Towards modular Hierarchical Poincaré-Steklov solvers

Michal Outrata and José Pablo Lucero Lorca

Abstract

We revisit the Hierarchical Poincaré–Steklov (HPS) method for the Poisson equation using standard Q1 finite elements, building on the original in [21]. While corner degrees of freedom were implicitly handled in that work, subsequent spectral-element implementations have typically avoided them. In Q1-FEM, however, corner coupling cannot be factored out, and we show how the HPS merge procedure naturally accommodates it when corners are enclosed by elements. This clarification bridges a conceptual gap between algebraic Schur-complement methods and operator-based formulations, providing a consistent path for the FEM community to adopt HPS to retain the Poincaré–Steklov interpretation at both continuous and discrete levels.

1 Introduction

Hierarchical Poincaré–Steklov (HPS) solvers are a class of hierarchical direct solvers designed for elliptic PDEs; the name was coined in [12, 22] but some of the ideas are in [4, 23, 15, 11]. Starting with many subdomains, the goal is to recursively merge local boundary operators – typically *Dirichlet-to-Neumann* (DtN) or *Impedance-to-Impedance* (ItI) maps – constructing a *global* one that we apply to the problem. The HPS approach achieves high accuracy and near-optimal complexity, combining ideas present in hierarchical matrix computations ($\mathcal{H}$), domain decomposition (DD) and direct solvers, i.e., it is poised to be the keystone connecting several communities, e.g., BDDC or FETI; in words of one of the authors after reading [18]: "For the love of God, they need to start talking to each other!". In our opinion this communication has been limited also due to the strong spectral element methods (SEM) background of the HPS community; the formulation, discretization and computation in HPS are often entangled together, making it difficult to relate pros and cons of the "package" to its parts. We want to provide a *modular* alternative, approachable for readers across multiple communities and for the sake of space we focus on the *corner points*[1], which are *routinely* considered an obstacle in the HPS context [12, 22, 13, 1, 2, 20]. Many aspects of what follows can be found *somewhere* in the literature, sometimes with limited references to the other fields but, to the best of our knowledge, a modular HPS exposition is *nowhere to be found* in the HPS literature.

As mentioned, the HPS community is using predominantly SEM on tensor product grids – it offers high (possible) accuracy *and* lets us avoid the *corner points*, e.g., with the Gauss-Legendre points. If the corner points appear, in SEM they usually come *decoupled* from

[1] We note that in the DD community we usually use the term *cross points*, e.g., [8, 6].

the interior nodes or can be avoided altogether by modifying the spectral discretization, see, e.g., [12, 13, 1, 2]. HPS using finite differences or finite volumes, [11, 5], also rely on avoiding the corner-coupling issue that arises in, e.g., FEM. The rationale is both *analytical* but also practical: the used Poincaré-Steklov (PS) operators need not be well-defined in the presence of corners and the tensor-product basis naturally isolates corner DoFs [9, 7, 24, 17]. Hence, for many new readers, the HPS methods are intrinsically connected with such discretization schemes.

However, essentially the same problems have been studied also from the algebraic perspective, e.g., nested dissection, hierarchical semi-separable and hierarchical multifrontal techniques, e.g., [10, 14, 26, 25], are *purely algebraic*: they operate directly on the discrete system, exploiting observed numerical blockwise low-rankness for compression and factorization. Although the foundational work on hierarchical matrices, see e.g., [3, 16], is built on the continuous operators, to the best of our knowledge, it does not include PS operators, nor incorporate static condensation or skeletonization. The recursive skeletonization can be viewed within the multilevel DD or multigrid framework – in [19], HPS has been identified with a specific multigrid V-cycle.

Our primary goal below is to separate the *discretization method* and the way in which the method treats the *corner points*, thereby helping to build the modular view of HPS. For that reason we choose the standard Poisson problem on a rectangle and use the Q1-FEM discretization on a tensor product grid, where the basis functions firmly couple the corner point DoFs with others. In the HPS community, this would be considered a major issue as it prevents a straightforward definition of the local DtN. However, having discretized we show this can be resolved with little extra effort. We are not aware of the HPS and Q1-FEM coupling (or other simple low-order FEM) anywhere in the literature; this set-up should also provide a simple entry point into HPS methods for broader audience and FEM enthusiasts will notice that we do not rely on the Q1 elements in any way. We again highlight that in *different communities* and in different context similar ideas already exists, see, e.g. [18], where the authors consider mixed-order curl-conforming FEM discretization for time-harmonic Maxwell equations in $\mathbb{R}^3$ – an involved setting in which HPS is not mentioned but corner points and edge points are considered.

2 The HPS method with corners

As noted above, we consider the simplest model problem

$$\Delta u = f \quad \text{in } \Omega := (\alpha, \beta) \times (\gamma, \delta) \quad \text{and} \quad u = g \quad \text{on } \partial\Omega, \tag{1}$$

and start by outlining the structure of a general HPS method:

1. *Partition* – partition the domain Ω into subdomains.
2. *Discretization & Assembly* – formulate, discretize and assemble the subdomain solution boundary operators for the subdomains.

3. *Merge* – merge the neighboring solution boundary operators and store the result.

4. *Recursion* – recurse and continue merging until we reach the entire domain Ω.

5. *Application* – given data, apply the global solution boundary operator and calculate the solution on the boundaries of the subdomains.

6. *Reconstruction* – reconstruct the solution in the subdomains from the boundaries.

As per the *partition* stage, we the standard, grid-like set-up

$$\Omega_e = [a_e^{(x_1)}, b_e^{(x_1)}] \times [a_e^{(x_2)}, b_e^{(x_2)}] \subset \Omega,$$

see Figure 1-right, forming a non-overlapping decomposition of Ω with corner points; other decompositions can be treated identically [18, Figure 2].

2.1 The *discretization & assembly* stage

The analytical background. We are interested in constructing the *subdomain solution boundary operators* – dealing with the Poisson problem, those are the subdomain DtNs. Let u_e denote the solution on the subdomain Ω_e, i.e.,

$$-\Delta u_e = f \quad \text{in } \Omega_e, \quad \text{and} \quad u_e = g_e \quad \text{on } \partial\Omega_e. \tag{2}$$

We can split u_e into the sum of the harmonic lift of the boundary data g_e, denoted by $u_e^{(g)}$ and the particular solution of the interior load f, denoted by $u_e^{(f)}$, obtaining

$$-\Delta u_e^{(g)} = 0 \text{ in } \Omega_e \;\&\; u_e^{(g)} = g_e \text{ on } \partial\Omega_e \quad \text{and} \quad -\Delta u_e^{(f)} = f \text{ in } \Omega_e \;\&\; u_e^{(f)} = 0 \text{ on } \partial\Omega_e.$$

Analogously, we also split the Neumann trace of the solution, denoted by $\partial_n u_e$,

$$\partial_n u_e = \partial_n u_e^{(g)} + \partial_n u_e^{(f)} =: \Lambda_e g_e + q_e \tag{3}$$

featuring homogeneous DtN Λ_e and the particular Neumann trace q_e on Ω_e.

Discretization. We first introduce grid nodes in Ω^e in a tensor-product manner along the x_1 and x_2 axis. On this grid we consider the Q_1 finite element discretization of (2) and index the local DoFs by integer pairs $\iota_e = (i, j)$; see Figure 1 for the details. Applying integration by parts to the continuous weak form of (2) gives

$$\int_{\Omega_e} \nabla u_e \cdot \nabla \phi_m \,\mathrm{d}\mathbf{x} = \int_{\Omega_e} f\,\phi_m \,\mathrm{d}\mathbf{x} + \int_{\partial\Omega_e} (\partial_n u_e)\,\phi_m \,\mathrm{d}s, \quad m \in \iota_e$$

and then, after approximating u_e in the Q1-FEM basis and reordering the DoFs, we get the discretized system for the unknown coefficients $\mathbf{u}_e$

$$\begin{bmatrix} A_e & B_e \\ C_e & D_e \end{bmatrix} \begin{bmatrix} \mathbf{u}_e^{\mathrm{int}} \\ \mathbf{u}_e^{\partial} \end{bmatrix} = \begin{bmatrix} \mathbf{f}_e^{\mathrm{int}} \\ \mathbf{f}_e^{\partial} \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \partial_n \mathbf{u}_e \end{bmatrix}, \tag{4}$$

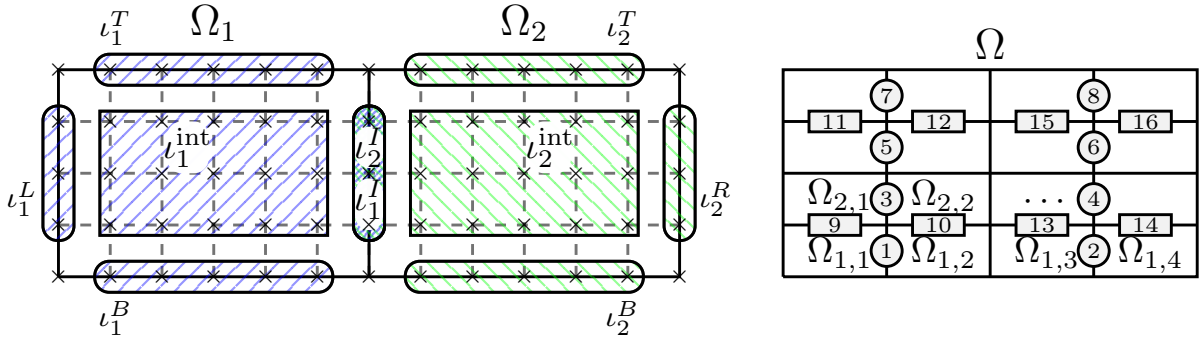

Figure 1: Left: two neighbouring subdomains Ω_1, Ω_2 with the index sets of the grids. We have $\iota_1^I \equiv \iota_1^R$ and $\iota_2^I \equiv \iota_2^L$ and also see the corner index sets ι_1^C, ι_2^C, although not separately highlighted. Finally, we set $\iota_e^\partial := \iota_e^L \cup \iota_e^R \cup \iota_e^T \cup \iota_e^B \cup \iota_e^C$ so that $\iota_e = \iota_e^{\text{int}} \cup \iota_e^\partial$ for any e. Right: (incomplete) illustration of the nested dissection merge hierarchy ordering, see [21, Appendix A].

where $\partial_n \mathbf{u}_e$ is the Q1-FEM discretization of the Neumann trace of $\partial_n u_e$ at ι_e^{int}.

Assembly. Equation (4) shows that the *negative residual along the interface* is the finite element representation of the approximate normal fluxes along the boundary. As these fluxes are unknown, the equations in the second block-row give the formula,

$$\partial_n \mathbf{u}_e = C_e \mathbf{u}_e^{\text{int}} + D_e \mathbf{u}_e^\partial - \mathbf{f}_e^\partial,$$

which, after elimination of the interior DoFs, becomes

$$\partial_n \mathbf{u}_e \equiv \mathbf{r}_e^\partial = \left(D_e - C_e A_e^{-1} B_e\right) \mathbf{u}_e^\partial + C_e A_e^{-1} \mathbf{f}_e^{\text{int}} - \mathbf{f}_e^\partial =: S_e \mathbf{u}_e^\partial + \mathbf{h}_e. \tag{5}$$

This relation mirrors the continuous decomposition (3): the Schur complement S_e acts as the discrete homogeneous DtN operator mapping boundary values $\mathbf{u}_e^\partial$ to their induced boundary fluxes, while $\mathbf{h}_e$ represents the discrete flux produced by the interior load under homogeneous Dirichlet conditions. The *assembly* stage of HPS methods consists of computing (or approximating) the matrices S_e so that the right-hand side of (5) can be evaluated *rapidly* in the *application* stage.

2.2 The *merge* stage

Having two subdomains, say Ω_1, Ω_2, with finished *assembly* stage that share an interface, we want to assemble the solution boundary operator for $\Omega_1 \cup \Omega_2$. The ordering in which we will pick the subdomain pairs matters as it highly influences the parallelizability of the resulting solver; we follow the nested dissection ordering as illustrated in Figure 1-right; first we merge *horizontally* and then *vertically*.

Horizontal merge (left Ω_1 and right Ω_2). The true solution of (1) is continuous and has balanced fluxes (i.e., residuals) across the interface, i.e.,

$$\mathbf{u}_1(\iota_1^I) = \mathbf{u}_2(\iota_2^I) \quad \text{and} \quad \mathbf{r}_1(\iota_1^I) + \mathbf{r}_2(\iota_2^I) = 0. \tag{6}$$

Recalling (5) and blocking it according to $\iota_e^\partial = \iota_e^I \cup \iota_e^{\partial\setminus I}$ for $e = 1, 2$ gives

$$\begin{bmatrix} \mathbf{r}_e(\iota_e^{\partial\setminus I}) \\ \mathbf{r}_e(\iota_e^I) \end{bmatrix} = \begin{bmatrix} S_e(\iota_e^{\partial\setminus I}, \iota_e^{\partial\setminus I}) & S_e(\iota_e^{\partial\setminus I}, \iota_e^I) \\ S_e(\iota_e^I, \iota_e^{\partial\setminus I}) & S_e(\iota_e^I, \iota_e^I) \end{bmatrix} \begin{bmatrix} \mathbf{u}_e(\iota_e^{\partial\setminus I}) \\ \mathbf{u}_e(\iota_e^I) \end{bmatrix} + \begin{bmatrix} \mathbf{h}_e(\iota_e^{\partial\setminus I}) \\ \mathbf{h}_e(\iota_e^I) \end{bmatrix}. \tag{7}$$

Summing the second block-rows for $e = 1, 2$, using (6) and reordering gives

$$\big(S_1(\iota_1^I, \iota_1^I) + S_2(\iota_2^I, \iota_2^I)\big)\, \mathbf{u}_1(\iota_1^I) = -\sum_{e=1,2} \mathbf{h}_e(\iota_e^I) + S_e(\iota_e^I, \iota_e^{\partial\setminus I})\mathbf{u}_e(\iota_e^{\partial\setminus I}). \tag{8}$$

Returning to (7), we concatenate the equations for the residuals on the "merged boundary" $r_e(\iota_e^{\partial\setminus I}), e = 1, 2$, use (8) in both and reorder so as to obtain

$$\begin{bmatrix} \mathbf{r}_1(\iota_1^{\partial\setminus I}) \\ \mathbf{r}_2(\iota_2^{\partial\setminus I}) \end{bmatrix} = S^{\mathrm{H}} \begin{bmatrix} \mathbf{u}_1(\iota_1^{\partial\setminus I}) \\ \mathbf{u}_2(\iota_2^{\partial\setminus I}) \end{bmatrix} + \mathbf{h}^{\mathrm{H}}, \tag{9}$$

i.e., the *horizontally* merged boundary solution operators as in (5) with

$$\begin{aligned} S^{\mathrm{H}} = &\begin{bmatrix} S_1(\iota_1^{\partial\setminus I}, \iota_1^{\partial\setminus I}) & 0 \\ 0 & S_2(\iota_2^{\partial\setminus I}, \iota_2^{\partial\setminus I}) \end{bmatrix} - \\ &\begin{bmatrix} S_1(\iota_1^{\partial\setminus I}, \iota_1^I) \\ S_2(\iota_2^{\partial\setminus I}, \iota_2^I) \end{bmatrix} \big(S_1(\iota_1^I, \iota_1^I) + S_2(\iota_2^I, \iota_2^I)\big)^{-1} \begin{bmatrix} S_1(\iota_1^I, \iota_1^{\partial\setminus I}) & S_2(\iota_2^I, \iota_2^{\partial\setminus I}) \end{bmatrix}, \\ \mathbf{h}^{\mathrm{H}} = &\begin{bmatrix} \mathbf{h}_1(\iota_1^{\partial\setminus I}) \\ \mathbf{h}_2(\iota_2^{\partial\setminus I}) \end{bmatrix} - \begin{bmatrix} S_1(\iota_1^{\partial\setminus I}, \iota_1^I) \\ S_2(\iota_2^{\partial\setminus I}, \iota_2^I) \end{bmatrix} \big(S_1(\iota_1^I, \iota_1^I) + S_2(\iota_2^I, \iota_2^I)\big)^{-1} \big(\mathbf{h}_1(\iota_1^I) + \mathbf{h}_2(\iota_2^I)\big). \end{aligned}$$

Vertical merge (bottom Ω_1 and top Ω_2). Say we have "horizontally" merged the boundary solution operators for two couples of subdomains Ω_{1L}, Ω_{1R} and Ω_{2L}, Ω_{2R}, e.g. the merges ① and ③ in Figure 1, and we are ready to merge along the vertical interface – labeled 9 and 10 – *and then also at the corner point enclosed between the already merged interfaces*. First, keeping the enclosed corner DoF, indexed[2] by $\iota_\Gamma^C \equiv c$, uneliminated, the steps in merges 9 and 10 carry through identically to the horizontal merges ① or ③, only now the index set ι_e^I is *disjoint*, e.g., $\iota_1^I = \iota_{1L}^T \cup \iota_{1R}^T$, and the index sets $\iota_1^{\partial\setminus I}, \iota_2^{\partial\setminus I}$ contain the enclosed corner index ι_Γ^C. That is, we have

$$\begin{bmatrix} \mathbf{r}_1(\iota_1^{\partial\setminus I}) \\ \mathbf{r}_2(\iota_2^{\partial\setminus I}) \end{bmatrix} = S^{\mathrm{V}} \begin{bmatrix} \mathbf{u}_1(\iota_1^{\partial\setminus I}) \\ \mathbf{u}_2(\iota_2^{\partial\setminus I}) \end{bmatrix} + \mathbf{h}^{\mathrm{V}}, \tag{10}$$

[2]We use $\iota_\Gamma^C \equiv c$ as an *absolute* index across the indexing in the four subdomains $\Omega_{1L}, \Omega_{1R}, \Omega_{2L}, \Omega_{2R}$, even though the point has likely a different index in each of them.

with the *vertically* merged boundary solution operators as in (5), i.e.,

$$S^{\mathrm{V}} = \begin{bmatrix} S_1(\iota_1^{\partial\setminus I}, \iota_1^{\partial\setminus I}) & 0 \\ 0 & S_2(\iota_2^{\partial\setminus I}, \iota_2^{\partial\setminus I}) \end{bmatrix} - \begin{bmatrix} S_1(\iota_1^{\partial\setminus I}, \iota_1^{I}) \\ S_2(\iota_2^{\partial\setminus I}, \iota_2^{I}) \end{bmatrix} \big(S_1(\iota_1^I, \iota_1^I) + S_2(\iota_2^I, \iota_2^I)\big)^{-1} \begin{bmatrix} S_1(\iota_1^I, \iota_1^{\partial\setminus I}) & S_2(\iota_2^I, \iota_2^{\partial\setminus I}) \end{bmatrix},$$

$$\mathbf{h}^{\mathrm{V}} = \begin{bmatrix} \mathbf{h}_1(\iota_1^{\partial\setminus I}) \\ \mathbf{h}_2(\iota_2^{\partial\setminus I}) \end{bmatrix} - \begin{bmatrix} S_1(\iota_1^{\partial\setminus I}, \iota_1^{I}) \\ S_2(\iota_2^{\partial\setminus I}, \iota_2^{I}) \end{bmatrix} \big(S_1(\iota_1^I, \iota_1^I) + S_2(\iota_2^I, \iota_2^I)\big)^{-1} \big(\mathbf{h}_1(\iota_1^I) + \mathbf{h}_2(\iota_2^I)\big).$$

Corner merge (corner of $\Omega_{1L}, \Omega_{1R}, \Omega_{2L}, \Omega_{2R}$). Analogously to (6), we have

$$\mathbf{u}_1(c) = \mathbf{u}_2(c) = \mathbf{u}(c) \quad \text{and} \quad \sum_{e=1,2} \mathbf{r}_e(c) = 0. \tag{11}$$

Collecting the $\iota_\Gamma^C \equiv c$ equations from (10) and inserting them into (11) gives

$$\sum_{e=1,2} S_e(c,c)\,\mathbf{u}(c) = -\sum_{e=1,2} S_e(c, E_e)\mathbf{u}_e(E_e) + \mathbf{h}_e(c),$$

where E_e are the indices of points on $\partial\Omega_e \cap \partial(\Omega_1 \cup \Omega_2)$. Solving for $\mathbf{u}(c)$ and inserting back in (10) gives a system on the exterior index set $E := E_1 \cup E_2$ for fluxes

$$\begin{aligned} \mathbf{r}(E) &= S^{\mathrm{corner}}\,\mathbf{u}(E) + \mathbf{h}^{\mathrm{corner}} \text{ with} \\ S^{\mathrm{corner}} &= S^{\mathrm{V}}(E,E) - S^{\mathrm{V}}(E,c)\Big(S^{\mathrm{V}}(c,c)\Big)^{-1} S^{\mathrm{V}}(c,E), \\ \mathbf{h}^{\mathrm{corner}} &= \mathbf{h}^{\mathrm{V}}(E) - S^{\mathrm{V}}(E,c)\Big(S^{\mathrm{V}}(c,c)\Big)^{-1} \mathbf{h}^{\mathrm{V}}(c), \end{aligned}$$

with identical structure of the resulting boundary solution operator as in (10) or (9).

What is the point? First, this treatment of the corner points is fundamentally different to the "change of basis" approach used, e.g., in [22] – no retabulation, rather following the same ground ideas behind HPS. Second, it is also fundamentally different from the "ignore" approach used, e.g., in [9], as that is simply not an option due to the corner point DoFs coupling. Third, this is in fact very similar to [21], but outlined only within the SEM context with the aforementioned benefits. Our point is that the corner points should be merged once the surrounding interfaces have been merged to maximize the efficiency and doing that follows analogous steps used before, even when using fully coupled corner DoFs of FEM.

2.3 The *recursion* stage

Having successfully eliminated the interface and enclosed corner DoFs, we recurse and continue until reaching a problem on $\partial\Omega$, where the Dirichlet trace is known. Let $\Omega = \bigcup_e \Omega_e$ be

Table 1: Average speedup (left) and break-even solves (right) relative to MATLAB's backslash.

	Speedup				Break-even solves			
# elements p/subdomain / # subdomains	4×4	8×8	16×16	32×32	4×4	8×8	16×16	32×32
4×4	1	3	5	12	N/A	4	2	2
8×8	2	6	9	17	5	4	3	2
16×16	3	7	21	28	5	5	4	3
32×32	4	11	26	35	7	5	4	4

the decomposition into subdomains, each with its local DtN $(S_e, \mathbf{h}_e)$. The global system on the *skeleton* – i.e., on $\bigcup_e \partial\Omega_e$ – reads

$$S\,\mathbf{u}^{(\partial)} = \mathbf{h}^{(\partial)},$$

where S is built from S_e based on the interface continuity and flux balances. We order the boundary indices $\iota^{(\partial)} = \bigcup_e \iota_e^{\partial}$ by the merging hierarchy: first domain boundaries, then merged interfaces and enclosed corners (following the nested dissection ordering). Hence the *to-internalize* indices come after the *active* exterior indices,

$$\iota^{(\partial)} = \big(\iota_{\text{ext}}^{(1)},\ \iota_{\text{merge}}^{(1)},\ \iota_{\text{merge}}^{(2)},\ \ldots,\ \iota_{\text{merge}}^{(L)}\big),$$

where $\iota_{\text{merge}}^{(\ell)}$ are the indices eliminated at recursion level ℓ. Then S has the structure

$$S = \begin{bmatrix} S_{EE} & S_{EM} \\ S_{ME} & S_{MM} \end{bmatrix},$$

where E and M representing the exterior and to-be-merged blocks. The *merge* step corresponds precisely to eliminating the S_{MM} block via its Schur complement:

$$\widehat{S}_{EE} = S_{EE} - S_{EM}\,S_{MM}^{-1}\,S_{ME} \quad \text{and} \quad \widehat{\mathbf{h}}_E = \mathbf{h}_E - S_{EM}\,S_{MM}^{-1}\,\mathbf{h}_M,$$

where $(\widehat{S}_{EE}, \widehat{\mathbf{h}}_E)$ defines the reduced DtN operator and right-hand side of the updated skeleton after that merge. Proceeding recursively the calculation always follows the same two-domain pattern, possibly extended by enclosed-corner junctions. That is the HPS skeleton solver can be interpreted as a single recursive Schur complement elimination applied to the global skeleton matrix S. Each recursion step in HPS corresponds to eliminating the block $(\iota_{\text{merge}}^{(\ell)}, \iota_{\text{merge}}^{(\ell)})$ corresponding to indices merged at that level of the hierarchy. At the top of the recursion we get the final reduced operator $S^{(L)}$ on $\partial\Omega$, whose equilibrium equation represents the DtN map of Ω.

3 Numerical illustration

We conclude by showcasing the performance of HPS, implemented in MATLAB with very few optimizations. The build stage is computationally costly so the method is useful when

we have several different right-hand sides. We run our tests on a laptop with 32GB RAM and a i7-12700H Intel microprocessor with six 4.7 GHz performance cores, eight 3.5GHz efficient cores and twenty total threads with performance-core hyperthreading. We take the MATLAB's backslash for the skeleton problem as the benchmark (ignoring the reconstruction). This comparison is stricter than a full solution comparison, where backslash would process a significantly larger operator.

References

[1] T. Babb, A. Gillman, S. Hao, and P.-G. Martinsson. An accelerated Poisson solver based on a multidomain spectral discretization. *BIT Num. Math.*, 58(4):851–879, 2018.

[2] N. N. Beams, A. Gillman, and R. J. Hewett. A parallel shared-memory implementation of a high-order accurate solution technique for variable coefficient Helmholtz problems. *Comp. & Math. w. Appl.*, 79(4):996–1011, 2020.

[3] S. Börm, L. Grasedyck, and W. Hacksbusch. Hierarchical Matrices: Lecture notes no. 21, 2006.

[4] Y. Chen. A fast, direct algorithm for the Lippmann–Schwinger integral equation in two dimensions. *Adv. in Comp. Math.*, 16(2-3):175–190, 2002.

[5] D. Chipman, D. Calhoun, and C. Burstedde. A fast direct solver for elliptic PDEs on a hierarchy of adaptively refined quadtrees. *arXiv:2402.14936*, 2024.

[6] X. Claeys and E. Parolin. Robust treatment of cross-points in optimized schwartz methods. *Numer. Math.*, 151:405–442, 2022.

[7] D. Fortunato. A high-order fast direct solver for surface PDEs. *SIAM J. on Sci. Comp.*, 46(4):A2582–A2606, 2024.

[8] M. J. Gander and K. Santugini-Repiquet. Cross-points in domain decomposition methods with a finite element discretization. *ETNA*, 45:219–240, 2016.

[9] P. Geldermans and A. Gillman. An adaptive high order direct solution technique for elliptic boundary value problems. *SIAM J. on Sci. Comp.*, 41(1):A292–A315, 2019.

[10] A. George. Nested dissection of a regular finite element mesh. *SIAM J. on Num. Anal.*, 10(2):345–363, 1973.

[11] A. Gillman. *Fast direct solvers for elliptic partial differential equations*. Ph.D. Thesis, University of Colorado Boulder, 2011.

[12] A. Gillman and P.-G. Martinsson. A direct solver with $\mathcal{O}(N)$ complexity for variable coefficient elliptic PDEs discretized via a high-order composite spectral collocation method. *SIAM J. on Sci. Comp.*, 36(4):A2023–A2046, 2014.

[13] A. Gillman, A. H. Barnett, and P.-G. Martinsson. A spectrally accurate direct solution technique for frequency-domain scattering problems with variable media. *BIT Num. Math.*, 55(1):141–170, 2015.

[14] L. Grasedyck, R. Kriemann, and S. Le Borne. Domain decomposition based $\mathcal{H}$-LU preconditioning. *Numerische Math.*, 112(4):565–600, 2009.

[15] L. Greengard, D. Gueyffier, P.-G. Martinsson, and V. Rokhlin. Fast direct solvers for integral equations in complex three-dimensional domains. *Acta Num.*, 18:243–275, 2009.

[16] W. Hackbusch. *Hierarchical Matrices: Algorithms and Analysis*. Springer Berlin, 2015.

[17] Y. Kump, A. Yesypenko, and P.-G. Martinsson. A two-level direct solver for the hierarchical Poincaré–Steklov method. *arXiv:2503.04033*, 2025.

[18] J. Lu and J.-F. Lee. A compression scheme for domain decomposition method in solving electromagnetic problems. *Journal of Computational Physics*, 503:112824, 2024.

[19] J. P. Lucero Lorca. Towards a multigrid preconditioner interpretation of hierarchical Poincaré–steklov solvers. *Submitted in the proceedings of DD29*, 2025.

[20] J. P. Lucero Lorca, N. Beams, D. Beecroft, and A. Gillman. An iterative solver for the HPS discretization applied to three dimensional Helmholtz problems. *SIAM J. Sci. Comp.*, 46(1):A80–A104, 2024.

[21] P.-G. Martinsson. A direct solver for variable coefficient elliptic PDEs discretized via a composite spectral collocation method. *J. of Comp. Phy.*, 242:460–479, 2013.

[22] P.-G. Martinsson. The hierarchical Poincaré–Steklov (HPS) solver for elliptic PDEs: A tutorial. *arXiv:1506.01308*, 2015.

[23] P.-G. Martinsson and V. Rokhlin. A fast direct solver for boundary integral equations in two dimensions. *J. of Comp. Phy.*, 205(1):1–23, 2005.

[24] D. Melia, D. Fortunato, A. Gillman, and M. O'Neil. Hardware acceleration for hierarchical Poincaré–Steklov algorithms in two and three dimensions. *arXiv:2503.17535*, 2025.

[25] P. G. Schmitz and L. Ying. A fast direct solver for elliptic problems on general meshes in 2D. *J. of Comp. Phy.*, 231(4):1314–1338, 2012.

[26] J. Xia, S. Chandrasekaran, Y. Gu, and X. S. Li. Superfast multifrontal method for large structured linear systems of equations. *SIAM J. on Matrix Anal. Appl.*, 31(3): 1382–1411, 2009.