BDDC Algorithms with Adaptive Choices of Primal Constraints

Olof B. Widlund
Courant Institute, New York University
and others to be named

DD23, July 6, 2015

Problems considered

- BDDC domain decomposition algorithms for finite element approximations for a variety of elliptic problems with very many degrees of freedom.

Problems considered

- BDDC domain decomposition algorithms for finite element approximations for a variety of elliptic problems with very many degrees of freedom.
- Mostly for low order finite element methods for self-adjoint elliptic problems, but also for solvers for isogeometric analysis.

Problems considered

- BDDC domain decomposition algorithms for finite element approximations for a variety of elliptic problems with very many degrees of freedom.
- Mostly for low order finite element methods for self-adjoint elliptic problems, but also for solvers for isogeometric analysis.
- All this work aims at developing preconditioners for the stiffness matrices. These approximate inverses are then combined with conjugate gradients or other Krylov space methods. Aim of our work: Decrease condition numbers.

Problems considered

- BDDC domain decomposition algorithms for finite element approximations for a variety of elliptic problems with very many degrees of freedom.
- Mostly for low order finite element methods for self-adjoint elliptic problems, but also for solvers for isogeometric analysis.
- All this work aims at developing preconditioners for the stiffness matrices. These approximate inverses are then combined with conjugate gradients or other Krylov space methods. Aim of our work: Decrease condition numbers.
- In recent years, considerable efforts to develop adaptive methods to select the primal constraints for BDDC algorithms; they provide the necessary coarse global component. My own efforts much inspired by a talk by Dohrmann at DD22 and his joint work with Clemens Pechstein.

Problems considered

- BDDC domain decomposition algorithms for finite element approximations for a variety of elliptic problems with very many degrees of freedom.
- Mostly for low order finite element methods for self-adjoint elliptic problems, but also for solvers for isogeometric analysis.
- All this work aims at developing preconditioners for the stiffness matrices. These approximate inverses are then combined with conjugate gradients or other Krylov space methods. Aim of our work: Decrease condition numbers.
- In recent years, considerable efforts to develop adaptive methods to select the primal constraints for BDDC algorithms; they provide the necessary coarse global component. My own efforts much inspired by a talk by Dohrmann at DD22 and his joint work with Clemens Pechstein.
- Why BDDC? Great performance record, especially for its deluxe version. No extension theorems required.

BDDC, finite element meshes, and equivalence classes

- BDDC algorithms work on decompositions of the domain Ω of the elliptic problem into non-overlapping subdomains Ω_i, each often with many tens of thousands of degrees of freedom. In between the subdomains the interface Γ. The local interface of Ω_i: $\Gamma_i := \partial\Omega_i \setminus \partial\Omega$. Γ does not cut any elements.

BDDC, finite element meshes, and equivalence classes

- BDDC algorithms work on decompositions of the domain Ω of the elliptic problem into non-overlapping subdomains Ω_i, each often with many tens of thousands of degrees of freedom. In between the subdomains the interface Γ. The local interface of Ω_i: $\Gamma_i := \partial\Omega_i \setminus \partial\Omega$. Γ does not cut any elements.
- Most of the finite element nodes (element edges or faces) are interior to individual subdomains while others belong to several subdomain interfaces. (We might have degrees of freedom on $\partial\Omega$ as well.)

BDDC, finite element meshes, and equivalence classes

- BDDC algorithms work on decompositions of the domain Ω of the elliptic problem into non-overlapping subdomains Ω_i, each often with many tens of thousands of degrees of freedom. In between the subdomains the interface Γ. The local interface of Ω_i: $\Gamma_i := \partial\Omega_i \setminus \partial\Omega$. Γ does not cut any elements.
- Most of the finite element nodes (element edges or faces) are interior to individual subdomains while others belong to several subdomain interfaces. (We might have degrees of freedom on $\partial\Omega$ as well.)
- The degrees of freedom on Γ are partitioned into equivalence classes of sets of indices of the local interfaces Γ_i to which they belong. For 3D and nodal finite elements, we have classes of face nodes, associated with two local interfaces, and classes of edge nodes and subdomain vertex nodes.

BDDC, finite element meshes, and equivalence classes

- BDDC algorithms work on decompositions of the domain Ω of the elliptic problem into non-overlapping subdomains Ω_i, each often with many tens of thousands of degrees of freedom. In between the subdomains the interface Γ. The local interface of Ω_i: $\Gamma_i := \partial\Omega_i \setminus \partial\Omega$. Γ does not cut any elements.
- Most of the finite element nodes (element edges or faces) are interior to individual subdomains while others belong to several subdomain interfaces. (We might have degrees of freedom on $\partial\Omega$ as well.)
- The degrees of freedom on Γ are partitioned into equivalence classes of sets of indices of the local interfaces Γ_i to which they belong. For 3D and nodal finite elements, we have classes of face nodes, associated with two local interfaces, and classes of edge nodes and subdomain vertex nodes.
- For $H(\mathbf{curl})$ and Nédélec (edge) elements, element edges on subdomain faces and edges. For $H(\text{div})$ and Raviart-Thomas elements, degrees of freedom for element faces only.

Partial assembly

- These equivalence classes play a central role in the design, analysis, and programming of domain decomposition methods.

Partial assembly

- These equivalence classes play a central role in the design, analysis, and programming of domain decomposition methods.
- The BDDC (Balancing Domain Decomposition by Constraints) algorithms introduced by Dohrmann in 2003, following the introduction of the FETI–DP algorithms by Farhat et al in 2000. These two families are related algorithmically and have a common theoretical foundation.

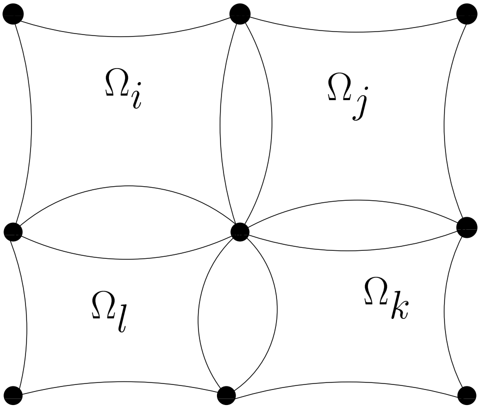
Partial assembly

- These equivalence classes play a central role in the design, analysis, and programming of domain decomposition methods.
- The BDDC (Balancing Domain Decomposition by Constraints) algorithms introduced by Dohrmann in 2003, following the introduction of the FETI–DP algorithms by Farhat et al in 2000. These two families are related algorithmically and have a common theoretical foundation.
- These preconditioners are based on using partially subassembled stiffness matrices assembled from the subdomain stiffness matrices $A^{(i)}$. We will first look at a nodal finite element problem in 2D.

Partial assembly

- These equivalence classes play a central role in the design, analysis, and programming of domain decomposition methods.
- The BDDC (Balancing Domain Decomposition by Constraints) algorithms introduced by Dohrmann in 2003, following the introduction of the FETI–DP algorithms by Farhat et al in 2000. These two families are related algorithmically and have a common theoretical foundation.
- These preconditioners are based on using partially subassembled stiffness matrices assembled from the subdomain stiffness matrices $A^{(i)}$. We will first look at a nodal finite element problem in 2D.
- The nodes of $\Omega_i \cup \Gamma_i$ are divided into those in the interior (I) and those on the interface (Γ). The interface set is further divided into a primal set (Π) and a dual set (Δ).

Torn 2D scalar elliptic problem



- The partially subassembled stiffness matrix of this alternative finite element model is used to define preconditioners; the resulting linear system is much cheaper to solve than the fully assembled system. The primal variables provide a global component of these preconditioners. Also makes all the matrices encountered invertible.

More on BDDC

- The partially subassembled stiffness matrix of this alternative finite element model is used to define preconditioners; the resulting linear system is much cheaper to solve than the fully assembled system. The primal variables provide a global component of these preconditioners. Also makes all the matrices encountered invertible.
- Much of the work involves using Cholesky's algorithm for finite element problems on individual subdomains each on an individual processor of a parallel or distributed computing system. The structure of the algorithm is quite simple and has a modular structure, which allows us to upgrade the performance if a faster Cholesky solver becomes available.

More on BDDC

- The partially subassembled stiffness matrix of this alternative finite element model is used to define preconditioners; the resulting linear system is much cheaper to solve than the fully assembled system. The primal variables provide a global component of these preconditioners. Also makes all the matrices encountered invertible.
- Much of the work involves using Cholesky's algorithm for finite element problems on individual subdomains each on an individual processor of a parallel or distributed computing system. The structure of the algorithm is quite simple and has a modular structure, which allows us to upgrade the performance if a faster Cholesky solver becomes available.
- In a BDDC algorithm, continuity is restored in each step by computing a weighted average across the interface. This leads to non-zero residuals at nodes next to Γ. In each iteration a subdomain Dirichlet solve is used to eliminate them.

Alternative sets of primal constraints

- For scalar 2D, second order elliptic equations and good coefficients, approach outlined yields condition number estimates of $C(1+\log(H/h))^2$. Results can be made independent of jumps in the coefficients, if the interface average chosen carefully. Edge lemma is central to this theory.

Alternative sets of primal constraints

- For scalar 2D, second order elliptic equations and good coefficients, approach outlined yields condition number estimates of $C(1 + \log(H/h))^2$. Results can be made independent of jumps in the coefficients, if the interface average chosen carefully. Edge lemma is central to this theory.
- Good numerical results in 2D but for competitive algorithms in 3D, certain average values (and moments) of the displacement over individual edges (and faces) should also take common values across interface Γ. Same matrix structure as before after a change of variables.

Alternative sets of primal constraints

- For scalar 2D, second order elliptic equations and good coefficients, approach outlined yields condition number estimates of $C(1 + \log(H/h))^2$. Results can be made independent of jumps in the coefficients, if the interface average chosen carefully. Edge lemma is central to this theory.
- Good numerical results in 2D but for competitive algorithms in 3D, certain average values (and moments) of the displacement over individual edges (and faces) should also take common values across interface Γ. Same matrix structure as before after a change of variables.
- Reliable recipes exist for selecting small sets of primal constraints for elasticity in 3D, which primarily use edge averages and first order moments as primal constraints. High quality PETSc-based codes have been developed and successfully tested on very large systems. Public domain software in PETSc, contributed by Stefano Zampini; his codes allow for more than two levels.

- The BDDC and FETI–DP algorithms can be described in terms of three product spaces of finite element functions/vectors defined by their interface nodal values:

$$\widehat{W}_\Gamma \subset \widetilde{W}_\Gamma \subset W_\Gamma.$$

W_Γ: no constraints; $\widehat{W}_\Gamma$: continuity at every point on Γ; $\widetilde{W}_\Gamma$: common values of the primal variables.

Schur complements

- The BDDC and FETI–DP algorithms can be described in terms of three product spaces of finite element functions/vectors defined by their interface nodal values:

$$\widehat{W}_\Gamma \subset \widetilde{W}_\Gamma \subset W_\Gamma.$$

  W_Γ: no constraints; $\widehat{W}_\Gamma$: continuity at every point on Γ; $\widetilde{W}_\Gamma$: common values of the primal variables.
- Change variables, explicitly introducing primal variables and complementary sets of dual displacement variables. Simplifies presentation and also makes methods more robust.

Schur complements

- The BDDC and FETI–DP algorithms can be described in terms of three product spaces of finite element functions/vectors defined by their interface nodal values:

$$\widehat{W}_\Gamma \subset \widetilde{W}_\Gamma \subset W_\Gamma .$$

  W_Γ: no constraints; $\widehat{W}_\Gamma$: continuity at every point on Γ; $\widetilde{W}_\Gamma$: common values of the primal variables.
- Change variables, explicitly introducing primal variables and complementary sets of dual displacement variables. Simplifies presentation and also makes methods more robust.
- After eliminating the interior variables, write the subdomain Schur complements as

$$S^{(i)} = \begin{pmatrix} S^{(i)}_{\Delta\Delta} & S^{(i)}_{\Delta\Pi} \\ S^{(i)}_{\Pi\Delta} & S^{(i)}_{\Pi\Pi} \end{pmatrix} .$$

Schur complements

- The BDDC and FETI–DP algorithms can be described in terms of three product spaces of finite element functions/vectors defined by their interface nodal values:

$$\widehat{W}_\Gamma \subset \widetilde{W}_\Gamma \subset W_\Gamma.$$

  W_Γ: no constraints; $\widehat{W}_\Gamma$: continuity at every point on Γ; $\widetilde{W}_\Gamma$: common values of the primal variables.
- Change variables, explicitly introducing primal variables and complementary sets of dual displacement variables. Simplifies presentation and also makes methods more robust.
- After eliminating the interior variables, write the subdomain Schur complements as

$$S^{(i)} = \begin{pmatrix} S^{(i)}_{\Delta\Delta} & S^{(i)}_{\Delta\Pi} \\ S^{(i)}_{\Pi\Delta} & S^{(i)}_{\Pi\Pi} \end{pmatrix}.$$

- Partially subassemble the $S^{(i)}$, obtaining $\tilde{S}$.

More details on BDDC

- Work with $\widetilde{W}_\Gamma$ and a set of primal constraints. At the end of each iterative step, the approximate solution will be made continuous at all nodal points of the interface; continuity is restored by applying a weighted average operator E_D, which maps $\widetilde{W}_\Gamma$ into $\widehat{W}_\Gamma$.

More details on BDDC

- Work with $\widetilde{W}_\Gamma$ and a set of primal constraints. At the end of each iterative step, the approximate solution will be made continuous at all nodal points of the interface; continuity is restored by applying a weighted average operator E_D, which maps $\widetilde{W}_\Gamma$ into $\widehat{W}_\Gamma$.
- In each iteration, first compute the residual of the fully assembled Schur complement. Then apply E_D^T to obtain right-hand side of the partially subassembled linear system. Solve this system and then apply E_D.

More details on BDDC

- Work with $\widetilde{W}_\Gamma$ and a set of primal constraints. At the end of each iterative step, the approximate solution will be made continuous at all nodal points of the interface; continuity is restored by applying a weighted average operator E_D, which maps $\widetilde{W}_\Gamma$ into $\widehat{W}_\Gamma$.
- In each iteration, first compute the residual of the fully assembled Schur complement. Then apply E_D^T to obtain right-hand side of the partially subassembled linear system. Solve this system and then apply E_D.
- This last step changes the values on Γ, unless the iteration has converged, and results in non-zero residuals at nodes next to Γ.

More details on BDDC

- Work with $\widetilde{W}_\Gamma$ and a set of primal constraints. At the end of each iterative step, the approximate solution will be made continuous at all nodal points of the interface; continuity is restored by applying a weighted average operator E_D, which maps $\widetilde{W}_\Gamma$ into $\widehat{W}_\Gamma$.
- In each iteration, first compute the residual of the fully assembled Schur complement. Then apply E_D^T to obtain right-hand side of the partially subassembled linear system. Solve this system and then apply E_D.
- This last step changes the values on Γ, unless the iteration has converged, and results in non-zero residuals at nodes next to Γ.
- In final step of iteration step, eliminate these residuals by solving a Dirichlet problem on each of the subdomains. Accelerate with preconditioned conjugate gradients.

More details on BDDC

- Work with $\widetilde{W}_\Gamma$ and a set of primal constraints. At the end of each iterative step, the approximate solution will be made continuous at all nodal points of the interface; continuity is restored by applying a weighted average operator E_D, which maps $\widetilde{W}_\Gamma$ into $\widehat{W}_\Gamma$.
- In each iteration, first compute the residual of the fully assembled Schur complement. Then apply E_D^T to obtain right-hand side of the partially subassembled linear system. Solve this system and then apply E_D.
- This last step changes the values on Γ, unless the iteration has converged, and results in non-zero residuals at nodes next to Γ.
- In final step of iteration step, eliminate these residuals by solving a Dirichlet problem on each of the subdomains. Accelerate with preconditioned conjugate gradients.
- The condition number of a BDDC algorithm bounded by $\|E_D\|_{\tilde{S}}$.

BDDC deluxe

- When designing a BDDC algorithm, we have to choose an effective set of primal constraints and also a recipe for the averaging across interface.

BDDC deluxe

- When designing a BDDC algorithm, we have to choose an effective set of primal constraints and also a recipe for the averaging across interface.
- Traditional averaging recipes found not to work uniformly well for 3D problems in $H(\mathbf{curl})$: With Dohrmann in DD20 paper and in CPAM; appeared electronically last month.

BDDC deluxe

- When designing a BDDC algorithm, we have to choose an effective set of primal constraints and also a recipe for the averaging across interface.
- Traditional averaging recipes found not to work uniformly well for 3D problems in $H(\mathbf{curl})$: With Dohrmann in DD20 paper and in CPAM; appeared electronically last month.
- Alternative found, also very robust for 3D $H(\text{div})$ problems: Duk-Soon Oh, OBW, and Clark Dohrmann; CIMS TR2013-951.

BDDC deluxe

- When designing a BDDC algorithm, we have to choose an effective set of primal constraints and also a recipe for the averaging across interface.
- Traditional averaging recipes found not to work uniformly well for 3D problems in $H(\mathbf{curl})$: With Dohrmann in DD20 paper and in CPAM; appeared electronically last month.
- Alternative found, also very robust for 3D $H(\mathrm{div})$ problems: Duk-Soon Oh, OBW, and Clark Dohrmann; CIMS TR2013-951.
- Both the $H(\mathbf{curl})$ and $H(\mathrm{div})$ problems have two material parameters; complicates the design of the average operator.

BDDC deluxe

- When designing a BDDC algorithm, we have to choose an effective set of primal constraints and also a recipe for the averaging across interface.
- Traditional averaging recipes found not to work uniformly well for 3D problems in $H(\mathbf{curl})$: With Dohrmann in DD20 paper and in CPAM; appeared electronically last month.
- Alternative found, also very robust for 3D $H(\text{div})$ problems: Duk-Soon Oh, OBW, and Clark Dohrmann; CIMS TR2013-951.
- Both the $H(\mathbf{curl})$ and $H(\text{div})$ problems have two material parameters; complicates the design of the average operator.
- A paper on isogeometric elements, joint with Beirão da Veiga, Pavarino, Scacchi, and Zampini in SIAM Sci. Comput. in 2014.

BDDC deluxe

- When designing a BDDC algorithm, we have to choose an effective set of primal constraints and also a recipe for the averaging across interface.
- Traditional averaging recipes found not to work uniformly well for 3D problems in $H(\mathbf{curl})$: With Dohrmann in DD20 paper and in CPAM; appeared electronically last month.
- Alternative found, also very robust for 3D $H(\text{div})$ problems: Duk-Soon Oh, OBW, and Clark Dohrmann; CIMS TR2013-951.
- Both the $H(\mathbf{curl})$ and $H(\text{div})$ problems have two material parameters; complicates the design of the average operator.
- A paper on isogeometric elements, joint with Beirão da Veiga, Pavarino, Scacchi, and Zampini in SIAM Sci. Comput. in 2014.
- My former student Jong Ho Lee has published a paper on Reissner-Mindlin plates: SINUM 53(1), 2014.

BDDC deluxe

- When designing a BDDC algorithm, we have to choose an effective set of primal constraints and also a recipe for the averaging across interface.
- Traditional averaging recipes found not to work uniformly well for 3D problems in $H(\mathbf{curl})$: With Dohrmann in DD20 paper and in CPAM; appeared electronically last month.
- Alternative found, also very robust for 3D $H(\mathrm{div})$ problems: Duk-Soon Oh, OBW, and Clark Dohrmann; CIMS TR2013-951.
- Both the $H(\mathbf{curl})$ and $H(\mathrm{div})$ problems have two material parameters; complicates the design of the average operator.
- A paper on isogeometric elements, joint with Beirão da Veiga, Pavarino, Scacchi, and Zampini in SIAM Sci. Comput. in 2014.
- My former student Jong Ho Lee has published a paper on Reissner-Mindlin plates: SINUM 53(1), 2014.
- Work on DG by Dryja, Galvis and Sarkis and Chung and Kim.

Deluxe scaling

- The average operator E_D across a face $F \subset \Gamma$, common to two subdomains Ω_i and Ω_j, defined in terms of principal minors $S_F^{(k)}$ of the $S^{(k)}, k = i, j$.

Deluxe scaling

- The average operator E_D across a face $F \subset \Gamma$, common to two subdomains Ω_i and Ω_j, defined in terms of principal minors $S_F^{(k)}$ of the $S^{(k)}, k = i, j$.
- The deluxe averaging operator, for F, is then defined by

$$\bar{w}_F := (E_D w)_F := (S_F^{(i)} + S_F^{(j)})^{-1}(S_F^{(i)} w_F^{(i)} + S_F^{(j)} w_F^{(j)}).$$

Deluxe scaling

- The average operator E_D across a face $F \subset \Gamma$, common to two subdomains Ω_i and Ω_j, defined in terms of principal minors $S_F^{(k)}$ of the $S^{(k)}, k = i, j$.
- The deluxe averaging operator, for F, is then defined by

$$\bar{w}_F := (E_D w)_F := (S_F^{(i)} + S_F^{(j)})^{-1}(S_F^{(i)} w_F^{(i)} + S_F^{(j)} w_F^{(j)}).$$

- The action of $(S_F^{(i)} + S_F^{(j)})^{-1}$ can be implemented by solving a Dirichlet problem on $\Omega_i \cup F \cup \Omega_j$. Here, F interface between the two subdomains. This can add significantly to the cost.

Deluxe scaling

- The average operator E_D across a face $F \subset \Gamma$, common to two subdomains Ω_i and Ω_j, defined in terms of principal minors $S_F^{(k)}$ of the $S^{(k)}, k = i, j$.
- The deluxe averaging operator, for F, is then defined by

$$\bar{w}_F := (E_D w)_F := (S_F^{(i)} + S_F^{(j)})^{-1}(S_F^{(i)} w_F^{(i)} + S_F^{(j)} w_F^{(j)}).$$

- The action of $(S_F^{(i)} + S_F^{(j)})^{-1}$ can be implemented by solving a Dirichlet problem on $\Omega_i \cup F \cup \Omega_j$. Here, F interface between the two subdomains. This can add significantly to the cost.
- Just using skinny domains built from one or two layers of elements next to the face results in very similar performance. Not a luxury any more. Not yet fully understood.

BDDC deluxe

- Similar formulas for subdomain edges and other equivalence classes of interface variables. The operator E_D is assembled from these components.

BDDC deluxe

- Similar formulas for subdomain edges and other equivalence classes of interface variables. The operator E_D is assembled from these components.
- The core of any estimate for a BDDC algorithm is in terms of the norm of the average operator E_D. By an algebraic argument known, for FETI–DP, since 2002,

$$\kappa(M^{-1}A) \leq \|E_D\|_{\tilde{S}}.$$

BDDC deluxe

- Similar formulas for subdomain edges and other equivalence classes of interface variables. The operator E_D is assembled from these components.
- The core of any estimate for a BDDC algorithm is in terms of the norm of the average operator E_D. By an algebraic argument known, for FETI–DP, since 2002,

$$\kappa(M^{-1}A) \leq \|E_D\|_{\tilde{S}}.$$

- We can show that the analysis of BDDC deluxe essentially can be reduced to bounds for individual subdomains.

BDDC deluxe

- Similar formulas for subdomain edges and other equivalence classes of interface variables. The operator E_D is assembled from these components.
- The core of any estimate for a BDDC algorithm is in terms of the norm of the average operator E_D. By an algebraic argument known, for FETI–DP, since 2002,

$$\kappa(M^{-1}A) \leq \|E_D\|_{\tilde{S}}.$$

- We can show that the analysis of BDDC deluxe essentially can be reduced to bounds for individual subdomains.
- Arbitrary jumps in two coefficients can often be accommodated.

BDDC deluxe

- Similar formulas for subdomain edges and other equivalence classes of interface variables. The operator E_D is assembled from these components.
- The core of any estimate for a BDDC algorithm is in terms of the norm of the average operator E_D. By an algebraic argument known, for FETI–DP, since 2002,

$$\kappa(M^{-1}A) \leq \|E_D\|_{\tilde{S}}.$$

- We can show that the analysis of BDDC deluxe essentially can be reduced to bounds for individual subdomains.
- Arbitrary jumps in two coefficients can often be accommodated.
- Analysis of traditional BDDC requires the use of an extension theorem; the deluxe version does not.

BDDC deluxe algebra

- Develop estimate for $P_D := I - E_D$; instead of estimating $(R_F^T \bar{w}_F)^T S^{(i)} R_F^T \bar{w}_F$, estimate the $S^{(i)}-$norm of $R_F^T(w_F^{(i)} - \bar{w}_F)$. Here R_F is the restriction to the face F. By simple algebra, we find that

$$w_F^{(i)} - \bar{w}_F = (S_F^{(i)} + S_F^{(j)})^{-1} S_F^{(j)} (w_F^{(i)} - w_F^{(j)}).$$

Here $S_F^{(i)} := R_F S^{(i)} R_F^T$.

BDDC deluxe algebra

- Develop estimate for $P_D := I - E_D$; instead of estimating $(R_F^T \bar{w}_F)^T S^{(i)} R_F^T \bar{w}_F$, estimate the $S^{(i)}-$norm of $R_F^T(w_F^{(i)} - \bar{w}_F)$. Here R_F is the restriction to the face F. By simple algebra, we find that

$$w_F^{(i)} - \bar{w}_F = (S_F^{(i)} + S_F^{(j)})^{-1} S_F^{(j)} (w_F^{(i)} - w_F^{(j)}).$$

  Here $S_F^{(i)} := R_F S^{(i)} R_F^T$.

- More algebra gives:

$$(R_F^T(w_F^{(i)} - \bar{w}_F))^T S^{(i)} (R_F^T(w_F^{(i)} - \bar{w}_F)) =$$

$$(w_F^{(i)} - w_F^{(j)})^T S_F^{(j)} (S_F^{(i)} + S_F^{(j)})^{-1} S_F^{(i)} (S_F^{(i)} + S_F^{(j)})^{-1} S_F^{(j)} (w_F^{(i)} - w_F^{(j)}).$$

Parallel sums

- Add contribution from Ω_j. Following Clemens Pechstein, we find that the relevant expression of the energy is

$$(w_F^{(i)} - w_F^{(j)})^T (S_F^{(i)^{-1}} + S_F^{(j)^{-1}})^{-1} (w_F^{(i)} - w_F^{(j)}).$$

Parallel sums

- Add contribution from Ω_j. Following Clemens Pechstein, we find that the relevant expression of the energy is

$$(w_F^{(i)} - w_F^{(j)})^T (S_F^{(i)^{-1}} + S_F^{(j)^{-1}})^{-1} (w_F^{(i)} - w_F^{(j)}).$$

- We will use the notation,

$$A : B := (A^{-1} + B^{-1})^{-1},$$

and similarly

$$A : B : C := (A^{-1} + B^{-1} + C^{-1})^{-1}, \quad etc.,$$

for *parallel sums* of symmetric matrices, which are at least positive semi-definite.

Parallel sums

- Add contribution from Ω_j. Following Clemens Pechstein, we find that the relevant expression of the energy is

$$(w_F^{(i)} - w_F^{(j)})^T (S_F^{(i)^{-1}} + S_F^{(j)^{-1}})^{-1} (w_F^{(i)} - w_F^{(j)}).$$

- We will use the notation,

$$A : B := (A^{-1} + B^{-1})^{-1},$$

  and similarly

$$A : B : C := (A^{-1} + B^{-1} + C^{-1})^{-1}, \quad etc.,$$

  for *parallel sums* of symmetric matrices, which are at least positive semi-definite.

- Trivially $A : B \leq A$ and $A : B \leq B$.

- It then easily follows that,

$$(w_F^{(i)} - w_F^{(j)})^T (S_F^{(i)} : S_F^{(j)})(w_F^{(i)} - w_F^{(j)})$$

$$\leq 2(w_F^{(i)} - w_\Pi)^T S_F^{(i)} (w_F^{(i)} - w_\Pi) + 2(w_F^{(j)} - w_\Pi)^T S_F^{(j)} (w_F^{(j)} - w_\Pi),$$

where $w_{F\Delta}^{(k)} = w_F^{(k)} - w_\Pi$ and w_Π is an arbitrary element of the primal space.

Continued

- It then easily follows that,

$$(w_F^{(i)} - w_F^{(j)})^T (S_F^{(i)} : S_F^{(j)})(w_F^{(i)} - w_F^{(j)})$$

$$\leq 2(w_F^{(i)} - w_\Pi)^T S_F^{(i)} (w_F^{(i)} - w_\Pi) + 2(w_F^{(j)} - w_\Pi)^T S_F^{(j)} (w_F^{(j)} - w_\Pi),$$

where $w_{F\Delta}^{(k)} = w_F^{(k)} - w_\Pi$ and w_Π is an arbitrary element of the primal space.

- Each of the terms local to only one subdomain.

Continued

- It then easily follows that,

$$(w_F^{(i)} - w_F^{(j)})^T (S_F^{(i)} : S_F^{(j)})(w_F^{(i)} - w_F^{(j)})$$

$$\leq 2(w_F^{(i)} - w_\Pi)^T S_F^{(i)}(w_F^{(i)} - w_\Pi) + 2(w_F^{(j)} - w_\Pi)^T S_F^{(j)}(w_F^{(j)} - w_\Pi),$$

where $w_{F\Delta}^{(k)} = w_F^{(k)} - w_\Pi$ and w_Π is an arbitrary element of the primal space.

- Each of the terms local to only one subdomain.
- Now remains to estimate $w_{F\Delta}^{(i)T} S_F^{(i)} w_{F\Delta}^{(i)}$ by $w_{F\Delta}^{(i)T} \tilde{S}_F^{(i)} w_{F\Delta}^{(i)}$, where the latter represents the minimum norm extension.

Continued

- It then easily follows that,

$$(w_F^{(i)} - w_F^{(j)})^T (S_F^{(i)} : S_F^{(j)})(w_F^{(i)} - w_F^{(j)})$$

$$\leq 2(w_F^{(i)} - w_\Pi)^T S_F^{(i)} (w_F^{(i)} - w_\Pi) + 2(w_F^{(j)} - w_\Pi)^T S_F^{(j)} (w_F^{(j)} - w_\Pi),$$

where $w_{F\Delta}^{(k)} = w_F^{(k)} - w_\Pi$ and w_Π is an arbitrary element of the primal space.

- Each of the terms local to only one subdomain.
- Now remains to estimate $w_{F\Delta}^{(i)T} S_F^{(i)} w_{F\Delta}^{(i)}$ by $w_{F\Delta}^{(i)T} \tilde{S}_F^{(i)} w_{F\Delta}^{(i)}$, where the latter represents the minimum norm extension.
- This can be done by using a *face lemma* in 3D, or an *edge lemma* in 2D if we have nice coefficients in each subdomain and the subdomains are polytopes.

Eigenvalues of $S_E^{(i)-1}(S_E^{(i)} - \tilde{S}_E^{(i)})$ for 2D problems

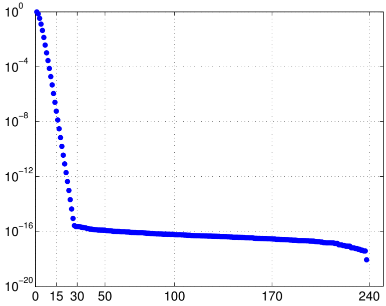


Figure : $H/h = 240$, $\rho = 1$, and irregular subdomains (METIS).

Eigenvalues of $S_E^{(i)-1}(S_E^{(i)} - \tilde{S}_E^{(i)})$ for 2D problems

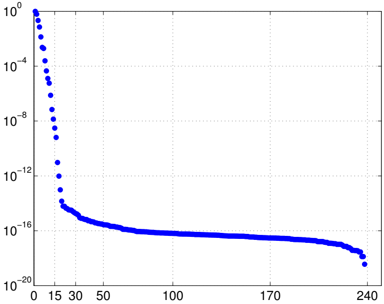


Figure : $H/h = 240$, random coefficients and irregular subdomains (METIS).

Adaptive choices of primal space

- Consider a problem in 2D. We can then generate elements for the primal space for an edge by solving a generalized eigenvalue problem

$$\tilde{S}_F^{(i)} : \tilde{S}_F^{(j)} \phi = \lambda S_F^{(i)} : S_F^{(j)} \phi.$$

Adaptive choices of primal space

- Consider a problem in 2D. We can then generate elements for the primal space for an edge by solving a generalized eigenvalue problem

$$\tilde{S}_F^{(i)} : \tilde{S}_F^{(j)} \phi = \lambda S_F^{(i)} : S_F^{(j)} \phi.$$

- Primal constraints are generate by eigenvectors corresponding to the smallest eigenvalues.

Adaptive choices of primal space

- Consider a problem in 2D. We can then generate elements for the primal space for an edge by solving a generalized eigenvalue problem

$$\tilde{S}_F^{(i)} : \tilde{S}_F^{(j)} \phi = \lambda S_F^{(i)} : S_F^{(j)} \phi.$$

- Primal constraints are generate by eigenvectors corresponding to the smallest eigenvalues.
- We find that the eigenvalues converge to 1 quite rapidly even for problems with large changes in the coefficients inside subdomains. Primal space does not grow a great deal and the iteration count can decline considerably.

An edge common to three subdomains

The discussion that follows can be extended straightforwardly to equivalence classes with more than three elements.

- We need an expression for the energy related to $I - E_D$ and a good generalized eigenvalue problem to select primal constraints.

An edge common to three subdomains

The discussion that follows can be extended straightforwardly to equivalence classes with more than three elements.

- We need an expression for the energy related to $I - E_D$ and a good generalized eigenvalue problem to select primal constraints.
- The relevant energy can be written in terms of $w_E^{(i)} - w_E^{(j)}$, etc., and operators of the form

$$T_E^{(i)} := S_E^{(i)} : (S_E^{(j)} + S_E^{(k)}),$$

etc.

An edge common to three subdomains

The discussion that follows can be extended straightforwardly to equivalence classes with more than three elements.

- We need an expression for the energy related to $I - E_D$ and a good generalized eigenvalue problem to select primal constraints.
- The relevant energy can be written in terms of $w_E^{(i)} - w_E^{(j)}$, etc., and operators of the form

  $$T_E^{(i)} := S_E^{(i)} : (S_E^{(j)} + S_E^{(k)}),$$

  etc.
- Can we estimate $T_E^{(i)}$ by $S_E^{(i)} : S_E^{(j)} : S_E^{(k)}$? If so, we could then choose a generalized eigenvalue problem with the matrices $S_E^{(i)} : S_E^{(j)} : S_E^{(k)}$ and $\tilde{S}_E^{(i)} : \tilde{S}_E^{(j)} : \tilde{S}_E^{(k)}$. But such an estimate does not hold without additional assumptions.

Recipes

Several generalized eigenvalue problems have been quite successful but some lack full theoretical justification.

- Simone Scacchi has used what would correspond to the matrices $S_E^{(i)} : S_E^{(j)} : S_E^{(k)}$ and $\tilde{S}_E^{(i)} + \tilde{S}_E^{(j)} + \tilde{S}_E^{(k)}$ for difficult, very ill-conditioned problems arising in IGA problems.

Recipes

Several generalized eigenvalue problems have been quite successful but some lack full theoretical justification.

- Simone Scacchi has used what would correspond to the matrices $S_E^{(i)} : S_E^{(j)} : S_E^{(k)}$ and $\tilde{S}_E^{(i)} + \tilde{S}_E^{(j)} + \tilde{S}_E^{(k)}$ for difficult, very ill-conditioned problems arising in IGA problems.
- Stefano Zampini has used $S_E^{(i)} : S_E^{(j)} : S_E^{(k)}$ and $\tilde{S}_E^{(i)} : \tilde{S}_E^{(j)} : \tilde{S}_E^{(k)}$ successfully for subdomain edges and 3D $H(\mathbf{curl})$ problems. (Also a lot of success with $H(\text{div})-$problems; only face constraints.)

Recipes

Several generalized eigenvalue problems have been quite successful but some lack full theoretical justification.

- Simone Scacchi has used what would correspond to the matrices $S_E^{(i)} : S_E^{(j)} : S_E^{(k)}$ and $\tilde{S}_E^{(i)} + \tilde{S}_E^{(j)} + \tilde{S}_E^{(k)}$ for difficult, very ill-conditioned problems arising in IGA problems.
- Stefano Zampini has used $S_E^{(i)} : S_E^{(j)} : S_E^{(k)}$ and $\tilde{S}_E^{(i)} : \tilde{S}_E^{(j)} : \tilde{S}_E^{(k)}$ successfully for subdomain edges and 3D $H(\mathbf{curl})$ problems. (Also a lot of success with $H(\text{div})-$problems; only face constraints.)
- More of a justification can be given if we choose the matrices $T_E^{(i)} + T_E^{(j)} + T_E^{(k)}$ and $\tilde{S}_E^{(i)} : \tilde{S}_E^{(j)} : \tilde{S}_E^{(k)}$ for the generalized eigenvalue problem to determine good primal constraints for subdomain edges in 3D. But are the spectrum of this generalized eigenvalue good?

Recipes

Several generalized eigenvalue problems have been quite successful but some lack full theoretical justification.

- Simone Scacchi has used what would correspond to the matrices $S_E^{(i)} : S_E^{(j)} : S_E^{(k)}$ and $\tilde{S}_E^{(i)} + \tilde{S}_E^{(j)} + \tilde{S}_E^{(k)}$ for difficult, very ill-conditioned problems arising in IGA problems.
- Stefano Zampini has used $S_E^{(i)} : S_E^{(j)} : S_E^{(k)}$ and $\tilde{S}_E^{(i)} : \tilde{S}_E^{(j)} : \tilde{S}_E^{(k)}$ successfully for subdomain edges and 3D $H(\mathbf{curl})$ problems. (Also a lot of success with $H(\text{div})-$problems; only face constraints.)
- More of a justification can be given if we choose the matrices $T_E^{(i)} + T_E^{(j)} + T_E^{(k)}$ and $\tilde{S}_E^{(i)} : \tilde{S}_E^{(j)} : \tilde{S}_E^{(k)}$ for the generalized eigenvalue problem to determine good primal constraints for subdomain edges in 3D. But are the spectrum of this generalized eigenvalue good?
- This experimental work is joint with Juan G. Calvo.

Numerical experiments: Scalability, $H/h = 8$

Cubic subdomains

ρ	N	Corners $I(\kappa)$	Corners $\lvert W_\Pi \rvert$	Wire $I(\kappa)$	Wire $\lvert W_\Pi \rvert$	Average $I(\kappa)$	Average $\lvert W_\Pi \rvert$	NE
1	3^3	12(14.9)	8	6(1.6)	260	12(13.9)	44	36
	4^3	17(16.6)	27	7(1.7)	783	17(15.6)	135	108
	5^3	24(17.2)	64	7(1.8)	1744	24(16.1)	304	240
	6^3	26(17.6)	125	8(1.8)	3275	25(16.5)	575	450
R	3^3	23(42.9)	8	10(2.5)	260	21(22.9)	44	36
	4^3	34(77.9)	27	12(2.9)	783	25(16.8)	135	108
	5^3	52(83.4)	64	12(2.9)	1744	34(23.1)	304	240
	6^3	68(107)	125	13(3.0)	3275	37(23.5)	575	450

Numerical experiments: Scalability, $H/h = 8$

Cubic subdomains

ρ	N	Adapt. 95%		Adapt. 50%		Adap. 25%		NE
		$I(\kappa)$	$\lvert W_\Pi \rvert$	$I(\kappa)$	$\lvert W_\Pi \rvert$	$I(\kappa)$	$\lvert W_\Pi \rvert$	
1	3^3	9(2.3)	92	9(2.3)	92	7(1.6)	116	36
	4^3	9(2.2)	351	9(2.3)	351	7(1.7)	405	108
	5^3	20(6.7)	564	20(6.7)	566	19(2.0)	665	240
	6^3	19(6.7)	1571	19(6.7)	1574	19(2.1)	1727	450
R	3^3	17(22.9)	92	14(4.5)	98	14(4.5)	113	36
	4^3	23(14.9)	213	22(14.6)	238	22(13.5)	269	108
	5^3	22(11.1)	655	22(11.0)	703	22(10.9)	782	240
	6^3	23(9.8)	1499	22(9.0)	1573	21(7.9)	1679	450

Numerical experiments: Scalability, $H/h = 8$

METIS subdomains

ρ	N	Corners $I(\kappa)$	Corners $\lvert W_\Pi \rvert$	Wire $I(\kappa)$	Wire $\lvert W_\Pi \rvert$	Average $I(\kappa)$	Average $\lvert W_\Pi \rvert$	NE
1	3^3	17(7.0)	51	8(1.6)	532	13(3.6)	154	126
	4^3	20(7.4)	164	8(1.6)	1594	14(4.0)	516	389
	5^3	22(8.2)	417	8(1.7)	3624	18(5.7)	1225	951
R	3^3	21(15.5)	51	10(2.3)	532	18(7.3)	169	126
	4^3	27(14.7)	164	11(2.6)	1594	20(8.5)	516	389
	5^3	34(19.5)	417	12(2.7)	3624	27(11.1)	1265	951

Numerical experiments: Scalability, $H/h = 8$

METIS subdomains

ρ	N	Adapt. 95% $I(\kappa)$	Adapt. 95% $\lvert W_\Pi \rvert$	Adapt. 50% $I(\kappa)$	Adapt. 50% $\lvert W_\Pi \rvert$	Adap. 10% $I(\kappa)$	Adap. 10% $\lvert W_\Pi \rvert$	NE
1	3^3	13(3.7)	161	13(3.6)	166	10(2.2)	258	126
	4^3	14(3.7)	568	14(3.6)	578	10(2.4)	821	389
	5^3	19(5.6)	1236	19(5.5)	1245	16(2.9)	1685	951
R	3^3	18(7.0)	161	18(8.0)	173	15(4.8)	225	126
	4^3	20(7.7)	519	20(7.5)	530	16(5.0)	649	389
	5^3	25(8.8)	1268	25(8.6)	1336	22(5.2)	1568	951

Final remarks

- Here, we have focused on an effort to work with only one generalized eigenvalue problem for equivalence classes with more than two subdomains such as for subdomain edges in 3D.
- We could also use several generalized eigenvalue problems and sequentially increase the primal space; that approach has been explored in a recent paper by Hyea Hyun Kim and Eric Chung.
- A lot of experimental work will be required to settle these issues.