{p}^* = +\infty$. In all cases dual feasibility could be verified, but not primal feasibility. Some problems are apparently primal infeasible, which is expressed by the poor accuracy.

The final numerical experiments investigate the NETLIB suite of linear programming problems [19]. This collection contains problems with up to 15695 variables and

Table 6.7
Performance comparison for some SDPLIB problems

problem	$t_{\tilde{p}^*}$	$t_{\overline{p}^*}$	$t_{\underline{p}^*}$
gpp100	3.86	5.42	4.17
gpp124-1	3.66	11.84	6.70
gpp124-4	3.94	11.92	6.28
gpp250-1	11.06	44.00	24.59
qap5	0.92	4.14	0.47
qap6	1.44	2.98	0.50
qap7	2.25	7.36	1.05
qap8	3.55	8.02	1.80
qap9	7.09	27.19	3.23
hinf1	1.34	5.70	0.27
hinf4	1.16	4.98	0.06
hinf7	1.05	7.11	0.06
hinf10	1.70	1.73	0.30
hinf11	1.94	4.44	0.39
hinf12	2.89	5.38	3.72
hinf15	1.81	4.42	0.64

16675 constraints. They originate from various applications, for example forestry, flap settings on aircraft, staff scheduling, and others, and Ordóñez and Freund have shown that 71% of these problems are ill-posed [24].

We have implemented the rigorous bounds for the special case of linear programming by using the interval library PROFIL/BIAS [14]. The slightly modified algorithms allow to treat equations and inequalities separately as well as free variables. Hence, converting free variables into the difference of two nonnegative variables is not necessary. Notice that this transformation would yield an ill-posed linear programming problem. The approximate optimal solutions were computed by the public domain linear programming solver lp_solve 4.0.1.0 [3]. All programs were compiled with gcc 3.3.1 [6]. The computations were performed on a PC with 2.8 GHz.

Table 6.8 compares the condition numbers cond(P) of the problems to the rigorous lower and upper bounds and the rigorous accuracy $\mu(\overline{p}^*, \underline{p}^*)$. The rigorous accuracy is displayed even if one of the bounds is infinite. In this case we have replaced this infinite bound by the approximate optimal value in $\mu(\overline{p}^*, \underline{p}^*)$. We have displayed the results for 68 problems. For the remaining problems in the NETLIB library either lp_solve 4.0.1.0 was unable to compute an approximate optimal solution due to numerical problems, or the verification failed due to memory limitations. The reason in the latter case is the missing of sparse structures in PROFIL/BIAS.

Finite lower bounds are computed for 62 of the 68 problems, and finite upper ones for 27 of them. Despite an infinite condition number, rigorous lower *and* upper bounds could be computed for *adlittle, gfrd-pnc, sc105, sc205, sc50a, sc50b,* and *stair*, demonstrating that the condition numbers are finite. We guess that these discrepancies stem from computing the condition numbers numerically without verification.

The relative error of the computed bounds varies between $1 \cdot 10^{-7}$ and $1 \cdot 10^{-16}$ for almost all problems. Taking lp_solve's accuracy of $1 \cdot 10^{-9}$ into account, this is close to the best one could expect. The large relative errors for the problems *sctap1, sctap2, sctap3* are due to bad upper bounds. We hope to improve this in the future.

Table 6.9 shows the time in seconds needed to solve the original problem $t_{\tilde{p}^*}$ and the times for computing the rigorous lower and upper bound $t_{\underline{p}^*}$ and $t_{\overline{p}^*}$, respectively.

One can see that the lower bound is almost always computed within a fraction of the time needed to solve the original problem approximately. The upper bound requires sometimes considerably more computational work, which is due to solving additional perturbed problems.

Table 6.8

Rigorous bounds for the NETLIB problems

problem	cond(P)	$\underline{p}^*$	$\overline{p}^*$	$\mu(\overline{p}^*, \underline{p}^*)$
80bau3b	∞	$9.8722e+05$	∞	$2.5801e-08$
adlittle	∞	$2.2549e+05$	$2.2549e+05$	$3.6470e-08$
afiro	$4.565e+03$	$-4.6475e+02$	$-4.6475e+02$	$2.1095e-08$
agg2	∞	$-2.0239e+07$	∞	$2.0868e-08$
agg3	∞	$1.0312e+07$	∞	$7.3999e-08$
agg	∞	$-3.5992e+07$	∞	$2.7323e-08$
bandm	∞	$-1.5863e+02$	∞	$7.0741e-08$
beaconfd	∞	$3.3592e+04$	∞	$1.0000e-08$
blend	$3.195e+05$	$-3.0812e+01$	$-3.0812e+01$	$1.4103e-07$
bnl2	∞	$1.8112e+03$	∞	$3.9661e-08$
bore3d	∞	$1.3731e+03$	∞	$1.3362e-08$
brandy	∞	$-\infty$	∞	
capri	$1.322e+08$	$2.6900e+03$	$2.6900e+03$	$1.7305e-07$
cycle	∞	$-5.2264e+00$	∞	$1.4591e-08$
czprob	∞	$2.1852e+06$	∞	$1.0915e-08$
d6cube	∞	$3.1549e+02$	∞	$1.1175e-08$
degen2	∞	$-1.4352e+03$	∞	$1.1225e-08$
e226	∞	$-2.5865e+01$	∞	$4.4384e-08$
etamacro	∞	$-7.5572e+02$	∞	$9.7768e-09$
finnis	∞	$-\infty$	∞	
fit1d	$1.577e+05$	$-9.1464e+03$	$-9.1464e+03$	$6.1899e-09$
fit1p	$6.616e+05$	$9.1464e+03$	$9.1464e+03$	$8.2528e-07$
fit2d	$6.522e+03$	$-6.8464e+04$	$-6.8464e+04$	$4.8478e-09$
fit2p	$6.398e+05$	$6.8464e+04$	$6.8468e+04$	$5.2357e-05$
ganges	∞	$-1.0959e+05$	∞	$3.5123e-09$
gfrd-pnc	∞	$6.9022e+06$	$6.9022e+06$	$5.5919e-08$
grow15	$7.888e+02$	$-1.0687e+08$	$-1.0687e+08$	$3.5979e-09$
grow7	$3.719e+02$	$-4.7788e+07$	$-4.7788e+07$	$3.6032e-09$
israel	$8.147e+07$	$-8.9664e+05$	$-8.9664e+05$	$1.5935e-08$
kb2	$5.606e+07$	$-1.7499e+03$	$-1.7499e+03$	$2.1799e-08$
lotfi	∞	$-\infty$	$-2.5265e+01$	$3.9057e-09$
modszk1	∞	$3.2057e+02$	∞	$1.5495e-04$
qap8	∞	$2.0350e+02$	∞	$3.3269e-08$
recipe	∞	$-2.6662e+02$	∞	$2.1320e-16$
sc105	∞	$-5.2202e+01$	$-5.2202e+01$	$7.7623e-08$
sc205	∞	$-5.2202e+01$	$-5.2202e+01$	$9.6644e-08$
sc50a	∞	$-6.4575e+01$	$-6.4575e+01$	$5.6764e-08$
sc50b	∞	$-7.0000e+01$	$-7.0000e+01$	$5.7599e-08$
scagr25	$2.045e+07$	$-1.4753e+07$	$-1.4753e+07$	$3.7852e-08$
scagr7	$5.307e+06$	$-2.3314e+06$	$-2.3314e+06$	$3.9152e-08$
scfxm1	∞	$-\infty$	∞	
scfxm2	∞	$-\infty$	∞	
scfxm3	∞	$-\infty$	∞	
scorpion	∞	$1.8781e+03$	∞	$2.9174e-08$
scrs8	∞	$9.0430e+02$	∞	$3.4248e-08$
sctap1	$3.674e+05$	$1.4122e+03$	$1.4178e+03$	$3.9182e-03$
sctap2	$8.358e+04$	$1.7248e+03$	$2.2955e+03$	$3.3085e-01$
sctap3	$1.526e+05$	$1.4240e+03$	$2.0462e+03$	$4.3697e-01$
share1b	$4.878e+09$	$-7.6589e+04$	$-7.6589e+04$	$1.7119e-07$
share2b	$1.233e+07$	$-4.1573e+02$	$-4.1573e+02$	$4.0883e-07$
shell	∞	$1.2088e+09$	∞	0
ship04l	∞	$1.7933e+06$	∞	$9.7666e-09$
ship04s	∞	$1.7987e+06$	∞	$1.0115e-08$
ship08l	∞	$1.9091e+06$	∞	$1.0593e-08$

continued...

problem	cond(P)	$\underline{p}^*$	$\overline{p}^*$	$\mu(\overline{p}^*, \underline{p}^*)$
ship08s	∞	$1.9201e+06$	∞	$1.1197e-08$
ship12l	∞	$1.4702e+06$	∞	$1.1950e-08$
ship12s	∞	$1.4892e+06$	∞	$1.3700e-08$
sierra	∞	$1.5394e+07$	∞	$3.1096e-14$
stair	∞	$-2.5127e+02$	$-2.5127e+02$	$6.2779e-09$
standata	∞	$1.2577e+03$	∞	$3.6157e-16$
standgub	∞	$1.2577e+03$	∞	$3.6157e-16$
standmps	∞	$1.4060e+03$	∞	$1.3776e-08$
stocfor1	$1.939e+07$	$-4.1132e+04$	$-4.1132e+04$	$4.2231e-08$
stocfor2	$7.267e+09$	$-3.9024e+04$	∞	$4.3136e-08$
truss	$2.981e+05$	$4.5882e+05$	∞	$1.0022e-07$
vtp.base	∞	$1.2983e+05$	∞	$3.4508e-08$
wood1p	∞	$1.4429e+00$	∞	$4.9400e-08$
woodw	∞	$1.3045e+00$	∞	$2.4401e-08$

TABLE 6.9

Performance of the NETLIB bounds

problem	$t_{\tilde{p}^*}$	$t_{\underline{p}^*}$	$t_{\overline{p}^*}$
80bau3b	8.51	14.90	41.91
adlittle	0.00	0.00	0.01
afiro	0.00	0.00	0.00
agg2	0.02	0.03	0.28
agg3	0.02	0.03	0.29
agg	0.01	0.02	0.15
bandm	0.20	0.06	5.61
beaconfd	0.01	0.02	0.40
blend	0.00	0.01	0.02
bnl2	11.11	10.40	398.65
bore3d	0.03	0.03	0.21
brandy	0.07	0.16	0.00
capri	0.03	0.03	0.36
cycle	14.40	3.24	3.02
czprob	0.90	0.82	119.93
d6cube	7.67	1.04	11.68
degen2	5.41	0.14	0.44
e226	0.07	0.02	0.25
etamacro	0.10	0.24	0.69
finnis	0.09	0.23	0.47
fit1d	0.16	0.00	0.03
fit1p	2.57	0.38	50.01
fit2d	16.74	0.02	0.16
fit2p	1381.90	31.00	6734.00
ganges	0.79	0.56	241.69
gfrd-pnc	0.10	0.14	18.94
grow15	0.49	0.07	6.03
grow7	0.06	0.02	0.51
israel	0.04	0.01	0.01
kb2	0.00	0.00	0.01
lotfi	0.02	0.06	0.09
modszk1	0.27	0.41	3.14
qap8	259.14	4.19	0.62
recipe	0.00	0.00	0.00
sc105	0.00	0.00	0.01
sc205	0.04	0.09	0.05
sc50a	0.00	0.00	0.00
sc50b	0.00	0.00	0.00
scagr25	0.16	0.06	4.88
scagr7	0.00	0.01	0.04

continued. . .

problem	$t_{\tilde{p}^*}$	$t_{\underline{p}^*}$	$t_{\overline{p}^*}$
scfxm1	0.08	0.18	1.28
scfxm2	0.35	0.68	10.06
scfxm3	1.05	1.67	31.49
scorpion	0.05	0.04	1.50
scrs8	0.23	0.15	13.18
sctap1	0.22	0.04	0.48
sctap2	0.30	0.47	27.70
sctap3	0.60	1.28	64.04
share1b	0.03	0.01	0.10
share2b	0.00	0.01	0.01
shell	0.14	0.04	7.24
ship04l	0.10	0.18	0.05
ship04s	0.07	0.13	0.03
ship08l	0.37	0.69	0.19
ship08s	0.19	0.39	0.11
ship12l	0.74	1.51	0.48
ship12s	0.37	0.75	0.21
sierra	0.19	0.13	3.63
stair	0.40	2.95	0.71
standata	0.01	0.02	1.34
standgub	0.01	0.03	0.01
standmps	0.08	0.12	4.12
stocfor1	0.00	0.01	0.09
stocfor2	1.57	1.68	380.91
truss	23.24	3.74	451.99
vtp.base	0.02	0.01	0.11
wood1p	0.58	0.86	3.40
woodw	2.02	3.97	323.02

7. Conclusions.

The computation of rigorous error bounds for semidefinite optimization problems can be viewed as a carefully postprocessing tool that uses only approximate solutions computed by an SDP or LP solver. The numerical results show that such rigorous error bounds can be computed at least for problems of medium size.

In the future we plan to investigate larger problems by implementing appropriate sparse structures into PROFIL/BIAS.

REFERENCES

[1] G. Alefeld and J. Herzberger. *Introduction to Interval Computations.* Academic Press, New York, 1983.

[2] V. Balakrishnan and E. Feron, editors. *Linear Matrix Inequalities in Control Theory and Applications,* special edition of *International Journal of Robust and Nonlinear Control*, volume 6, no. 9/10. 1996.

[3] M. Berkelaar, P. Notebaert, and K. Eikland. lp_solve. World Wide Web. `http://groups.yahoo.com/group/lp_solve`.

[4] B. Borchers. SDPLIB 1.2, A Library of Semidefinite Programming Test Problems. *Optimization Methods and Software*, 11(1):683–690, 1999.

[5] C.A. Floudas. *Deterministic Global Optimization - Theory, Methods and Applications*, volume 37 of *Nonconvex Optimization and Its Applications.* Kluwer Academic Publishers, Dordrecht, Boston, London, 2000.

[6] Free Software Foundation. gcc. `http://gcc.gnu.org`.

[7] M. Grötschel, L. Lovász, and A. Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, 1:169–197, 1981.

[8] C. Jansson. A Self-Validating Method for Solving Linear Programming Problems with Interval Input Data. *Computing*, Suppl. 6:33–45, 1988.

[9] C. Jansson. A rigorous lower bound for the optimal value of convex optimization problems. *J. Global Optimization*, 28:121–137, 2004.

[10] C. Jansson. Rigorous Lower and Upper Bounds in Linear Programming. *SIAM J. Optim.*, 14(3):914–935, 2004.

[11] R.B. Kearfott. On proving existence of feasible points in equality constrained optimization problems. Preprint, Department of Mathematics, Univ. of Southwestern Louisiana, U.S.L. Box 4-1010, Lafayette, La 70504, 1994.

[12] R.B. Kearfott. On verifying feasiblility in equality constrained optimization problems. Technical report, Department of Mathematics, Univ. of Southwestern Louisiana, 1994.

[13] R.B. Kearfott. *Rigorous Global Search: Continuous Problems.* Kluwer Academic Publisher, Dordrecht, 1996.

[14] O. Knüppel. PROFIL/BIAS and extensions, Version 2.0. Technical report, Inst. f. Informatik III, Technische Universität Hamburg-Harburg, 1998.

[15] R. Krawczyk. Fehlerabschätzung bei linearer Optimierung. In K. Nickel, editor, *Interval Mathematics*, volume 29 of *Lecture Notes in Computer Science*, pages 215–222. Springer Verlag, Berlin, 1975.

[16] MATLAB User's Guide, Version 6. The MathWorks Inc., 2000.

[17] G. Mayer. Result verification for eigenvectors and eigenvalues. In J. Herzberger, editor, *Topics in validated computations. Proceedings of the IMACS-GAMM international workshop, Oldenburg, Germany, 30 August - 3 September 1993*, Stud. Comput. Math. 5, pages 209–276, Amsterdam, 1994. Elsevier.

[18] R.E. Moore. *Methods and Applications of Interval Analysis.* SIAM, Philadelphia, 1979.

[19] Netlib linear programming library. `http://www.netlib.org/lp`.

[20] A. Neumaier. *Interval Methods for Systems of Equations.* Encyclopedia of Mathematics and its Applications. Cambridge University Press, 1990.

[21] A. Neumaier. *Introduction to Numerical Analysis.* Cambridge University Press, 2001.

[22] A. Neumaier. Complete Search in Continuous Global Optimization and Constraint Satisfaction. *Acta Numerica*, 13:271–369, 2004.

[23] A. Neumaier and O. Shcherbina. Safe bounds in linear and mixed-integer programming. *Mathematical Programming*, 99:283–296, 2004.

[24] F. Ordóñez and R.M. Freund. Computational experience and the explanatory value of condition measures for linear optimization. *SIAM J. Optimization*, 14(2):307–333, 2003.

[25] M.V. Ramana, L. Tunçel, and H. Wolkowicz. Strong duality for semidefinite programming. *SIAM J. on Optimization*, 7(3):641–662, 1997.

[26] S.M. Rump. Validated Solution of Large Linear Systems. In R. Albrecht, G. Alefeld, and H.J. Stetter, editors, *Validation numerics: theory and applications*, volume 9 of *Computing Supplementum*, pages 191–212. Springer, 1993.

[27] S.M. Rump. Verification Methods for Dense and Sparse Systems of Equations. In J. Herzberger, editor, *Topics in Validated Computations — Studies in Computational Mathematics*, pages 63–136, Elsevier, Amsterdam, 1994.

[28] S.M. Rump. INTLAB - Interval Laboratory, Version 1, 1998. http://www.ti3.tu-harburg.de/rump/intlab/index.html.

[29] R.E. Skelton and T. Iwasaki. Increased roles of linear algebra in control education. *IEEE Control Syst. Mag.*, 15(4):76–89, 1995.

[30] R.H. Tütüncü, K.C. Toh, and M.J. Todd. Solving semidefinite-quadratic-linear programs using SDPT3. *Math. Program.*, 95B(2):189–217, 2003.

[31] L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM Review*, 38(1):49–95, 1996.