Scalable AMG preconditioners for PDE-constrained optimization problems

A. Borzì[*], V. De Simone[†], D. di Serafino[†§]

[*]University of Würzburg, Germany
[†]Second University of Naples, Italy
[§]Inst. High-Performance Computing and Networking (ICAR), CNR, Naples, Italy

daniela.diserafino@unina2.it

SC2011 - International Conference on Scientific Computing
S. Margherita di Pula, Sardinia, Italy, October 10-14, 2011

Work partially supported by INdAM-GNCS under grant "Progetti 2011"
(Advanced Numerical Methods for Large-Scale Constrained Nonlinear Optimization Problems)

Outline

PDE-constrained optimization and multilevel methods

- Solving PDE-constrained optimization problems is a computationally demanding task, requiring
  - accurate and efficient numerical methods
  - high-performance computing resources
- Multigrid/multilevel methods
  - achievable optimal convergence rates
  - flexibility and wide applicability
  - robustness with respect to optimization parameters
  - possibility of developing scalable implementations

Borzì & Schultz, *Multigrid Methods for PDE Optimization*, SIREV, 51, 2009

PDE-constrained optimization and multilevel methods

- Solving PDE-constrained optimization problems is a computationally demanding task, requiring
  - accurate and efficient numerical methods
  - high-performance computing resources
- Multigrid/multilevel methods
  - achievable optimal convergence rates
  - flexibility and wide applicability
  - robustness with respect to optimization parameters
  - possibility of developing scalable implementations

Borzì & Schultz, *Multigrid Methods for PDE Optimization*, SIREV, 51, 2009

Our goal

developing efficient parallel algebraic multilevel Schwarz preconditioners for use in PDE-constrained optimization

Model Problem

Elliptic distributed control problem

$$\begin{cases} \min\limits_{u \in L^2(\Omega)} J(y,u) \equiv \dfrac{1}{2}||y-z||^2_{L^2(\Omega)} + \dfrac{\nu}{2}||u||^2_{L^2(\Omega)} \\ \text{s.t.} \quad -\Delta y = u + f \quad \text{in } \Omega \\ \qquad\quad y = 0 \qquad\quad \text{on } \partial\Omega \end{cases}$$

$\Omega \subset \Re^2$ convex, $\nu > 0$, and $f, z \in L^2(\Omega)$ given

Model Problem

Elliptic distributed control problem

$$\begin{cases} \min\limits_{u \in L^2(\Omega)} J(y,u) \equiv \dfrac{1}{2}||y-z||^2_{L^2(\Omega)} + \dfrac{\nu}{2}||u||^2_{L^2(\Omega)} \\ \text{s.t.} \quad -\Delta y = u + f \quad \text{in } \Omega \\ \qquad\quad y = 0 \qquad\quad \text{on } \partial\Omega \end{cases}$$

$\Omega \subset \Re^2$ convex, $\nu > 0$, and $f, z \in L^2(\Omega)$ given

Optimality conditions

$$\begin{cases} -\Delta y - u = f & \text{in } \Omega, \quad y = 0 \text{ on } \partial\Omega \\ -\Delta p + y = z & \text{in } \Omega, \quad p = 0 \text{ on } \partial\Omega \\ \nu u - p = 0 & \text{in } \Omega \end{cases}$$

Discrete optimality system

Discretization (e.g.) by second-order central finite differences on a $n \times n$ grid

$$\mathcal{A}\mathbf{v} = \mathbf{w}, \quad \mathcal{A} \in \Re^{N \times N}, \; N = 3n$$

$$\mathcal{A} = \begin{pmatrix} A & 0 & -I \\ I & A & 0 \\ 0 & -I & \nu I \end{pmatrix}, \quad \mathbf{v} = \begin{pmatrix} y \\ p \\ u \end{pmatrix}, \quad \mathbf{w} = \begin{pmatrix} f \\ z \\ 0 \end{pmatrix},$$

$A =$ matrix arising from the discretization of $-\Delta$

Discrete optimality system

Discretization (e.g.) by second-order central finite differences on a $n \times n$ grid

$$\mathcal{A}\mathbf{v} = \mathbf{w}, \quad \mathcal{A} \in \Re^{N \times N}, \; N = 3n$$

$$\mathcal{A} = \begin{pmatrix} A & 0 & -I \\ I & A & 0 \\ 0 & -I & \nu I \end{pmatrix}, \quad \mathbf{v} = \begin{pmatrix} y \\ p \\ u \end{pmatrix}, \quad \mathbf{w} = \begin{pmatrix} f \\ z \\ 0 \end{pmatrix},$$

$A =$ matrix arising from the discretization of $-\Delta$

Multilevel Schwarz preconditioners

- naturally fit in a parallel environment (domain decomposition)
- # iterations of preconditioned solver independent of # subdomains

Algebraic approach

- wide applicability and capability of adapting to a specific problem
- allows software reusability

Algebraic multilevel preconditioners

$$\mathcal{A}\mathbf{v} = \mathbf{w}, \quad \mathcal{A} \in \Re^{N \times N}$$

Multilevel strategy

- two-level: apply a basic preconditioner (smoother) to the given linear system and then improve it by solving a projection of the associated error system in a coarse space (coarse-space, or coarse-level, correction)
- multilevel: recursive application of the previous strategy

Algebraic multilevel preconditioners

$$\mathcal{A}\mathbf{v} = \mathbf{w}, \quad \mathcal{A} \in \Re^{N \times N}$$

Multilevel strategy

- two-level: apply a basic preconditioner (smoother) to the given linear system and then improve it by solving a projection of the associated error system in a coarse space (coarse-space, or coarse-level, correction)
- multilevel: recursive application of the previous strategy

Two phases

- setup: build all the operators needed by the multilevel strategy
- application: apply the multilevel preconditioner through a suitable combination of these operators

Setup phase

$$\mathcal{A}_1 \equiv \mathcal{A} \text{ finest matrix}$$

$$\mathcal{V}_1 \equiv \mathcal{V} = \{1, 2, \ldots, N_1 \equiv N\} \text{ finest index space (row indices of } \mathcal{A})$$

Setup phase

$$\mathcal{A}_1 \equiv \mathcal{A} \text{ finest matrix}$$

$$\mathcal{V}_1 \equiv \mathcal{V} = \{1, 2, \ldots, N_1 \equiv N\} \text{ finest index space (row indices of } \mathcal{A})$$

build $\mathcal{M}_1$

`for` $k = 1, nlev - 1$ `do`

generate $\mathcal{V}_{k+1}$ from $\mathcal{V}_k$	coarsening
build $\mathcal{P}_k$ and $\mathcal{R}_k$	definition of restr. & prolong. operators
compute $\mathcal{A}_{k+1} = \mathcal{R}_k \mathcal{A}_k \mathcal{P}_k$	(Petrov-)Galerkin approach
build $\mathcal{M}_{k+1}$	setup of smoother ($\approx \mathcal{A}_{k+1}^{-1}$)

`endfor`

$$\mathcal{V}_1 \subset \mathcal{V}^2 \ldots \subset \mathcal{V}^{nlev} \text{ hierarchy of index spaces}$$

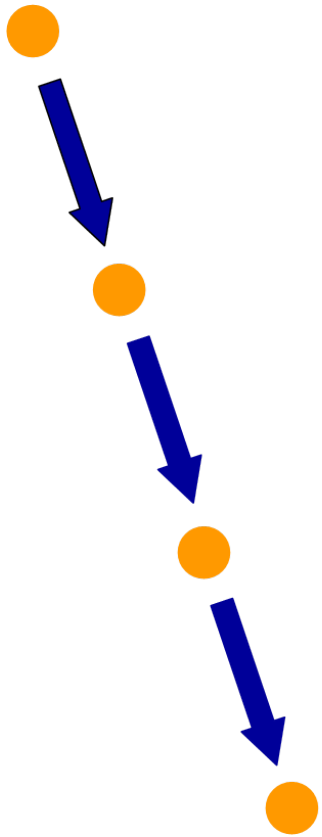
$$\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}^{nlev} \text{ hierarchy of matrices}$$

$$\mathcal{P}_k : \Re^{N_{k+1}} \longrightarrow \Re^{N_k}, \quad \mathcal{R}_k : \Re^{N_k} \longrightarrow \Re^{N_{k+1}}, \quad N_k = |\mathcal{V}_k|$$

$$\text{usually } \mathcal{R}_k = (\mathcal{P}_k)^T$$

Application phase

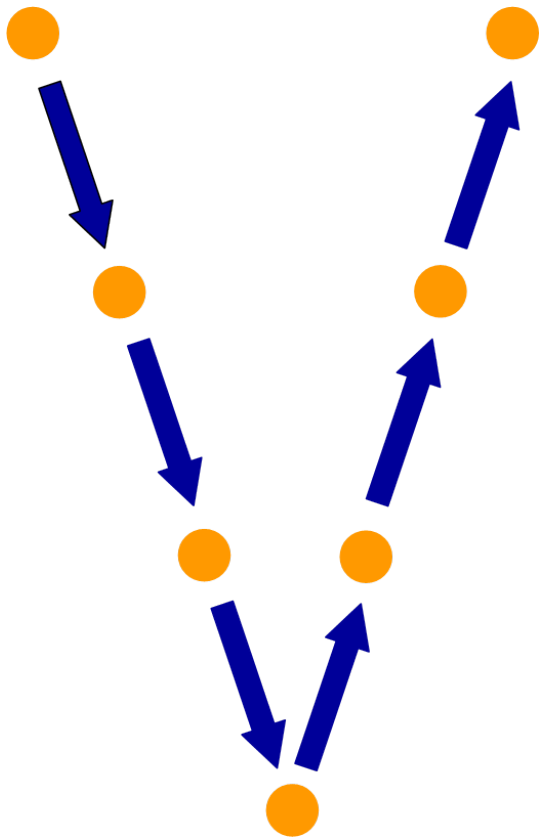
Example: V-cycle



```
procedure Vcycle(k, A_k, w_k)
if(k ≠ nlev) then
   v_k = M_k w_k
   w_{k+1} = R_k(w_k − A_k v_k)
   v_{k+1} = Vcycle(k + 1, A_{k+1}, w_{k+1})
else
   v_k = A_k^{-1} w_k
endif
return v_k
```

Application phase

Example: V-cycle



```
procedure Vcycle(k, A_k, w_k)
if(k ≠ nlev) then
   v_k = M_k w_k
   w_{k+1} = R_k(w_k − A_k v_k)
   v_{k+1} = Vcycle(k + 1, A_{k+1}, w_{k+1})
   v_k = v_k + P_k v_{k+1}
   v_k = v_k + M_k(w_k − A_k v_k)
else
   v_k = A_k^{-1} w_k
endif
return v_k
```

Setup of coarse-level correction

Focus on smoothed aggregation (Vaněk, Mandel, Brezina, Computing, 56, 1996)

Coarsening

- scalar problems: obtain coarse indices by aggregating the indices of $\mathcal{V}$ into N' subsets that form a disjoint coverage of $\mathcal{V}$

Setup of coarse-level correction

Focus on smoothed aggregation (Vaněk, Mandel, Brezina, Computing, 56, 1996)

Coarsening

- scalar problems: obtain coarse indices by aggregating the indices of $\mathcal{V}$ into N' subsets that form a disjoint coverage of $\mathcal{V}$

  each subset $\mathcal{C}_s \equiv \mathcal{C}_s(j)$ contains a "root" index $j \in \mathcal{V}$ and other indices $i \in \mathcal{V}$ that are strongly coupled to j according to the rule

$$|\alpha_{ij}| \geq \varepsilon \sqrt{|\alpha_{ii}\alpha_{jj}|},$$

  with $\mathcal{A} = (\alpha_{ij})$ and ε given threshold

$$\begin{pmatrix}
\frac{4}{h^2} & -\frac{1}{h^2} & & -\frac{1}{h^2} & & & & & & & & \\
-\frac{1}{h^2} & \frac{4}{h^2} & -\frac{1}{h^2} & & -\frac{1}{h^2} & & & & & & & \\
 & -\frac{1}{h^2} & \frac{4}{h^2} & & & -\frac{1}{h^2} & & & & & & \\
-\frac{1}{h^2} & & & \frac{4}{h^2} & -\frac{1}{h^2} & & -\frac{1}{h^2} & & & & & \\
 & -\frac{1}{h^2} & & -\frac{1}{h^2} & \frac{4}{h^2} & -\frac{1}{h^2} & & -\frac{1}{h^2} & & & & \\
 & & -\frac{1}{h^2} & & -\frac{1}{h^2} & \frac{4}{h^2} & & & -\frac{1}{h^2} & & & \\
 & & & -\frac{1}{h^2} & & & \frac{4}{h^2} & -\frac{1}{h^2} & & -\frac{1}{h^2} & & \\
 & & & & -\frac{1}{h^2} & & -\frac{1}{h^2} & \frac{4}{h^2} & -\frac{1}{h^2} & & -\frac{1}{h^2} & \\
 & & & & & -\frac{1}{h^2} & & -\frac{1}{h^2} & \frac{4}{h^2} & & & -\frac{1}{h^2} \\
 & & & & & & -\frac{1}{h^2} & & & \frac{4}{h^2} & -\frac{1}{h^2} & \\
 & & & & & & & -\frac{1}{h^2} & & -\frac{1}{h^2} & \frac{4}{h^2} & -\frac{1}{h^2} \\
 & & & & & & & & -\frac{1}{h^2} & & -\frac{1}{h^2} & \frac{4}{h^2}
\end{pmatrix}$$

Setup of coarse-level correction

Focus on smoothed aggregation (Vaněk, Mandel, Brezina, Computing, 56, 1996)

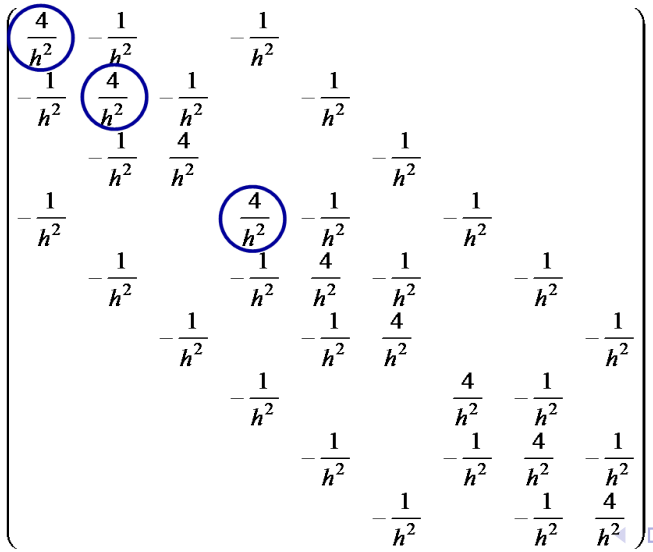
Coarsening

- scalar problems: obtain coarse indices by aggregating the indices of $\mathcal{V}$ into N' subsets that form a disjoint coverage of $\mathcal{V}$

  each subset $\mathcal{C}_s \equiv \mathcal{C}_s(j)$ contains a "root" index $j \in \mathcal{V}$ and other indices $i \in \mathcal{V}$ that are strongly coupled to j according to the rule

$$|\alpha_{ij}| \geq \varepsilon \sqrt{|\alpha_{ii}\alpha_{jj}|},$$

  with $\mathcal{A} = (\alpha_{ij})$ and ε given threshold



Setup of coarse-level correction

Focus on smoothed aggregation (Vaněk, Mandel, Brezina, Computing, 56, 1996)

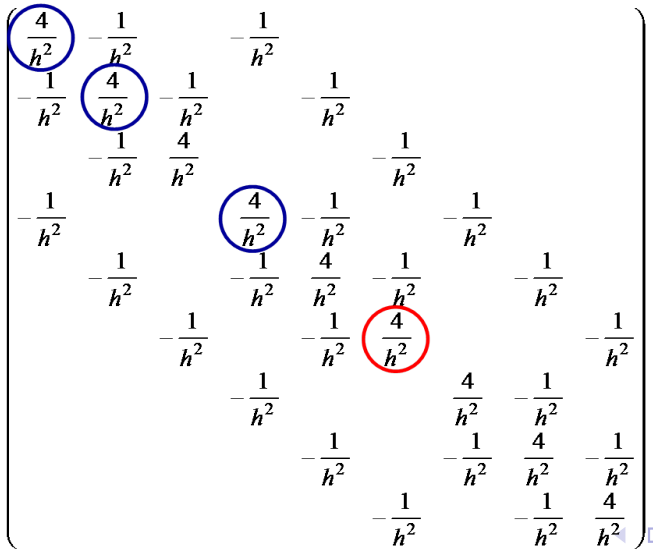
Coarsening

- scalar problems: obtain coarse indices by aggregating the indices of $\mathcal{V}$ into N' subsets that form a disjoint coverage of $\mathcal{V}$

  each subset $\mathcal{C}_s \equiv \mathcal{C}_s(j)$ contains a "root" index $j \in \mathcal{V}$ and other indices $i \in \mathcal{V}$ that are strongly coupled to j according to the rule

$$|\alpha_{ij}| \geq \varepsilon \sqrt{|\alpha_{ii}\alpha_{jj}|},$$

  with $\mathcal{A} = (\alpha_{ij})$ and ε given threshold



Setup of coarse-level correction

Focus on smoothed aggregation (Vaněk, Mandel, Brezina, Computing, 56, 1996)

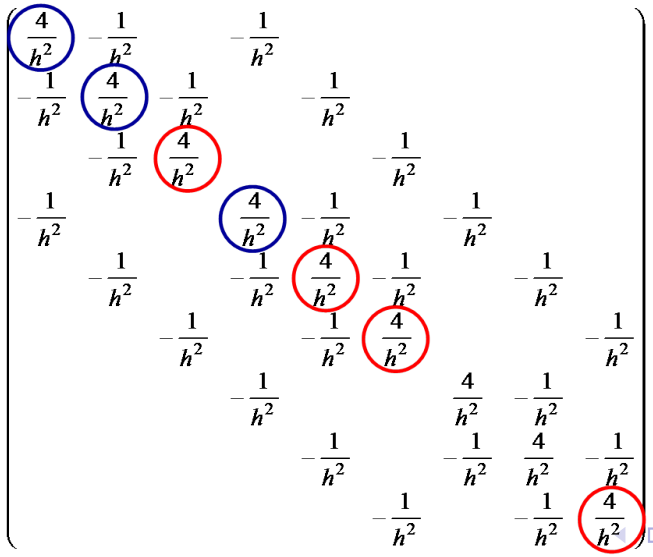
Coarsening

- scalar problems: obtain coarse indices by aggregating the indices of $\mathcal{V}$ into N' subsets that form a disjoint coverage of $\mathcal{V}$

  each subset $\mathcal{C}_s \equiv \mathcal{C}_s(j)$ contains a "root" index $j \in \mathcal{V}$ and other indices $i \in \mathcal{V}$ that are strongly coupled to j according to the rule

$$|\alpha_{ij}| \geq \varepsilon \sqrt{|\alpha_{ii}\alpha_{jj}|},$$

  with $\mathcal{A} = (\alpha_{ij})$ and ε given threshold



Setup of coarse-level correction

Focus on smoothed aggregation (Vaněk, Mandel, Brezina, Computing, 56, 1996)

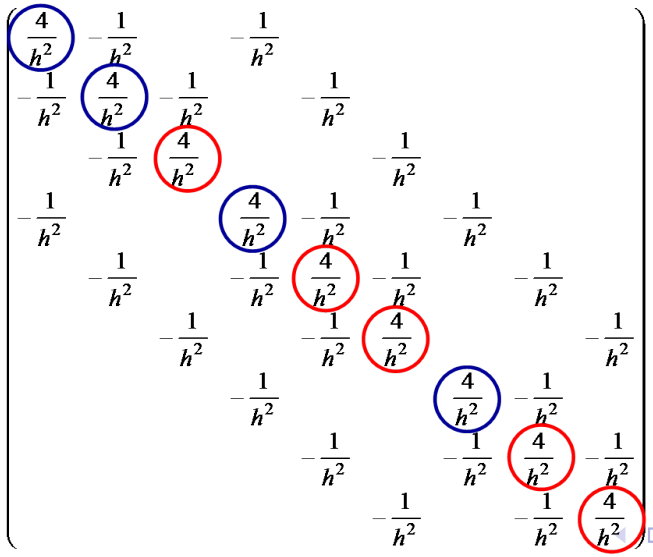
Coarsening

- scalar problems: obtain coarse indices by aggregating the indices of $\mathcal{V}$ into N' subsets that form a disjoint coverage of $\mathcal{V}$

  each subset $\mathcal{C}_s \equiv \mathcal{C}_s(j)$ contains a "root" index $j \in \mathcal{V}$ and other indices $i \in \mathcal{V}$ that are strongly coupled to j according to the rule

$$|\alpha_{ij}| \geq \varepsilon \sqrt{|\alpha_{ii}\alpha_{jj}|},$$

  with $\mathcal{A} = (\alpha_{ij})$ and ε given threshold



Setup of coarse-level correction

Focus on smoothed aggregation (Vaněk, Mandel, Brezina, Computing, 56, 1996)

Coarsening

- scalar problems: obtain coarse indices by aggregating the indices of $\mathcal{V}$ into N' subsets that form a disjoint coverage of $\mathcal{V}$

  each subset $\mathcal{C}_s \equiv \mathcal{C}_s(j)$ contains a "root" index $j \in \mathcal{V}$ and other indices $i \in \mathcal{V}$ that are strongly coupled to j according to the rule

  $$|\alpha_{ij}| \geq \varepsilon \sqrt{|\alpha_{ii}\alpha_{jj}|},$$

  with $\mathcal{A} = (\alpha_{ij})$ and ε given threshold

- nonscalar problems: aggregate simultaneously the dof's associated with each grid point, using

  $$\|H_{ij}\| \geq \varepsilon \sqrt{\|H_{ii}\|\|H_{jj}\|},$$

  with H_{ij} block counterpart of α_{ij} (point-block version)

Setup of coarse-level correction (cont'd)

Prolongation and restriction

- build a tentative prolongator $\widetilde{\mathcal{P}}$ whose range includes the near null space of the given matrix
  - scalar problems (e.g. discrete Laplace operator):

$$\widetilde{\mathcal{P}} = (\widetilde{p}_{ij}), \quad \widetilde{p}_{ij} = \begin{cases} 1 & \text{if } i \in \mathcal{V}_j \\ 0 & \text{otherwise} \end{cases}$$

  - nonscalar problems (point-block version): replace 1 and 0 in $\widetilde{\mathcal{P}}$ by identity and zero blocks of dim equal to # dof's

Setup of coarse-level correction (cont'd)

Prolongation and restriction

- build a tentative prolongator $\widetilde{\mathcal{P}}$ whose range includes the near null space of the given matrix
  - scalar problems (e.g. discrete Laplace operator):

$$\widetilde{\mathcal{P}} = (\widetilde{p}_{ij}), \quad \widetilde{p}_{ij} = \begin{cases} 1 & \text{if } i \in \mathcal{V}_j \\ 0 & \text{otherwise} \end{cases}$$

  - nonscalar problems (point-block version): replace 1 and 0 in $\widetilde{\mathcal{P}}$ by identity and zero blocks of dim equal to # dof's

- apply a smoother to reduce the energy of the prolongator basis vectors:

$$\mathcal{P} = (\mathcal{I} - \omega \mathcal{D}^{-1} \mathcal{A}) \widetilde{\mathcal{P}},$$

where $\mathcal{D} = diag(\mathcal{A})$ and $\omega = 4/(3\rho)$, with ρ upper bound on the spectral radius of $\mathcal{D}^{-1}\mathcal{A}$

Setup of coarse-level correction (cont'd)

Prolongation and restriction

- build a tentative prolongator $\widetilde{\mathcal{P}}$ whose range includes the near null space of the given matrix
  - scalar problems (e.g. discrete Laplace operator):

$$\widetilde{\mathcal{P}} = (\widetilde{p}_{ij}), \quad \widetilde{p}_{ij} = \begin{cases} 1 & \text{if } i \in \mathcal{V}_j \\ 0 & \text{otherwise} \end{cases}$$

  - nonscalar problems (point-block version): replace 1 and 0 in $\widetilde{\mathcal{P}}$ by identity and zero blocks of dim equal to # dof's

- apply a smoother to reduce the energy of the prolongator basis vectors:

$$\mathcal{P} = (\mathcal{I} - \omega \mathcal{D}^{-1} \mathcal{A}) \widetilde{\mathcal{P}},$$

  where $\mathcal{D} = diag(\mathcal{A})$ and $\omega = 4/(3\rho)$, with ρ upper bound on the spectral radius of $\mathcal{D}^{-1}\mathcal{A}$

- set

$$\mathcal{R} = \mathcal{P}^T$$

Our approach: basic idea

With the classical point-block approach, the transfer operators may significantly depend on the parameter $\nu \Longrightarrow$ loss of multilevel effectiveness

Our approach: basic idea

With the classical point-block approach, the transfer operators may significantly depend on the parameter $\nu \Longrightarrow$ loss of multilevel effectiveness

Decompose the system matrix $\mathcal{A}$ as

$$\mathcal{A} = \hat{\mathcal{A}} + \mathcal{B},$$

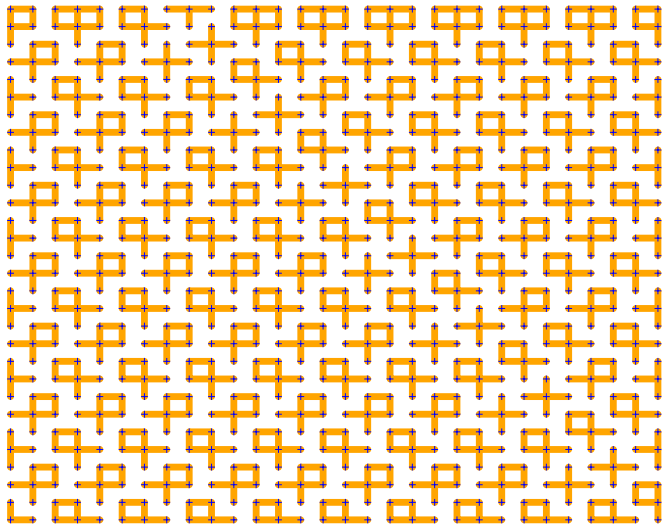
$$\hat{\mathcal{A}} = \begin{pmatrix} A & 0 & 0 \\ 0 & A & 0 \\ 0 & 0 & 0 \end{pmatrix}, \qquad \mathcal{B} = \begin{pmatrix} 0 & 0 & -I \\ I & 0 & 0 \\ 0 & -I & \nu I \end{pmatrix}$$

second-order terms zero-order terms

and build restriction and prolongation operators by using the block A instead of the whole matrix $\mathcal{A}$.

Our approach: coarsening and transfer operators

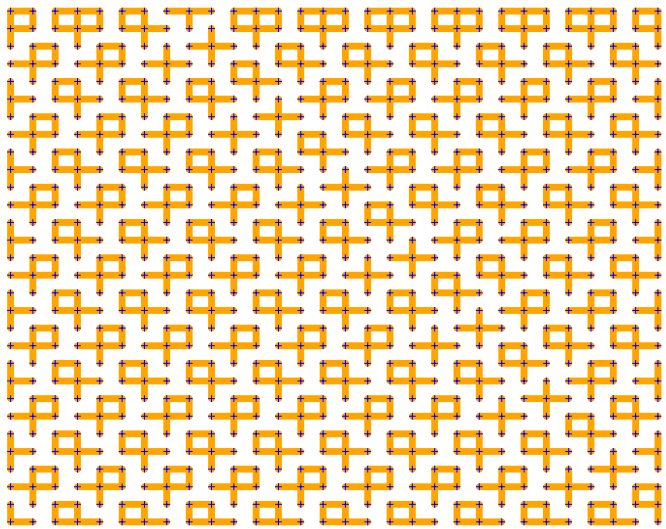
- apply scalar aggregation to the index set V associated with the block A (discrete negative Laplacian)



grid view

Our approach: coarsening and transfer operators

- apply scalar aggregation to the index set V associated with the block A (discrete negative Laplacian)



grid view

- build a tentative prolongator $\widetilde{P}$ corresponding only to the aggregates associated with the block A
- apply smoothing to $\widetilde{P}$:

$$P = (I_n - \omega D^{-1} A)\widetilde{P},$$

where $D = diag(A)$ and $\omega_B = 4/(3\rho_A)$, with ρ_A upper bound of the spectral radius of $D^{-1}A$

Our approach: coarsening and transfer operators (cont'd)

- extend the smoothed prolongator to the index set $\mathcal{V}$ associated with the whole matrix $\mathcal{A}$:

$$\hat{\mathcal{P}} = I_3 \otimes P = \begin{pmatrix} P & & \\ & P & \\ & & P \end{pmatrix}$$

- set

$$\hat{\mathcal{R}} = \hat{\mathcal{P}}^T$$

Setup of smoothers (one-level preconditioners)

Additive Schwarz (AS) preconditioners - basic ideas:

- divide the matrix (domain) into overlapping submatrices (subdomains)
- apply a "local preconditioning" in each subdomain
- build the global preconditioner from the local ones

Setup of smoothers (one-level preconditioners)

Additive Schwarz (AS) preconditioners - basic ideas:

- divide the matrix (domain) into overlapping submatrices (subdomains)
- apply a "local preconditioning" in each subdomain
- build the global preconditioner from the local ones

Our approach:

build the overlapping submatrices by using the sparsity pattern of the block A instead of the whole matrix $\mathcal{A}$

Our approach: one-level AS preconditioners

$G = (V, E)$ adjacency graph of the block A (symmetric pattern)

$$V = \bigcup V_i^0, \quad V_i^0 \cap V_j^0 \ \ (i \neq j) \text{ partition of } V$$

- build δ-overlap partition of V

$$V = \bigcup V_i^\delta, \quad V_i^\delta \supset V_i^{\delta-1}$$

$$j \in V_i^\delta \iff \exists k \in V_i^{\delta-1} : (j,k) \in E$$

- build restriction and prolongation operators associated with this partition

$$R_i^\delta = (e_{j_1}, e_{j_2} \dots e_{j_m})^T, \ j_k \in V_i^\delta, \quad P_i^\delta = (R_i^\delta)^T$$

- extend R_i^δ and P_i^δ

$$\hat{\mathcal{P}}_i^\delta = I_3 \otimes P_i^\delta, \quad \hat{\mathcal{R}}_i^\delta = I_3 \otimes R_i^\delta$$

- build restrictions of the whole matrix $\mathcal{A}$

$$\mathcal{A}_i^\delta = \hat{\mathcal{R}}_i^\delta \mathcal{A} \hat{\mathcal{P}}_i^\delta$$

AS preconditioner

$$\mathcal{M}_{AS}^{-1} = \sum_{i=1}^{m} \hat{\mathcal{P}}_i^\delta (\mathcal{A}_i^\delta)^{-1} \hat{\mathcal{R}}_i^\delta$$

Remarks

Prolongation and restriction operators built by neglecting the zero-order part of the optimality system

- reduction of computational cost (time and memory)
- prolongation and restriction operators independent of the regularization parameter ν (can be reused)

Parallel version

Decomposition of the matrix $\mathcal{A}$ among the processors

- reordering of the rows and columns of $\mathcal{A}$ according to a point-block ordering of the unknowns (dof's associated with the same grid point are consecutive):

$$\bar{\mathcal{A}}\bar{\mathbf{v}} = \bar{\mathbf{w}}$$

where

$$\bar{\mathcal{A}} = \begin{pmatrix} F & E & \cdots & E & & & & & \\ E & F & E & \cdots & E & & & & \\ & \ddots & \ddots & \ddots & \ddots & \ddots & & & \\ E & \cdots & E & F & E & \cdots & E & & \\ & \ddots & & \ddots & \ddots & \ddots & & \ddots & \\ & & E & \cdots & E & F & E & \cdots & E \\ & & & \ddots & & \ddots & \ddots & \ddots & \\ & & & & E & \cdots & E & F & E \\ & & & & & E & \cdots & E & F \end{pmatrix}$$

$$F = \begin{pmatrix} \frac{4}{h^2} & 0 & -1 \\ 1 & \frac{4}{h^2} & 0 \\ 0 & -1 & \nu \end{pmatrix}$$

$$E = \begin{pmatrix} -\frac{1}{h^2} & 0 & 0 \\ 0 & -\frac{1}{h^2} & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

h = grid spacing in the x and y directions

Parallel version

Decomposition of the matrix $\mathcal{A}$ among the processors

- reordering of the rows and columns of $\mathcal{A}$ according to a point-block ordering of the unknowns (dof's associated with the same grid point are consecutive):

$$\bar{\mathcal{A}}\bar{\mathbf{v}} = \bar{\mathbf{w}}$$

where

$$\bar{\mathcal{A}} = \begin{pmatrix} F & E & \cdots & E & & & & & \\ E & F & E & \cdots & E & & & & \\ & \ddots & \ddots & \ddots & \ddots & \ddots & & & \\ E & \cdots & E & F & E & \cdots & E & & \\ & \ddots & & \ddots & \ddots & \ddots & & \ddots & \\ & & E & \cdots & E & F & E & \cdots & E \\ & & & \ddots & & \ddots & \ddots & \ddots & \\ & & & & E & \cdots & E & F & E \\ & & & & & E & \cdots & E & F \end{pmatrix}$$

$$F = \begin{pmatrix} \frac{4}{h^2} & 0 & -1 \\ 1 & \frac{4}{h^2} & 0 \\ 0 & -1 & \nu \end{pmatrix}$$

$$E = \begin{pmatrix} -\frac{1}{h^2} & 0 & 0 \\ 0 & -\frac{1}{h^2} & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

h = grid spacing in the x and y directions

- row-block decomposition of the reordered matrix, where rows corresponding to the same grid point are assigned to the same processor
- induced row-block decomposition of the block A and 0-overlap partition of the corresponding vertex set V; each processor holds a single subdomain

Parallel version (cont'd)

Decoupled aggregation, i.e., performed locally on each processor

- embarrassingly parallel
- may produce nonuniform aggregates near the boundary indices and depends on the number of processors, but works well in practice
  [Tuminaro & Tong, Supercomputing Conference, 2000; Buttari, D'Ambra, dS, Filippone, APNUM 57 & AAECC 18, 2007]

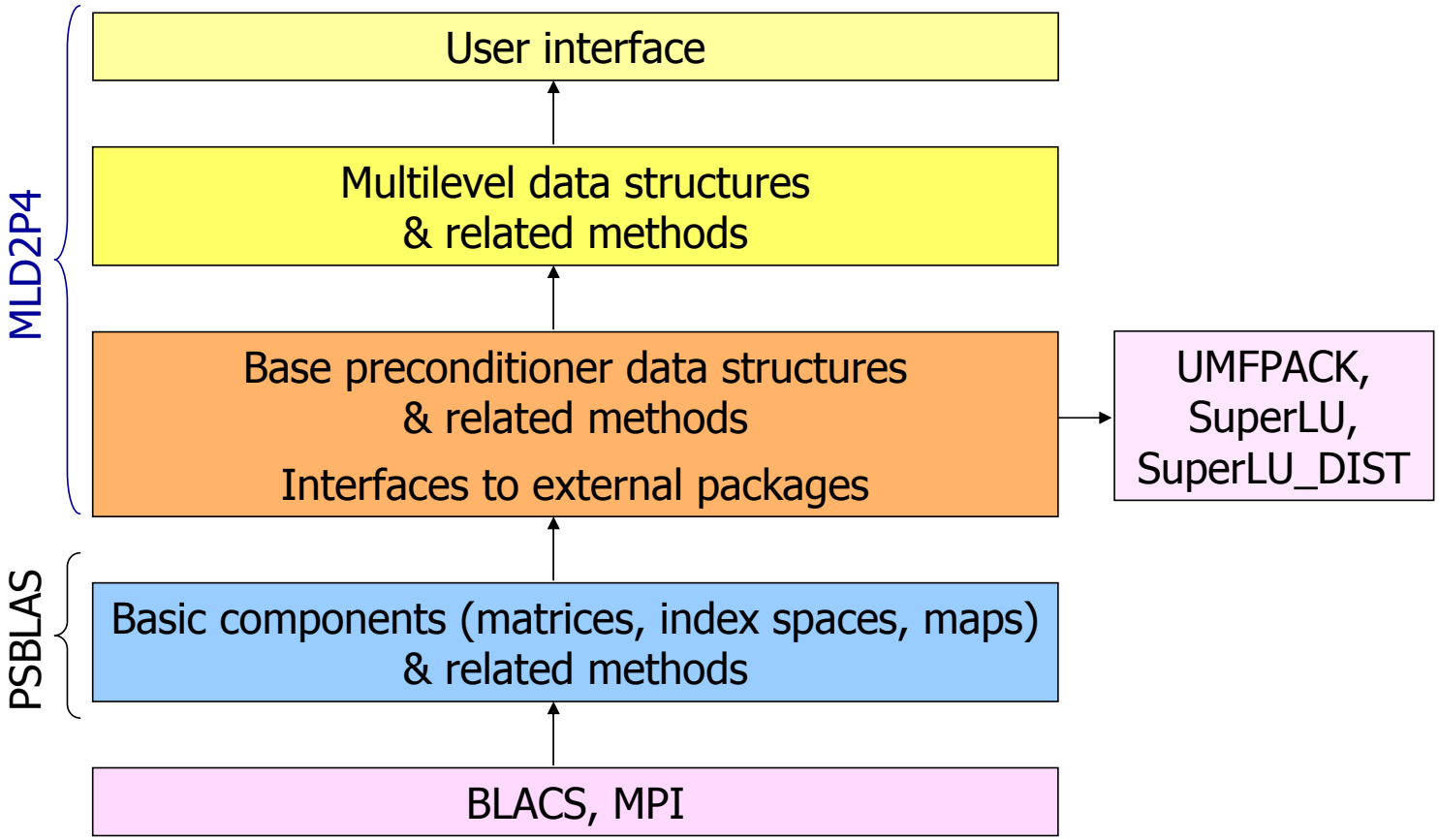
Formulation of the preconditioner setup and application phases in terms of basic sparse linear algebra computation and communication kernels

- explotation of "de-facto" standard basic linear algebra software (PSBLAS)

Software framework

http://www.mld2p4.it

MLD2P4 - MultiLevel Domain Decomposition Parallel Preconditioners Package based on PSBLAS



D'Ambra, dS, Filippone, *MLD2P4: a Package of Parallel Algebraic Multilevel Domain Decomposition Preconditioners in Fortran 95*, ACM TOMS, 37, 2010

Testing details

Test problem

- target function: $z(x,y) = \sin(2\pi x)\cos(2\pi y)$ (non attainable)
- rhs in the state equation: $f(x,y) = \sin(2\pi x)\cos(2\pi y)$
- regularization parameter: $\nu = 10^{-3}, 10^{-5}, 10^{-7}$
- grid size: $n \times n$, $n = 250, 500, 1000, 2000, 4000$ (matrix dim $= 187.5K, 750K, 3M, 12M, 48M$)

Testing details

Test problem

- target function: $z(x, y) = \sin(2\pi x)\cos(2\pi y)$ (non attainable)
- rhs in the state equation: $f(x, y) = \sin(2\pi x)\cos(2\pi y)$
- regularization parameter: $\nu = 10^{-3}, 10^{-5}, 10^{-7}$
- grid size: $n \times n$, $n = 250, 500, 1000, 2000, 4000$ (matrix dim $= 187.5K, 750K, 3M, 12M, 48M$)

Krylov solver

- BiCGStab from PSBLAS, with zero initial guess and stopping criterion $\|r^{(i)}\|_2/\|r^{(0)}\|_2 \leq \text{tol}$, with tol $= 10^{-6}, 10^{-12}$, or max # iters $= 1000$

Preconditioners

- 1-lev: block Jacobi (BJAC), restricted additive Schwarz with overlap 1 (RAS)
- multi-lev: V-cycle with 2–6 levels, BJAC or RAS as smoother, coarsest matrix replicated on the procs, LU from UMFPACK as coarsest-level solver
- aggreg. threshold: $\varepsilon = 0.08 \cdot 1/2^{k-1}$, $k =$ current lev. (Vaněk et al., Computing, 56, 1996)

Testing details

Test problem

- target function: $z(x, y) = \sin(2\pi x)\cos(2\pi y)$ (non attainable)
- rhs in the state equation: $f(x, y) = \sin(2\pi x)\cos(2\pi y)$
- regularization parameter: $\nu = 10^{-3}, 10^{-5}, 10^{-7}$
- grid size: $n \times n$, $n = 250, 500, 1000, 2000, 4000$ (matrix dim $= 187.5K, 750K, 3M, 12M, 48M$)

Krylov solver

- BiCGStab from PSBLAS, with zero initial guess and stopping criterion $\|r^{(i)}\|_2/\|r^{(0)}\|_2 \leq$ tol, with tol $= 10^{-6}, 10^{-12}$, or max # iters $= 1000$

Preconditioners

- 1-lev: block Jacobi (BJAC), restricted additive Schwarz with overlap 1 (RAS)
- multi-lev: V-cycle with 2–6 levels, BJAC or RAS as smoother, coarsest matrix replicated on the procs, LU from UMFPACK as coarsest-level solver
- aggreg. threshold: $\varepsilon = 0.08 \cdot 1/2^{k-1}$, $k =$ current lev. (Vaněk et al., Computing, 56, 1996)

Parallel machine

HP XC 6000 Linux cluster operated by the Naples branch of ICAR-CNR:

- Intel Itanium 2 Madison dual-processor nodes (1.4 Ghz), 4GB RAM
- Quadrics QsNetII Elan 4 interc. network (900 MB/sec sust. bandwidth, 5μsec latency)

Iterations as ν varies – grid size $= 1000 \times 1000$, tol $= 10^{-6}$, BJAC smoother

ν	NP	BJAC	2LEV	3LEV	4LEV	5LEV	6LEV
10^{-3}	1	526	4	6	7	9	10
	2	800	4	5	7	9	10
	4	588	4	6	8	10	11
	8	643	4	6	7	10	10
	16	602	4	5	7	10	15
	32	830	5	6	8	11	16
10^{-5}	1	—	4	7	8	11	12
	2	—	5	7	8	12	12
	4	—	5	6	9	13	13
	8	—	5	7	9	10	11
	16	—	5	7	9	12	17
	32	—	5	7	9	14	25
10^{-7}	1	—	5	8	9	12	—
	2	—	5	7	9	11	—
	4	—	5	6	9	9	—
	8	—	5	7	9	9	—
	16	—	5	6	8	10	40
	32	—	5	7	9	11	31

— = max # iters achieved

Iterations as ν varies (cont'd)

grid size $= 2000 \times 2000$, tol $= 10^{-6}$, BJAC smoother, 6 levels

NP	$\nu = 10^{-3}$	$\nu = 10^{-5}$	$\nu = 10^{-7}$
1	9	11	16
2	10	14	16
4	12	16	25
8	13	17	15
16	10	13	13
32	15	23	14

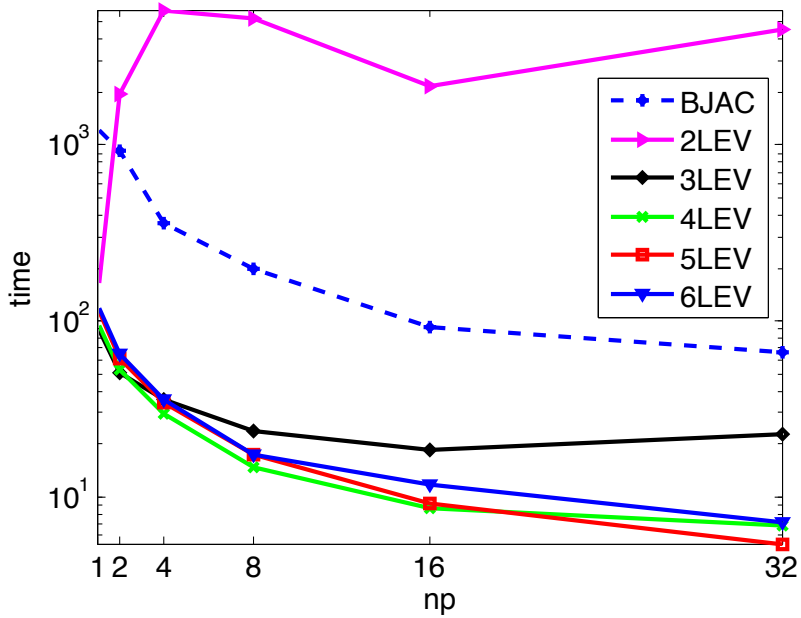
Iterations as grid size varies

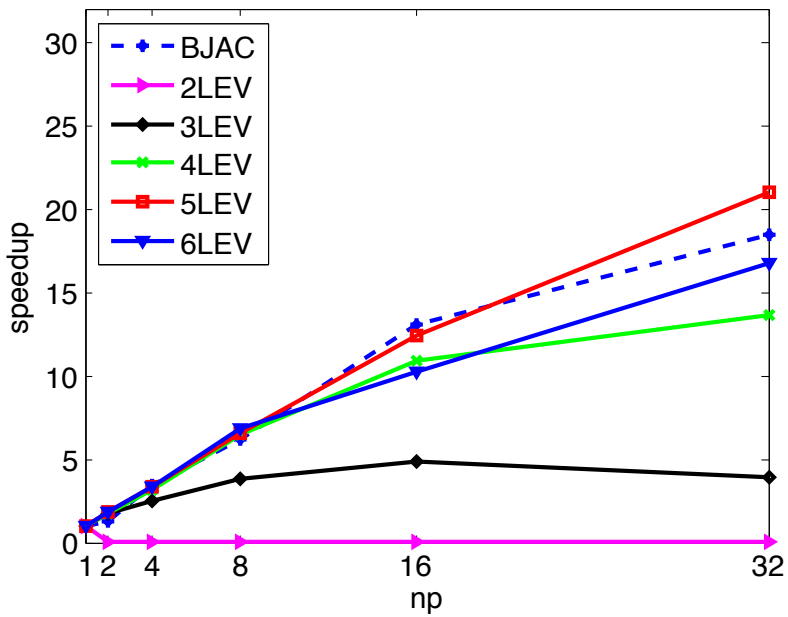
$\nu = 10^{-5}$, tol $= 10^{-6}$, BJAC smoother

grid size	NP	2LEV	3LEV	4LEV	5LEV	6LEV
250×250	1	4	5	6	7	7
	2	4	5	7	8	8
	4	4	6	7	8	8
	8	5	5	7	9	10
	16	5	5	7	8	9
	32	4	5	7	10	10
500×500	1	4	5	7	8	8
	2	4	6	7	9	9
	4	4	6	7	8	9
	8	4	5	7	10	10
	16	4	6	7	12	14
	32	5	6	7	12	13
1000×1000	1	4	5	7	10	10
	2	4	5	7	9	10
	4	4	6	8	10	13
	8	4	6	7	10	11
	16	5	6	8	11	16
	32	5	6	8	11	16

Execution time (sec.) & strong scaling – grid size $= 1000 \times 1000$, tol $= 10^{-6}$

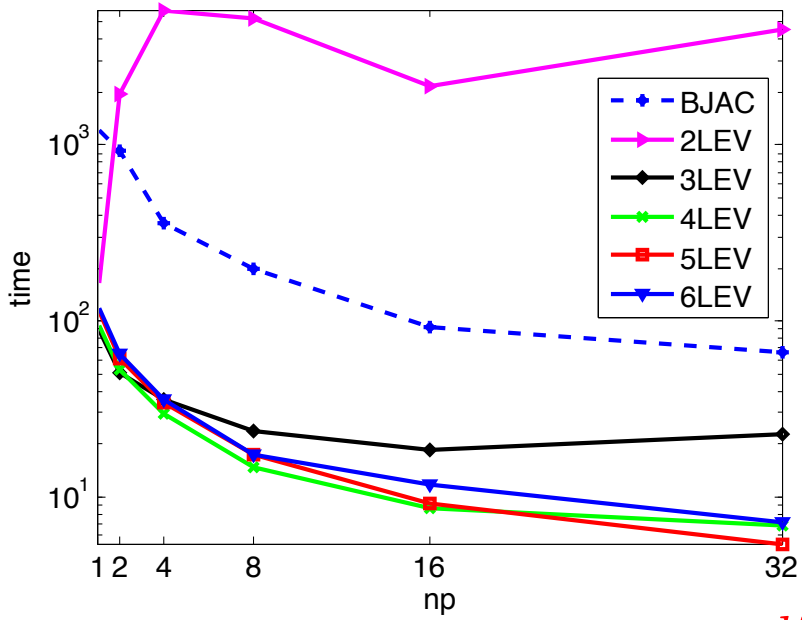
$\nu = 10^{-3}$

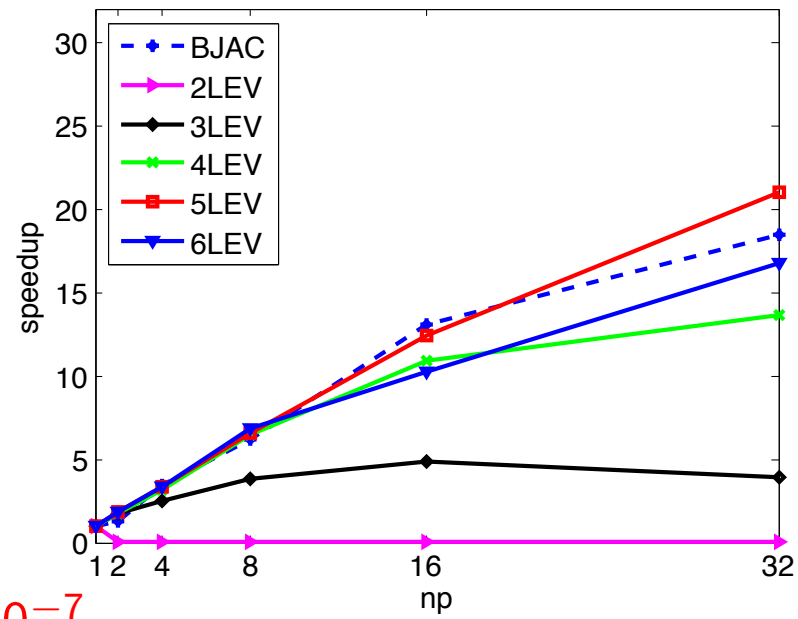




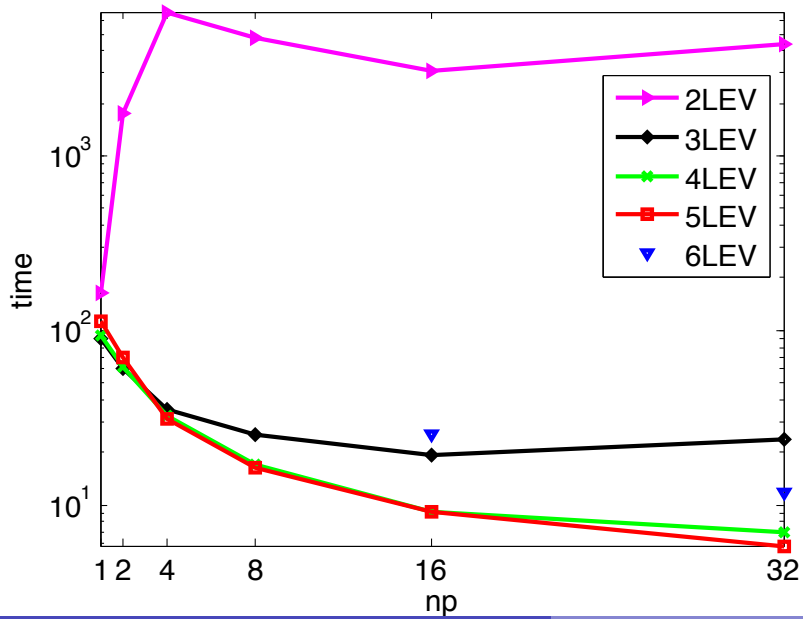
Execution time (sec.) & strong scaling – grid size $= 1000 \times 1000$, tol $= 10^{-6}$

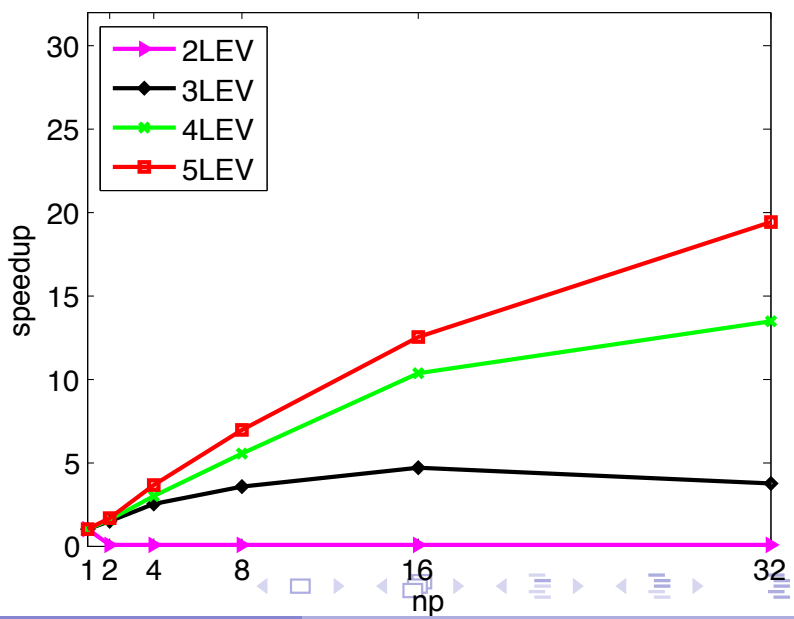
$\nu = 10^{-3}$





$\nu = 10^{-7}$

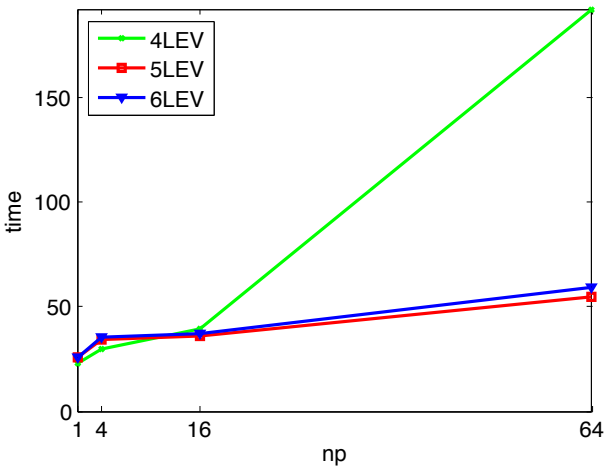




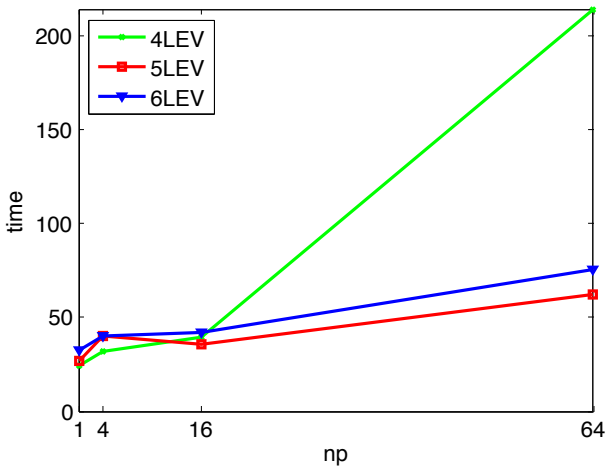
Weak scaling

$\#$ grid points per proc $= 500^2$, tol $= 10^{-6}$

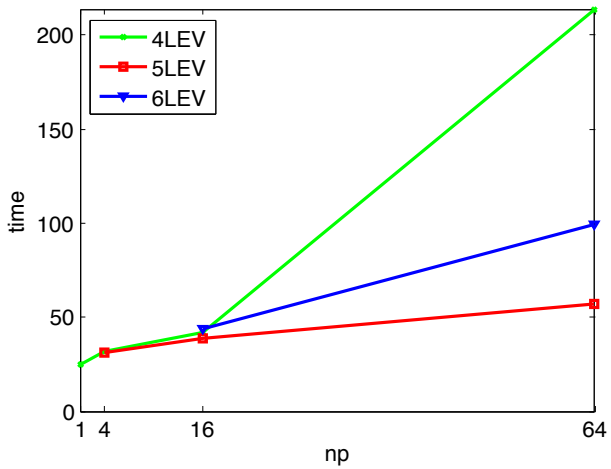
largest matrix dim (64 procs) $= 48M$



$\nu = 10^{-3}$



$\nu = 10^{-5}$



$\nu = 10^{-7}$

Comparison with classical point-block smoothed aggregation: iterations

- classical smoothed aggregation from Trilinos/ML (Gee et al., 2006 - http://trilinos.sandia.gov/packages/ml/)
  - V-cycle, BJAC as smoother, UMFPACK as coarsest-level solver
  - fixed aggregation threshold: $\varepsilon = 0, 0.1, 0.01, 0.001$ (adaptive choice not available)
- BiCGStab from Trilinos/AztecOO

Comparison with classical point-block smoothed aggregation: iterations

- classical smoothed aggregation from Trilinos/ML (Gee et al., 2006 - http://trilinos.sandia.gov/packages/ml/)
  - V-cycle, BJAC as smoother, UMFPACK as coarsest-level solver
  - fixed aggregation threshold: $\varepsilon = 0, 0.1, 0.01, 0.001$ (adaptive choice not available)
- BiCGStab from Trilinos/AztecOO

grid size $= 500 \times 500$, tol $= 10^{-6}$, max # iters $= 200$, $\varepsilon = 0.01$, NP=1

PREC	ν	2LEV	3LEV	4LEV	5LEV	6LEV
Classical	10^{-3}	4	5	5	6	6
New		4	5	7	8	8
Classical	10^{-5}	5	81	—	—	—
New		4	7	8	8	11
Classical	10^{-7}	5	—	—	—	—
New		5	7	7	—	—

significant increase of residuals observed with classical algorithm for $\nu = 10^{-5}, 10^{-7}$

Conclusions

Our preconditioner appears to be

- independent of the problem/grid size, if the coarsest matrix is not "too coarse"
- pretty robust w.r.t. regularization parameter
- suitable for parallel implementation

Conclusions

Our preconditioner appears to be

- independent of the problem/grid size, if the coarsest matrix is not "too coarse"
- pretty robust w.r.t. regularization parameter
- suitable for parallel implementation

Ongoing and future work

- theoretical analysis (extension of classical convergence results in an abstract framework)
- extension to other PDE-constrained pbs. / PDE systems

THANK YOU FOR YOUR ATTENTION