The cost of iterative computations at scale

Erin C. Carson

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

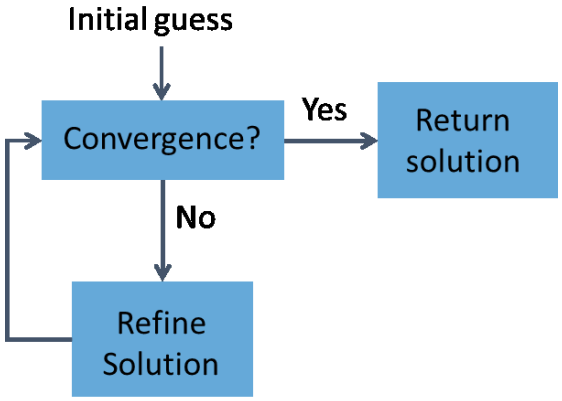
Irish Numerical Analysis Forum

Dublin, Ireland

July 1, 2021

FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

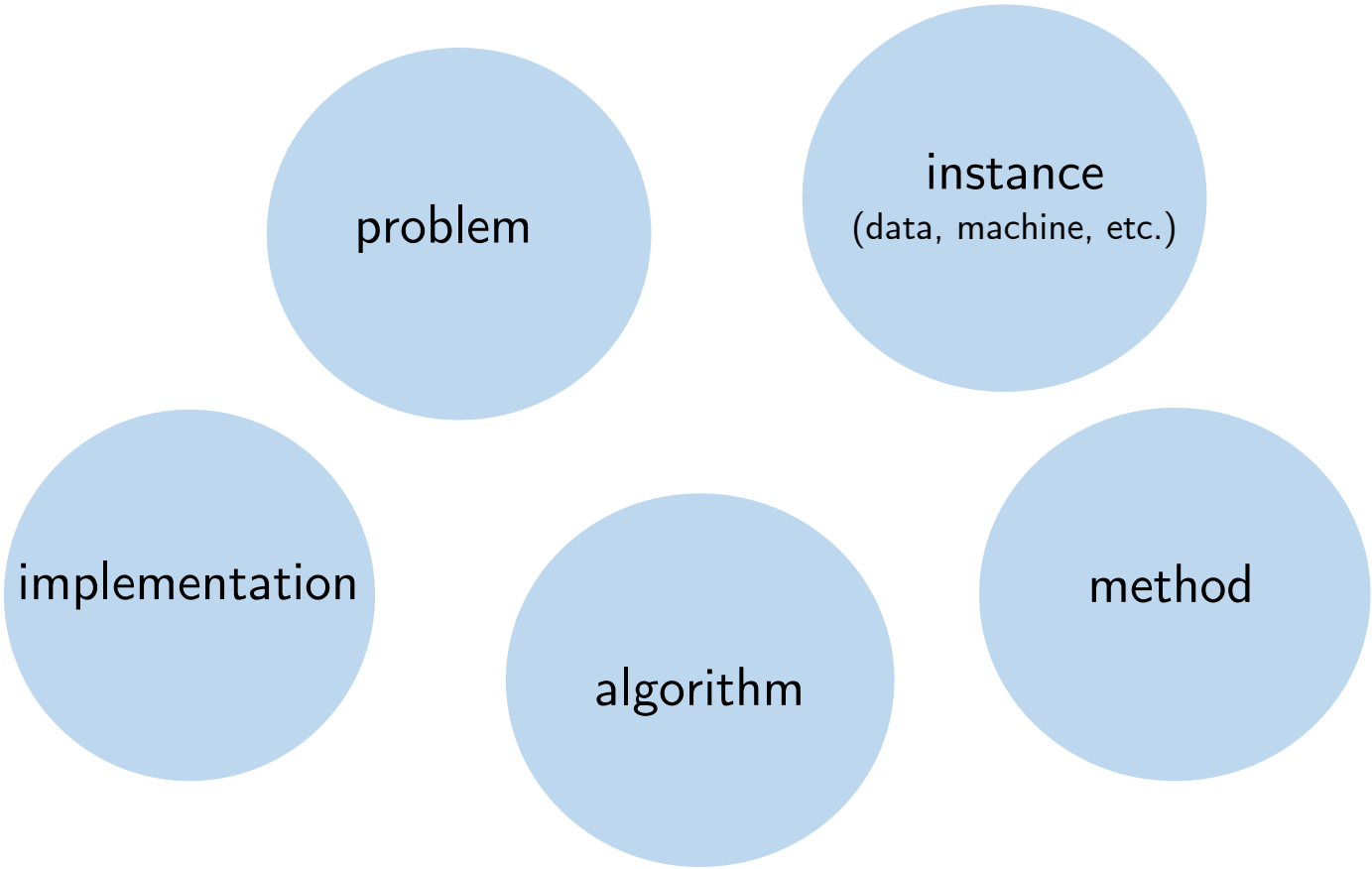
- Iterative linear algebra computations
  - Eigenvalue problems, linear systems, etc.
  - Matrix A typically large and sparse
- Ubiquitous in applications, but typically incapable of fully exploiting underlying hardware
- Has motivated the development of new methods, algorithms, and implementations (and hardware!) to improve performance



- Goal: determine the optimal computational approach for a given machine and a given instance of data A and b

What is the cost of an iterative computation?

What is a computation?



1.The Problem

Ultimately, the computation is performed in order to solve some *problem*

Running example: Solve $N \times N$ linear system $Ax = b$

Must also consider greater context

- The origin of the problem dictates mathematical structure

Consider the infinite dimensional problem

$$\mathcal{G}u = f$$

where $\mathcal{G}: S \to S$ is a bounded invertible operator on Hilbert space S

The problem is approximated on a finite dimensional subspace $S_h \subset S$ by the finite dimensional operator $\mathcal{G}_h$, giving

$$\mathcal{G}_h u_h = f_h$$

Choosing a basis for S_h gives rise to the matrix problem

$$Ax = b$$

2.The Method

- Mathematical approach for transforming the data
- Many possible choices depending on the properties and structure of A
  - Dense $A \rightarrow$ LU (or Cholesky if SPD)
  - Large, sparse, and SPD $A \rightarrow$ Conjugate Gradient method

Krylov subspace method

Krylov subspace method constructs at step i an approximation A_i of A with desired approximate solution

$$x_i = \rho_{i-1}(A_i)b \approx A^{-1}b = x$$

where $\rho_{i-1}(\lambda)$ is associated polynomial of degree at most $i-1$.

A_i is obtained by restricting and projecting A onto the ith Krylov subspace

$$\mathcal{K}_i(A, r_0) = \text{span}\{r_0, Ar_0, \dots, A^{i-1}r_0\}$$ where $r_0 = b - Ax_0$

*for connection to infinite dimensional setting see [Málek & Strakoš, 2015]

The conjugate gradient method

Let $V_i = [v_1, \ldots, v_i]$ be a basis for the Krylov subspace $\mathcal{K}_i(A, r_0)$

In step i, we have the operator (on a subspace of small dimension):

$$A_i = V_i \underbrace{V_i^* A V_i}_{T_i} V_i^*$$

In the conjugate gradient method, projected matrix is a tridiagonal Jacobi matrix T_i

In each step i, solve the projected system

$$T_i y_i = \|r_0\| e_1$$

can be seen as a model reduction of the original system $Ax = b$

and update the approximate solution $x_i = x_0 + V_i y_i$

The conjugate gradient method

The residuals in CG are of the form

$$r_i = \rho_i^{CG}(A) r_0$$

where $\rho_i^{CG}(\lambda)$ is the CG polynomial which satisfies $\rho_0^{CG}(\lambda) = 1$ and

$$\rho_i^{CG}(\lambda) = \frac{\left(\lambda - \theta_1^{(i)}\right) \cdots \left(\lambda - \theta_i^{(i)}\right)}{(-1)^i \theta_1^{(i)} \cdots \theta_i^{(i)}}$$

where $\theta_1^{(i)}, \ldots, \theta_i^{(i)}$ are the eigenvalues of the Jacobi matrix T_i

CG polynomial is uniquely defined by the minimization problem

$$\|x - x_i\|_A = \min_{\substack{\rho(0)=1 \\ \deg(\rho) \le i}} \|\rho(A)(x - x_0)\|_A = \left\|\rho_i^{CG}(A)(x - x_0)\right\|_A$$

which is equivalent to

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

In each step i, CG picks the approximate solution from the shifted Krylov subspace $x_0 + \mathcal{K}_i(A, r_0)$ that ***minimizes the A-norm (energy norm) of the error***.

$\Rightarrow$ CG (and other Krylov subspace methods) are highly nonlinear

3.The Algorithm

- Many potential algorithms to choose from to implement a given method
- CG: 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

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

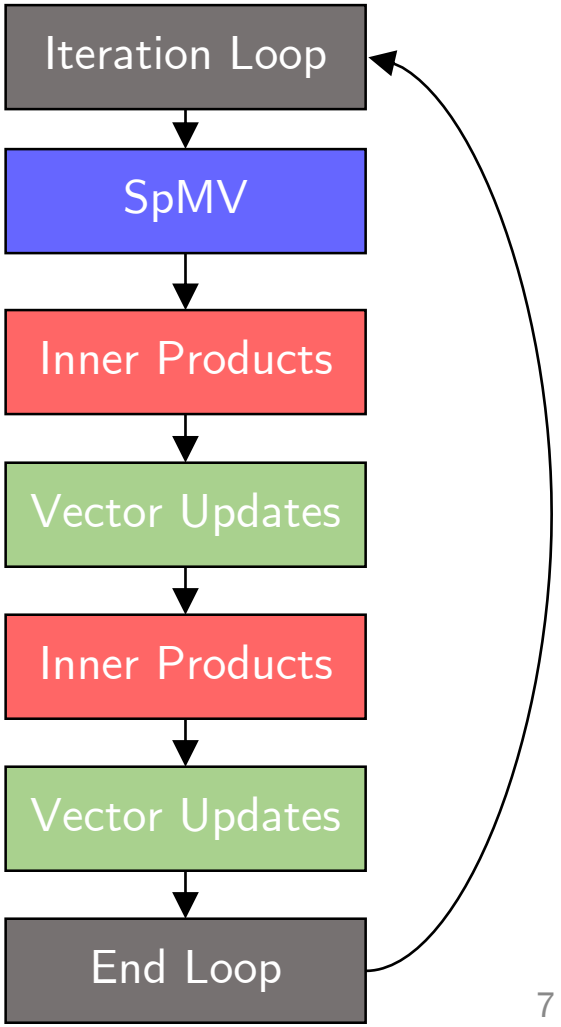
$$x_i = x_{i-1} + \alpha_{i-1} p_{i-1}$$

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

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

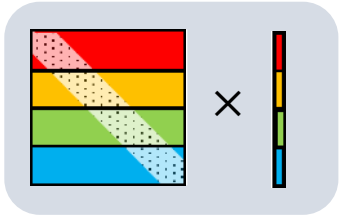
$$p_i = r_i + \beta_i p_{i-1}$$

end

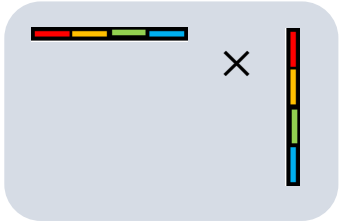


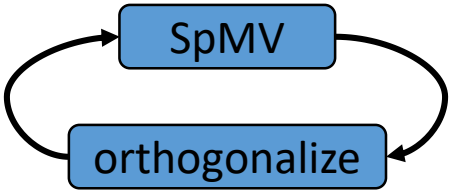
Communication in HSCG

→ Sparse matrix-vector multiplication (SpMV)
- Must communicate vector entries w/ neighboring processors (P2P communication)



→ 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!

TOP500 HPCG Benchmark, June 28, 2021

Rank	System	Rpeak (Tflops/s)	HPCG (Tflops/s)	HPCG %peak	HPL (Tflops/s)	HPL % peak
1	Supercomputer Fugaku, RIKEN, Japan	537,212	16004.50	**3.0%**	442,010	82.3%
2	Summit, ORNL, USA	200,794.9	2925.75	**1.5%**	148,600	74.0%
3	Perlmutter, LBNL, USA	89,794.5	1905.44	**2.0%**	64,590	72.0%
4	Sierra, LLNL, USA	125,712.0	1795.67	**1.4%**	94,640	75.3%
5	Selene, NVIDIA, USA	79,215.0	1622.51	**2.1%**	63,460	80.1%
6	JUWELS Booster Module, FZJ, Germany	70,980.0	1275.36	**1.8%**	44,120	62.2%

Pipelined CG [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

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

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

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

$$q_i = Aw_i$$

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

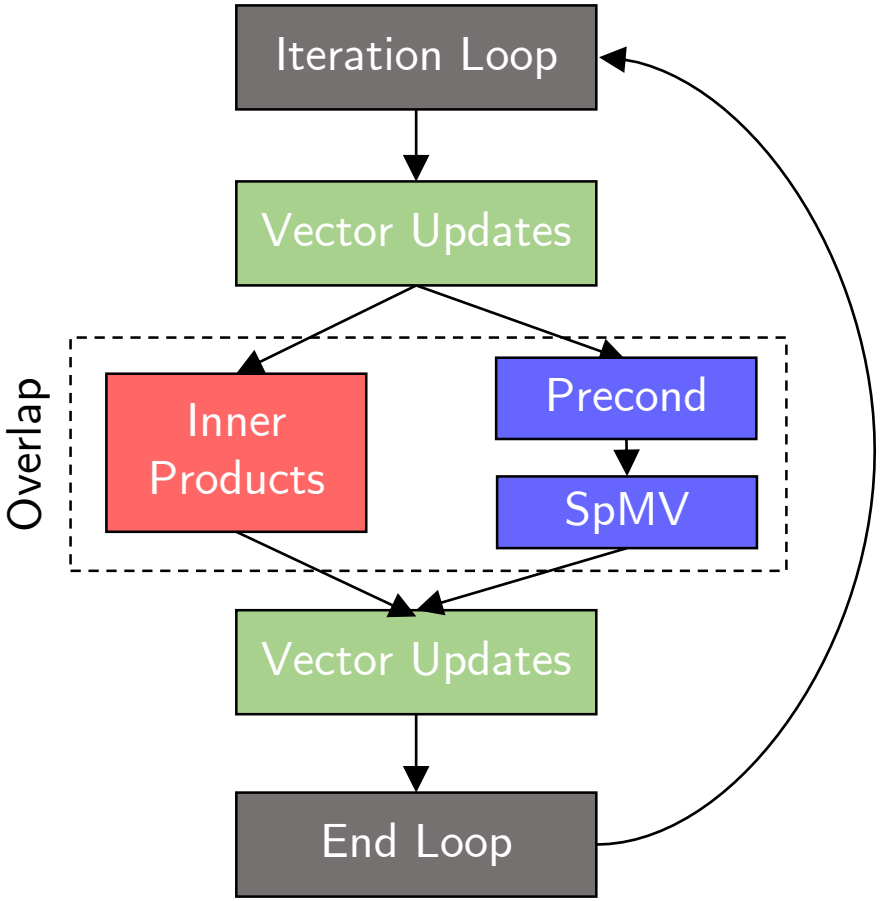
$$\alpha_i = \frac{r_i^T r_i}{w_i^T r_i - (\beta_i / \alpha_{i-1}) r_i^T r_i}$$

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

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

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

end



s-step CG e.g.,[Van Rosendale, 1983],[Chronopoulos & Gear, 1989],[Toledo,1995]

$r_0 = b - Ax_0, p_0 = r_0$

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 $j = 1:s$

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

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

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

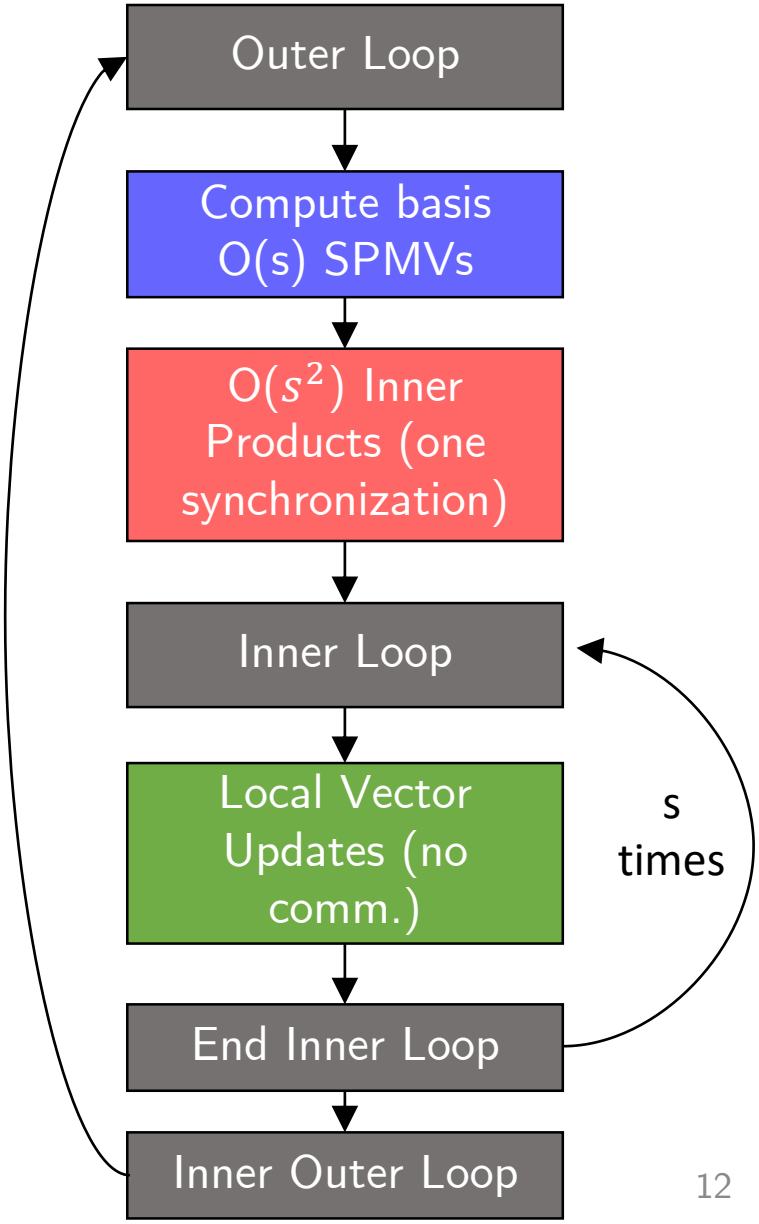
$$\beta_{sk+j} = \frac{r'^T_j\mathcal{G}_k r'_j}{r'^T_{j-1}\mathcal{G}_k r'_{j-1}}$$

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

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

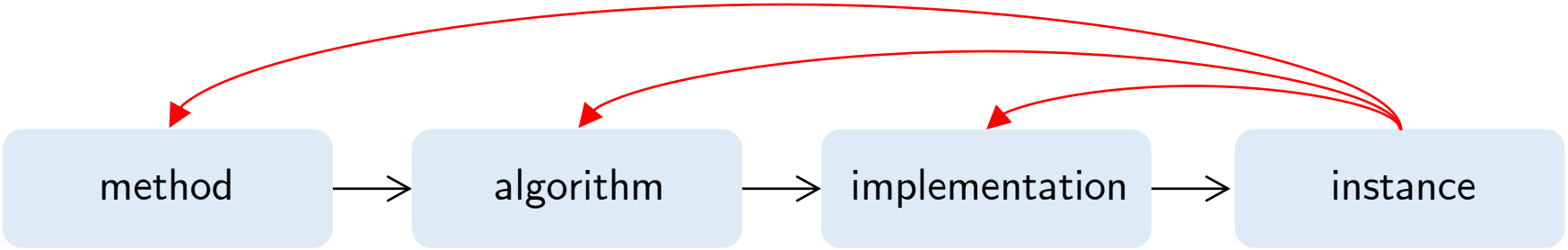


4.The implementation & 5.The instance

- Implementation: translation of the algorithm into instructions to be run on a computer
  - Programming language, parallelization, machine precision, etc.

- Instance: the particular data A, b, the particular machine

- Numerical properties of the data A, b and the hardware parameters of the particular machine ultimately determine the cost of the iterative computation
  - Selection of optimal method, algorithm, and implementation cannot be done a priori



How is cost defined?

Work on extending the Turing machine model to numerical algorithms motivated by apparent lack of formal notion of cost ...

"There is not even a formal definition of algorithm in the subject ... Thus we view numerical analysis as an eclectic subject with weak foundations."

[Blum, Cucker, Shub, Smale, 1998]

"Phrases like 'cost', 'gap', etc., used within mathematical formalism, are ambiguous. Rigorous proofs require rigorous definitions."

[Iserles, 2000]

Viewpoint

1. Cost is well-defined in terms of energy and/or runtime
   - Measurable and meaningful even in total absence of a formal Turing machine model

2. The computational transformation of the data, consisting of a problem, method, algorithm, implementation, and instance is the *key concept*.
   - Focus on the *algorithm* misses the bigger picture

Important! : for iterative computations:

cost = (cost per iteration) x (number of iterations)

The cost per iteration

- Cost per iteration depends on particular algorithm, implementation, and data/machine
- Single iteration broken down into computational kernels (SpMV, inner products, etc.)
- The cost of these kernels can be modeled in terms of computation (flops) and communication (data movement)
- Simplified cost model: If a processor performs F flops, and sends/receives S messages containing a total of W words, we then model its worst-case cost as

$$\text{cost} = \gamma F + \beta W + \alpha S$$

where γ is cost of a flop on local data, β is cost per word of data moved (inverse bandwidth), α is cost per message (latency)

  - Machine structure: CPUs, GPUs, half-precision tensor cores, accelerators, etc.

- Rates of improvement: $\gamma \gg \beta \gg \alpha$
  - Communication more expensive than computation, trend will continue

Describing CG Convergence

Infinite precision: CG convergence rate depends strongly on the distribution of eigenvalues

$$\frac{\|x - x_i\|_A}{\|x - x_0\|_A} \leq \min_{\substack{\rho(0)=1 \\ \deg(\rho)\leq i}} \max_{1\leq j\leq N} |\rho(\lambda_j)|$$

Let d be the number of distinct eigenvalues of A. For $i = 1, \ldots, d-1$, there exist $i+1$ distinct eigenvalues of A, $\hat{\lambda}_1, \ldots \hat{\lambda}_{i+1}$, such that

$$\min_{\substack{\rho(0)=1 \\ \deg(\rho)\leq i}} \max_{1\leq j\leq N} |\rho(\hat{\lambda}_j)| = \left(\sum_{k=1}^{i+1} \prod_{\substack{j=1 \\ j\neq k}}^{i+1} \frac{|\hat{\lambda}_j|}{|\hat{\lambda}_j - \hat{\lambda}_k|} \right)^{-1}$$

[Greenbaum, 1979]

Frequent (over)simplification: estimate by replacing set of eigenvalues of A by continuous interval $[\lambda_1, \lambda_N]$ and use scaled and shifted Chebyshev polynomials:

$$\frac{\|x - x_i\|_A}{\|x - x_0\|_A} \leq 2 \left(\frac{\sqrt{\kappa(A)} - 1}{\sqrt{\kappa(A)} + 1} \right)^i \qquad \kappa(A) = \lambda_N / \lambda_1$$

Chebyshev polynomial-based bound holds for ***any*** distribution of eigenvalues between λ_1 and λ_N and ***any*** distribution of the components of the initial residuals in the individual invariant subspaces!

⇒Linearization of highly nonlinear phenomena

Example

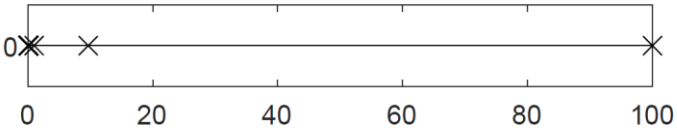
Example: diagonal matrix with $N = 25$,

$$\lambda_1 = 0.1,\ \lambda_N = 100$$

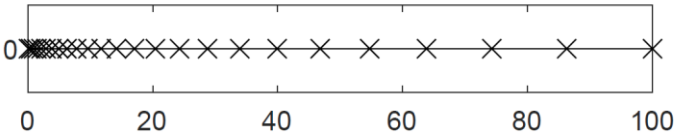
$$\lambda_j = \lambda_1 + \frac{j-1}{N-1}(\lambda_N - \lambda_1)\gamma^{N-j}$$

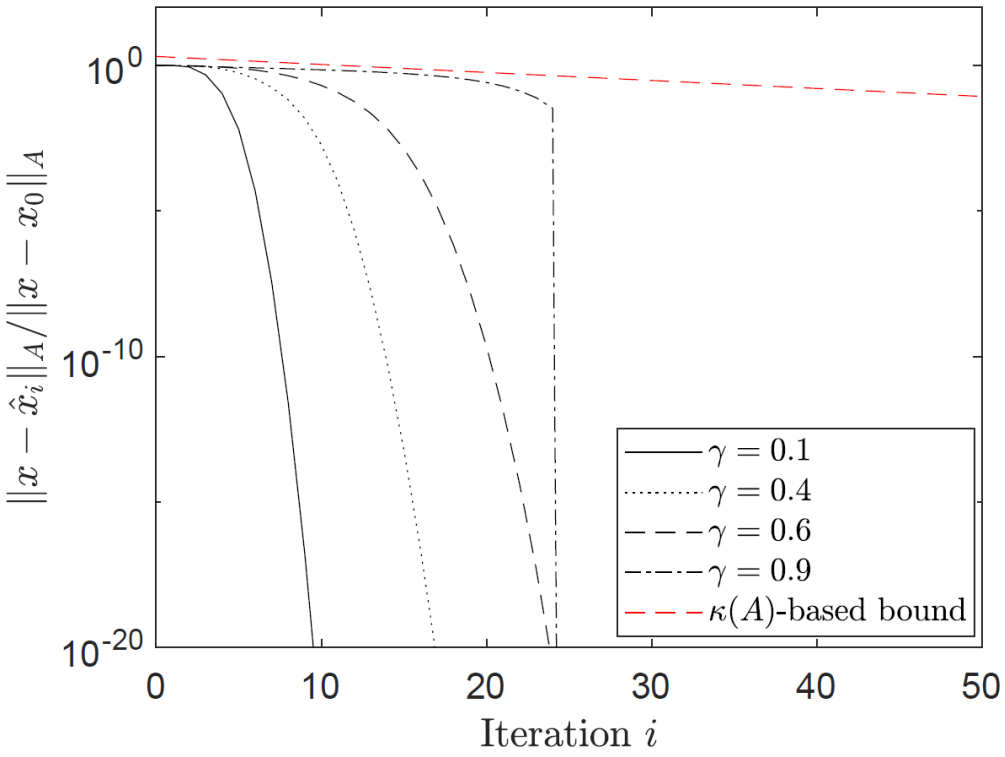
RHS: $b_i = 1/5$

$\gamma = 0.1$:



$\gamma = 0.9$:

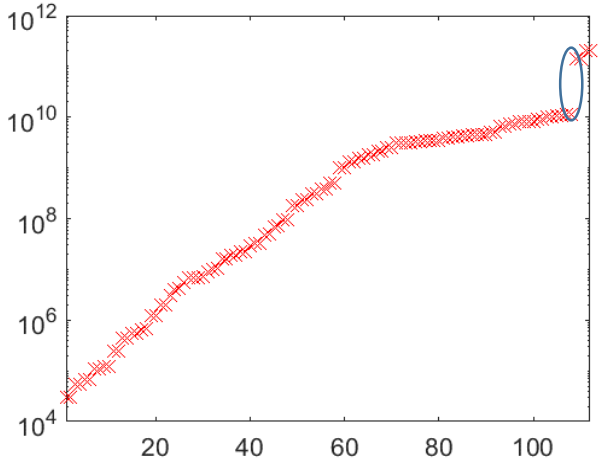




The Distribution of Ill-Conditioning

Matrix: bcsstk03 from SuiteSparse

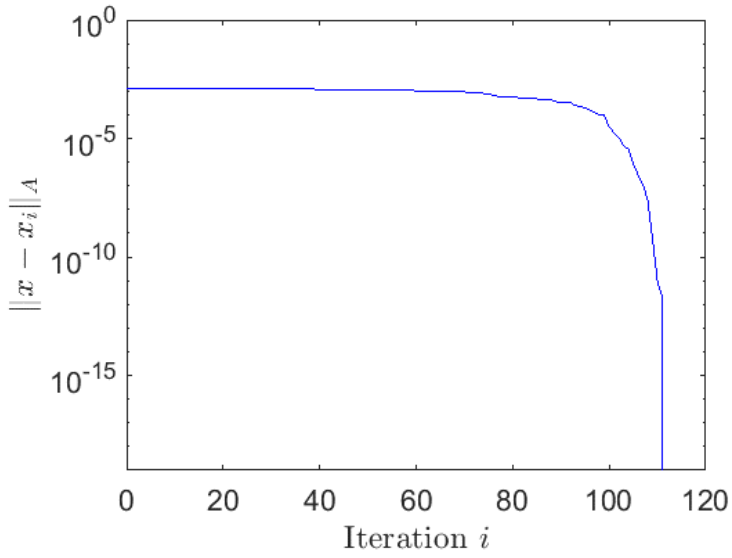
$N = 112, \kappa(A) = 6.8 \times 10^6$

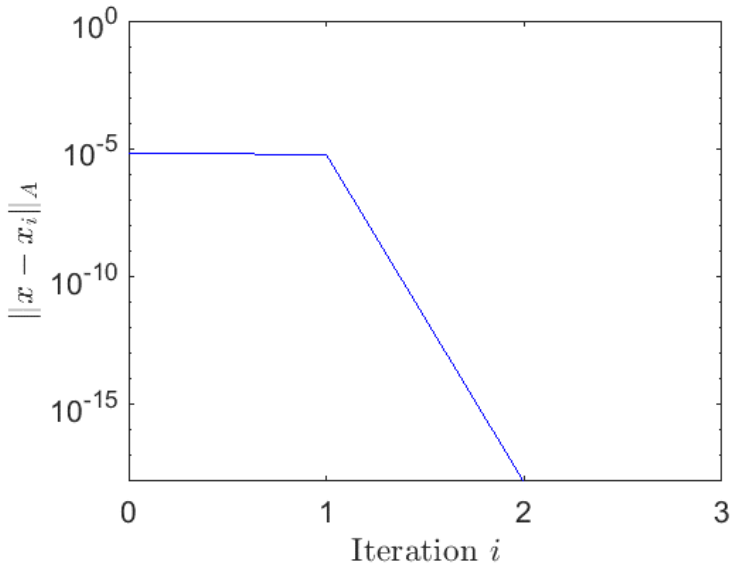


$A = VDV^T$, CG in infinite precision

$$b_1 = V\begin{bmatrix}1\\ \vdots \\ 1\end{bmatrix};\ b_1 = \frac{b_1}{\|b_1\|_2}$$

$$b_2 = V[0,\dots,0,1,1,0,0,0]^T;\ b_2 = \frac{b_2}{\|b_2\|_2}$$





Describing CG Convergence

[Publication, 2015]:

> *"Soon after the introduction of $\kappa(A)$ for error analysis, Hestenes and Stiefel showed that this quantity also played a role in complexity analysis. More precisely, they showed that the* ***number of iterations of the conjugate gradient method*** *(assuming infinite precision) needed to ensure that the current approximation to the solution of a linear system attained a given accuracy* ***is proportional to*** $\sqrt{\kappa(A)}$*."*

- The Chebyshev polynomial-based bound *does not* appear anywhere in the original 1952 paper of Hestenes and Stiefel (1952)
- First appearance in the literature in [Daniel, 1967], although he did not identify it with the CG "convergence rate":

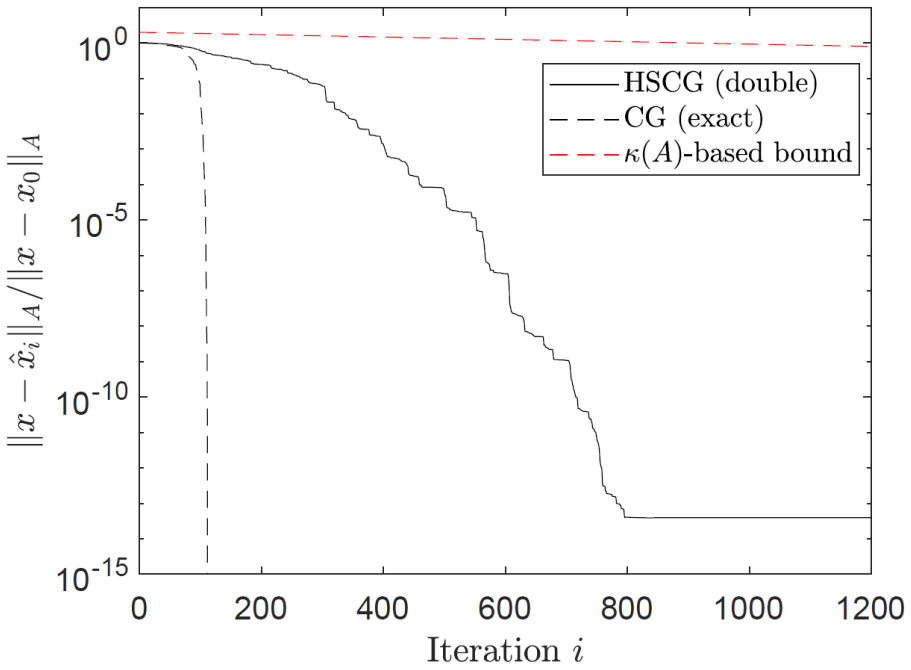
  "Assuming only that the spectrum of the matrix A lies inside the interval $[\lambda_1, \lambda_N]$*, we can do no better than [the* $\kappa(A)$*-based bound]"*

The effects of rounding errors

Well-known that roundoff error has two effects:

1. Delay of convergence
   - No longer have Krylov subspace
   - Can lose numerical rank deficiency
   - Residuals no longer orthogonal - Minimization of $\|x - x_i\|_A$?

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



A: bcsstk03 from SuiteSparse,
b: equal components in the eigenbasis of A, $\|b\| = 1$
$N = 112, \kappa(A) \approx 6.8e6$

Much work on these results for classical CG algorithms; See [Meurant & Strakoš, 2006] for a thorough summary of early developments

Algorithms designed for HPC

- Computational complexity is a poor measure of runtime/energy cost
  - Cost depends heavily on communication complexity
- Communication-avoiding algorithms
  - Prove lower bounds on amount of data moved, number of messages
  - Find algorithms that meet lower bounds
  - Many successes in direct numerical linear algebra
- Extension to iterative numerical linear algebra is tricky
  - Idea: Modify algorithms to reduce synchronization cost over fixed number of steps
  - Long history of work on synchronization-reducing algorithms mathematically equivalent to HSCG (e.g., pipelined CG, s-step CG)

But how many steps are required to converge to prescribed accuracy?

Can we even still converge to prescribed accuracy?

- Makes little sense to claim these algorithms to be "high-performance algorithms" or "exascale algorithms" without answering these questions
  - Must understand behavior in finite precision and potential **amplification** of rounding errors

Maximum attainable accuracy

- Accuracy $\|x - \hat{x}_i\|$ generally not computable, *but* $x - \hat{x}_i = A^{-1}(b - A\hat{x}_i)$
- Size of the true residual, $\|b - A\hat{x}_i\|$, used as computable measure of accuracy
- 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 $\|b - A\hat{x}_i - \hat{r}_i\|$

- 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

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

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

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

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

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

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

Attainable Accuracy of Pipelined CG

[Cools, et al., 2018]

Computed explicitly: $q_i \equiv Aw_i$

Pipelined CG uses 3 auxiliary recurrences:

$$s_i \equiv Ap_i, \qquad w_i = Ar_i, \qquad z_i \equiv Aw_i$$

$$\hat{p}_i = \hat{r}_i + \hat{\beta}_i \hat{p}_{i-1} + \delta_i^p \qquad \hat{x}_i = \hat{x}_{i-1} + \hat{\alpha}_{i-1} \hat{p}_{i-1} + \delta_i^x$$

$$\hat{s}_i = \hat{w}_i + \hat{\beta}_i \hat{s}_{i-1} + \delta_i^s \qquad \hat{r}_i = \hat{r}_{i-1} - \hat{\alpha}_{i-1} \hat{s}_{i-1} + \delta_i^r$$

$$\hat{z}_i = \hat{q}_i + \hat{\beta}_i \hat{z}_{i-1} + \delta_i^z \qquad \hat{w}_i = \hat{w}_{i-1} - \hat{\alpha}_{i-1} \hat{z}_{i-1} + \delta_i^w$$

Attainable Accuracy of Pipelined CG

$$f_i = f_0 - \sum_{j=0}^{i} \hat{\alpha}_j g_j - \sum_{j=0}^{i} (A\delta_j^x + \delta_j^r) \qquad (f_i \equiv b - A\hat{x}_i - \hat{r}_i)$$

$$g_j = \left(\prod_{k=1}^{j} \hat{\beta}_k\right) g_0 + \sum_{k=1}^{j} \left(\prod_{\ell=k+1}^{j} \hat{\beta}_\ell\right) ((A\delta_k^p - \delta_k^s) + h_k)$$

$$h_k = h_0 + \sum_{\ell=0}^{k-1} ((A\delta_\ell^r - \delta_\ell^w) - \hat{\alpha}_\ell c_\ell)$$

$$c_\ell = \left(\prod_{m=1}^{\ell} \hat{\beta}_m\right) c_0 + \sum_{m=1}^{\ell} \left(\prod_{n=m+1}^{\ell} \hat{\beta}_n\right) (A\delta_m^q - \delta_m^z)$$

Attainable Accuracy of Pipelined CG

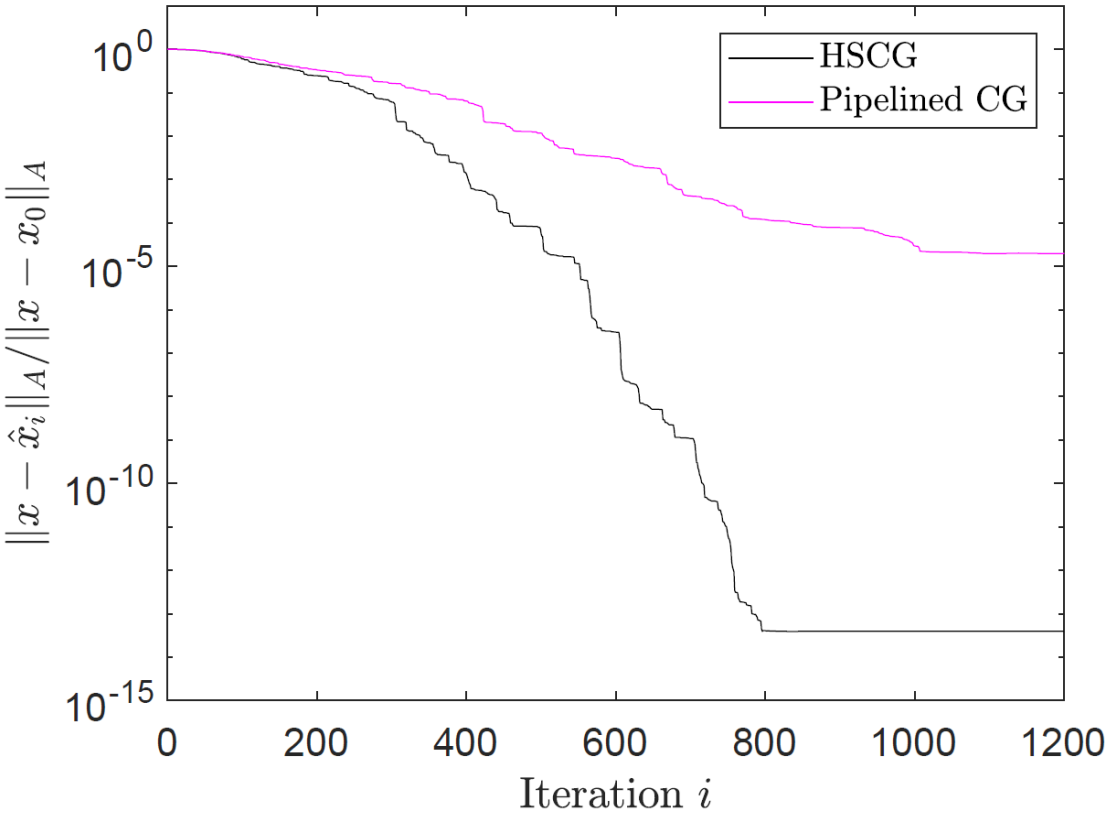
$$f_i = f_0 - \sum_{j=0}^{i} \hat{\alpha}_j g_j - \sum_{j=0}^{i} (A\delta_j^x + \delta_j^r)$$

$$g_j = \left(\prod_{k=1}^{j} \hat{\beta}_k \right) g_0 + \sum_{k=1}^{j} \left(\prod_{\ell=k+1}^{j} \hat{\beta}_\ell \right) (\quad + h_k)$$

$$h_k = h_0 + \sum_{\ell=0}^{k-1} (\quad - \hat{\alpha}_\ell c_\ell)$$

$$c_\ell = \left(\prod_{m=1}^{\ell} \hat{\beta}_m \right) c_0 + \sum_{m=1}^{\ell} \left(\prod_{n=m+1}^{\ell} \hat{\beta}_n \right) (\quad)$$

Local rounding errors all potentially amplified!



A: bcsstk03 from SuiteSparse,
b: equal components in the eigenbasis of $A, \|b\| = 1$
$N = 112, \kappa(A) \approx 6.8e6$

Attainable accuracy of s-step CG

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

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

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

$$\|f_{sk+j}\| \leq \|f_{sk}\| + \varepsilon \boldsymbol{\Gamma_k} \sum_{\ell=1}^{j} (1+N)\|A\|\|\hat{x}_{sk+\ell}\| + \|\hat{r}_{sk+\ell}\|$$

where $\Gamma_k = c \cdot \|\hat{\mathcal{Y}}_k^+\| \, \|\hat{\mathcal{Y}}_k\|$, c is a low-degree polynomial in s

Local rounding errors amplified; amplification is "local" within s-steps

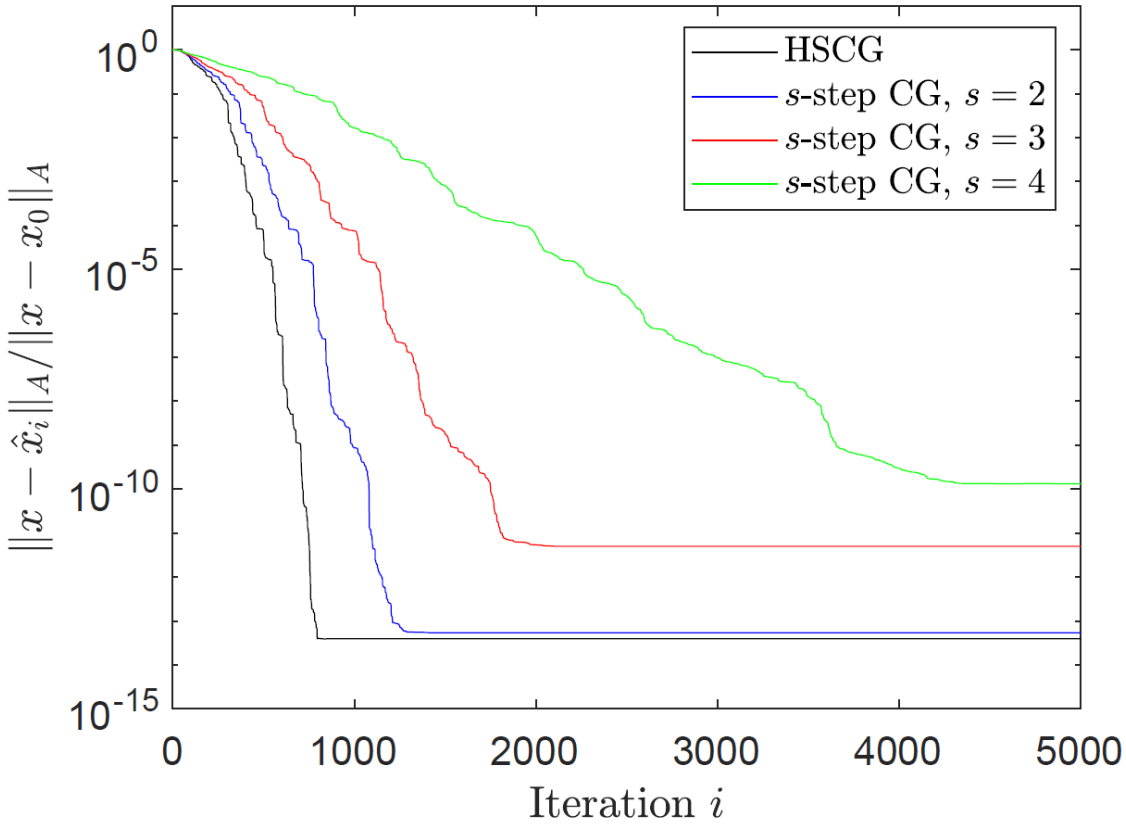
$$\|f_{sk+j}\| \leq \|f_0\| + \varepsilon \bar{\boldsymbol{\Gamma}}_{\boldsymbol{k}} \sum_{m=1}^{sk+j} (1+N)\|A\|\|\hat{x}_m\| + \|\hat{r}_m\|$$

where $\bar{\Gamma}_k = \max_{m \leq k} \Gamma_m$

[see C., 2015]

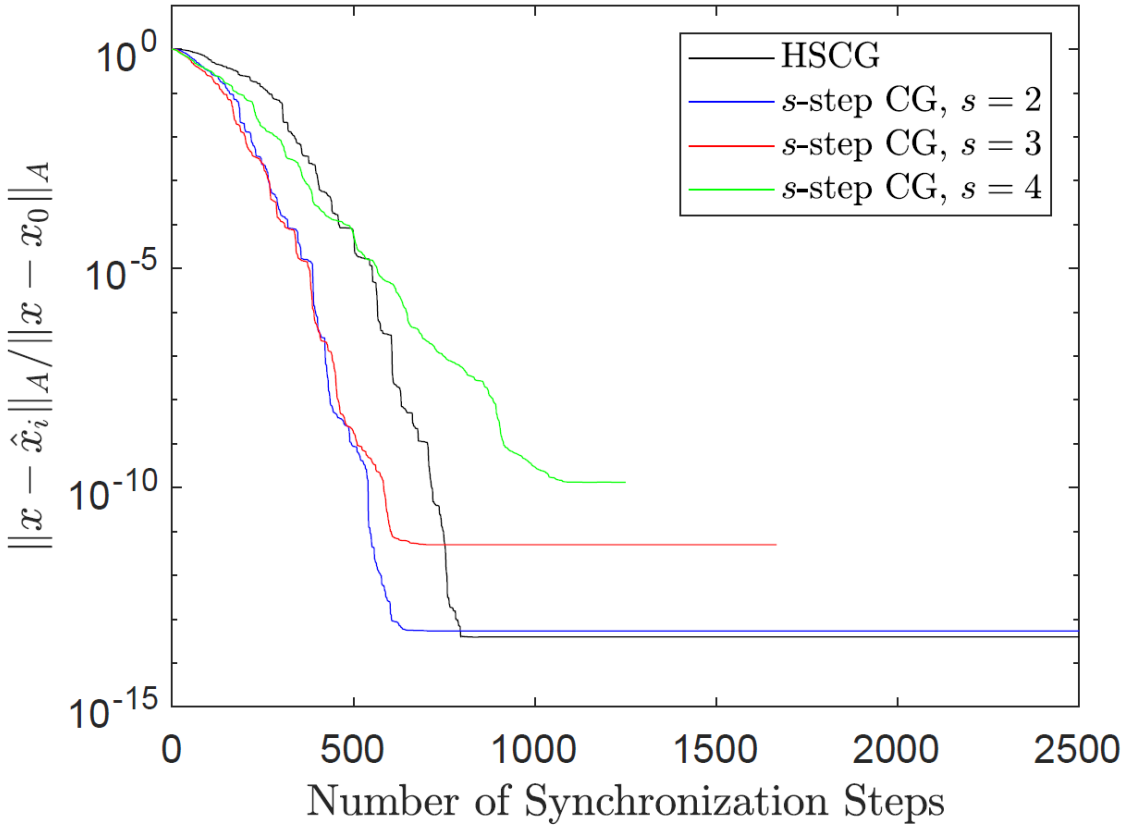
Accuracy of s-step CG in prec. ε ↔ Accuracy of HSCG in prec. $\varepsilon\bar{\Gamma}_k$

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



A: bcsstk03 from SuiteSparse,
b: equal components in the eigenbasis of A, $\|b\| = 1$
$N = 112, \kappa(A) \approx 6.8e6$

Even assuming cost per iteration decreases by factor of s, already at $s = 4$ we are worse than HSCG in terms of number of synchronizations!

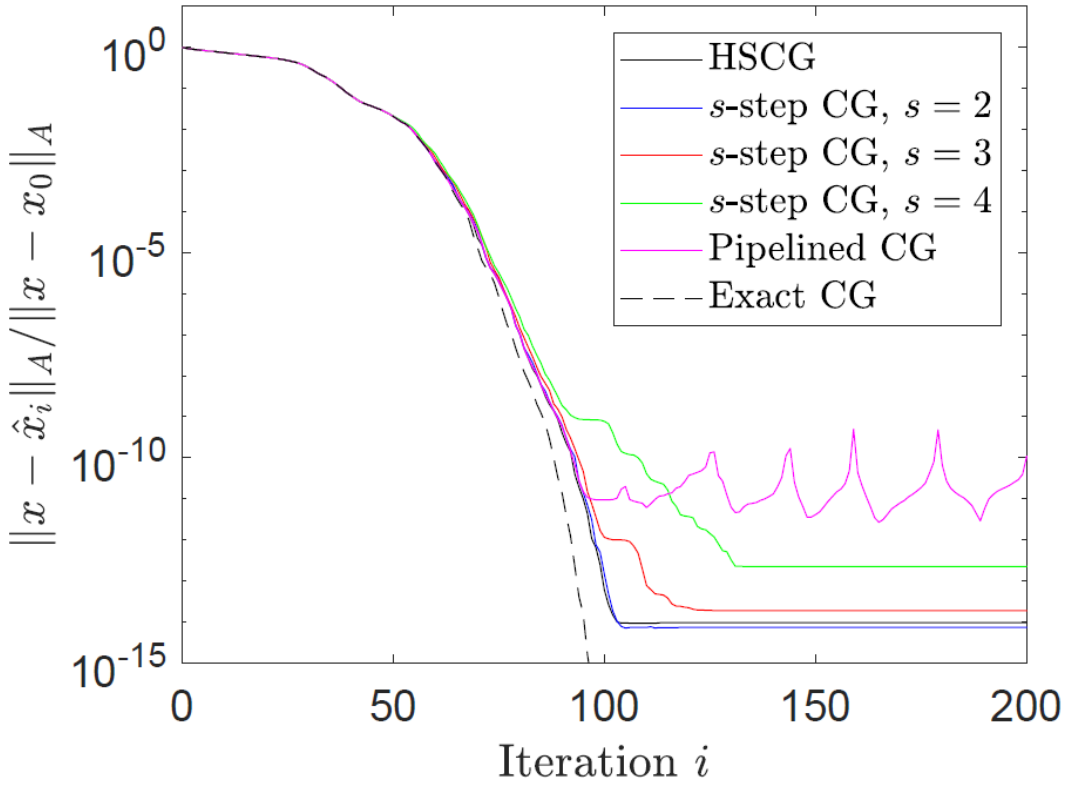


A: bcsstk03 from SuiteSparse,
b: equal components in the eigenbasis of $A, \|b\| = 1$
$N = 112, \kappa(A) \approx 6.8e6$

A different problem...

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

If application only requires $\frac{\|x-\hat{x}_i\|_A}{\|x-x_0\|_A} \approx 10^{-6}$, any of these algorithms will work!



Towards understanding convergence delay

- CG method = matrix formulation of Gauss-Christoffel quadrature (see [Liesen & Strakoš, 2013])
- Coefficients α and β (related to entries of T_i) determine distribution functions $\omega^{(i)}(\lambda)$ which approximate distribution function $\omega(\lambda)$ determined by inputs A, b, x_0 in terms of the ith Gauss-Christoffel quadrature
- A-norm of CG error for $f(\lambda) = \lambda^{-1}$ given as scaled quadrature error

$$\int \lambda^{-1} d\omega(\lambda) = \sum_{\ell=1}^{i} \omega_\ell^{(i)} \left\{\theta_\ell^{(i)}\right\}^{-1} + \frac{\|x - x_i\|_A^2}{\|r_0\|^2} \qquad = \left(\frac{\|x - x_0\|_A^2}{\|r_0\|^2}\right)$$

- For particular CG implementation, can the computed $\widehat{\omega}^{(i)}(\lambda)$ be associated with some distribution function $\widehat{\omega}(\lambda)$ related to the distribution function $\omega(\lambda)$, i.e.,

$$\int \lambda^{-1} d\omega(\lambda) \approx \int \lambda^{-1} d\widehat{\omega}(\lambda) = \sum_{\ell=1}^{i} \widehat{\omega}_\ell^{(i)} \left\{\hat{\theta}_\ell^{(i)}\right\}^{-1} + \frac{\|x - \hat{x}_i\|_A^2}{\|r_0\|^2} + F_i$$

  where F_i is small relative to error term?
- For classical CG, yes; proved by Greenbaum [1989]
- For pipelined CG and s-step CG, THOROUGH ANALYSIS NEEDED!

Designing preconditioners

- Approach: design preconditioner M such that preconditioned linear system $M^{-1}Ax = M^{-1}b$ converges in few iterations
- Frequent assertion:

# of clusters of eigenvalues = # iterations for Krylov subspace method to converge

[Greenbaum, 1989]: finite precision HSCG on matrix A with simple eigenvalues behaves like exact CG on larger matrix $\tilde{A}$ whose eigenvalues are in tight clusters around the eigenvalues of A

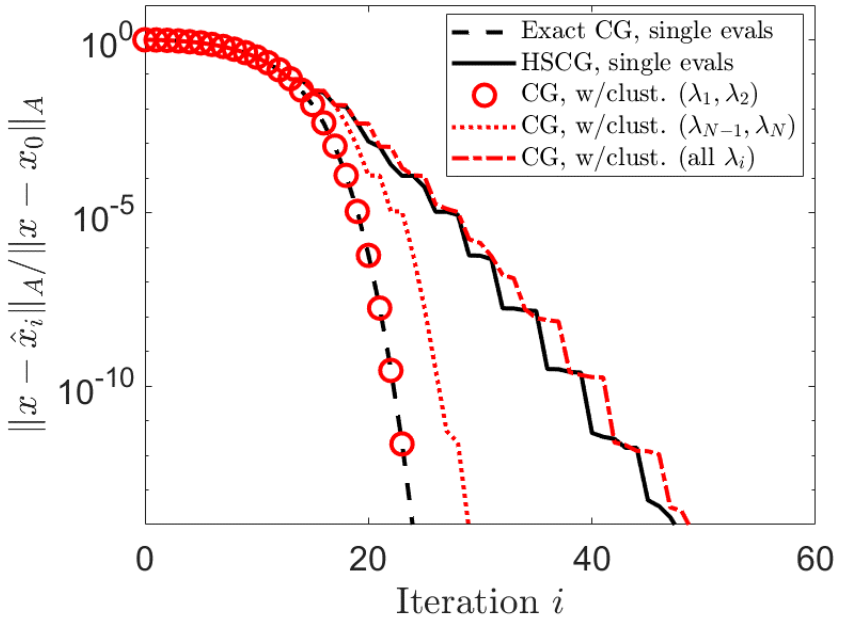
If clustering argument were true, Greenbaum's results imply that behavior of HSCG in finite precision is similar to infinite precision CG!

Example: diagonal matrix with

$$N = 25, \lambda_1 = 0.1, \lambda_N = 100, \gamma = 0.65$$

$$\lambda_j = \lambda_1 + \frac{j-1}{N-1}(\lambda_N - \lambda_1)\gamma^{N-j}$$

RHS: $b_i = 1/5$



Designing preconditioners

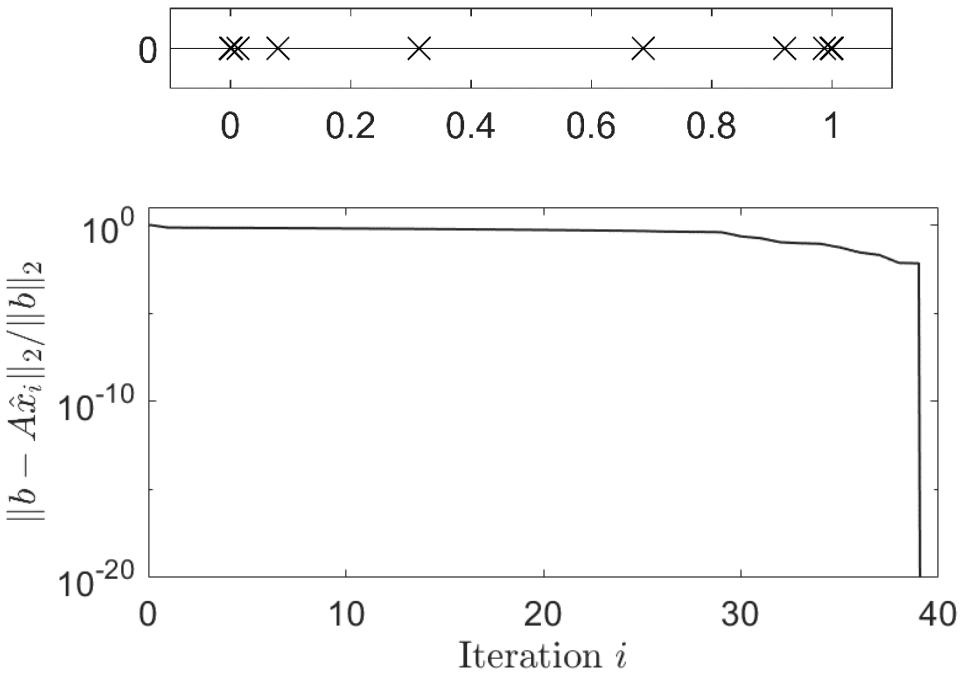
- Approach: design preconditioner M such that preconditioned linear system $M^{-1}Ax = M^{-1}b$ converges in few iterations
- Frequent assertion:

# of clusters of eigenvalues = # iterations for Krylov subspace method to converge

Argument also fails for GMRES (see [Greenbaum & Strakoš, 1994], [Greenbaum, Pták, Strakoš, 1996], [Arioli, Pták, Strakoš, 1998])

Example [Liesen & Tichý, 2004]:
exact GMRES on
$A =$ `gallery('prolate', 40)`
RHS: b: equal components in the eigenbasis of A, $\|b\| = 1$

Note: the matrix A is normal!



Designing preconditioners

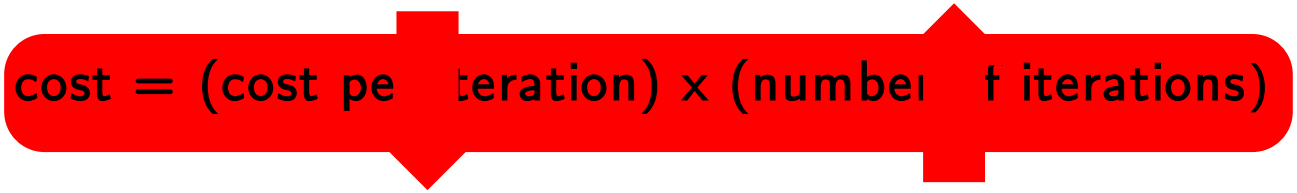
Other considerations:

- If A is ill conditioned, M is likely to be ill conditioned

$$\hat{z}_i = (M^{-1}A + \Delta)\,\hat{r}_i \qquad \|\Delta\|_2 \leq O(\varepsilon)N^{3/2}\big(\kappa(A) + \kappa(M)\big)\|M^{-1}A\|_F$$

  - Even if preconditioned system is well conditioned, application can still introduce significant roundoff error
    - Potentially diminished attainable accuracy

- Must consider tradeoff between cost per iteration and convergence rate
  - Fewer iterations, but potentially much more expensive



The place of numerical analysis

> *"Numerical analysis lies at the meeting point of pure mathematics, computer science, and application areas. It often attracts some degree of hostility from all three."*
>
> [Baxter and Iserles, *On the foundations of computational mathematics*, Handbook of numerical analysis, 11 (2003), pp.3-34]

- Various approaches to describe and predict the cost of iterative computations
- All perspectives are valuable and can lead to interesting insights; none alone gives complete description
- For bigger picture, must considered all aspects of a computation together
  → holistic approach needed

- Confluence of data science/informatics and computational science
  - Motivating changes in hardware, new algorithms, new approaches

References

M. Arioli, V. Pták, and Z. Strakoš. *Krylov sequences of maximal length and convergence of GMRES*. BIT Numer. Math., 38(4):636–643, 1998.

E. Carson. *Communication-avoiding Krylov subspace methods in theory and practice.* PhD thesis, U.C. Berkeley, 2015.

E. Carson, M. Rozložník, Z. Strakoš, P. Tichý, and M. Tůma. *The numerical stability analysis of pipelined conjugate gradient methods: Historical context and methodology*. SIAM J. Sci. Comput., 40(5):A3549–A3580, 2018.

E. Carson and Z. Strakoš. On the Cost of Iterative Computations, Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences 378(2166), 2020

T. Gergelits, K.-A. Mardal, B. Nielsen, and Z. Strakoš. *Laplacian preconditioning of elliptic PDEs: Localization of the eigenvalues of the discretized operator.* SIAM J. Numer. Anal. (to appear), 2019.

G. H. Golub and G. Meurant. *Matrices, Moments, and Quadrature with Applications*. Princeton Univ. Press, USA 2010.

A. Greenbaum. *Behavior of slightly perturbed Lanczos and conjugate-gradient recurrences*. Lin. Alg. Appl., 113:7–63, 1989.

A. Greenbaum, V. Pták, and Z. Strakoš. *Any nonincreasing convergence curve is possible for GMRES*. SIAM J. Matrix Anal. Appl., 17(3):465–469, 1996.

A. Greenbaum and Z. Strakoš. *Predicting the behavior of finite precision Lanczos and conjugate gradient computations.* SIAM J. Matrix Anal. Appl., 13(1):121–137, 1992.

A. Greenbaum and Z. Strakoš. *Matrices that generate the same Krylov residual spaces*. Recent advances in iterative methods. Springer, New York, NY, 1994.

M. R. Hestenes and E. Stiefel. *Methods of conjugate gradients for solving linear systems*. J.Research Nat. Bur. Standards, 49:409–436, 1952.

J. Liesen and Z. Strakoš. *Krylov subspace methods: Principles and analysis.* Numerical Mathematics and Scientific Computation. Oxford Univ. Press, 2013.

J. Málek and Z. Strakoš. *Preconditioning and the conjugate gradient method in the context of solving PDEs*. SIAM, 2015.

G. Meurant and Z. Strakoš. *The Lanczos and conjugate gradient algorithms in finite precision arithmetic*. Acta Numerica, 15:471–542, 2006.

Thank you!

carson@karlin.mff.cuni.cz

www.karlin.mff.cuni.cz/~carson/