Parallel sparse linear solvers and applications in CFD

Jocelyne Erhel
Joint work with Désiré Nuentsa Wakam (GMRES)
and Baptiste Poirriez (PCG)

SAGE team, Inria Rennes, France

journée Calcul Intensif Distribué dans l’Industrie,
Université Paris 13, 22 janvier 2014
Outline

Solver interface
- Interface to direct and iterative solvers: MUMPS, SuperLU_Dist, Hypre, Petsc, pArms, etc
- SLSI [Nuentes Wakam et al 2010] available on demand
- System solver in H2OLab platform [Erhel et al 2009]
- Application to CFD problems

GMRES(m): a Krylov method for general matrices
- combining Domain Decomposition and deflation

PCG: a Krylov method for SPD matrices
- combining Domain Decomposition and deflation
Preconditioned GMRES

\[ Ax = b, \quad A \in \mathbb{R}^{n \times n}, \quad x, b \in \mathbb{R}^n \quad B = AM^{-1} \]

**GMRES(m): a Krylov subspace method**

- Fix \( x_0 \), then \( r_0 = b - Ax_0 \)
- \( \mathcal{K}_m(B, r_0) = \text{span}\{r_0, Br_0, \ldots B^{m-1}r_0\} \)
- Find \( x_m \in x_0 + \mathcal{K}_m(B, r_0) \) such that \( \|r_m\|_2 = \|b - Bx_m\|_2 = \min_{u \in x_0 + \mathcal{K}_m(B, r_0)} \|b - Bu\|_2 \)

**Building blocks of GMRES**

- Initial step: choose \( x_0 \), compute \( r_0 \)
- First step: generate an orthonormal basis \( V_{m+1} = [v_0, \ldots, v_m] \) of \( \mathcal{K}_{m+1}(B, r_0) \) such that
  \[ v_0 = r_0 / \beta, \quad \beta = \|r_0\|_2, \quad BV_m = V_{m+1}H_m \]
- Second step: approximate solution \( x_m = x_0 + M^{-1}V_m y_m \)
  \[ \Rightarrow r_m = r_0 - BV_m y_m = V_{m+1}(\beta e_1 - H_m y_m) \]
  \[ \Rightarrow y_m = \min_{y \in \mathbb{R}^m} \|\beta e_1 - H_m y\|_2 \]
GMRES ... practical issues

Arnoldi process

1: \( v_0 = r_0 / \| r_0 \|_2 \)
2: \( \text{for } k = 0, \ldots \text{ do} \)
3: \( p = B v_k \)
4: \( \text{for } i = 1 : k \text{ do} \)
5: \( h_{ik} = v_i^T p \)
6: \( p = p - h_{ik} v_i \)
7: \text{end for} \)
8: \( h_{k+1,k} = \| p \|_2 \)
9: \( v_{k+1} = p / h_{k+1,k} \)
10: \text{end for} \)

\[ BV_m = V_{m+1} \tilde{H}_m \]

Granularity issues in parallel algorithms

⇒ Communication-avoiding strategies
- Generate the basis vectors [Reichel 1990, Bai et al 1994]
- Improve the strategy [Hoemmen 2010, Demmel et al 2011]

Preconditioning issues

⇒ use multilevel methods to deal with large systems
- Filtering and Schur complement [Li et al 2003, Grigori et al 2011]

Complexity and stagnation issues with restarted GMRES(\( m \))

⇒ Use deflation to recover possible loss of information
- Deflation by augmented basis [Morgan 1995, Morgan 2002, ...]

Work in the team SAGE

Combine 'communication-avoiding' GMRES ... and Deflation ... and domain decomposition preconditioners [Nuentsa Wakam 2011, Nuentsa Wakam+Erhel+Gropp 2013, Nuentsa Wakam+Pacull 2013, Nuentsa Wakam+Erhel 2014]
GMRES ... practical issues

Granularity issues in parallel algorithms

⇒ Communication-avoiding strategies

- Generate the basis vectors [Reichel 1990, Bai et al 1994]
- Improve the strategy [Hoemmen 2010, Demmel et al 2011]

Preconditioning issues

⇒ use multilevel methods to deal with large systems

- Filtering and Schur complement [Li et al 2003, Grigori et al 2011]

Complexity and stagnation issues with restarted GMRES(m)

⇒ Use deflation to recover possible loss of information

- Deflation by augmented basis [Morgan 1995, Morgan 2002, ...]

Arnoldi process

\begin{align*}
1: & \quad v_0 = r_0 / \| r_0 \|_2 \\
2: & \quad \text{for } k = 0, \ldots \text{ do} \\
3: & \quad p = Bv_k \\
4: & \quad \text{for } i = 1 : k \text{ do} \\
5: & \quad h_{ik} = v_i^T p \\
6: & \quad p = p - h_{ik} v_i \\
7: & \quad \text{end for} \\
8: & \quad h_{k+1,k} = \| p \|_2 \\
9: & \quad v_{k+1} = p / h_{k+1,k} \\
10: & \quad \text{end for}
\end{align*}

$$BV_m = V_{m+1} H_m$$

Work in the team SAGE

Combine ‘communication-avoiding’ GMRES ... and Deflation ... and domain decomposition preconditioners [Nuentsa Wakam 2011, Nuentsa Wakam+Erhel+Gropp 2013, Nuentsa Wakam+Pacull 2013, Nuentsa Wakam+Erhel 2014]
### GMRES ... practical issues

#### Arnoldi process

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$v_0 = r_0 /</td>
</tr>
<tr>
<td>2</td>
<td>for $k = 0, \ldots$ do</td>
</tr>
<tr>
<td>3</td>
<td>$p = Bv_k$</td>
</tr>
<tr>
<td>4</td>
<td>for $i = 1 : k$ do</td>
</tr>
<tr>
<td>5</td>
<td>$h_{ik} = v_i^T p$</td>
</tr>
<tr>
<td>6</td>
<td>$p = p - h_{ik} v_i$</td>
</tr>
<tr>
<td>7</td>
<td>end for</td>
</tr>
<tr>
<td>8</td>
<td>$h_{k+1,k} =</td>
</tr>
<tr>
<td>9</td>
<td>$v_{k+1} = p / h_{k+1,k}$</td>
</tr>
<tr>
<td>10</td>
<td>end for</td>
</tr>
</tbody>
</table>

$$BV_m = V_{m+1} \bar{H}_m$$

#### Granularity issues in parallel algorithms

- **Communication-avoiding strategies**
  - Generate the basis vectors [Reichel 1990, Bai et al 1994]
  - Improve the strategy [Hoemmen 2010, Demmel et al 2011]

#### Preconditioning issues

- use multilevel methods to deal with large systems
  - Filtering and Schur complement [Li et al 2003, Grigori et al 2011]

#### Complexity and stagnation issues with restarted GMRES($m$)

- Use deflation to recover possible loss of information
  - Deflation by augmented basis [Morgan 1995, Morgan 2002,...]

#### Work in the team SAGE

Combine 'communication-avoiding' GMRES ... and Deflation ... and domain decomposition preconditioners [Nuentsa Wakam 2011, Nuentsa Wakam+Erhel+Gropp 2013, Nuentsa Wakam+Pacull 2013, Nuentsa Wakam+Erhel 2014]
GMRES ... practical issues

Granularity issues in parallel algorithms

⇒ Communication-avoiding strategies

- Generate the basis vectors [Reichel 1990, Bai et al 1994]
- Improve the strategy [Hoemmen 2010, Demmel et al 2011]

Preconditioning issues

⇒ use multilevel methods to deal with large systems

- Filtering and Schur complement [Li et al 2003, Grigori et al 2011]

Complexity and stagnation issues with restarted GMRES(m)

⇒ Use deflation to recover possible loss of information

- Deflation by augmented basis [Morgan 1995, Morgan 2002,...]

Work in the team SAGE

Combine 'communication-avoiding' GMRES ... and Deflation ... and domain decomposition preconditioners
GMRES ... practical issues

Granularity issues in parallel algorithms

⇒ Communication-avoiding strategies

○ Generate the basis vectors [Reichel 1990, Bai et al 1994]
○ Improve the strategy [Hoemmen 2010, Demmel et al 2011]

Preconditioning issues

⇒ use multilevel methods to deal with large systems

○ Filtering and Schur complement [Li et al 2003, Grigori et al 2011]
○ Multilevel parallelism [Nuentsa Wakam et al 2011, Giraud et al 2010, ...]

Complexity and stagnation issues with restarted GMRES(\(m\))

⇒ Use deflation to recover possible loss of information

○ Deflation by augmented basis [Morgan 1995, Morgan 2002,...]

Work in the team SAGE

Combine 'communication-avoiding' GMRES ... and Deflation ... and domain decomposition preconditioners [Nuentsa Wakam 2011, Nuentsa Wakam+Erhel+Gropp 2013, Nuentsa Wakam+Pacull 2013, Nuentsa Wakam+Erhel 2014]
Communication-avoiding strategy: Newton basis

**Building blocks**

- **Initial step:** run one cycle of GMRES(m) and compute shifts for the Newton basis
- **First step:** build a basis $K_{m+1} = [k_0, k_1, \ldots, k_m]$ of the Krylov subspace $K_{m+1}(B, r_0)$ such that

  $$BK_m = K_{m+1} \tilde{T}_m$$

- **Second step:** compute an orthonormal basis of $K_{m+1}(B, r_0)$
  Compute the QR factorization $K_{m+1} = V_{m+1} R_{m+1}$

  $$BK_m = V_{m+1} R_{m+1} \tilde{T}_m \Rightarrow BV_m = V_{m+1} R_{m+1} \underbrace{\tilde{T}_m R_{m}^{-1}}_{\tilde{H}_m}$$

- **Third step:** approximate solution $x_m = x_0 + M^{-1} V_m y_m$

  $$r_m = r_0 - BK_m y_m = V_{m+1} (\beta e_1 - \tilde{H}_m y_m) \quad \text{with } \beta = \|r_0\|_2$$

  $$y_m = \min_{y \in \mathbb{R}^m} \|\beta e_1 - \tilde{H}_m y\|_2$$
Communication-avoiding strategy: Newton basis

**Building blocks**

- **Initial step:** run one cycle of GMRES(m) and compute shifts for the Newton basis
- **First step:** build a basis $K_{m+1} = [k_0, k_1, \ldots, k_m]$ of the Krylov subspace $K_{m+1}(B, r_0)$ such that
  \[ BK_m = K_{m+1} \tilde{T}_m \]

- **Second step:** compute an orthonormal basis of $K_{m+1}(B, r_0)$
  Compute the QR factorization $K_{m+1} = V_{m+1} R_{m+1}$
  \[ \Rightarrow BK_m = V_{m+1} R_{m+1} \tilde{T}_m \Rightarrow BV_m = V_{m+1} \underbrace{R_{m+1} \tilde{T}_m R_{m}^{-1}}_{\tilde{H}_m} \]

- **Third step:** approximate solution $x_m = x_0 + M^{-1} V_m y_m$
  \[ \Rightarrow r_m = r_0 - BK_m y_m = V_{m+1} (\beta e_1 - \tilde{H}_m y_m) \quad \text{with} \quad \beta = \| r_0 \|_2 \]
  \[ \Rightarrow y_m = \min_{y \in \mathbb{R}^m} \| \beta e_1 - \tilde{H}_m y \|_2 \]
Communication-avoiding strategy: Newton basis

building blocks

• **Initial step:** run one cycle of GMRES(m) and compute shifts for the Newton basis

• **First step:** build a basis $K_{m+1} = [k_0, k_1, \ldots, k_m]$ of the Krylov subspace $K_{m+1}(B, r_0)$ such that

$$BK_m = K_{m+1} \tilde{T}_m$$

• **Second step:** compute an orthonormal basis of $K_{m+1}(B, r_0)$

  Compute the QR factorization $K_{m+1} = V_{m+1}R_{m+1}$


  $$\Rightarrow BK_m = V_{m+1}R_{m+1} \tilde{T}_m \Rightarrow BV_m = V_{m+1} \begin{pmatrix} R_{m+1} \tilde{T}_m R_{m}^{-1} \\ \tilde{H}_m \end{pmatrix}$$

• **Third step:** approximate solution $x_m = x_0 + M^{-1}V_m y_m$

  $$\Rightarrow r_m = r_0 - BK_m y_m = V_{m+1}(\beta e_1 - \tilde{H}_m y_m) \quad \text{with } \beta = \|r_0\|_2$$

  $$\Rightarrow y_m = \min_{y \in \mathbb{R}^m} \|\beta e_1 - \tilde{H}_m y\|_2$$
Communication-avoiding strategy: Newton basis

<table>
<thead>
<tr>
<th>building blocks</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Initial step: run one cycle of GMRES(m) and compute shifts for the Newton basis</td>
</tr>
<tr>
<td>• First step: build a basis $K_{m+1} = [k_0, k_1, \ldots, k_m]$ of the Krylov subspace $K_{m+1}(B, r_0)$ such that $BK_m = K_{m+1} \tilde{T}_m$</td>
</tr>
<tr>
<td>• Second step: compute an orthonormal basis of $K_{m+1}(B, r_0)$</td>
</tr>
<tr>
<td>Compute the QR factorization $K_{m+1} = V_{m+1}R_{m+1}$</td>
</tr>
<tr>
<td>$\Rightarrow BK_m = V_{m+1}R_{m+1}\tilde{T}<em>m \Rightarrow BV_m = V</em>{m+1} R_{m+1} \underbrace{\tilde{T}_m R^{-1}<em>m}</em>{\tilde{H}_m}$</td>
</tr>
<tr>
<td>• Third step: approximate solution $x_m = x_0 + M^{-1}V_m y_m$</td>
</tr>
<tr>
<td>$\Rightarrow r_m = r_0 - BK_m y_m = V_{m+1}(\beta e_1 - \tilde{H}_m y_m)$ with $\beta =</td>
</tr>
<tr>
<td>$\Rightarrow y_m = \min_{y \in \mathbb{R}^m}</td>
</tr>
</tbody>
</table>
GMRES combined with Domain Decomposition

Main steps with Domain Decomposition preconditioning

- Partition the weighted graph of the matrix in parallel with PARMETIS.
- Redistribute the matrix and right-hand-side according to the PARMETIS partitioning.
- Perform a parallel iterative row and column scaling on the matrix and the right-hand side vector [Amestoy et al, 2008].
- Define the overlap between the submatrices for the additive Schwarz preconditioner.

\[
M_{RAS}^{-1} = \sum_{k=1}^{D} (R_k^0)^T (A_k^\delta)^{-1} R_k^\delta
\]

- Setup the submatrices (ILU or LU factorization).
- Solve iteratively the preconditioned system using GMRES.
Deflation strategies

**Restarted GMRES(m)**

- $x_m = x_0 + M^{-1} V_m y_m$ where $y_m$ minimizes $\|r_m\|_2$
- The convergence rate depends on the spectral distribution in $B$
- Smallest eigenvalues slow down the convergence
- Deflation occurs when the Krylov subspace is large enough
- With restarting: loss of spectral information, risk of stalling

**Accelerating the restarted GMRES** [Simoncini and Szyld, 2007]

- Approximate the smallest eigenvalues and the associated invariant subspace $U_r$
- Explicit deflation technique [Erhel et al 1996; Burrage et al 1998; Moriya et al 2000]:

  $$B \bar{M}^{-1} \bar{x} = b$$

  with $\bar{M}^{-1} = (I_n + U_r(\lambda_n | T^{-1} - I_r) U_r^T$ and $T = U_r^T B U_r$

  $$x_m \in x_0 + \text{span}\{U_r\} + K_m(B, r_0)$$
Deflation strategies

### Restarted GMRES(m)

- \( x_m = x_0 + M^{-1} V_m y_m \) where \( y_m \) minimizes \( \| r_m \|_2 \)
- The convergence rate depends on the spectral distribution in \( B \)
- Smallest eigenvalues slow down the convergence
- Deflation occurs when the Krylov subspace is large enough
- With restarting: loss of spectral information, risk of stalling

### Accelerating the restarted GMRES [Simoncini and Szyld, 2007]

- Approximate the smallest eigenvalues and the associated invariant subspace \( U_r \)
- Explicit deflation technique [Erhel et al 1996; Burrage et al 1998; Moriya et al 2000]:
  \[
  B \bar{M}^{-1} \tilde{x} = b
  \]
  with \( \bar{M}^{-1} = (I_n + U_r |\lambda_n| T^{-1} - I_r) U_r^T \) and \( T = U_r^T B U_r \)
  \[
  x_m \in x_0 + \text{span}\{U_r\} + \mathcal{K}_m(B, r_0)
  \]
**DGMRES: GMRES with adaptive preconditioning deflation**

**DGMRES(m, r)**

- Perform one cycle of restarted GMRES(m) and compute shifts for the Newton basis
- Compute $U_r$, a basis of a coarse subspace
- Build $\bar{M}_r^{-1} \equiv I_n + U_r(\lambda_n |T^{-1} - I_r)U_r^T$, $T = U_r^TBU_r$
- Apply GMRES(m) to $B\bar{M}_r^{-1}$
- At each restart, update $r$ and the basis $U_r$

**Adaptive DGMRES(m,r)**

- Detect a potential slow convergence [Sosonkina et al 1998]
- Switch to DGMRES(m,r) only if necessary [Nuentesa Wakam et al 2013]
DGMRES: GMRES with adaptive preconditioning deflation

DGMRES(m, r)
- Perform one cycle of restarted GMRES(m) and compute shifts for the Newton basis
- Compute $U_r$, a basis of a coarse subspace
- Build $\tilde{M}_r^{-1} \equiv I_n + U_r(|\lambda_n|T^{-1} - I_r)U_r^T$, $T = U_r^TBU_r$
- Apply GMRES(m) to $B\tilde{M}_r^{-1}$
- At each restart, update $r$ and the basis $U_r$

Adaptive DGMRES(m,r)
- Detect a potential slow convergence [Sosonkina et al 1998]
- Switch to DGMRES(m,r) only if necessary [Nuentsa Wakam et al 2013]
DGMRES: GMRES with adaptive preconditioning deflation

DGMRES(m, r)

- Perform one cycle of restarted GMRES(m) and compute shifts for the Newton basis
- Compute $U_r$, a basis of a coarse subspace
- Build $\tilde{M}_r^{-1} \equiv I_n + U_r(|\lambda_n|^{-1} - I_r)U_r^T$, $T = U_r^TBU_r$
- Apply GMRES(m) to $B\tilde{M}_r^{-1}$
- At each restart, update $r$ and the basis $U_r$

Adaptive DGMRES(m,r)

- Detect a potential slow convergence [Sosonkina et al 1998]
- Switch to DGMRES(m,r) only if necessary [Nunteersa Wakam et al 2013]
Implementation of Deflated GMRES in PETSc

New KSP type: DGMRES

Usage in Petsc

- Available in PETSc
- Use DGMRES just as any other KSP with the following options
Experiments with CFD matrices

**FLUOREM matrices**
- in MatrixMarket collection
- large, sparse, nonsymmetric matrices
- linearization of Navier-Stokes: symmetric profile with structured blocks
- Schwarz preconditioning combined with GMRES or DGMRES

[Nuentsa Wakam+Erhel+Gropp 2013; Nuentsa Wakam+Pacull 2013]

RM07R n= 381,689; nnz=37,464,962

HV15R n= 2,017,169; nnz=283,073,458
Combining DGMRES with domain decomposition

### CPU time on parallel computers

<table>
<thead>
<tr>
<th></th>
<th>RM07R, n = 381,689, nz = 37,464,962</th>
<th>HV15R, n = 2,017,169, nz = 283,073,458</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>FULL-GMRES</td>
<td>GMRES(48)</td>
</tr>
<tr>
<td></td>
<td>ITS Time (s)</td>
<td>ITS Time (s)</td>
</tr>
<tr>
<td>16</td>
<td>92   214</td>
<td>169   297</td>
</tr>
<tr>
<td>32</td>
<td>117  103</td>
<td>355   260</td>
</tr>
<tr>
<td>64</td>
<td>149   66</td>
<td>860   166</td>
</tr>
</tbody>
</table>
AGMRES: deflation with an adaptive augmented basis

Building blocks

- **Initial step:** run one cycle of GMRES(m) and compute shifts for the Newton basis
  
  **Compute** $U_r = [u_0, u_1, \ldots, u_{r-1}]$ a basis of a coarse subspace

- **First step:** build a basis $K_{m+1} = [k_0, k_1, \ldots, k_m]$ of the Krylov subspace $\mathcal{K}_{m+1}(B, r_0)$ such that
  
  $$BK_m = K_{m+1} \tilde{T}_m$$

  Define the augmented subspace $C_s = \mathcal{K}_m(B, r_0) + \text{span}\{U_r\}$ with $s = m + r$ with the basis
  
  $$[ K_m \quad U_r ]$$

  compute

  $$BU_r = \hat{K}_r D_r$$

  Define the augmented subspace $\hat{C}_{s+1} = \mathcal{K}_{m+1}(B, r_0) + \text{span}\{BU_r\}$ with the basis

  $$[ K_{m+1} \quad \hat{K}_r ]$$
Building blocks

- **Initial step:** run one cycle of GMRES(m) and compute shifts for the Newton basis
  
  Compute $U_r = [u_0, u_1, \ldots, u_{r-1}]$ a basis of a coarse subspace

- **First step:** build a basis $K_{m+1} = [k_0, k_1, \ldots, k_m]$ of the Krylov subspace $\mathcal{K}_{m+1}(B, r_0)$ such that

$$BK_m = K_{m+1} \tilde{T}_m$$

Define the augmented subspace $C_s = \mathcal{K}_m(B, r_0) + \text{span}\{U_r\}$ with $s = m + r$ with the basis

$$[ \begin{array}{cc} K_m & U_r \end{array} ]$$

compute

$$BU_r = \hat{K}_r D_r$$

Define the augmented subspace $\hat{C}_{s+1} = \mathcal{K}_{m+1}(B, r_0) + \text{span}\{BU_r\}$ with the basis

$$[ \begin{array}{cc} K_{m+1} & \hat{K}_r \end{array} ]$$
AGMRES: deflation with an adaptive augmented basis

Building blocks

- **Second step:** Compute an orthonormal basis of $\hat{C}_{s+1}$
  
  QR factorize the augmented basis $[ K_{m+1} \quad \hat{K}_r ] = V_{s+1} R_{s+1}$
  
  $$\Rightarrow BK_m = V_{m+1} R_{m+1} \bar{T}_m \Rightarrow BV_m = V_{m+1} R_{m+1} \bar{T}_m R_{m-1}^{-1}$$
  
  $$\Rightarrow BU_r = (V_{m+1} R_{m+1, r} + V_r R_r) D_r$$

  Define the basis $W_s = [ V_m \quad U_r ]$
  
  $$\Rightarrow BW_s = V_{s+1} \bar{H}_s$$

- **Third step:** $x_s = x_0 + M^{-1} W_s y_s$
  
  $$\Rightarrow r_s = r_0 - BW_s y_s = V_{s+1} (\beta e_1 - \bar{H}_s y_s) \quad \text{and} \quad \beta = ||r_0||_2$$

  $$y_s = \min_{y \in \mathbb{R}^s} ||\beta e_1 - \bar{H}_s y||_2$$

- **Final step:** Adaptively update $r$ and the coarse basis $U_r$
AGMRES: deflation with an adaptive augmented basis

Building blocks

- Second step: Compute an orthonormal basis of $\hat{C}_{s+1}$
  QR factorize the augmented basis $[K_{m+1} \quad \hat{K}_r] = V_{s+1}R_{s+1}$

  $BK_m = V_{m+1}R_{m+1}\bar{T}_m \Rightarrow BV_m = V_{m+1}R_{m+1}\bar{T}_mR_m^{-1}$

  $BU_r = (V_{m+1}R_{m+1,r} + V_rR_r)D_r$

- Define the basis $W_s = [V_m \quad U_r]$

  $BW_s = V_{s+1}\bar{H}_s$

- Third step: $x_s = x_0 + M^{-1}W_s y_s$

  $r_s = r_0 - BW_s y_s = V_{s+1}(\beta e_1 - \bar{H}_s y_s)$ and $\beta = \|r_0\|_2$

  $y_s = min_{y \in \mathbb{R}^s} \|\beta e_1 - \bar{H}_s y\|_2$

- Final step: Adaptively update $r$ and the coarse basis $U_r$
AGMRES: deflation with an adaptive augmented basis

**Building blocks**

- **Second step**: Compute an orthonormal basis of $\hat{C}_{s+1}$
  - QR factorize the augmented basis $[K_{m+1} \quad \hat{K}_r] = V_{s+1} R_{s+1}
  
  $BK_m = V_{m+1} R_{m+1} \hat{T}_m \Rightarrow BV_m = V_{m+1} R_{m+1} \hat{T}_m R_m^{-1}$
  
  $BU_r = (V_{m+1} R_{m+1}, r + V_r R_r) D_r$

- Define the basis $W_s = [V_m \quad U_r]$
  
  $BW_s = V_{s+1} \bar{H}_s$

- **Third step**: $x_s = x_0 + M^{-1} W_s y_s$
  
  $\Rightarrow r_s = r_0 - BW_s y_s = V_{s+1} (\beta e_1 - \bar{H}_s y_s)$ and $\beta = \|r_0\|_2$
  
  $y_s = \min_{y \in \mathbb{R}^s} \|\beta e_1 - \bar{H}_s y\|_2$

- **Final step**: Adaptively update $r$ and the coarse basis $U_r$
New KSP type: AGMRES

Usage in Petsc

- Use AGMRES just as GMRES
  ⇒ KSPSetType(ksp, KSPAGMRES) or -ksp_type agmres, -pc_type asm, ...
- Options: -ksp_gmres_restart m, -ksp_agmres_eig r,
- -ksp_max_its maxits, -ksp_agmres_smv smv -ksp_agmres_bgv bgv, ...

AGMRES in PETSc
Experiments with Augmented GMRES

**RM07R: effect of the restarting**

![Graph showing the effect of restarting on the number of iterations]

**RM07R: effect of the number of subdomains**

![Graph showing the effect of the number of subdomains on the number of iterations]

**CPU time on parallel computers**

<table>
<thead>
<tr>
<th>D</th>
<th>GMRES(32)</th>
<th>AGMRES(32,r)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ITS</td>
<td>Time (s)</td>
</tr>
<tr>
<td>16</td>
<td>254</td>
<td>379.3</td>
</tr>
<tr>
<td>32</td>
<td>886</td>
<td>573.4</td>
</tr>
<tr>
<td>64</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

RM07R, n = 381,689, nz = 37,464,962
Parallel CPU Time with AGMRES

Convection-Diffusion test cases

- 3DCONSKY_121: size = 1,771,561; nonzeros = 50,178,241
- 3DCOSKY_161: size = 4,173,281; nonzeros = 118,645,121

[Nuentesa Wakam+Erhel 2014]
Augmented Conjugate Gradient

PCG

- A Symmetric Positive Definite (SPD) matrix
- PCG is a Krylov method
- short recurrences and minimization properties
- preconditioning $M^{-1}$

Coarse grid and balancing

[Nicolaides 1987, Mandel 1993, DD proceedings, Giraud et al.]

- $Z$ basis of a coarse subspace
- restriction of $A$: nonsingular small matrix $A_c = Z^T AZ$
- projections: $P = I - A Z A_c^{-1} Z^T$ and $P^T = I - Z A_c^{-1} (AZ)^T$
- coarse grid: $Z A_c^{-1} Z^T$
- balancing: $C_b = P^T M^{-1} P + Z A_c^{-1} Z^T$

Coarse grid and augmented CG


- $x_0 = Z A_c^{-1} Z^T b$
- $C_a = P^T M^{-1}$
- $C_a$ is equivalent to $C_b$ (if no loss of orthogonality)
SIDNUR: AugCG and Domain Decomposition

Balancing Neumann Neumann

- PCG applied to a Schur complement
- Neumann-Neumann preconditioning $M^{-1}$
- Balancing with a coarse grid $Z$

SIDNUR

[Poirriez 2011, Pichot et al. 2014]

- Domain decomposition provided by the user
- Coarse grid: signature of subdomains [Frank and Vuik 2001]
- C++ library
- Mutual factorization of local Schur complements and local matrices
- Management of floating subdomains
- Numerical experiments with 3D fracture networks
Experiments with SIDNUR

Flow computations in Discrete Fracture Networks

- random domain generated with MPFRAC software
- SPD sparse matrix
- solving with SIDNUR

[Poirriez 2011]

Number of iterations

CPU time

- [Schur]

- [Schur]
Comparing SIDNUR with other solvers

- UMFPACK: direct solver
- Boomer-AMG: algebraic multigrid
- PCG: Conjugate Gradient preconditioned by Boomer-AMG

[Poirriez 2011]
## Conclusion

### GMRES
- DGMRES KSP module: deflation in GMRES(m) with or without Newton basis
- AGMRES KSP module: augmented Newton basis in GMRES(m)
- Deflation combined with Schwarz domain decomposition preconditioning
- Robustness: reduce the restarting effects and the domain decomposition effects
- Efficiency: increase granularity and scalability
- Numerical experiments with CFD problems: DGMRES and AGMRES faster than GMRES

### PCG
- Deflation combined with Schur domain decomposition
- SIDNUR: Balancing Domain Decomposition
- Robustness: reduce the domain decomposition effects
- Efficiency: parallel Schur and Neumann Neumann computations
- Numerical experiments with 3D fracture networks: faster than multigrid and PCG
- Library soon available as free software