A Parallel Augmented GMRES algorithm

Application to design optimization in CFD

Jocelyne Erhel and Désiré NUENTSA WAKAM

SAGE team, Inria Rennes, France

3rd Dolomites Workshop on Constructive Approximation and Applications (DWCAA12), September 9-14, 2012

Preconditioned GMRES

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

GMRES(m): a Krylov subspace method

- [Saad and Schultz 1986, Meurant's book 1999, Saad's book 2003, Simoncini and Szyld 2007, Erhel 2011, ...]
- Fix x_0, then $r_0 = b - Ax_0$
- $\mathcal{K}_m(B, r_0) = span\{r_0, Br_0, \ldots B^{m-1} r_0\}$
- 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: generation of an orthonormal basis $V_{m+1} = [v_0, \ldots, v_m]$ of $\mathcal{K}_{m+1}(B, r_0)$ such that

$$v_0 = r_0/\beta, \quad \beta = \|r_0\|, \quad BV_m = V_{m+1}\bar{H}_m$$

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

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

$$\Rightarrow y_m = min_{y \in \mathbb{R}^m} \|\beta e_1 - \bar{H}_m y\|_2$$

GMRES ... practical issues

Arnoldi process

```
v_0 = r_0/||r_0||_2
for k = 0, ... do
    p = B v_k
    for i = 1 : k do
        h_ik = v_i^T p
        p = p - h_ik v_i
    end for
    h_{k+1,k} = ||p||_2
    v_{k+1} = p/h_{k+1,k}
end for
```

$\Downarrow$

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

Granularity issues in parallel algorithms

$\Rightarrow$ Communication-avoiding strategies

- Generate the basis vectors [Reichel 1990, Bai et al 1994]
- Orthogonalize the basis [De Sturler 1994, Erhel 1995, Sidje 1997]
- Improve the strategy [Hoemmen 2010, Demmel et al 2011]

Complexity issues with restarted GMRES(m)

$\Rightarrow$ Use deflation to recover possible loss of information

- Deflation by preconditioning [Erhel et al 1996, Burrage et al 1998, Baglama et al 1998, ...]
- Deflation by augmented basis [Morgan 1995, Morgan 2002,...]

Preconditioning issues

$\Rightarrow$ use multilevel methods to deal with large systems

- Schwarz preconditioning [Atenekeng Kahou et al 2007, Dufaud+Tromeur-Dervout 2010, Giraud+Haidar 2009, Smith et al's book 1996,...]
- Filtering and Schur complement [Li et al 2003, Grigori et al 2011]
- Multilevel parallelism [Nuentsa Wakam et al 2011, Giraud et al 2010, ...]

Proposal of this work

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

GMRES ... practical issues

Arnoldi process

```
v_0 = r_0/||r_0||_2
for k = 0, ... do
    p = B v_k
    for i = 1 : k do
        h_ik = v_i^T p
        p = p - h_ik v_i
    end for
    h_{k+1,k} = ||p||_2
    v_{k+1} = p / h_{k+1,k}
end for
```

⇓

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

Granularity issues in parallel algorithms

⇒ Communication-avoiding strategies

- Generate the basis vectors [Reichel 1990, Bai et al 1994]
- Orthogonalize the basis [De Sturler 1994, Erhel 1995, Sidje 1997]
- Improve the strategy [Hoemmen 2010, Demmel et al 2011]

Complexity issues with restarted GMRES(m)

⇒ Use deflation to recover possible loss of information

- Deflation by preconditioning [Erhel et al 1996, Burrage et al 1998, Baglama et al 1998, ...]
- Deflation by augmented basis [Morgan 1995, Morgan 2002,...]

Preconditioning issues

⇒ use multilevel methods to deal with large systems

- Schwarz preconditioning [Atenekeng Kahou et al 2007, Dufaud+Tromeur-Dervout 2010, Giraud+Haidar 2009, Smith et al's book 1996,...]
- Filtering and Schur complement [Li et al 2003, Grigori et al 2011]
- Multilevel parallelism [Nuentsa Wakam et al 2011, Giraud et al 2010, ...]

Proposal of this work

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

GMRES ... practical issues

Arnoldi process

$v_0 = r_0/\|r_0\|_2$
for $k = 0, \dots$ **do**
  $p = Bv_k$
  for $i = 1 : k$ **do**
    $h_{ik} = v_i^T p$
    $p = p - h_{ik} v_i$
  end for
  $h_{k+1,k} = ||p||_2$
  $v_{k+1} = p/h_{k+1,k}$
end for

$\Downarrow$

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

Granularity issues in parallel algorithms

$\Rightarrow$ Communication-avoiding strategies

- Generate the basis vectors [Reichel 1990, Bai et al 1994]
- Orthogonalize the basis [De Sturler 1994, Erhel 1995, Sidje 1997]
- Improve the strategy [Hoemmen 2010, Demmel et al 2011]

Complexity issues with restarted GMRES(m)

$\Rightarrow$ Use deflation to recover possible loss of information

- Deflation by preconditioning [Erhel et al 1996, Burrage et al 1998, Baglama et al 1998, ...]
- Deflation by augmented basis [Morgan 1995, Morgan 2002,...]

Preconditioning issues

$\Rightarrow$ use multilevel methods to deal with large systems

- Schwarz preconditioning [Atenekeng Kahou et al 2007, Dufaud+Tromeur-Dervout 2010, Giraud+Haidar 2009, Smith et al's book 1996,...]
- Filtering and Schur complement [Li et al 2003, Grigori et al 2011]
- Multilevel parallelism [Nuentsa Wakam et al 2011, Giraud et al 2010, ...]

Proposal of this work

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

GMRES ... practical issues

Arnoldi process

```
v_0 = r_0/||r_0||_2
for k = 0, ... do
    p = B v_k
    for i = 1 : k do
        h_ik = v_i^T p
        p = p − h_ik v_i
    end for
    h_{k+1,k} = ||p||_2
    v_{k+1} = p/h_{k+1,k}
end for
```

⇓

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

Granularity issues in parallel algorithms

⇒ Communication-avoiding strategies

- Generate the basis vectors [Reichel 1990, Bai et al 1994]
- Orthogonalize the basis [De Sturler 1994, Erhel 1995, Sidje 1997]
- Improve the strategy [Hoemmen 2010, Demmel et al 2011]

Complexity issues with restarted GMRES(m)

⇒ Use deflation to recover possible loss of information

- Deflation by preconditioning [Erhel et al 1996, Burrage et al 1998, Baglama et al 1998, ...]
- Deflation by augmented basis [Morgan 1995, Morgan 2002,...]

Preconditioning issues

⇒ use multilevel methods to deal with large systems

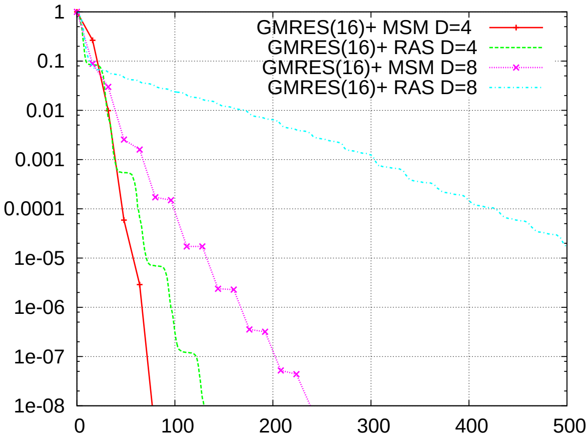
- Schwarz preconditioning [Atenekeng Kahou et al 2007, Dufaud+Tromeur-Dervout 2010, Giraud+Haidar 2009, Smith et al's book 1996,...]
- Filtering and Schur complement [Li et al 2003, Grigori et al 2011]
- Multilevel parallelism [Nuentsa Wakam et al 2011, Giraud et al 2010, ...]

Proposal of this work

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

Illustration : Domain decomposition and Restarting

- 2D Helmholtz problem on a 164×164 grid
- GMRES(16) + RAS or MSM (4 or 8 domains), LU as subdomain solver



Outline

1. Newton basis
2. Adaptive deflation
3. Numerical experiments
   - Software package
   - CFD application
   - Numerical convergence
   - CPU Time

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 $\mathcal{K}_{m+1}(B, r_0)$ such that

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

- Second step: compute an orthonormal basis of $\mathcal{K}_{m+1}(B, r_0)$
  Compute the QR factorization $K_{m+1} = V_{m+1}R_{m+1}$
  RODDEC [Sidje 1997, Erhel 1995] or TSQR [Demmel et al 2011]

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

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

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

$$\Rightarrow y_m = min_{y \in \mathbb{R}^m} \|\beta e_1 - \bar{H}_m y\|_2$$

GMRES with a 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 $\mathcal{K}_{m+1}(B, r_0)$ such that

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

- Second step: compute an orthonormal basis of $\mathcal{K}_{m+1}(B, r_0)$
  Compute the QR factorization $K_{m+1} = V_{m+1}R_{m+1}$
  RODDEC [Sidje 1997, Erhel 1995] or TSQR [Demmel et al 2011]

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

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

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

$$\Rightarrow y_m = min_{y \in \mathbb{R}^m} \|\beta e_1 - \bar{H}_m y\|_2$$

GMRES with a 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 $\mathcal{K}_{m+1}(B, r_0)$ such that

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

- Second step: compute an orthonormal basis of $\mathcal{K}_{m+1}(B, r_0)$
  Compute the QR factorization $K_{m+1} = V_{m+1}R_{m+1}$
  RODDEC [Sidje 1997, Erhel 1995] or TSQR [Demmel et al 2011]

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

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

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

$$\Rightarrow y_m = min_{y\in\mathbb{R}^m}\|\beta e_1 - \bar{H}_m y\|_2$$

GMRES with a 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 $\mathcal{K}_{m+1}(B, r_0)$ such that

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

- Second step: compute an orthonormal basis of $\mathcal{K}_{m+1}(B, r_0)$
  Compute the QR factorization $K_{m+1} = V_{m+1}R_{m+1}$
  RODDEC [Sidje 1997, Erhel 1995] or TSQR [Demmel et al 2011]

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

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

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

$$\Rightarrow y_m = min_{y \in \mathbb{R}^m} \|\beta e_1 - \bar{H}_m y\|_2$$

Computation of the Newton basis

Shifts and matrix-vector products

- Initial step: Compute m Ritz values $\lambda_j \quad j = 0, \ldots m-1$ and get their Leja ordering
  Alternative [Philippe+Reichel, 2011]
- At each cycle:

  $tmp = r_0$
  $\sigma_0 = \|tmp\|_2$
  $k_0 = tmp/\sigma_0$
  $j = 0$
  while $j \leq m-1$ **do**
      if $Im(\lambda_{j+1}) = 0$ **then**
          $tmp = (B - \lambda_{j+1} I)k_j$
          $\sigma_{j+1} = \|tmp\|_2$
          $k_{j+1} = tmp/\sigma_{j+1}$
          $j = j+1$
      else if $Im(\lambda_{j+1} > 0)$ **then**
          $tmp = (B - Re(\lambda_{j+1})I)k_j$
          $\sigma_{j+1} = \|tmp\|_2$
          $k_{j+1} = tmp/\sigma_{j+1}$
          $tmp = (B - Re(\lambda_{j+1})I)tmp + Im(\lambda_{j+1})^2 k_j$
          $\sigma_{j+2} = \|tmp\|_2$
          $k_{j+2} = tmp/\sigma_{j+2}$
          $j = j+2$
      end if
  end while

- Arnolidi-like relation $BK_m = K_{m+1}\bar{T}_m$

Computation of the Newton basis

Shifts and matrix-vector products

- Initial step: Compute m Ritz values $\lambda_j \quad j = 0, \ldots m-1$ and get their Leja ordering
  Alternative [Philippe+Reichel, 2011]
- At each cycle:

$tmp = r_0$
$\sigma_0 = \|tmp\|_2$
$k_0 = tmp/\sigma_0$
$j = 0$
while $j \leq m-1$ **do**
  if $Im(\lambda_{j+1}) = 0$ **then**
    $tmp = (B - \lambda_{j+1} I)k_j$
    $\sigma_{j+1} = \|tmp\|_2$
    $k_{j+1} = tmp/\sigma_{j+1}$
    $j = j+1$
  else if $Im(\lambda_{j+1} > 0)$ **then**
    $tmp = (B - Re(\lambda_{j+1})I)k_j$
    $\sigma_{j+1} = \|tmp\|_2$
    $k_{j+1} = tmp/\sigma_{j+1}$
    $tmp = (B - Re(\lambda_{j+1})I)tmp + Im(\lambda_{j+1})^2 k_j$
    $\sigma_{j+2} = \|tmp\|_2$
    $k_{j+2} = tmp/\sigma_{j+2}$
    $j = j+2$
  end if
end while

- Arnolidi-like relation $BK_m = K_{m+1}\bar{T}_m$

Computation of the Newton basis

Shifts and matrix-vector products

- Initial step: Compute m Ritz values $\lambda_j \quad j = 0, \ldots m-1$ and get their Leja ordering
  Alternative [Philippe+Reichel, 2011]
- At each cycle:

  $tmp = r_0$
  $\sigma_0 = \|tmp\|_2$
  $k_0 = tmp/\sigma_0$
  $j = 0$
  while $j \leq m-1$ **do**
      if $Im(\lambda_{j+1}) = 0$ **then**
          $tmp = (B - \lambda_{j+1} I)k_j$
          $\sigma_{j+1} = \|tmp\|_2$
          $k_{j+1} = tmp/\sigma_{j+1}$
          $j = j+1$
      else if $Im(\lambda_{j+1} > 0)$ **then**
          $tmp = (B - Re(\lambda_{j+1})I)k_j$
          $\sigma_{j+1} = \|tmp\|_2$
          $k_{j+1} = tmp/\sigma_{j+1}$
          $tmp = (B - Re(\lambda_{j+1})I)tmp + Im(\lambda_{j+1})^2 k_j$
          $\sigma_{j+2} = \|tmp\|_2$
          $k_{j+2} = tmp/\sigma_{j+2}$
          $j = j+2$
      end if
  end while

- Arnolidi-like relation $BK_m = K_{m+1}\bar{T}_m$

Building blocks

- Initial step: run one cycle of GMRES(m) and compute shifts for the Newton basis
  Compute $U_r = [u_0, u_1, \dots, u_{r-1}]$ a basis of a coarse subspace
- First step: build a basis $K_{m+1} = [k_0, k_1, \dots, k_m]$ of the Krylov subspace $\mathcal{K}_{m+1}(B, r_0)$ such that

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

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

$$\begin{bmatrix} K_m & U_r \end{bmatrix}$$

- compute

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

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

$$\begin{bmatrix} K_{m+1} & \hat{K}_r \end{bmatrix}$$

GMRES with an adaptive augmented Newton 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

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

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

$$\begin{bmatrix} K_m & U_r \end{bmatrix}$$

- compute

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

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

$$\begin{bmatrix} K_{m+1} & \hat{K}_r \end{bmatrix}$$

GMRES with an adaptive augmented Newton 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

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

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

$$\begin{bmatrix} K_m & U_r \end{bmatrix}$$

- compute

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

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

$$\begin{bmatrix} K_{m+1} & \hat{K}_r \end{bmatrix}$$

Building blocks

- Second step: Compute an orthonormal basis of $\hat{\mathcal{C}}_{s+1}$
  QR factorize the augmented basis $\begin{bmatrix} K_{m+1} & \hat{K}_r \end{bmatrix} = V_{s+1} R_{s+1}$

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

$$\Rightarrow BU_r = (V_{m+1} R_{m+1,r} + V_r R_r) D_r$$

  Define the basis $W_s = \begin{bmatrix} V_m & U_r \end{bmatrix}$

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

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

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

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

- Final step: Adaptively update r and the coarse basis U_r

GMRES with an adaptive augmented Newton basis

Building blocks

- Second step: Compute an orthonormal basis of $\hat{\mathcal{C}}_{s+1}$
  QR factorize the augmented basis $\begin{bmatrix} K_{m+1} & \hat{K}_r \end{bmatrix} = V_{s+1} R_{s+1}$

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

$$\Rightarrow BU_r = (V_{m+1} R_{m+1,r} + V_r R_r) D_r$$

  Define the basis $W_s = \begin{bmatrix} V_m & U_r \end{bmatrix}$

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

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

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

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

- Final step: Adaptively update r and the coarse basis U_r

GMRES with an adaptive augmented Newton basis

Building blocks

- Second step: Compute an orthonormal basis of $\hat{\mathcal{C}}_{s+1}$
  QR factorize the augmented basis $\begin{bmatrix} K_{m+1} & \hat{K}_r \end{bmatrix} = V_{s+1} R_{s+1}$

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

$$\Rightarrow BU_r = (V_{m+1} R_{m+1,r} + V_r R_r) D_r$$

  Define the basis $W_s = \begin{bmatrix} V_m & U_r \end{bmatrix}$

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

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

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

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

- Final step: Adaptively update r and the coarse basis U_r

Adaptive strategy

Update r and U_r using convergence estimation

- At each restart, estimate *Iter*, the remaining number of steps [Sosonkina et al, 1998]

$$Iter = s * log(\frac{\epsilon}{||r_s||})/log(\frac{||r_s||}{||r_0||});$$

- If ($Iter \leq smv * itmax$): fast convergence $\Rightarrow$ keep U_r
- If ($smv * itmax < Iter \leq bgv * itmax$): slow convergence $\Rightarrow$ update U_r
- If ($Iter > bgv * itmax$): possible stagnation $\Rightarrow$ Increase r by l until r_{max} and update U_r

Coarse subspace

Approximate invariant subspace

- Initial step: Compute m Ritz values $\lambda_j \quad j = 0, \ldots m-1$ and get their Leja ordering extract r Ritz vectors $u_j, \quad j = 0, \ldots r-1$ with $u_j = V_m g_j$ in the subspace $\mathcal{K}_m$ using the Galerkin condition $V_m^T(B - \lambda_j I)V_m g_j = 0$

$$\Rightarrow H_m g_j = \lambda_j g_j$$

- At each cycle: update U_r with $u_j = W_s g_j$ in the augmented subspace $\mathcal{C}_s$ using the Galerkin condition $(BW_s)^T(B - \lambda_j I)W_s g_j = 0$

$$\Rightarrow \bar{H}_s^T \bar{H}_s g_j = \lambda_j \bar{H}_s^T V_{s+1}^T W_s g_j$$

with

$$V_{s+1}^T W_s = \begin{bmatrix} V_{s+1}^T V_m & V_{s+1}^T U_r \end{bmatrix}$$

$$V_{s+1}^T V_m = \begin{bmatrix} I_m \\ 0 \end{bmatrix}$$

Coarse subspace

Approximate invariant subspace

- Initial step: Compute m Ritz values $\lambda_j \quad j = 0, \ldots m-1$ and get their Leja ordering extract r Ritz vectors $u_j, \quad j = 0, \ldots r-1$ with $u_j = V_m g_j$ in the subspace $\mathcal{K}_m$ using the Galerkin condition $V_m^T (B - \lambda_j I) V_m g_j = 0$

$$\Rightarrow H_m g_j = \lambda_j g_j$$

- At each cycle: update U_r with $u_j = W_s g_j$ in the augmented subspace $\mathcal{C}_s$ using the Galerkin condition $(BW_s)^T (B - \lambda_j I) W_s g_j = 0$

$$\Rightarrow \bar{H}_s^T \bar{H}_s g_j = \lambda_j \bar{H}_s^T V_{s+1}^T W_s g_j$$

with

$$V_{s+1}^T W_s = \begin{bmatrix} V_{s+1}^T V_m & V_{s+1}^T U_r \end{bmatrix}$$

$$V_{s+1}^T V_m = \begin{bmatrix} I_m \\ 0 \end{bmatrix}$$

AGMRES algorithm

AGMRES(m,itmax, r, l, rmax)

B, $r_0 = b/\|b\|_2$
Compute one cycle of Arnoldi-GMRES
Generate m shifts λ_j
Compute r vectors U_r
while no convergence **do**
    Compute K_{m+1} such that $BK_m = K_{m+1}\bar{T}_m$
    Compute $BU_r = \hat{K}_r D_r$
    Orthogonolize $\begin{bmatrix} K_{m+1} & \hat{K}_r \end{bmatrix} = V_{s+1}R_{s+1}$
    Define $W_s = \begin{bmatrix} V_m & U_r \end{bmatrix}$
    Get $BW_s = V_{s+1}\bar{H}_s$
    solve $y_s = \min_y J(y)$ with $J(y) = ||\beta e_1 - \bar{H}_s y||_2$
    Compute $x_s = x_0 + W_s y_s$
    Test of convergence
    Adaptively udpate r and U_r
end while

Complexity issues: comparison with GMRES(m)

- memory additional requirements: $2r$ vectors U_r and $[v_{m+1} \dots v_{m+r}]$
- CPU additional requirements in GMRES process: BU_r and $[v_{m+1} \dots v_{m+r}]$
- CPU overhead in adaptive strategy: $V_{s+1}^T U_r$

AGMRES algorithm

AGMRES(m,itmax, r, l, rmax)

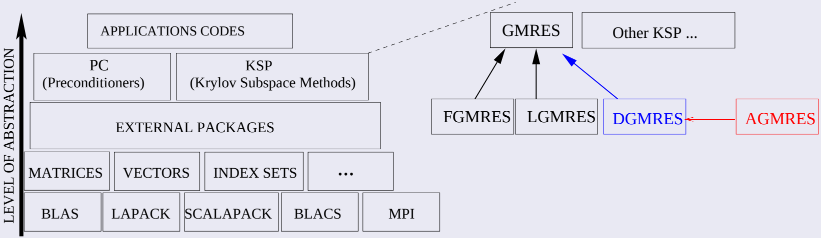
```
B, r_0 = b/||b||_2
Compute one cycle of Arnoldi-GMRES
Generate m shifts λ_j
Compute r vectors U_r
while no convergence do
    Compute K_{m+1} such that B K_m = K_{m+1} T̄_m
    Compute B U_r = K̂_r D_r
    Orthogonolize [ K_{m+1}  K̂_r ] = V_{s+1} R_{s+1}
    Define W_s = [ V_m  U_r ]
    Get B W_s = V_{s+1} H̄_s
    solve y_s = min_y J(y) with J(y) = ||β e_1 − H̄_s y||_2
    Compute x_s = x_0 + W_s y_s
    Test of convergence
    Adaptively udpate r and U_r
end while
```

Complexity issues: comparison with GMRES(m)

- memory additional requirements: $2r$ vectors U_r and $[v_{m+1} \dots v_{m+r}]$
- CPU additional requirements in GMRES process: BU_r and $[v_{m+1} \dots v_{m+r}]$
- CPU overhead in adaptive strategy: $V_{s+1}^T U_r$

Implementation using PETSc

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, ...

Using AGMRES with Schwarz preconditioning

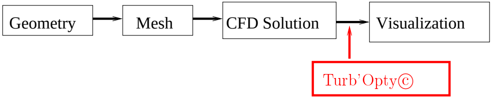
Main steps when using AGMRES

- 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 either AGMRES or GMRES.

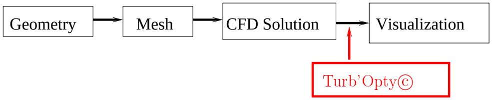
A CFD application



- At steady state, the solution of the stationary Navier-Stokes writes $F(q_{ref}, p_{ref}) = 0$
- $q = \{\rho, \rho U, \rho V, \rho W, \rho E, \rho k, \rho\omega\}$ flow variables (mass, momentum, energy, turbulence)
- $p =$ physical and geometrical flow parameters (pressure, temperature, shape, distance, ...)
- Turb'Opty©, FLUOREM : Find new solutions q with respect to the parameters p

About Turb'Opty [S. Aubert et al 2001]
About using AGMRES for Turb'Opty matrices [Nuentsa Wakam + Pacull, Computer & Fluids 2012]

A CFD application

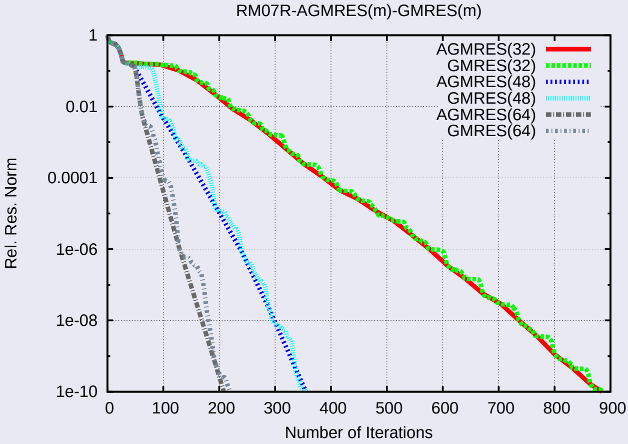


- At steady state, the solution of the stationary Navier-Stokes writes $F(q_{ref}, p_{ref}) = 0$
- $q = \{\rho, \rho U, \rho V, \rho W, \rho E, \rho k, \rho\omega\}$ flow variables (mass, momentum, energy, turbulence)
- $p =$ physical and geometrical flow parameters (pressure, temperature, shape, distance, ...)
- Turb'Opty©, FLUOREM : Find new solutions q with respect to the parameters p

About Turb'Opty [S. Aubert et al 2001]
About using AGMRES for Turb'Opty matrices [Nuentsa Wakam + Pacull, Computer & Fluids 2012]

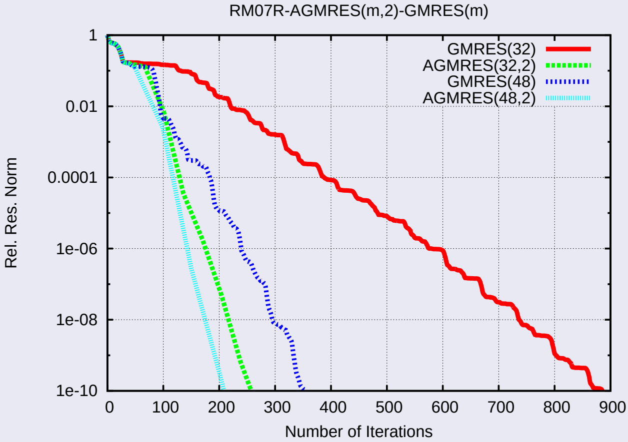
RM07R size = 381,689 nonzeros = 37,464,962

Experimental stability of AGMRES (without deflation)



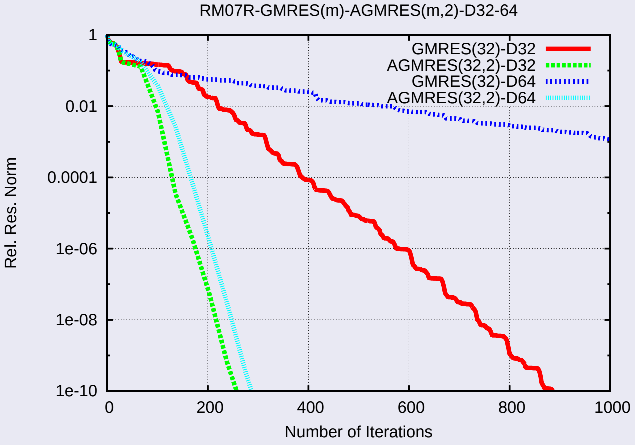
RM07R size = 381,689 nonzeros = 37,464,962

Influence of the augmented basis (no adaptive strategy)



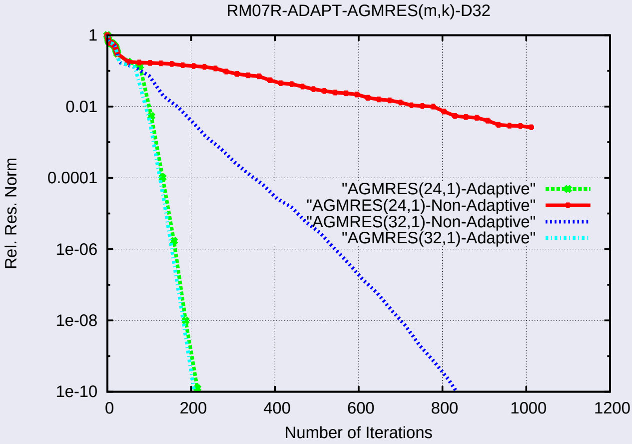
RM07R size $= 381{,}689$ nonzeros $= 37{,}464{,}962$

Influence of the number of subdomains (no adaptive strategy)



RM07R size = 381,689 nonzeros = 37,464,962

Influence of the adaptive strategy ($l = 2$ and $r_{max} = 5$)



IM07R and VV11R: CPU Time and communications

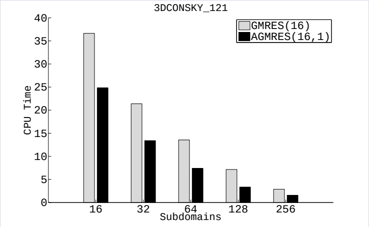
- IM07R Size = 261,465 entries : 26,872,530
- VV11R Size = 277,095 entries : 30,000,952
- $r = 2$, $l = 2$, $r_{max} = 6$

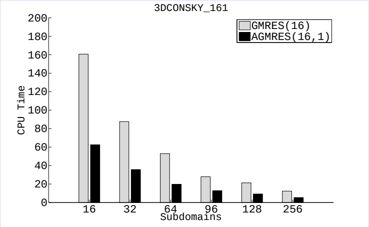
D \ m	24		32		48		
	Iter. Time	MSG	Iter. Time	MSG	Iter. Time	MSG	
GMRES(m)							VV11R
8	92.84	2.05	68.95	1.69	77.7	1.47	
16	101.1	12.27	89.37	11.47	63.2	7.66	
32	-	-	31.2	22.5	29.7	18.54	
AGMRES(m, r)							
8	52.8	1.28	38.5	1.02	40.5	1.05	
16	51.8	7.4	34.5	4.91	28.08	3.87	
32	38.3	25.6	31.2	22.5	29.7	18.5	
GMRES(m)							IM07R
8	76.219	2.6	73.3	2.63	63.669	2.31	
16	111.74	20.06	96.246	18.25	83.583	15.76	
32	-	-	-	-	77.066	59.87	
AGMRES(m, r)							
8	45.781	1.65	40.905	5.48	40.85	1.52	
16	36.492	21.65	34.803	24.12	33.65	23.64	
32	33.262	94.54	27.837	93.27	27.109	105.35	

3D Convection-Diffusion problems

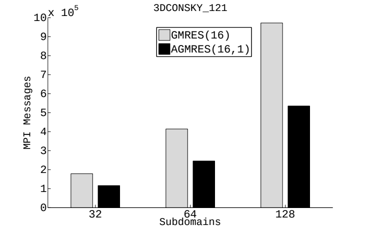
- 3DCONSKY_121 : size = 1,771,561; nonzeros = 50,178,241
- 3DCOSKY_161 : size= 4,173,281; nonzeros = 118,645,121

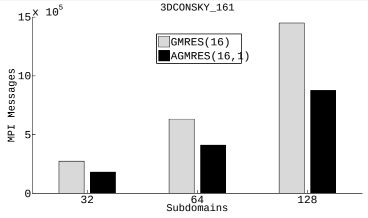
CPU Time





Number of MPI messages





Conclusion

- AGMRES: augmented Newton basis in GMRES(m)
- AGMRES + Schwarz: domain decomposition preconditioning
- Robustness: reduce the restarting effects and the domain decomposition effects
- Efficiency: increase granularity and scalability
- Numerical experiments with CFD problems: AGMRES faster than GMRES

AGMRES module

- Will be made available in PETSc in 2012

Paper in revision for publication in ETNA
preprint at http://www.irisa.fr/sage/desire