Why Worry about Rounding Errors in Hermitian Krylov Subspace Methods?

Anne Greenbaum[1] Hexuan Liu Tyler Chen

University of Washington

Feb. 12, 2019

[1]Boeing Professor of Applied Mathematics

The Lanczos and Conjugate Gradient Algorithms

- CG is a widely used iterative method for solving large Hermitian positive definite linear systems $Ax = b$.
- It is equivalent to the Lanczos algorithm, which is often used to compute eigenvalues/vectors or to compute $f(A)b$ (e.g., $f(A) = A^{1/2}$ or $\exp(A)$).
- Many different CG/Lanczos implementations have been proposed to, for example, make better use of parallelism. All of these are *mathematically* equivalent.

CG in Finite Precision Arithmetic

When implemented in finite precision arithmetic, CG does **not** behave the way exact arithmetic theory predicts! A priori, there is no reason to expect it to converge at all, once agreement with exact arithmetic is lost.

CG in Finite Precision Arithmetic

When implemented in finite precision arithmetic, CG does **not** behave the way exact arithmetic theory predicts! A priori, there is no reason to expect it to converge at all, once agreement with exact arithmetic is lost.

Two main questions:

1. What level of accuracy can ultimately be attained?
2. What is the rate of convergence before the ultimately attainable accuracy is achieved?

CG in Finite Precision Arithmetic

When implemented in finite precision arithmetic, CG does **not** behave the way exact arithmetic theory predicts! A priori, there is no reason to expect it to converge at all, once agreement with exact arithmetic is lost.

Two main questions:

1. What level of accuracy can ultimately be attained?
2. What is the rate of convergence before the ultimately attainable accuracy is achieved?

Lots of theory dating back to the 1980's about how the method behaves *under certain assumptions* that *might* be expected to hold in finite precision, depending on the implementation.

Question 1: Level of Accuracy

For standard CG implementations:

$$x_k = x_{k-1} + a_{k-1}p_{k-1} + \delta_{x_k}, \quad (1)$$

$$r_k = r_{k-1} - a_{k-1}Ap_{k-1} + \delta_{r_k}. \quad (2)$$

Typically, $r_k \to 0$ (although this has not been proved!), so look at how r_k differs from $b - Ax_k$. From (1),

$$b - Ax_k = b - Ax_{k-1} - a_{k-1}Ap_{k-1} - A\delta_{x_k},$$

so if $d_k := (b - Ax_k) - r_k$, then

$$d_k = d_{k-1} - A\delta_{x_k} - \delta_{r_k} = \cdots = d_0 - \sum_{j=1}^{k}(A\delta_{x_j} + \delta_{r_j}).$$

If ϵ is the machine precision, expect eventually to have

$$\frac{\|b - Ax_k\|}{\|A\|\,\|x\|} \lesssim_\epsilon k\,\frac{\max_j \|x_j\|}{\|x\|}.$$

Question 1, Cont.

But not all implementations satisfy

$$\begin{aligned} x_k &= x_{k-1} + a_{k-1}p_{k-1} + \delta_{x_k}, \\ r_k &= r_{k-1} - a_{k-1}Ap_{k-1} + \delta_{r_k}, \end{aligned}$$

with $r_k \to 0$.

Question 1, Cont.

But not all implementations satisfy

$$\begin{aligned} x_k &= x_{k-1} + a_{k-1}p_{k-1} + \delta_{x_k}, \\ r_k &= r_{k-1} - a_{k-1}Ap_{k-1} + \delta_{r_k}, \end{aligned}$$

with $r_k \to 0$.

For example, 3-term recurrence:

$$x_k = \alpha_{k-1}r_{k-1} + x_{k-1} + \frac{\alpha_{k-1}\beta_{k-1}}{\alpha_{k-2}}(x_{k-1} - x_{k-2}).$$

Gutknecht and Strakoš (2000) analyzed the attainable accuracy with this implementation.

Question 2: Rate of Convergence

Paige (1971, 1980) showed that (what is now) a standard implementation of the Lanczos algorithm satisfied 3 properties:

$$AQ_J = Q_J T_J + \beta_J q_{J+1} e_J^T + F_J.$$

Question 2: Rate of Convergence

Paige (1971, 1980) showed that (what is now) a standard implementation of the Lanczos algorithm satisfied 3 properties:

$$AQ_J = Q_J T_J + \beta_J q_{J+1} e_J^T + F_J.$$

1. $\max_{j=1,\ldots,J} \|f_j\| \leq \epsilon \|A\|$,
2. $\|q_j\| \in [1-\epsilon, 1+\epsilon]$,
3. $|\langle \beta_j q_{j+1}, q_j \rangle| \leq \epsilon \|A\|$,

where ϵ is a modest multiple of the machine precision. He used these properties to prove things about eigenvalue/vector convergence.

Question 2, Cont.

G. (1989) showed that *if* these properties are satisfied by a CG implementation, then:

Question 2, Cont.

G. (1989) showed that *if* these properties are satisfied by a CG implementation, then:
(Until r_k and $b - Ax_k$ start to differ significantly): The A-norm of the error in the CG computation behaves like the $\hat{A}$-norm of the error in exact CG applied to a larger matrix $\hat{A}$ with eigenvalues distributed throughout tiny intervals about the eigenvalues of A, the size of the intervals being a function of the machine precision.

Question 2, Cont.

G. (1989) showed that *if* these properties are satisfied by a CG implementation, then:
(Until r_k and $b - Ax_k$ start to differ significantly): The A-norm of the error in the CG computation behaves like the $\hat{A}$-norm of the error in exact CG applied to a larger matrix $\hat{A}$ with eigenvalues distributed throughout tiny intervals about the eigenvalues of A, the size of the intervals being a function of the machine precision.

As far as I know, no one has actually *proved* that any particular CG implementation satisfies these properties, but standard implementations appear to.

Implications

$$\frac{\|e_k\|_A}{\|e_0\|_A} \leq 2\left(\frac{\sqrt{\hat{\kappa}}-1}{\sqrt{\hat{\kappa}}+1}\right)^k, \quad \hat{\kappa} \approx \frac{\lambda_{max}(A)}{\lambda_{min}(A)},$$

Implications

$$\frac{\|e_k\|_A}{\|e_0\|_A} \leq 2\left(\frac{\sqrt{\hat{\kappa}}-1}{\sqrt{\hat{\kappa}}+1}\right)^k, \quad \hat{\kappa} \approx \frac{\lambda_{max}(A)}{\lambda_{min}(A)},$$

$$\frac{\|e_k\|_A}{\|e_0\|_A} \leq \min_{p_k(0)=1} \max_{z \in \cup_{i=1}^{n} [\lambda_i - \delta, \lambda_i + \delta]} |p_k(z)|.$$

Implications

$$\frac{\|e_k\|_A}{\|e_0\|_A} \leq 2\left(\frac{\sqrt{\hat{\kappa}}-1}{\sqrt{\hat{\kappa}}+1}\right)^k, \quad \hat{\kappa} \approx \frac{\lambda_{max}(A)}{\lambda_{min}(A)},$$

$$\frac{\|e_k\|_A}{\|e_0\|_A} \leq \min_{p_k(0)=1} \max_{z \in \cup_{i=1}^{n} [\lambda_i - \delta, \lambda_i + \delta]} |p_k(z)|.$$

Proven bounds on interval size δ were not as small as one might hope, but a procedure was given to construct such a matrix $\hat{A}$.

Hestenes and Stiefel CG

HSCG:
Given x_0, compute $r_0 = b - Ax_0$, and set $p_0 = r_0$.
Compute $\nu_0 = \langle r_0, r_0 \rangle$.
For $k = 1, 2, \ldots,$
  Compute Ap_{k-1}.
  Compute $\eta_{k-1} = \langle p_{k-1}, Ap_{k-1} \rangle$, $a_{k-1} = \nu_{k-1}/\eta_{k-1}$.
  Set $x_k = x_{k-1} + a_{k-1}p_{k-1}$,
    $r_k = r_{k-1} - a_{k-1}Ap_{k-1}$.
  Compute $\nu_k = \langle r_k, r_k \rangle$, $b_k = \nu_k/\nu_{k-1}$.
  Set $p_k = r_k + b_k p_{k-1}$.
Endfor

Relation Between CG Residuals and Lanczos Vectors

If $q_{k+1} := (-1)^k \frac{r_k}{\|r_k\|}$, then

$$Aq_k = \frac{\|r_k\|}{a_{k-1}\|r_{k-1}\|} q_{k+1} + \left(\frac{1}{a_{k-1}} + \frac{b_{k-1}}{a_{k-2}} \right) q_k + \frac{\|r_{k-1}\|}{a_{k-2}\|r_{k-2}\|} q_{k-1}.$$

Let Q_J be the n by J matrix whose columns are $q_1, \ldots, q_J$. Then

$$AQ_J = Q_J T_J + \beta_J q_{J+1} e_J^T.$$

Finite Precision CG

In finite precision arithmetic,

$$AQ_J = Q_J T_J + \beta_J q_{J+1} e_J^T + F_J.$$

To see if previous analysis holds, check:

1. $\epsilon_1 := \max_{j=1,\ldots,J} \|f_j\|/\|A\|$,
2. $\epsilon_2 := \max_{j=1,\ldots,J} |\langle \beta_j q_{j+1}, q_J \rangle|/\|A\|$.

If these are near the machine precision, expect convergence that is as good as the best known implementations (except ones that save vectors and reorthogonalize).

Another Approach to Analysis of Convergence

In exact arithmetic,

$$x_k = x_0 + Q_k T_k^{-1} \beta e_1, \quad \beta = \|r_0\|.$$

For finite precision computation, define:

$$\epsilon_3 := \|x_k - (x_0 + Q_k T_k^{-1} \beta e_1)\| / \|A^{-1} b\|.$$

If ϵ_3 is near the machine precision *and* eigenvalues of T_k lie between the largest and smallest eigenvalues of A, then [Druskin, G., Knizhnerman, 1998]:

$$\frac{\|r_k\|}{\|r_0\|} \lesssim 2\sqrt{\kappa} \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^k.$$

For similar estimates of $\frac{\|f(A)b - Q_k f(T_k)\beta e_1\|}{\|b\|}$, see [Musco, Musco, and Sidford, 2018].

Open Questions

1. What is ultimately attainable accuracy for variants that do not involve a pair of coupled two-term recurrences? For which variants does $r_k \to 0$, and why?

Open Questions

1. What is ultimately attainable accuracy for variants that do not involve a pair of coupled two-term recurrences? For which variants does $r_k \to 0$, and why?
2. Which, if any, variants of CG/Lanczos have small values of ϵ_1, ϵ_2, ϵ_3? Can check this numerically on test problems. Can we prove it?

Open Questions

1. What is ultimately attainable accuracy for variants that do not involve a pair of coupled two-term recurrences? For which variants does $r_k \to 0$, and why?
2. Which, if any, variants of CG/Lanczos have small values of ϵ_1, ϵ_2, ϵ_3? Can check this numerically on test problems. Can we prove it?
3. If an implementation does not have small ϵ_1, ϵ_2, ϵ_3, does it perform poorly? If so, then we should keep these quantities in mind when designing implementations to take better advantage of parallelism, etc. If not, is there a different way to do the analysis that might establish good convergence for such variants?

Summary

The CG/Lanczos algorithm is unusual in that it is known *not* to behave the way exact arithmetic theory predicts when implemented in finite precision arithmetic; the algorithm is, in a sense, **unstable**. Yet it is widely used and often delivers impressive results!

Summary

The CG/Lanczos algorithm is unusual in that it is known *not* to behave the way exact arithmetic theory predicts when implemented in finite precision arithmetic; the algorithm is, in a sense, **unstable**. Yet it is widely used and often delivers impressive results!

This means, however, that any change to an existing implementation may alter the properties that make it converge well in finite precision arithmetic. The better our understanding of this, the more likely we will be able to devise new variants that have desired properties without destroying what makes the algorithm work in practice.