Technical Report Series
Center for Data and Simulation Science

A. Klawonn, M. Lanser, J. Weber

# Learning Adaptive Coarse Basis Functions of FETI-DP

Technical Report ID: CDS-2022-02
Available at http://kups.ub.uni-koeln.de/id/eprint/62001
Submitted on June 13, 2022

# LEARNING ADAPTIVE COARSE BASIS FUNCTIONS OF FETI-DP

AXEL KLAWONN[†‡], MARTIN LANSER[†‡], AND JANINE WEBER[†]

June 13, 2022

**Abstract.** Domain decomposition methods are successful and highly parallel scalable iterative solution methods for discretized partial differential equations. Nevertheless, for many classes of problems, for example, elliptic partial differential equations with arbitrary coefficient distributions, adaptive coarse spaces are necessary to obtain robustness or, in other words, to guarantee a reliable and fast convergence. Adaptive coarse spaces are usually computed by solving many localized eigenvalue problems related to edges or faces of the domain decomposition. This results in a computationally expensive setup of the domain decomposition preconditioner or system operator. In this paper, we suggest to directly learn the adaptive constraints using a deep feedforward neural network and thus completely skip the computationally most expensive part of the setup, i.e., the solution of local eigenvalue problems. We consider a specific adaptive FETI-DP (Finite Tearing and Interconnecting - Dual Primal) approach and concentrate on stationary diffusion problems in two dimensions with arbitrary coefficient functions with large jumps. As an input for the neural network, we use an image representation of the coefficient function which resolves the structure of the coefficient distribution but is not necessarily identical to the discretization of the partial differential equation. Therefore, our approach is independent of the finite element mesh and can, in principle, be easily extended to other adaptive coarse spaces, problems, and domain decomposition methods. We show the robustness of our method for different problems and the generalization property of our trained neural networks by considering different coefficient distributions not contained in the training set. We also combine the learned constraints with computationally cheap frugal constraints to further improve our approach.

**Key words.** Machine Learning, Domain Decomposition, FETI-DP, BDDC, Adaptive Coarse Spaces, Scientific Machine Learning

**AMS subject classifications.** 65F10, 65N30, 65N55, 68T05

## 1. Introduction.

Recently, the relatively new research area of scientific machine learning has drawn increasing attention in various applications. Its growing importance was also emphasized in [2] by characterizing scientific machine learning as “a core component of artificial intelligence” and by attributing the “potential to transform science and energy research” to scientific machine learning. The main idea of this field is to combine existing methods from supervised or unsupervised machine learning and different research areas as numerical simulations or iterative solvers to develop new, hybrid methods which benefit from the expertise and experience in both areas. Also within the area of domain decomposition methods and multilevel methods, the most successful types of preconditioned iterative solution methods for solving discretized partial differential equations, recently, different approaches have been developed which use machine learning algorithms to accelerate the convergence and/or robustness of the iterative solvers and vice versa; see, e.g., [7, 14, 28] and the references therein. An overview of different approaches considering the combination of domain decomposition methods and machine learning can be found in the recent review article [14]. Let us note that the robustness of the convergence of a domain decomposition method with respect to arbitrary coefficient jumps depends on the correct choice of the coarse space. In our previous works [9, 10, 13], we have used

[†] Department of Mathematics and Computer Science, University of Cologne, Weyertal 86-90, 50931 Köln, Germany, {axel.klawonn, martin.lanser, janine.weber}@uni-koeln.de, url: http://www.numerik.uni-koeln.de

[‡] Center for Data and Simulation Science, University of Cologne, Germany, url: http://www.cds.uni-koeln.de

deep feedforward neural networks to define a robust and efficient coarse space for the FETI-DP (Finite Tearing and Interconnecting - Dual Primal) domain decomposition method which resulted in a robust convergence behavior for different realistic test problems. The coarse space consists of adaptive constraints, which are computed by solving localized eigenvalue problems on edges and/or faces of the interface of the domain decomposition. In general, adaptive coarse spaces guarantee a small condition number bound of the preconditioned system for a broad range of discretized partial differential equations with arbitrary jumps in certain coefficient functions. However, the solution of many localized eigenvalue problems tends to be expensive and time consuming. Moreover, for many edges and/or faces, the solution of the eigenvalue problem often turns out to be unnecessary afterwards since no coefficient jumps exist on those edges and/or faces. In [9, 10, 13], we therefore have trained a neural network to predict the geometric location of critical edges and/or faces, where we have to solve the eigenvalue problem and uncritical ones, where we can omit the eigenvalue problem completely. This has reduced the amount of necessary eigenvalue solves drastically and thus has increased the efficiency of the resulting domain decomposition approach.

In this work, we significantly extend our results from [9, 10, 13]

- by directly learning the adaptive edge constraints in 2D, which reduces the number of necessary eigenvalue problems to zero,
- by applying the new method to stationary diffusion problems with complicated coefficient distributions based on a microsection of a dual-phase steel,
- by showing numerically that the new approach results in a robust FETI-DP method for problems with arbitrary coefficient jumps,
- and by combining the learned constraints with the simple frugal approach (see [11]), which finally leads to our favored method.

So far, we restrict ourselves to regular domain decompositions in two spatial dimensions and the same adaptive FETI-DP coarse space as in [10]. We train $k$ separate regression neural networks for the prediction of the first $k$ constraints on an edge and make sure that our approach is independent of the resolution of the finite element mesh by exclusively relying on an image representation of the coefficient distribution instead of including explicit information on the finite element mesh.

Let us remark that in [4], a different approach to learn adaptive constraints was suggested, which is restricted to the case of stochastic elliptic partial differential equations. In contrast to our method, it is not independent of the discretization and uses a truncation of the Karhunen-Loève expansion as an input for neural networks with a single hidden layer.

The remainder of this paper is organized as follows. In the next chapter, we briefly introduce our model problem in form of a stationary diffusion equation and provide a brief algorithmic introduction of the classic FETI-DP method. Subsequently, we present a very specific adaptive FETI-DP coarse space as introduced in [29, 36] for two spatial dimensions which relies on the solution of local eigenvalue problems based on local Schur complement matrices and local jump operators. In section 3, we provide a detailed description of our machine learning approach to learn the coarse basis function for a problem-dependent FETI-DP coarse space and present first results with respect to our training and validation data. Finally, in section 4, we use the trained regression networks to provide numerical results for different realistic test problems and discuss the robustness of our proposed approach.

## 2. Classic and adaptive FETI-DP coarse spaces for stationary diffusion problems.

In this chapter, we give a brief algorithmic description of the FETI-DP [5, 25,37] method and the considered model problem, i.e., a stationary diffusion problem. It is known that the classic condition number bounds for FETI-DP only guarantee robustness of the iterative solver under fairly restrictive assumptions on the underlying coefficient distribution of the considered problem. Hence, we introduce a very specific adaptive coarse space for the FETI-DP method, which relies on the solution of local eigenvalue problems based on a local jump operator and which guarantees a robust convergence behavior for completely arbitrary coefficient distributions; see, e.g., [18–21, 27, 29, 30, 34].

**2.1. Model problem.** As the model problem for the following numerical experiments in this paper, we consider the following scalar elliptic boundary value problem, commonly known as *stationary diffusion*,

$$\begin{aligned} -\nabla \cdot (\rho \nabla u) &= f \text{ in } \Omega \\ u &= 0 \text{ on } \partial\Omega_D \\ \rho \nabla u \cdot n &= g \text{ on } \partial\Omega_N. \end{aligned} \tag{2.1}$$

Here, $\rho : \Omega \subset \mathbb{R}^2 \to \mathbb{R}$ is a sufficiently smooth coefficient function, $f : \Omega \to \mathbb{R}$ and $g : \partial\Omega_N \to \mathbb{R}$ are appropriate right-hand sides, and $n$ denotes the outer unit normal on $\partial\Omega_N$. For the remainder of this paper, we exclusively consider a homogeneous flow $g = 0$. We define the Sobolev space $V := H_0^1(\Omega, \partial\Omega_D) := \{v \in H^1(\Omega) : v_{|\partial\Omega_D} = 0\}$. Thus, for a piecewise constant parameter distribution $\rho \in L^\infty(\Omega)$ with $\rho \geq \rho_{\min} > 0$ and $f \in L^2(\Omega)$, we obtain the weak formulation: Find $u \in V$ such that

$$a(u, v) = F(v) \quad \forall v \in V, \tag{2.2}$$

where

$$a(u, v) := \int_\Omega \rho \nabla u \cdot \nabla v \; dx \quad \text{and} \quad F(v) := \int_\Omega f v \; dx. \tag{2.3}$$

To compute a numerical solution of the given stationary diffusion problem, we discretize (2.2) with finite elements and denote the respective finite element space by $V^h$. We thus obtain the linear system of equations

$$K_g u_g = f_g. \tag{2.4}$$

with $u_g$, $f_g \in V^h$.

**2.2. Classic FETI-DP.** Let us now give a short algorithmic description of the FETI-DP method as first introduced in [5]. The following presentation follows the descriptions in [10, 21, 38].

The FETI-DP method is a nonoverlapping domain decomposition method and relies on a divide-and-conquer strategy which is based on a geometric decomposition of the domain $\Omega \subset \mathbb{R}^2$ into a finite number of nonoverlapping subdomains $\Omega_i, i = 1, \ldots, N,\ N \in \mathbb{N}$ such that $\overline{\Omega} = \bigcup_{i=1}^N \overline{\Omega}_i$. We further assume that each of the subdomains $\Omega_i$ is the union of finite elements with matching finite element nodes on the interface

$$\Gamma := \left( \bigcup_{i=1}^N \partial\Omega_i \right) \setminus \partial\Omega_D.$$

In our case, each subdomain is the union of shape regular elements of diameter $\mathcal{O}(h)$. The diameter of a subdomain $\Omega_i$ is denoted by $H_i$ or, generically, by $H = \max_i(H_i)$.

Additionally, we denote by $W_i$ the local finite element space associated with a subdomain $\Omega_i$. Since, in this paper, we exclusively consider two-dimensional domains $\Omega \subset \mathbb{R}^2$, the finite element nodes on the interface $\Gamma$ are either vertex nodes, belonging to the boundary of more than two subdomains, or edge nodes, belonging to the boundary of exactly two subdomains; see, e.g., [25, Def. 3.1]. For the remainder of this paper, we denote by $\mathcal{E}_{ij}$ an edge between two subdomains $\Omega_i$ and $\Omega_j$.

**2.2.1. The FETI-DP preconditioner.** In the first step of the FETI-DP algorithm, we compute local stiffness matrices $K^{(i)}$ and local right-hand sides $f^{(i)}$ for each subdomain $\Omega_i, i = 1, \ldots, N$. These local problems are completely decoupled and the matrices $K^{(i)}$ are, in general, not invertible for subdomains without contact to the Dirichlet boundary $\partial\Omega_D$. We refer to these subdomains also as floating subdomains. For floating subdomains, the local solution is, in general, not unique and thus, usually, the local stiffness matrices of floating subdomains have a non-trivial null space. In particular, for stationary diffusion problems, the non-trivial null space consists of the constant functions. The FETI-DP method controls this non-trivial null space by sub-assembling the decoupled system $K = \operatorname{diag}\left(K^{(1)}, ..., K^{(N)}\right)$ in selected primal variables $\Pi$ to obtain a continuous global solution.

For a detailed description of this sub-assembly process, we first define local restriction operators $R_i : V^h \to W_i,\ i = 1, ..., N$, the block vectors $u^T := \left(u^{(1)T}, ..., u^{(N)T}\right)$ and $f^T := \left(f^{(1)T}, ..., f^{(N)T}\right)$, and the block matrix $R^T := \left(R_1^T, ..., R_N^T\right)$. The fully assembled system matrix can then be written as

$$K_g = R^T K R \tag{2.5}$$

and the fully assembled right-hand side as

$$f_g = R^T f. \tag{2.6}$$

Let us note again that the block matrix $K$ is not invertible as long as a single subdomain has no contact to the Dirichlet boundary and thus, a solution $u$ of the system

$$Ku = f$$

might be discontinuous on the interface.

To describe how the continuity of $u \in W := W_1 \times ... \times W_N$ on the interface is enforced using FETI-DP, we introduce a partitioning of the local stiffness matrices $K^{(i)}$, the local load vectors $f^{(i)}$, and the local solutions $u^{(i)}$ using a subdivision of the degrees of freedom into interior ($I$), primal ($\Pi$), and dual ($\Delta$) variables:

$$K^{(i)} = \begin{bmatrix} K_{II}^{(i)} & K_{\Delta I}^{(i)T} & K_{\Pi I}^{(i)T} \\ K_{\Delta I}^{(i)} & K_{\Delta\Delta}^{(i)} & K_{\Pi\Delta}^{(i)T} \\ K_{\Pi I}^{(i)} & K_{\Pi\Delta}^{(i)} & K_{\Pi\Pi}^{(i)} \end{bmatrix},\ u^{(i)} = \begin{bmatrix} u_I^{(i)} \\ u_\Delta^{(i)} \\ u_\Pi^{(i)} \end{bmatrix},\ \text{and}\ f^{(i)} = \begin{bmatrix} f_I^{(i)} \\ f_\Delta^{(i)} \\ f_\Pi^{(i)} \end{bmatrix}.$$

Throughout this paper, we always choose - at least - the vertices shared by different subdomains as the primal variables $\Pi$. Thus, in this case, $\Delta$ consists of all remaining variables on the interface, i.e., all edge nodes in case of a domain decomposition in two spatial dimensions. Additionally, we will also implement a coarse space that integrates additional edge constraints as primal variables to further increase the robustness of the FETI-DP preconditioner; see also subsection 2.3. To obtain a more compact notation of the FETI-DP algorithm, we additionally introduce the index $B$ as the union of interior and dual degrees of freedom leading to the definition of the following matrices and vectors

$$K_{BB}^{(i)} = \begin{bmatrix} K_{II}^{(i)} & K_{\Delta I}^{(i)T} \\ K_{\Delta I}^{(i)} & K_{\Delta\Delta}^{(i)} \end{bmatrix},\ K_{\Pi B}^{(i)} = \begin{bmatrix} K_{\Pi I}^{(i)} & K_{\Pi\Delta}^{(i)} \end{bmatrix},\ \text{and}\ f_B^{(i)} = \begin{bmatrix} f_I^{(i)T} & f_\Delta^{(i)T} \end{bmatrix}^T,$$

and the corresponding block diagonal matrices

$$\begin{aligned} K_{BB} &= \operatorname{diag}_{i=1}^N K_{BB}^{(i)}, \\ K_{II} &= \operatorname{diag}_{i=1}^N K_{II}^{(i)}, \\ K_{\Delta\Delta} &= \operatorname{diag}_{i=1}^N K_{\Delta\Delta}^{(i)}, \\ \text{and } K_{\Pi\Pi} &= \operatorname{diag}_{i=1}^N K_{\Pi\Pi}^{(i)}. \end{aligned} \tag{2.7}$$

Analogously, we obtain the corresponding block vector $u_B = [u_B^{(1)T}, \ldots, u_B^{(N)T}]^T$ and the corresponding block right-hand side $f_B = \left[f_B^{(1)T}, \ldots, f_B^{(N)T}\right]^T$, respectively, which can be partitioned accordingly.

For the FETI-DP algorithm, continuity in the primal variables $\Pi$ is enforced by a finite element assembly process, while continuity in the dual variables $\Delta$ is enforced iteratively by Lagrangian multipliers $\lambda$. For an algorithmic description of the primal assembly process, we introduce the primal assembly operators $R_\Pi^{(i)^T}, i = 1, \ldots, N$, which consist of values in $\{0, 1\}$. We can then obtain the primally assembled matrices by

$$\widetilde{K}_{\Pi\Pi} = \sum_{i=1}^N R_\Pi^{(i)T} K_{\Pi\Pi}^{(i)} R_\Pi^{(i)} \text{ and } \widetilde{K}_{\Pi B} = \left[R_\Pi^{(1)T} K_{\Pi B}^{(1)}, \ldots, R_\Pi^{(N)T} K_{\Pi B}^{(N)}\right], \tag{2.8}$$

as well as the corresponding right-hand side as

$$\widetilde{f} = \left[f_B^T, \left(\textstyle\sum_{i=1}^N R_\Pi^{(i)T} f_\Pi^{(i)}\right)^T\right]^T.$$

To further enforce continuity in the dual degrees of freedom, in a second step, we introduce Lagrange multipliers $\lambda$ which act between two degrees of freedom each. Additionally, we define a corresponding jump operator $B_B = [B_B^{(1)}, \ldots, B_B^{(N)}]$. Each row of $B_B$ enforces equality of two variables associated with the same physical point but two different subdomains and thus each row corresponds to one Lagrange multiplier. In particular, the entries of the local matrices $B_B^{(i)}, i = 1, \ldots, N$ have zero entries for the interior degrees of freedom and, for the dual degrees of freedom, exactly one $+1$ and one $-1$ for each row such that $B_B u_B = 0$ holds if and only if $u_B$ is continuous on the interface. By integrating the continuity constraint $B_B u_B = 0$ into our FETI-DP system, we obtain the FETI-DP master system given by

$$\begin{pmatrix} K_{BB} & \widetilde{K}_{\Pi B}^T & B_B^T \\ \widetilde{K}_{\Pi B} & \widetilde{K}_{\Pi\Pi} & O \\ B_B & O & O \end{pmatrix} \begin{pmatrix} u_B \\ \tilde{u}_\Pi \\ \lambda \end{pmatrix} = \begin{pmatrix} f_B \\ \tilde{f}_\Pi \\ 0 \end{pmatrix}. \tag{2.9}$$

In a preliminary step, we eliminate the variables $u_B$ and $\tilde{u}_\Pi$ in (2.9) and obtain a linear system in the Lagrange multipliers $\lambda$. As a result, we obtain the (unpreconditioned) classic FETI-DP system

$$F\lambda = d, \tag{2.10}$$

where

$$F = B_B K_{BB}^{-1} B_B^T + B_B K_{BB}^{-1} \widetilde{K}_{\Pi B}^T \widetilde{S}_{\Pi\Pi}^{-1} \widetilde{K}_{\Pi B} K_{BB}^{-1} B_B^T \text{ and}$$
$$d = B_B K_{BB}^{-1} f_B + B_B K_{BB}^{-1} \widetilde{K}_{\Pi B}^T \widetilde{S}_{\Pi\Pi}^{-1} \left( \left( \sum_{i=1}^{N} R_{\Pi}^{(i)T} f_{\Pi}^{(i)} \right) - \widetilde{K}_{\Pi B} K_{BB}^{-1} f_B \right). \tag{2.11}$$

In (2.11), the Schur complement $\widetilde{S}_{\Pi\Pi}$ for the primal variables is defined as

$$\widetilde{S}_{\Pi\Pi} = \widetilde{K}_{\Pi\Pi} - \widetilde{K}_{\Pi B} K_{BB}^{-1} \widetilde{K}_{\Pi B}^T. \tag{2.12}$$

Let us note that the application of $F$ in (2.11) can be split into two additive parts. Due to its block structure, the first part requires only local operations and can be executed completely in parallel. The second part, however, requires the solution of a coupled *coarse problem* in form of the application of $\widetilde{S}_{\Pi\Pi}^{-1}$. The exact solution of the coarse problem, in general, requires a serial solver and can become a scaling bottleneck in a parallel implementation. Therefore, we are usually interested in a coarse space which is preferably small but, at the same time, robust for heterogeneous coefficient or material distributions.

The system of equations (2.10) in the Lagrange multipliers is then solved by a Krylov subspace method, such as the PCG [35] or the GMRES method [35]. Hence, the FETI-DP method is the iterative solution of the preconditioned system

$$M^{-1} F \lambda = M^{-1} d. \tag{2.13}$$

For our numerical experiments in section 4, we always use the PCG method and the standard Dirichlet preconditioner $M_D^{-1} =: M^{-1}$ given by

$$M_D^{-1} = B_{B,D} \left[0 \quad I_\Delta\right]^T \left(K_{\Delta\Delta} - K_{\Delta I} K_{II}^{-1} K_{\Delta I}^T\right) \left[0 \quad I_\Delta\right] B_{B,D}^T = B_D \widetilde{S} B_D^T;$$

see, e.g., [5], where $I_\Delta$ is the identity matrix on the dual degrees of freedom and the matrices $B_{B,D}$ and $B_D$ are scaled variants of the jump operators $B_B$ and $B$, respectively. As the scaling procedure for the scaled matrices $B_{B,D}$ and $B_D$, in this paper, we exclusively consider the $\rho$-scaling approach; see, e.g., [22, 27, 34, 37] for a mathematical description of this approach in two and three dimensions.

**2.2.2. Condition number bound.** As briefly mentioned in the introduction, the convergence rate and the condition number estimate for the FETI-DP method depend strongly on the chosen primal constraints $\Pi$, i.e., the coarse space and the induced scaling matrices for the preconditioner of the FETI-DP system. Without recapitulating all the technical details, let us recall from the literature that the estimate of the spectral condition number $\kappa(M_D^{-1}F)$ of the preconditioned FETI-DP system is strongly connected to an estimate of the operator

$$P_D := B_D^T B; \tag{2.14}$$

see also [24, 26] for a first use of the $P_D$ operator in this context.

In two dimensions, we can derive a polylogarithmic upper bound for $|P_D w|_{\widetilde{S}}$ with $w \in \widetilde{W}$ and $\widetilde{W} \subset W$ the primally assembled subspace of $W$. This results in the following polylogarithmic condition number bound for the preconditioned FETI-DP method with a standard vertex coarse space

$$\kappa(M_D^{-1}F) \leq \widetilde{C}\Big(1 + \log\left(\frac{H}{h}\right)\Big)^2 \tag{2.15}$$

with the constant $\widetilde{C}$ independent of $H$, $h$, and jumps in the PDE coefficients; see [23, 25, 26]. However, the independence of this condition number bound on the coefficient contrast does only hold under certain assumptions on the coefficient function or the material distribution, e.g., for constant or slowly varying coefficients within each subdomain; see, e.g., [31, 37]. For a detailed and technical proof of the cited condition number estimates based on an upper and a lower bound of the Rayleigh quotient of the preconditioned FETI-DP system and Poincaré inequalities, we refer to, e.g., [23, 31] for two-dimensional model problems and [25, 26] for three dimensions.

**2.3. Adaptive FETI-DP based on a local jump operator.** For completely arbitrary coefficient distributions, the constant $\widetilde{C}$ in (2.15) usually depends on the contrast of the coefficient function and we thus experience a deteriorating convergence behavior of the classic methods. A remedy is obtained by *adaptive* coarse spaces, which have been proposed by several authors for both overlapping and nonoverlapping domain decomposition methods; see, for example, [14, Section 2.2] for a recent list of references. These methods enhance the coarse space with selected eigenvectors or coarse modes and are thus problem-dependent and robust for arbitrary heterogeneities. In general, most of these adaptive methods rely on the solution of local eigenvalue problems on edges, local interfaces, or subdomains of the domain decomposition. Thus, most adaptive coarse spaces are built in a local fashion and exploit the parallel structure of the underlying domain decomposition algorithm. In this paper, we focus on a very specific adaptive coarse space for the FETI-DP method, which has also successfully been applied to the BDDC method [18, 20, 21, 27, 29, 30, 34].

**2.3.1. Adaptive constraints based on local generalized edge eigenvalue problems.** To motivate the approach first introduced in [29, 30], let us briefly recall that we can establish an algebraic relation between the Rayleigh quotient of the preconditioned FETI-DP system and the operator $P_D$, i.e., $|P_D w|_{\widetilde{S}}$ as introduced in subsection 2.2.2. Using this relation, the following adaptive coarse space relies on the solution of local eigenvalue problems which are directly connected to the $P_D$ operator and local Schur complement matrices. To provide a mathematical description of the cited adaptive coarse space, let us now introduce some basic notation. Note that we always assume the existence of an a priori nonadaptive coarse space that ensures the invertibility of the local problems of each subdomain. In this article, we therefore assume that all vertices are chosen as primal variables. The following description is roughly based on [11] and [27, 34, 38].

In two dimensions, for each edge $\mathcal{E}_{ij}$ between two neighboring subdomains $\Omega_i$ and $\Omega_j$, a single eigenvalue problem has to be solved. To give a mathematical formulation of the respective local eigenvalue problem, we first introduce the local restriction of the jump matrix $B$ to the edge $\mathcal{E}_{ij}$. We thus define

$$B_{\mathcal{E}_{ij}} := \left(B^{(i)}_{\mathcal{E}_{ij}},\ B^{(j)}_{\mathcal{E}_{ij}}\right)$$

as the submatrix of $\left(B^{(i)}, B^{(j)}\right)$ with consisting of exactly one 1 and one $-1$ for each row and being zero elsewhere. Analogously, we denote by

$$B_{D,\mathcal{E}_{ij}} := \left(B^{(i)}_{D,\mathcal{E}_{ij}},\ B^{(j)}_{D,\mathcal{E}_{ij}}\right)$$

the corresponding scaled submatrix of $\left(B_D^{(i)}, B_D^{(j)}\right)$, i.e., the scaled variant of $B_{\mathcal{E}_{ij}}$. We then define the block-diagonal Schur complement matrix

$$S_{ij} := \begin{pmatrix} S^{(i)} & 0 \\ 0 & S^{(j)} \end{pmatrix} \in \mathbb{R}^{(n_i+n_j)\times(n_i+n_j)} \tag{2.16}$$

with $S^{(i)}$ and $S^{(j)}$ being the local Schur complements of the local stiffness matrices $K^{(i)}$ and $K^{(j)}$, respectively, with respect to the interface variables and $n_l, l \in \{i, j\}$, the number of degrees of freedom on the local part of the interface. We further define a local version of the jump operator $P_D = B_D^T B$ as

$$P_{D_{ij}} := B_{D,\mathcal{E}_{ij}}^T B_{\mathcal{E}_{ij}}.$$

Then, according to [21,29,30,34], one has to solve the following generalized eigenvalue problem for each edge $\mathcal{E}_{ij}$: Find $w_{ij} \in (\mathrm{ker\,S_{ij}})^{\perp}$ such that

$$\langle P_{D_{ij}} v_{ij},\ S_{ij} P_{D_{ij}} w_{ij}\rangle = \mu_{ij} \langle v_{ij},\ S_{ij} w_{ij}\rangle \quad \forall v_{ij} \in (\mathrm{ker\,S_{ij}})^{\perp}. \tag{2.17}$$

To finally compute the adaptive constraints to enhance the a priori coarse space, we then select all eigenvectors $w_{ij}^l, l = 1, \ldots, L$ belonging to eigenvalues $\mu_{ij}^l,\ l = 1, \ldots, L$, which are larger than or equal to a user-defined tolerance $TOL$. Next, we enforce the constraints

$$w_{ij}^{lT} P_{D_{ij}}^T S_{ij} P_{D_{ij}} w_{ij} = c_{ij}^{lT} B_{\mathcal{E}_{ij}} w_{ij} = 0, \tag{2.18}$$

for given constraint vectors

$$c_{ij}^l := B_{D,\mathcal{E}_{ij}} S_{ij} P_{D_{ij}} w_{ij}^l,\ l = 1, \ldots, L, \tag{2.19}$$

for example, with a projector preconditioning/deflation or a transformation-of-basis approach.

**2.3.2. Condition number bound.** As for the classic FETI-DP and BDDC methods, the spectral condition number for the adaptive FETI-DP method presented in subsection 2.3.1 is also closely related to an estimate of the $P_D$ operator. For the sake of completeness, let us briefly cite the respective condition number bound for two-dimensional problems which is valid for arbitrary coefficient distributions.

For a two-dimensional domain $\Omega$, enhancing the FETI-DP and BDDC coarse space with the adaptive coarse constraints from (2.18), a condition number bound of the form

$$\kappa(\widetilde{M}^{-1} F) \leq N_{\mathcal{E}}^2 \cdot TOL \tag{2.20}$$

has been derived in [21]. Here, $\widetilde{M}^{-1}$ is either the projector or the balancing preconditioner and with $N_{\mathcal{E}}$ denoting the maximum number of edges of a subdomain. In particular, the constant in (2.20) does only depend on geometric constants of the domain decomposition and not on the contrast of the underlying coefficient distribution. For a detailed proof of the condition number bound (2.20) in two dimensions, we refer to [34, Theorem 3.3.1] and [21, Theorem 5.1].

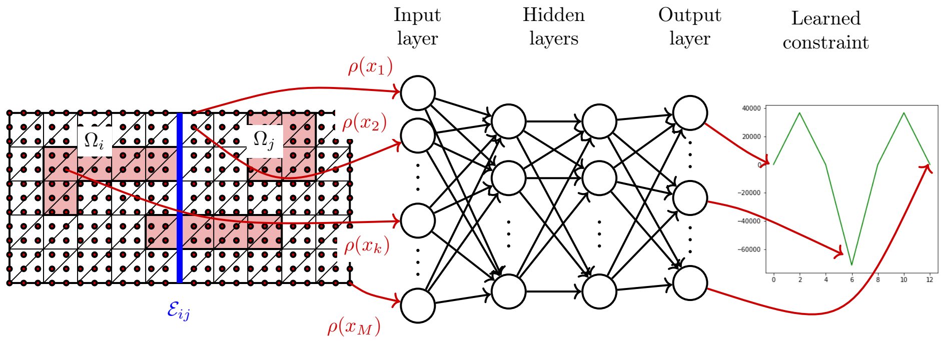

FIGURE 1. *Visualization of our network models $N_l$ and $\widetilde{N}_l$, $l \leq k$. As input data for the neural network, we use samples of the coeffcient function for the two neighboring subdomains of an edge* **(left)**. *Here, dark red corresponds to a high coefficient and white corresponds to a low coefficient. In this representation, the samples are used as input data for a feedforward neural network with two hidden layers* **(middle)**. *The output of the network is a discretized egde constraint* **(right)**.

**3. Learning adaptive constraints using supervised regression neural network models.** The aim of this work is to compute a discrete approximation of adaptive edge constraints resulting from the local eigenvalue problem (2.17) by training a supervised regression machine learning model. The set-up and the solution of the eigenvalue problem (2.17) is, especially for three-dimensional domains, usually computationally relative expensive whereas the evaluation of a trained and saved network model is typically cheap. In subsection 3.1, we introduce our machine learning approach in more detail by providing a concrete description of our mesh-independent regression neural networks. In subsection 3.2, we describe the training and validation data and present first results of the trained models with respect to the training data.

**3.1. Defining a mesh-independent neural network model.** We extend our approach from [10] and [13] by directly learning a discretized approximation of the adaptive FETI-DP constraints for the coarse space described in subsection 2.2. In [10, 13], we have successfully developed a supervised machine learning approach which uses a carefully designed representation of the coefficient or material distribution, respectively, of the underlying model problem as input data for a neural network in order to predict the geometric location of additional coarse basis functions which are necessary to obtain a robust algorithm. As an effect, a high amount of the eigenvalue problems on edges or faces do not need to be setup and solved and thus the computational effort of the underlying adaptive FETI-DP method can be reduced. In partciular, we were able to save up to 94% of the eigenvalue problems for irregular decompositions in two spatial dimensions; see, e.g., [10] for more details.

Here, we go one step further and train a supervised regression model which is able to compute an approximation of the first $k$ adaptive constraints themselves. In contrast to our approach presented in [10, 13], this does not require the setup and solution of any local eigenvalue problems at all. Let us mention that, in principle, both concepts can be combined with each other as follows. One could first use the framework described in [10] to obtain an a priori prediction of the critical edges where adaptive constraints are necessary and afterwards apply the procedure as described below to replace them by those predicted by the trained neural network. In fact, since the frugal constraints are often a good approximation of the first adaptive constraint, the constraints predicted by the neural network are only needed on those critical

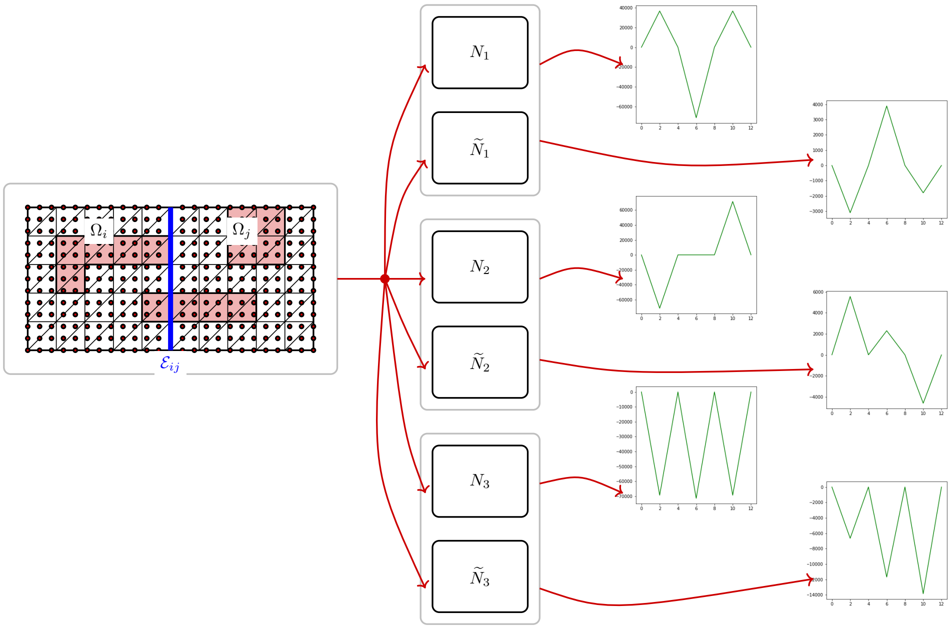

FIGURE 2. *Visualization of the different network models for the computation of approximated edge constraints. We train three network models $N_l$, $l \leq k = 3$ for pairs of floating subdomains and three network models $\widetilde{N}_l$, $l \leq k = 3$ for pairs of subdomains that share an edge $\mathcal{E}_{ij}$ with direct contact to the Dirichlet boundary $\Omega_D$. For each network model, we use the same input data, i.e., sampling points as shown on the left, but a different ground truth, i.e., different approximated edge constraints as shown on the right. See also Figure 1 for more details on the input and output data.*

edges, where at least two constraints have to be used.

The core idea of our concept is to train $k \in \{1, \dots, n_{\mathcal{E}_{ij}}\}$ separate models $N_l$, $l \leq k$ for the prediction of the first $k$ edge constraints resulting from the generalized edge eigenvalue problem (2.17), where $n_{\mathcal{E}_{ij}}$ is the number of the degrees of freedom of the edge $\mathcal{E}_{ij}$ between two subdomains $\Omega_i$ and $\Omega_j$. For each of the models $N_l$, $l \leq k$, we train a dense feedforward regression neural network [3,6] with the hyperparameters as given in Table 1. An exemplary dense feedforward neural network model $N_l$ is shown in Figure 1. The reported choice of the hyperparameters in Table 1 has been optimized by a grid search and using cross validation on the training and validation data; see also the related discussion in subsection 3.2. As listed in Table 1, we use the MSE (mean squared error) as the loss function to evaluate the training and validation accuracy of the trained networks and we train the networks by using a stochastic gradient descent (SGD) method with a batch size of 32 and using the Adam optimizer [17] with its default parameters $\{\beta_1 = 0.9,\ \beta_2 = 0.999\}$. We have trained the network for 600 epochs while using an early stopping criterion ( [33] or [6, Sect. 7.8]) with a patience of 10 epochs. Let us note that, for the numerical experiments in section 4, we exclusively choose $k \leq 3$, computing an approximation for at most three constraints for each edge.

As input for the neural network models $N_l$, $l \leq k$, we use a mesh-independent image representation of the coefficient function $\rho$ of the two subdomains $\Omega_i$ and $\Omega_j$ sharing the edge $\mathcal{E}_{ij}$. Here, we use the same sampling procedure as described in detail in [10] to compute function evaluations of the coefficient function with a consistent ordering and a fixed length as input data for the neural networks. Basically, we compute a geometric grid of points which covers the coefficient function in the two neighboring subdomains of an edge. This grid is independent of the finite element mesh and the only assumption we make is that the grid is fine enough to capture all jumps and geometric details in the coefficient function. We then evaluate the coefficient function $\rho$ in all computed grid points and use dummy values of $-1$ for *sampling points* which lay outside the two subdomains. A consistent ordering of the regular grid of sampling points is then given by the increasing Euclidean distance of the sampling points to the edge $\mathcal{E}_{ij}$. Let us note that our approach, as first proposed in [10], is independent of the finite element discretization and can be used for both, regular and irregular domain decompositions as obtained by METIS [16]. Furthermore, the described *sampling approach* can be extended to three-dimensional domains; see [13, 14]. However, we restrict ourselves to regular, two-dimensional domains for the remainder of this paper.

As output for the different network models we use discrete values of the adaptive edge constraints obtained by the solution of the local edge eigenvalue problem (2.17). For the training of the $l$-th network $N_l$, we use the respective constraint as output data which results from the eigenvector $w_{ij}^l$ belonging to the eigenvalue $\mu_{ij}^l$. Thus, the different network models $N_l,\ l \leq k$, in principle, only differ by their respective output data but are trained with the same input data to approximate nonlinear functions $\mathcal{F}_l : I_l \rightarrow O_l,\ l \leq k$, where $I_l$ is the input space and $O_l$ is the output space, respectively, of the network model $N_l$. For the implementation and training of the different network models we use TensorFlow [1] as well as Scikit-learn [32] for the preprocessing of the data. Based on the above descriptions, we use the following input and output spaces $I_l$ and $O_l, l \leq k$, respectively, for the different network models:

- $I_l \in \mathbb{R}^{3200}$, i.e., we use 3 200 sampling points as input data for the neural networks. Generically, this corresponds to a mesh discretization defined by $H/h = 40$.
- $O_l \in \mathbb{R}^{19}$, i.e., we evaluate the different edge constraints in 19 degrees of freedom to obtain the output data for the neural network models.

Let us note that, in principle, one has to decide for a fixed number of output nodes of the neural network models $N_l,\ l \leq k$, and thus for a fixed number of degrees of freedom for the discretized representation of the adaptive edge constraints. However, in order to provide an approach which is applicable for different finite element discretizations, we set a fixed number of degrees of freedom per edge as a basis which determines the number of output nodes for our neural networks and use an interpolation technique to generalize our approach to more general mesh discretizations. For the numerical experiments presented in section 4, we choose a mesh discretization defined by $H/h = 20$ as a basis and thus have 19 output nodes for each of our regression networks $N_l,\ l \leq k$. In order to obtain an approximation of the adaptive edge constraints for different, that is, finer mesh discretizations, we compute a linear interpolation of the discretized output of the respective neural network $N_l$ using the finite element mesh points as interpolation points and the finite element basis functions as the basis. Let us note that, in principle, also a polynomial interpolation with polynomial basis functions of a higher order could be used.

Besides using different networks $N_l,\ l \leq k,\ k \in \{1, \ldots, n_{\mathcal{E}_{ij}}\}$, for the approximation of the first $k$ adaptive constraints, we additionally train separate neural networks for edges which are directly connected to the Dirichlet boundary $\partial\Omega_D$ of our domain $\Omega$. In general, the resulting adaptive edge constraints (2.18) are different for edges with direct contact to the Dirichlet boundary $\partial\Omega_D$ than for edges between two float-

| **Hyperparameter** | **Optimal choice** |
|---|---|
| # Hidden layers | 4 |
| # Neurons per hidden layer | 50 |
| Dropout per hidden layer | 20% |
| Activation function | ReLU |
| Optimizer | Adam |
| Initial learning rate | 0.001 |
| Loss function | MSE |

TABLE 1
*Hyperparameters for the regression neural networks obtained by a grid search.*

ing subdomains due to the direct influence of the Dirichlet boundary condition on the local Schur complement matrices $S_{ij}$ of the local edge eigenvalue problem (2.17). This observation is also illustrated by the exemplary constraints for the coefficient function $\rho$ visualized in Figure 4 (top row, left) with three straight channels of a high coefficient horizontally crossing each subdomain. In Figure 3 (top row), we show the adaptive constraints resulting from the edge eigenvalue problem (2.17) and the tolerance TOL $= 100$ for an edge $\mathcal{E}_{ij}$ shared by two floating subdomains. In particular, the selected tolerance TOL $= 100$ results in three adaptive constraints for the given coefficient distribution. For a direct comparison, in Figure 3 (bottom row), we show the respective constraints for the edge $\mathcal{E}_{12}$ between the subdomains $\Omega_1$ and $\Omega_2$ which has a direct contact to the Dirichlet boundary $\partial\Omega_D := \partial\Omega$. As we can observe from directly comparing the respective edge constraints in Figure 3, we obtain different edge constraints in both cases. Therefore, we have decided to train and evaluate separate neural networks $\widetilde{N}_l,\ l \leq k,\ k \in \{1, \ldots, n_{\mathcal{E}_{ij}}\}$, for edges which are directly connected to $\partial\Omega_D$ in order to improve the accuracy and thus the robustness of our proposed approach; see also Figure 2 for an overview of the different trained networks. Finally, to further increase the training and generalization properties of all networks, we scale both the input and output data of the networks by using a min-max-scaling as implemented in Scikit-learn [32]. As a result, both the input and the output data range between values of 0 and 1 after the scaling procedure.

Let us note that the training of all networks, that is, the networks for the different $k$ adaptive constraints consisting of those for floating pairs of subdomains and for subdomains sharing an edge connected to the Dirichlet boundary, can be executed completely in parallel since the different networks are completely independent of each other. This makes our approach well suited for the application on parallel computers in general and especially on GPU clusters.

**3.2. Results on training and validation data.** Let us now describe the data used for the training and validation of the networks in more detail and summarize the results of our trained networks with respect to the training and validation data. For the generation of the input data of the neural networks, we use the same sampling approach as described in detail in [10, 14]. To generate the training and validation data, we compute *sampling points*, that is, function evaluations of the diffusion coefficient function $\rho$ for two neighboring subdomains $\Omega_i$ and $\Omega_j$ sharing an edge $\mathcal{E}_{ij}$. In particular, this means that the training of the neural networks is completely based on local information of the domain decomposition. As explained in subsection 3.1, we train different neural networks $N_l$ and $\widetilde{N}_l$, respectively, for edges between two floating subdomains and pairs of subdomains which share an edge with direct contact

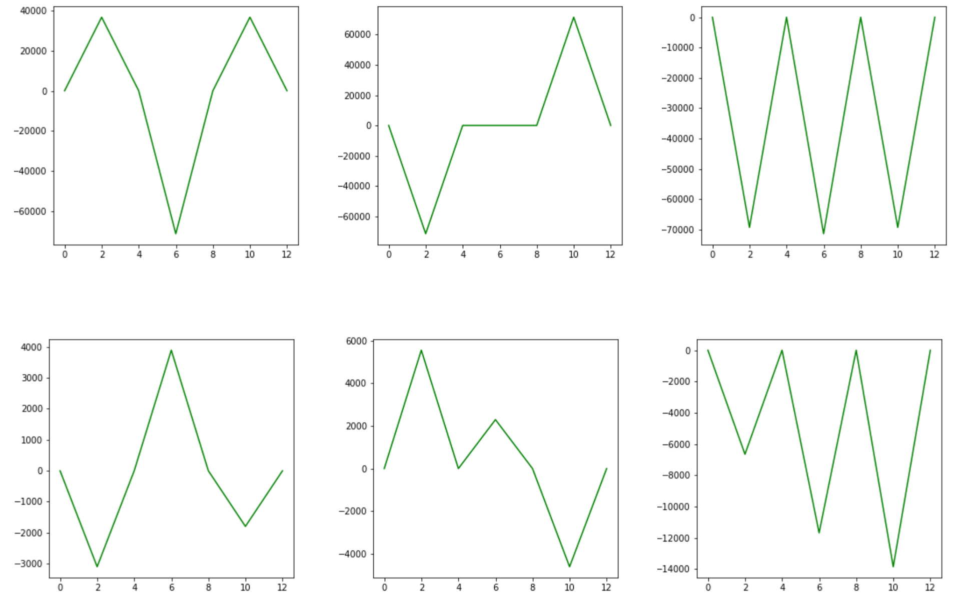

FIGURE 3. *Exemplary adaptive constraints for a stationary diffusion problem and the coefficient distribution as in Figure 4 (top row, right) and* TOL $= 100$*.* ***Top row:*** *Edge constraints for two floating subdomains.* ***Bottom row:*** *Edge constraints for two subdomains where the adjacent edge has direct contact to the Dirichlet boundary $\partial\Omega_D$. In both cases, the pair of subdomains is extracted from a decomposition of the unit square into $4 \times 4$ subdomains and $H/h = 14$. Coefficient contrast* $1e6$*.*

to the Dirichlet boundary $\partial\Omega_D$. To generate an appropriate amount of training and validation data, we use a set of carefully constructed coefficient distributions which are exemplarily visualized in Figure 4. The coefficient distributions shown in Figure 4 are mirrored and varied in their size and position to obtain a total of 4 500 coefficient distributions used for the training and validation of the neural networks. In analogy to [10, 12, 14], we denote the resulting training data set by *smart data*. Let us note that we have shown in [12] and [13] that it is also possible to train the classification neural networks used there with randomized training data both in two and three dimensions. However, we restrict ourselves to the use of the smart training data for the remainder of this paper since this training data set provided the best results in [12]. For all training data configurations, we always set the high coefficient to $\rho_1 = 1e6$ in the dark blue pixels and $\rho_2 = 1$ otherwise.

For each configuration in the training and validation data, we solve the local eigenvalue problem (2.17) and use the tolerance value TOL $= 100$ to determine the eigenvectors for the computation of the respective adaptive constraints which are necessary to obtain a robust coarse space. Since our aim is to train $k$ regression neural network models with a fixed number of output nodes, we always save a discretized representation, that is, a fixed number of values of the $l$-th adaptive edge constraint as output data for the network $N_l$ or $\widetilde{N}_l$, respectively. To generate the input and output data for the training and validation of the different network models $N_l$ and $\widetilde{N}_l,\ l \leq k$, we have used a decomposition of the unit square into $4 \times 4$ subdomains and a mesh size defined by $H/h = 20$.

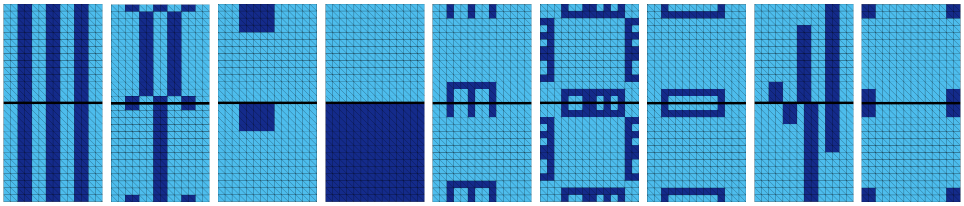

FIGURE 4. *Nine different types of coefficient functions used for the training and validation of the neural network models. The inclusions, channels, boxes, and combs with high coefficient are displaced, modified in sized, and mirrored with respect to the edge in order to generate the complete training data set. We refer to the resulting data set as smart data. Taken from [10].*

The hyperparameters and details of the training process of the optimized regression networks are summarized in Table 1 and have already been described in detail in subsection 3.1. Let us mention that we have also included neural network models in our grid search that did not use any dropout within the hidden layers. In particular, we have considered the same network as defined by the hyperparameters in Table 1 but without using dropout. Even though this resulted in a regression model with a slightly lower MSE with respect to the training data, the model resulted in a higher MSE loss on the validation data and especially a worse performance with respect to our test data; cf. the related comparison in section 4 with respect to our test data. In Table 2, we provide the MSE loss values for both models, that is, the neural network defined in Table 1 with and without dropout for the training and validation data. Due to the better generalization properties, especially with respect to previously unseen test data, we decided to use the model with dropout as the final model for our experiments in section 4.

In Figure 5, we present exemplary comparison results for a specific training data configuration obtained by the networks $N_1$ and $N_2$ for an edge between two floating subdomains. In the top row (in green), we show the ground truth, that is, the adaptive constraints resulting from the first and second eigenmodes of the local edge eigenvalue problem (2.17) for a given coefficient distribution. In the bottom row (in blue), we show the respective discrete approximations obtained by the trained networks $N_1$ and $N_2$, respectively. We can observe that the approximated constraints show a very similar qualitative behavior as the true adaptive constraints, that is, the peaks and plateaus occur at approximately the same edge degrees of freedom. From our previous experiments with different adaptive and heuristic coarse spaces (e.g., [8,11,18,20–22]) we know that the location of peaks and plateaus of the respective constraints is usually more important to provide a robust preconditioner than the exact absolute value of the function at these degrees of freedom. Thus, the shown output by the regression networks serves as a good approximation of the adaptive constraints for the given coefficient function. This is also confirmed by our numerical results for the unseen test data.

## 4. Numerical results.

In this chapter, we provide numerical results for different stationary diffusion problems using our supervised machine learning approach as described in section 3. As different test problems for our trained regression models we consider various heterogeneous coefficient functions $\rho$ which are explicitly not included in our training and validation data set. As iterative solver to solve the resulting preconditioned FETI-DP systems, we always use the PCG (preconditioned conjugate gradient) method with a relative reduction of the residuum of $1e$-8 as the stopping

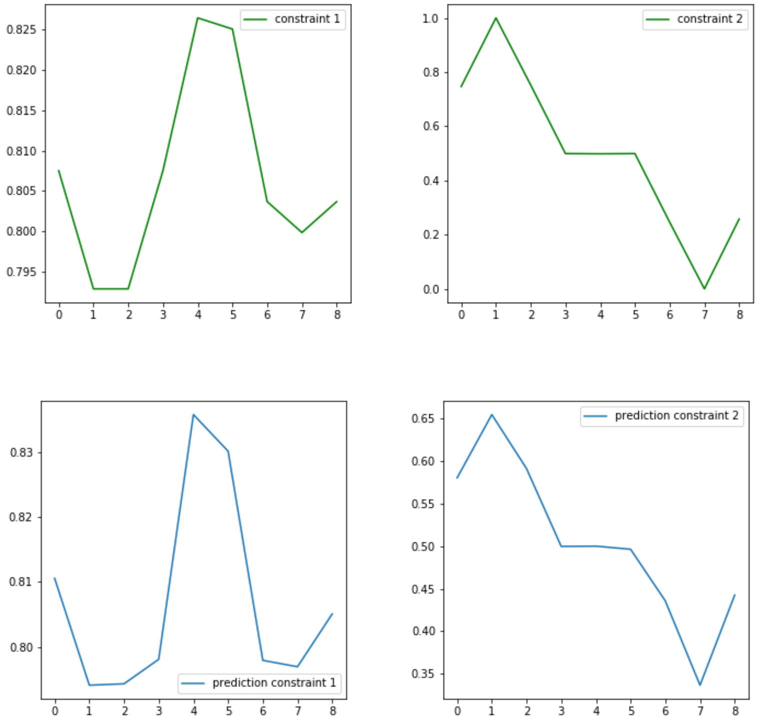

FIGURE 5. *Results on exemplary training and validation data. Here, we consider a coefficient distribution for two neighboring subdomains sharing an edge which is defined by two channels of a high coefficient. Coefficient contrast* 1e6. *Green: ground truth, blue: prediction as obtained by the neural networks.*

| **Evaluated data** | **Neural network model** | |
|---|---|---|
| | With dropout | Without dropout |
| Training data | 1.27e-03 | 6.24e-04 |
| Validation data | 2.01e-03 | 9.53e-03 |

TABLE 2

*MSE loss values for the training and validation data for the neural network model defined by the hyperparameters in Table 1 with and without using dropout within the hidden layers.*

criterion.

As a first test problem, we consider a heterogeneous stationary diffusion problem with a coefficient distribution as defined in Figure 6. For this test problem, we decompose our domain $\Omega = [0,1]^2$ into $4 \times 4$ subdomains and we use a mesh discretization defined by $H/h = 10$. We choose all vertices as primal variables and set $\rho_1 = 1e6$ in the dark blue pixels in Figure 6 and $\rho_2 = 1$ otherwise. Let us note again that this specific combination of U-shaped and straight channels with a high coefficient is not included in our training and validation data set. Thus, this problem serves as a first sanity check whether the trained regression models are able to generalize to different coefficient distributions. In Figure 7 (upper row), we show the adaptive constraints which result from the local edge eigenvalue problem (2.17) when using the tolerance TOL $= 100$ for the selection of the respective eigenvectors for an edge between two floating subdomains. As we can observe from Figure 7, the selected tolerance leads to two additional adaptive constraints for each horizontal edge. In Figure 8 (left), we have visualized the eigenvalues of the resulting preconditioned FETI-DP system $M^{-1}F$, that is, after the implementation of the computed adaptive edge constraints. As we can see in Figure 8, the highest eigenvalue of the global system $M^{-1}F$ is around 2.8 which also results in a condition number estimate of 2.8 and an iteration number of 10. Hence, the condition number of the preconditioned system is clearly independent of the coeffcient contrast as we are expecting from the implemented adaptive coarse space. For a direct comparison, in Figure 7 (bottom row), we show the approximated constraints as obtained by our neural networks $N_1$ and $N_2$ for the first and second edge constraints. Obviously, both, the qualitative as well as quantitative behavior of the approximated constraints is very similar to the exact adaptive edge constraints in Figure 7 (top row). Additionally, we also show the eigenvalues of the preconditioned FETI-DP system $M^{-1}F$ when implementing the approximated adaptive constraints in Figure 8 (right). Here, we obtain the largest eigenvalue of approximately 35.5 which results in a condition number estimate of about the same value and an iteration number of 14. Thus, we also obtain a condition number which is independent of the coefficient contrast when using the predictions of the regression neural network models and an iteration number which is very close to the iteration count of the adaptive FETI-DP coarse space indicating that our machine learning model works fairly well for this synthetic coefficient distribution.

As a second and more realistic test problem, we consider a subsection of a microsection of a dual-phase steel. The complete microsection is shown in Figure 9 (left) and the subsection which is used in our numerical experiments is shown in Figure 9 (middle and right). We use this microsection subsection for a stationary diffusion problem in the unit square which is decomposed into $4 \times 4$ subdomains and discretized with a mesh defined by $H/h = 20$. To obtain a binary coefficient distribution, we set $\rho_1 = 1e6$ in the black part of the microsection and $\rho_2 = 1$ elsewhere. In Table 3, we provide comparison results for the defined microsection problem for different adaptive and approximate FETI-DP coarse spaces. First, we show the condition number (cond) and iteration number (iter) for the adaptive FETI-DP coarse space (*coarse space i*)) as introduced in subsection 2.3. In particular, in this case, we setup and solve the eigenvalue problem (2.17) for each edge of the domain decomposition. As shown in Table 3, the corresponding adaptive FETI-DP coarse space results in a condition number estimate of 9.42 which is clearly independent of the coefficient contrast $1e6$. Furthermore, we obtain an iteration number of 24. We also show the corresponding spectrum of the global preconditioned FETI-DP system in Figure 10 (left). As second coarse space, we consider the frugal coarse space introduced in [11] (*coarse space ii*)). Here, we use generalized weighted averages which are strongly motivated by the adaptive constraints in subsection 2.3 for each edge shared by two subdomains. Note that the frugal coarse space is smaller than the adaptive coarse space since we exclusively implement one frugal constraint for each edge. Even though, to the best of our knowledge, no theoretical condition number bound exists for this heuristic coarse space, the frugal coarse space shows a robust convergence behavior for many heterogeneous coefficient or material distributions; see also [11]. For the microsection problem, however, the obtained condition number is of the order of the coefficient contrast even though the iteration number is still satisfactory; cf. Table 3. The discussed results for the frugal coarse space can be further improved by expanding the frugal edge constraints by a second and a third, at maximum, adaptive edge constraint which are obtained by the edge eigenvalue problem (2.17) (*coarse space iii*)). As we can observe from Table 3 this leads to a condition number and iteration count which is very close to the values obtained by the adaptive coarse space. This confirms our experience from [11] that the computed frugal constraint serves as a good approximation of the adaptive constraint resulting from the first eigenmode or, more general, as a good low-dimensional approximation of the respective adaptive coarse space [29, 30].

Eventually, we present results for an approximate FETI-DP coarse space using approximated constraints as obtained by the trained regression networks $N_l$ and $\widetilde{N}_l,\ l \leq k = 3$, respectively, for the given microsection problem. In particular, we do not solve any edge eigenvalue problems but compute sampling points for the given coefficient distribution for each pair of neighboring subdomains as described in subsection 3.1 and evaluate the trained network for these input data (*coarse space v*)). Implementing the obtained discrete approximations of the first three coarse constraints leads to a condition number of 342.09 and an iteration number of 28. Hence, we clearly obtain a condition number estimate that is independent of the coefficient contrast which ensures a robust convergence behavior. The robustness of the machine learning-based approach is also indicated by the iteration number similar to the adaptive coarse space. Let us also note that when comparing the (min-max-scaled) approximated edge constraints for the microsection problem with the (min-max-scaled) ground truth, i.e., the adaptive constraints resulting from the eigenvalue problem (2.17), we obtain an MSE loss value of 0.0051 for the neural network models with the hyperparameters as in Table 1. As a comparison, when using the same neural network models but without dropout, we obtain an MSE of 0.27. This indicates that sparsification of the neural network by integrating dropout into our regression model indeed helps to increase the generalization properties of the trained neural networks; cf. also the related discussion with respect to the loss values in Table 2 for the training and validation data. Additionally, we propose a hybrid coarse space (*coarse space iv*)) which always imposes a frugal edge constraint [11] for each edge of the domain decomposition and uses the neural networks $N_2$ and $N_3$ or $\widetilde{N}_2$ and $\widetilde{N}_3$ to obtain an approximation of the constraints obtained from second and third eigenmodes. The respective results are also reported in Table 3 and we can see that this hybrid approach can further reduce both the condition number as well as the required number of CG iterations for the solution of the corresponding preconditioned FETI-DP system. For a direct comparison with the adaptive coarse space as presented in subsection 2.3, we also show the spectrum of the global preconditioned FETI-DP system in Figure 10 (right). Obviously, the reported condition number of 156.16 is caused by only one outlier of the spectrum whereas the other eigenvalues have a value lower than 10. Since, as already discussed in detail in [11], the construction of frugal edge constraints has only low computational cost and only a slightly higher computational effort than classic edge constraints [22] we propose the combination of frugal constraints and approximated constraints by a neural network as the favored variant for practical experiments. Let us note that the coarse spaces iv) and i) are of a larger size than the adaptive coarse space i). This is due the fact that we always integrate three learned constraints for each edge of the domain decomposition. The coarse space size can be reduced by only enforcing the learned constraints on edges labeled as critical by the classification neural network defined in [10,12,15]. This only requires, as already mentioned, the additional evaluation of one further neural network. Another possible approach to further reduce the size of the coarse space could be the following: Instead of training the networks to directly

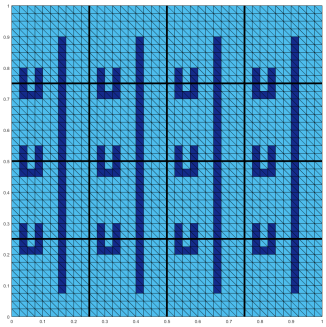

FIGURE 6. *First test problem: combination of channels and U-shaped structures of a high coefficient. Stationary diffusion problem, decomposition of $\Omega = [0, 1]^2$ into $4 \times 4$ subdomains, mesh size defined by $H/h = 10$. Coefficient contrast* 1e6.

| | Coarse space | cond | iter | # c. |
|---|---|---|---|---|
| **i)** | **Adaptive** | **9.42** | **24** | **21** |
| ii) | Frugal | 168,790 | 76 | 13 |
| iii) | Frugal + adaptive | 23.78 | 23 | 21 |
| **iv)** | **Frugal + learned** | **156.16** | **27** | **72** |
| v) | Learned | 342.09 | 28 | 72 |

TABLE 3

*Comparison of different adaptive, heuristic and approximate coarse spaces for FETI-DP for the microsection problem in Figure 9 (right). Decomposition of the unit square into $4 \times 4$ subdomains and a mesh size defined by $H/h = 20$. Coefficient contrast* 1e6. *We denote by **cond** the condition number estimate of the global preconditioned FETI-DP system , by **iter** the number of CG iterations for the iterative solution of the system, and by **# c.** the size of the FETI-DP coarse space.*

predict the edge constraints one could also train different networks to approximate the eigenvectors of the local eigenvalue problem (2.17). Then, one could implement an energy check as proposed in [11] to discard eigenvectors which are not necessary for the robustness of the coarse space. However, in contrast to the approach presented in this paper, the described procedure does still require additional computations and communication in the set-up of the FETI-DP coarse problem. We leave an evaluation of this approach and a comparison to our present method for future research.

Furthermore, to prove that our proposed machine learning algorithm is robust for different microsection problems, we provide average condition numbers and iteration counts for ten different subsection of the dual-phase steel microsection in Table 4. The different subsections used for the numerical computations are shown in Figure 11. As we can observe from Table 4, using the hybrid coarse space iv), i.e., implementing a frugal constraint for each edge and two additional constraints approximated by the trained neural network models, provides robust condition numbers independent of the coefficient contrast as well as iteration numbers which are very close to the adaptive coarse space which is used as the ground truth for the training of the networks (coarse space i)).

## REFERENCES

[1] M. ABADI, A. AGARWAL, P. BARHAM, E. BREVDO, Z. CHEN, C. CITRO, G. S. CORRADO, A. DAVIS, J. DEAN, M. DEVIN, S. GHEMAWAT, I. GOODFELLOW, A. HARP, G. IRV-

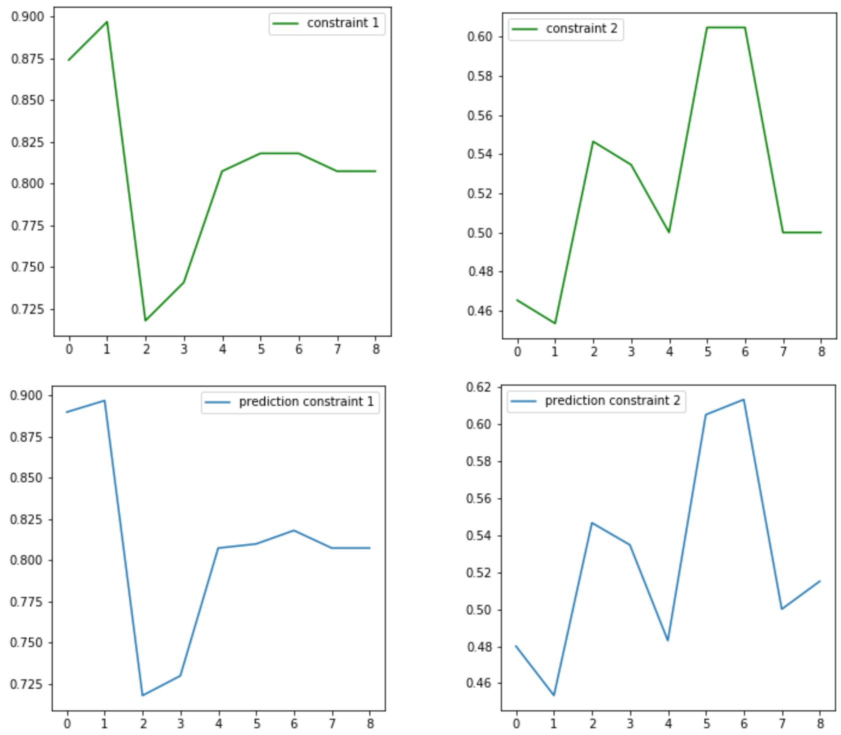

Figure 7. *Results for a first test problem, i.e., the coefficient distribution as shown in* Figure 6*. Green: ground truth, resulting in the iteration number 10 and the condition number estimate 2.8 for the tolerance* TOL = 100*. Blue: prediction as obtained by the neural networks, resulting in iter.: 14, cond.: 35.5.*

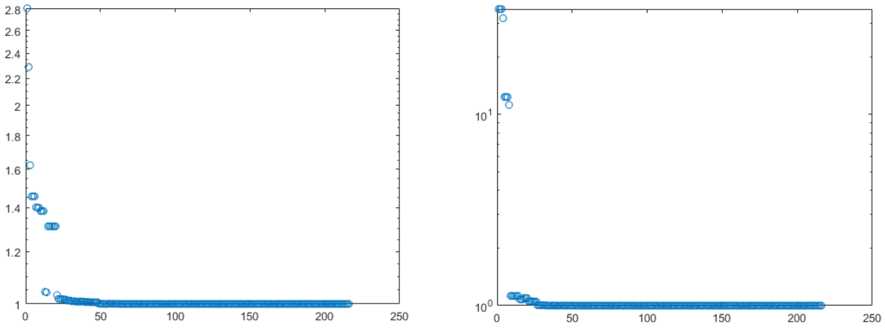

Figure 8. *Eigenvalues of the global, preconditioned FETI-DP system enhanced by the edge constraints as shown in* Figure 7*.* ***Left:*** *Spectrum for the adaptive FETI-DP coarse space.* ***Right:*** *Spectrum for the approximated constraints obtained by the neural networks.*

ing, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, *TensorFlow: Large-scale machine learning on heterogeneous systems*, 2015, https://www.tensorflow.org/. Software available from https://www.tensorflow.org/.

[2] N. Baker, F. Alexander, T. Bremer, A. Hagberg, Y. Kevrekidis, H. Najm, M. Parashar, A. Patra, J. Sethian, S. Wild, and K. Willcox, *Brochure on Basic*

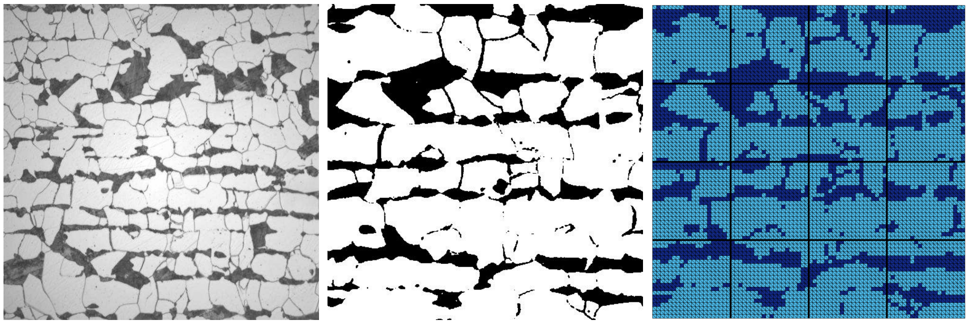

Figure 9. *Heterogeneous microstructure of a dual-phase steel used as a test problem for our trained machine learning models.* ***Left:*** *Complete microsection of a dual-phase steel. Courtesy of Jörg Schröder, University of Duisburg-Essen, Germany, orginating from a cooperation with thyssenkrupp.* ***Middle:*** *Subsection of a microsection of a dual-phase steel obtained from the image on the left. We consider $\rho = 1\mathrm{e}6$ in the black part of the microsection and $\rho = 1$ elsewhere.* ***Right:*** *Visualization of the chosen domain decomposition into $4 \times 4$ subdomains and a mesh size defined by $H/h = 20$ for the given coefficient distribution in the middle which is used in our computations.*

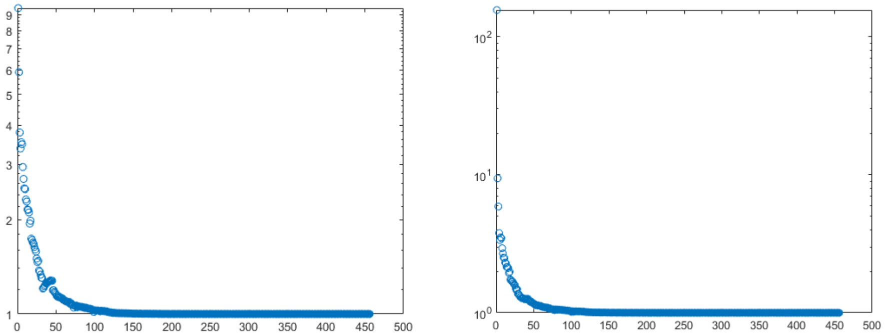

Figure 10. *Eigenvalues of the preconditioned FETI-DP system $M^{-1}F$ for the adaptive FETI-DP coarse space with* $\mathrm{TOL} = 100$ ***(left)*** *and an approximate coarse space which uses the frugal constraint as a first edge constraint and a second and third edge constraint which is approximated by a regression neural network* ***(right)*** *for the microsection problem in Figure 9.*

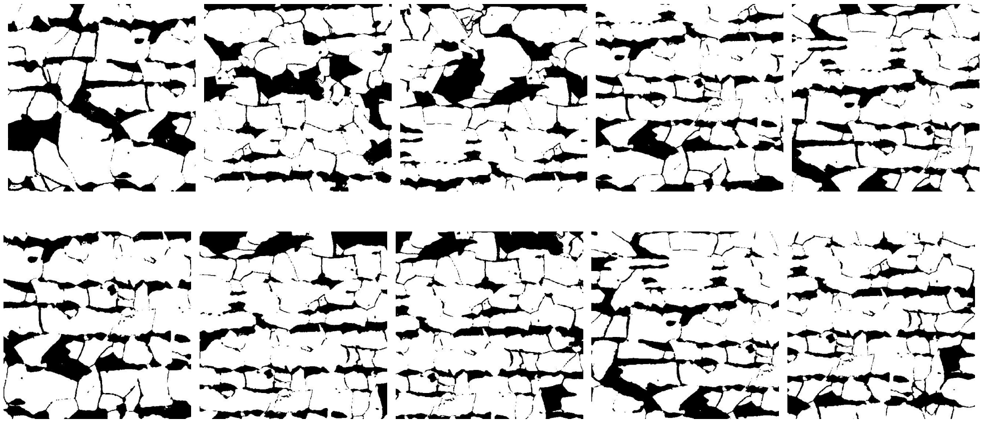

Figure 11. *Ten different subsections of the microsection of a dual-phase steel in Figure 9 (left) suitable for our MATLAB computations to prove the robustness of our machine learning approach.*

| | Coarse space | cond | iter | # c. |
|---|---|---|---|---|
| **i)** | **Adaptive** | **13.21 (15.07)** | **25.6 (27)** | **26.4 (29)** |
| ii) | Frugal | 223,564 (241,994) | 84.4 (94) | 15.2 (17) |
| iii) | Frugal + adaptive | 26.54 (29.33) | 26.0 (28) | 26.4 (29) |
| **iv)** | **Frugal + learned** | **166.32 (189.22)** | **29.2 (30)** | **72.0 (72)** |
| v) | Learned | 389.27 (453.78) | 30.2 (31) | 72.0 (72) |

Table 4

*Comparison of different adaptive, heuristic and approximate coarse spaces for FETI-DP for 10 different subsections of the microsection of a dual-phase steel; see Figure 11. Decomposition of the unit square into $4 \times 4$ subdomains and a mesh size defined by $H/h = 20$. We present the average values for all 10 subsections and the maximum values, respectively (in parentheses). Coefficient contrast* 1e6*. See Table 3 for the column labeling.*

*Research Needs for Scientific Machine Learning: Core Technologies for Artificial Intelligence. USDOE Office of Science (SC) (United States)*, (2018), https://doi.org/10.2172/1484362, https://www.osti.gov/biblio/1484362.

[3] C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006.

[4] E. Chung, H.-H. Kim, M.-F. Lam, and L. Zhao, *Learning adaptive coarse spaces of BDDC algorithms for stochastic elliptic problems with oscillatory and high contrast coefficients*, Mathematical and Computational Applications, 26 (2021), https://doi.org/10.3390/mca26020044, https://www.mdpi.com/2297-8747/26/2/44.

[5] C. Farhat, M. Lesoinne, P. LeTallec, K. Pierson, and D. Rixen, *FETI-DP: a dual-primal unified FETI method. I. A faster alternative to the two-level FETI method*, Internat. J. Numer. Methods Engrg., 50 (2001), pp. 1523–1544.

[6] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*, MIT press Cambridge, 2016.

[7] L. Gu, W. Zhang, J. Liu, and X.-C. Cai, *Decomposition and composition of deep convolutional neural networks and training acceleration via sub-network transfer learning*, Electronic Transactions on Numerical Analysis, 56 (2022), pp. 157–186.

[8] A. Heinlein, A. Klawonn, J. Knepper, and O. Rheinbach, *Adaptive GDSW coarse spaces for overlapping Schwarz methods in Three Dimensions*, SIAM Journal on Scientific Computing, 41 (2019), pp. A3045–A3072.

[9] A. Heinlein, A. Klawonn, M. Lanser, and J. Weber, *Predicting the geometric location of critical edges in adaptive GDSW overlapping domain decomposition methods using deep learning*. TR series, Center for Data and Simulation Science, University of Cologne, Germany, Vol. 2021-2. https://kups.ub.uni-koeln.de/id/eprint/36257. Accepted for publication in the proceedings of the International Conference on Domain Decomposition Methods 26, Springer LNCSE, August 2021.

[10] A. Heinlein, A. Klawonn, M. Lanser, and J. Weber, *Machine Learning in Adaptive Domain Decomposition Methods - Predicting the Geometric Location of Constraints*, SIAM J. Sci. Comput., 41 (2019), pp. A3887–A3912.

[11] A. Heinlein, A. Klawonn, M. Lanser, and J. Weber, *A Frugal FETI-DP and BDDC Coarse Space for Heterogeneous Problems*, Electr. Trans. Numer. Anal., 53 (2020), pp. 562–591.

[12] A. Heinlein, A. Klawonn, M. Lanser, and J. Weber, *Machine Learning in Adaptive FETI-DP - A Comparison of Smart and Random Training Data*, in Domain decomposition methods in science and engineering XXV, vol. 138 of Lect. Notes Comput. Sci. Eng., Springer, 2020, pp. 218–226.

[13] A. Heinlein, A. Klawonn, M. Lanser, and J. Weber, *Combining Machine Learning and Adaptive Coarse Spaces - A Hybrid Approach for Robust FETI-DP Methods in Three Dimensions*, SIAM J. Sci. Comput., 43 (2021), pp. S816–S838, https://doi.org/10.1137/20M1344913. Special Section Copper Mountain 2020.

[14] A. Heinlein, A. Klawonn, M. Lanser, and J. Weber, *Combining Machine Learning and Domain Decomposition Methods for the Solution of Partial Differential Equations–A Review*, GAMM-Mitt., 44 (2021), p. e202100001, https://doi.org/10.1002/gamm.202100001, https://doi.org/10.1002/gamm.202100001.

[15] A. Heinlein, A. Klawonn, M. Lanser, and J. Weber, *Machine Learning in Adaptive FETI-DP - Reducing the Effort in Sampling*, in Numerical Mathematics and Advanced Applications ENUMATH 2019, vol. 139 of Lect. Notes Comput. Sci. Eng., Springer, 2021, pp. 593–603.

[16] G. Karypis and V. Kumar, *METIS: Unstructured Graph Partitioning and Sparse Matrix Ordering System, Version 4.0*. http://www.cs.umn.edu/~metis, 2009.

[17] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, arXiv preprint arXiv:1412.6980, (2014).

[18] A. Klawonn, M. Kühn, and O. Rheinbach, *Adaptive coarse spaces for FETI-DP in three dimensions*, SIAM J. Sci. Comput., 38 (2016), pp. A2880–A2911.

[19] A. Klawonn, M. Kühn, and O. Rheinbach, *Adaptive FETI-DP and BDDC methods with a generalized transformation of basis for heterogeneous problems*, Electron. Trans. Numer. Anal., 49 (2018), pp. 1–27.

[20] A. Klawonn, P. Radtke, and O. Rheinbach, *FETI-DP methods with an adaptive coarse space*, SIAM J. Numer. Anal., 53 (2015), pp. 297–320.

[21] A. Klawonn, P. Radtke, and O. Rheinbach, *A comparison of adaptive coarse spaces for iterative substructuring in two dimensions*, Electron. Trans. Numer. Anal., 45 (2016), pp. 75–106.

[22] A. Klawonn and O. Rheinbach, *Robust FETI-DP methods for heterogeneous three dimensional elasticity problems*, Comput. Methods Appl. Mech. Engrg., 196 (2007), pp. 1400–1414.

[23] A. Klawonn, O. Rheinbach, and O. B. Widlund, *An analysis of a FETI-DP algorithm on irregular subdomains in the plane*, SIAM J. Numer. Anal., 46 (2008), pp. 2484–2504.

[24] A. Klawonn and O. Widlund, *FETI and Neumann-Neumann iterative substructuring methods: connections and new results*, Communications on Pure and Applied Mathematics, 54 (2001), pp. 57–90.

[25] A. Klawonn and O. B. Widlund, *Dual-primal FETI methods for linear elasticity*, Comm. Pure Appl. Math., 59 (2006), pp. 1523–1572.

[26] A. Klawonn, O. B. Widlund, and M. Dryja, *Dual-primal FETI methods for three-dimensional elliptic problems with heterogeneous coefficients*, SIAM J. Numer. Anal., 40 (2002), pp. 159–179.

[27] M. J. Kühn, *Adaptive FETI-DP and BDDC methods for highly heterogeneous elliptic finite element problems in three dimensions*, PhD thesis, Universität zu Köln, Cologne, 2018. Online available at http://kups.ub.uni-koeln.de/id/eprint/8234.

[28] J. Li and X.-C. Cai, *Summation pollution of principal component analysis and an improved algorithm for location sensitive data*, Numerical Linear Algebra with Applications, 28 (2021), p. e2370.

[29] J. Mandel and B. Sousedík, *Adaptive selection of face coarse degrees of freedom in the BDDC and the FETI-DP iterative substructuring methods*, Comput. Methods Appl. Mech. Engrg., 196 (2007), pp. 1389–1399.

[30] J. Mandel, B. Sousedík, and J. Šístek, *Adaptive BDDC in three dimensions*, Math. Comput. Simulation, 82 (2012), pp. 1812–1831.

[31] J. Mandel and R. Tezaur, *On the convergence of a dual-primal substructuring method*, Numer. Math., 88 (2001), pp. 543–558.

[32] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, *Scikit-learn: Machine learning in Python*, Journal of Machine Learning Research, 12 (2011), pp. 2825–2830.

[33] L. Prechelt, *Early stopping-but when?*, in Neural Networks: Tricks of the trade, Springer, 1998, pp. 55–69.

[34] P. Radtke, *Adaptive Coarse Spaces for FETI-DP and BDDC Methods*, PhD thesis, Universität zu Köln, 2015. Online available at http://kups.ub.uni-koeln.de/id/eprint/6426.

[35] Y. Saad, *Iterative methods for sparse linear systems*, SIAM, 2nd ed., 2003.

[36] B. Sousedík, *Comparison of some domain decomposition methods*, PhD thesis, Czech Technical University in Prague, 2008.

[37] A. Toselli and O. Widlund, *Domain Decomposition Methods—Algorithms and Theory*, vol. 34 of Springer Series in Computational Mathematics, Springer-Verlag, Berlin, 2005.

[38] J. Weber, *Efficient and robust FETI-DP and BDDC methods – Approximate coarse spaces and deep learning-based adaptive coarse space*, PhD thesis, Universität zu Köln, 2022. Online available at http://kups.ub.uni-koeln.de/id/eprint/55179.