Inexact Subspace Iteration to Accelerate the Solution of Linear Systems with Multiple Right-Hand Sides

Carlos Balsa

ENSEEIHT - Institut National Polytechnique de Toulouse
FEUP - Universidade do Porto

May 12, 2006
1 Motivation

- Statement of the problem
- Basic problem that we studied
Outline

1 Motivation
   - Statement of the problem
   - Basic problem that we studied

2 Monitoring and improving the BlockCGSI algorithm
   - Improvements of BlockCGSI algorithm
   - Convergence analysis
Outline

1. **Motivation**
   - Statement of the problem
   - Basic problem that we studied

2. **Monitoring and improving the BlockCGSI algorithm**
   - Improvements of BlockCGSI algorithm
   - Convergence analysis

3. **Application to an airflow problem**
   - Working with an inaccurate basis
   - Optimal dimension of the basis
   - Costs-benefits
Outline

1. Motivation
   - Statement of the problem
   - Basic problem that we studied

2. Monitoring and improving the BlockCGSI algorithm
   - Improvements of BlockCGSI algorithm
   - Convergence analysis

3. Application to an airflow problem
   - Working with an inaccurate basis
   - Optimal dimension of the basis
   - Costs-benefits

4. Closure
   - Conclusions
   - Future work
MULTIPLE SOLUTION WITH CG (linear systems given in sequence).
Convergence of the CG depends on the eigenvalue distribution.

- Solve $M^{-1}A_jx = M^{-1}b_j$ with $j = 1, \ldots, N_{\text{its}}$ with the Conjugate Gradient algorithm.

- $A$ and $M$ are SPD and have constant spectral properties, i.e. $\sigma(M^{-1}A_j) \approx \{\lambda_{\text{min}}, \ldots, \lambda_{\text{max}}\}$ for $j = 1, \ldots, N_{\text{its}}$. 

\[ ||x(i) - x^\star||_A \leq 2 ||x(0) - x^\star||_A (\sqrt{\kappa} - 1 \sqrt{\kappa} + 1) \]
where $\kappa = \lambda_{\text{max}} / \lambda_{\text{min}}$, $x^\star$ is the exact solution and $x(i)$ is the approximated solution in the $i$th iteration.
MULTIPLE SOLUTION WITH CG (linear systems given in sequence).
Convergence of the CG depends on the eigenvalue distribution.

- Solve $M^{-1}A_jx = M^{-1}b_j$ with $j = 1, \ldots, N_{\text{its}}$ with the Conjugate Gradient algorithm.
- $A$ and $M$ are SPD and have constant spectral properties, i.e. $\sigma(M^{-1}A_j) \approx \{\lambda_{\min}, \ldots, \lambda_{\max}\}$ for $j = 1, \ldots, N_{\text{its}}$.
- $A$-norm of the error at CG iteration $i$ (Concus et al (1976)):

$$
\|x^{(i)} - x^*\|_A \leq 2 \|x^{(0)} - x^*\|_A \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1}\right)^i,
$$

where $\kappa = \lambda_{\max}/\lambda_{\min}$, $x^*$ is the exact solution and $x^{(i)}$ is the approximated solution in the $ith$ iteration.
Basic idea:

1. Partial spectral factorization of the coefficient matrix.

2. Build a spectral projector to improve the CG.
TWO PHASES ACCELERATING STRATEGY.
Use the Spectral Information to Improve the CG.

1. Partial spectral factorization of the coefficient matrix.
   - BlockCGSI algorithm (Arioli and Ruiz, 1995)

2. Build a spectral projector to improve the CG.
TWO PHASES ACCELERATING STRATEGY.
Use the Spectral Information to Improve the CG.

1. Partial spectral factorization of the coefficient matrix.
   - BlockCGSI algorithm (Arioli and Ruiz, 1995)
   - Chebyshev-PSF algorithm (Arioli and Ruiz, 2002)

2. Build a spectral projector to improve the CG.
TWO PHASES ACCELERATING STRATEGY.
Use the Spectral Information to Improve the CG.

1. Partial spectral factorization of the coefficient matrix.
   - BlockCGSI algorithm (Arioli and Ruiz, 1995)
   - Chebyshev-PSF algorithm (Arioli and Ruiz, 2002)
   - ChebFilter algorithm (Golub, Ruiz and Touhami, 2005)

2. Build a spectral projector to improve the CG.
TWO PHASES ACCELERATING STRATEGY.
Use the Spectral Information to Improve the CG.

1. Partial spectral factorization of the coefficient matrix.
   - BlockCGSI algorithm (Arioli and Ruiz, 1995)
   - Chebyshev-PSF algorithm (Arioli and Ruiz, 2002)
   - ChebFilter algorithm (Golub, Ruiz and Touhami, 2005)

2. Build a spectral projector to improve the CG.
   - Applied at the beginning of the CG run.
   - Applied at each CG Iteration.
TWO PHASES ACCELERATING STRATEGY.
Use the Spectral Information to Improve the CG.

1. Partial spectral factorization of the coefficient matrix.
   - BlockCGSI algorithm (Arioli and Ruiz, 1995)
   - Chebyshev-PSF algorithm (Arioli and Ruiz, 2002)
   - ChebFilter algorithm (Golub, Ruiz and Touhami, 2005)

2. Build a spectral projector to improve the CG.
   - Applied at the beginning of the CG run.
   - Applied at each CG Iteration.
FIRST PHASE: BlockCGSI Algorithm

**Inputs:** $A, M = R^T R \in \mathbb{R}^{n \times n}, s \in \mathbb{N}, m \in \mathbb{N}$

**Output:** a *near*-invariant subspace $\mathcal{W}$ with dimension $s$

**Begin**

$Z^{(0)} = \text{RANDOM}(n, s)$

$V^{(0)} \Gamma = Z^{(0)}$ such that $V^{(0)\top} M V^{(0)} = I_{s \times s}$

**For** $k = 1, \ldots, m$ **Do:**

Solve $M^{-1} A Z^{(k)} = V^{(k-1)}$ with blockCG

$Q^{(k)} \Gamma_k = Z^{(k)}$ such that $Q^{(k)\top} M Q^{(k)} = I_{s \times s}$

$\beta_k = Q^{(k)\top} A Q^{(k)}$

Diagonalize $\beta_k = U_k \Delta_k U_k^T$

where $U_k^T = U_k^{-1}$

and $\Delta_k = \text{Diag}(\delta_1, \ldots, \delta_s)$ (Ritz Values)

$V^{(k)} = Q^{(k)} U_k$ (Ritz Vectors)

**EndDo**

**End**
Assume that $W$ is set of $q$ generalized eigenvectors associated with the $q$ smallest eigenvalues of $(A, M_1)$, i.e. $M^{-1}AW = W\Delta$ with $\Delta = \text{diag}(\lambda_1, \ldots, \lambda_q)$ and $W^TM_1W = I_{q\times q}$. $M_1 = IC(A)$ for instance.
SECOND PHASE:

Assume that $W$ is set of $q$ generalized eigenvectors associated with the $q$ smallest eigenvalues of $(A, M_1)$, i.e. $M^{-1}AW = W\Delta$ with $\Delta = \text{diag}(\lambda_1, \ldots, \lambda_q)$ and $W^T M_1 W = I_{q \times q}$. $M_1 = IC(A)$ for instance.

SLRU-CG (Carpentieri et al, 2003)

Run the CG on $MAx = Mb$
where $M = M_1^{-1} + W\Delta^{-1} W^T$.
The new spectrum is

\[
\sigma(MA) = \begin{cases} 
\lambda_j & \text{if } j > q, \\
1 + \lambda_j & \text{if } j \leq q.
\end{cases}
\]
Assume that \( W \) is set of \( q \) generalized eigenvectors associated with the \( q \) smallest eigenvalues of \((A, M_1)\), i.e. \( M^{-1}AW = W\Delta \) with \( \Delta = \text{diag}(\lambda_1, \ldots, \lambda_q) \) and \( W^TM_1W = I_{q \times q} \). \( M_1 = IC(A) \) for instance.

**SLRU-CG** (Carpentieri et al, 2003)

Run the CG on
\[ MAx = Mb \]
where
\[ M = M_1^{-1} + W\Delta^{-1}W^T. \]
The new spectrum is
\[ \sigma(MA) = \begin{cases} \lambda_j & \text{if } j > q, \\ 1 + \lambda_j & \text{if } j \leq q. \end{cases} \]

**INIT-CG**

Given a starting guess \( x^{(0)} \), let
\[ x_1 = x^{(0)} + W\Delta^{-1}W^T(b - Ax^{(0)}). \]
Run the CG to solve
\[ M_1^{-1}Ax = M_1^{-1}b, \]
with \( x_1 \) as starting guess.
We expect faster convergence to the exact solution \( x^* \).
THE SIMPLE ALGORITHM FOR CFD SIMULATIONS.

- Discretised 2D driven cavity problem,
THE SIMPLE ALGORITHM FOR CFD SIMULATIONS.

- Discretised 2D driven cavity problem,
- Solve pressure correction systems $A_i x = b_i$ for $i = 1, \ldots, \text{NGits}$,
THE SIMPLE ALGORITHM FOR CFD SIMULATIONS.

- Discretised 2D driven cavity problem,
- Solve pressure correction systems $A_i x = b_i$ for $i = 1, \ldots, \text{NGits}$,
- $A_i$ are SPD and maintain the same spectral properties:

<table>
<thead>
<tr>
<th>Eigenvalues</th>
<th>$A_1$</th>
<th>$M_1^{-1}A_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\lambda_1$</td>
<td>$3.211006e-08$</td>
<td>$4.428912e-05$</td>
</tr>
<tr>
<td>$\lambda_2$</td>
<td>$8.877776e-07$</td>
<td>$1.232523e-03$</td>
</tr>
<tr>
<td>$\lambda_3$</td>
<td>$9.572325e-07$</td>
<td>$1.328789e-03$</td>
</tr>
<tr>
<td>$\lambda_4$</td>
<td>$1.769538e-06$</td>
<td>$2.473183e-03$</td>
</tr>
<tr>
<td>$\lambda_5$</td>
<td>$3.550269e-06$</td>
<td>$4.915760e-03$</td>
</tr>
<tr>
<td>$\lambda_{10}$</td>
<td>$7.985443e-06$</td>
<td>$1.100848e-02$</td>
</tr>
<tr>
<td>$\lambda_{15}$</td>
<td>$1.164407e-05$</td>
<td>$1.611350e-02$</td>
</tr>
<tr>
<td>$\lambda_{20}$</td>
<td>$1.591812e-05$</td>
<td>$2.193773e-02$</td>
</tr>
<tr>
<td>$\lambda_{\text{max}}$</td>
<td>$1.005278e-02$</td>
<td>$1.226147e+00$</td>
</tr>
<tr>
<td>$\kappa_2$</td>
<td>$3.130730e+05$</td>
<td>$2.768510e+04$</td>
</tr>
</tbody>
</table>
First Phase: Eigencomputation with BlockCGSI algo.

Inner stopping criterion: \[ \omega_1 = \frac{||v_1 - M^{-1}Az_1||_M}{||z_1||_{M+1}} \leq \epsilon \]
First Phase: Eigencomputation with BlockCGSI algo.

Inner stopping criterion: \[ \omega_1 = \frac{||v_1 - M^{-1}Az_1||_M}{||z_1||_{M+1}} \leq \epsilon \]

Outer error: \[ S_j = ||M^{-1}Av_j^{(k)} - \delta_j^{(k)}v_j^{(k)}||_M \]
**First Phase: Eigencomputation with BlockCGSI algo.**

**Inner stopping criterion:**
\[ \omega_1 = \frac{\|v_1 - M^{-1}Az_1\|_M}{\|z_1\|_M + 1} \leq \epsilon \]

**Outer error:**
\[ S_j = \|M^{-1}A v_j^{(k)} - \delta_j^{(k)} v_j^{(k)}\|_M \]

**block size** = 5, \( \epsilon = 10^{-8} \)

---

**Motivation**
Monitoring and improving the BlockCGSI algorithm
Application to an airflow problem

**Statement of the problem**
Basic problem that we studied
First Phase: Eigencomputation with BlockCGSI algo.

Inner stopping criterion: \[ \omega_1 = \frac{||v_1 - M^{-1}Az_1||_M}{||z_1||_{M+1}} \leq \epsilon \]

Outer error: \[ S_j = ||M^{-1}A\nu_j^{(k)} - \delta_j^{(k)}\nu_j^{(k)}||_M \]

block size = 5, \( \epsilon = 10^{-8} \)

![Graph showing error bound of the Ritz values versus the BlockCGSI iterations for block size = 5, \( \epsilon = 10^{-8} \) and block size = 5, \( \epsilon = 10^{-10} \).]
Second Phase: SLRU-CG

**System** \( A_{10} x = b_{10}, \epsilon = 10^{-4} \)

**SLRU-CG:** Classical CG algorithm with the SLRU preconditioner.
Second Phase: SLRU-CG

System \( A_{10}x = b_{10}, \epsilon = 10^{-4} \)

System \( A_{10}x = b_{10}, \epsilon = 10^{-10} \)

SLRU-CG: Classical CG algorithm with the SLRU preconditioner.
Second Phase: INIT-CG

System \( A_{10} x = b_{10}, \epsilon = 10^{-4} \)

INIT-CG: Classical CG algorithm with the deflated starting guess.
Second Phase: INIT-CG

System $A_{10}x = b_{10}$, $\epsilon = 10^{-4}$

INIT-CG: Classical CG algorithm with the deflated starting guess.
## Cost-Benefit

### BlockCGSI and SLRU-CG algorithms

<table>
<thead>
<tr>
<th>$s$</th>
<th>$\epsilon$</th>
<th>Eq. Iter. (costs)</th>
<th>SLRU-CG</th>
<th>Benefits</th>
<th>Systems Amortization</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Eq. Iter.</td>
<td>Its</td>
<td>Time</td>
</tr>
<tr>
<td>5</td>
<td>1e-04</td>
<td>320</td>
<td>79</td>
<td>7.65</td>
<td>74</td>
</tr>
<tr>
<td></td>
<td>1e-10</td>
<td>1195</td>
<td>53</td>
<td>5.18</td>
<td>100</td>
</tr>
<tr>
<td>10</td>
<td>1e-04</td>
<td>470</td>
<td>62</td>
<td>8.65</td>
<td>91</td>
</tr>
<tr>
<td></td>
<td>1e-10</td>
<td>1590</td>
<td>38</td>
<td>5.32</td>
<td>115</td>
</tr>
<tr>
<td>15</td>
<td>1e-04</td>
<td>580</td>
<td>56</td>
<td>9.32</td>
<td>97</td>
</tr>
<tr>
<td></td>
<td>1e-10</td>
<td>1590</td>
<td>30</td>
<td>4.96</td>
<td>123</td>
</tr>
<tr>
<td>20</td>
<td>1e-04</td>
<td>680</td>
<td>54</td>
<td>9.96</td>
<td>99</td>
</tr>
<tr>
<td></td>
<td>1e-10</td>
<td>1760</td>
<td>29</td>
<td>5.32</td>
<td>124</td>
</tr>
</tbody>
</table>
## Cost-Benefit

<table>
<thead>
<tr>
<th>$s$</th>
<th>$\epsilon$</th>
<th>Eq. Iter. (costs)</th>
<th>SLRU-CG</th>
<th>Benefits</th>
<th>Systems Amortization</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Its</td>
<td>Time</td>
<td>Its</td>
</tr>
<tr>
<td>5</td>
<td>1e-04</td>
<td>320</td>
<td>79</td>
<td>7.65</td>
<td>74</td>
</tr>
<tr>
<td></td>
<td>1e-10</td>
<td>1195</td>
<td>53</td>
<td>5.18</td>
<td>100</td>
</tr>
<tr>
<td>10</td>
<td>1e-04</td>
<td>470</td>
<td>62</td>
<td>8.65</td>
<td>91</td>
</tr>
<tr>
<td></td>
<td>1e-10</td>
<td>1590</td>
<td>38</td>
<td>5.32</td>
<td>115</td>
</tr>
<tr>
<td>15</td>
<td>1e-04</td>
<td>580</td>
<td>56</td>
<td>9.32</td>
<td>97</td>
</tr>
<tr>
<td></td>
<td>1e-10</td>
<td>1590</td>
<td>30</td>
<td>4.96</td>
<td>123</td>
</tr>
<tr>
<td>20</td>
<td>1e-04</td>
<td>680</td>
<td>54</td>
<td>9.96</td>
<td>99</td>
</tr>
<tr>
<td></td>
<td>1e-10</td>
<td>1760</td>
<td>29</td>
<td>5.32</td>
<td>124</td>
</tr>
</tbody>
</table>
### Cost-Benefit

<table>
<thead>
<tr>
<th>s</th>
<th>$\epsilon$</th>
<th>Eq. Iter. (costs)</th>
<th>SLRU-CG</th>
<th>Benefits</th>
<th>Systems Amortization</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Eq. Iter.</td>
<td>SLRU-CG</td>
<td>Benefits</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Eq. Iter.</td>
<td>SLRU-CG</td>
<td>Benefits</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Eq. Iter.</td>
<td>SLRU-CG</td>
<td>Benefits</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Eq. Iter.</td>
<td>SLRU-CG</td>
<td>Benefits</td>
<td></td>
</tr>
</tbody>
</table>

**BlockCGSI and SLRU-CG algorithms**

<table>
<thead>
<tr>
<th>s</th>
<th>$\epsilon$</th>
<th>Eq. Iter. (costs)</th>
<th>SLRU-CG</th>
<th>Benefits</th>
<th>Systems Amortization</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Eq. Iter.</td>
<td>SLRU-CG</td>
<td>Benefits</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Eq. Iter.</td>
<td>SLRU-CG</td>
<td>Benefits</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Eq. Iter.</td>
<td>SLRU-CG</td>
<td>Benefits</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Eq. Iter.</td>
<td>SLRU-CG</td>
<td>Benefits</td>
<td></td>
</tr>
</tbody>
</table>

**BlockCGSI and INIT-CG algorithms**
Open Questions

First-phase

- Can we improve the convergence of the Inverse Iteration?
Open Questions

First-phase
- Can we improve the convergence of the Inverse Iteration?
- Which block size should we choose?
Open Questions

First-phase

- Can we improve the convergence of the Inverse Iteration?
- Which block size should we choose?
- When to stop the inner (blockCG) iterations?
Open Questions

First-phase
- Can we improve the convergence of the Inverse Iteration?
- Which block size should we choose?
- When to stop the inner (blockCG) iterations?

Second-phase
- SLRU-CG or INIT-CG?
Chebyshev polynomials as a spectral filtering tool.

- Chebyshev polynomial uniformly convergent below $\xi$ (near zero) on the interval $]\mu_f, \lambda_{max}[\$ with the fixed point value 1 on 0.
Chebyshev polynomials as a spectral filtering tool.

- Chebyshev polynomial uniformly convergent below $\xi$ (near zero) on the interval $]\mu_f, \lambda_{\text{max}}[$ with the fixed point value 1 on 0.

**EXAMPLE:** Filtering a set of 5 random vectors on $]\mu_f, \lambda_{\text{max}}[$ with $\mu_f = 1e-04$, $\lambda_{\text{min}} = 3.07e-09$ and $\lambda_{\text{max}} = 2.08e+00$.

Eigencomponents without filtering
Chebyshev polynomials as a spectral filtering tool.

- Chebyshev polynomial uniformly convergent below $\xi$ (near zero) on the interval $[\mu_f, \lambda_{\text{max}}]$ with the fixed point value 1 on 0.

**EXAMPLE:** Filtering a set of 5 random vectors on $[\mu_f, \lambda_{\text{max}}]$ with $\mu_f = 1 \times 10^{-04}$, $\lambda_{\text{min}} = 3.07 \times 10^{-09}$ and $\lambda_{\text{max}} = 2.08 \times 10^{00}$.

Eigencomponents without filtering

After filtering with $\xi = 1 \times 10^{-08}$
Sliding Window

- **Idea:** Enlarge (or reduce) dynamically the dimension of the targeted invariant subspace.
**Sliding Window**

**Idea:** Enlarge (or reduce) dynamically the dimension of the targeted invariant subspace.

<table>
<thead>
<tr>
<th>INCORPORATE NEW VECTORS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Inputs:</strong> $\ell \in \mathbb{N}, M = R^T R \in \mathbb{R}^{n \times n}$, $V^{(0)} \in \mathbb{R}^{n \times s}, W^{(k)} \in \mathbb{R}^{n \times p}, V^{(k)} \in \mathbb{R}^{n \times (s-\ell)}$</td>
</tr>
<tr>
<td><strong>Begin</strong></td>
</tr>
<tr>
<td>a) $Y = \text{RANDOM}(s, \ell)$</td>
</tr>
<tr>
<td>b) $P = V^{(0)} Y$</td>
</tr>
<tr>
<td>c) $P = Q \Gamma$ such that $Q^T M Q = I_{\ell \times \ell}$</td>
</tr>
<tr>
<td>d) $P = Q - W^{(k)} W^{(k)^T} M Q$</td>
</tr>
<tr>
<td>e) $V^{(k)} = [V^{(k)} P]$</td>
</tr>
<tr>
<td><strong>End</strong></td>
</tr>
</tbody>
</table>
Monitoring the Inverse Subspace (outer) Iteration

Find $v_j$ and $\delta_j$ such that

\[
M^{-1} A v_j \approx \delta_j v_j \iff A v_j \approx \delta_j M v_j \quad \text{for} \quad j = 1, \ldots, q
\]
Monitoring the Inverse Subspace (outer) Iteration

Find $v_j$ and $\delta_j$ such that

$$M^{-1}Av_j \approx \delta_j v_j \iff Av_j \approx \delta_j Mv_j \text{ for } j = 1, \ldots, q$$

Eigenvalue error bound (Parlett, 1998):

$$|\lambda_j - \delta_j| \leq \frac{\|Av_j - \delta_j Mv_j\|_{M^{-1}}}{\|Mv_j\|_{M^{-1}}} = \|M^{-1}Av_j - \delta_j v_j\|_M$$
Monitoring the Inverse Subspace (outer) Iteration

Find \( v_j \) and \( \delta_j \) such that

\[
M^{-1} Av_j \approx \delta_j v_j \quad \Leftrightarrow \quad Av_j \approx \delta_j Mv_j \quad \text{for} \quad j = 1, \ldots, q
\]

Eigenvalue error bound (Parlett, 1998):

\[
|\lambda_j - \delta_j| \leq \frac{\|Av_j - \delta_j Mv_j\|_{M^{-1}}}{\|Mv_j\|_{M^{-1}}} = \frac{\|M^{-1} Av_j - \delta_j v_j\|_M}{\|M^{-1}\|_M}
\]

Convergence criterion at inverse iteration \((k)\):

\[
\frac{\|M^{-1} Av_j^{(k)} - \delta_j^{(k)} v_j^{(k)}\|_M}{\delta_j^{(k)}} \leq 10^{-t}
\]
blockCG (inner) Iteration

Solve $M^{-1} A z_j^{(k)} \approx v_j^{(k-1)}$ iteratively for $j = 1, ..., s$

For each $z_j^{[1]}, z_j^{[2]}, \ldots, z_j^{[i]} \rightarrow z_j^{(k)}$ consider $r_j^{[i]} = v_j^{(k-1)} - M^{-1} A z_j^{[i]}$
Motivation
Monitoring and improving the BlockCGSI algorithm
Application to an airflow problem
Closure

Improvements of BlockCGSI algorithm
Convergence analysis

blockCG (inner) Iteration

Solve $M^{-1}A z_j^{(k)} \approx v_j^{(k-1)}$ iteratively for $j = 1, \ldots, s$

For each $z_j^{[1]}, z_j^{[2]}, \ldots, z_j^{[i]} \rightarrow z_j^{(k)}$ consider $r_j^{[i]} = v_j^{(k-1)} - M^{-1}A z_j^{[i]}

Eigenvalue error bound associated with $\delta_j^{[i]} = z_j^{[i]}^T A z_j^{[i]} / z_j^{[i]}^T M z_j^{[i]}$ is

$$\left| \lambda_j - \delta_j^{[i]} \right| \leq \left\| M^{-1}A z_j^{[i]} - \delta_j^{[i]} z_j^{[i]} \right\|_M = \sqrt{\frac{\left\| v_j^{(k-1)} - \delta_j^{[i]} z_j^{[i]} \right\|_M^2 + \left\| r_j^{[i]} \right\|_M^2}{\left\| z_j^{[i]} \right\|_M^2}} \equiv \sqrt{\phi_j^{[i]}^2 + \omega_j^{[i]}^2}$$
blockCG (inner) Iteration

Solve $M^{-1} A z_j^{(k)} \approx v_j^{(k-1)}$ iteratively for $j = 1, \ldots, s$

For each $z_j^{[1]}, z_j^{[2]}, \ldots, z_j^{[i]} \rightarrow z_j^{(k)}$ consider $r_j^{[i]} = v_j^{(k-1)} - M^{-1} A z_j^{[i]}$

Eigenvalue error bound associated with $\delta_j^{[i]} = z_j^{[i]} T A z_j^{[i]} / z_j^{[i]} T M z_j^{[i]}$ is

$$
|\lambda_j - \delta_j^{[i]}| \leq \frac{\left\| M^{-1} A z_j^{[i]} - \delta_j^{[i]} z_j^{[i]} \right\|_M}{\left\| z_j^{[i]} \right\|_M} = \sqrt{\left\| v_j^{(k-1)} - \delta_j^{[i]} z_j^{[i]} \right\|_M^2 + \left\| r_j^{[i]} \right\|_M^2} \equiv \sqrt{\phi_j^{[i]}^2 + \omega_j^{[i]}^2}
$$

where $\omega_j^{[i]} = \frac{\left\| r_j^{[i]} \right\|_M}{\left\| z_j^{[i]} \right\|_M}$ and $\phi_j^{[i]} = \frac{\left\| v_j^{(k-1)} - \delta_j^{[i]} z_j^{[i]} \right\|_M}{\left\| z_j^{[i]} \right\|_M}$

\[
\phi_j^{[i]} = \frac{\|v_j^{(k-1)} - \delta_j^{[i]} z_j^{[i]}\|_M}{\|z_j^{[i]}\|_M} \rightarrow \delta_j^* \tan(\theta_j) \quad \text{as } i \rightarrow \infty
\]

\[
\delta_j^* = v_j^{(k-1)^T} z_j^* / \|z_j^*\|_M, \quad z_j^* = A^{-1} M v_j^{(k-1)} \quad \text{and} \quad \theta_j = \angle (z_j^*, v_j^{(k-1)})_M
\]
Motivation
Monitoring and improving the BlockCGSI algorithm
Application to an airflow problem
Closure

Improvements of BlockCGSI algorithm
Convergence analysis

\[
\phi_j[i] = \frac{\|v_j^{(k-1)} - \delta_j[i] z_j[i]\|_M}{\|z_j[i]\|_M} \rightarrow \delta_j^* \tan(\theta_j) \quad i \rightarrow \infty
\]

\[
\delta_j^* = v_j^{(k-1)T} z_j^* / \|z_j^*\|_M, \quad z_j^* = A^{-1} M v_j^{(k-1)} \quad \text{and} \quad \theta_j = \angle(z_j^*, v_j^{(k-1)})_M
\]

Without initial filtering
\[ \phi_j[i] = \frac{\|v_j^{(k-1)} - \delta_j[i]z_j[i]\|_M}{\|z_j[i]\|_M} \xrightarrow{i \to \infty} \delta_j^* \tan(\theta_j) \]

\[ \delta_j^* = v_j^{(k-1)T}z_j^*/\|z_j^*\|_M, \quad z_j^* = A^{-1}Mv_j^{(k-1)} \text{ and } \theta_j = \angle(z_j^*, v_j^{(k-1)})_M \]

**Without initial filtering**

**With initial filtering**
blockCG (inner) stopping criterion.

Outer stopping criterion:
\[
\left\| M^{-1} A v_j^{(k-1)} - \delta_j^{(k-1)} v_j^{(k-1)} \right\|_M \leq 10^{-t}
\]
blockCG (inner) stopping criterion.

Outer stopping criterion:

\[
\frac{\left\| M^{-1} A v_j^{(k-1)} - \delta_j^{(k-1)} v_j^{(k-1)} \right\|_M}{\delta_j^{(k-1)}} \leq 10^{-t}
\]

Inner stopping criterion:

\[
\omega_1^{[i]} \leq \varepsilon \quad \text{with} \quad \varepsilon = 10^{-t} \delta_1^{(k-1)}
\]
**blockCG (inner) stopping criterion.**

Outer stopping criterion:

\[
\left\lVert M^{-1} A v_j^{(k-1)} - \delta_j^{(k-1)} v_j^{(k-1)} \right\rVert_M \leq 10^{-t}
\]

Inner stopping criterion:

\[
\omega_1^{[i]} \leq \varepsilon \quad \text{with} \quad \varepsilon = 10^{-t \delta_1^{(k-1)}}
\]
Description of the problem

- Airflow simulation code named ICARE (Braza, 1986).
- Flow around a wing in 2D.
- The code describes the transition of the flow from laminar to turbulent state.
- Solution using finite elements through prediction-correction algorithm and a semi-implicit discretization scheme.
- At each time step: solution of a linear system with the same coefficient matrix and changing right-hand sides

\[ Ax = b_i. \]
Matrix of order 27 283 with 187 487 nonzeros.

\[ \sigma(A) \in \left[ 5.0e-05, 1.1e+01 \right] \text{ i.e. } \kappa_2 \approx 2.2e+5. \]
Matrix of order 27 283 with 187 487 nonzeros.

\[ \sigma(A) \in [5.0e-05, 1.1e+01] \text{ i.e. } \kappa_2 \approx 2.2e+5. \]

\[ \sigma(M^{-1}A) \in [6.5e-05, 1.7e+00] \text{ i.e. } \kappa_2 \approx 2.6e+4. \]
Using an inaccurate basis with INIT-CG

INIT-CG: small residuals $\Rightarrow$ high accuracy (first phase expensive).

INIT-CG: $q = 17$
Using an inaccurate basis with INIT-CG

INIT-CG: \( q = 17 \)

Restarted INIT-CG: \( q = 17, t = 1 \)

INIT-CG: small residuals \( \rightarrow \) high accuracy (first phase expensive).

Restarted INIT-CG: small residuals \( \rightarrow \) low accuracy (first phase not expensive).
Pre-computational and solution costs

Pre-computational cost ($C_{BCGSI}$)

- Pre-computational cost depends almost from the basis dimension $q$ and from the block size $s$. 

![Graph showing pre-computational cost ($C_{BCGSI}$) vs. basis dimension $q$ for different block sizes.](image)
Pre-computational and solution costs

- Pre-computational cost depends almost from the basis dimension $q$ and from the block size $s$.
- Solution cost decreases with the basis dimension $q$ and stagnates.
Optimal dimension of the basis

Total cost = $C_{BCGSI} + C_{\text{Init-CG}} \times \text{NGits}$
Optimal dimension of the basis

Total cost = $C_{BCGSI} + C_{Init-CG} \times NGits$

Total costs: block size = 10, NGits = 100
Optimal dimension of the basis

\[ \text{Total cost} = C_{BCGSI} + C_{\text{Init-CG}} \times \text{NGits} \]

**Total costs: block size = 10, NGits = 100**

**Total costs: block size = 10, NGits = 1000**
Motivation
Monitoring and improving the BlockCGSI algorithm
Application to an airflow problem
Closure

Working with an inaccurate basis
Optimal dimension of the basis
Costs-benefits

Optimal dimension of the basis

Total cost = \( C_{BCGSI} + C_{\text{Init-CG}} \times NGits \)

- Optimal dimension basis \( q \) varies very slightly with increasing \( NGits \).
- In first phase should approximate just the extremal eigenvalues.

C. Balsa  PhD Thesis
Costs-benefits of the two-phase approach

<table>
<thead>
<tr>
<th>Spectral fact.</th>
<th>INIT-CG Amor.</th>
<th>Total cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>( q )</td>
<td>( C_{BCGSI} )</td>
<td>( \text{Mflops} )</td>
</tr>
<tr>
<td>----------------</td>
<td>---------------</td>
<td>------------</td>
</tr>
<tr>
<td>0</td>
<td>–</td>
<td>423</td>
</tr>
<tr>
<td>5</td>
<td>2041</td>
<td>261</td>
</tr>
<tr>
<td>10</td>
<td>2377</td>
<td>189</td>
</tr>
<tr>
<td>15</td>
<td>2767</td>
<td>162</td>
</tr>
<tr>
<td>20</td>
<td>2767</td>
<td>145</td>
</tr>
<tr>
<td>25</td>
<td>3225</td>
<td>127</td>
</tr>
<tr>
<td>30</td>
<td>4242</td>
<td>121</td>
</tr>
<tr>
<td>35</td>
<td>4242</td>
<td>122</td>
</tr>
<tr>
<td>40</td>
<td>7473</td>
<td>116</td>
</tr>
<tr>
<td>45</td>
<td>11015</td>
<td>117</td>
</tr>
<tr>
<td>50</td>
<td>14972</td>
<td>118</td>
</tr>
<tr>
<td>55</td>
<td>24950</td>
<td>114</td>
</tr>
</tbody>
</table>

- Pre-computation costs (block size = 15 and \( q = 25 \)) : 3225 mflops.
- Solution of one system using INIT-CG : 127 mflops.
- CG requires 427 mflops.
- Computation of spectral iteration is paid back after 11 time steps.
- I.e. reduction of 62% in the total amount of work.
Costs-benefits of the two-phase approach

<table>
<thead>
<tr>
<th>Spectral fact.</th>
<th>INIT-CG</th>
<th>Amor.</th>
<th>Total cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>q</td>
<td>$c_{BCGSI}$</td>
<td>Mflops</td>
<td>rhs</td>
</tr>
<tr>
<td>0</td>
<td>–</td>
<td>423</td>
<td>–</td>
</tr>
<tr>
<td>5</td>
<td>2041</td>
<td>261</td>
<td>13</td>
</tr>
<tr>
<td>10</td>
<td>2377</td>
<td>189</td>
<td>11</td>
</tr>
<tr>
<td>15</td>
<td>2767</td>
<td>162</td>
<td>11</td>
</tr>
<tr>
<td>20</td>
<td>2767</td>
<td>145</td>
<td>10</td>
</tr>
<tr>
<td>25</td>
<td>3225</td>
<td>127</td>
<td>11</td>
</tr>
<tr>
<td>30</td>
<td>4242</td>
<td>121</td>
<td>15</td>
</tr>
<tr>
<td>35</td>
<td>4242</td>
<td>122</td>
<td>15</td>
</tr>
<tr>
<td>40</td>
<td>7473</td>
<td>116</td>
<td>25</td>
</tr>
<tr>
<td>45</td>
<td>11015</td>
<td>117</td>
<td>36</td>
</tr>
<tr>
<td>50</td>
<td>14972</td>
<td>118</td>
<td>50</td>
</tr>
<tr>
<td>55</td>
<td>24950</td>
<td>114</td>
<td>81</td>
</tr>
</tbody>
</table>

- Pre-computation costs (block size = 15 and $q = 25$): 3225 mflops.
- Solution of one system using INIT-CG: 127 mflops.
- CG requires 427 mflops.
- Computation of spectral iteration is paid back after 11 time steps.
- I.e. reduction of 62% in the total amount of work.
- With $NGits = 1000, q = 40$ $\implies$ a reduction of 71%.
Conclusions

Two-phase approach:

- Effective to reduce the total amount of work,
- First level of preconditioning improves the strategy,
- Purely iterative (does not need $A$ explicitly),
- Enables a good control of the memory requirements,
- Efficient implementation using level 3 BLAS kernels.
First-Phase: BlockCGSI algorithm.

- Set the block size large in agreement with the computer performance,
- The dimension of the subspace to be computed is adjusted dynamically through *Sliding Window*,
- Theoretical bounds on the Subspace Iteration combined with blockCG are derived,
- The accuracy of the spectral information is controlled explicitly.
First-Phase: BlockCGSI algorithm.
- Set the block size large in agreement with the computer performance,
- The dimension of the subspace to be computed is adjusted dynamically through *Sliding Window*,
- Theoretical bounds on the Subspace Iteration combined with blockCG are derived,
- The accuracy of the spectral information is controlled explicitly.

Second-Phase: Restarted INIT-CG is the right choice.
- Effective even if the spectral information is inaccurate,
- It is not expensive.
Future Work

- Compare the BlockCGSI algorithm with other techniques for eigencomputation,
- Implementation on a large scale PDE numerical simulation (3D problem),
- Adapt the strategy to indefinite matrices,
- Adapt the strategy to non-linear problems.