A backward error analysis framework for GMRES

Speaker: Bastien Vieublé
Co-authors: Alfredo Buttari, Nick Higham, and Théo Mary

16/05/2024

SIAM LA24

What is GMRES?

Throughout the presentation, we focus on the Generalized Minimal RESidual (GMRES) algorithm.

Algorithm: GMRES(A, b, x_0, τ)

Require: $A \in \mathbb{R}^{n \times n}$, $b, x_0 \in \mathbb{R}^n$, $\tau \in \mathbb{R}$

$r_0 = b - Ax_0$
$\beta = \|r_0\|,\ v_1 = r_0/\beta,\ k = 1$
repeat
    $w_k = Av_k$
    for $i = 1, \dots, k$ **do**
        $h_{i,k} = v_i^T w_k$
        $w_k = w_k - h_{i,k} v_i$
    end for
    $h_{k+1,k} = \|w_k\|, v_{k+1} = w_k / h_{k+1,k}$
    $V_k = [v_1, \dots, v_k]$
    $H_k = \{h_{i,j}\}_{1 \le i \le j+1; 1 \le j \le k}$
    $y_k = \textbf{argmin}_y \|\beta e_1 - H_k y\|$
    $k = k + 1$
until $\|\beta e_1 - H_k y_k\| \le \tau$
$x_k = x_0 + V_k y_k$

What is GMRES?

Throughout the presentation, we focus on the Generalized Minimal RESidual (GMRES) algorithm.

➤ GMRES = Krylov-based iterative solver for the solution of **general square linear systems $Ax = b$**.

Algorithm: GMRES(A, b, x_0, τ)

Require: $A \in \mathbb{R}^{n \times n}$, $b, x_0 \in \mathbb{R}^n$, $\tau \in \mathbb{R}$

```
r_0 = b − A x_0
β = ||r_0||,  v_1 = r_0/β,  k = 1
repeat
    w_k = A v_k

    for i = 1, ..., k do
        h_{i,k} = v_i^T w_k
        w_k = w_k − h_{i,k} v_i
    end for
    h_{k+1,k} = ||w_k||, v_{k+1} = w_k / h_{k+1,k}
    V_k = [v_1, ..., v_k]
    H_k = {h_{i,j}}_{1≤i≤j+1; 1≤j≤k}
    y_k = argmin_y ||β e_1 − H_k y||
    k = k + 1
until ||β e_1 − H_k y_k|| ≤ τ
x_k = x_0 + V_k y_k
```

What is GMRES?

Throughout the presentation, we focus on the Generalized Minimal RESidual (GMRES) algorithm.

- GMRES = Krylov-based iterative solver for the solution of **general square linear systems** $Ax = b$.
- Computes iteratively an orthonormal **Krylov basis** V_k through an Arnoldi process.

Algorithm: GMRES(A, b, x_0, τ)

Require: $A \in \mathbb{R}^{n\times n}$, $b, x_0 \in \mathbb{R}^n$, $\tau \in \mathbb{R}$

```
r_0 = b − A x_0
β = ||r_0||,  v_1 = r_0/β,  k = 1
repeat
    w_k = A v_k

    for i = 1, ..., k do
        h_{i,k} = v_i^T w_k
        w_k = w_k − h_{i,k} v_i
    end for
    h_{k+1,k} = ||w_k||, v_{k+1} = w_k / h_{k+1,k}
    V_k = [v_1, ..., v_k]
    H_k = {h_{i,j}}_{1≤i≤j+1; 1≤j≤k}
    y_k = argmin_y ||β e_1 − H_k y||
    k = k + 1
until ||β e_1 − H_k y_k|| ≤ τ
x_k = x_0 + V_k y_k
```

What is GMRES?

Throughout the presentation, we focus on the Generalized Minimal RESidual (GMRES) algorithm.

- ➤ GMRES = Krylov-based iterative solver for the solution of **general square linear systems $Ax = b$**.
- ➤ Computes iteratively an orthonormal **Krylov basis V_k** through an Arnoldi process.
- ➤ Chooses the vector x_k in $span\{V_k\}$ that **minimizes $\|Ax_k - b\|$**.

Algorithm: GMRES(A, b, x_0, τ)

Require: $A \in \mathbb{R}^{n \times n}$, $b, x_0 \in \mathbb{R}^n$, $\tau \in \mathbb{R}$

$r_0 = b - Ax_0$
$\beta = \|r_0\|,\ v_1 = r_0/\beta,\ k = 1$
repeat
  $w_k = Av_k$
  for $i = 1, \ldots, k$ **do**
    $h_{i,k} = v_i^T w_k$
    $w_k = w_k - h_{i,k} v_i$
  end for
  $h_{k+1,k} = \|w_k\|, v_{k+1} = w_k / h_{k+1,k}$
  $V_k = [v_1, \ldots, v_k]$
  $H_k = \{h_{i,j}\}_{1 \le i \le j+1; 1 \le j \le k}$
  $y_k = \text{argmin}_y \|\beta e_1 - H_k y\|$
  $k = k + 1$
until $\|\beta e_1 - H_k y_k\| \le \tau$
$x_k = x_0 + V_k y_k$

What is GMRES?

Throughout the presentation, we focus on the Generalized Minimal RESidual (GMRES) algorithm.

- GMRES = Krylov-based iterative solver for the solution of **general square linear systems $Ax = b$**.
- Computes iteratively an orthonormal **Krylov basis V_k** through an Arnoldi process.
- Chooses the vector x_k in $span\{V_k\}$ that **minimizes $\|Ax_k - b\|$**.
- **Reiterate** until x_k is a satisfying approximant of x.

Algorithm: GMRES(A, b, x_0, τ)

Require: $A \in \mathbb{R}^{n\times n}$, $b, x_0 \in \mathbb{R}^n$, $\tau \in \mathbb{R}$

$r_0 = b - Ax_0$

$\beta = \|r_0\|,\ v_1 = r_0/\beta,\ k = 1$

repeat

  $w_k = Av_k$

  for $i = 1, \dots, k$ **do**

    $h_{i,k} = v_i^T w_k$

    $w_k = w_k - h_{i,k} v_i$

  end for

  $h_{k+1,k} = \|w_k\|, v_{k+1} = w_k / h_{k+1,k}$

  $V_k = [v_1, \dots, v_k]$

  $H_k = \{h_{i,j}\}_{1\leq i \leq j+1; 1 \leq j \leq k}$

  $y_k = \textbf{argmin}_y \|\beta e_1 - H_k y\|$

  $k = k + 1$

until $\|\beta e_1 - H_k y_k\| \leq \tau$

$x_k = x_0 + V_k y_k$

GMRES comes in many flavors

Preconditioning

GMRES might converge too slowly. It is essential to use a preconditioner M that transforms $Ax = b$ into an "easier" linear system to solve.

$$M^{-1}Ax = M^{-1}b \quad \text{(left)}, \qquad Au = b, \quad u = Mx \quad \text{(right)}$$

More possibilities: split preconditioning, non-constant preconditioners (FGMRES).

Example of M: ILU, polynomial, block Jacobi, approximate inverse, an iterative method, ...

Restart

Principle: under a chosen restart criterion, stop the iteration, erase V_k, restart GMRES with the initial guess $x_0 = x_k$.

The cost in memory and execution time of an iteration grows as we iterate ⇒ Restart cumulates more iterations while bounding the cost.

Orthogonalization

The Arnoldi process can be constructed with any orthogonalization procedures: Householder QR, CGS, MGS, CGS2, ...

Warning: Different tradeoffs between numerical stability and performance!

What is a backward error analysis?

Backward and forward errors

Even for $k = n$, GMRES computed in finite precision won't deliver the exact solution. We quantify the quality of the computed solution $\widehat{x}_k$ by the quantities

$$bwd = \frac{\|A\widehat{x}_k - b\|}{\|A\|\|\widehat{x}_k\| + \|b\|}, \qquad fwd = \frac{\|x - \widehat{x}_k\|}{\|x\|}.$$

"*The process of bounding the backward error of a computed solution is called backward error analysis*" **N. J. Higham**, Accuracy and Stability of Numerical Algorithms.

Why we care?

- Formal proof that GMRES is able to compute a correct solution.
- Reveals the sensitivity to rounding errors of the different operations.
- Is needed to derive a backward error analysis of an algorithm using GMRES.

Existing backward error analysis of GMRES

Bounding the backward and forward error of GMRES is **NOT EASY**:

- GMRES is a complex algorithm made of different sub-algorithms $\rightarrow$ we need a **backward error analysis on every sub-algorithm**.
- GMRES is an iterative process, **bounds** on the errors are only **valid from a certain** k $\rightarrow$ we need to answer the question: at which k the errors are satisfying.

Existing backward error analysis of GMRES

1995 — Householder GMRES

“*Numerical stability of GMRES*” by **J. Drkošová, A. Greenbaum, M. Rozložník and Z. Strakoš**, BIT Numerical Mathematics.

Existing backward error analysis of GMRES

1995

Householder GMRES

"*Numerical stability of GMRES*" by **J. Drkošová, A. Greenbaum, M. Rozložník and Z. Strakoš**, BIT Numerical Mathematics.

2006

MGS GMRES

"*Modified Gram-Schmidt (MGS), least squares, and backward stability of MGS-GMRES*" by **C. C. Paige, M. Rozložník, and Z. Strakoš**, 2006, SIAM SIMAX.

Existing backward error analysis of GMRES

1995 — Householder GMRES

“*Numerical stability of GMRES*” by **J. Drkošová, A. Greenbaum, M. Rozložník and Z. Strakoš**, BIT Numerical Mathematics.

2006 — MGS GMRES

“*Modified Gram-Schmidt (MGS), least squares, and backward stability of MGS-GMRES*” by **C. C. Paige, M. Rozložník, and Z. Strakoš**, 2006, SIAM SIMAX.

2007-2008 — Flexible MGS GMRES

“*A Note on GMRES Preconditioned by a Perturbed* LDL^T *Decomposition with Static Pivoting*” by **M. Arioli, I. S. Duff, S. Gratton, and S. Pralet**, SIAM SISC.

“*Using FGMRES to obtain backward stability in mixed precision*” by **M. Arioli and I. S. Duff**, ETNA.

Why do we need a new backward error analysis?

The range of possible variants of GMRES is astonishing!

Number of variants =

Why do we need a new backward error analysis?

The range of possible variants of GMRES is astonishing!

Number of variants =

A plethora of preconditioners...

Why do we need a new backward error analysis?

The range of possible variants of GMRES is astonishing!

Number of variants =

A plethora of preconditioners...

× Four ways to apply them: left, right, split, flexible.

Why do we need a new backward error analysis?

The range of possible variants of GMRES is astonishing!

Number of variants =

A plethora of preconditioners...

$\times$ Four ways to apply them: left, right, split, flexible.

$\times$ Restart or not.

Why do we need a new backward error analysis?

The range of possible variants of GMRES is astonishing!

Number of variants =

A plethora of preconditioners...

× Four ways to apply them: left, right, split, flexible.

× Restart or not.

× Possible orthogonalization methods: CGS, MGS, CGS2, Householder, ...

Why do we need a new backward error analysis?

The range of possible variants of GMRES is astonishing!

Number of variants =

A plethora of preconditioners...

× Four ways to apply them: left, right, split, flexible.

× Restart or not.

× Possible orthogonalization methods: CGS, MGS, CGS2, Householder, ...

× All the "more exotic" techniques: communication avoiding, randomization, mixed precision, compression of the basis, ...

Why do we need a new backward error analysis?

The range of possible variants of GMRES is astonishing!

Number of variants =

A plethora of preconditioners...

× Four ways to apply them: left, right, split, flexible.

× Restart or not.

× Possible orthogonalization methods: CGS, MGS, CGS2, Householder, ...

× All the "more exotic" techniques: communication avoiding, randomization, mixed precision, compression of the basis, ...

$\Rightarrow$ An almost infinite number of variants...

Why do we need a new backward error analysis?

... BUT only a tiny subset of them are covered by the previous analyses.

Why do we need a new backward error analysis?

... BUT only a tiny subset of them are covered by the previous analyses.

In addition:

- These analyses were **not** made to be **modular** $\Rightarrow$ Changing one element requires redoing a big part of the analysis.
- They are very smart, long, and hard $\Rightarrow$ Understanding and **adapting them is a challenge**.

Why do we need a new backward error analysis?

... BUT only a tiny subset of them are covered by the previous analyses.

In addition:

- These analyses were **not** made to be **modular** $\Rightarrow$ Changing one element requires redoing a big part of the analysis.
- They are very smart, long, and hard $\Rightarrow$ Understanding and **adapting them is a challenge**.

Consequences:

- A few GMRES variants have error bounds on their computed solution.
- Bounding errors of a new variant is inconvenient and tedious.

Toward a generic and modular tool

Can we provide an analysis...

Toward a generic and modular tool

Can we provide an analysis…

- … that gives the sharpest error bounds?

Toward a generic and modular tool

Can we provide an analysis…

- … that gives the sharpest error bounds?
- … that is generic enough to cover “a lot” of possible GMRES variants (i.e., different preconditioners, orthogonalization, restart, mixed precision, …)?

Toward a generic and modular tool

Can we provide an analysis…

- … that gives the sharpest error bounds?
- … that is generic enough to cover “a lot” of possible GMRES variants (i.e., different preconditioners, orthogonalization, restart, mixed precision, …)?
- … that is modular (if you change the orthogonalization method, you do not need to redo all the analysis)?

Toward a generic and modular tool

Can we provide an analysis...

- ... that gives the sharpest error bounds?
- ... that is generic enough to cover “a lot” of possible GMRES variants (i.e., different preconditioners, orthogonalization, restart, mixed precision, ...)?
- ... that is modular (if you change the orthogonalization method, you do not need to redo all the analysis)?
- ... that is easy to use to some extent?

Toward a generic and modular tool

Can we provide an analysis...

- ... that gives the sharpest error bounds?
- ... that is generic enough to cover "a lot" of possible GMRES variants (i.e., different preconditioners, orthogonalization, restart, mixed precision, ...)?
- ... that is modular (if you change the orthogonalization method, you do not need to redo all the analysis)?
- ... that is easy to use to some extent?

$\Rightarrow$ We aim to propose a modular and generic backward error analysis tool for GMRES.

Modular GMRES: an abstract algorithm

Algorithm: MOD-GMRES(A, b, M_l, Z_k)

1: Compute $C_k = \widetilde{A} Z_k$ where $\widetilde{A} = M_l^{-1} A$.
2: Compute $\widetilde{b} = M_l^{-1} b$.
3: Solve $y_k = \mathbf{argmin}_y \, \|\widetilde{b} - C_k y\|$.
4: Compute the approximant $x_k = Z_k y_k$.

Algorithm: MOD-GMRES(A, b, M_l, Z_k)

Compute $C_k = \widetilde{A} Z_k$ where $\widetilde{A} = M_l^{-1} A$.
Compute $\widetilde{b} = M_l^{-1} b$.
Solve $y_k = \textbf{argmin}_y \|\widetilde{b} - C_k y\|$.
Compute the approximant $x_k = Z_k y_k$.

Principle: Finding $x_k \in span\{Z_k\}$ minimizing the left-preconditioned residual $\|\widetilde{b} - \widetilde{A}x\|$.

- Little assumptions on the operations.
- Z_k can be any basis of rank k.
- Do not assume Arnoldi process.
- Not presented as an iterative process.

➤ Can be seen as a subspace projection method solving the left-preconditioned system in span$\{Z_k\}$, where the left-preconditioner M_l, the basis Z_k, and the least squares solver are not specified.

Modular GMRES: an abstract algorithm

Algorithm: MOD-GMRES(A, b, M_l, Z_k)

1: Compute $C_k = \widetilde{A} Z_k$ where $\widetilde{A} = M_l^{-1} A$.
2: Compute $\widetilde{b} = M_l^{-1} b$.
3: Solve $y_k = \textbf{argmin}_y \|\widetilde{b} - C_k y\|$.
4: Compute the approximant $x_k = Z_k y_k$.

Specialization to:

Algorithm: MGS GMRES without preconditioner

1: Compute $C_k = A\widehat{V}_k$, where $M_l = I$ and $\widehat{V}_k$ is the computed Arnoldi basis.
2:
3: Solve $y_k = \textbf{argmin}_y \|b - A\widehat{V}_k y\|$ by MGS Arnoldi.
4: Compute the approximant $x_k = \widehat{V}_k y_k$.

Modular GMRES: an abstract algorithm

Algorithm: MOD-GMRES(A, b, M_l, Z_k)

1: Compute $C_k = \widetilde{A}Z_k$ where $\widetilde{A} = M_l^{-1}A$.
2: Compute $\widetilde{b} = M_l^{-1}b$.
3: Solve $y_k = \mathbf{argmin}_y \|\widetilde{b} - C_k y\|$.
4: Compute the approximant $x_k = Z_k y_k$.

Specialization to:

Algorithm: MGS GMRES with left- LU preconditioner

1: Compute $C_k = \widetilde{A}\widehat{V}_k$, where $\widetilde{A} = U \backslash L \backslash A$ and $\widehat{V}_k$ is the Arnoldi basis.
2: Compute $\widetilde{b} = U \backslash L \backslash b$.
3: Solve $y_k = \mathbf{argmin}_y \|\widetilde{b} - \widetilde{A}\widehat{V}_k y\|$ by MGS Arnoldi.
4: Compute the approximant $x_k = \widehat{V}_k y_k$.

Algorithm: MOD-GMRES(A, b, M_l, Z_k)

1: Compute $C_k = \widetilde{A} Z_k$ where $\widetilde{A} = M_l^{-1} A$.
2: Compute $\widetilde{b} = M_l^{-1} b$.
3: Solve $y_k = \mathbf{argmin}_y \|\widetilde{b} - C_k y\|$.
4: Compute the approximant $x_k = Z_k y_k$.

Specialization to:

Algorithm: CGS2 GMRES with flexible LU preconditioner

1: Compute $C_k = A Z_k$, where $M_l = I$ and $Z_k = U \backslash L \backslash \widehat{V}_k$.
2:
3: Solve $y_k = \mathbf{argmin}_y \|b - A Z_k y\|$ by CGS2 Arnoldi.
4: Compute the approximant $x_k = Z_k y_k$.

Modular GMRES: an abstract algorithm

Algorithm: MOD-GMRES(A, b, M_l, Z_k)

1: Compute $C_k = \widetilde{A} Z_k$ where $\widetilde{A} = M_l^{-1} A$.
2: Compute $\widetilde{b} = M_l^{-1} b$.
3: Solve $y_k = \mathbf{argmin}_y \|\widetilde{b} - C_k y\|$.
4: Compute the approximant $x_k = Z_k y_k$.

MOD-GMRES is an abstract generic algorithm that can be specialized to many GMRES algorithms $\Rightarrow$ Any result on MOD-GMRES holds for its specializations.

Our goal: Make a backward error analysis of MOD-GMRES.

One analysis to rule them all!

Modular rounding error model

The terms $\epsilon_{\tilde{A}}$, ϵ_{b}, ϵ_{LS}, and ϵ_{Z} quantify the accuracies of every operation and are unspecified. They are only specified for a given specialization of MOD-GMRES.

Matrix–matrix product with the basis (step 1)

$$\mathrm{fl}(\widetilde{A}Z_k) = \widetilde{A}Z_k + \Delta_{\widetilde{A}Z_k}, \qquad \|\Delta_{\widetilde{A}Z_k}\| \leq \epsilon_{\tilde{A}} \|\widetilde{A}Z_k\|.$$

Preconditioned RHS (step 2)

$$\mathrm{fl}(M_l^{-1}b) = \widetilde{b} + \Delta\widetilde{b}, \qquad \|\Delta\widetilde{b}\| \leq \epsilon_{b} \|\widetilde{b}\|.$$

Least squares solution (step 3)

$$\widehat{y}_k = \mathrm{argmin}_y \|\widetilde{b} + \Delta b' - (\mathrm{fl}(AZ_k) + \Delta'_{\widetilde{A}Z_k})\|$$

$$\|[\Delta\widetilde{b}', \Delta'_{\widetilde{A}Z_k}]e_j\| \leq \epsilon_{LS} \|[\widetilde{b}, \mathrm{fl}(AZ_k)]e_j\|$$

Compute the kth approximant (step 4)

$$\widehat{x}_k = \mathrm{fl}(Z_k\widehat{y}_k) = (Z_k + \Delta Z_k)\widehat{y}_k, \qquad \|\Delta Z_k\| \leq \epsilon_Z \|Z_k\|$$

A key dimension(/iteration)

We need to define the special dimension(/iteration) k at which we can demonstrate that the computed solution has attained a satisfying error.

Key dimension

We define the key dimension k as the first $k \leq n$ such that, for all $\phi > 0$, we have

$$\sigma_{\min}([\widetilde{b}\phi, \widetilde{A}Z_k]) \leq (\epsilon_{\tilde{A}} + \epsilon_b + \epsilon_{LS}) \|[\widetilde{b}\phi, \widetilde{A}Z_k]\|_F$$

and

$$\sigma_{\min}(\widetilde{A}Z_k) \gg (\epsilon_{\tilde{A}} + \epsilon_b + \epsilon_{LS}) \|\widetilde{A}Z_k\|_F.$$

The philosophy of these conditions is to capture the exact moment where $\widetilde{b}$ lies in the range of $\widetilde{A}Z_k$, which is the moment where the basis Z_k contains the solution.

"Modified Gram-Schmidt (mgs), least squares, and backward stability of MGS-GMRES" by **C. C. Paige, M. Rozložník, and Z. Strakoš**, 2006, SIAM SIMAX.

Error bounds of MOD-GMRES

Theorem

Consider the solution of a nonsingular linear system

$$Ax = b, \quad A \in \mathbb{R}^{n\times n}, \quad 0 \neq b \in \mathbb{R}^n,$$

with MOD-GMRES under the previous **error model**. **If there exists a key dimension** k as defined previously, then, MOD-GMRES produces a computed solution $\widehat{x}_k$ whose **backward** and **forward** error satisfies respectively

$$\frac{\|b - A\widehat{x}_k\|}{\|b\| + \|A\|\|\widehat{x}_k\|} \lesssim \Phi\kappa(M_l), \qquad \frac{\|\widehat{x}_k - x\|}{\|x\|} \lesssim \Phi\kappa(\widetilde{A}),$$

where

$$\Phi \equiv \alpha\epsilon_{\tilde{A}} + \beta\epsilon_{b} + \beta\epsilon_{LS} + \lambda\epsilon_{Z}$$

with

$$\alpha \equiv \sigma_{\min}^{-1}(Z_k)\frac{\|\widetilde{A}Z_k\|}{\|\widetilde{A}\|}, \quad \beta \equiv \max(1, \sigma_{\min}^{-1}(Z_k)\frac{\|\widetilde{A}Z_k\|}{\|\widetilde{A}\|}), \quad \lambda \equiv \sigma_{\min}^{-1}(Z_k)\|Z_k\|.$$

How to use?

How to use the previous result to **derive** forward and backward error bounds **for real GMRES algorithms**?

How to use?

How to use the previous result to **derive** forward and backward error bounds **for real GMRES algorithms**?

Using the previous theorem requires some work:

- Show that your algorithm is a **specialization of MOD-GMRES**.
- **Determine** $\epsilon_{\tilde{A}}$, ϵ_{b}, ϵ_{LS}, and ϵ_{Z}. The difficulty of this step varies according to the existing literature of the sub-algorithms used.
- Show the existence of **the key dimension**. The difficulty also varies according to the existing literature.

How to use?

How to use the previous result to **derive** forward and backward error bounds **for real GMRES algorithms**?

Using the previous theorem requires some work:

- Show that your algorithm is a **specialization of MOD-GMRES**.
- **Determine** $\epsilon_{\tilde{A}}$, ϵ_{b}, ϵ_{LS}, and ϵ_{Z}. The difficulty of this step varies according to the existing literature of the sub-algorithms used.
- Show the existence of **the key dimension**. The difficulty also varies according to the existing literature.

This Theorem is **backward compatible with the previous analyses**: Applying it on Householder GMRES, MGS GMRES, and Flexible MGS GMRES gives the same results as the existing analyses.

Conclusion

Takeaways

- Many GMRES variants not covered by a backward error analysis.
- We propose a backward error analysis framework to efficiently derive error bounds on new variants.
- We can apply this framework on most existing GMRES using approximate computing.

"A modular framework for the backward error analysis of GMRES" by **A. Buttari, N. J. Higham, T. Mary, B. Vieublé**, Preprint.