



# Smaller stencil preconditioners for linear systems in RBF-FD discretizations

Michael Koch<sup>1</sup> · Sabine Le Borne<sup>1</sup> · Willi Leinen<sup>2</sup>

Received: 28 March 2024 / Accepted: 3 April 2024  
© The Author(s) 2024

## Abstract

Radial basis function finite difference (RBF-FD) discretization has recently emerged as an alternative to classical finite difference or finite element discretization of (systems) of partial differential equations. In this paper, we focus on the construction of preconditioners for the iterative solution of the resulting linear systems of equations. In RBF-FD, a higher discretization accuracy may be obtained by increasing the stencil size. This, however, leads to a less sparse and often also worse conditioned stiffness matrix which are both challenges for subsequent iterative solvers. We propose to construct preconditioners based on stiffness matrices resulting from RBF-FD discretization with smaller stencil sizes compared to the one for the actual system to be solved. In our numerical results, we focus on RBF-FD discretizations based on polyharmonic splines (PHS) with polynomial augmentation. We illustrate the performance of smaller stencil preconditioners in the solution of the three-dimensional convection-diffusion equation.

**Keywords** Preconditioner · Radial basis function finite difference (RBF-FD) · Meshfree method · Polyharmonic spline · Polynomial augmentation · Iterative solver

**Mathematics Subject Classification (2010)** 65D12 · 65D25 · 65F08 · 65F10 · 65F55 · 65M12 · 65N22

---

✉ Sabine Le Borne  
leborne@tuhh.de

Michael Koch  
michael.koch@tuhh.de

Willi Leinen  
leinen@hsu-hh.de

<sup>1</sup> Hamburg University of Technology, Am Schwarzenberg-Campus 3, 21073 Hamburg, Germany

<sup>2</sup> Helmut Schmidt University, Holstenhofweg 85, 22043 Hamburg, Germany

## 1 Introduction

(Linear) partial differential equations can be solved numerically by a suitable discretization followed by the solution of a linear system of equations. Among possible discretization techniques, the radial basis function finite difference (RBF-FD) method can be viewed as a generalization of the finite difference method to unstructured point sets. The earlier papers on RBF-FD include [45, 46]. Since then, it has gained in popularity and entered into several applications including geosciences [13, 14], heat flow [26], the financial sector [29], linear elasticity [42], fluid mechanics [8, 20], and neuronal dynamics [33].

As in classical finite difference methods, RBF-FD replaces (linear) differential operators by differentiation stencils which encode formulas for weighted sums of function values at neighboring nodes. The weights are chosen such that the formulas become exact for a set of test functions which typically include radial as well as polynomial functions [13, 24]. There exists a large variety of radial basis functions that may be used in RBF-FD. Since many recent papers suggest to use polyharmonic splines (PHS) in combination with high-order polynomials [3–5, 10, 11, 18, 38, 39], we will also follow this approach here. In this setting, an increase in polynomial order leads to an improvement of the discretization accuracy but also requires (significantly) larger stencils. We will review some recommendations from the literature about the relationship between the degree of polynomial augmentation, the degree of the polyharmonic spline, and the stencil size.

The stiffness matrices resulting from larger polynomial degrees and hence larger stencils are less sparse and typically also worse conditioned compared to systems resulting from smaller polynomial degrees (and stencils). In this paper, we will address the challenge of their iterative solution. While earlier papers on RBF-FD often assumed direct solvers for the linear systems, more recent literature includes discussions on iterative solvers (and preconditioners) but mostly uses well-known methods in a blackbox manner, i.e., without any adaptations taking into account the RBF-FD origin of the linear systems. For example, a (restarted) generalized minimal residual (GMRES) method is used in [6, 28, 39] while a stabilized biconjugate gradient (BiCGstab) method solves the systems in [2, 5, 12, 18, 30, 51, 52]. Krylov solvers with ILU preconditioners are used in [2, 5, 6, 12, 18, 28, 30, 39, 51, 52]. Iterative solvers with ILU preconditioners typically suffer from an increase in the problem size and become less efficient if the number of nonzeros in the matrix increases [5, 6]. Moreover, there exist (ILU) preconditioners based on a (problem-dependent) simplified version of the original PDE (e.g., with changed differential operator or boundary conditions) [6, 27, 39] or preconditioners based on an approximate Schur complement [40]. Solvers relying on multilevel or multigrid ideas are introduced in [1, 34, 49, 50] while a sparse direct solver is proposed in [18].

In this paper, we propose to build preconditioners based on an auxiliary stiffness matrix that results from RBF-FD discretization using a smaller stencil size. This auxiliary system in turn will be replaced by its approximate LU factorization where we use either a sparse ILU or an  $\mathcal{H}$ -LU factorization based on hierarchical ( $\mathcal{H}$ -) matrices [17]. An advantage of  $\mathcal{H}$ -matrices is their ability to construct high-accuracy preconditioners (with storage and computation cost of almost linear complexity) [21, 23]. Our

motivation for using smaller-size stencils is that the auxiliary matrices are better conditioned and sparser compared to systems resulting from larger stencils. As a result, their approximate LU factorizations can be constructed faster while still accelerating convergence and hence yield efficient preconditioners with reduced setup times and cost per iteration step.

The remainder of this paper is organized as follows. In Sect. 2, we briefly review the RBF-FD discretization of linear partial differential equations. More details can be found in most of the given references (see, e.g., [12, 13]). Section 3 is devoted to the construction of smaller stencil preconditioners. In Sect. 4, we illustrate the performance of the proposed preconditioners through numerical tests. Our model problem is a three-dimensional convection-diffusion equation on two different domains (unit cube and bunny-shaped) with a recirculating convection and Dirichlet as well as mixed (Dirichlet and Neumann) boundary conditions. Finally, a conclusion and an outlook are presented in Sect. 5.

## 2 The RBF-FD method

In this section, we review radial basis functions as a tool for the approximation of linear differential operators (Sect. 2.1) and explain the discretization of a partial differential equation with Dirichlet boundary conditions by means of the RBF-FD method (Sect. 2.2).

### 2.1 RBF and RBF-FD approximation

Radial basis functions typically refer to a set of functions that are characterized by individual centers but are otherwise identical and radial in shape, i.e., shifted copies of one another, the function value at a point depending only on its distance to the center. We concentrate in this paper on radial basis functions which are based on polyharmonic splines (PHS).

**Definition 2.1** For a spatial dimension  $d \in \mathbb{N}$ , a set of pairwise distinct nodes  $X := \{x_1, \dots, x_N\} \subset \mathbb{R}^d$  with  $N \in \mathbb{N}$  and some  $k \in \mathbb{N}$ , we define PHS( $k$ ) radial basis functions (RBFs) centered at nodes  $x_i$  as

$$\Phi_{x_i} : \mathbb{R}^d \rightarrow \mathbb{R}, \quad \Phi_{x_i}(x) := \|x - x_i\|_2^{2k-1}, \quad i \in \{1, \dots, N\}.$$

RBFs can be used to approximate the action of a linear differential operator  $\mathcal{L}$  as follows. Let  $u : \Omega \rightarrow \mathbb{R}$  be a sufficiently smooth function on an open domain  $\Omega \subseteq \mathbb{R}^d$ , and let the boundary  $\partial\Omega = \Gamma_D \cup \Gamma_N$  be divided into two disjoint parts to allow for different treatment of Dirichlet and Neumann (or mixed) boundary conditions. Let

$$X_\Omega \cup X_{\Gamma_N} := \{x_1, \dots, x_{N_I}\}, \quad X_{\Gamma_D} \cup X_G := \{x_{N_I+1}, \dots, x_N\} \subset \mathbb{R}^d \setminus \Omega$$

be sets of distinct nodes in the interior ( $X_\Omega \subset \Omega$ ), on the Neumann boundary ( $X_{\Gamma_N} \subset \Gamma_N$ ), neighboring  $\Gamma_N$  outside of  $\Omega \cup \partial\Omega$  (called ghost or fictitious nodes,  $X_G \subset$

$\mathbb{R}^d \setminus \overline{\Omega}$ , here with  $\#X_G = \#X_N$ ) and the Dirichlet boundary ( $X_{\Gamma_D} \subset \Gamma_D$ ), resp., with  $N_I, N \in \mathbb{N}, N_I \leq N$ . Such node sets  $X := X_\Omega \cup X_{\Gamma_D} \cup X_{\Gamma_N} \cup X_G$  also occur in finite difference discretizations, but now, the nodes may be scattered and need not be connected by edges of a (structured tensor product) grid. We add ghost nodes only in the case of Neumann boundary conditions, i.e.,  $\Gamma_N \neq \emptyset$ . Ghost nodes are additional nodes outside the domain  $\Omega$  and may increase stability and accuracy by avoiding lop-sided stencils [5, 18, 47].

For a node  $x_j \in X_\Omega \cup X_{\Gamma_N}, j \in \{1, \dots, N_I\}$ , and stencil size  $n \leq N$ , let  $X_j \subseteq X$  be a stencil associated with the (stencil) center  $x_j$ . We use  $\mathcal{I}_j \subseteq \{1, \dots, N\}$  to denote the index set of nodes included in the stencil  $X_j$ , i.e.,

$$X_j = \{x_{s_1^j}, \dots, x_{s_n^j}\} = \{x_i \in X : i \in \mathcal{I}_j\} \text{ for } \mathcal{I}_j := \{s_1^j, \dots, s_n^j\}. \tag{2.1}$$

The goal is to determine weights  $\{w_1^j, \dots, w_n^j\} \subset \mathbb{R}$  such that the differential operator applied to the function  $u$  and evaluated at  $x_j$  can be expressed (approximately) as a weighted sum of function values at the nodes in the stencil  $X_j$ , i.e.,

$$\mathcal{L}u(x_j) := \mathcal{L}u(x)|_{x=x_j} \approx \sum_{i=1}^n w_i^j u(x_{s_i^j}). \tag{2.2}$$

PHS RBFs are usually augmented by polynomials. We denote the space of  $d$ -variate polynomials of degree at most  $\ell \in \mathbb{N}_0$  by  $\Pi_\ell$  and let  $\{p_1, \dots, p_M\}$  be a basis of  $\Pi_\ell$  with  $M := \dim \Pi_\ell = \binom{\ell+d}{d}$ . Furthermore, for each node  $x_j \in X_\Omega \cup X_{\Gamma_N}$ , we define

$$(\mathcal{L}\Phi_j)^T := \left[ \mathcal{L}\Phi_{x_{s_1^j}}(x_j) \cdots \mathcal{L}\Phi_{x_{s_n^j}}(x_j) \right], \quad (\mathcal{L}\mathbf{p}_j)^T := \left[ \mathcal{L}p_1(x_j) \cdots \mathcal{L}p_M(x_j) \right],$$

$$\text{and } \begin{bmatrix} A_j & P_j \\ P_j^T & \mathbf{0} \end{bmatrix} := \begin{bmatrix} \Phi_{x_{s_1^j}}(x_{s_1^j}) \cdots \Phi_{x_{s_n^j}}(x_{s_1^j}) & p_1(x_{s_1^j}) \cdots p_M(x_{s_1^j}) \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \Phi_{x_{s_1^j}}(x_{s_n^j}) \cdots \Phi_{x_{s_n^j}}(x_{s_n^j}) & p_1(x_{s_n^j}) \cdots p_M(x_{s_n^j}) \\ p_1(x_{s_1^j}) \cdots p_1(x_{s_n^j}) & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ p_M(x_{s_1^j}) \cdots p_M(x_{s_n^j}) & 0 & \cdots & 0 \end{bmatrix}.$$

The desired stencil weights  $w^j := [w_1^j, \dots, w_n^j]^T$  in (2.2) (along with weights  $\tilde{w}^j := [\tilde{w}_1^j, \dots, \tilde{w}_M^j]^T$  which can be viewed as Lagrange multipliers and will not be used for the stencil) are then given by the solution of the linear system of equations

$$\begin{bmatrix} A_j & P_j \\ P_j^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} w^j \\ \tilde{w}^j \end{bmatrix} = \begin{bmatrix} \mathcal{L}\Phi_j \\ \mathcal{L}\mathbf{p}_j \end{bmatrix} \tag{2.3}$$

(see, e.g., [13, 24] for further details).

In the following, we discuss the (relationship between the) parameters involved in the construction of the system (2.3):

- The degree  $\ell$  of polynomial augmentation (and resulting number  $M$  of polynomial basis functions which yields the number of columns of the matrix block  $P_j$ )
- The stencil size  $n$  (which yields the size of the square block  $A_j$ )
- The parameter  $k$  in the exponent of the polyharmonic spline  $\text{PHS}(k)$  (which influences the numerical entries in the block  $A_j$ )

Typically, one first chooses the polynomial degree  $\ell$  since it determines the approximation order of the finite difference stencil [39]. Since the dimension  $M$  of the space of polynomials of maximum degree  $\ell$  corresponds to the number of columns of the matrix  $P_j$  in (2.3), the stencil size  $n$  (which corresponds to the number of rows and columns of  $A_j$  in (2.3)) needs to satisfy  $n \geq M$  since otherwise the matrix in (2.3) would be singular. We will follow the recommendation in [39] and set the stencil size to

$$n = 2M + \lfloor \ln(2M) \rfloor \quad \text{with } M = \binom{\ell + d}{d}. \tag{2.4}$$

For the parameter  $k$  in the generating function  $r^{2k-1}$  of  $\text{PHS}(k)$ , there hold the following constraints:  $\text{PHS}(k)$  is conditionally positive definite of order  $k$  [9, 48]; hence, regularity of the matrix in (2.3) can only be guaranteed for polynomial degree  $\ell \geq k - 1$  (and nodes that satisfy  $\Pi_\ell$ -unisolvency constraints, i.e., the only polynomial in  $\Pi_\ell$  that is zero at all nodes is the zero polynomial). We will pursue the following two possibilities [24]:

$$k = k_1 := \begin{cases} \max\{D, \frac{\ell+1}{2}\}, & \text{if } \ell \text{ is odd,} \\ \max\{D, \frac{\ell}{2}\}, & \text{if } \ell \text{ is even} \end{cases} \tag{2.5}$$

with  $D$  denoting the order of the differential operator (i.e.,  $D = 2$  in our numerical examples) or

$$k = k_2 := \begin{cases} \ell + 1, & \text{if } \ell \leq 3, \\ \ell, & \text{if } \ell \geq 4. \end{cases} \tag{2.6}$$

The following Table 1 lists the parameter values obtained for polynomial degrees  $\ell \in \{2, 3, 4, 5, 8\}$  (and spatial dimension  $d = 3$ ) following (2.4), (2.5), and (2.6).

### 2.2 RBF-FD for partial differential equations

In the previous subsection, we reviewed an approximation of the application of a linear differential operator to a function and its evaluation at a fixed node by a sum of weighted function values at stencil nodes. We next extend this approach to the discretization of a linear partial differential equation

$$\mathcal{L}u(x) = f(x) \quad \forall x \in \Omega, \tag{2.7a}$$

**Table 1** Combination of parameters  $\ell, n, k$  in RBF-FD with PHS( $k$ ) in  $d = 3$  spatial dimensions

Polym. degree $\ell$	Stencil size $n$	$k = k_1$ in PHS( $k$ )	$k = k_2$ in PHS( $k$ )
2	22	2	3
3	43	2	4
4	74	2	4
5	116	3	5
8	335	4	8

$$u(x) = g_D(x) \quad \forall x \in \Gamma_D, \tag{2.7b}$$

$$n \cdot \nabla u(x) = g_N(x) \quad \forall x \in \Gamma_N, \tag{2.7c}$$

where  $\partial\Omega = \Gamma_D \cup \Gamma_N$  with disjoint  $\Gamma_D, \Gamma_N$ ,  $f: \bar{\Omega} \rightarrow \mathbb{R}$  is a sufficiently smooth function and  $g_D: \Gamma_D \rightarrow \mathbb{R}, g_N: \Gamma_N \rightarrow \mathbb{R}$  are functions to specify Dirichlet and Neumann boundary conditions.

For each node  $x_j \in X_\Omega \cup \Gamma_N$ ,  $j \in \{1, \dots, N_I\}$ , we compute a stencil  $X_j \subset X$  of size  $n$  (2.1) and stencil weights  $\{w_1^j, \dots, w_n^j\}$  by solving (2.3). The stencil weights are used to compute approximations  $u_i$  to the solution  $u(x_i)$  of (2.7a), (2.7b), and (2.7c) at all nodes  $x_i \in X_\Omega \cup X_{\Gamma_N}$ , i.e.,  $u_i \approx u(x_i)$  for all  $x_i \in X_\Omega \cup X_{\Gamma_N}, i \in \{1, \dots, N_I\}$ , by enforcing equality in (2.2), i.e.,

$$\sum_{i=1}^n w_i^j u_{s_i^j} = f(x_j), \quad j \in \{1, \dots, N_I\}. \tag{2.8}$$

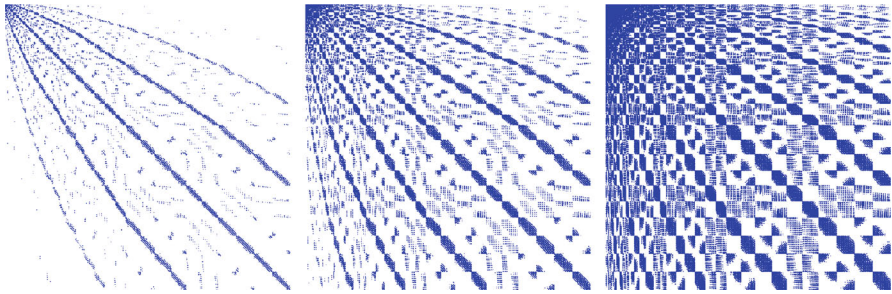
Given the Dirichlet boundary data from (2.7b), we set  $u_{s_i^j} := g(x_{s_i^j})$  for all  $x_{s_i^j} \in \Gamma_D$ , substitute this boundary information into (2.8) and obtain

$$\sum_{\substack{i \in \{1, \dots, n\} \\ \text{s.t. } x_{s_i^j} \notin \Gamma_D}} w_i^j u_{s_i^j} = f(x_j) - \underbrace{\sum_{\substack{i \in \{1, \dots, n\} \\ \text{s.t. } x_{s_i^j} \in \Gamma_D}} w_i^j g(x_{s_i^j})}_{=: \tilde{f}_j}, \quad j \in \{1, \dots, N_I\}.$$

Hence, we define the right-hand side vector  $\tilde{f} := (\tilde{f}_1, \dots, \tilde{f}_{N_I})^T \in \mathbb{R}^{N_I}$  and the global stiffness matrix  $\tilde{B} \in \mathbb{R}^{N_I, N_I + N_G}$  where  $N_G := \#X_G = \#X_G$  denotes the number of ghost nodes, containing row-wise the weights of the (non-Dirichlet) nodes of the  $j$ 'th stencil,

$$\tilde{B}_{j,\ell} := \begin{cases} w_i^j & : \exists i \in \{1, \dots, n\} \text{ such that } \ell = s_i^j \in \mathcal{I}_j, \\ & \text{i.e., the non-Dirichlet node } x_\ell = x_{s_i^j} \text{ is in the stencil } X_j, \\ 0 & : \text{else,} \end{cases} \tag{2.9}$$

for all  $j \in \{1, \dots, N_I\}, \ell \in \{1, \dots, N_I + N_G\}$ . The sparsity structure of this matrix is illustrated in Fig. 1 for several combinations of polynomial degree  $\ell$  and stencil size  $n$  and without using ghost nodes, i.e.,  $X_G = \emptyset$ .



**Fig. 1** Sparsity structures of the stiffness matrices  $B$  (2.9) for  $N_I = 1000$  nodes ( $N = 1728$ ), polynomial degrees  $\ell = 2, 5, 8$  (from left to right), and stencil sizes via (2.4)

In the presence of a Neumann boundary ( $\Gamma_N \neq \emptyset$  in (2.7c)), we compute weights to discretize the (left-hand side of the) Neumann boundary condition (2.7c) at each node  $x_j \in \Gamma_N$  by solving (2.3) with an adjusted right-hand side according to the Neumann differential operator. Together with the right-hand side in (2.7c), we obtain another  $N_G$  equations, resulting in the linear system

$$Bu = \tilde{f} \tag{2.10}$$

with  $B \in \mathbb{R}^{N_I+N_G, N_I+N_G}$  whose solution  $u \in \mathbb{R}^{N_I+N_G}$  yields the RBF-FD approximation  $u_j \approx u(x_j)$  to the solution of the partial differential equation (2.7a), (2.7b), (2.7c) at the nodes  $x_j$  for  $j \in \{1, \dots, N_I\}$  (as well as values on the ghost nodes).

**Remark 2.2** We have introduced the RBF-FD approximation using polyharmonic splines (PHS) in combination with high-order polynomials since this has been suggested in previous papers [3–5, 10, 11, 18, 38, 39]. However, the preconditioners that we will propose in the following section are applicable to RBF-FD discretization using any other kernel function as well. In fact, for a given discretization accuracy, stencil sizes do not differ much between different kernel functions such that the sparsity patterns of the resulting stiffness matrices are comparable [11].

### 3 Smaller stencil preconditioners

In this section, we propose a novel type of preconditioner for the iterative solution of the linear system (2.10) that results from an RBF-FD discretization. In Sect. 3.1, we motivate and introduce the general framework of a smaller stencil preconditioner. In the following two subsections, we then discuss particular implementations based on sparse (Sect. 3.2) or hierarchical matrix (Sect. 3.3) incomplete LU factorizations.

#### 3.1 General framework of smaller stencil preconditioners

The stencil size  $n$  determines the (maximum) number of nonzero entries per row in the system matrix  $B$  defined in (2.9) and (2.10). According to Table 1, stencil sizes may be

significantly larger than typical finite difference stencils (such as 7-point or 27-point stencils in three spatial dimensions). Considering the construction of a preconditioner for this (not so sparse) matrix  $B$ , we face two problems:

- Setup times for preconditioners as well as the computational cost to apply these preconditioners will typically increase due to the larger number of nonzero entries.
- Preconditioners have to be computed at higher accuracies in view of the worse conditioned stiffness matrices (matrices resulting from higher-order discretization are typically worse conditioned than those from lower order discretization—this is observed across various discretization schemes). This again increases setup and application cost of the preconditioners.

Our proposed preconditioner is described fairly simple. In order to indicate the polynomial order  $\ell$  that has been used to obtain the matrix  $B$  in (2.10), we now denote it by  $B_\ell$ . Let  $B_{\ell'}$  be the matrix that results from using some lower polynomial order  $\ell' < \ell$  instead (with  $D \leq \ell'$  to ensure convergence of the RBF-FD approximation of an operator of order  $D$  [10, 24]). Following Table 1,  $B_{\ell'}$  will have (significantly) fewer nonzero entries compared to  $B_\ell$ . We propose to construct a preconditioner  $P_{\ell'}$  for  $B_{\ell'}$  and then use a Krylov iterative solver for the preconditioned system  $P_{\ell'}^{-1} B_{\ell'} u = P_{\ell'}^{-1} \tilde{f}$  (or for  $B_\ell P_{\ell'}^{-1} \tilde{u} = \tilde{f}$ ,  $u = P_{\ell'}^{-1} \tilde{u}$  in case of preconditioning from the right) (2.10). We will now discuss two particular choices for  $P_{\ell'}$  based on incomplete LU factorizations of  $B_{\ell'}$ .

### 3.2 Smaller stencil ILU preconditioners

An incomplete LU (ILU) factorization decomposes a matrix  $B$  into  $B = LU + R$  where  $L$  is a unit lower triangular matrix,  $U$  is an upper triangular matrix, and  $R = (r_{ij})$  denotes a remainder matrix. An exact (i.e., complete) LU factorization is given when  $R = 0$ . This, however, typically implies substantial fill-in (additional nonzero entries) in the matrices  $L$  and  $U$  compared to the (sparse) matrix  $B$  and is of computational complexity  $\mathcal{O}(N^3)$ . The idea of an incomplete LU factorization is to allow only a reduced number of fill-ins and enforce  $r_{ij} = 0$  only for the positions of fill-in in  $L$  and  $U$ . The basic ILU(0) factorization allows no additional fill-in besides the nonzero entries already present in  $B$ . There exist various strategies to select entries for additional fill-in such as ILU(p) which uses the (Gauss) elimination graph or ILUT which applies additional threshold strategies based on the magnitude and number of entries per matrix row.

(Incomplete) LU factorizations depend on the ordering of the row and column indices. The two most popular orderings in the context of LU factorizations are the reverse Cuthill McKee (RCM) and the nested dissection (ND) ordering. RCM aims to reduce the bandwidth, i.e., it reorders nonzero entries close to the diagonal since any fill-in in the LU factorization occurs only within the bandwidth of the input matrix. ND ordering first divides the index set into three subsets, two so-called domain index subsets  $D_1$  and  $D_2$  that satisfy  $b_{ij} = 0$  for all  $(i, j) \in D_1 \times D_2$  and  $(i, j) \in D_2 \times D_1$ , i.e., the two respective matrix subblocks are zero blocks, and a so-called separator index subset  $S$  containing the remaining indices. The indices in  $D_1$  are ordered first,

the ones in  $D_2$  second, and those in  $S$  last. The zero blocks corresponding to  $D_1 \times D_2$  and  $D_2 \times D_1$  in the reordered matrix remain zero blocks during an LU factorization. This ordering is applied recursively to the two domain index sets  $D_1, D_2$ .

We refer to [37] for further details on ILU factorizations, suitable reorderings, and iterative solvers in general.

### 3.3 Smaller stencil $\mathcal{H}$ -LU preconditioners

Hierarchical ( $\mathcal{H}$ -) matrices are a class of matrices with a hierarchical subdivision of the block structure into so-called (large) admissible and (small) inadmissible blocks [17]. The matrix data in the admissible blocks is given in factored low-rank form  $B|_{s \times t} = UV^T$  for matrices  $U \in \mathbb{R}^{\#s, k}, V \in \mathbb{R}^{\#t, k}$  with  $k \ll \min\{\#s, \#t\}$  while inadmissible blocks store all their entries explicitly in standard dense format. The distinction into admissible and inadmissible blocks typically takes the source of the matrix data into account. In the case of a sparse matrix (which includes RBF-FD generated stiffness matrices), one may use a so-called weak admissibility condition where blocks are admissible as long as they have no nonzero entries.

The storage and computational complexity of the resulting  $\mathcal{H}$ -matrix depends on the particular block subdivision and the ranks within the admissible blocks. For many matrix classes from applications, it can be shown that there exist highly accurate  $\mathcal{H}$ -matrix approximations of complexity  $\mathcal{O}(N \log^\alpha N)$  (with  $\alpha \in \{1, 2\}$ ).  $\mathcal{H}$ -arithmetic can be used to compute an approximate (block) LU factorization. Whereas the ILU factorization in Sect. 3.2 enforces certain matrix entries in  $L, U$  to be zero, the  $\mathcal{H}$ -LU factorization enforces certain matrix subblocks in  $L, U$  to be of low rank.

The construction of hierarchical matrix approximations typically includes a reordering of the indices in order to obtain preferably large subblocks of low ranks. For a subsequent  $\mathcal{H}$ -LU factorization, the  $\mathcal{H}$ -block structure can be aligned with a nested dissection ordering such that off-diagonal blocks corresponding to different index domain sets remain zero in the LU factorization.

Without going into further details, we just point out one important feature of  $\mathcal{H}$ -matrices that we will exploit: One can prescribe an accuracy  $\delta_{\mathcal{H}}$  that bounds the relative error of the (low-rank) truncations in  $\mathcal{H}$ -arithmetic. Decreasing  $\delta_{\mathcal{H}}$  allows to construct more accurate factorizations, typically at only moderate additional cost. Hence, we will be able to analyze the importance of the accuracy of the  $\mathcal{H}$ -LU factorization of  $B_{\ell'}$  for the performance as a preconditioner for  $B_{\ell}$ .

For details on the theory and implementation of hierarchical matrices such as the parameter  $n_{\min}$  (which influences the maximal size of the full matrix blocks) and in particular nested dissection based  $\mathcal{H}$ -LU factorization, we refer to [15–17]. The construction of  $\mathcal{H}$ -LU factorizations of the original system  $B_{\ell}$  as preconditioners has been developed in detail in [25]. We are not aware of any other literature that uses  $\mathcal{H}$ -matrices in the solution of RBF-FD systems.

In the following, we briefly list a few aspects to consider when applying  $\mathcal{H}$ -arithmetic to matrices arising from RBF-FD discretization (as opposed to finite difference or finite element discretization).

- The RBF-FD system matrix  $B$  typically has a non-symmetric sparsity pattern. We use the matrix graph of the symmetrized matrix  $B + B^T$  to determine the nested dissection ordering and admissibility of blocks.
- The (nested dissection) index partitioning is stopped when an index set cannot be subdivided into three (nontrivial) subsets where a separator subset  $S$  separates the other two domain subsets  $D_1, D_2$  in the sense that  $b_{ij} = 0$  for all  $i \in D_1, j \in D_2$  (i.e., the matrix block corresponding to row indices in  $D_1$  and column indices in  $D_2$  is a zero block). For larger stencil sizes as in RBF-FD, the index partitioning typically stops for index sets that could still be partitioned for smaller stencil sizes or in a finite difference or finite element discretization.
- Using the weak admissibility condition, there exist index block clusters that are inadmissible for large stencil discretization but admissible for smaller stencil (or FD/FEM) discretization.
- There exist index block clusters that are inadmissible for both large as well as smaller stencil discretizations where, for smaller stencils, the block can be subdivided such that some of its subblocks become admissible whereas such a subdivision does not exist for the large stencil matrix. The latter hence has to store the respective large block as a full matrix.

## 4 Numerical results

This section contains numerical tests for several smaller stencil preconditioner setups for the iterative solution of the linear systems of equations arising in an RBF-FD discretization of the convection-diffusion equation in  $d = 3$  spatial dimensions on two types of domain, a unit cube  $\Omega = (0, 1)^3$  and a bunny-shaped domain (scaled to a size comparable to the unit cube) shown in Fig. 2.

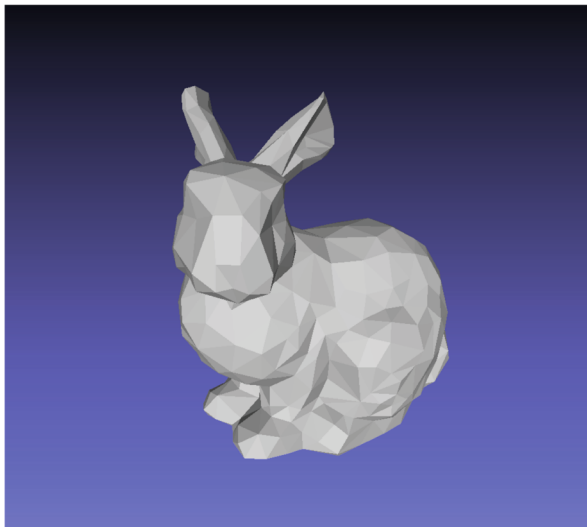


Fig. 2 Bunny domain (data from <https://github.com/OpenGP/OpenGP/blob/master/data/bunny.off>)

In (2.7a), we set  $\mathcal{L} = -\nu\Delta + b^T\nabla$  with  $\nu = 10^{-3}$ . The function  $b$  is the convective flow from problem 3D1 in [32], namely  $b: \mathbb{R}^3 \rightarrow \mathbb{R}^3$  with

$$b(x) = \begin{pmatrix} 2x_1(1-x_1)(2x_2-1)x_3 \\ -(2x_1-1)x_2(1-x_2) \\ -(2x_1-1)(2x_2-1)x_3(1-x_3) \end{pmatrix}.$$

We choose the right-hand side functions  $f, g_D, g_N$  in (2.7a), (2.7b), and (2.7c) such that the analytical solution  $u$  is given by the three-dimensional version of Franke’s function  $F: \mathbb{R}^3 \rightarrow \mathbb{R}$  [22, 35] with

$$\begin{aligned} F(x) = & 0.75 \exp[-(9x_1-2)^2 + (9x_2-2)^2 + (9x_3-2)^2] / 4] \\ & + 0.75 \exp[-(9x_1+1)^2 / 49 - (9x_2+1) / 10 - (9x_3+1) / 10] \\ & + 0.5 \exp[-(9x_1-7)^2 + (9x_2-3)^2 + (9x_3-5)^2] / 4] \\ & - 0.2 \exp[-(9x_1-4)^2 - (9x_2-7)^2 - (9x_3-5)^2]. \end{aligned} \tag{4.1}$$

In Sect. 4.1, we present some general information about the setting of the numerical tests such as node and stencil generation as well as the computation of the approximation error.

In Sect. 4.2, we present numerical results for smaller stencil ILU preconditioners whereas in Sect. 4.3, we present numerical results for smaller stencil  $\mathcal{H}$ -LU preconditioners. In particular, we will show numerical results for the following settings:

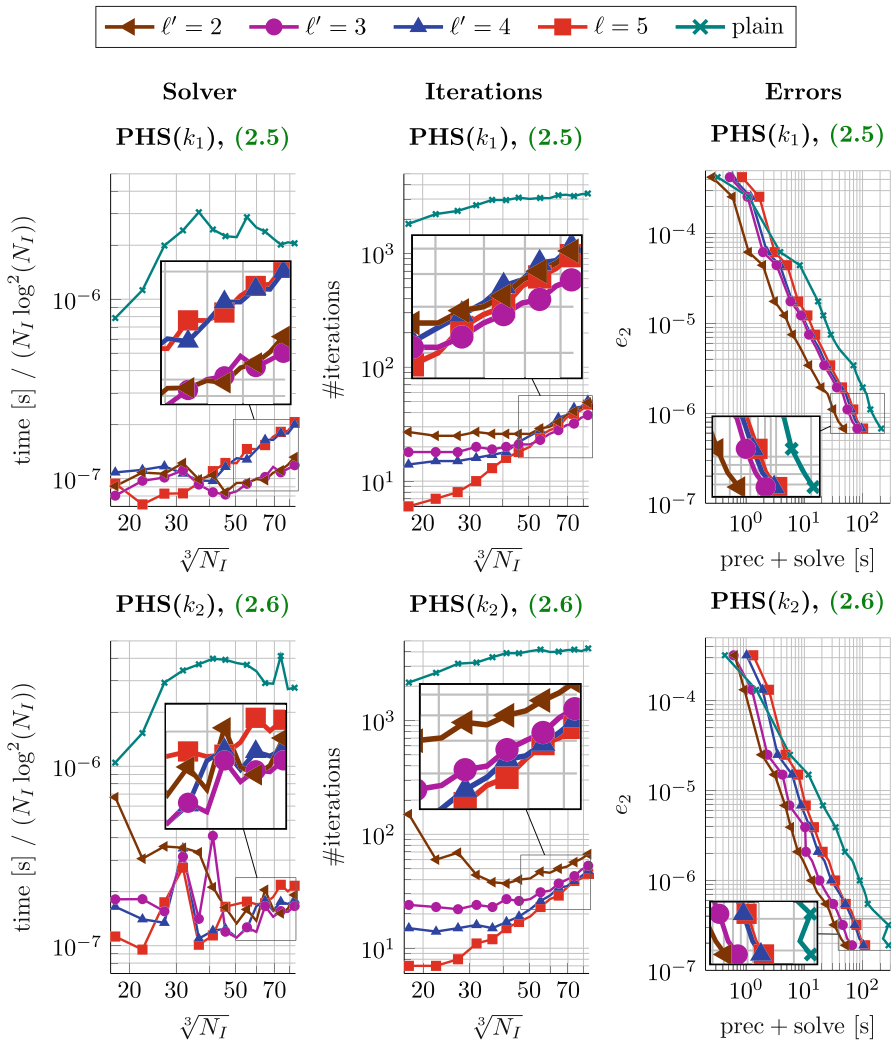
- A cube domain with Dirichlet boundary conditions (see Figs. 3, 6, and 8)
- A cube domain with mixed boundary conditions (Neumann on left and right faces, Dirichlet on front, back, top, and bottom faces) (see Fig. 4)
- A bunny domain with Dirichlet boundary conditions (see Figs. 5, 7, and 8)

### 4.1 General information

For the cube, the number of interior (and boundary) nodes is chosen to be the (approximately) the same as in a typical Cartesian grid, i.e.,  $N_I \approx m^d$  interior nodes and  $N \approx (m+2)^d$  total nodes for some  $m \in \mathbb{N}$ . In the interior of the cube, we generate node distributions using PNP node generation [44] (created by [41]). On the cube’s boundary, we use Cartesian nodes. We perform tests on node distributions with up to  $N \approx 86^3 = 636,056$  total nodes ( $N_I \approx 84^3 = 592,704$  interior nodes).

The bunny boundary is a polyhedron consisting of triangles. We use equidistant nodes on the edges of the triangles and generate nodes inside the triangles as well as in the interior of the bunny using PNP. The numbers  $N_\Omega, N_\Gamma$  of interior and boundary nodes, resp., are selected such that there holds  $N_\Gamma^{1/2} / N_\Omega^{1/3} \approx 2$ . We perform tests on node distributions with up to  $N = 545,229$  total nodes ( $N_I = 517,610$ ).

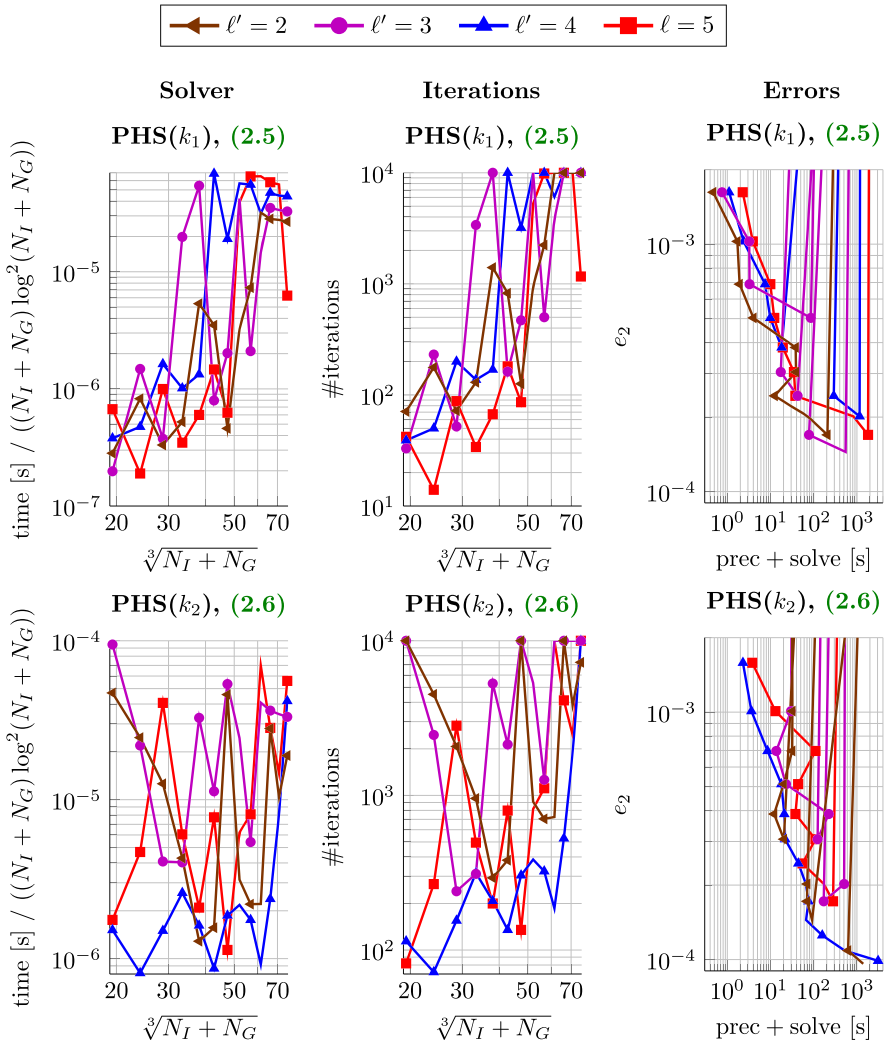
We use direct solvers based on LU factorizations with pivoting for the computation of the stencil weights (2.3).



**Fig. 3** Smaller stencil ILUT preconditioners for a system with polynomial degree  $\ell = 5$  on the cube with Dirichlet boundary conditions, without preconditioner (“plain”) or with preconditioners with  $\ell' \in \{2, 3, 4\}$  or  $\ell = 5$ , PHS degrees via  $k_1$  (2.5) (top row) and  $k_2$  (2.6) (bottom row). Left: scaled iteration times (in seconds). Middle: number of iterations w. r. t. the number of nodes. Right: approximation errors (4.2) w. r. t. the computation time (without setup time of the nodes and the linear system)

We will perform tests using a preconditioned BiCGstab solver with the zero vector as a starting vector. We stop the iteration if the relative residual satisfies  $\|r^k\|_2 / \|r^0\|_2 \leq 10^{-13}$  or a maximum number of  $10^4$  iterations has been reached.

In order to measure the approximation accuracy of the RBF-FD computed discrete solution, we determine the relative 2-norm error between the exact solution (4.1) of the

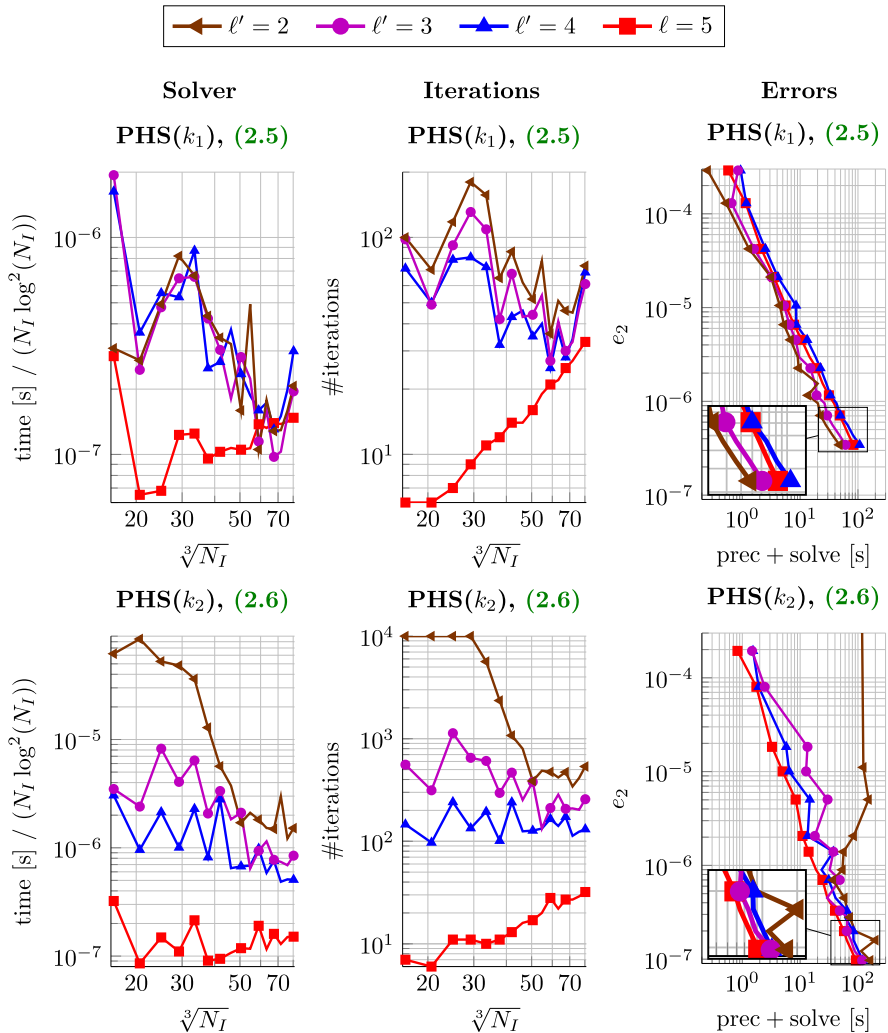


**Fig. 4** Smaller stencil ILUT preconditioners for a system with polynomial degree  $\ell = 5$  on the cube domain with mixed boundary conditions, with preconditioners with  $\ell' \in \{2, 3, 4\}$  or  $\ell' = 5$ , PHS degrees via  $k_1$  (2.5) (top row) and  $k_2$  (2.6) (bottom row). Left: scaled iteration times (in seconds). Middle: number of iterations w. r. t. the number of nodes. Right: approximation errors (4.2) w. r. t. the computation time (without setup time of the nodes and the linear system)

PDE (2.7a), (2.7b), (2.7c) evaluated at the nodes,  $u_s := (F(x_j))_{j=1}^{N_I}$ , and the computed solution  $u$  obtained by iteratively solving the system (2.10),

$$e_2 := \frac{\|u_s - u\|_2}{\|u_s\|_2}. \tag{4.2}$$

In particular, we do not distinguish between discretization and iteration errors in view of our stopping criterion: If the relative residual accuracy has been reached, the error



**Fig. 5** Smaller stencil ILUT preconditioners for a system with polynomial degree  $\ell = 5$  on the bunny domain with Dirichlet boundary conditions, with preconditioners with  $\ell' \in \{2, 3, 4\}$  or  $\ell = 5$ , PHS degrees via  $k_1$  (2.5) (top row) and  $k_2$  (2.6) (bottom row). Left: scaled iteration times (in seconds). Middle: number of iterations w.r.t. the number of nodes. Right: approximation errors (4.2) w.r.t. the computation time (without setup time of the nodes and the linear system)

$e_2$  is dominated by the discretization error (see also [24]). If the maximum number of iterations has been reached, the error  $e_2$  is dominated by the iteration error (and indicates divergence of the iterative solver).

In our plots, we will display timings (in seconds) for the setup time of the preconditioners as well as for the iterative solution of the systems. Since the best asymptotic complexity (of the overall solution time) that we observe w.r.t. the system size  $N_I$  is  $O(N_I \log^2 N_I)$ , we have scaled all timings by  $N_I \log^2 N_I$ .

## 4.2 Numerical results for smaller stencil ILU preconditioners

In this subsection, we present numerical results for smaller stencil ILU preconditioners in a right preconditioned BiCGstab solver. Here, we have used the Medusa library [43] for the RBF-FD discretizations. Before the construction of an ILU preconditioner, the nodes are reordered with a fill-reducing nested dissection ordering. For this, we use the routine METIS\_NodeND of the METIS reordering package [19] where all parameters have been set to their default options. The ILUT factorizations have been computed with a drop tolerance of 0.001 and a fill factor of 10 (see [36] for further details on ILUT and its parameters). The computation time measurements in this subsection were done for computations in double precision on twelve cores of a computer with an *Intel(R) Core i7-1365U @ 5.20 GHz* processor and 32 GB of LPDDR5 memory.

Figure 3 illustrates the performance of smaller stencil ILUT preconditioners on a cube domain with Dirichlet boundary conditions. In the iterative solution of an RBF-FD system obtained by a discretization with degree  $\ell = 5$ , we use the plain BiCGstab solver without any preconditioner or with ILUT decompositions of (auxiliary) discretizations with polynomial degrees  $\ell' \in \{2, 3, 4\}$  or  $\ell = 5$  as preconditioners. The stencil sizes are given by (2.4), and the PHS degrees are given by  $k_1$  (2.5) in the top row and by  $k_2$  (2.6) in the bottom row. Each color/marker represents a different degree of the (auxiliary) discretization.

We start with a discussion of the results in the top row where we used PHS( $k_1$ ). The left plot shows the scaled iteration times (in seconds). For larger problems sizes, the lowest times correspond to the polynomial degrees  $\ell' \in \{2, 3\}$  whereas larger degrees require larger solution times. The middle plot shows the number of iterations and demonstrates that for smaller problem sizes, a larger polynomial degree corresponds to a smaller number of iterations. However, since the cost per iteration step increases with the degree, the largest degree does not necessarily imply the smallest iteration time as was seen in the left plot. The right plot illustrates the approximation errors (4.2) w. r. t. the (unscaled) computation times (in seconds) which now include both the setup times for the preconditioner (which includes the reordering) and the iteration times. The markers correspond to different problem sizes ranging from  $N_I \approx 18^3 = 5832$  to  $N_I \approx 84^3 = 592,704$  interior nodes. This plot demonstrates that the fastest method is obtained by using a preconditioner based on degree  $\ell' = 2$  whereas the slowest solution (aside from the plain solver) corresponds to degree  $\ell = 5$ . The conclusion from these numerical tests is that smaller stencil preconditioners have smaller setup cost and can lead to smaller iteration times where the lowest total time to reach a certain accuracy has been obtained for  $\ell' = 2$ .

In the bottom row, we show results now using larger PHS degrees  $k_2$  (2.6) instead of  $k_1$  (2.5), leading to systems of higher condition numbers which are in general more difficult to solve iteratively. Especially for discretizations with a small number of interior nodes  $N_I$ , both the iteration times and number of iterations are now often higher. For the smallest tested problem sizes, the plain BiCGstab solver even leads to the lowest solution times (see top left area of bottom right plot). However, as the problem size increases, the plain BiCGstab as well as the preconditioners for larger  $\ell'$  typically lead to the highest overall computation times whereas the lowest times are

again obtained for preconditioners for  $\ell' = 2$ . The bottom right plot also shows that we now may reach errors of approximately  $10^{-7}$ . Hence, for worse conditioned systems obtained with  $k = k_2$ , smaller stencil ILUT preconditioners are also advantageous.

In Fig. 4, we have performed numerical tests on the cube with mixed boundary conditions (Neumann on left and right faces, Dirichlet on all other faces). Here, as long as preconditioners with  $\ell' = 2$  converge (i.e., stop after reaching the relative residual stopping criterion and not the maximum number of iterations), they are still among the fastest methods to reach a certain accuracy. However, especially for larger problem sizes, preconditioners with  $\ell' = 2$  often fail to converge. For this type of problem, one may have to adjust the ILUT parameters (lower the drop tolerance, increase the fill factor) or design entirely different types of preconditioners.

In Fig. 5, we show numerical results for tests performed on the bunny domain with Dirichlet boundary conditions. For  $k_1$  (2.5) (top row), the results are qualitatively comparable to those on the cube shown in Fig. 3: Increasing  $\ell'$  leads to smaller iteration numbers, but setup times and times per iteration step are increased such that the most efficient solver to reach a certain accuracy is still given by  $\ell' = 2$ . However, the situation is different when using  $k_2$  (2.6) (bottom row). Here, the significantly smaller number of iterations for  $\ell = 5$  compensates for the larger setup time, making it the most efficient solver.

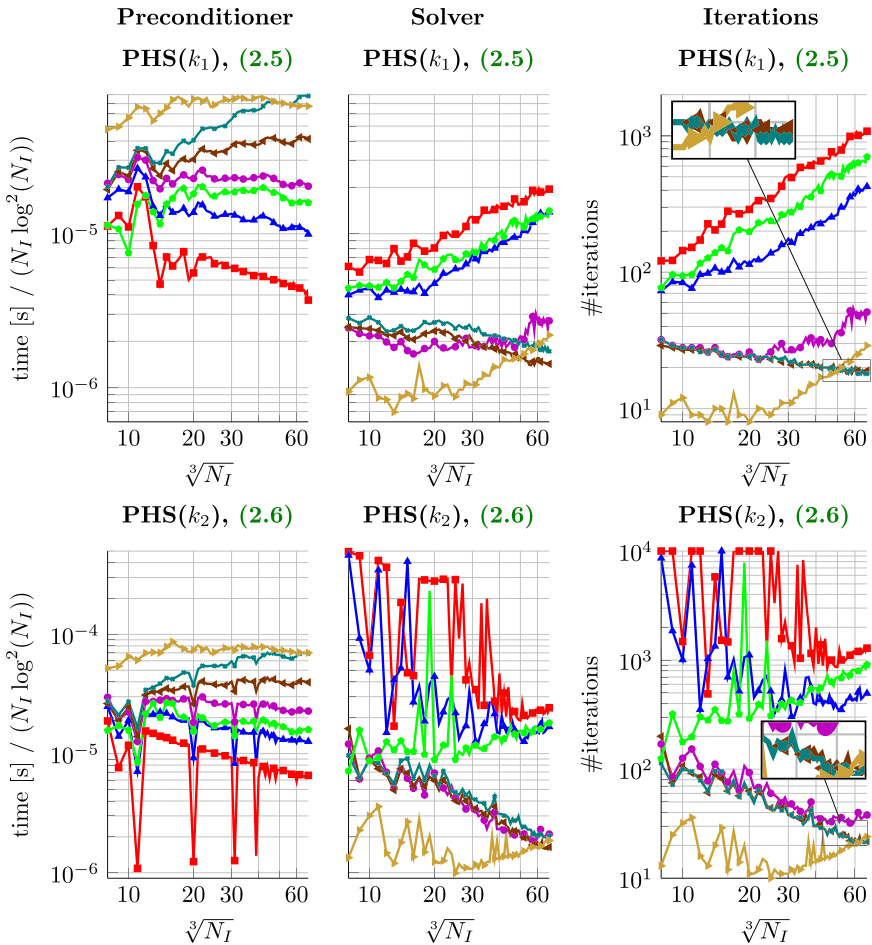
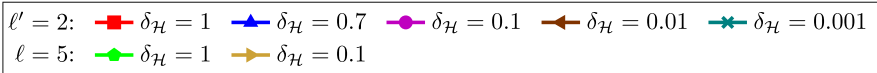
### 4.3 Numerical results for smaller stencil $\mathcal{H}$ -LU preconditioners

In this subsection, we show numerical results obtained for smaller stencil  $\mathcal{H}$ -LU preconditioners in a left preconditioned BiCGstab solver on the cube and bunny domains with Dirichlet boundary conditions. Here, the RBF-FD discretization is based to a large extent on our own implementation. For the computation of stencils, we use the ANN library [31] that utilizes a kd-tree for the nearest neighbor search w. r. t. the norm  $\|\cdot\|_\mu$  for some  $\mu \in \mathbb{N}$ . In our experiments, we use the Euclidean norm by setting  $\mu = 2$ . We use the H2Lib library [7] when working with hierarchical matrices. The computation time measurements in this subsection were done for computations in double precision on a single core of a computer with an *Intel(R) Xeon(R) CPU E5-2665 0 @ 2.40 GHz* processor and 16 GB of DDR3 memory.

Figure 6 illustrates the performance of  $\mathcal{H}$ -LU preconditioners with degrees  $\ell' = 2$  or  $\ell = 5$  for discretizations with polynomial degree  $\ell = 5$  and up to  $N_I = 68^3 = 314432$  interior nodes on the cube. The stencil sizes are given by (2.4), and the PHS degrees are determined as  $k_1$  (2.5) (top row) or  $k_2$  (2.6) (bottom row). Each color/marker represents a different combination of degree  $\ell'$  or  $\ell$  of the preconditioner and truncation tolerance  $\delta_{\mathcal{H}}$  used in the  $\mathcal{H}$ -arithmetic (i.e., the smaller the tolerance, the more accurate becomes the  $\mathcal{H}$ -LU factorization, see [17] for details).

The first five markers correspond to preconditioners of degree  $\ell' = 2$ , tolerances  $\delta_{\mathcal{H}} \in \{1, 0.7, 0.1, 0.01, 0.001\}$  and  $n_{\min} = 35$ , whereas the last two markers represent results for preconditioners based on the original system of degree  $\ell = 5$ , tolerances  $\delta_{\mathcal{H}} \in \{1, 0.1\}$  and  $n_{\min} = 50$ .

We first discuss the results shown in the top row. The left plot illustrates that the setup times for the  $\mathcal{H}$ -LU factorization of the auxiliary stiffness matrix for polynomial



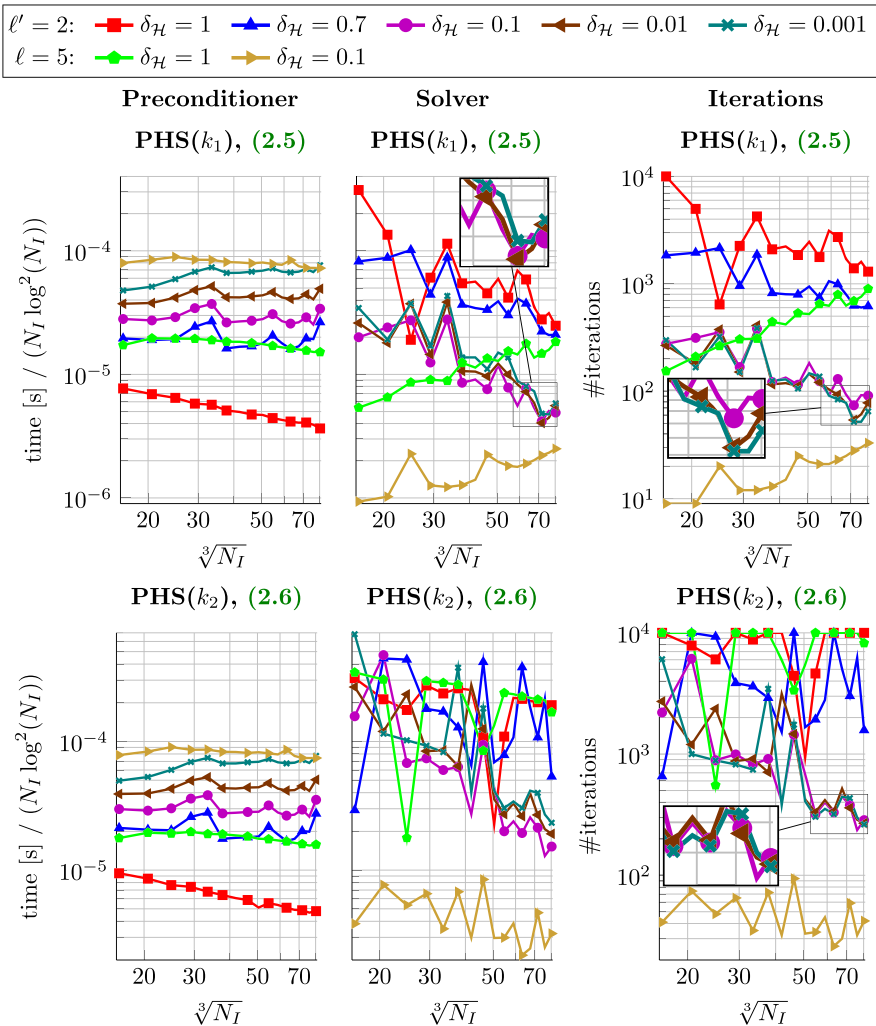
**Fig. 6** Smaller stencil  $\mathcal{H}$ -LU preconditioners on the cube: (Scaled) setup times (left), iteration times (middle), and number of iterations (right) w.r.t. the number of nodes for a system with polynomial degree  $\ell = 5$ . The first five markers show results for preconditioners with  $\ell' = 2$ , tolerances  $\delta_{\mathcal{H}} \in \{1, 0.7, 0.1, 0.01, 0.001\}$ , and  $n_{\min} = 35$ . The last two markers show results for preconditioners with  $\ell = 5$ , tolerances  $\delta_{\mathcal{H}} \in \{1, 0.1\}$ , and  $n_{\min} = 50$ . PHS degrees are set to  $k_1$  (2.5) (top row) and  $k_2$  (2.6) (bottom row)

degree  $\ell' = 2$  (which includes the setup times of this matrix) are typically significantly lower than the setup time of the  $\mathcal{H}$ -LU factorization of the original stiffness matrix (using  $\delta_{\mathcal{H}} = 0.1$ ). However, higher setup times (by decreasing  $\delta_{\mathcal{H}}$  or using the original system) typically result in lower computational times for the iterative solver (middle) and smaller iteration counts (right).

In the bottom row, we report our results for using the higher PHS degree  $k_2$ . It can be observed that using the smaller stencil preconditioner with a low accuracy  $\mathcal{H}$ -LU

factorization leads to instabilities whereas there are hardly any differences in iteration time and number of iterations when using accuracies  $\delta_{\mathcal{H}} \in \{0.1, 0.01, 0.001\}$ .

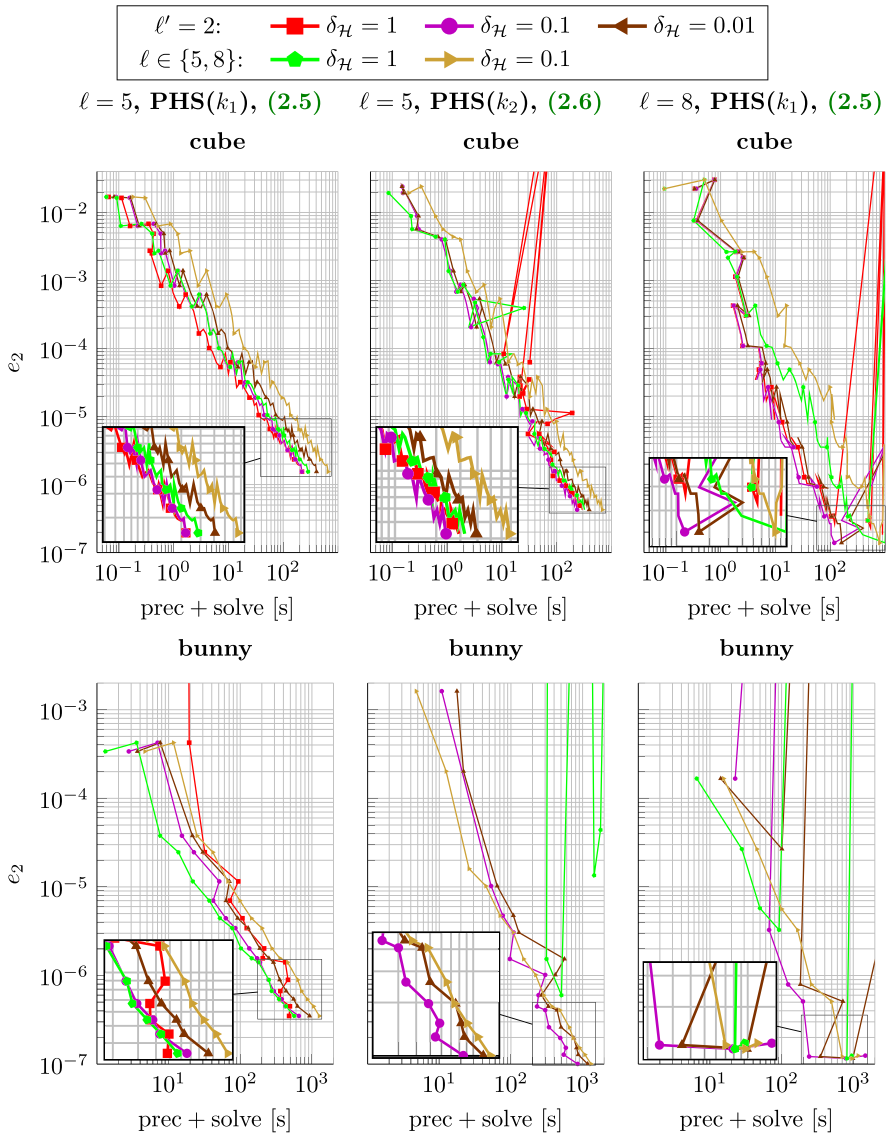
Figure 7 illustrates the respective numerical results on the bunny instead of the cube domain. The results are qualitatively similar:  $\ell = 5$  with  $\delta_{\mathcal{H}} = 0.1$  leads to the largest setup times and smallest solver times and iteration numbers. Second best (w.r.t. solver times and iteration numbers) are  $\ell' = 2$  with  $\delta_{\mathcal{H}} \in \{0.1, 0.001, 0.0001\}$  whereas less accurate  $\mathcal{H}$ -LU factorizations ( $\delta_{\mathcal{H}} \geq 0.7$ ) now often lead to divergence.



**Fig. 7** Smaller stencil  $\mathcal{H}$ -LU preconditioners on the bunny domain: (Scaled) setup times (left), iteration times (middle), and number of iterations (right) w.r.t. the number of nodes for a system with polynomial degree  $\ell = 5$ . The first five markers show results for preconditioners with  $\ell' = 2$ , tolerances  $\delta_{\mathcal{H}} \in \{1, 0.7, 0.1, 0.01, 0.001\}$ , and  $n_{\min} = 35$ . The last two markers show results for preconditioners with  $\ell = 5$ , tolerances  $\delta_{\mathcal{H}} \in \{1, 0.1\}$ , and  $n_{\min} = 35$ . PHS degrees are set to  $k_1$  (2.5) (top row) and  $k_2$  (2.6) (bottom row)

Plots that relate the approximation error  $e_2$  (4.2) of the iterative solution to the computational time of both the preconditioner setup and the iteration (as in the right columns of Figs. 3, 4, 5) are shown in Fig. 8 for both the cube (top row) and bunny domain (bottom row).

The left and middle plots correspond to the cases tested in Figs. 6 and 7 (we left out the tolerances  $\delta_{\mathcal{H}} \in \{0.7, 0.001\}$  for better visibility). The right plot shows tests



**Fig. 8** Approximation errors  $e_2$  (4.2) w. r. t. the computation time (in seconds) for systems with polynomial degrees  $\ell \in \{5, 8\}$ ,  $\mathcal{H}$ -LU preconditioners with  $\ell' = 2$  or  $\ell \in \{5, 8\}$ ,  $n_{\min} \in \{35, 50, 70\}$ , tolerances  $\delta_{\mathcal{H}} \in \{1, 0.1, 0.01\}$ , and PHS degrees  $k_1$  (2.5) and  $k_2$  (2.6)

for an original system of polynomial degree  $\ell = 8$  with problem sizes ranging from  $N_I = 8^3 = 512$  to  $N_I = 51^3 = 132651$ , stencil size via (2.4), PHS degree  $k_1$  (2.5), and preconditioners with  $\ell' = 2$  (and  $n_{\min} = 35$ ) or  $\ell = 8$  (and  $n_{\min} = 70$ ). Especially in the case of polynomial degree  $\ell = 8$  (which is appealing if the desired approximation accuracy is high [18]), smaller stencil preconditioners can reach solutions of an accuracy  $e_2 \in (10^{-7}, 10^{-3})$  in significantly lower computational time compared to preconditioners based on the original system.

## 5 Conclusion and outlook

We introduced and numerically illustrated preconditioners for RBF-FD systems (2.9) based on (sparse or hierarchically structured) incomplete factorizations of auxiliary stiffness matrices based on RBF-FD discretization with smaller stencil sizes. The setup times of these smaller stencil preconditioners (including the cost to set up the auxiliary stiffness matrix) can be significantly smaller than computing respective factorizations for the original system. In the case of PHS( $k$ ) with the smaller  $k = k_1$ , the reduced setup times more than compensate for a (moderate) increase in iteration steps, leading to overall time savings to reach the prescribed accuracy. In the case of PHS( $k$ ) with the larger  $k = k_2$ , the systems can be highly ill-conditioned and this has a negative effect especially on low-accuracy preconditioners.

We have included ILU preconditioners in our study since they are well-established and already included in many (numerical linear algebra) software packages. However, as long as the (sparse) LU factorizations are incomplete, there exist no error bounds for the quality of the factorizations (except for some very special model problems). Hence, we also included  $\mathcal{H}$ -LU preconditioners where we can control the accuracy through the parameter  $\delta_{\mathcal{H}}$ . This allowed us to study the difference between less and highly accurate  $\mathcal{H}$ -LU factorizations of smaller stencil matrices as preconditioners. In a direct comparison between these two preconditioners, the ILU preconditioners are almost always superior due to the much larger setup costs of  $\mathcal{H}$ -LU factorizations. Only for highly ill-conditioned systems where ILU preconditioners fail (or if systems have to be solved for multiple right-hand sides), it may be advisable to use an  $\mathcal{H}$ -LU preconditioner.

In conclusion, smaller stencil preconditioners provide a promising option for the iterative solution of RBF-FD systems. In this work, we restricted our attention to PHS( $k$ ) with nearest neighbor stencils on PNP nodes. Future work will lie in the analysis of smaller stencil preconditioners for further generating functions in RBF-FD, stabilized versions, and other than nearest neighbor stencils.

**Author contribution** All authors S.L, M.K., and W.L. contributed to the work presented in the manuscript. M. K. and W. L. performed the numerical experiments. All authors S.L, M.K., and W.L. read and approved the final manuscript.

**Funding** Open Access funding enabled and organized by Projekt DEAL. This work was supported by the Deutsche Forschungsgemeinschaft (DFG) within the Research Training Group GRK 2583 "Modeling, Simulation and Optimization of Fluid Dynamic Applications."

**Data availability** No datasets were generated or analyzed during the current study.

## Declarations

**Ethical approval** Not applicable

**Conflict of interest** The authors declare no competing interests.

**Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

## References

1. Barik, N.B., Sekhar, T.V.S.: Multilevel meshfree RBF-FD method for elliptic partial differential equations. In: *Engineering Mathematics and Computing*, pp. 1–10. Springer, (2023)
2. Bartwal, N., Shahane, S., Roy, S., Vanka, S.P.: Application of a high order accurate meshless method to solution of heat conduction in complex geometries. [arXiv:2106.08535](https://arxiv.org/abs/2106.08535), (2021)
3. Bayona, V.: An insight into RBF-FD approximations augmented with polynomials. *Comput. Math. Appl.* **77**, 2337–2353 (2019)
4. Bayona, V., Flyer, N., Fornberg, B.: On the role of polynomials in RBF-FD approximations: III. Behavior near domain boundaries. *J. Comput. Phys.* **380**, 378–399 (2019)
5. Bayona, V., Flyer, N., Fornberg, B., Barnett, G.A.: On the role of polynomials in RBF-FD approximations: II. Numerical solution of elliptic PDEs. *J. Comput. Phys.* **332**, 257–273 (2017)
6. Bayona, V., Flyer, N., Lucas, G.M., Baumgaertner, A.J.G.: A 3-D RBF-FD solver for modeling the atmospheric global electric circuit with topography (GEC-RBFFD v1. 0). *Geosci. Model. Dev.* **8**, 3007–3020 (2015)
7. Börm, S.: H2Lib. <http://www.h2lib.org/>, (2017). Retrieved 27 Feb 2021
8. Chinchapatnam, P., Djidjeli, K., Nair, P., Tan, M.: A compact RBF-FD based meshless method for the incompressible Navier-Stokes equations. *Proc. Inst. Mech. Eng. Part M J. Eng. Marit. Environ.* **223**, 275–290 (2009)
9. Fasshauer, G.E.: *Meshfree approximation methods with Matlab*, vol. 6 of *Interdiscip. Math. Sci.*, World Scientific Publishing Company, (2007)
10. Flyer, N., Barnett, G.A., Wicker, L.J.: Enhancing finite differences with radial basis functions: experiments on the Navier-Stokes equations. *J. Comput. Phys.* **316**, 39–62 (2016)
11. Flyer, N., Fornberg, B., Bayona, V., Barnett, G.A.: On the role of polynomials in RBF-FD approximations: I. Interpolation and accuracy. *J. Comput. Phys.* **321**, 21–38 (2016)
12. Flyer, N., Wright, G.B., Fornberg, B.: Radial basis function-generated finite differences: a mesh-free method for computational geosciences. *Handbook of Geomathematics 1–30* (2014)
13. Fornberg, B., Flyer, N.: A primer on radial basis functions with applications to the geosciences, vol. 87 of *CBMS-NSF Regional Conf. Ser. in Appl. Math.*, SIAM, (2015)
14. Fornberg, B., Flyer, N.: Solving PDEs with radial basis functions. *Acta Numer.* **24**, 215–258 (2015)
15. Grasedyck, L., Kriemann, R., Le Borne, S.: Parallel black box H-LU preconditioning for elliptic boundary value problems. *Comput. Vis. Sci.* **11**, 273–291 (2008)
16. Grasedyck, L., Kriemann, R., Le Borne, S.: Domain decomposition based H-LU preconditioning. *Numer. Math.* **112**, 565–600 (2009)
17. Hackbusch, W.: *Hierarchical matrices: algorithms and analysis*, vol. 49 of *Springer Ser. Comput. Math.*, Springer (2015)
18. Jančič, M., Slak, J., Kosec, G.: Monomial augmentation guidelines for RBF-FD from accuracy versus computational time perspective. *J. Sci. Comput.* **87**, 1–18 (2021)

19. Karypis, G.: METIS. <http://glaros.dtc.umn.edu/gkhome/metis/metis/overview>, (2023). Retrieved 23 Feb 2024
20. Kosec, G.: A local numerical solution of a fluid-flow problem on an irregular domain. *Adv. Eng. Softw.* **120**, 36–44 (2018)
21. Kriemann, R., Le Borne, S.: H-FAINV: hierarchically factored approximate inverse preconditioners. *Comput. Vis. Sci.* **17**, 135–150 (2015)
22. Lazzaro, D., Montefusco, L.B.: Radial basis functions for the multivariate interpolation of large scattered data sets. *J. Comput. Appl. Math.* **140**, 521–536 (2002)
23. Le Borne, S., Grasedyck, L.: H-matrix preconditioners in convection-dominated problems. *SIAM J. Matrix Anal. Appl.* **27**, 1172–1183 (2006)
24. Le Borne, S., Leinen, W.: Guidelines for RBF-FD discretization: numerical experiments on the interplay of a multitude of parameter choices. *J. Sci. Comput.* **95**, 8 (2023)
25. Leinen, W.: Iterative solvers for RBF-FD discretized flow problems. PhD thesis, Hamburg University of Technology, (2024)
26. Martin, B., Fornberg, B.: Using radial basis function-generated finite differences (RBF-FD) to solve heat transfer equilibrium problems in domains with interfaces. *Eng. Anal. Bound. Elem.* **79**, 38–48 (2017)
27. Mathews, N.H., Flyer, N., Gibson, S.E.: Solving 3D magnetohydrostatics with RBF-FD: applications to the solar corona. *J. Comput. Phys.* **462**, 111214 (2022)
28. Milovanović, S.: Pricing financial derivatives using radial basis function generated finite differences with polyharmonic splines on smoothly varying node layouts. [arXiv:1808.02365](https://arxiv.org/abs/1808.02365), (2018)
29. Milovanovic, S., von Sydow, L.: Radial basis function generated finite differences for option pricing problems. *Comput. Math. Appl.* **75**, 1462–1481 (2018)
30. Mohammadi, V., Dehghan, M., De Marchi, S.: Numerical simulation of a prostate tumor growth model by the RBF-FD scheme and a semi-implicit time discretization. *J. Comput. Appl. Math.* **388**, 113314 (2021)
31. Mount, D.M., Arya, S.: ANN: a library for approximate nearest neighbor searching. <http://www.cs.umd.edu/~mount/ANN/>, (2010). Retrieved 25 June 2018
32. Notay, Y.: Aggregation-based algebraic multigrid for convection-diffusion equations. *SIAM J. Sci. Comput.* **34**, A2288–A2316 (2012)
33. Oruç, Ö.: A local hybrid kernel meshless method for numerical solutions of two-dimensional fractional cable equation in neuronal dynamics. *Numer. Methods Partial Differ. Equ* **36**, 1699–1717 (2020)
34. Radhakrishnan, A., Xu, M., Shahane, S., Vanka, S.P.: A non-nested multilevel method for meshless solution of the Poisson equation in heat transfer and fluid flow. [arXiv:2104.13758](https://arxiv.org/abs/2104.13758), (2021)
35. Renka, R.J.: Multivariate interpolation of large sets of scattered data. *ACM Trans. Math. Software* **14**, 139–148 (1988)
36. Saad, Y.: ILUT: a dual threshold incomplete LU factorization. *Numer. Linear Algebra Appl.* **1**, 387–402 (1994)
37. Saad, Y.: Iterative methods for sparse linear systems, SIAM, (2003)
38. Santos, L.G.C., Manzares-Filho, N., Menon, G.J., Abreu, E.: Comparing RBF-FD approximations based on stabilized Gaussians and on polyharmonic splines with polynomials. *Internat. J. Numer. Methods Engrg.* **115**, 462–500 (2018)
39. Shankar, V., Fogelson, A.L.: Hyperviscosity-based stabilization for radial basis function-finite difference (RBF-FD) discretizations of advection-diffusion equations. *J. Comput. Phys.* **372**, 616–639 (2018)
40. Shankar, V., Wright, G.B., Fogelson, A.L.: An efficient high-order meshless method for advection-diffusion equations on time-varying irregular domains. [arXiv:2011.06715](https://arxiv.org/abs/2011.06715), (2020)
41. Slak, J., Kosec, G.: Standalone implementation of the proposed node placing algorithm. <http://e6.ijs.si/medusa/static/PNP.zip> (2021). Retrieved 11 Jan 2022
42. Slak, J., Kosec, G.: Refined meshless local strong form solution of Cauchy-Navier equation on an irregular domain. *Eng. Anal. Bound. Elem.* **100**, 3–13 (2018)
43. Slak, J., Kosec, G.: Medusa: A C++ library for solving PDEs using strong form mesh-free methods. <http://e6.ijs.si/medusa/>, (2019). Retrieved 23 Feb 2024
44. Slak, J., Kosec, G.: On generation of node distributions for meshless PDE discretizations. *SIAM J. Sci. Comput.* **41** (2019)

45. Tolstykh, A.I.: On using RBF-based differencing formulas for unstructured and mixed structured-unstructured grid calculations. In: Proceedings of the 16th IMACS world congress, vol. 228, pp. 4606–4624 (2000)
46. Tolstykh, A.I., Shirobokov, D.A.: On using radial basis functions in a finite difference mode with applications to elasticity problems. *Comput. Mech.* **33**, 68–79 (2003)
47. Tominec, I., Larsson, E., Heryudono, A.: A least squares radial basis function finite difference method with improved stability properties. *SIAM J. Sci. Comput.* **43**, A1441–A1471 (2021)
48. Wendland, H.: Scattered data approximation. Cambridge Monogr. Appl. Comput. Math., Cambridge University Press (2005)
49. Wright, G.B., Jones, A.M., Shankar, V.: MGM: a meshfree geometric multilevel method for systems arising from elliptic equations on point cloud surfaces. [arXiv:2204.06154](https://arxiv.org/abs/2204.06154), (2022)
50. Zamolo, R., Nobile, E., Šarler, B.: Novel multilevel techniques for convergence acceleration in the solution of systems of equations arising from RBF-FD meshless discretizations. *J. Comput. Phys.* **392**, 311–334 (2019)
51. Zamolo, R., Parussini, L.: Geometric uncertainty propagation in laminar flows solved by RBF-FD meshless technique. In: *Journal of Physics: Conference Series*, vol. 1599, IOP Publishing, (2020)
52. Zamolo, R., Parussini, L., Nobile, E.: Propagation of geometric uncertainties in heat transfer problems solved by RBF-FD meshless method. In: *Journal of Physics: Conference Series*, vol. 1868, IOP Publishing, (2021)

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.