

Overlapping Partitioning and Applications to Preconditioning

David Fritzsche¹, Andreas Frommer¹, Daniel B. Szyld²

¹Bergische Universität Wuppertal, Germany

²Temple University, Philadelphia, USA

8th GAMM Workshop on Applied and Numerical Linear Algebra
September 11–12 2008, Hamburg-Harburg, Germany



Outline

- ▶ Introduction
- ▶ Iterative Solvers and Preconditioners
(esp. Block Gauss-Seidel Preconditioning)
- ▶ preconditioning with PABLO
- ▶ Schwarz Methods
- ▶ Algebraic Schwarz Methods
- ▶ Graph partitioning / preconditioning with PABLO
- ▶ Adding Overlap
- ▶ Numerical Results



Introduction

We consider the linear system

$$Ax = b$$

where $A \in \mathbb{R}^{n \times n}$ is *large, sparse and non-symmetric*.

In this talk

- ▶ We solve it using preconditioned, restarted GMRES
- ▶ We show how to permute and partition A to obtain diagonal blocks **with overlap** to produce an effective preconditioner, based on algebraic Schwarz methods



Outline

- ▶ Schwarz Methods
- ▶ Algebraic Schwarz Methods
- ▶ Graph partitioning / preconditioning with PABLO
- ▶ Adding Overlap
- ▶ Numerical Results



Iterative Solvers and Preconditioners

Problem: In many applications $Ax = b$ can not be solved directly.

↪ **iterative solvers** (Jacobi, CG, GMRES, ...), Here: **GMRES**.

GMRES finds an approximate solution $x_n = \operatorname{argmin}_{x' \in K_n} \|Ax' - b\|$
where $K_n = \operatorname{span}\{b, Ab, A^2b, \dots, A^{n-1}b\}$.

Next Problem: Convergence of iterative solvers is often slow.

↪ **preconditioning**, Here: **left** preconditioning.

Left Preconditioning: Find $M^{-1} \approx A^{-1}$ and replace $Ax - b$ by the equivalent system

$$M^{-1}Ax = M^{-1}b$$



Block Gauss-Seidel Preconditioning

Suppose A has the $q \times q$ block structure

$$A = \begin{bmatrix} A_{11} & \cdots & A_{1q} \\ \vdots & \ddots & \vdots \\ A_{q1} & \cdots & A_{qq} \end{bmatrix}.$$

Let now D , L and U be the block diagonal, block lower triangular and block upper triangular part of A .

Block Gauss-Seidel: Choose $M = D + L$. Then $M^{-1}Ax = M^{-1}b$ becomes $(L + D)^{-1}Ax = (L + D)^{-1}b$.

Very little additional cost since

$$M^{-1}Ax = (L + D)^{-1}Ax = (I + (L + D)^{-1}U)x$$



Finding a “good” block structure

Problems:

- ▶ What is a “good” block structure?
- ▶ How to find a “good” block structure?

What is a “good” block structure?

Heuristic (inspired on a Theorem in [Varga,1962]):

The block structure is “good” if

- ▶ The diagonal blocks A_{ii} are quite **full** and contain most (all) **large entries** of the matrix.
- ▶ The off-diagonal blocks are **sparse** and contain only **small entries**.

Then the block diagonal D is a “good” approximation of A .

How to find a “good” block structure?

↪ Find a partitioning of the graph of A



Preconditioning with PABLO, part I

PABLO (**PA**ramterized **BL**ock **O**rding) is a simple algorithm for graph partitioning.

Basic features:

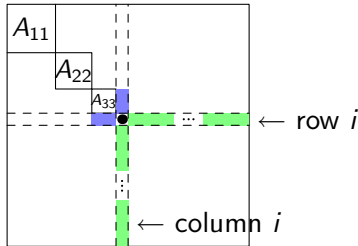
- ▶ Builds only one block at a time
- ▶ Considers only one vertex at a time for inclusion
- ▶ Checks all adjacent vertices for inclusion
- ▶ PABLO tries to put large matrix entries into the diagonal blocks.
- ▶ Time complexity is $\mathcal{O}(n + \text{nnz}(A))$: **PABLO is fast**

[Fritzsche, Frommer, and Szyld, 2007]



How PABLO works

Situation: Check vertex i for inclusion into block A_{33}



Several Criteria:

- ▶ **Fullness:** The blue part is "full" enough
- ▶ **Connectivity:** More entries in the blue part than in the green part
- ▶ **Threshold:** We want large entries in the blue part



Preconditioning with PABLO, part II

PABLO needs to know which matrix entries are large.

Preprocessing

- ▶ Scaling and permuting A into an I -matrix using MC64, i.e., compute permutation matrix P and scaling matrices R and C such that

$$\begin{aligned}(PRAC)_{ii} &= 1 && \text{for } i = 1, \dots, n \\ (PRAC)_{ij} &\leq 1 && \text{for } i \neq j\end{aligned}$$

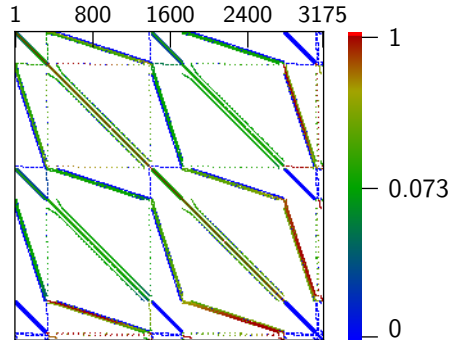
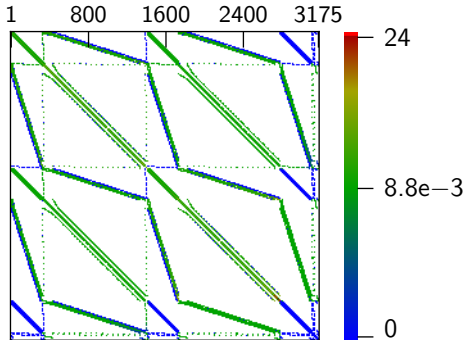
Set $A \leftarrow PRAC$

- ▶ Numerical experiments have shown that this kind of scaling is crucial to get good results with PABLO

[Duff and Koster, 1999 and 2001]



PABLO Example: GARON1

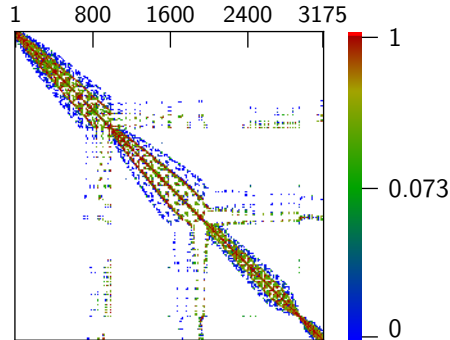
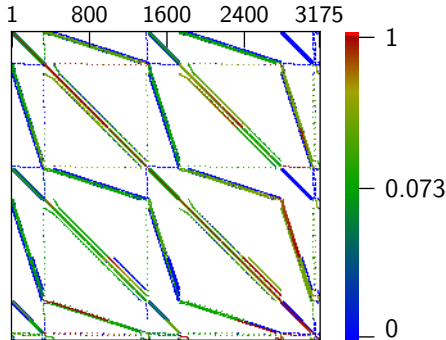


Left: Original matrix;

Right: The matrix scaled by MC64, but not yet permuted to be an I -matrix



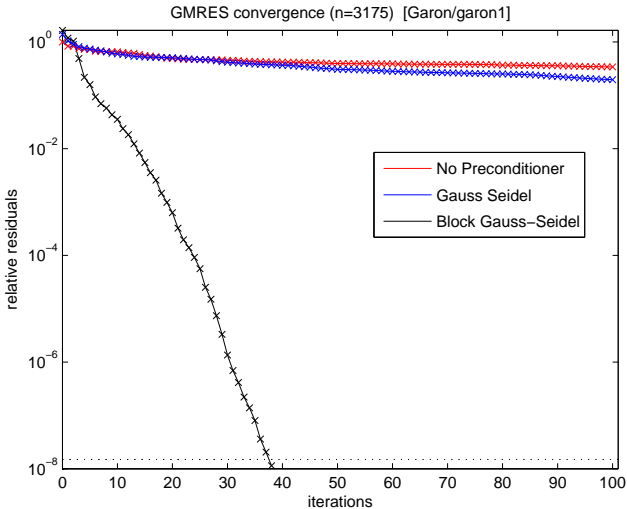
PABLO Example: GARON1



Left: After scaling and permuting by MC64 (*I*-matrix);
Right: The matrix after the PABLO permutation



PABLO Example: GARON1

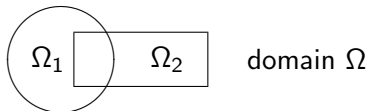




Schwarz Alternating Method

Schwarz introduced in 1870 an *alternating method* for solving partial differential equations on irregular domains.

Setting: Let Ω be the union of the “regular” shaped subdomains Ω_i , $i = 1, \dots, m$



Schwarz alternating method:

- ▶ Solve the PDE on Ω_i , $i = 1, \dots, m$
For $\partial\Omega_i \setminus \partial\Omega$ use interior values of a different subdomain
- ▶ Repeat until convergence

[Toselli and Widlund, 2005]



Multiplicative Schwarz Method I

Notation

- ▶ Let $Ax = b$ be a discretization of the PDE
- ▶ $R_i :=$ restriction operator onto Ω_i , $i = 1, \dots, m$
- ▶ $A_i := R_i A R_i^T$, $A_i \in \mathbb{R}^{n_i \times n_i}$, $i = 1, \dots, m$

The k -th iteration of the *Multiplicative Schwarz Method* is

$$x_{k,0} := x_k$$

for $i = 1, \dots, m$ **do**

$$x_{k,i} := x_{k,i-1} + R_i^T A_i^{-1} R_i (b - A x_{k,i-1})$$

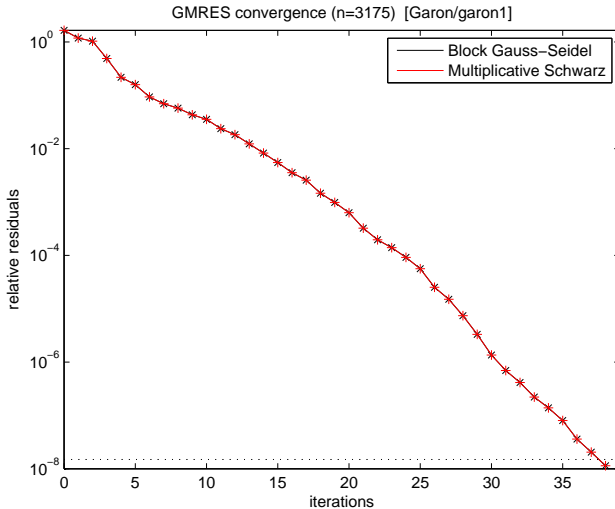
end for

$$x_{k+1} := x_{k,m}$$

Note: The Multiplicative Schwarz Method is equivalent to the block Gauss-Seidel iteration if $\Omega_i \cap \Omega_j = \emptyset$ for $i \neq j$.



Multiplicative Schwarz w/o Overlap = Block Gauss-Seidel





Multiplicative Schwarz Method II

Notation

- ▶ $\Pi_i := R_i^T A_i^{-1} R_i A$
- ▶ $Q_m := (I - \Pi_m)(I - \Pi_{m-1}) \cdots (I - \Pi_1)$

Lemma: One iteration of the *Multiplicative Schwarz Method* is equivalent to

$$x_{k+1} := Q_m x_k + (I - Q_m) A^{-1} b$$



Algebraic Schwarz Methods

Idea: Generalize the Multiplicative Schwarz Method by considering just the “algebraic” parts of the method:

- ▶ A linear system $Ax = b$
- ▶ restrictions $R_i : \mathbb{R}^n \rightarrow \mathbb{R}^{n_i}$ into overlapping subspaces of \mathbb{R}^n
- ▶ using the iterative method just shown

- + Works without knowledge about the PDE/discretization/...
- A lot of information is not used

Note: It is possible to prove convergence for several classes of matrices (easiest: A spd)



Schwarz Preconditioning

Left preconditioning: $A \rightarrow M^{-1}A$

For the Multiplicative Schwarz Method:

$$A \rightarrow I - Q_m$$

(follows from the iteration $x_{k+1} := Q_m x_k + (I - Q_m)A^{-1}b$)

Preconditioned matrix-vector multiplication:

input: vector v

output: $z := M^{-1}Av$ $\{z := (I - Q_m)v\}$

$z := 0$

for $i = 1, \dots, m$ **do**

$z := z + (R_i^T A_i^{-1} R_i)A(v - z)$

end for



PABLO → OPABLO

PABLO:

1. Preprocess
2. Find blocks using PABLO
3. Solve with GMRES + Block Gauss-Seidel

OPABLO:

1. Preprocess
2. Find blocks using PABLO
- 2b. Add some overlap to each block
3. Solve with GMRES + Multiplicative Schwarz



Level Sets

How to add overlap?

- ▶ Add (partial) *level sets*

Definition: Let $G = (V, E)$ be a graph, $W_i \subset V$. The *adjacency set* $\text{adj}(W_i)$ is the set of all nodes adjacent to W_i , but not in W_i , i.e.,

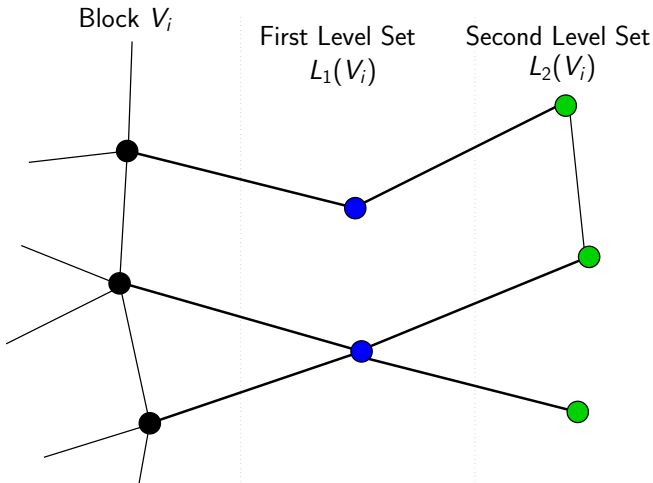
$$\text{adj}(W_i) := \{v \in V \setminus W_i \mid v \text{ adjacent to } W_i\}.$$

Definition: The *kth level set* with respect to W_i is defined as

$$L_k(W_i) := \begin{cases} W_i & \text{if } k = 0, \\ \text{adj}(W_i) & \text{if } k = 1, \\ \text{adj}(L_{k-1}(W_i)) \setminus L_{k-2}(W_i) & \text{if } k > 1. \end{cases}$$



Level Sets (cont.)





Weighting Nodes

Problem: In many cases the complete level set is too large.
How to find a suitable subset?

(Obvious) Idea: Introduce some node-weights and choose only nodes with large weights.

Definition: Let $W_i \subset V$ be a set of nodes and $k \in V$.
The weight $w(k, W_i)$ is defined as

$$w(k, W_i) := \begin{cases} \sum_{j \in W_i} |a_{kj}| + |a_{jk}| & \text{if } k \notin W_i, \\ 0 & \text{if } k \in W_i. \end{cases}$$

Note: Naturally you can choose other ways of weighting



Selecting a Partial Level Set

- ▶ We only add nodes from $L_1(W_i)$
- ▶ We add at most $m(W_i) := \sqrt{|W_i|}$ nodes.
Note: Choosing $m(W_i) = \sqrt{|W_i|}$ is quite arbitrary, but worked well in experiments. The choice of $m(W_i)$ was motivated by the size of the border in a discretization of a 2D PDE.
The challenge is in fact not to add too many nodes
- ▶ Let $N = N(W_i) \subset L_1(W_i)$ be the set we actually add to W_i . It consist of the nodes with highest weights, i.e.,

$$w(j, W_i) \geq w(k, W_i) \quad \text{if } j \in N(W_i) \text{ and } k \in L_1(W_i) \setminus N(W_i).$$

OPABLO(ℓ) – repeat the block growing process ℓ times



Complexity of OPABLO

Notation for this slide:

q is the number of blocks found by PABLO

d is the maximum degree in $G(A)$

Theorem.

- ▶ OPABLO(1) is $\mathcal{O}(n + \text{nnz}(A))$.
- ▶ OPABLO(ℓ) is $\mathcal{O}(q(n \log n + \text{nnz}(A)))$. (**worst case**)
- ▶ OPABLO(ℓ) is $\mathcal{O}((d + \ell)n)$ if $m(W_i), \ell = \mathcal{O}(\sqrt{|W_i|})$

Note: If the matrix stems from a finite-difference discretization, then OPABLO(ℓ) is $\mathcal{O}(\ell n)$.

Thus, **OPABLO is still fast.**

Working now on some implementations to make it more efficient.



Numerical Results

Definition:

Let $S \subset V$. Let $R(S)$ be the restriction operator corresponding to S . The *submatrix of A corresponding to S* is

$$A(S) := R(S)AR(S)^T.$$

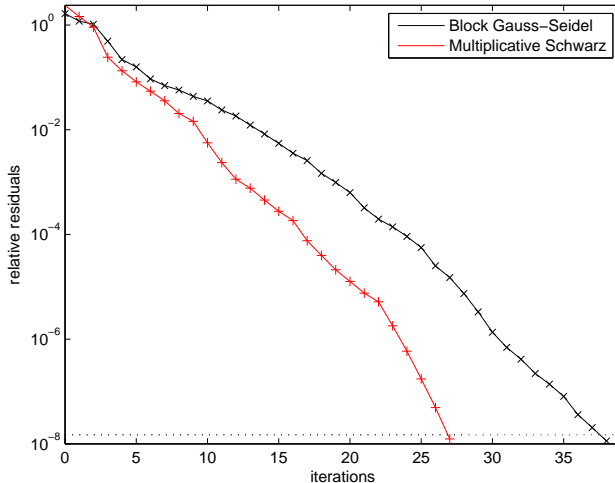
Notation:

- ▶ $W_i, i = 1, \dots, q$, are the blocks found by PABLO (no overlap)
- ▶ $nz_i := \text{nnz}(A(W_i))$
- ▶ $V_i, i = 1, \dots, q$, are the blocks after growing them (overlap)
- ▶ $nz'_i := \text{nnz}(A(V_i))$



GARON1 — adding 1 level set

GMRES convergence (n=3175) [Garon/garon1]



$$n = 3175$$

$$nnz = 84\,723$$

$$\sum |W_i| = 3175$$

$$\sum nz_i = 79\,123$$

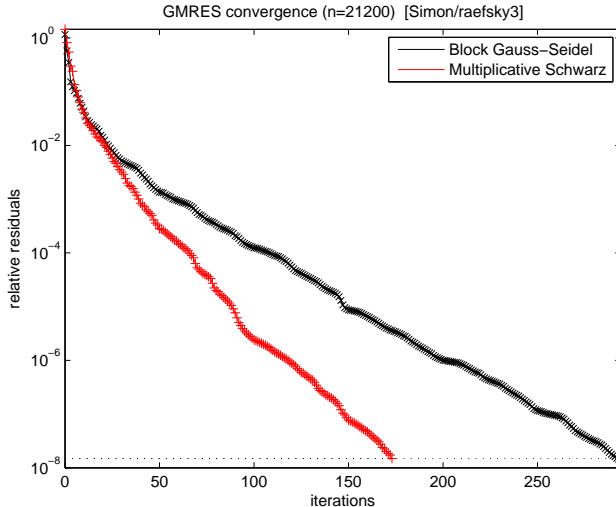
$$1 \times \text{add_level_set}$$

$$\sum |V_i| = 3619$$

$$\sum nz'_i = 91\,028$$



RAEFSKY3 — adding 1 partial level set



$$n = 21\,200$$

$$nnz = 1\,488\,768$$

$$\sum |W_i| = 21\,200$$

$$\sum nz_i = 864\,540$$

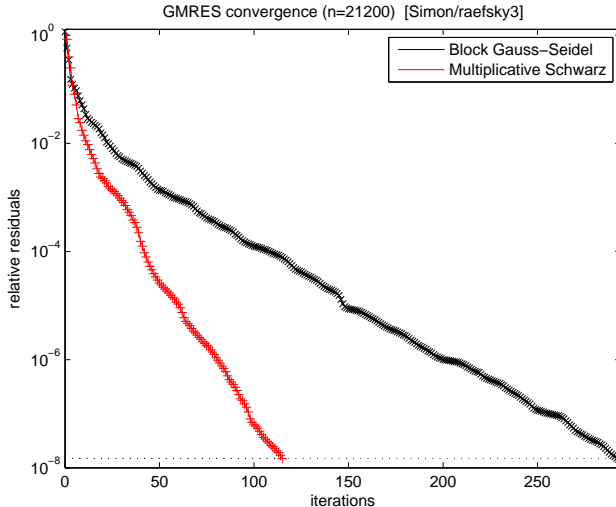
$$1 \times \text{add_level_set}$$

$$\sum |V_i| = 21\,887$$

$$\sum nz'_i = 896\,409$$



RAEFSKY3 — adding 3 partial level sets



$$n = 21\,200$$

$$nnz = 1\,488\,768$$

$$\sum |W_i| = 21\,200$$

$$\sum nz_i = 864\,540$$

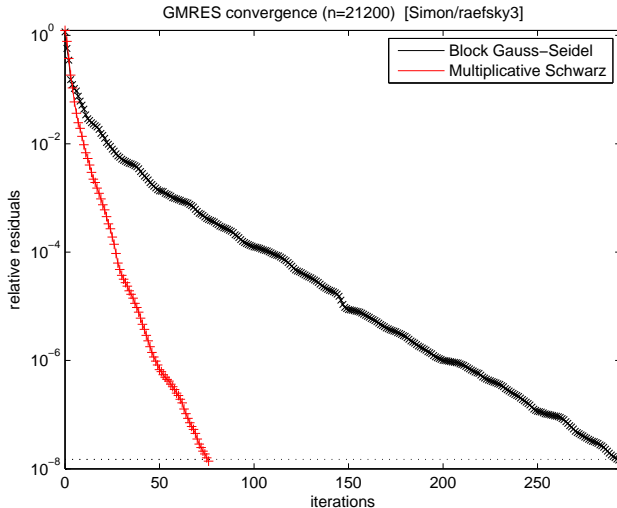
$$3 \times \text{add_level_set}$$

$$\sum |V_i| = 23\,304$$

$$\sum nz'_i = 973\,038$$



RAEFSKY3 — adding 5 partial level sets



$$n = 21\,200$$

$$nnz = 1\,488\,768$$

$$\sum |W_i| = 21\,200$$

$$\sum nz_i = 864\,540$$

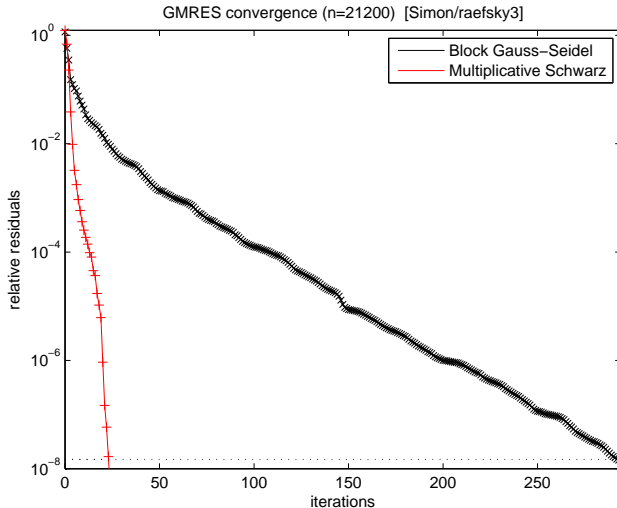
$$5 \times \text{add_level_set}$$

$$\sum |V_i| = 24\,765$$

$$\sum nz'_i = 1\,053\,087$$



RAEFSKY3 — adding 10 partial level sets



$$n = 21\,200$$

$$nnz = 1\,488\,768$$

$$\sum |W_i| = 21\,200$$

$$\sum nz_i = 864\,540$$

$$10 \times \text{add_level_set}$$

$$\sum |V_i| = 28\,615$$

$$\sum nz'_i = 1\,261\,235$$



Conclusions

- ▶ Partitioning and permuting matrices can lead to very effective block Gauss-Seidel preconditioners
- ▶ You can improve on block Gauss-Seidel preconditioning by adding overlap
- ▶ Selecting the “right” nodes for overlap is important
- ▶ Open Problem: How to order the blocks to improve performance?

Papers and Codes in:

<http://www-ai.math.uni-wuppertal.de/~dfritzsche/>



Thank you!