High performance Krylov subspace method variants

and their behavior in finite precision

Erin Carson

New York University

HPCSE17, May 24, 2017

Collaborators

Miroslav Rozložník

Institute of Computer Science, Czech Academy of Sciences

Zdeněk Strakoš

Department of Numerical Mathematics, Faculty of Mathematics and Physics, Charles University

Petr Tichý

Department of Numerical Mathematics, Faculty of Mathematics and Physics, Charles University

Miroslav Tůma

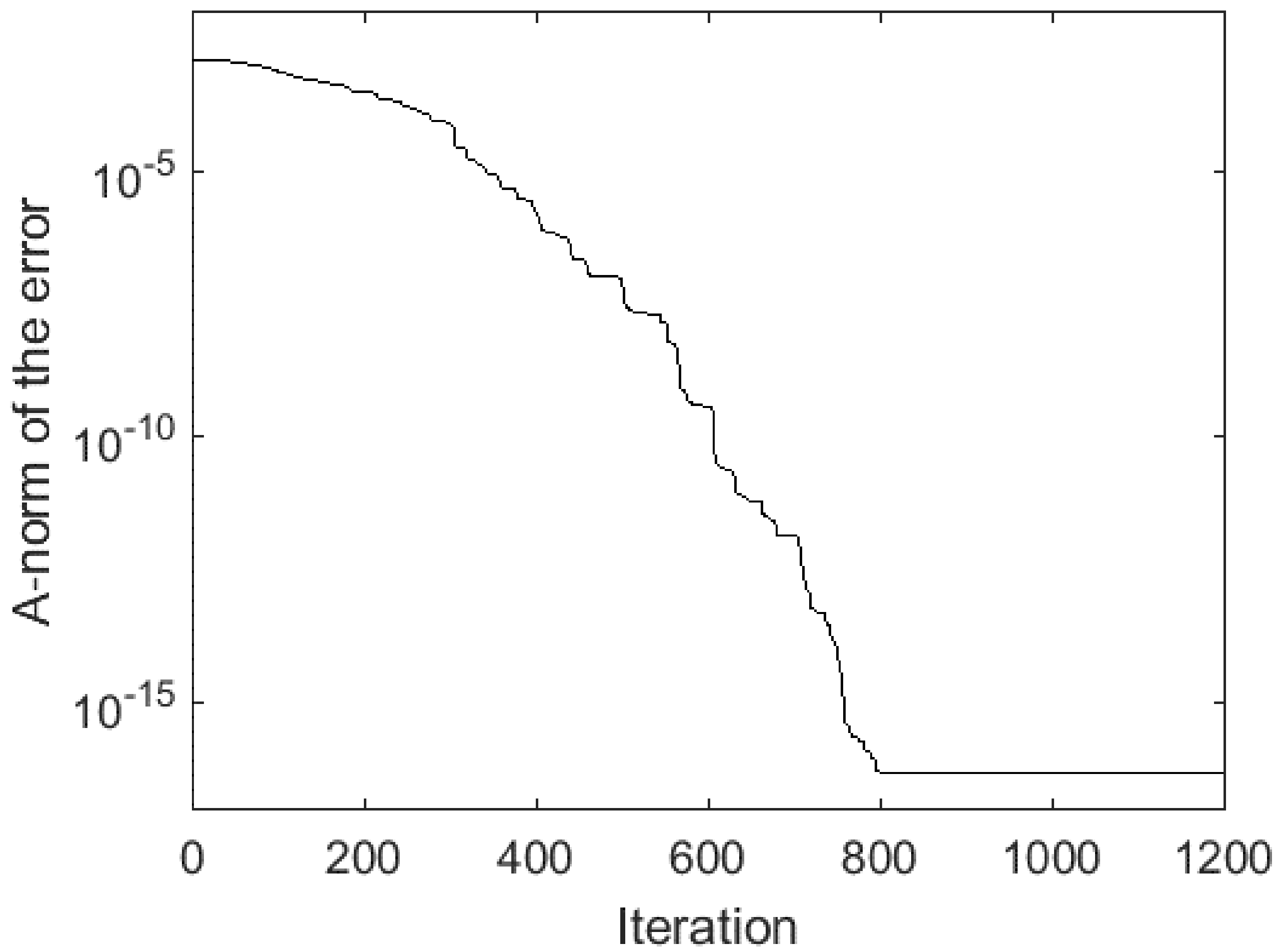
Department of Numerical Mathematics, Faculty of Mathematics and Physics, Charles University

Preprint NCMM/2016/08:
http://www.karlin.mff.cuni.cz/~strakos/download/2016_CarRozStrTicTum_16.pdf

Conjugate Gradient method for solving Ax = b
double precision ($\varepsilon = 2^{-53}$)

$$\|x_i - x\|_A = \sqrt{(x_i - x)^T A (x_i - x)}$$

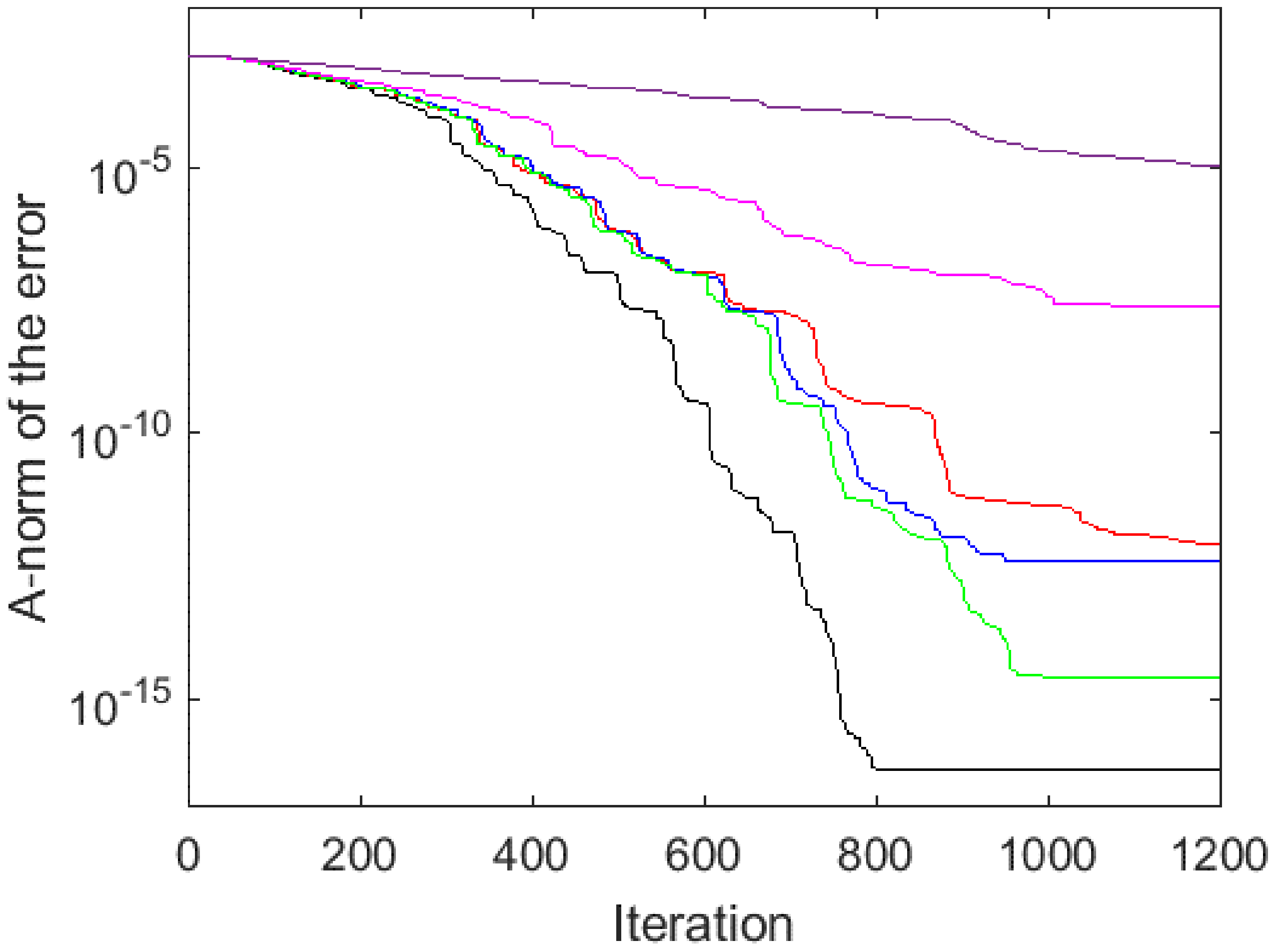
$$x_i = x_{i-1} + \alpha_i p_i$$
$$r_i = r_{i-1} - \alpha_i A p_i$$
$$p_i = r_i + \beta_i p_i$$



Conjugate Gradient method for solving Ax = b
double precision ($\varepsilon = 2^{-53}$)

$$\|x_i - x\|_A = \sqrt{(x_i - x)^T A (x_i - x)}$$

$$x_i = x_{i-1} + \alpha_i p_i$$
$$r_i = r_{i-1} - \alpha_i A p_i$$
$$p_i = r_i + \beta_i p_i$$



Krylov subspace methods

- **Linear systems** $Ax = b$, eigenvalue problems, singular value problems, least squares, etc.
- Best for: A large & very sparse, stored implicitly, or only approximation needed

- **Krylov Subspace Method** is a projection process onto the Krylov subspace

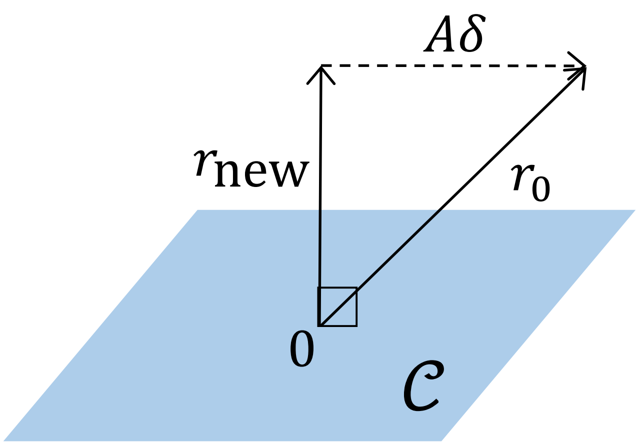
$$\mathcal{K}_i(A, r_0) = \text{span}\{r_0, Ar_0, A^2 r_0, \ldots, A^{i-1} r_0\}$$

where A is an $N \times N$ matrix and $r_0 = b - Ax_0$ is a length-N vector

- In each iteration,
  - Add a dimension to the Krylov subspace
    - Forms nested sequence of Krylov subspaces

$$\mathcal{K}_1(A, r_0) \subset \mathcal{K}_2(A, r_0) \subset \cdots \subset \mathcal{K}_i(A, r_0)$$

  - Orthogonalize (with respect to some $\mathcal{C}_i$)
  - Select approximate solution $x_i \in x_0 + \mathcal{K}_i(A, r_0)$ using $r_i = b - Ax_i \perp \mathcal{C}_i$



- Ex: Lanczos/**Conjugate Gradient (CG)**, Arnoldi/Generalized Minimum Residual (GMRES), Biconjugate Gradient (BICG), BICGSTAB, GKL, LSQR, etc.

The conjugate gradient method

A is symmetric positive definite, $\mathcal{C}_i = \mathcal{K}_i(A, r_0)$

The conjugate gradient method

A is symmetric positive definite, $\mathcal{C}_i = \mathcal{K}_i(A, r_0)$

$$r_i \perp \mathcal{K}_i(A, r_0) \quad \Longleftrightarrow \quad \|x - x_i\|_A = \min_{z \in x_0 + \mathcal{K}_i(A, r_0)} \|x - z\|_A$$

The conjugate gradient method

A is symmetric positive definite, $\mathcal{C}_i = \mathcal{K}_i(A, r_0)$

$$r_i \perp \mathcal{K}_i(A, r_0) \quad \Leftrightarrow \quad \|x - x_i\|_A = \min_{z \in x_0 + \mathcal{K}_i(A, r_0)} \|x - z\|_A$$

$$\Longrightarrow \quad r_{N+1} = 0$$

The conjugate gradient method

A is symmetric positive definite, $\mathcal{C}_i = \mathcal{K}_i(A, r_0)$

$$r_i \perp \mathcal{K}_i(A, r_0) \quad \Leftrightarrow \quad \|x - x_i\|_A = \min_{z \in x_0 + \mathcal{K}_i(A, r_0)} \|x - z\|_A$$

$$\Longrightarrow \quad r_{N+1} = 0$$

Connection with Lanczos

- With $v_1 = r_0 / \|r_0\|$, i iterations of Lanczos produces $N \times i$ matrix $V_i = [v_1, \dots, v_i]$, and $i \times i$ tridiagonal matrix T_i such that
$$AV_i = V_i T_i + \delta_{i+1} v_{i+1} e_i^T, \qquad T_i = V_i^* A V_i$$
- CG approximation x_i is obtained by solving the reduced model
$$T_i y_i = \|r_0\| e_1, \qquad x_i = x_0 + V_i y_i$$

The conjugate gradient method

A is symmetric positive definite, $\mathcal{C}_i = \mathcal{K}_i(A, r_0)$

$$r_i \perp \mathcal{K}_i(A, r_0) \quad \Longleftrightarrow \quad \|x - x_i\|_A = \min_{z \in x_0 + \mathcal{K}_i(A, r_0)} \|x - z\|_A$$

$$\Longrightarrow \quad r_{N+1} = 0$$

Connection with Lanczos

- With $v_1 = r_0/\|r_0\|$, i iterations of Lanczos produces $N \times i$ matrix $V_i = [v_1, \dots, v_i]$, and $i \times i$ tridiagonal matrix T_i such that
$$AV_i = V_i T_i + \delta_{i+1} v_{i+1} e_i^T, \qquad T_i = V_i^* A V_i$$
- CG approximation x_i is obtained by solving the reduced model
$$T_i y_i = \|r_0\| e_1, \qquad x_i = x_0 + V_i y_i$$
- Connections with orthogonal polynomials, Stieltjes problem of moments, Gauss-Cristoffel quadrature, others (see 2013 book of Liesen and Strakoš)

The conjugate gradient method

A is symmetric positive definite, $\mathcal{C}_i = \mathcal{K}_i(A, r_0)$

$$r_i \perp \mathcal{K}_i(A, r_0) \quad \Longleftrightarrow \quad \|x - x_i\|_A = \min_{z \in x_0 + \mathcal{K}_i(A, r_0)} \|x - z\|_A$$

$$\Longrightarrow \quad r_{N+1} = 0$$

Connection with Lanczos

- With $v_1 = r_0/\|r_0\|$, i iterations of Lanczos produces $N \times i$ matrix $V_i = [v_1, \dots, v_i]$, and $i \times i$ tridiagonal matrix T_i such that
$$AV_i = V_i T_i + \delta_{i+1} v_{i+1} e_i^T, \qquad T_i = V_i^* A V_i$$
- CG approximation x_i is obtained by solving the reduced model
$$T_i y_i = \|r_0\| e_1, \qquad x_i = x_0 + V_i y_i$$
- Connections with orthogonal polynomials, Stieltjes problem of moments, Gauss-Cristoffel quadrature, others (see 2013 book of Liesen and Strakoš)

⇒ CG (and other Krylov subspace methods) are highly nonlinear

- Good for convergence, bad for ease of finite precision analysis

Implementation of CG

- Standard implementation due to Hestenes and Stiefel (1952) (HSCG)
- Uses three 2-term recurrences for updating x_i, r_i, p_i

$$r_0 = b - Ax_0, \quad p_0 = r_0$$

for i = 1:nmax

$$\alpha_{i-1} = \frac{r_{i-1}^T r_{i-1}}{p_{i-1}^T A p_{i-1}}$$

$$x_i = x_{i-1} + \alpha_{i-1} p_{i-1}$$

$$r_i = r_{i-1} - \alpha_{i-1} A p_{i-1}$$

$$\beta_i = \frac{r_i^T r_i}{r_{i-1}^T r_{i-1}}$$

$$p_i = r_i + \beta_i p_{i-1}$$

end

Implementation of CG

- Standard implementation due to Hestenes and Stiefel (1952) (HSCG)
- Uses three 2-term recurrences for updating x_i, r_i, p_i

$r_0 = b - Ax_0, \quad p_0 = r_0$

for i = 1:nmax

$$\alpha_{i-1} = \frac{r_{i-1}^T r_{i-1}}{p_{i-1}^T A p_{i-1}}$$

$$x_i = x_{i-1} + \alpha_{i-1} p_{i-1}$$

$$r_i = r_{i-1} - \alpha_{i-1} A p_{i-1}$$

$$\beta_i = \frac{r_i^T r_i}{r_{i-1}^T r_{i-1}}$$

$$p_i = r_i + \beta_i p_{i-1}$$

end

minimizes $\|x - x_i\|_A$ along line

$$z(\alpha) = x_{i-1} + \alpha p_{i-1}$$

Implementation of CG

- Standard implementation due to Hestenes and Stiefel (1952) (HSCG)
- Uses three 2-term recurrences for updating x_i, r_i, p_i

$r_0 = b - Ax_0, \;\; p_0 = r_0$
for i = 1:nmax

$$\alpha_{i-1} = \frac{r_{i-1}^T r_{i-1}}{p_{i-1}^T A p_{i-1}}$$

$$x_i = x_{i-1} + \alpha_{i-1} p_{i-1}$$

$$r_i = r_{i-1} - \alpha_{i-1} A p_{i-1}$$

$$\beta_i = \frac{r_i^T r_i}{r_{i-1}^T r_{i-1}}$$

$$p_i = r_i + \beta_i p_{i-1}$$

end

minimizes $\|x - x_i\|_A$ along line
$$z(\alpha) = x_{i-1} + \alpha p_{i-1}$$

If
$$p_i \perp_A p_j \text{ for } i \neq j,$$

1-dimensional minimizations in each iteration give i-dimensional minimization over the whole subspace

$$x_0 + \mathcal{K}_i(A, r_0) = x_0 + \operatorname{span}\{p_0, \dots p_{i-1}\}$$

Communication in CG

Projection process in terms of communication:

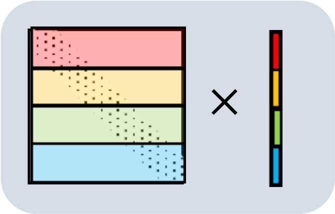
Communication in CG

Projection process in terms of communication:

"Add a dimension to $\mathcal{K}_i$"

$\longrightarrow$ Sparse matrix-vector multiplication (SpMV)

- Must communicate vector entries w/ neighboring processors (P2P communication)



SpMV

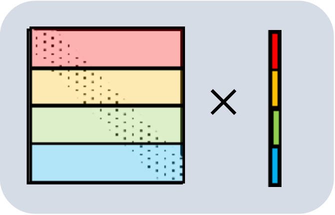
Communication in CG

Projection process in terms of communication:

"Add a dimension to $\mathcal{K}_i$"

$\rightarrow$ Sparse matrix-vector multiplication (SpMV)

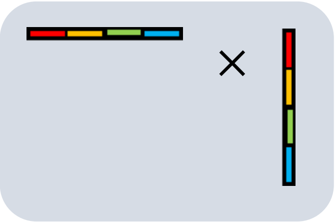
- Must communicate vector entries w/ neighboring processors (P2P communication)



"Orthogonalize with respect to $\mathcal{C}_i$"

$\rightarrow$ Inner products

- global synchronization (MPI_Allreduce)
- all processors must exchange data and wait for *all* communication to finish before proceeding



SpMV

orthogonalize

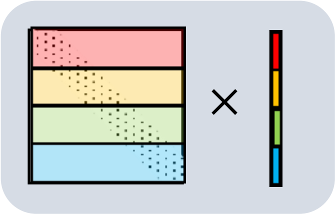
Communication in CG

Projection process in terms of communication:

"Add a dimension to $\mathcal{K}_i$"

$\rightarrow$ Sparse matrix-vector multiplication (SpMV)

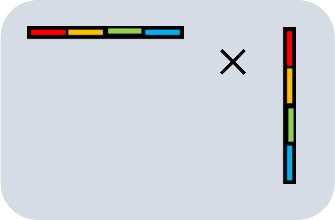
- Must communicate vector entries w/ neighboring processors (P2P communication)

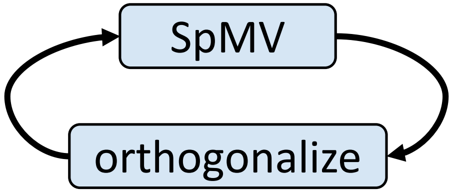


"Orthogonalize with respect to $\mathcal{C}_i$"

$\rightarrow$ Inner products

- global synchronization (MPI_Allreduce)
- all processors must exchange data and wait for *all* communication to finish before proceeding





Dependencies between communication-bound kernels in each iteration limit performance!

Communication in HSCG

$r_0 = b - Ax_0, \;\; p_0 = r_0$
for i = 1:nmax

$$\alpha_{i-1} = \frac{r_{i-1}^T r_{i-1}}{p_{i-1}^T A p_{i-1}}$$

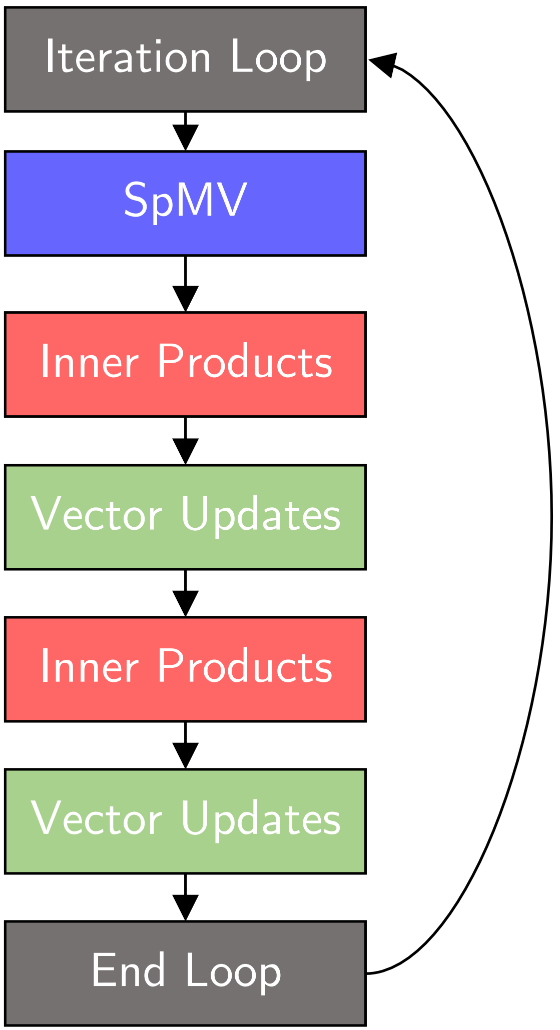
$$x_i = x_{i-1} + \alpha_{i-1} p_{i-1}$$

$$r_i = r_{i-1} - \alpha_{i-1} A p_{i-1}$$

$$\beta_i = \frac{r_i^T r_i}{r_{i-1}^T r_{i-1}}$$

$$p_i = r_i + \beta_i p_{i-1}$$

end



Communication in HSCG

$r_0 = b - Ax_0, \;\; p_0 = r_0$

for i = 1:nmax

$$\alpha_{i-1} = \frac{r_{i-1}^T r_{i-1}}{p_{i-1}^T A p_{i-1}}$$

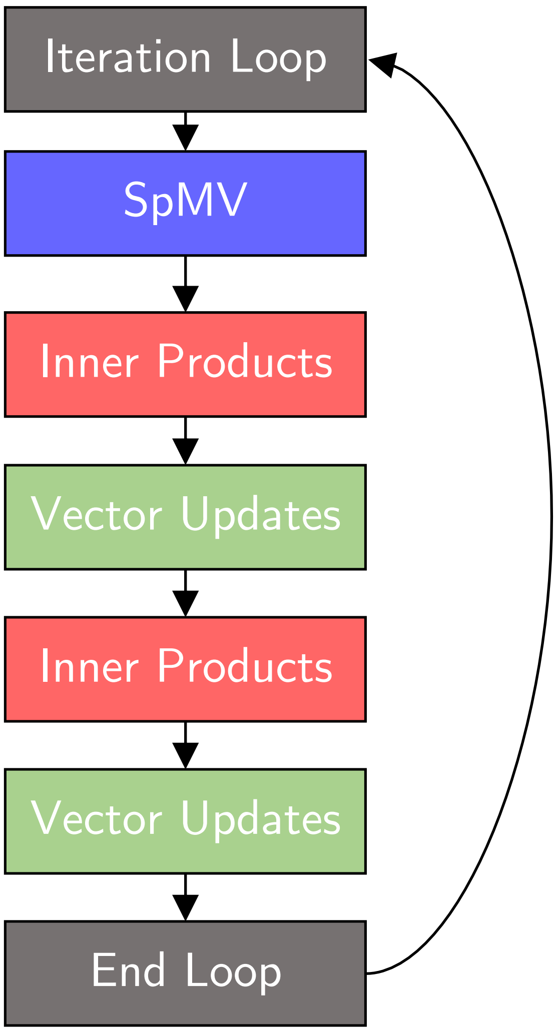
$$x_i = x_{i-1} + \alpha_{i-1} p_{i-1}$$

$$r_i = r_{i-1} - \alpha_{i-1} A p_{i-1}$$

$$\beta_i = \frac{r_i^T r_i}{r_{i-1}^T r_{i-1}}$$

$$p_i = r_i + \beta_i p_{i-1}$$

end



Communication in HSCG

$r_0 = b - Ax_0, \;\; p_0 = r_0$

for i = 1:nmax

$$\alpha_{i-1} = \frac{r_{i-1}^T r_{i-1}}{p_{i-1}^T A p_{i-1}}$$

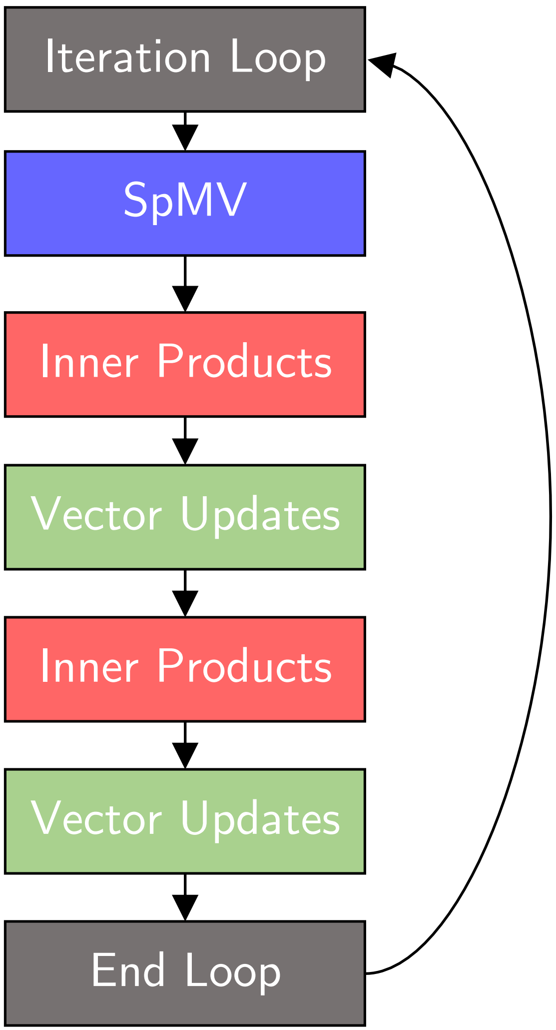
$$x_i = x_{i-1} + \alpha_{i-1} p_{i-1}$$

$$r_i = r_{i-1} - \alpha_{i-1} A p_{i-1}$$

$$\beta_i = \frac{r_i^T r_i}{r_{i-1}^T r_{i-1}}$$

$$p_i = r_i + \beta_i p_{i-1}$$

end



Communication in HSCG

$r_0 = b - Ax_0, \;\; p_0 = r_0$

for i = 1:nmax

$$\alpha_{i-1} = \frac{r_{i-1}^T r_{i-1}}{p_{i-1}^T A p_{i-1}}$$

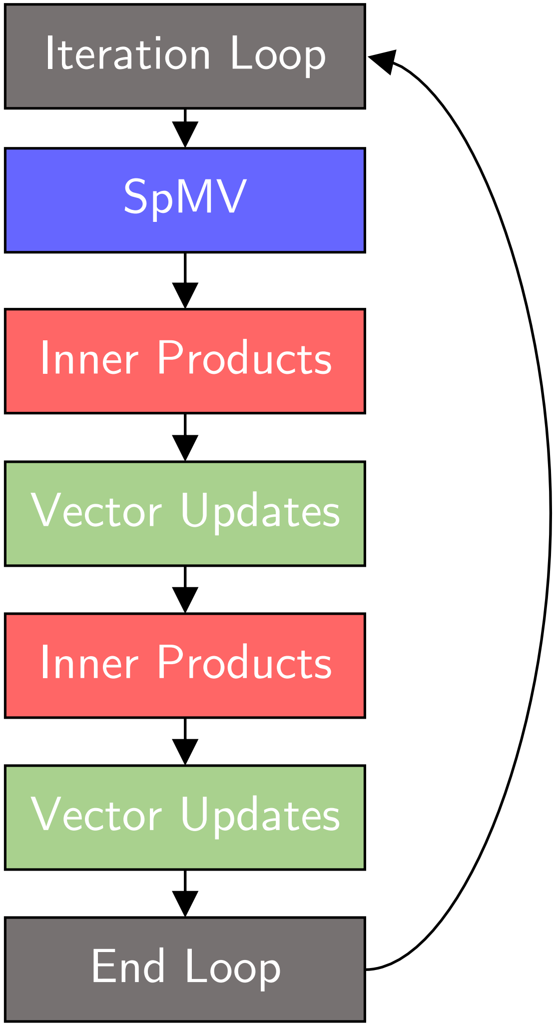
$$x_i = x_{i-1} + \alpha_{i-1} p_{i-1}$$

$$r_i = r_{i-1} - \alpha_{i-1} A p_{i-1}$$

$$\beta_i = \frac{r_i^T r_i}{r_{i-1}^T r_{i-1}}$$

$$p_i = r_i + \beta_i p_{i-1}$$

end



Communication in HSCG

$r_0 = b - Ax_0, \;\; p_0 = r_0$

for i = 1:nmax

$$\alpha_{i-1} = \frac{r_{i-1}^T r_{i-1}}{p_{i-1}^T A p_{i-1}}$$

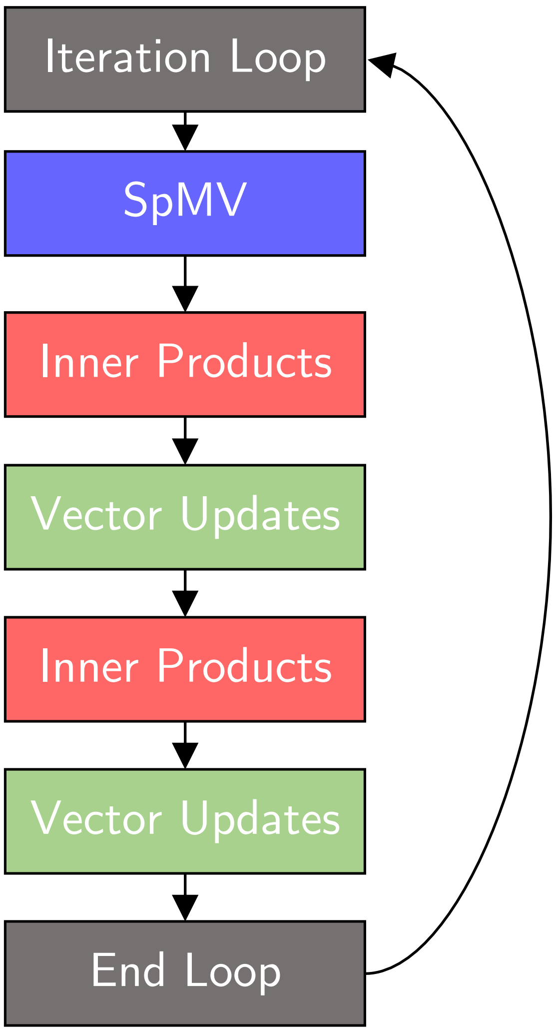
$$x_i = x_{i-1} + \alpha_{i-1} p_{i-1}$$

$$r_i = r_{i-1} - \alpha_{i-1} A p_{i-1}$$

$$\beta_i = \frac{r_i^T r_i}{r_{i-1}^T r_{i-1}}$$

$$p_i = r_i + \beta_i p_{i-1}$$

end



Communication in HSCG

$r_0 = b - Ax_0, \quad p_0 = r_0$

for i = 1:nmax

$$\alpha_{i-1} = \frac{r_{i-1}^T r_{i-1}}{p_{i-1}^T A p_{i-1}}$$

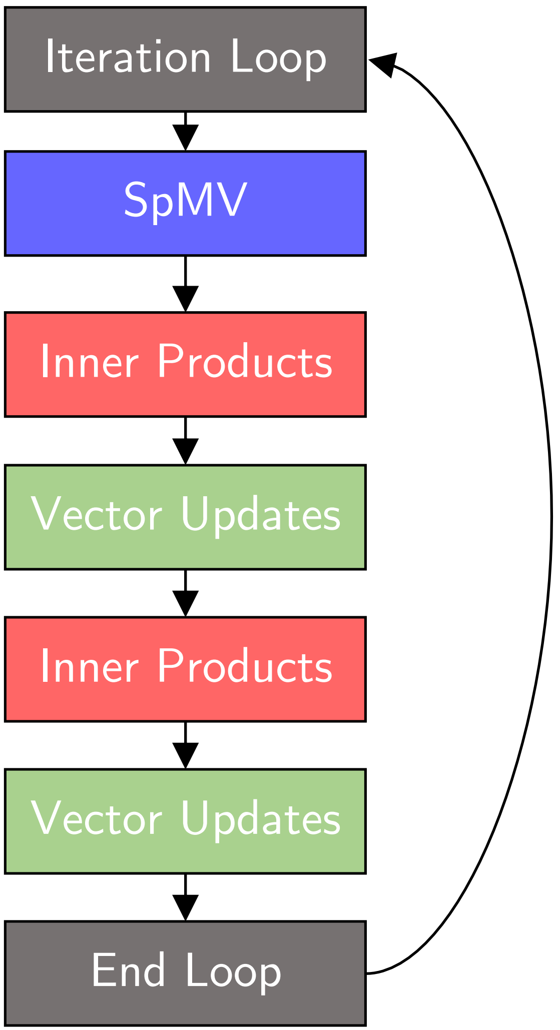
$$x_i = x_{i-1} + \alpha_{i-1} p_{i-1}$$

$$r_i = r_{i-1} - \alpha_{i-1} A p_{i-1}$$

$$\beta_i = \frac{r_i^T r_i}{r_{i-1}^T r_{i-1}}$$

$$p_i = r_i + \beta_i p_{i-1}$$

end



Future exascale systems

	Petascale Systems (2009)
System Peak	$2 \cdot 10^{15}$ flops/s
Node Memory Bandwidth	25 GB/s
Total Node Interconnect Bandwidth	3.5 GB/s
Memory Latency	100 ns
Interconnect Latency	1 μs

*Sources: from P. Beckman (ANL), J. Shalf (LBL), and D. Unat (LBL)

Future exascale systems

	Petascale Systems (2009)	Predicted Exascale Systems
System Peak	$2 \cdot 10^{15}$ flops/s	10^{18} flops/s
Node Memory Bandwidth	25 GB/s	0.4-4 TB/s
Total Node Interconnect Bandwidth	3.5 GB/s	100-400 GB/s
Memory Latency	100 ns	50 ns
Interconnect Latency	1 μs	0.5 μs

*Sources: from P. Beckman (ANL), J. Shalf (LBL), and D. Unat (LBL)

Future exascale systems

	Petascale Systems (2009)	Predicted Exascale Systems	Factor Improvement
System Peak	$2 \cdot 10^{15}$ flops/s	10^{18} flops/s	~1000
Node Memory Bandwidth	25 GB/s	0.4-4 TB/s	~10-100
Total Node Interconnect Bandwidth	3.5 GB/s	100-400 GB/s	~100
Memory Latency	100 ns	50 ns	~1
Interconnect Latency	1 μs	0.5 μs	~1

*Sources: from P. Beckman (ANL), J. Shalf (LBL), and D. Unat (LBL)

Future exascale systems

	Petascale Systems (2009)	Predicted Exascale Systems	Factor Improvement
System Peak	$2 \cdot 10^{15}$ flops/s	10^{18} flops/s	~1000
Node Memory Bandwidth	25 GB/s	0.4-4 TB/s	~10-100
Total Node Interconnect Bandwidth	3.5 GB/s	100-400 GB/s	~100
Memory Latency	100 ns	50 ns	~1
Interconnect Latency	1 μs	0.5 μs	~1

*Sources: from P. Beckman (ANL), J. Shalf (LBL), and D. Unat (LBL)

- Gaps between communication/computation cost only growing larger in future systems
- **Reducing time spent moving data/waiting for data will be essential for applications at exascale!**

Synchronization-reducing variants

Communication cost has motivated many approaches to reducing synchronization in CG:

Synchronization-reducing variants

Communication cost has motivated many approaches to reducing synchronization in CG:

- Early work: CG with a single synchronization point per iteration
  - 3-term recurrence CG
  - Using modified computation of recurrence coefficients
  - Using auxiliary vectors

Synchronization-reducing variants

Communication cost has motivated many approaches to reducing synchronization in CG:

- Early work: CG with a single synchronization point per iteration
  - 3-term recurrence CG
  - Using modified computation of recurrence coefficients
  - Using auxiliary vectors

- Pipelined Krylov subspace methods
  - Uses modified coefficients and auxiliary vectors to reduce synchronization points to 1 per iteration
  - Modifications also allow decoupling of SpMV and inner products - enables overlapping

Synchronization-reducing variants

Communication cost has motivated many approaches to reducing synchronization in CG:

- Early work: CG with a single synchronization point per iteration
  - 3-term recurrence CG
  - Using modified computation of recurrence coefficients
  - Using auxiliary vectors

- Pipelined Krylov subspace methods
  - Uses modified coefficients and auxiliary vectors to reduce synchronization points to 1 per iteration
  - Modifications also allow decoupling of SpMV and inner products - enables overlapping

- s-step Krylov subspace methods
  - Compute iterations in blocks of s using a different Krylov subspace basis
  - Enables one synchronization per s iterations

The effects of finite precision

Well-known that roundoff error has two effects:

1. Delay of convergence
    - No longer have exact Krylov subspace
    - Can lose numerical rank deficiency
    - Residuals no longer orthogonal - Minimization no longer exact!

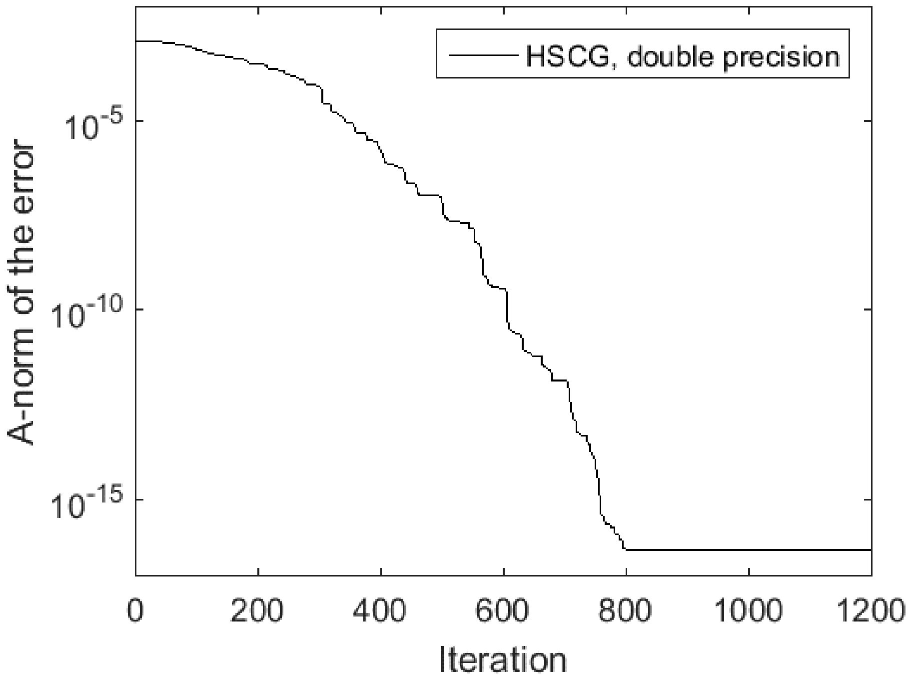
2. Loss of attainable accuracy
    - Rounding errors cause true residual $b - Ax_i$ and updated residual r_i deviate!

The effects of finite precision

Well-known that roundoff error has two effects:

1. Delay of convergence
   - No longer have exact Krylov subspace
   - Can lose numerical rank deficiency
   - Residuals no longer orthogonal - Minimization no longer exact!

2. Loss of attainable accuracy
   - Rounding errors cause true residual $b - Ax_i$ and updated residual r_i deviate!



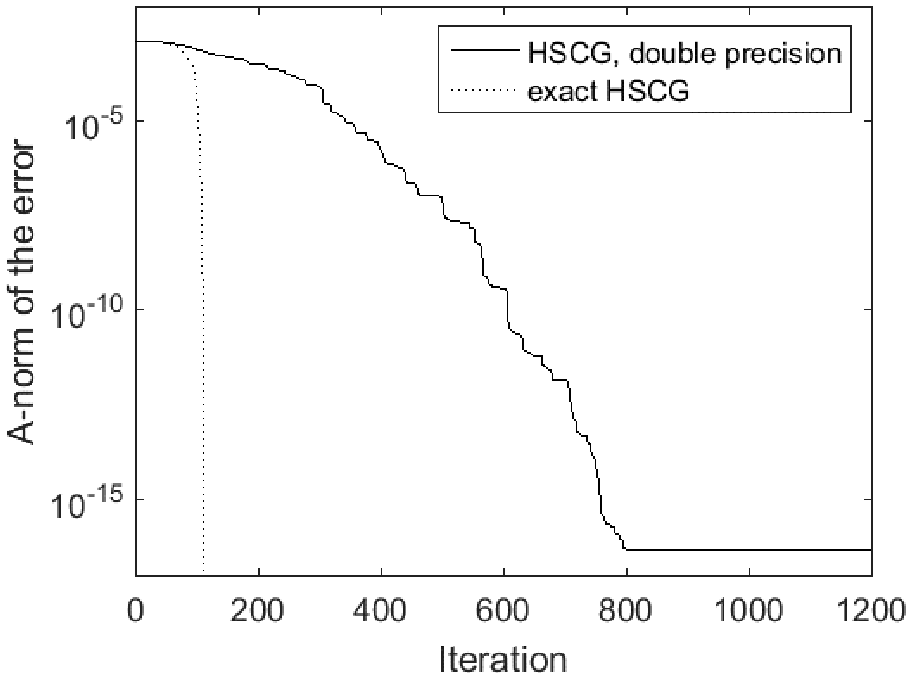
A: bcsstk03 from UFSMC, b: equal components in the eigenbasis of A and $\|b\| = 1$
$N = 112, \kappa(A) \approx 7e6$

The effects of finite precision

Well-known that roundoff error has two effects:

1. Delay of convergence
   - No longer have exact Krylov subspace
   - Can lose numerical rank deficiency
   - Residuals no longer orthogonal - Minimization no longer exact!

2. Loss of attainable accuracy
   - Rounding errors cause true residual $b - Ax_i$ and updated residual r_i deviate!



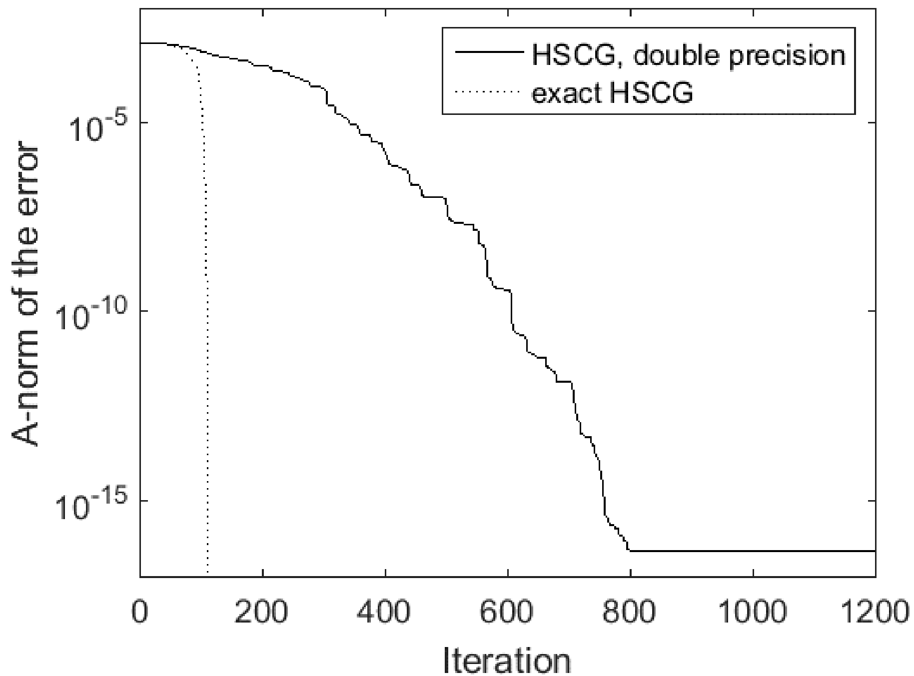
A: bcsstk03 from UFSMC, b: equal components in the eigenbasis of A and $\|b\| = 1$
$N = 112, \kappa(A) \approx 7e6$

The effects of finite precision

Well-known that roundoff error has two effects:

1. Delay of convergence
    - No longer have exact Krylov subspace
    - Can lose numerical rank deficiency
    - Residuals no longer orthogonal - Minimization no longer exact!

2. Loss of attainable accuracy
    - Rounding errors cause true residual $b - Ax_i$ and updated residual r_i deviate!



A: bcsstk03 from UFSMC, b: equal components in the eigenbasis of A and $\|b\| = 1$
$N = 112, \kappa(A) \approx 7e6$

Much work on these results for CG; See Meurant and Strakoš (2006) for a thorough summary of early developments in finite precision analysis of Lanczos and CG

Optimizing high performance iterative solvers

- Synchronization-reducing variants are designed to reduce the time/iteration

Optimizing high performance iterative solvers

- Synchronization-reducing variants are designed to reduce the time/iteration
- But this is not the whole story!

Optimizing high performance iterative solvers

- Synchronization-reducing variants are designed to reduce the time/iteration
- But this is not the whole story!
- What we really want to minimize is the **runtime, subject to some constraint on accuracy,**

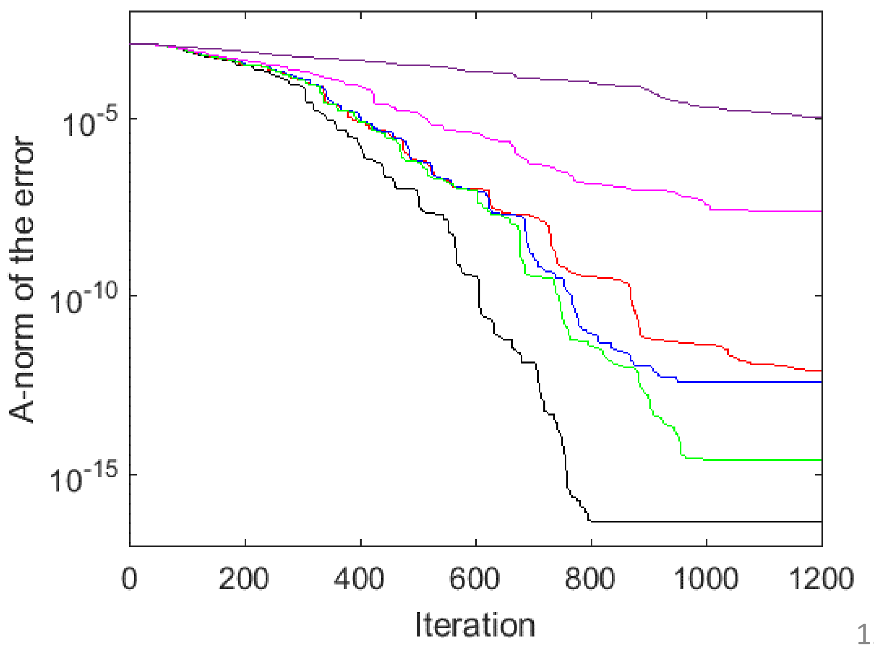
runtime = (time/iteration) x (# iterations)

Optimizing high performance iterative solvers

- Synchronization-reducing variants are designed to reduce the time/iteration
- But this is not the whole story!
- What we really want to minimize is the **runtime, subject to some constraint on accuracy,**

runtime = (time/iteration) x (# iterations)

- Changes to how the recurrences are computed can exacerbate finite precision effects of convergence delay and loss of accuracy



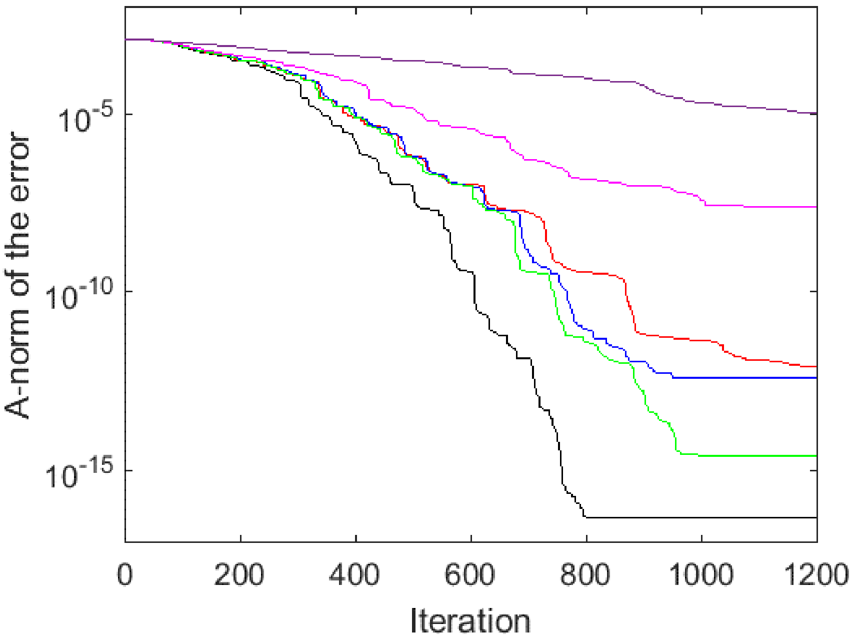
Optimizing high performance iterative solvers

- Synchronization-reducing variants are designed to reduce the time/iteration
- But this is not the whole story!
- What we really want to minimize is the **runtime, subject to some constraint on accuracy,**

runtime = (time/iteration) x (# iterations)

- Changes to how the recurrences are computed can exacerbate finite precision effects of convergence delay and loss of accuracy

- Crucial that we understand and take into account how algorithm modifications will affect the convergence rate and attainable accuracy!



Maximum attainable accuracy

- Accuracy depends on the size of the true residual: $\|b - A\hat{x}_i\|$

Maximum attainable accuracy

- Accuracy depends on the size of the true residual: $\|b - A\hat{x}_i\|$
- Rounding errors cause the **true residual, $\boldsymbol{b - A\hat{x}_i}$**, and the **updated residual, $\boldsymbol{\hat{r}_i}$**, to deviate

Maximum attainable accuracy

- Accuracy depends on the size of the true residual: $\|b - A\hat{x}_i\|$
- Rounding errors cause the **true residual, $\boldsymbol{b - A\hat{x}_i}$**, and the **updated residual, $\boldsymbol{\hat{r}_i}$**, to deviate

- Writing $b - A\hat{x}_i = \hat{r}_i + b - A\hat{x}_i - \hat{r}_i$,

$$\|b - A\hat{x}_i\| \leq \|\hat{r}_i\| + \|b - A\hat{x}_i - \hat{r}_i\|$$

Maximum attainable accuracy

- Accuracy depends on the size of the true residual: $\|b - A\hat{x}_i\|$
- Rounding errors cause the **true residual,** $\boldsymbol{b - A\hat{x}_i}$, and the **updated residual,** $\boldsymbol{\hat{r}_i}$, to deviate

- Writing $b - A\hat{x}_i = \hat{r}_i + b - A\hat{x}_i - \hat{r}_i$,

$$\|b - A\hat{x}_i\| \leq \|\hat{r}_i\| + \|b - A\hat{x}_i - \hat{r}_i\|$$

  - As $\|\hat{r}_i\| \to 0$, $\|b - A\hat{x}_i\|$ depends on $\|b - A\hat{x}_i - \hat{r}_i\|$

Maximum attainable accuracy

- Accuracy depends on the size of the true residual: $\|b - A\hat{x}_i\|$
- Rounding errors cause the **true residual,** $\boldsymbol{b - A\hat{x}_i}$, and the **updated residual,** $\boldsymbol{\hat{r}_i}$, to deviate

- Writing $b - A\hat{x}_i = \hat{r}_i + b - A\hat{x}_i - \hat{r}_i$,

$$\|b - A\hat{x}_i\| \leq \|\hat{r}_i\| + \|b - A\hat{x}_i - \hat{r}_i\|$$

  - As $\|\hat{r}_i\| \rightarrow 0$, $\|b - A\hat{x}_i\|$ depends on $\|b - A\hat{x}_i - \hat{r}_i\|$

- Many results on bounding attainable accuracy, e.g.: Greenbaum (1989, 1994, 1997), Sleijpen, van der Vorst and Fokkema (1994), Sleijpen, van der Vorst and Modersitzki (2001), Björck, Elfving and Strakoš (1998) and Gutknecht and Strakoš (2000).

Maximum attainable accuracy of HSCG

- In finite precision HSCG, iterates are updated by

$$\hat{x}_i = \hat{x}_{i-1} + \hat{\alpha}_{i-1}\hat{p}_{i-1} - \boldsymbol{\delta x_i}$$ and $$\hat{r}_i = \hat{r}_{i-1} - \hat{\alpha}_{i-1}A\hat{p}_{i-1} - \boldsymbol{\delta r_i}$$

Maximum attainable accuracy of HSCG

- In finite precision HSCG, iterates are updated by

$$\hat{x}_i = \hat{x}_{i-1} + \hat{\alpha}_{i-1}\hat{p}_{i-1} - \boldsymbol{\delta x_i}$$

and

$$\hat{r}_i = \hat{r}_{i-1} - \hat{\alpha}_{i-1}A\hat{p}_{i-1} - \boldsymbol{\delta r_i}$$

- Let $f_i \equiv b - A\hat{x}_i - \hat{r}_i$

Maximum attainable accuracy of HSCG

- In finite precision HSCG, iterates are updated by

$$\hat{x}_i = \hat{x}_{i-1} + \hat{\alpha}_{i-1}\hat{p}_{i-1} - \boldsymbol{\delta x_i}$$ and $$\hat{r}_i = \hat{r}_{i-1} - \hat{\alpha}_{i-1}A\hat{p}_{i-1} - \boldsymbol{\delta r_i}$$

- Let $f_i \equiv b - A\hat{x}_i - \hat{r}_i$

$$f_i = b - A(\hat{x}_{i-1} + \hat{\alpha}_{i-1}\hat{p}_{i-1} - \delta x_i) - (\hat{r}_{i-1} - \hat{\alpha}_{i-1}A\hat{p}_{i-1} - \delta r_i)$$

Maximum attainable accuracy of HSCG

- In finite precision HSCG, iterates are updated by

$$\hat{x}_i = \hat{x}_{i-1} + \hat{\alpha}_{i-1}\hat{p}_{i-1} - \boldsymbol{\delta x_i}$$

and

$$\hat{r}_i = \hat{r}_{i-1} - \hat{\alpha}_{i-1}A\hat{p}_{i-1} - \boldsymbol{\delta r_i}$$

- Let $f_i \equiv b - A\hat{x}_i - \hat{r}_i$

$$\begin{aligned} f_i &= b - A(\hat{x}_{i-1} + \hat{\alpha}_{i-1}\hat{p}_{i-1} - \delta x_i) - (\hat{r}_{i-1} - \hat{\alpha}_{i-1}A\hat{p}_{i-1} - \delta r_i) \\ &= f_{i-1} + A\delta x_i + \delta r_i \end{aligned}$$

Maximum attainable accuracy of HSCG

- In finite precision HSCG, iterates are updated by

$$\hat{x}_i = \hat{x}_{i-1} + \hat{\alpha}_{i-1}\hat{p}_{i-1} - \boldsymbol{\delta x_i} \quad \text{and} \quad \hat{r}_i = \hat{r}_{i-1} - \hat{\alpha}_{i-1}A\hat{p}_{i-1} - \boldsymbol{\delta r_i}$$

- Let $f_i \equiv b - A\hat{x}_i - \hat{r}_i$

$$\begin{aligned} f_i &= b - A(\hat{x}_{i-1} + \hat{\alpha}_{i-1}\hat{p}_{i-1} - \delta x_i) - (\hat{r}_{i-1} - \hat{\alpha}_{i-1}A\hat{p}_{i-1} - \delta r_i) \\ &= f_{i-1} + A\delta x_i + \delta r_i \\ &= f_0 + \sum_{m=1}^{i}(A\delta x_m + \delta r_m) \end{aligned}$$

Maximum attainable accuracy of HSCG

- In finite precision HSCG, iterates are updated by

$$\hat{x}_i = \hat{x}_{i-1} + \hat{\alpha}_{i-1}\hat{p}_{i-1} - \boldsymbol{\delta x_i}$$ and $$\hat{r}_i = \hat{r}_{i-1} - \hat{\alpha}_{i-1}A\hat{p}_{i-1} - \boldsymbol{\delta r_i}$$

- Let $f_i \equiv b - A\hat{x}_i - \hat{r}_i$

$$\begin{aligned} f_i &= b - A(\hat{x}_{i-1} + \hat{\alpha}_{i-1}\hat{p}_{i-1} - \delta x_i) - (\hat{r}_{i-1} - \hat{\alpha}_{i-1}A\hat{p}_{i-1} - \delta r_i) \\ &= f_{i-1} + A\delta x_i + \delta r_i \\ &= f_0 + \textstyle\sum_{m=1}^{i}(A\delta x_m + \delta r_m) \end{aligned}$$

$\|f_i\| \le O(\varepsilon)\sum_{m=0}^{i} N_A\|A\|\|\hat{x}_m\| + \|\hat{r}_m\|$ van der Vorst and Ye, 2000

$\|f_i\| \le O(\varepsilon)\|A\|\left(\|x\| + \max_{m=0,\dots,i}\|\hat{x}_m\|\right)$ Greenbaum, 1997

$\|f_i\| \le O(\varepsilon)N_A\||A|\|\|A^{-1}\|\sum_{m=0}^{i}\|\hat{r}_m\|$ Sleijpen and van der Vorst, 1995

Early approaches to reducing synchronization

- Goal: Reduce the 2 synchronization points per iteration in HSCG to 1 synchronization point per iteration

Early approaches to reducing synchronization

- Goal: Reduce the 2 synchronization points per iteration in HSCG to 1 synchronization point per iteration
- Compute β_i from α_{i-1} and Ap_{i-1} using relation

$$\|r_i\|^2 = \alpha_{i-1}^2 \|Ap_{i-1}\|^2 - \|r_{i-1}\|^2$$

  - Can then also merge the updates of x_i, r_i, and p_i
  - Developed independently by Johnson (1983, 1984), van Rosendale (1983, 1984), Saad (1985)
  - Many other similar approaches

Early approaches to reducing synchronization

- Goal: Reduce the 2 synchronization points per iteration in HSCG to 1 synchronization point per iteration
- Compute β_i from α_{i-1} and Ap_{i-1} using relation

$$\|r_i\|^2 = \alpha_{i-1}^2 \|Ap_{i-1}\|^2 - \|r_{i-1}\|^2$$

  - Can then also merge the updates of x_i, r_i, and p_i
  - Developed independently by Johnson (1983, 1984), van Rosendale (1983, 1984), Saad (1985)
  - Many other similar approaches

- Could also compute α_{i-1} from β_{i-1}:

$$\alpha_{i-1} = \left(\frac{r_{i-1}^T A r_{i-1}}{r_{i-1}^T r_{i-1}} - \frac{\beta_{i-1}}{\alpha_{i-2}} \right)^{-1}$$

Modified recurrence coefficient computation

- What is the effect of changing the way the recurrence coefficients (α and β) are computed in HSCG?

Modified recurrence coefficient computation

- What is the effect of changing the way the recurrence coefficients (α and β) are computed in HSCG?

- Notice that neither α nor β appear in the bounds on $\|f_i\|$

$$f_i = b - A\hat{x}_i - \hat{r}_i$$
$$= b - A(\hat{x}_{i-1} + \hat{\alpha}_{i-1}\hat{p}_{i-1} - \delta x_i) - (\hat{r}_{i-1} - \hat{\alpha}_{i-1}A\hat{p}_{i-1} - \delta r_i)$$

Modified recurrence coefficient computation

- What is the effect of changing the way the recurrence coefficients (α and β) are computed in HSCG?

- Notice that neither α nor β appear in the bounds on $\|f_i\|$

$$\begin{aligned} f_i &= b - A\hat{x}_i - \hat{r}_i \\ &= b - A(\hat{x}_{i-1} + \hat{\alpha}_{i-1}\hat{p}_{i-1} - \delta x_i) - (\hat{r}_{i-1} - \hat{\alpha}_{i-1}A\hat{p}_{i-1} - \delta r_i) \end{aligned}$$

- As long as the same $\hat{\alpha}_{i-1}$ is used in updating $\hat{x}_i$ and $\hat{r}_i$,

$$f_i = f_{i-1} + A\delta x_i + \delta r_i$$

still holds

Modified recurrence coefficient computation

- What is the effect of changing the way the recurrence coefficients (α and β) are computed in HSCG?

- Notice that neither α nor β appear in the bounds on $\|f_i\|$

$$\begin{aligned} f_i &= b - A\hat{x}_i - \hat{r}_i \\ &= b - A(\hat{x}_{i-1} + \hat{\alpha}_{i-1}\hat{p}_{i-1} - \delta x_i) - (\hat{r}_{i-1} - \hat{\alpha}_{i-1}A\hat{p}_{i-1} - \delta r_i) \end{aligned}$$

- As long as the same $\hat{\alpha}_{i-1}$ is used in updating $\hat{x}_i$ and $\hat{r}_i$,

$$f_i = f_{i-1} + A\delta x_i + \delta r_i$$

still holds

- Rounding errors made in computing $\hat{\alpha}_{i-1}$ do not contribute to the residual gap

Modified recurrence coefficient computation

- What is the effect of changing the way the recurrence coefficients (α and β) are computed in HSCG?

- Notice that neither α nor β appear in the bounds on $\|f_i\|$

$$\begin{aligned} f_i &= b - A\hat{x}_i - \hat{r}_i \\ &= b - A(\hat{x}_{i-1} + \hat{\alpha}_{i-1}\hat{p}_{i-1} - \delta x_i) - (\hat{r}_{i-1} - \hat{\alpha}_{i-1}A\hat{p}_{i-1} - \delta r_i) \end{aligned}$$

- As long as the same $\hat{\alpha}_{i-1}$ is used in updating $\hat{x}_i$ and $\hat{r}_i$,

$$f_i = f_{i-1} + A\delta x_i + \delta r_i$$

still holds

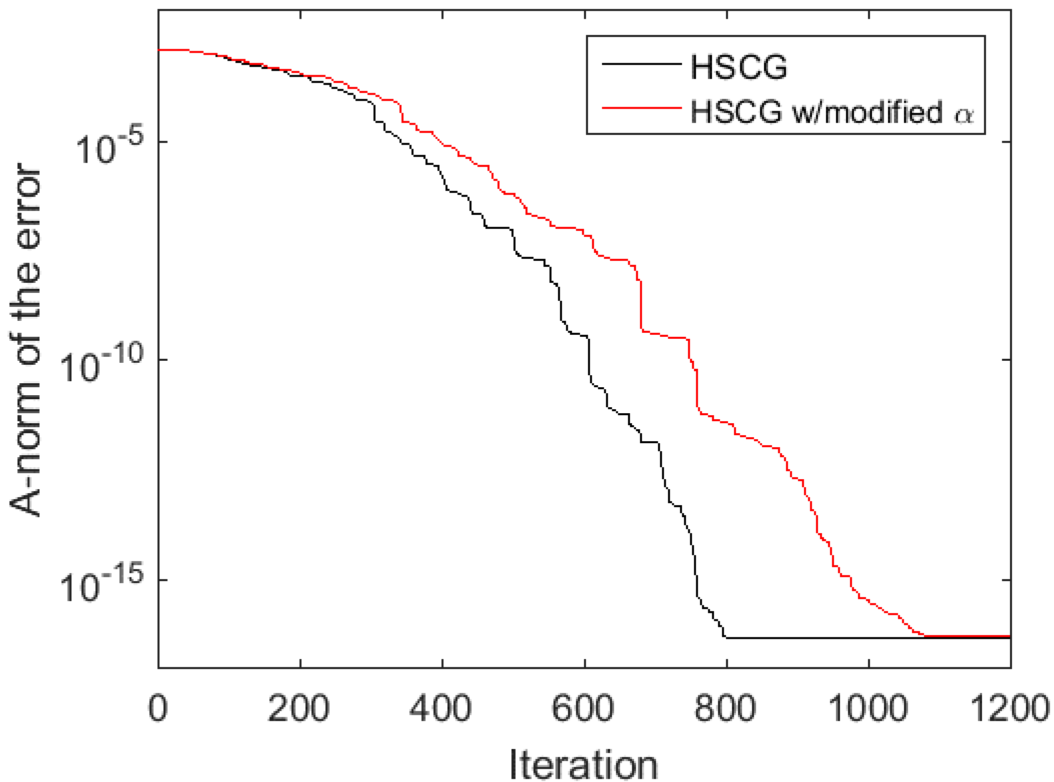
- Rounding errors made in computing $\hat{\alpha}_{i-1}$ do not contribute to the residual gap

- But may change computed $\hat{x}_i$, $\hat{r}_i$, which can affect convergence rate...

Modified recurrence coefficient computation

Example: HSCG with modified formula for α_{i-1}

$$\alpha_{i-1} = \left(\frac{r_{i-1}^T A r_{i-1}}{r_{i-1}^T r_{i-1}} - \frac{\beta_{i-1}}{\alpha_{i-2}} \right)^{-1}$$



CG with two three-term recurrences (STCG)

- HSCG recurrences can be written as

$$AP_i = R_{i+1}\underline{L}_i, \qquad R_i = P_i U_i$$

we can combine these to obtain a 3-term recurrence for the residuals (STCG):

$$AR_i = R_{i+1}\underline{T}_i, \qquad \underline{T}_i = \underline{L}_i U_i$$

CG with two three-term recurrences (STCG)

- HSCG recurrences can be written as

$$AP_i = R_{i+1}\underline{L}_i, \qquad R_i = P_i U_i$$

we can combine these to obtain a 3-term recurrence for the residuals (STCG):

$$AR_i = R_{i+1}\underline{T}_i, \qquad \underline{T}_i = \underline{L}_i U_i$$

- First developed by Stiefel (1952/53), also Rutishauser (1959) and Hageman and Young (1981)
- Motivated by relation to three-term recurrences for orthogonal polynomials

$r_0 = b - Ax_0, \;\; p_0 = r_0, \;\; x_{-1} = x_0, \;\; r_{-1} = r_0, \;\; e_{-1} = 0$
for i = 1:nmax

$$q_{i-1} = \frac{(r_{i-1}, Ar_{i-1})}{(r_{i-1}, r_{i-1})} - e_{i-2}$$

$$x_i = x_{i-1} + \frac{1}{q_{i-1}}(r_{i-1} + e_{i-2}(x_{i-1} - x_{i-2}))$$

$$r_i = r_{i-1} + \frac{1}{q_{i-1}}(-Ar_{i-1} + e_{i-2}(r_{i-1} - r_{i-2}))$$

$$e_{i-1} = q_{i-1}\frac{(r_i, r_i)}{(r_{i-1}, r_{i-1})}$$

end

CG with two three-term recurrences (STCG)

- HSCG recurrences can be written as

$$AP_i = R_{i+1}\underline{L}_i, \qquad R_i = P_i U_i$$

we can combine these to obtain a 3-term recurrence for the residuals (STCG):

$$AR_i = R_{i+1}\underline{T}_i, \qquad \underline{T}_i = \underline{L}_i U_i$$

- First developed by Stiefel (1952/53), also Rutishauser (1959) and Hageman and Young (1981)
- Motivated by relation to three-term recurrences for orthogonal polynomials

$r_0 = b - Ax_0, \;\; p_0 = r_0, \;\; x_{-1} = x_0, \;\; r_{-1} = r_0, \;\; e_{-1} = 0$

for i = 1:nmax

$$q_{i-1} = \frac{(r_{i-1}, Ar_{i-1})}{(r_{i-1}, r_{i-1})} - e_{i-2}$$

$$x_i = x_{i-1} + \frac{1}{q_{i-1}}\left(r_{i-1} + e_{i-2}(x_{i-1} - x_{i-2})\right)$$

$$r_i = r_{i-1} + \frac{1}{q_{i-1}}\left(-Ar_{i-1} + e_{i-2}(r_{i-1} - r_{i-2})\right)$$

$$e_{i-1} = q_{i-1}\frac{(r_i, r_i)}{(r_{i-1}, r_{i-1})}$$

end

Can be accomplished with a single synchronization point on parallel computers (Strakoš 1985, 1987)

CG with two three-term recurrences (STCG)

- HSCG recurrences can be written as

$$AP_i = R_{i+1}\underline{L}_i, \qquad R_i = P_i U_i$$

we can combine these to obtain a 3-term recurrence for the residuals (STCG):

$$AR_i = R_{i+1}\underline{T}_i, \qquad \underline{T}_i = \underline{L}_i U_i$$

- First developed by Stiefel (1952/53), also Rutishauser (1959) and Hageman and Young (1981)
- Motivated by relation to three-term recurrences for orthogonal polynomials

$r_0 = b - Ax_0,\ \ p_0 = r_0,\ \ x_{-1} = x_0,\ \ r_{-1} = r_0,\ \ e_{-1} = 0$

for i = 1:nmax

$$q_{i-1} = \frac{(r_{i-1}, Ar_{i-1})}{(r_{i-1}, r_{i-1})} - e_{i-2}$$

$$x_i = x_{i-1} + \frac{1}{q_{i-1}}\left(r_{i-1} + e_{i-2}(x_{i-1} - x_{i-2})\right)$$

$$r_i = r_{i-1} + \frac{1}{q_{i-1}}\left(-Ar_{i-1} + e_{i-2}(r_{i-1} - r_{i-2})\right)$$

$$e_{i-1} = q_{i-1}\frac{(r_i, r_i)}{(r_{i-1}, r_{i-1})}$$

end

Can be accomplished with a single synchronization point on parallel computers (Strakoš 1985, 1987)

- Similar approach (computing α_i using β_{i-1}) used by D'Azevedo, Eijkhout, Romaine (1992, 1993)

Attainable accuracy of STCG

- Analyzed by Gutknecht and Strakoš (2000)
- Attainable accuracy for STCG can be much worse than for HSCG

Attainable accuracy of STCG

- Analyzed by Gutknecht and Strakoš (2000)
- Attainable accuracy for STCG can be much worse than for HSCG

- Residual gap bounded by sum of local errors PLUS local errors multiplied by factors which depend on

$$\max_{0 \le \ell < j \le i} \frac{\left\| r_j \right\|^2}{\left\| r_\ell \right\|^2}$$

Attainable accuracy of STCG

- Analyzed by Gutknecht and Strakoš (2000)
- Attainable accuracy for STCG can be much worse than for HSCG

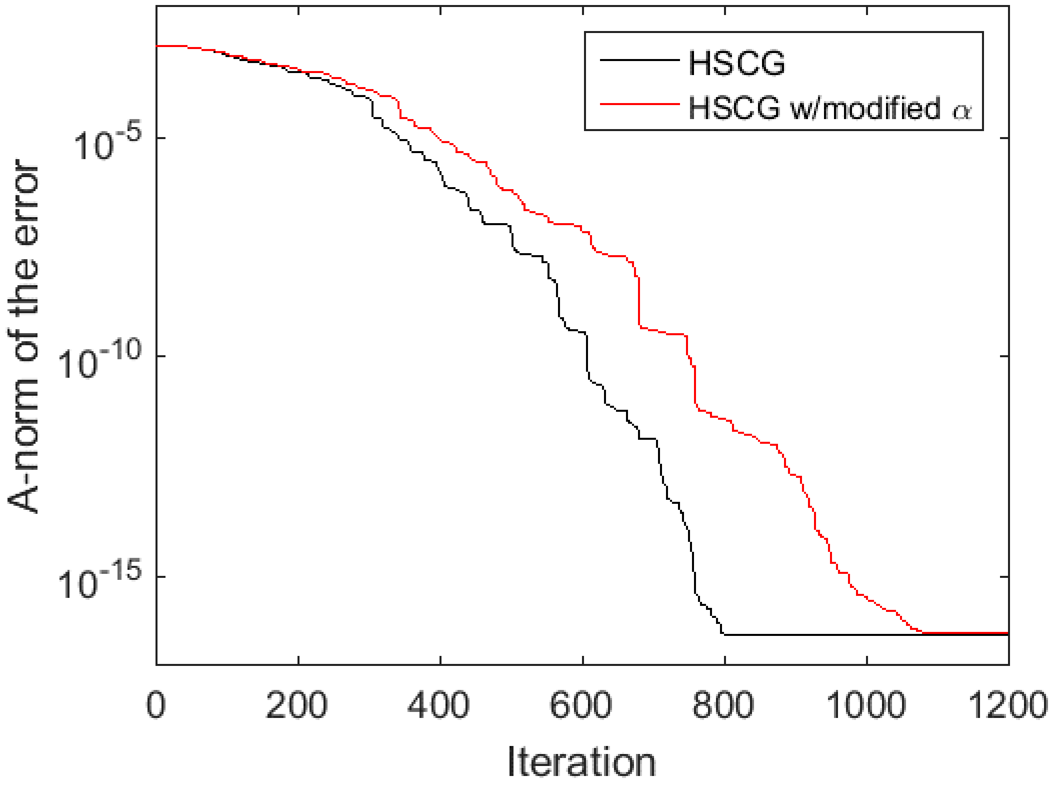
- Residual gap bounded by sum of local errors PLUS local errors multiplied by factors which depend on

$$\max_{0\leq \ell < j \leq i} \frac{\left\| r_j \right\|^2}{\left\| r_\ell \right\|^2}$$

$\Rightarrow$ Large residual oscillations can cause these factors to be large!

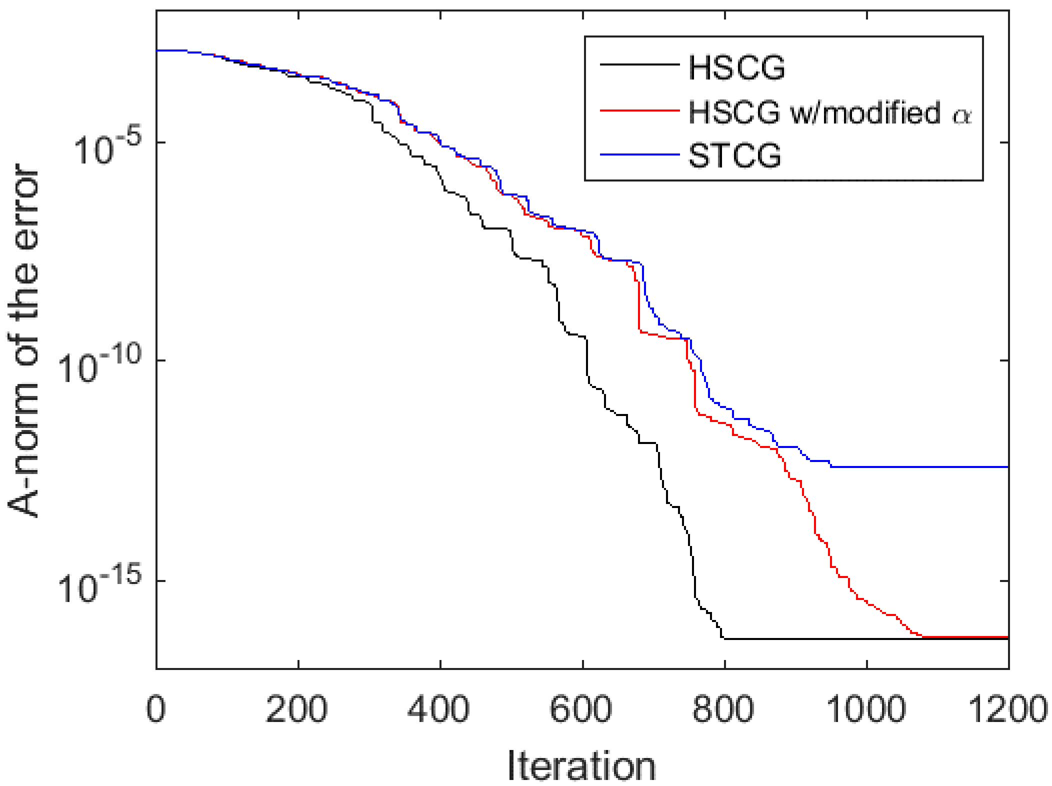
$\Rightarrow$ Local errors can be amplified!

STCG



HSCG
HSCG w/modified α
A-norm of the error
10^{-5}
10^{-10}
10^{-15}
0
200
400
600
800
1000
1200
Iteration

STCG


HSCG
HSCG w/modified α
STCG
10^{-5}
10^{-10}
10^{-15}
A-norm of the error
0
200
400
600
800
1000
1200
Iteration

Chronopoulos and Gear's CG (ChG CG)

- Chronopoulos and Gear (1989)
- Looks like HSCG, but very similar to 3-term recurrence CG (STCG)
- Reduces synchronizations/iteration to 1 by changing computation of α_i and using an auxiliary recurrence for Ap_i

$r_0 = b - Ax_0,\ p_0 = r_0,$
$s_0 = Ap_0,\ \ \alpha_0 = (r_0, r_0)/(p_0, s_0)$
for i = 1:nmax

$$x_i = x_{i-1} + \alpha_{i-1} p_{i-1}$$
$$r_i = r_{i-1} - \alpha_{i-1} s_{i-1}$$
$$w_i = Ar_i$$
$$\beta_i = \frac{(r_i, r_i)}{(r_{i-1}, r_{i-1})}$$
$$\alpha_i = \frac{(r_i, r_i)}{(w_i, r_i) - (\beta_i/\alpha_{i-1})(r_i, r_i)}$$
$$p_i = r_i + \beta_i p_{i-1}$$
$$s_i = w_i + \beta_i s_{i-1}$$

end

Chronopoulos and Gear's CG (ChG CG)

- Chronopoulos and Gear (1989)
- Looks like HSCG, but very similar to 3-term recurrence CG (STCG)
- Reduces synchronizations/iteration to 1 by changing computation of α_i and using an auxiliary recurrence for Ap_i

$$
\begin{aligned}
&r_0 = b - Ax_0,\ \ p_0 = r_0,\\
&s_0 = Ap_0,\ \ \alpha_0 = (r_0, r_0)/(p_0, s_0)\\
&\text{for } i \ = \ 1\text{:nmax}\\
&\qquad x_i = x_{i-1} + \alpha_{i-1} p_{i-1}\\
&\qquad r_i = r_{i-1} - \alpha_{i-1} s_{i-1}\\
&\qquad w_i = Ar_i\\
&\qquad \beta_i = \frac{(r_i, r_i)}{(r_{i-1}, r_{i-1})}\\
&\qquad \alpha_i = \frac{(r_i, r_i)}{(w_i, r_i) - (\beta_i/\alpha_{i-1})(r_i, r_i)}\\
&\qquad p_i = r_i + \beta_i p_{i-1}\\
&\qquad s_i = w_i + \beta_i s_{i-1}\\
&\text{end}
\end{aligned}
$$

Chronopoulos and Gear's CG (ChG CG)

- Chronopoulos and Gear (1989)
- Looks like HSCG, but very similar to 3-term recurrence CG (STCG)
- Reduces synchronizations/iteration to 1 by changing computation of α_i and using an auxiliary recurrence for Ap_i

$r_0 = b - Ax_0,\ p_0 = r_0,$

$s_0 = Ap_0,\quad \alpha_0 = (r_0, r_0)/(p_0, s_0)$

for i = 1:nmax

$$x_i = x_{i-1} + \alpha_{i-1} p_{i-1}$$

$$r_i = r_{i-1} - \alpha_{i-1} s_{i-1}$$

$$w_i = Ar_i$$

$$\beta_i = \frac{(r_i, r_i)}{(r_{i-1}, r_{i-1})}$$

$$\alpha_i = \frac{(r_i, r_i)}{(w_i, r_i) - (\beta_i/\alpha_{i-1})(r_i, r_i)}$$

$$p_i = r_i + \beta_i p_{i-1}$$

$$s_i = w_i + \beta_i s_{i-1}$$

end

Chronopoulos and Gear's CG (ChG CG)

- Chronopoulos and Gear (1989)
- Looks like HSCG, but very similar to 3-term recurrence CG (STCG)
- Reduces synchronizations/iteration to 1 by changing computation of α_i and using an auxiliary recurrence for Ap_i

$r_0 = b - Ax_0,\ p_0 = r_0,$

$s_0 = Ap_0,\ \ \alpha_0 = (r_0, r_0)/(p_0, s_0)$

for i = 1:nmax

$$x_i = x_{i-1} + \alpha_{i-1} p_{i-1}$$

$$r_i = r_{i-1} - \alpha_{i-1} s_{i-1}$$

$$w_i = Ar_i$$

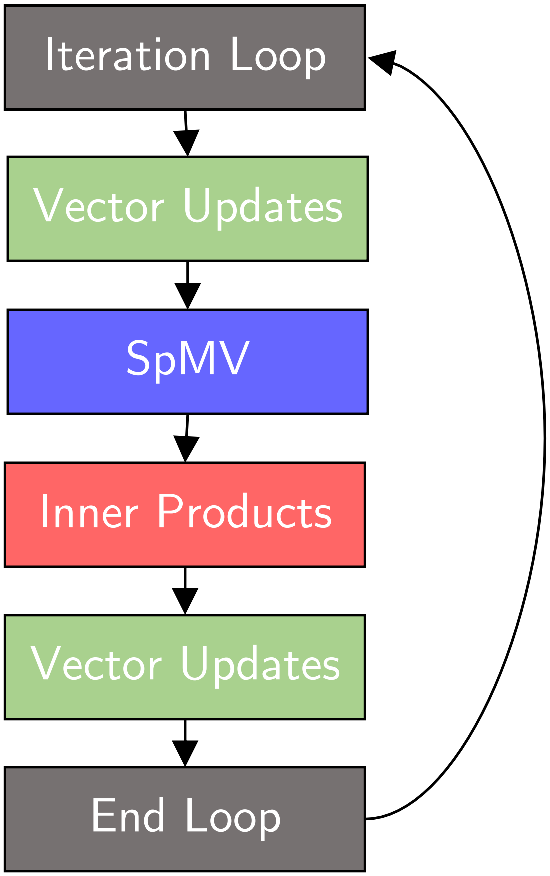
$$\beta_i = \frac{(r_i, r_i)}{(r_{i-1}, r_{i-1})}$$

$$\alpha_i = \frac{(r_i, r_i)}{(w_i, r_i) - (\beta_i/\alpha_{i-1})(r_i, r_i)}$$

$$p_i = r_i + \beta_i p_{i-1}$$

$$s_i = w_i + \beta_i s_{i-1}$$

end



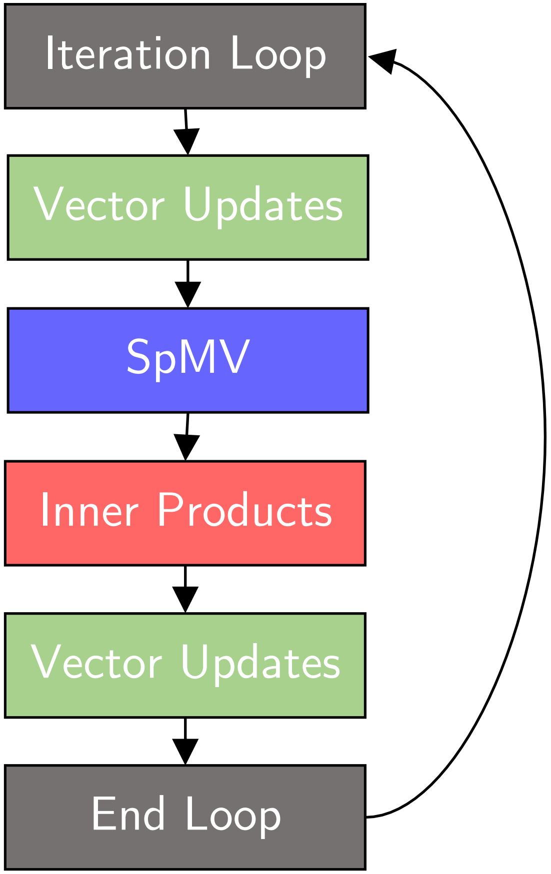
Chronopoulos and Gear's CG (ChG CG)

- Chronopoulos and Gear (1989)
- Looks like HSCG, but very similar to 3-term recurrence CG (STCG)
- Reduces synchronizations/iteration to 1 by changing computation of α_i and using an auxiliary recurrence for Ap_i

$r_0 = b - Ax_0,\ p_0 = r_0,$
$s_0 = Ap_0,\ \ \alpha_0 = (r_0, r_0)/(p_0, s_0)$
for i = 1:nmax

$$x_i = x_{i-1} + \alpha_{i-1} p_{i-1}$$
$$r_i = r_{i-1} - \alpha_{i-1} s_{i-1}$$
$$w_i = Ar_i$$
$$\beta_i = \frac{(r_i, r_i)}{(r_{i-1}, r_{i-1})}$$
$$\alpha_i = \frac{(r_i, r_i)}{(w_i, r_i) - (\beta_i/\alpha_{i-1})(r_i, r_i)}$$
$$p_i = r_i + \beta_i p_{i-1}$$
$$s_i = w_i + \beta_i s_{i-1}$$

end



Chronopoulos and Gear's CG (ChG CG)

- Chronopoulos and Gear (1989)
- Looks like HSCG, but very similar to 3-term recurrence CG (STCG)
- Reduces synchronizations/iteration to 1 by changing computation of α_i and using an auxiliary recurrence for Ap_i

$r_0 = b - Ax_0,\ p_0 = r_0,$

$s_0 = Ap_0,\ \ \alpha_0 = (r_0, r_0)/(p_0, s_0)$

for i = 1:nmax

$$x_i = x_{i-1} + \alpha_{i-1} p_{i-1}$$

$$r_i = r_{i-1} - \alpha_{i-1} s_{i-1}$$

$$w_i = Ar_i$$

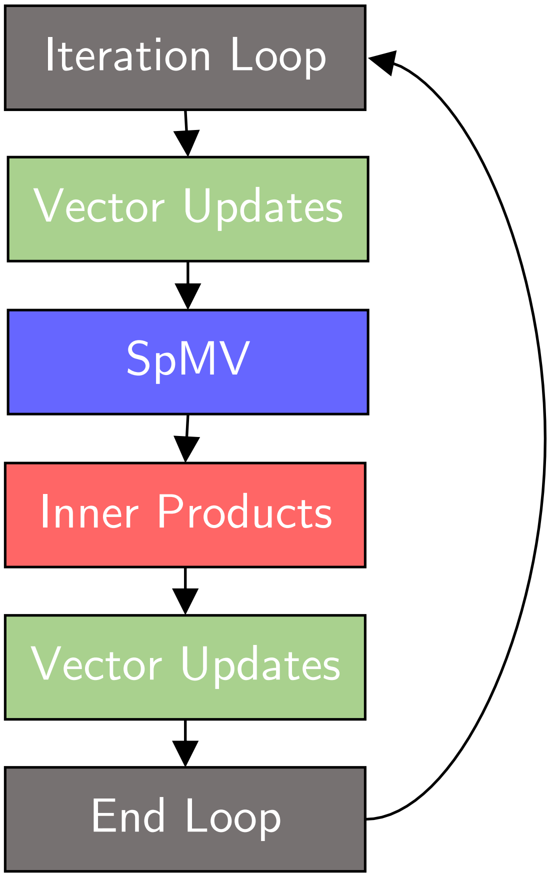
$$\beta_i = \frac{(r_i, r_i)}{(r_{i-1}, r_{i-1})}$$

$$\alpha_i = \frac{(r_i, r_i)}{(w_i, r_i) - (\beta_i/\alpha_{i-1})(r_i, r_i)}$$

$$p_i = r_i + \beta_i p_{i-1}$$

$$s_i = w_i + \beta_i s_{i-1}$$

end



Chronopoulos and Gear's CG (ChG CG)

- Chronopoulos and Gear (1989)
- Looks like HSCG, but very similar to 3-term recurrence CG (STCG)
- Reduces synchronizations/iteration to 1 by changing computation of α_i and using an auxiliary recurrence for Ap_i

$r_0 = b - Ax_0,\ p_0 = r_0,$

$s_0 = Ap_0,\ \ \alpha_0 = (r_0, r_0)/(p_0, s_0)$

for i = 1:nmax

$$x_i = x_{i-1} + \alpha_{i-1} p_{i-1}$$

$$r_i = r_{i-1} - \alpha_{i-1} s_{i-1}$$

$$w_i = Ar_i$$

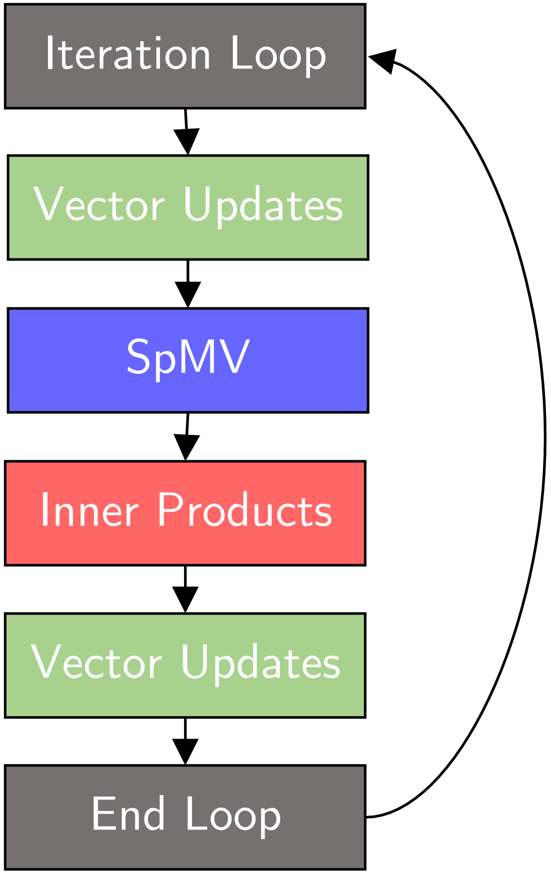
$$\beta_i = \frac{(r_i, r_i)}{(r_{i-1}, r_{i-1})}$$

$$\alpha_i = \frac{(r_i, r_i)}{(w_i, r_i) - (\beta_i / \alpha_{i-1})(r_i, r_i)}$$

$$p_i = r_i + \beta_i p_{i-1}$$

$$s_i = w_i + \beta_i s_{i-1}$$

end



Chronopoulos and Gear's CG (ChG CG)

- Chronopoulos and Gear (1989)
- Looks like HSCG, but very similar to 3-term recurrence CG (STCG)
- Reduces synchronizations/iteration to 1 by changing computation of α_i and using an auxiliary recurrence for Ap_i

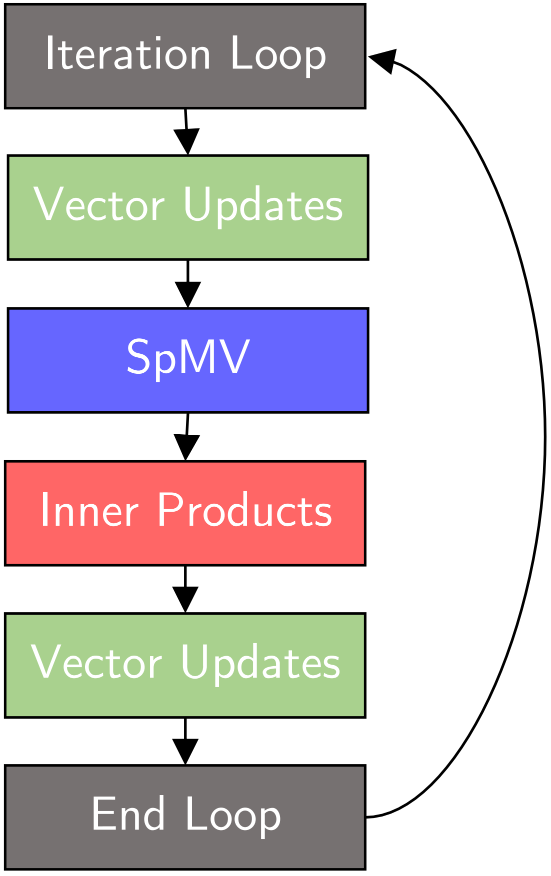
$r_0 = b - Ax_0,\ \ p_0 = r_0,$

$s_0 = Ap_0,\ \ \alpha_0 = (r_0, r_0)/(p_0, s_0)$

for i = 1:nmax

$$x_i = x_{i-1} + \alpha_{i-1} p_{i-1}$$
$$r_i = r_{i-1} - \alpha_{i-1} s_{i-1}$$
$$w_i = Ar_i$$
$$\beta_i = \frac{(r_i, r_i)}{(r_{i-1}, r_{i-1})}$$
$$\alpha_i = \frac{(r_i, r_i)}{(w_i, r_i) - (\beta_i/\alpha_{i-1})(r_i, r_i)}$$
$$p_i = r_i + \beta_i p_{i-1}$$
$$s_i = w_i + \beta_i s_{i-1}$$

end



Pipelined CG (GVCG)

- Pipelined CG of Ghysels and Vanroose (2014)

- Similar to Chronopoulos and Gear approach
  - Uses auxiliary vector $s_i \equiv Ap_i$ and same formula for α_i

Pipelined CG (GVCG)

- Pipelined CG of Ghysels and Vanroose (2014)

- Similar to Chronopoulos and Gear approach
  - Uses auxiliary vector $s_i \equiv Ap_i$ and same formula for α_i

- Also uses auxiliary vectors for Ar_i and A^2r_i to remove sequential dependency between SpMV and inner products

Pipelined CG (GVCG)

- Pipelined CG of Ghysels and Vanroose (2014)

- Similar to Chronopoulos and Gear approach
  - Uses auxiliary vector $s_i \equiv Ap_i$ and same formula for α_i

- Also uses auxiliary vectors for Ar_i and A^2r_i to remove sequential dependency between SpMV and inner products

  - Allows the use of nonblocking (asynchronous) MPI communication to *overlap* SpMV and inner products

  - Hides the latency of global communications

GVCG (Ghysels and Vanroose 2014)

$r_0 = b - Ax_0,\ p_0 = r_0$

$s_0 = Ap_0, w_0 = Ar_0, z_0 = Aw_0,$
$\alpha_0 = r_0^T r_0 / p_0^T s_0$

for i = 1:nmax

$$x_i = x_{i-1} + \alpha_{i-1} p_{i-1}$$

$$r_i = r_{i-1} - \alpha_{i-1} s_{i-1}$$

$$w_i = w_{i-1} - \alpha_{i-1} z_{i-1}$$

$$q_i = Aw_i$$

$$\beta_i = \frac{r_i^T r_i}{r_{i-1}^T r_{i-1}}$$

$$\alpha_i = \frac{r_i^T r_i}{w_i^T r_i - (\beta_i/\alpha_{i-1}) r_i^T r_i}$$

$$p_i = r_i + \beta_i p_{i-1}$$

$$s_i = w_i + \beta_i s_{i-1}$$

$$z_i = q_i + \beta_i z_{i-1}$$

end

GVCG (Ghysels and Vanroose 2014)

$r_0 = b - Ax_0,\ p_0 = r_0$

$s_0 = Ap_0, w_0 = Ar_0, z_0 = Aw_0,$

$\alpha_0 = r_0^T r_0 / p_0^T s_0$

for i = 1:nmax

$$x_i = x_{i-1} + \alpha_{i-1} p_{i-1}$$

$$r_i = r_{i-1} - \alpha_{i-1} s_{i-1}$$

$$w_i = w_{i-1} - \alpha_{i-1} z_{i-1}$$

$$q_i = Aw_i$$

$$\beta_i = \frac{r_i^T r_i}{r_{i-1}^T r_{i-1}}$$

$$\alpha_i = \frac{r_i^T r_i}{w_i^T r_i - (\beta_i / \alpha_{i-1}) r_i^T r_i}$$

$$p_i = r_i + \beta_i p_{i-1}$$

$$s_i = w_i + \beta_i s_{i-1}$$

$$z_i = q_i + \beta_i z_{i-1}$$

end

GVCG (Ghysels and Vanroose 2014)

$r_0 = b - Ax_0,\ p_0 = r_0$

$s_0 = Ap_0, w_0 = Ar_0, z_0 = Aw_0,$
$\alpha_0 = r_0^T r_0 / p_0^T s_0$

for i = 1:nmax

$$x_i = x_{i-1} + \alpha_{i-1} p_{i-1}$$

$$r_i = r_{i-1} - \alpha_{i-1} s_{i-1}$$

$$w_i = w_{i-1} - \alpha_{i-1} z_{i-1}$$

$$q_i = Aw_i$$

$$\beta_i = \frac{r_i^T r_i}{r_{i-1}^T r_{i-1}}$$

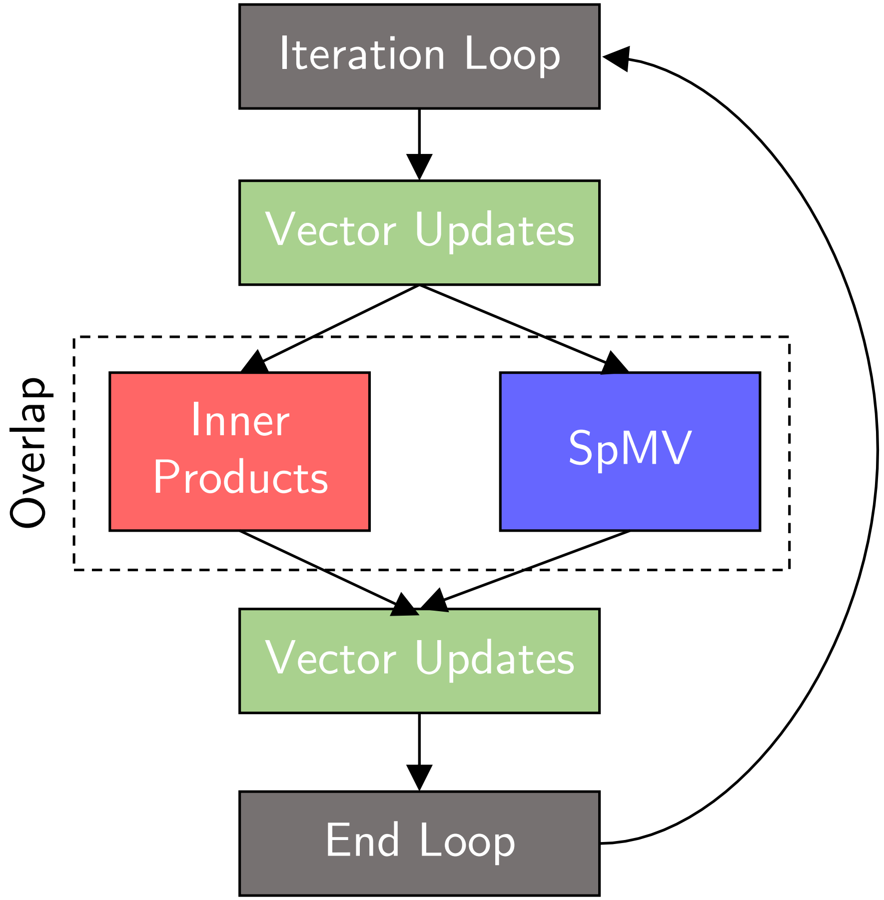
$$\alpha_i = \frac{r_i^T r_i}{w_i^T r_i - (\beta_i / \alpha_{i-1}) r_i^T r_i}$$

$$p_i = r_i + \beta_i p_{i-1}$$

$$s_i = w_i + \beta_i s_{i-1}$$

$$z_i = q_i + \beta_i z_{i-1}$$

end



GVCG (Ghysels and Vanroose 2014)

$r_0 = b - Ax_0,\ p_0 = r_0$

$s_0 = Ap_0, w_0 = Ar_0, z_0 = Aw_0,$
$\alpha_0 = r_0^T r_0 / p_0^T s_0$

for i = 1:nmax

$x_i = x_{i-1} + \alpha_{i-1} p_{i-1}$

$r_i = r_{i-1} - \alpha_{i-1} s_{i-1}$

$w_i = w_{i-1} - \alpha_{i-1} z_{i-1}$

$q_i = Aw_i$

$\beta_i = \frac{r_i^T r_i}{r_{i-1}^T r_{i-1}}$

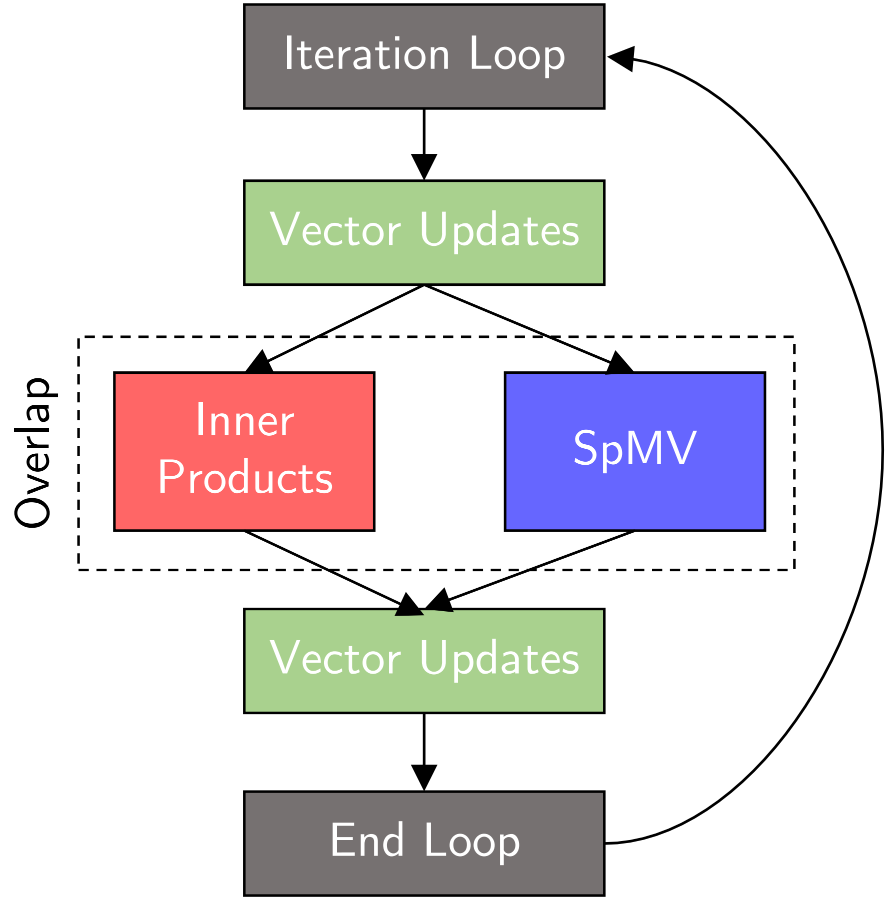
$\alpha_i = \frac{r_i^T r_i}{w_i^T r_i - (\beta_i / \alpha_{i-1}) r_i^T r_i}$

$p_i = r_i + \beta_i p_{i-1}$

$s_i = w_i + \beta_i s_{i-1}$

$z_i = q_i + \beta_i z_{i-1}$

end



GVCG (Ghysels and Vanroose 2014)

$r_0 = b - Ax_0,\ p_0 = r_0$

$s_0 = Ap_0, w_0 = Ar_0, z_0 = Aw_0,$
$\alpha_0 = r_0^T r_0 / p_0^T s_0$

for i = 1:nmax

$x_i = x_{i-1} + \alpha_{i-1} p_{i-1}$

$r_i = r_{i-1} - \alpha_{i-1} s_{i-1}$

$w_i = w_{i-1} - \alpha_{i-1} z_{i-1}$

$q_i = Aw_i$

$\beta_i = \frac{r_i^T r_i}{r_{i-1}^T r_{i-1}}$

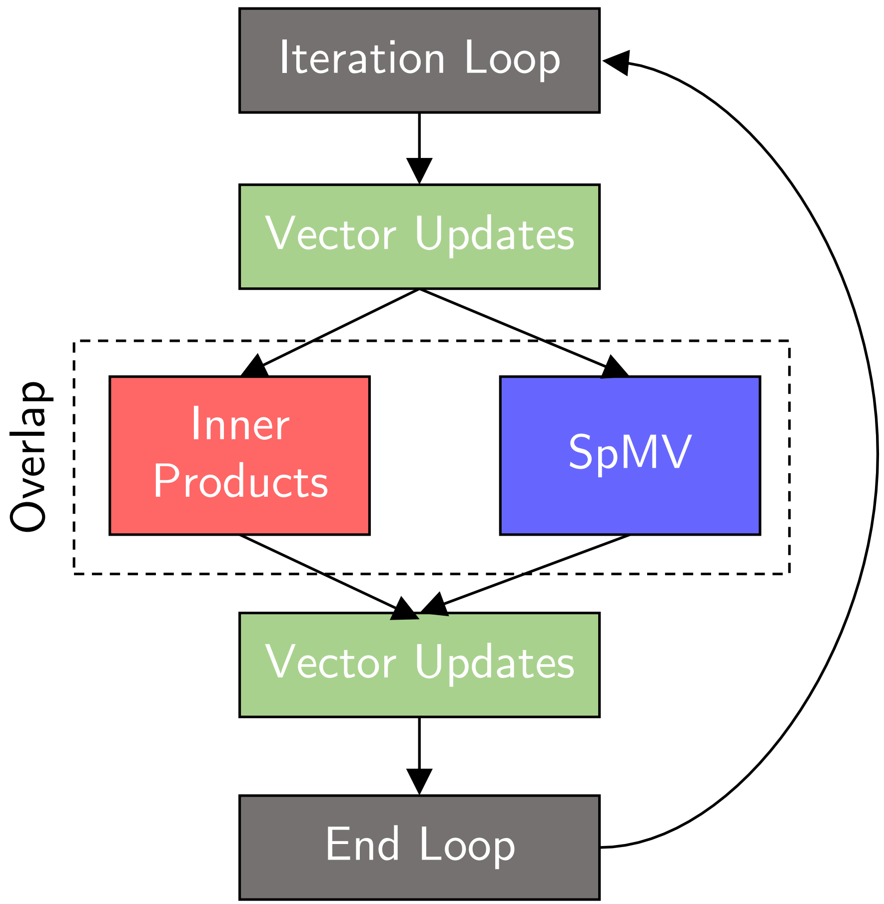
$\alpha_i = \frac{r_i^T r_i}{w_i^T r_i - (\beta_i / \alpha_{i-1}) r_i^T r_i}$

$p_i = r_i + \beta_i p_{i-1}$

$s_i = w_i + \beta_i s_{i-1}$

$z_i = q_i + \beta_i z_{i-1}$

end



GVCG (Ghysels and Vanroose 2014)

$r_0 = b - Ax_0,\ p_0 = r_0$

$s_0 = Ap_0, w_0 = Ar_0, z_0 = Aw_0,$
$\alpha_0 = r_0^T r_0 / p_0^T s_0$

for i = 1:nmax

$x_i = x_{i-1} + \alpha_{i-1} p_{i-1}$

$r_i = r_{i-1} - \alpha_{i-1} s_{i-1}$

$w_i = w_{i-1} - \alpha_{i-1} z_{i-1}$

$q_i = Aw_i$

$\beta_i = \frac{r_i^T r_i}{r_{i-1}^T r_{i-1}}$

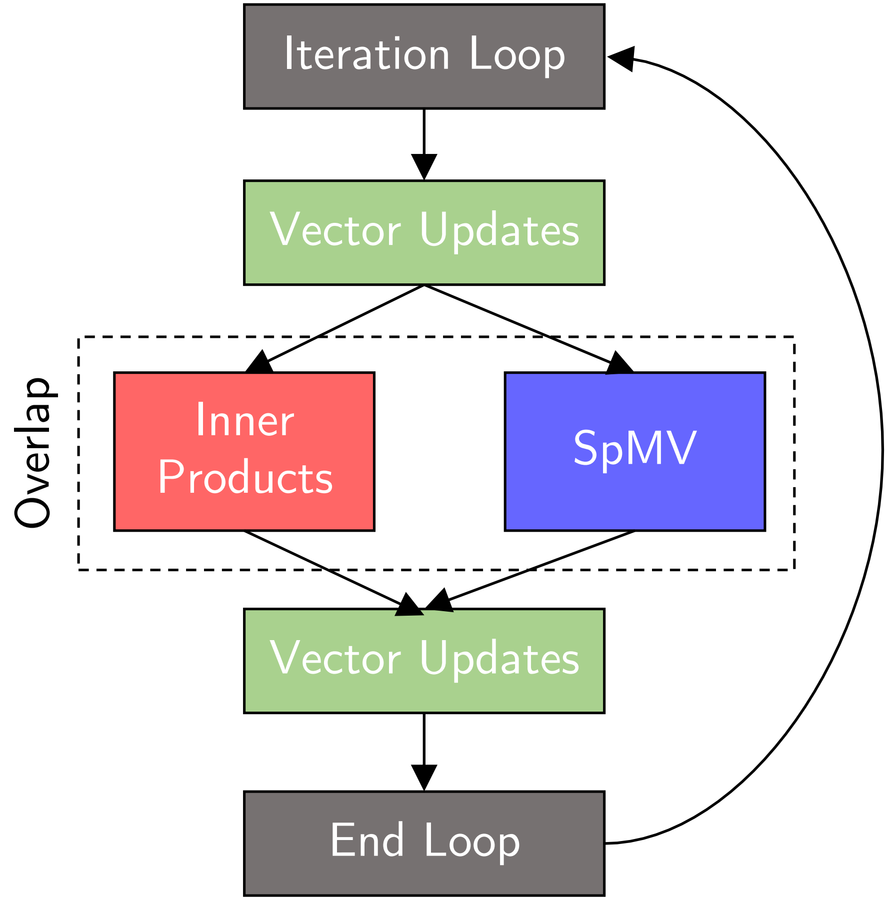
$\alpha_i = \frac{r_i^T r_i}{w_i^T r_i - (\beta_i/\alpha_{i-1}) r_i^T r_i}$

$p_i = r_i + \beta_i p_{i-1}$

$s_i = w_i + \beta_i s_{i-1}$

$z_i = q_i + \beta_i z_{i-1}$

end



GVCG (Ghysels and Vanroose 2014)

$r_0 = b - Ax_0,\ p_0 = r_0$

$s_0 = Ap_0, w_0 = Ar_0, z_0 = Aw_0,$
$\alpha_0 = r_0^T r_0 / p_0^T s_0$

for i = 1:nmax

$x_i = x_{i-1} + \alpha_{i-1} p_{i-1}$

$r_i = r_{i-1} - \alpha_{i-1} s_{i-1}$

$w_i = w_{i-1} - \alpha_{i-1} z_{i-1}$

$q_i = Aw_i$

$\beta_i = \frac{r_i^T r_i}{r_{i-1}^T r_{i-1}}$

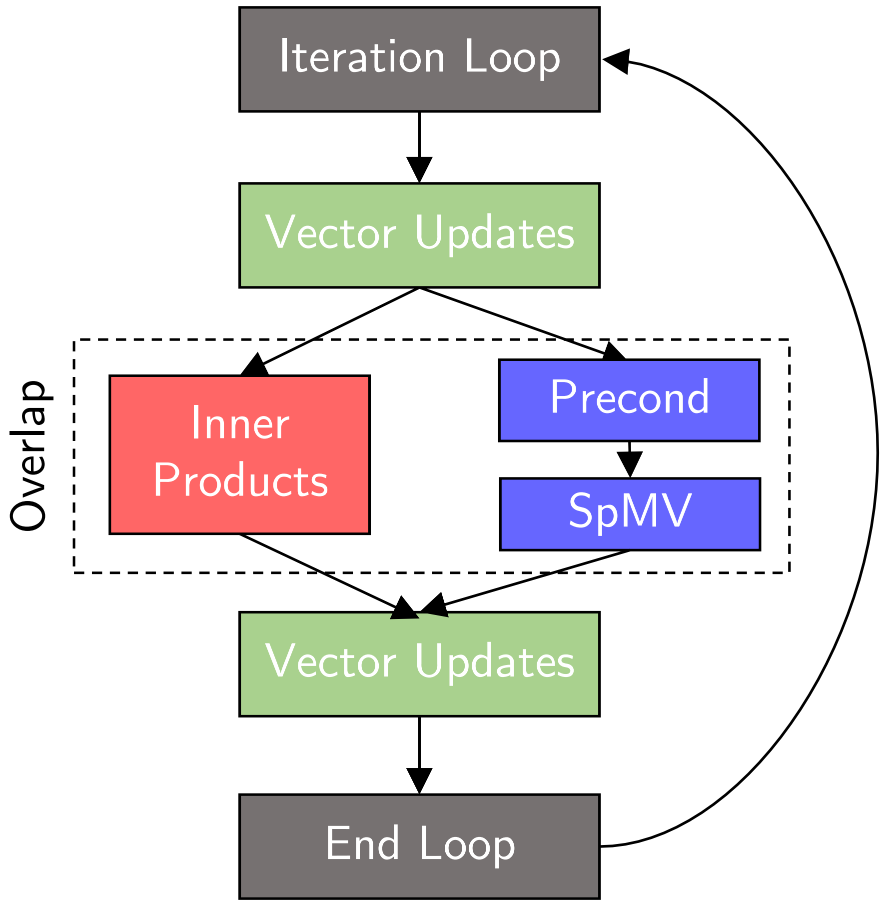
$\alpha_i = \frac{r_i^T r_i}{w_i^T r_i - (\beta_i / \alpha_{i-1}) r_i^T r_i}$

$p_i = r_i + \beta_i p_{i-1}$

$s_i = w_i + \beta_i s_{i-1}$

$z_i = q_i + \beta_i z_{i-1}$

end



Attainable accuracy of pipelined CG

- What is the effect of adding auxiliary recurrences to the CG method?

Attainable accuracy of pipelined CG

- What is the effect of adding auxiliary recurrences to the CG method?
- To isolate the effects, we consider a simplified version of a pipelined method

$r_0 = b - Ax_0, p_0 = r_0, s_0 = Ap_0$

for i = 1:nmax

$$\alpha_{i-1} = \frac{(r_{i-1}, r_{i-1})}{(p_{i-1}, s_{i-1})}$$

$$x_i = x_{i-1} + \alpha_{i-1} p_{i-1}$$

$$r_i = r_{i-1} - \alpha_{i-1} s_{i-1}$$

$$\beta_i = \frac{(r_i, r_i)}{(r_{i-1}, r_{i-1})}$$

$$p_i = r_i + \beta_i p_{i-1}$$

$$s_i = Ar_i + \beta_i s_{i-1}$$

end

Attainable accuracy of pipelined CG

- What is the effect of adding auxiliary recurrences to the CG method?
- To isolate the effects, we consider a simplified version of a pipelined method
  - Uses same update formulas for α and β as HSCG, but uses additional recurrence for Ap_i

$r_0 = b - Ax_0, p_0 = r_0, s_0 = Ap_0$

for i = 1:nmax

$$\alpha_{i-1} = \frac{(r_{i-1}, r_{i-1})}{(p_{i-1}, s_{i-1})}$$

$$x_i = x_{i-1} + \alpha_{i-1} p_{i-1}$$

$$r_i = r_{i-1} - \alpha_{i-1} s_{i-1}$$

$$\beta_i = \frac{(r_i, r_i)}{(r_{i-1}, r_{i-1})}$$

$$p_i = r_i + \beta_i p_{i-1}$$

$$s_i = Ar_i + \beta_i s_{i-1}$$

end

Attainable accuracy of simple pipelined CG

$$\hat{x}_i = \hat{x}_{i-1} + \hat{\alpha}_{i-1}\hat{p}_{i-1} + \boldsymbol{\delta x_i} \qquad \hat{r}_i = \hat{r}_{i-1} - \hat{\alpha}_{i-1}\,\hat{s}_{i-1} + \boldsymbol{\delta r_i}$$

Attainable accuracy of simple pipelined CG

$$\hat{x}_i = \hat{x}_{i-1} + \hat{\alpha}_{i-1}\hat{p}_{i-1} + \boldsymbol{\delta x_i} \qquad \hat{r}_i = \hat{r}_{i-1} - \hat{\alpha}_{i-1}\,\hat{s}_{i-1} + \boldsymbol{\delta r_i}$$

$$f_i = \hat{r}_i - (b - A\hat{x}_i)$$

Attainable accuracy of simple pipelined CG

$$\hat{x}_i = \hat{x}_{i-1} + \hat{\alpha}_{i-1}\hat{p}_{i-1} + \boldsymbol{\delta x_i} \qquad \hat{r}_i = \hat{r}_{i-1} - \hat{\alpha}_{i-1}\,\hat{s}_{i-1} + \boldsymbol{\delta r_i}$$

$$\begin{aligned} f_i &= \hat{r}_i - (b - A\hat{x}_i) \\ &= f_{i-1} - \hat{\alpha}_{i-1}(\hat{s}_{i-1} - A\hat{p}_{i-1}) + \delta r_i + A\delta x_i \end{aligned}$$

Attainable accuracy of simple pipelined CG

$$\hat{x}_i = \hat{x}_{i-1} + \hat{\alpha}_{i-1}\hat{p}_{i-1} + \boldsymbol{\delta x_i} \qquad \hat{r}_i = \hat{r}_{i-1} - \hat{\alpha}_{i-1}\,\hat{s}_{i-1} + \boldsymbol{\delta r_i}$$

$$\begin{aligned} f_i &= \hat{r}_i - (b - A\hat{x}_i) \\ &= f_{i-1} - \hat{\alpha}_{i-1}(\hat{s}_{i-1} - A\hat{p}_{i-1}) + \delta r_i + A\delta x_i \\ &= f_0 + \textstyle\sum_{m=1}^{i}(\delta r_m + A\delta x_m) - G_i d_i \end{aligned}$$

where

$$G_i = \hat{S}_i - A\hat{P}_i, \quad d_i = [\hat{\alpha}_0, \dots, \hat{\alpha}_{i-1}]^T$$

Attainable accuracy of simple pipelined CG

$$\hat{x}_i = \hat{x}_{i-1} + \hat{\alpha}_{i-1}\hat{p}_{i-1} + \boldsymbol{\delta x_i} \qquad \hat{r}_i = \hat{r}_{i-1} - \hat{\alpha}_{i-1}\,\hat{s}_{i-1} + \boldsymbol{\delta r_i}$$

$$\begin{aligned} f_i &= \hat{r}_i - (b - A\hat{x}_i) \\ &= f_{i-1} - \hat{\alpha}_{i-1}(\hat{s}_{i-1} - A\hat{p}_{i-1}) + \delta r_i + A\delta x_i \\ &= f_0 + \textstyle\sum_{m=1}^{i}(\delta r_m + A\delta x_m) - G_i d_i \end{aligned}$$

where

$$G_i = \hat{S}_i - A\hat{P}_i, \quad d_i = [\hat{\alpha}_0, \dots, \hat{\alpha}_{i-1}]^T$$

Attainable accuracy of simple pipelined CG

$$\hat{x}_i = \hat{x}_{i-1} + \hat{\alpha}_{i-1}\hat{p}_{i-1} + \boldsymbol{\delta x_i} \qquad \hat{r}_i = \hat{r}_{i-1} - \hat{\alpha}_{i-1}\,\hat{s}_{i-1} + \boldsymbol{\delta r_i}$$

$$\begin{aligned} f_i &= \hat{r}_i - (b - A\hat{x}_i) \\ &= f_{i-1} - \hat{\alpha}_{i-1}(\hat{s}_{i-1} - A\hat{p}_{i-1}) + \delta r_i + A\delta x_i \\ &= f_0 + \textstyle\sum_{m=1}^{i}(\delta r_m + A\delta x_m) - G_i d_i \end{aligned}$$

where

$$G_i = \hat{S}_i - A\hat{P}_i, \quad d_i = [\hat{\alpha}_0, \dots, \hat{\alpha}_{i-1}]^T$$

Attainable accuracy of simple pipelined CG

$$\|G_i\| \leq \frac{O(\varepsilon)}{1 - O(\varepsilon)} \left(\kappa(\widehat{U}_i) \|A\| \|\widehat{P}_i\| + \|A\| \|\widehat{R}_i\| \|\widehat{U}_i^{-1}\| \right)$$

$$\widehat{U}_i = \begin{bmatrix} 1 & -\hat{\beta}_1 & 0 & 0 \\ 0 & 1 & \ddots & 0 \\ \vdots & \ddots & 1 & -\hat{\beta}_{i-1} \\ 0 & \dots & 0 & 1 \end{bmatrix} \qquad \widehat{U}_i^{-1} = \begin{bmatrix} 1 & \hat{\beta}_1 & \dots & \dots & \hat{\beta}_1 \hat{\beta}_2 \cdots \hat{\beta}_{i-1} \\ 0 & 1 & \hat{\beta}_2 & \dots & \hat{\beta}_2 \cdots \hat{\beta}_{i-1} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & 1 & \hat{\beta}_{i-1} \\ 0 & \cdots & \cdots & 0 & 1 \end{bmatrix}$$

Attainable accuracy of simple pipelined CG

$$\|G_i\| \leq \frac{O(\varepsilon)}{1 - O(\varepsilon)} \left(\kappa(\widehat{U}_i) \|A\| \left\|\widehat{P}_i\right\| + \|A\| \left\|\widehat{R}_i\right\| \left\|\widehat{U}_i^{-1}\right\| \right)$$

$$\widehat{U}_i = \begin{bmatrix} 1 & -\hat{\beta}_1 & 0 & 0 \\ 0 & 1 & \ddots & 0 \\ \vdots & \ddots & 1 & -\hat{\beta}_{i-1} \\ 0 & \dots & 0 & 1 \end{bmatrix} \qquad \widehat{U}_i^{-1} = \begin{bmatrix} 1 & \hat{\beta}_1 & \dots & \dots & \hat{\beta}_1\hat{\beta}_2 \cdots \hat{\beta}_{i-1} \\ 0 & 1 & \hat{\beta}_2 & \dots & \hat{\beta}_2 \cdots \hat{\beta}_{i-1} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & 1 & \hat{\beta}_{i-1} \\ 0 & \cdots & \cdots & 0 & 1 \end{bmatrix}$$

$$\beta_\ell \beta_{\ell+1} \cdots \beta_j = \frac{\left\|r_j\right\|^2}{\|r_{\ell-1}\|^2}, \qquad \ell < j$$

Attainable accuracy of simple pipelined CG

$$\|G_i\| \leq \frac{O(\varepsilon)}{1 - O(\varepsilon)} \left(\kappa(\hat{U}_i) \|A\| \|\hat{P}_i\| + \|A\| \|\hat{R}_i\| \|\hat{U}_i^{-1}\| \right)$$

$$\hat{U}_i = \begin{bmatrix} 1 & -\hat{\beta}_1 & 0 & 0 \\ 0 & 1 & \ddots & 0 \\ \vdots & \ddots & 1 & -\hat{\beta}_{i-1} \\ 0 & \dots & 0 & 1 \end{bmatrix} \qquad \hat{U}_i^{-1} = \begin{bmatrix} 1 & \hat{\beta}_1 & \dots & \dots & \hat{\beta}_1 \hat{\beta}_2 \cdots \hat{\beta}_{i-1} \\ 0 & 1 & \hat{\beta}_2 & \dots & \hat{\beta}_2 \cdots \hat{\beta}_{i-1} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & 1 & \hat{\beta}_{i-1} \\ 0 & \cdots & \cdots & 0 & 1 \end{bmatrix}$$

$$\beta_\ell \beta_{\ell+1} \cdots \beta_j = \frac{\|r_j\|^2}{\|r_{\ell-1}\|^2}, \qquad \ell < j$$

- Residual oscillations can cause these factors to be large!
- Errors in computed recurrence coefficients can be amplified!

Attainable accuracy of simple pipelined CG

$$\|G_i\| \leq \frac{O(\varepsilon)}{1-O(\varepsilon)}\left(\kappa(\widehat{U}_i)\|A\|\left\|\widehat{P}_i\right\| + \|A\|\left\|\widehat{R}_i\right\|\left\|\widehat{U}_i^{-1}\right\|\right)$$

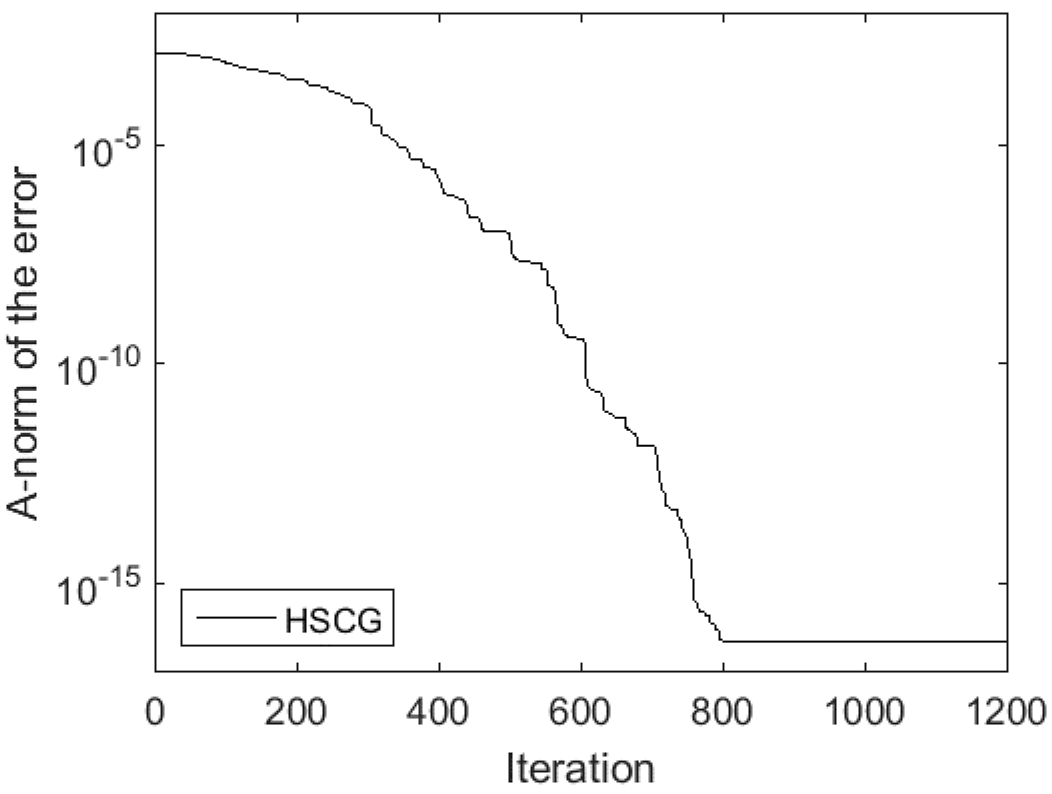
$$\widehat{U}_i = \begin{bmatrix} 1 & -\hat{\beta}_1 & 0 & 0 \\ 0 & 1 & \ddots & 0 \\ \vdots & \ddots & 1 & -\hat{\beta}_{i-1} \\ 0 & \dots & 0 & 1 \end{bmatrix} \qquad \widehat{U}_i^{-1} = \begin{bmatrix} 1 & \hat{\beta}_1 & \dots & \dots & \hat{\beta}_1\hat{\beta}_2\cdots\hat{\beta}_{i-1} \\ 0 & 1 & \hat{\beta}_2 & \dots & \hat{\beta}_2\cdots\hat{\beta}_{i-1} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & 1 & \hat{\beta}_{i-1} \\ 0 & \cdots & \cdots & 0 & 1 \end{bmatrix}$$

$$\beta_\ell \beta_{\ell+1} \cdots \beta_j = \frac{\left\|r_j\right\|^2}{\|r_{\ell-1}\|^2}, \qquad \ell < j$$

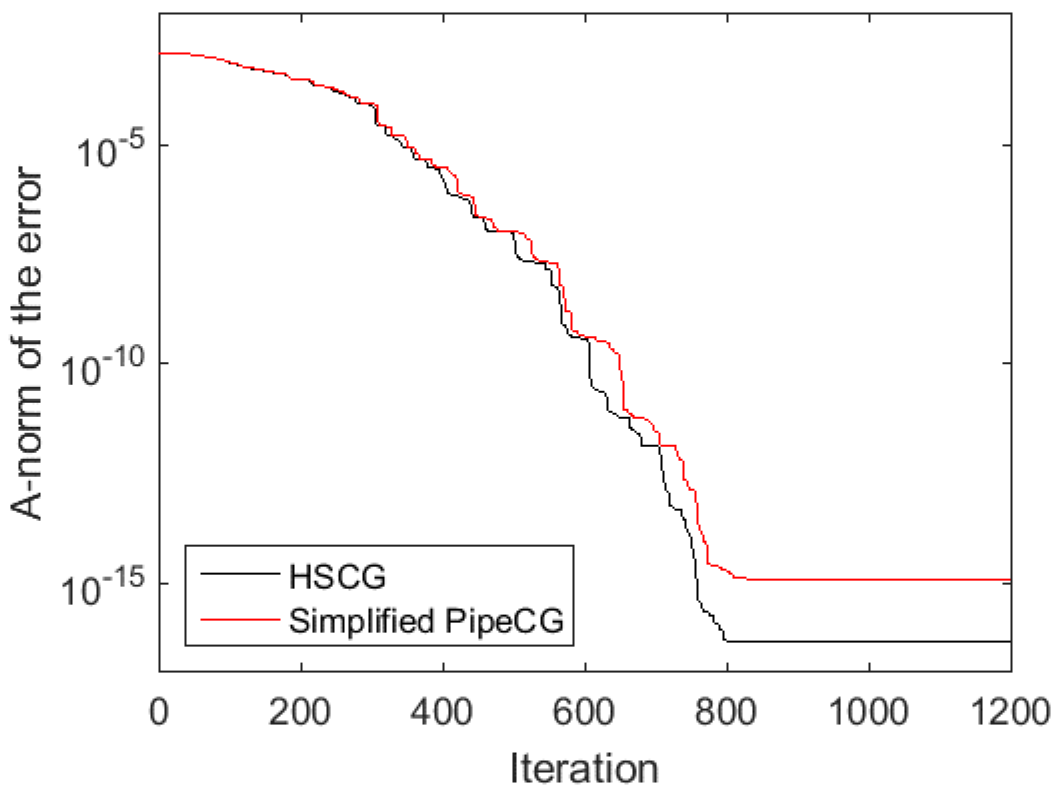
- Residual oscillations can cause these factors to be large!
- Errors in computed recurrence coefficients can be amplified!

- Very similar to the results for attainable accuracy in the 3-term STCG
- Seemingly innocuous change can cause drastic loss of accuracy

Simple pipelined CG

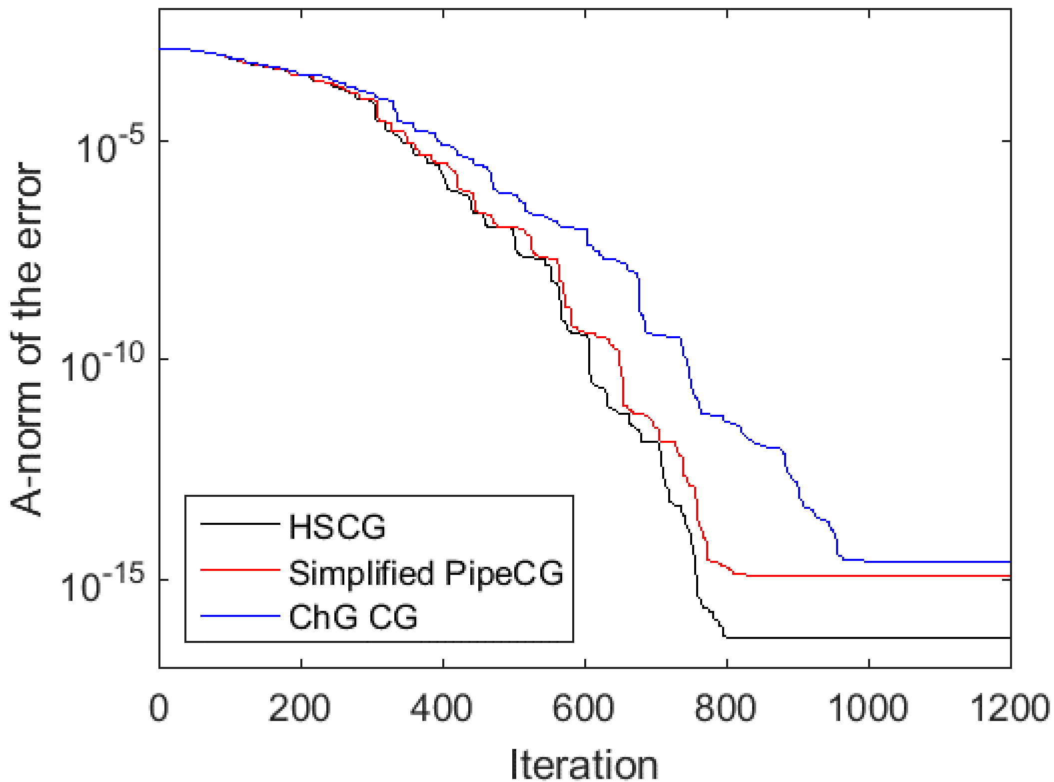


Simple pipelined CG



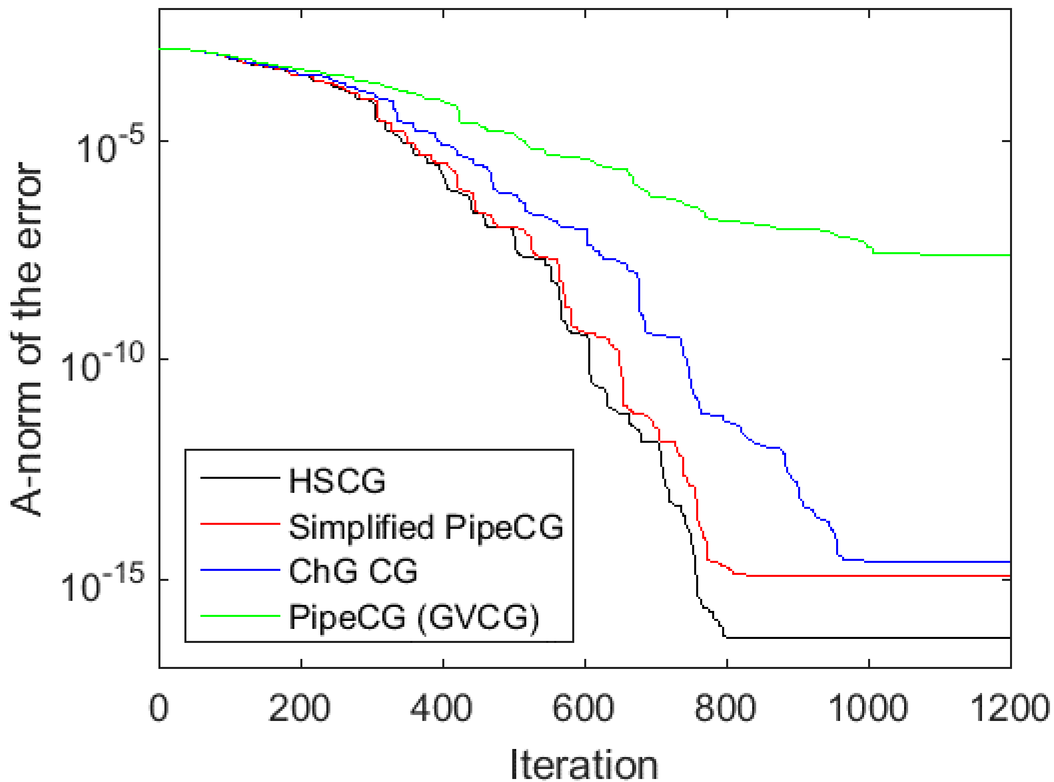
effect of using auxiliary vector $s_i \equiv Ap_i$

Simple pipelined CG



effect of changing formula for recurrence coefficient α and using auxiliary vector $s_i \equiv Ap_i$

Simple pipelined CG



effect of changing formula for recurrence coefficient α and using auxiliary vectors $s_i \equiv Ap_i, w_i \equiv Ar_i, z_i \equiv A^2 r_i$

s-step CG

- Idea: Compute blocks of s iterations at once
  - Compute updates in a different basis
  - Communicate every s iterations instead of every iteration
  - Reduces number of synchronizations per iteration by a factor of s

s-step CG

- Idea: Compute blocks of s iterations at once
  - Compute updates in a different basis
  - Communicate every s iterations instead of every iteration
  - Reduces number of synchronizations per iteration by a factor of s

- An idea rediscovered many times…

s-step CG

- Idea: Compute blocks of s iterations at once
  - Compute updates in a different basis
  - Communicate every s iterations instead of every iteration
  - Reduces number of synchronizations per iteration by a factor of s

- An idea rediscovered many times…
- First related work: s-dimensional steepest descent, least squares
  - Khabaza (‘63), Forsythe (‘68), Marchuk and Kuznecov (‘68)

s-step CG

- Idea: Compute blocks of s iterations at once
  - Compute updates in a different basis
  - Communicate every s iterations instead of every iteration
  - Reduces number of synchronizations per iteration by a factor of s

- An idea rediscovered many times…
- First related work: s-dimensional steepest descent, least squares
  - Khabaza ('63), Forsythe ('68), Marchuk and Kuznecov ('68)
- Flurry of work on s-step Krylov methods in '80s/early '90s: see, e.g., Van Rosendale (1983); Chronopoulos and Gear (1989)

s-step CG

- Idea: Compute blocks of s iterations at once
  - Compute updates in a different basis
  - Communicate every s iterations instead of every iteration
  - Reduces number of synchronizations per iteration by a factor of s

- An idea rediscovered many times…
- First related work: s-dimensional steepest descent, least squares
  - Khabaza ('63), Forsythe ('68), Marchuk and Kuznecov ('68)
- Flurry of work on s-step Krylov methods in '80s/early '90s: see, e.g., Van Rosendale (1983); Chronopoulos and Gear (1989)

- Resurgence of interest in recent years due to growing problem sizes; growing relative cost of communication

s-step CG

Key observation: After iteration i, for $j \in \{0, .., s\}$,

$$x_{i+j} - x_i, \quad r_{i+j}, \quad p_{i+j} \quad \in \quad \mathcal{K}_{s+1}(A, p_i) + \mathcal{K}_s(A, r_i)$$

s-step CG

Key observation: After iteration i, for $j \in \{0, .., s\}$,

$$x_{i+j} - x_i, \quad r_{i+j}, \quad p_{i+j} \quad \in \quad \mathcal{K}_{s+1}(A, p_i) + \mathcal{K}_s(A, r_i)$$

s steps of s-step CG:

s-step CG

Key observation: After iteration i, for $j \in \{0, .., s\}$,

$$x_{i+j} - x_i, \;\; r_{i+j}, \;\; p_{i+j} \;\in\; \mathcal{K}_{s+1}(A, p_i) + \mathcal{K}_s(A, r_i)$$

s steps of s-step CG:

Expand solution space s dimensions at once

Compute "basis" matrix $\mathcal{Y}$ such that $\operatorname{span}(\mathcal{Y}) = \mathcal{K}_{s+1}(A, p_i) + \mathcal{K}_s(A, r_i)$ according to the recurrence $A\underline{\mathcal{Y}} = \mathcal{Y}\,\mathcal{B}$

s-step CG

Key observation: After iteration i, for $j \in \{0, .., s\}$,

$$x_{i+j} - x_i, \ \ r_{i+j}, \ \ p_{i+j} \ \ \in \ \ \mathcal{K}_{s+1}(A, p_i) + \mathcal{K}_s(A, r_i)$$

s steps of s-step CG:

Expand solution space s dimensions at once

Compute "basis" matrix $\mathcal{Y}$ such that $\mathrm{span}(\mathcal{Y}) = \mathcal{K}_{s+1}(A, p_i) + \mathcal{K}_s(A, r_i)$ according to the recurrence $A\underline{\mathcal{Y}} = \mathcal{Y}\,\mathcal{B}$

Compute inner products between basis vectors in one synchronization

$$\mathcal{G} = \mathcal{Y}^T \mathcal{Y}$$

s-step CG

Key observation: After iteration i, for $j \in \{0, .., s\}$,

$$x_{i+j} - x_i, \quad r_{i+j}, \quad p_{i+j} \quad \in \quad \mathcal{K}_{s+1}(A, p_i) + \mathcal{K}_s(A, r_i)$$

s steps of s-step CG:

Expand solution space s dimensions at once

Compute "basis" matrix $\mathcal{Y}$ such that $\mathrm{span}(\mathcal{Y}) = \mathcal{K}_{s+1}(A, p_i) + \mathcal{K}_s(A, r_i)$ according to the recurrence $A\underline{\mathcal{Y}} = \mathcal{Y}\,\mathcal{B}$

Compute inner products between basis vectors in one synchronization

$$\mathcal{G} = \mathcal{Y}^T\mathcal{Y}$$

Compute s iterations of vector updates

Perform s iterations of vector updates by updating coordinates in basis $\mathcal{Y}$:

$$x_{i+j} - x_i = \mathcal{Y}x_j', \qquad r_{i+j} = \mathcal{Y}r_j', \qquad p_{i+j} = \mathcal{Y}p_j'$$

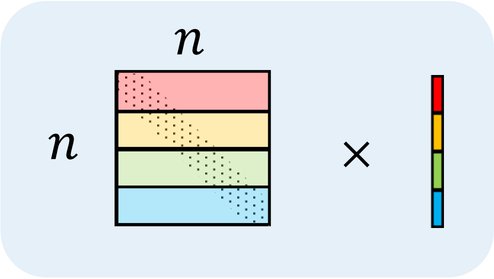
s-step CG

For s iterations of updates, inner products and SpMVs (in basis $\mathcal{Y}$) can be computed by independently by each processor without communication:

s-step CG

For s iterations of updates, inner products and SpMVs (in basis $\mathcal{Y}$) can be computed by independently by each processor without communication:

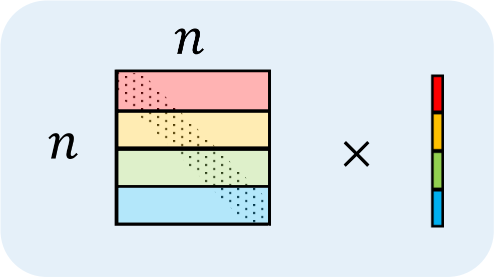
Ap_{i+j}



s-step CG

For s iterations of updates, inner products and SpMVs (in basis $\mathcal{Y}$) can be computed by independently by each processor without communication:

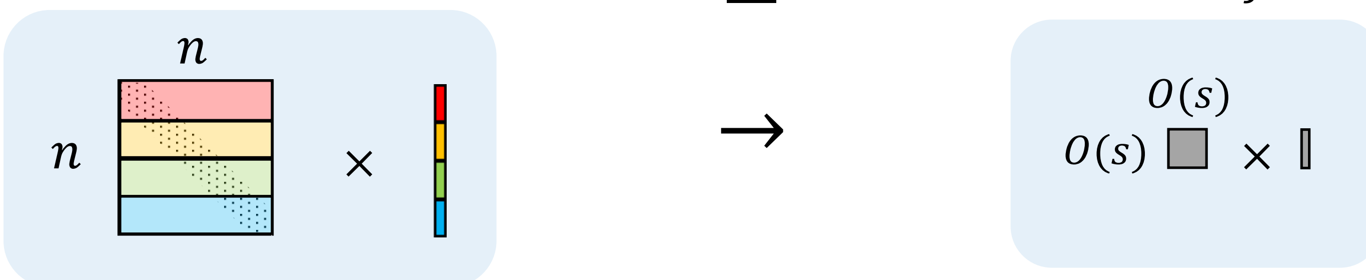
$$A p_{i+j} \qquad = \qquad A\underline{\mathcal{Y}} p'_j$$



s-step CG

For s iterations of updates, inner products and SpMVs (in basis $\mathcal{Y}$) can be computed by independently by each processor without communication:

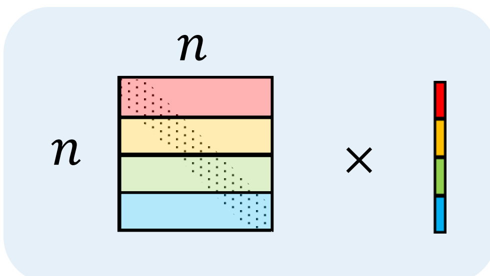
$$Ap_{i+j} \quad = \quad A\underline{\mathcal{Y}}p'_j \quad = \quad \mathcal{Y}(\mathcal{B}p'_j)$$



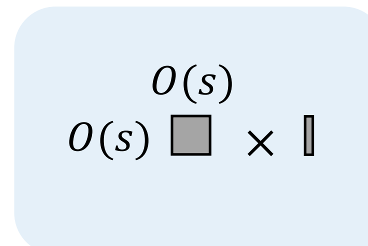
s-step CG

For s iterations of updates, inner products and SpMVs (in basis $\mathcal{Y}$) can be computed by independently by each processor without communication:

$$Ap_{i+j} \quad = \quad A\underline{\mathcal{Y}}p'_j \quad = \quad \mathcal{Y}(\mathcal{B}p'_j)$$



$\rightarrow$

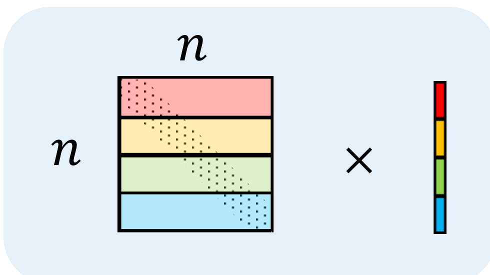


$$(r_{i+j}, r_{i+j})$$

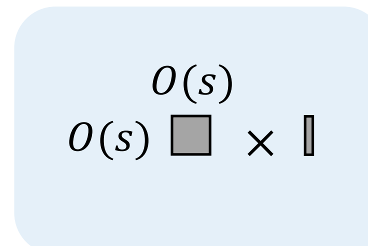
s-step CG

For s iterations of updates, inner products and SpMVs (in basis $\mathcal{Y}$) can be computed by independently by each processor without communication:

$$Ap_{i+j} \quad = \quad A\underline{\mathcal{Y}}p_j' \quad = \quad \mathcal{Y}(\mathcal{B}p_j')$$



$\rightarrow$

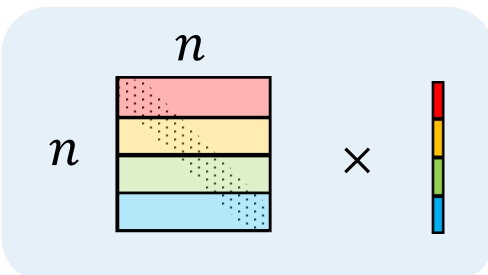


$$(r_{i+j}, r_{i+j}) \quad = \quad r_j'^T \mathcal{Y}^T \mathcal{Y} r_j'$$

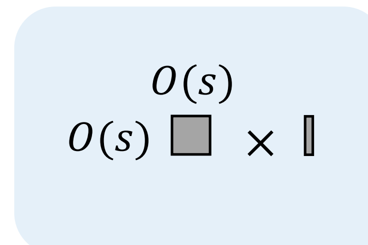
s-step CG

For s iterations of updates, inner products and SpMVs (in basis $\mathcal{Y}$) can be computed by independently by each processor without communication:

$$Ap_{i+j} \quad = \quad A\underline{\mathcal{Y}}p'_j \quad = \quad \mathcal{Y}(\mathcal{B}p'_j)$$

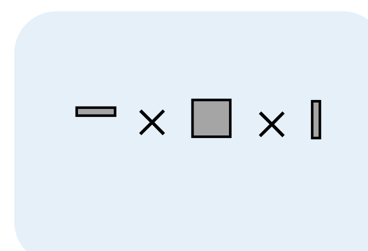


$\rightarrow$



$$(r_{i+j}, r_{i+j}) \quad = \quad r_j'^T \mathcal{Y}^T \mathcal{Y} r_j' \quad = \quad r_j'^T \mathcal{G} r_j'$$

$\rightarrow$



s-step CG

$r_0 = b - Ax_0, p_0 = r_0$

for $k = 0$:nmax/s

Compute $\mathcal{Y}_k$ and $\mathcal{B}_k$ such that $A\underline{\mathcal{Y}}_k = \mathcal{Y}_k \mathcal{B}_k$ and

$$\text{span}(\mathcal{Y}_k) = \mathcal{K}_{s+1}(A, p_{sk}) + \mathcal{K}_s(A, r_{sk})$$

$\mathcal{G}_k = \mathcal{Y}_k^T \mathcal{Y}_k$

$x'_0 = 0, r'_0 = e_{s+2}, p'_0 = e_1$

for $j = 1: s$

$$\alpha_{sk+j-1} = \frac{r'^T_{j-1} \mathcal{G}_k r'_{j-1}}{p'^T_{j-1} \mathcal{G}_k \mathcal{B}_k p'_{j-1}}$$

$$x'_j = x'_{j-1} + \alpha_{sk+j-1} p'_{j-1}$$

$$r'_j = r'_{j-1} - \alpha_{sk+j-1} \mathcal{B}_k p'_{j-1}$$

$$\beta_{sk+j} = \frac{r'^T_j \mathcal{G}_k r'_j}{r'^T_{j-1} \mathcal{G}_k r'_{j-1}}$$

$$p'_j = r'_j + \beta_{sk+j} p'_{j-1}$$

end

$[x_{s(k+1)} - x_{sk}, r_{s(k+1)}, p_{s(k+1)}] = \mathcal{Y}_k [x'_s, r'_s, p'_s]$

end

s-step CG

$r_0 = b - Ax_0, p_0 = r_0$

for $k = 0$:nmax/s

Compute $\mathcal{Y}_k$ and $\mathcal{B}_k$ such that $A\underline{\mathcal{Y}}_k = \mathcal{Y}_k\mathcal{B}_k$ and

$$\text{span}(\mathcal{Y}_k) = \mathcal{K}_{s+1}(A, p_{sk}) + \mathcal{K}_s(A, r_{sk})$$

$\mathcal{G}_k = \mathcal{Y}_k^T\mathcal{Y}_k$

$x_0' = 0, r_0' = e_{s+2}, p_0' = e_1$

for $j = 1: s$

$$\alpha_{sk+j-1} = \frac{r_{j-1}'^T \mathcal{G}_k r_{j-1}'}{p_{j-1}'^T \mathcal{G}_k \mathcal{B}_k p_{j-1}'}$$

$$x_j' = x_{j-1}' + \alpha_{sk+j-1} p_{j-1}'$$

$$r_j' = r_{j-1}' - \alpha_{sk+j-1} \mathcal{B}_k p_{j-1}'$$

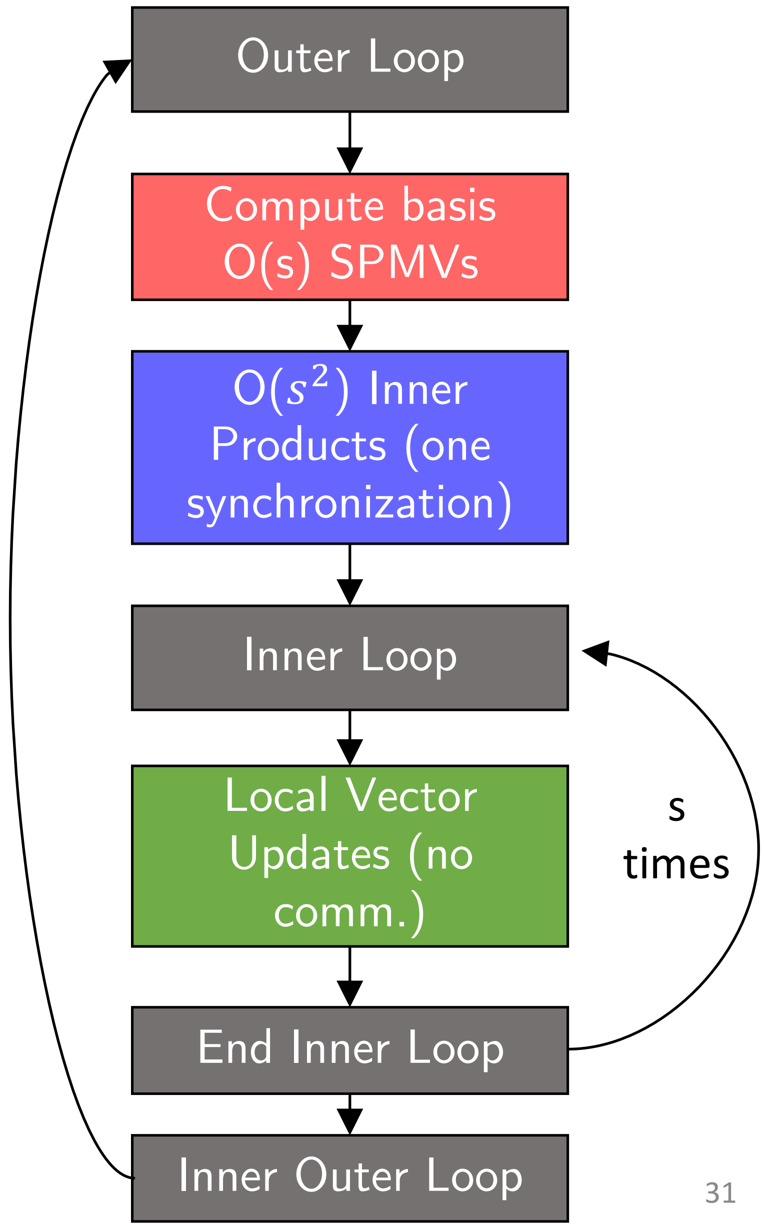
$$\beta_{sk+j} = \frac{r_j'^T \mathcal{G}_k r_j'}{r_{j-1}'^T \mathcal{G}_k r_{j-1}'}$$

$$p_j' = r_j' + \beta_{sk+j} p_{j-1}'$$

end

$[x_{s(k+1)} - x_{sk}, r_{s(k+1)}, p_{s(k+1)}] = \mathcal{Y}_k[x_s', r_s', p_s']$

end



s-step CG

$r_0 = b - Ax_0, p_0 = r_0$

for k = 0:nmax/s

Compute $\mathcal{Y}_k$ and $\mathcal{B}_k$ such that $A\mathcal{Y}_k = \mathcal{Y}_k\mathcal{B}_k$ and span$(\mathcal{Y}_k) = \mathcal{K}_{s+1}(A, p_{sk}) + \mathcal{K}_s(A, r_{sk})$

$\mathcal{G}_k = \mathcal{Y}_k^T\mathcal{Y}_k$

$x_0' = 0, r_0' = e_{s+2}, p_0' = e_1$

for $j = 1{:}\, s$

$$\alpha_{sk+j-1} = \frac{r_{j-1}'^T \mathcal{G}_k r_{j-1}'}{p_{j-1}'^T \mathcal{G}_k \mathcal{B}_k p_{j-1}'}$$

$$x_j' = x_{j-1}' + \alpha_{sk+j-1} p_{j-1}'$$

$$r_j' = r_{j-1}' - \alpha_{sk+j-1} \mathcal{B}_k p_{j-1}'$$

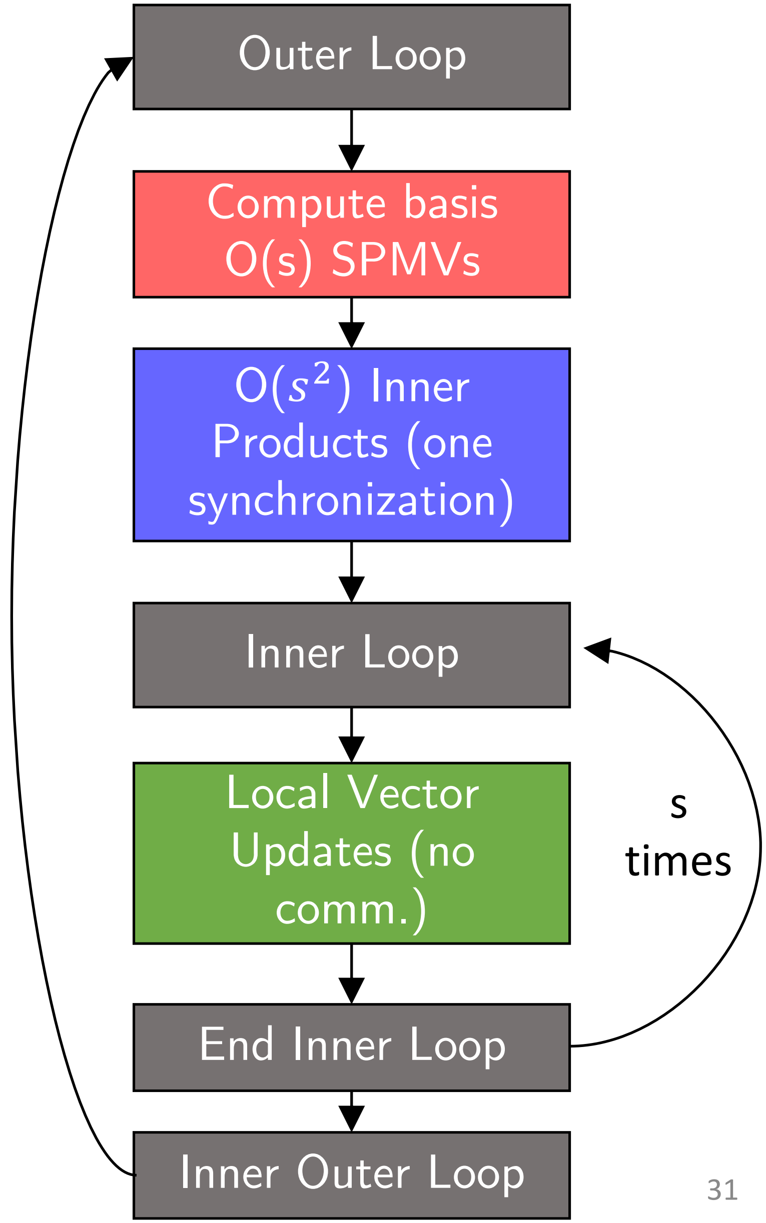
$$\beta_{sk+j} = \frac{r_j'^T \mathcal{G}_k r_j'}{r_{j-1}'^T \mathcal{G}_k r_{j-1}'}$$

$$p_j' = r_j' + \beta_{sk+j} p_{j-1}'$$

end

$[x_{s(k+1)} - x_{sk}, r_{s(k+1)}, p_{s(k+1)}] = \mathcal{Y}_k[x_s', r_s', p_s']$

end



s-step CG

$r_0 = b - Ax_0, p_0 = r_0$

for k = 0:nmax/s

Compute $\mathcal{Y}_k$ and $\mathcal{B}_k$ such that $A\mathcal{Y}_k = \mathcal{Y}_k\mathcal{B}_k$ and

span$(\mathcal{Y}_k) = \mathcal{K}_{s+1}(A, p_{sk}) + \mathcal{K}_s(A, r_{sk})$

$\mathcal{G}_k = \mathcal{Y}_k^T\mathcal{Y}_k$

$x'_0 = 0, r'_0 = e_{s+2}, p'_0 = e_1$

for $j = 1{:}\,s$

$$\alpha_{sk+j-1} = \frac{r'^T_{j-1}\mathcal{G}_k r'_{j-1}}{p'^T_{j-1}\mathcal{G}_k\mathcal{B}_k p'_{j-1}}$$

$$x'_j = x'_{j-1} + \alpha_{sk+j-1}p'_{j-1}$$

$$r'_j = r'_{j-1} - \alpha_{sk+j-1}\mathcal{B}_k p'_{j-1}$$

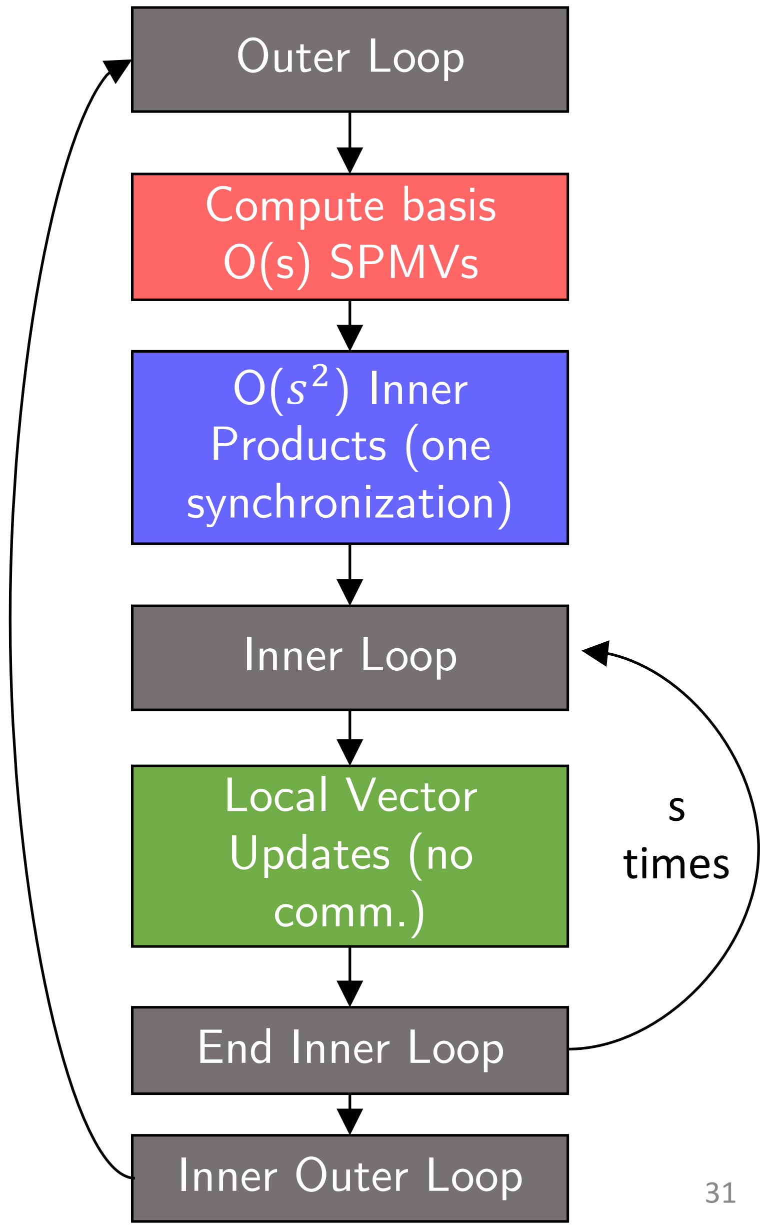
$$\beta_{sk+j} = \frac{r'^T_j\mathcal{G}_k r'_j}{r'^T_{j-1}\mathcal{G}_k r'_{j-1}}$$

$$p'_j = r'_j + \beta_{sk+j}p'_{j-1}$$

end

$[x_{s(k+1)} - x_{sk}, r_{s(k+1)}, p_{s(k+1)}] = \mathcal{Y}_k[x'_s, r'_s, p'_s]$

end



s-step CG

$r_0 = b - Ax_0, p_0 = r_0$

for k = 0:nmax/s

Compute $\mathcal{Y}_k$ and $\mathcal{B}_k$ such that $A\underline{\mathcal{Y}}_k = \mathcal{Y}_k\mathcal{B}_k$ and

$\text{span}(\mathcal{Y}_k) = \mathcal{K}_{s+1}(A, p_{sk}) + \mathcal{K}_s(A, r_{sk})$

$\mathcal{G}_k = \mathcal{Y}_k^T\mathcal{Y}_k$

$x_0' = 0, r_0' = e_{s+2}, p_0' = e_1$

for $j = 1{:}\,s$

$$\alpha_{sk+j-1} = \frac{r_{j-1}'^T\mathcal{G}_k r_{j-1}'}{p_{j-1}'^T\mathcal{G}_k\mathcal{B}_k p_{j-1}'}$$

$$x_j' = x_{j-1}' + \alpha_{sk+j-1}p_{j-1}'$$

$$r_j' = r_{j-1}' - \alpha_{sk+j-1}\mathcal{B}_k p_{j-1}'$$

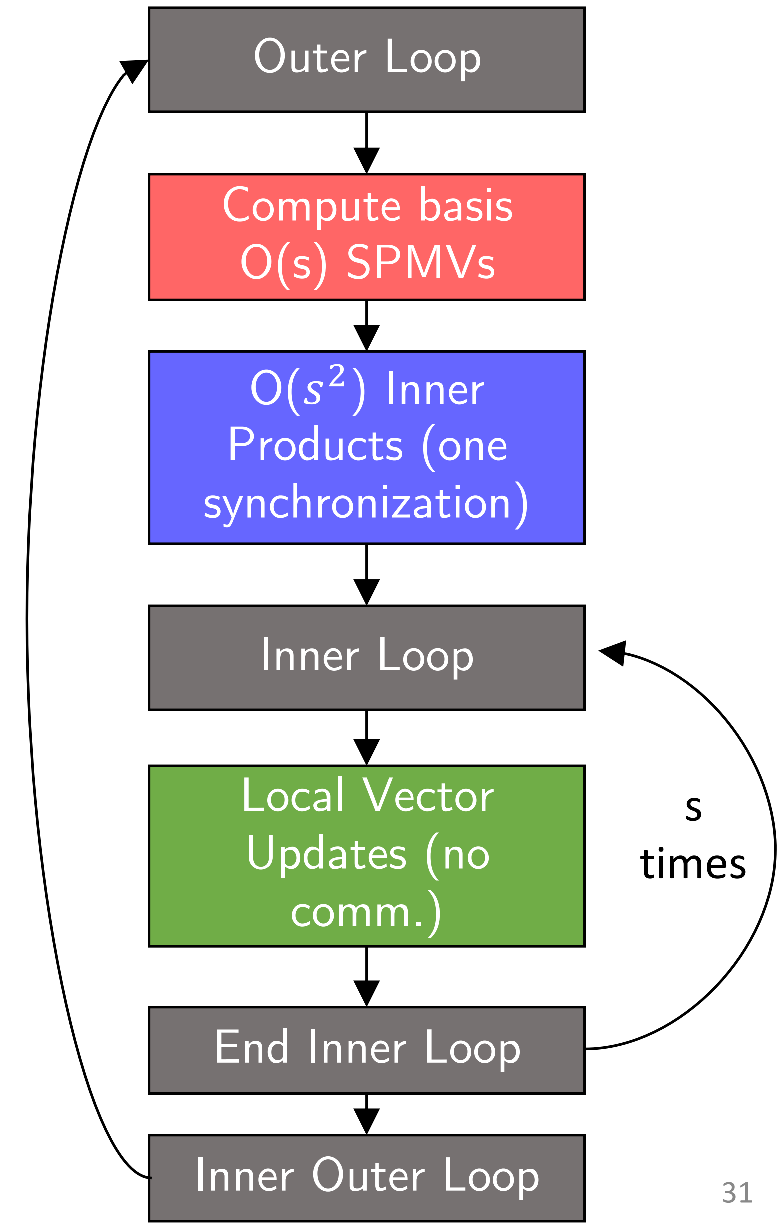
$$\beta_{sk+j} = \frac{r_j'^T\mathcal{G}_k r_j'}{r_{j-1}'^T\mathcal{G}_k r_{j-1}'}$$

$$p_j' = r_j' + \beta_{sk+j}p_{j-1}'$$

end

$[x_{s(k+1)} - x_{sk}, r_{s(k+1)}, p_{s(k+1)}] = \mathcal{Y}_k[x_s', r_s', p_s']$

end



Sources of local roundoff error in s-step CG

Computing the s-step Krylov subspace basis:

$$A\underline{\hat{\mathcal{Y}}}_k = \hat{\mathcal{Y}}_k \mathcal{B}_k + \Delta \mathcal{Y}_k$$

Updating coordinate vectors in the inner loop:

$$\hat{x}'_{k,j} = \hat{x}'_{k,j-1} + \hat{q}'_{k,j-1} + \xi_{k,j}$$

$$\hat{r}'_{k,j} = \hat{r}'_{k,j-1} - \mathcal{B}_k\, \hat{q}'_{k,j-1} + \eta_{k,j}$$

$$\text{with} \quad \hat{q}'_{k,j-1} = \mathrm{fl}(\hat{\alpha}_{sk+j-1}\hat{p}'_{k,j-1})$$

Recovering CG vectors for use in next outer loop:

$$\hat{x}_{sk+j} = \hat{\mathcal{Y}}_k \hat{x}'_{k,j} + \hat{x}_{sk} + \phi_{sk+j}$$

$$\hat{r}_{sk+j} = \hat{\mathcal{Y}}_k \hat{r}'_{k,j} + \psi_{sk+j}$$

Sources of local roundoff error in s-step CG

Computing the s-step Krylov subspace basis:

$$A\underline{\hat{\mathcal{Y}}}_k = \hat{\mathcal{Y}}_k \mathcal{B}_k + \boxed{\Delta \mathcal{Y}_k}$$

Error in computing s-step basis

Updating coordinate vectors in the inner loop:

$$\hat{x}'_{k,j} = \hat{x}'_{k,j-1} + \hat{q}'_{k,j-1} + \xi_{k,j}$$

$$\hat{r}'_{k,j} = \hat{r}'_{k,j-1} - \mathcal{B}_k\, \hat{q}'_{k,j-1} + \eta_{k,j}$$

$$\text{with} \quad \hat{q}'_{k,j-1} = \mathrm{fl}(\hat{\alpha}_{sk+j-1} \hat{p}'_{k,j-1})$$

Recovering CG vectors for use in next outer loop:

$$\hat{x}_{sk+j} = \hat{\mathcal{Y}}_k \hat{x}'_{k,j} + \hat{x}_{sk} + \phi_{sk+j}$$

$$\hat{r}_{sk+j} = \hat{\mathcal{Y}}_k \hat{r}'_{k,j} + \psi_{sk+j}$$

Sources of local roundoff error in s-step CG

Computing the s-step Krylov subspace basis:

$$A\underline{\hat{\mathcal{Y}}}_k = \hat{\mathcal{Y}}_k \mathcal{B}_k + \boxed{\Delta\mathcal{Y}_k}$$

Error in computing s-step basis

Updating coordinate vectors in the inner loop:

$$\hat{x}'_{k,j} = \hat{x}'_{k,j-1} + \hat{q}'_{k,j-1} + \boxed{\xi_{k,j}}$$

$$\hat{r}'_{k,j} = \hat{r}'_{k,j-1} - \mathcal{B}_k\, \hat{q}'_{k,j-1} + \boxed{\eta_{k,j}}$$

$$\text{with} \quad \hat{q}'_{k,j-1} = \mathrm{fl}(\hat{\alpha}_{sk+j-1}\hat{p}'_{k,j-1})$$

Error in updating coefficient vectors

Recovering CG vectors for use in next outer loop:

$$\hat{x}_{sk+j} = \hat{\mathcal{Y}}_k \hat{x}'_{k,j} + \hat{x}_{sk} + \phi_{sk+j}$$

$$\hat{r}_{sk+j} = \hat{\mathcal{Y}}_k \hat{r}'_{k,j} + \psi_{sk+j}$$

Sources of local roundoff error in s-step CG

Computing the s-step Krylov subspace basis:

$$A\underline{\hat{\mathcal{Y}}_k} = \hat{\mathcal{Y}}_k \mathcal{B}_k + \Delta\mathcal{Y}_k$$

Error in computing s-step basis

Updating coordinate vectors in the inner loop:

$$\hat{x}'_{k,j} = \hat{x}'_{k,j-1} + \hat{q}'_{k,j-1} + \xi_{k,j}$$

$$\hat{r}'_{k,j} = \hat{r}'_{k,j-1} - \mathcal{B}_k\, \hat{q}'_{k,j-1} + \eta_{k,j}$$

$$\text{with} \quad \hat{q}'_{k,j-1} = \mathrm{fl}(\hat{\alpha}_{sk+j-1}\hat{p}'_{k,j-1})$$

Error in updating coefficient vectors

Recovering CG vectors for use in next outer loop:

$$\hat{x}_{sk+j} = \hat{\mathcal{Y}}_k \hat{x}'_{k,j} + \hat{x}_{sk} + \phi_{sk+j}$$

$$\hat{r}_{sk+j} = \hat{\mathcal{Y}}_k \hat{r}'_{k,j} + \psi_{sk+j}$$

Error in basis change

Attainable accuracy of s-step CG

- We can write the gap between the true and updated residuals f in terms of these errors:

$$f_{sk+j} = f_0$$
$$-\sum_{\ell=0}^{k-1}\left[A\phi_{s\ell+s} + \psi_{s\ell+s} + \sum_{i=1}^{s}\left[A\hat{\mathcal{Y}}_\ell \xi_{\ell,i} + \hat{\mathcal{Y}}_\ell \eta_{\ell,i} - \Delta\mathcal{Y}_\ell \hat{q}'_{\ell,i-1}\right]\right]$$
$$-A\phi_{sk+j} - \psi_{sk+j} - \sum_{i=1}^{j}\left[A\hat{\mathcal{Y}}_k \xi_{k,i} + \hat{\mathcal{Y}}_k \eta_{k,i} - \Delta\mathcal{Y}_\ell \hat{q}'_{k,i-1}\right]$$

- Using standard rounding error results, this allows us to obtain an upper bound on $\|f_{sk+j}\|$.

Attainable accuracy of s-step CG

- We can write the gap between the true and updated residuals f in terms of these errors:

$$
\begin{aligned}
f_{sk+j} &= f_0 \\
&\quad - \sum_{\ell=0}^{k-1}\left[A\phi_{s\ell+s} + \psi_{s\ell+s} + \sum_{i=1}^{s}\left[A\hat{\mathcal{Y}}_\ell \xi_{\ell,i} + \hat{\mathcal{Y}}_\ell \eta_{\ell,i} - \Delta\mathcal{Y}_\ell \hat{q}'_{\ell,i-1}\right]\right] \\
&\quad - A\phi_{sk+j} - \psi_{sk+j} - \sum_{i=1}^{j}\left[A\hat{\mathcal{Y}}_k \xi_{k,i} + \hat{\mathcal{Y}}_k \eta_{k,i} - \Delta\mathcal{Y}_\ell \hat{q}'_{k,i-1}\right]
\end{aligned}
$$

- Using standard rounding error results, this allows us to obtain an upper bound on $\|f_{sk+j}\|$.

Attainable accuracy of s-step CG

- We can write the gap between the true and updated residuals f in terms of these errors:

$$f_{sk+j} = f_0$$

$$-\sum_{\ell=0}^{k-1}\left[A\phi_{s\ell+s} + \psi_{s\ell+s} + \sum_{i=1}^{s}\left[A\hat{\mathcal{Y}}_\ell\xi_{\ell,i} + \hat{\mathcal{Y}}_\ell\eta_{\ell,i} - \Delta\mathcal{Y}_\ell\hat{q}'_{\ell,i-1}\right]\right]$$

$$-A\phi_{sk+j} - \psi_{sk+j} - \sum_{i=1}^{j}\left[A\hat{\mathcal{Y}}_k\xi_{k,i} + \hat{\mathcal{Y}}_k\eta_{k,i} - \Delta\mathcal{Y}_\ell\hat{q}'_{k,i-1}\right]$$

- Using standard rounding error results, this allows us to obtain an upper bound on $\|f_{sk+j}\|$.

Attainable accuracy of s-step CG

- We can write the gap between the true and updated residuals f in terms of these errors:

$$f_{sk+j} = f_0$$

$$-\sum_{\ell=0}^{k-1}\left[A\phi_{s\ell+s} + \psi_{s\ell+s} + \sum_{i=1}^{s}\left[A\hat{\mathcal{Y}}_\ell\xi_{\ell,i} + \hat{\mathcal{Y}}_\ell\eta_{\ell,i} - \Delta\mathcal{Y}_\ell\hat{q}'_{\ell,i-1}\right]\right]$$

$$-A\phi_{sk+j} - \psi_{sk+j} - \sum_{i=1}^{j}\left[A\hat{\mathcal{Y}}_k\xi_{k,i} + \hat{\mathcal{Y}}_k\eta_{k,i} - \Delta\mathcal{Y}_\ell\hat{q}'_{k,i-1}\right]$$

- Using standard rounding error results, this allows us to obtain an upper bound on $\|f_{sk+j}\|$.

Attainable accuracy of s-step CG

$f_i \equiv b - A\hat{x}_i - \hat{r}_i$

For CG:

$$\|f_i\| \leq \|f_0\| + \varepsilon \sum_{m=1}^{i} (1+N)\|A\|\|\hat{x}_m\| + \|\hat{r}_m\|$$

Attainable accuracy of s-step CG

$f_i \equiv b - A\hat{x}_i - \hat{r}_i$

For CG:

$$\|f_i\| \leq \|f_0\| + \varepsilon \sum_{m=1}^{i} (1+N)\|A\|\|\hat{x}_m\| + \|\hat{r}_m\|$$

For s-step CG: $i \equiv sk + j$

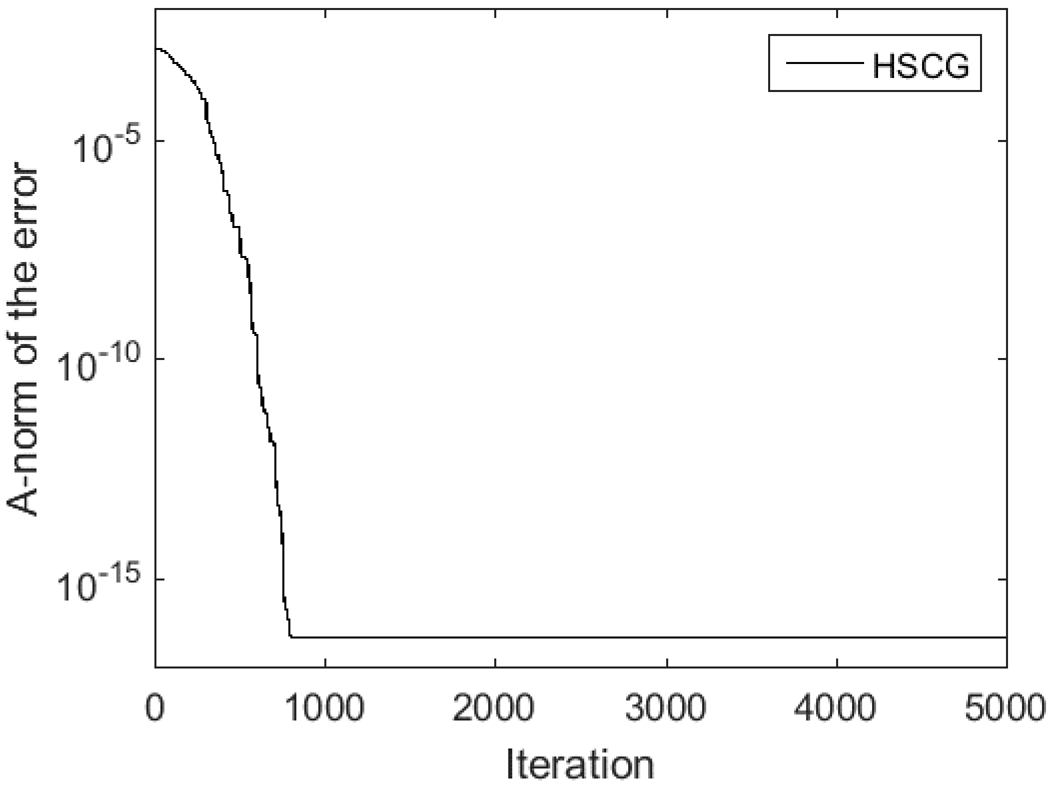
$$\|f_{sk+j}\| \leq \|f_0\| + \varepsilon \boldsymbol{c}\boldsymbol{\bar{\Gamma}_k} \sum_{m=1}^{sk+j} (1+N)\|A\|\|\hat{x}_m\| + \|\hat{r}_m\|$$

where c is a low-degree polynomial in s, and

$$\bar{\Gamma}_k = \max_{\ell \leq k} \Gamma_\ell\ , \qquad \text{where} \qquad \Gamma_\ell = \|\hat{\mathcal{Y}}_\ell^+\| \cdot \||\hat{\mathcal{Y}}_\ell|\|$$

(see C., 2015)

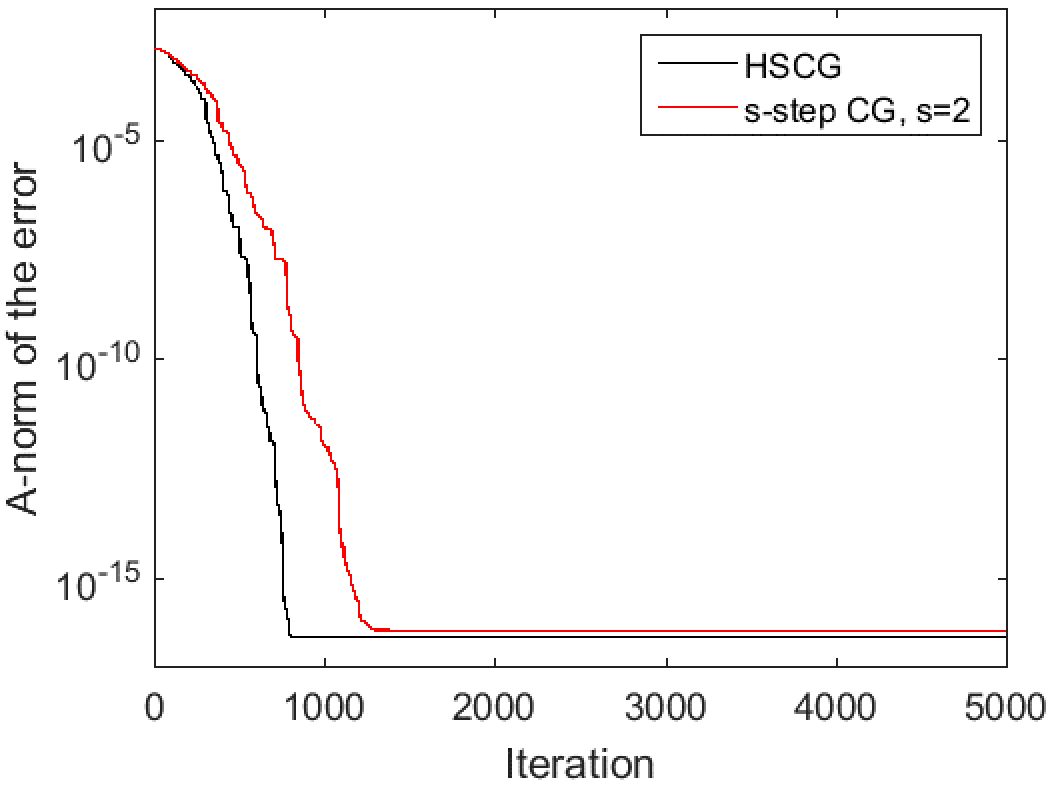
s-step CG



HSCG
10^{-5}
10^{-10}
10^{-15}
A-norm of the error
0
1000
2000
3000
4000
5000
Iteration

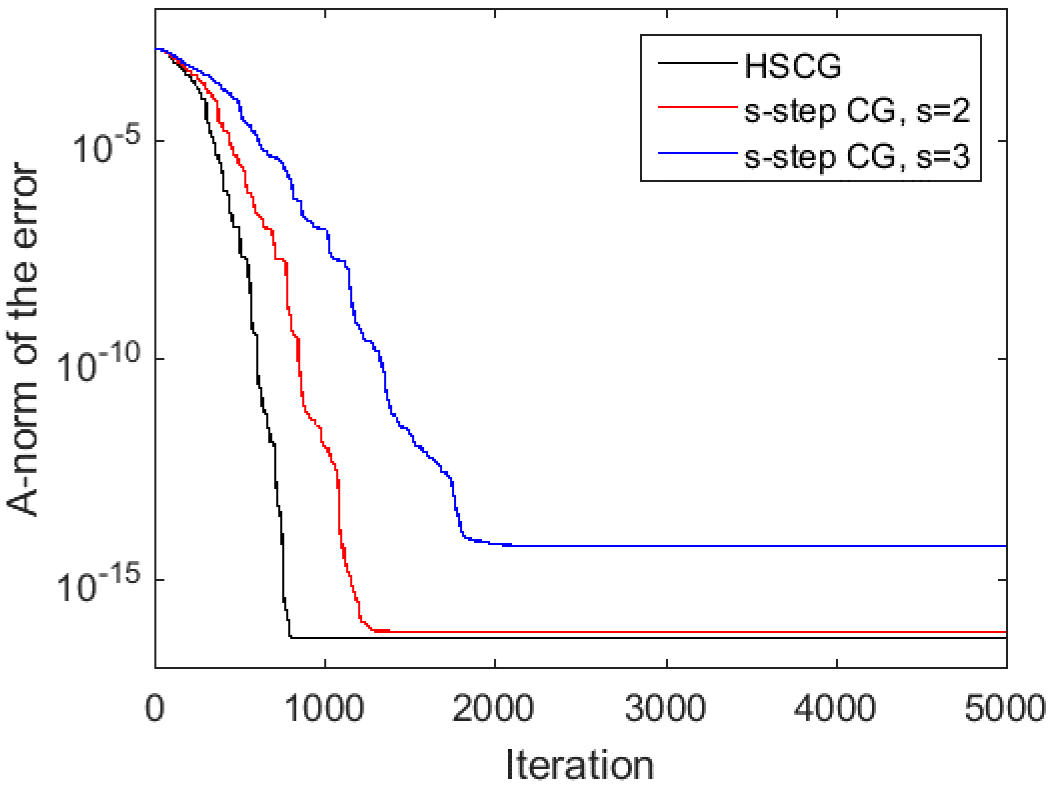
s-step CG

s-step CG with monomial basis ($\mathcal{Y} = [p_i, Ap_i, \dots, A^s p_i, r_i, Ar_i, \dots A^{s-1} r_i]$)



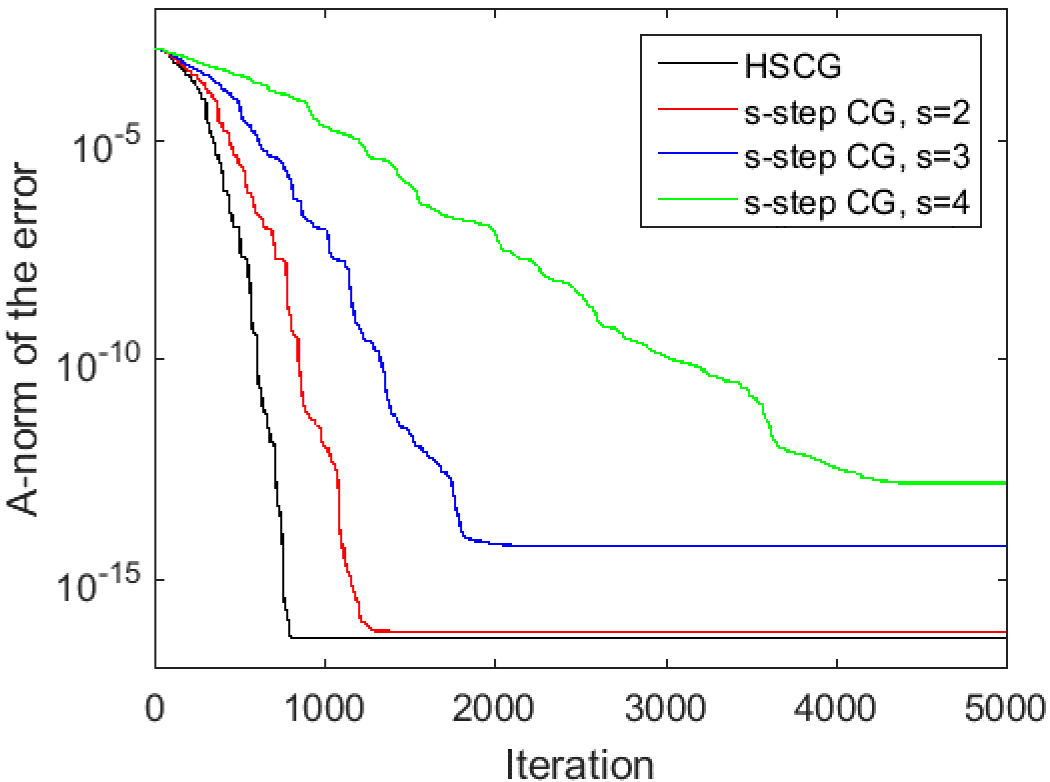
s-step CG

s-step CG with monomial basis ($\mathcal{Y} = [p_i, Ap_i, \dots, A^s p_i, r_i, Ar_i, \dots A^{s-1} r_i]$)



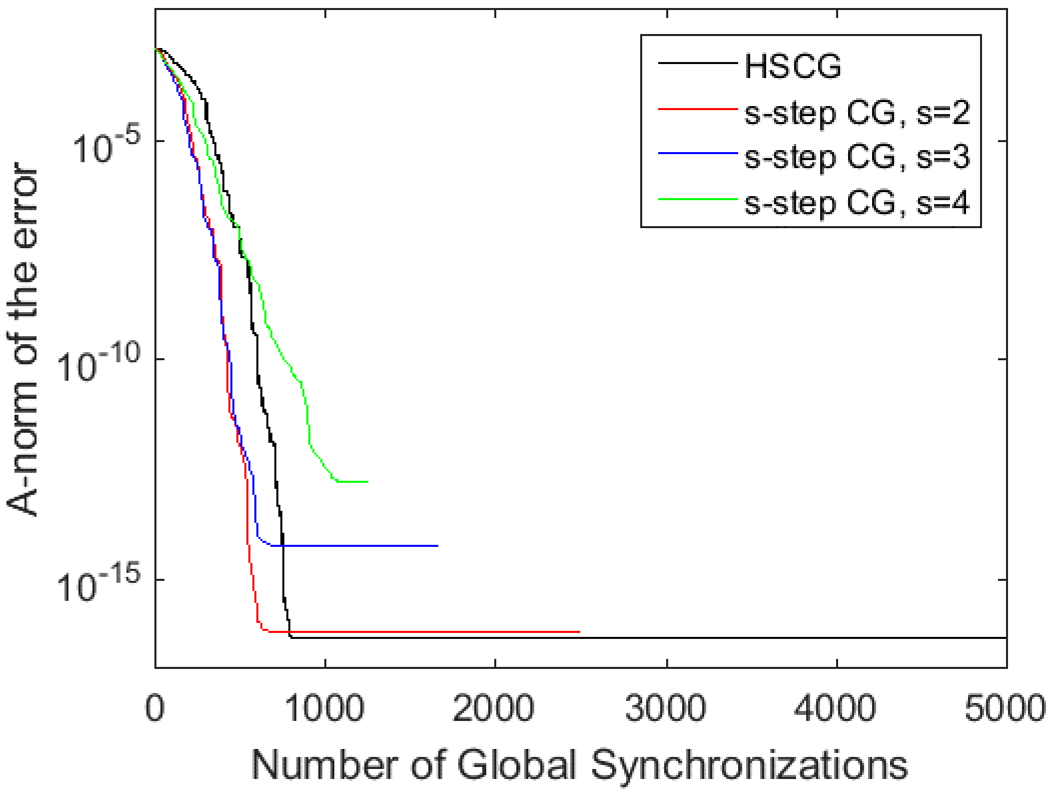
s-step CG

s-step CG with monomial basis ($\mathcal{Y} = [p_i, Ap_i, \dots, A^s p_i, r_i, Ar_i, \dots A^{s-1} r_i]$)



Can also use other, more well-conditioned bases to improve convergence rate and accuracy (see, e.g. Philippe and Reichel, 2012).

s-step CG



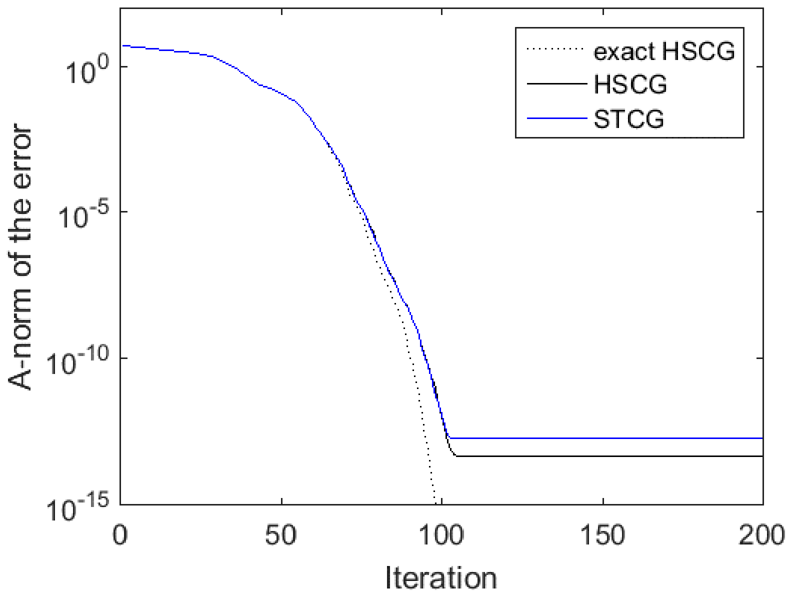
- Even assuming perfect parallel scalability with s (which is usually not the case due to extra SpMVs and inner products), already at $s = 4$ we are worse than HSCG in terms of number of synchronizations!

A different problem...

A: **nos4** from UFSMC,
b: equal components in the eigenbasis of A and $\|b\| = 1$
$N = 100, \kappa(A) \approx 2\text{e}3$

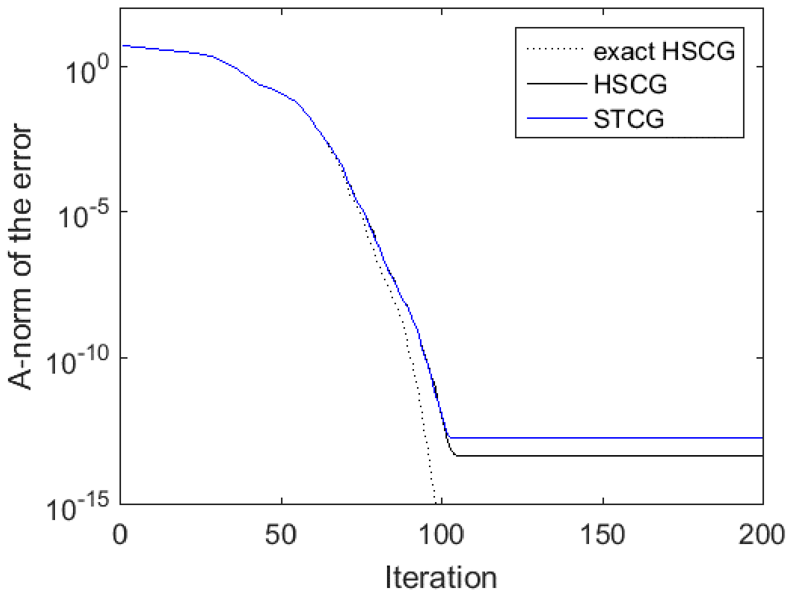
A different problem...

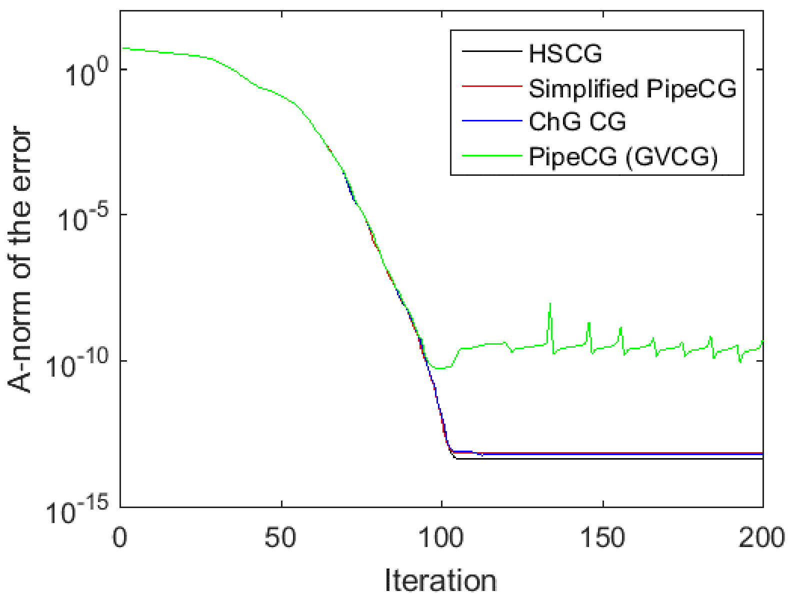
A: **nos4** from UFSMC,
b: equal components in the eigenbasis
of A and $\|b\| = 1$
$N = 100, \kappa(A) \approx 2e3$



A different problem...

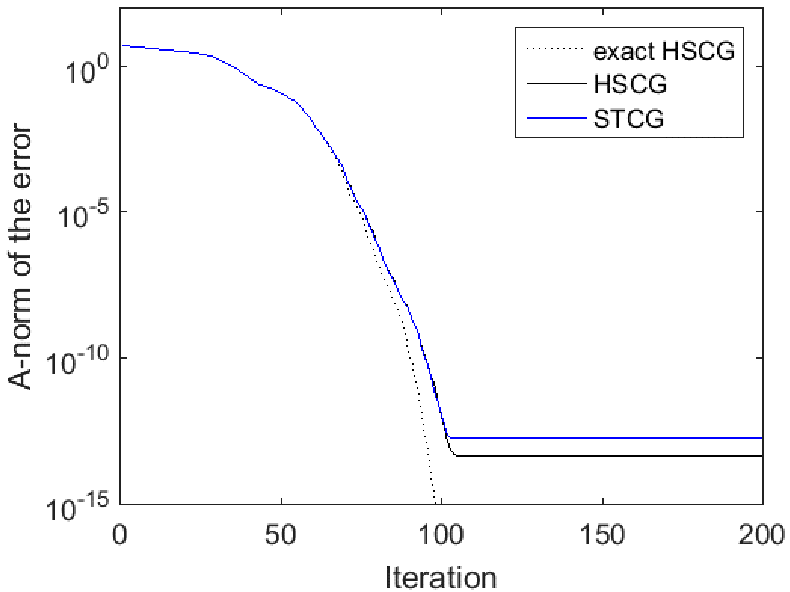
A: **nos4** from UFSMC,
b: equal components in the eigenbasis of A and $\|b\| = 1$
$N = 100, \kappa(A) \approx 2e3$

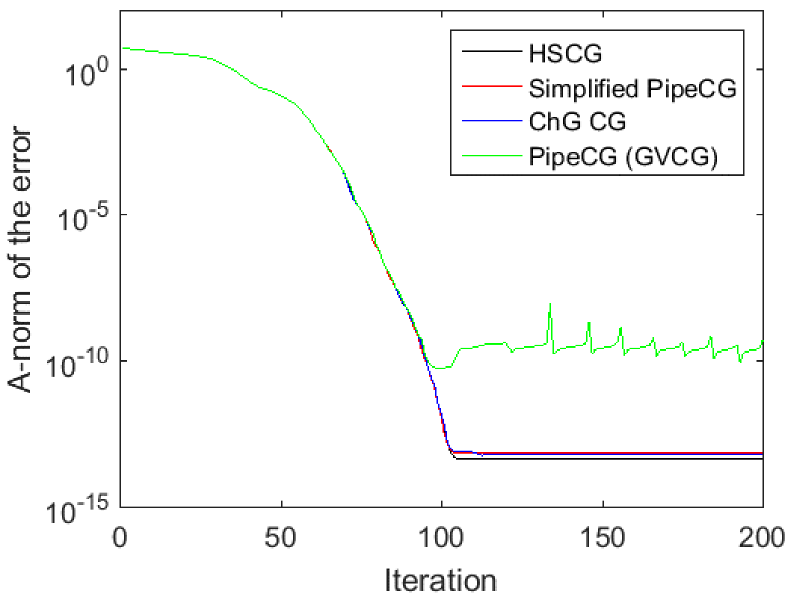


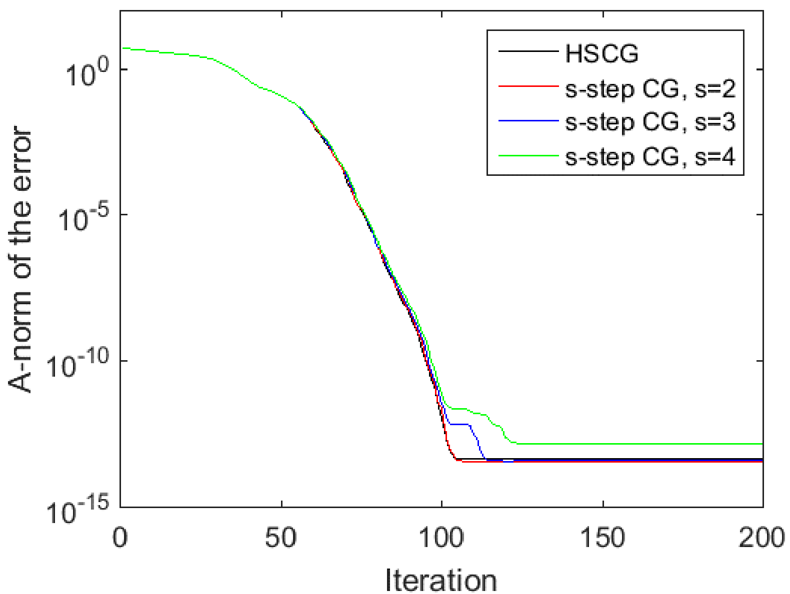


A different problem...

A: **nos4** from UFSMC,
b: equal components in the eigenbasis of A and $\|b\| = 1$
$N = 100, \kappa(A) \approx 2e3$



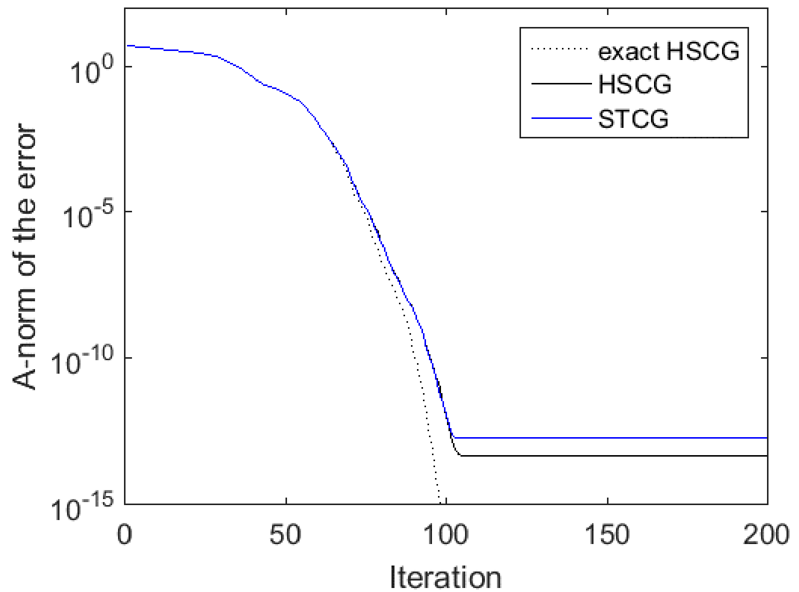


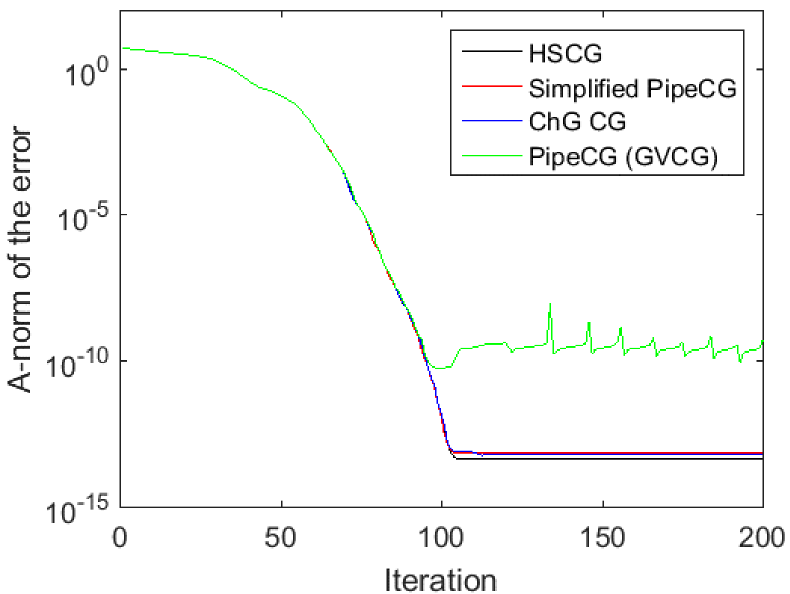


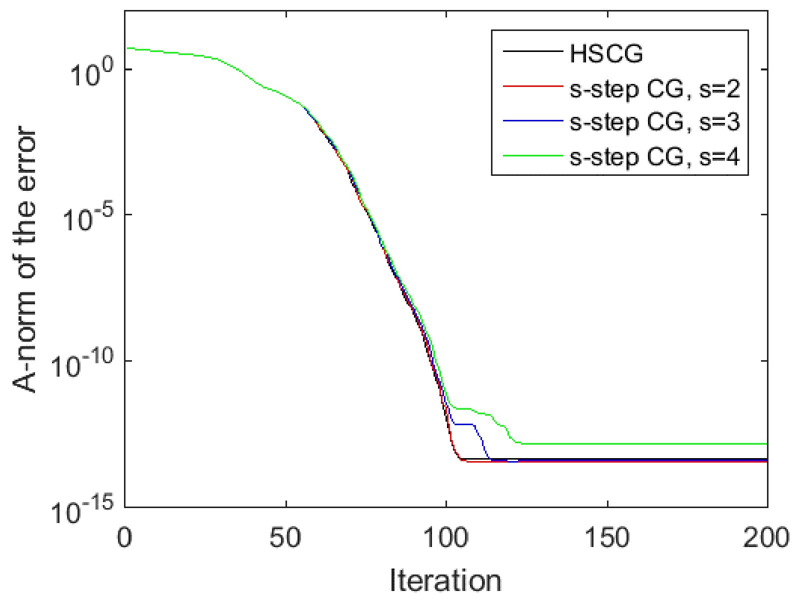
A different problem...

A: **nos4** from UFSMC,
b: equal components in the eigenbasis of A and $\|b\| = 1$
$N = 100, \kappa(A) \approx 2\text{e}3$

If application only requires $\|x - x_i\|_A \leq 10^{-10}$, any of these methods will work!





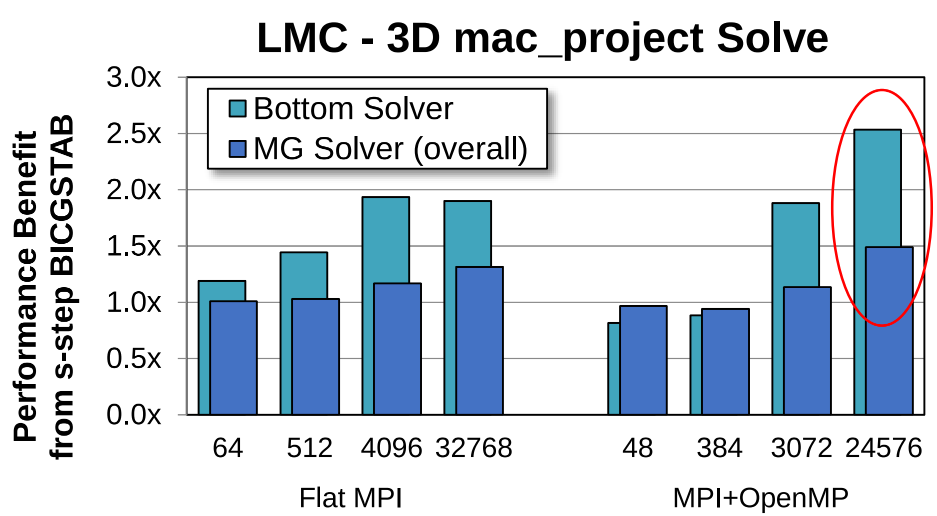


Speedups for real applications

- s-step BICGSTAB bottom-solver implemented in BoxLib (AMR framework from LBL)

Low Mach Number Combustion Code (LMC): gas-phase combustion simulation

- Compared GMG with BICGSTAB vs. GMG with s-step BICGSTAB (s=4) on a Cray XE6 for two different applications
- Up to 2.5x speedup in bottom solve; up to 1.5x in overall MG solve



(see Williams et al., IPDPS 2014)

Conclusions and takeaways

- Think of the bigger picture
  - Much focus on modifying methods to speed up iterations
  - But the speed of an iteration only part of the runtime:

runtime = (time/iteration) x (#iterations)

Conclusions and takeaways

- Think of the bigger picture
  - Much focus on modifying methods to speed up iterations
  - But the speed of an iteration only part of the runtime:

    runtime = (time/iteration) x (#iterations)
  - And a solver that can't solve to required accuracy is useless!

Conclusions and takeaways

- Think of the bigger picture
  - Much focus on modifying methods to speed up iterations
  - But the speed of an iteration only part of the runtime:

    runtime = (time/iteration) x (#iterations)
  - And a solver that can't solve to required accuracy is useless!
  - Solver runtime is only part of a larger scientific code

Conclusions and takeaways

- Think of the bigger picture
  - Much focus on modifying methods to speed up iterations
  - But the speed of an iteration only part of the runtime:

    runtime = (time/iteration) x (#iterations)
  - And a solver that can't solve to required accuracy is useless!
  - Solver runtime is only part of a larger scientific code

- Design and implementation of iterative solvers requires a *holistic* approach
  - Selecting the right method, parameters, stopping criteria
  - Selecting the right preconditioner (closely linked with the discretization! see Málek and Strakoš, 2015)

Conclusions and takeaways

- Think of the bigger picture
  - Much focus on modifying methods to speed up iterations
  - But the speed of an iteration only part of the runtime:

    runtime = (time/iteration) x (#iterations)
  - And a solver that can't solve to required accuracy is useless!
  - Solver runtime is only part of a larger scientific code

- Design and implementation of iterative solvers requires a *holistic* approach
  - Selecting the right method, parameters, stopping criteria
  - Selecting the right preconditioner (closely linked with the discretization! see Málek and Strakoš, 2015)

- Key challenge: identify problems (or classes of problems) for which synchronization-reducing Krylov subspace methods can reduce runtime while meeting application-specific accuracy constraints

⇒ Requires understanding the effects of finite precision computations on convergence rate and accuracy

Thank You!

erinc@cims.nyu.edu

math.nyu.edu/~erinc