R^3GMRES: Including Prior Information in GMRES-Type Methods for Discrete Inverse problems

Per Christian Hansen Technical University of Denmark

Joint work with:
- Yiqiu Dong
- Henrik Garde

With thanks to Nao Kuroiwa and Emil B. Kærgaard.

DTU Compute
Department of Applied Mathematics and Computer Science

Overview of Talk

Forward problem

Inverse Problem

- Discrete inverse problem: $A\, x = b$ or $\min_x \| A\, x - b \|_2$
- Iterative Krylov-subspace methods – regularizing iterations
- Augmenting the Krylov subspace – different approaches
- Implementation issues – the R^3GMRES algorithm
- Numerical examples

Regularization Algorithm

Variational formulations take the form

$$\min_x \left\{ \|A\,x - b\|_2^2 + \lambda\,\mathcal{R}(x) \right\}$$

where $\mathcal{R}(x)$ is a regularization terms that penalizes unwanted features in the solution, and λ is a user-chosen regularization parameter.

An alternative formulation:

$$\min_x \|A\,x - b\|_2^2 \quad \text{s.t.} \quad x \in \mathcal{S}_k \ ,$$

where $\mathcal{S}_k$ is a linear subspace of dimension k – the "signal subspace."

If $\mathcal{S}_k$ is chosen such that it captures the main features in the solution, then this approach can be an interesting alternative for large-scale problems.

The Signal Subspace

In some applications we can use a pre-determined subspace, e.g., spanned by the Fourier basis, the discrete cosine bases, a wavelet basis, etc.

Alternatively we can use a subspace determined by the given problem, e.g., the Krylov subspace associated with a specific iterative method

$$\begin{aligned} \text{CGLS} &: \quad \mathrm{span}\{A^T b, A^T A\, A^T b, (A^T A)^2 A^T b, \ldots\}\ , \\ \text{GMRES} &: \quad \mathrm{span}\{b, A\, b, A^2\, b, \ldots\}\ , \\ \text{RRGMRES} &: \quad \mathrm{span}\{A\, b, A^2 b, A^3 b, \ldots\}\ . \end{aligned}$$

For noisy data RRGMRES is preferable to GMRES (the noisy b is not in the Krylov subspace).

Thus our concern here is with the RRGMRES method for a square matrix $A \in \mathbb{R}^{n\times n}$, where the jth iterate $x^{(j)}$ is in

$$\mathcal{K}_j(A, A\, b) = \mathrm{span}\{A\, b, A^2 b, A^3 b, \ldots, A^j b\}\ .$$

The Augmented Signal Subspace

Let $\mathcal{W}_p$ denote a linear subspace that captures additional specific components of the desired solution; $\dim(\mathcal{W}_p) = p \ll j =$ no. its.

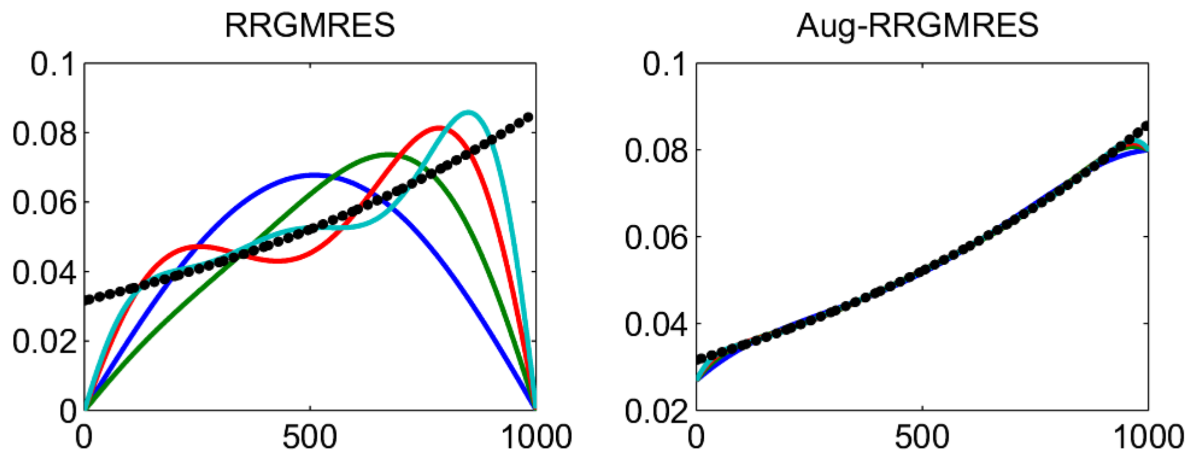
Then it can be advantageous (Baglama & Reichel, several papers) to consider the *augmented* linear subspace, in the case of RRGMRES:

$$\mathcal{S}_{p,j} \equiv \mathcal{W}_p + \mathcal{K}_j(A, Ab) \ , \quad \mathcal{W}_p = \mathcal{R}(W_p) = \text{span}\{w_1, w_2, \ldots, w_p\} \ .$$

Example: deriv2.

All vectors in the Krylov subspace $\rightarrow$ 0 at the ends.

$w_1 = (1,1,\ldots,1)^T$
$w_2 = (1,2,\ldots,n)^T$



Thus we want an efficient iterative algorithm to solve the problem

$$\boxed{\min_x \|A\,x - b\|_2^2 \quad \text{s.t.} \quad x \in \mathcal{S}_{p,j} \ .}$$

Augmented (RR)GMRES

Baglama and Reichel (2007) developed A(RR)GMRES which leaves the component of x in $\mathcal{W}_p$ unchanged and builds a Krylov subspace from that.

The implementation is nice and simple, and very similar to (RR)GMRES.

Starting with the QR fact. $A\,W_p = V_p\,R$ it builds the factorization

$$A\,[\,W_p\,,\,\bar{V}_{p+1:p+1}\,] = \bar{V}_{p+j+1}\,\bar{H}_{p+j}\ ,$$

where $\bar{H}_{p+j}$ is upper Hessenberg and $\bar{V}_{p+j+1} = [\,V_p\,,\,\bar{V}_{p+1:p+j}\,,\,\bar{v}_{p+j+1}\,]$ has orthonormal columns. Then

$$x^{(j)} = [\,W_p\,,\,\bar{V}_{p+1,p+j}\,]\,y^{(j)}\ ,\qquad y^{(j)} = \mathrm{argmin}\|\bar{H}_{p+1}y - \bar{V}_{p+j+1}^T b\|_2^2\ .$$

But this algorithm actually solves the problem

$$\boxed{\min_x \|A\,x - b\|_2^2 \quad \text{s.t.} \quad x \in \mathcal{W}_p + \mathcal{K}_j\big((I - V_pV_p^T)A, (I - V_pV_p^T)Ab\big)\ .}$$

Regularized RRGMRES

We derive the algorithm *Regularized RRGMRES*, or R^3GMRES, that solves

$$\min_x \|A\,x - b\|_2^2 \quad \text{s.t.} \quad x \in \mathcal{S}_{p,j} \ .$$

Key observation; we should restrict the Krylov subspace to $\mathcal{W}_p^\perp$, instead of $\mathcal{R}(V_p)^\perp = \mathcal{R}(A\,W_p)$.

Our algorithm is perhaps less simple than ARRGMRES.

The intuitive/naive formulation uses the Hessenberg decomposition

$$A\,[\,W_p\,,\ Ab\,,A^2b\,,\cdots,\ A^jb\,] = V_{p+j+1}\,H_{p+j}$$

and computes the solution as

$$\begin{aligned} x^{(j)} &= [\,W_p\,,\ Ab\,,A^2b\,,\cdots,\ A^jb\,]\,y^{(j)} \ , \\ y^{(j)} &= \mathrm{argmin}_y \|H_{p+1}y - V_{p+j+1}^T b\|_2^2 \ . \end{aligned}$$

Algorithm: Naïve Version

1. Compute the QR factorization $AW_p = V_p H_p$, where $V_p \in \mathbb{R}^{n\times p}$ and $H_p \in \mathbb{R}^{p\times p}$.
2. Let $u_1 = Ab$, $v_{p+1} = P_{V_p}^{\perp} u_1$ and normalize $u_1 = u_1/\|u_1\|_2$, $v_{p+1} = v_{p+1}/\|v_{p+1}\|_2$. Then expand $V_{p+1} := [\, V_p \, , \, v_{p+1} \,]$ and $W_{p+1} := [\, W_p \, , \, u_1 \,]$.
3. Initialize $R_1 := 1$, and set $j := 1$.
4. Compute $v_{p+j+1} = Au_j$ and $u_{j+1} = v_{p+j+1}/\|v_{p+j+1}\|_2$.
5. Apply MGS orthonormalization to v_{p+j+1} and expand $V_{p+j+1} := [\, V_{p+j} \, , \, v_{p+j+1} \,]$, $H_{p+j} := \begin{bmatrix} H_{p+j-1} & h_{p+j} \\ 0 & \end{bmatrix} \in \mathbb{R}^{(p+j+1)\times(p+j)}$, where h_{p+j} is from the MGS.
6. Solve $\min_y \left\| H_{p+j} \begin{bmatrix} I_p & 0 \\ 0 & R_j^{-1} \end{bmatrix} y - V_{p+j+1}^{\top} b \right\|_2^2$ to obtain $y^{(j)}$. Then $x^{(j)} = W_{p+j} y^{(j)}$.
7. Apply MGS orthonormalization to u_{j+1} such that $\{u_1, \ldots, u_{j+1}\}$ becomes an orthonormal basis for $\mathcal{K}_{j+1}(A, Ab)$, expand $W_{p+j+1} = [\, W_{p+1} \, , \, u_{j+1} \,]$, and expand $R_{j+1} := \begin{bmatrix} R_j & r_{j+1} \\ 0 & \end{bmatrix} \in \mathbb{R}^{(j+1)\times(j+1)}$, where r_{j+1} is from the MGS.
8. Stop, or set $j := j+1$ and return to step 4.

Two MGS: needs additional $O(j^2 n)$ flops compared to ARRGMRES.

Towards Our Algorithm

The key idea is to run the standard Arnoldi process from RRGMRES to compute an orthonormal basis of $\mathcal{K}_j(A, Ab)$, and then augment it by $\mathcal{W}_p$ in each step of the iterative algorithm.

This seems cumbersome – but the overhead is favorably small!

At step j we have the decomposition

$$A\,[\,V_j\,,\,W_p\,] = \left[\,V_{j+1}\,,\,\widetilde{V}_j\,\right] \begin{bmatrix} H_j & G_j \\ 0 & F_j \end{bmatrix}$$

- $AV_j = V_{j+1}H_j$ is obtained after j steps of the Arnoldi process.
- $V_j \in \mathbb{R}^{n\times j}$ has orthonormal columns with $v_1 = Ab/\|Ab\|_2$.
- $H_j \in \mathbb{R}^{(j+1)\times j}$ is an upper Hessenberg matrix.
- The columns of V_j form a orthonormal basis of $\mathcal{K}_j(A, Ab)$.

We then augment this basis to a basis of $\mathcal{S}_{p,j}$, namely, $[\,V_j\,,\,W_p\,]$.

More Details

Recall that

$$A\,[\,V_j\,,\,W_p\,] = \Big[\,V_{j+1}\,,\,\widetilde{V}_j\,\Big] \left[\begin{array}{cc} H_j & G_j \\ 0 & F_j \end{array}\right] .$$

We must augment V_{j+1} with a basis of $\mathcal{R}(AW_p)$, which gives the augmented matrix $[\,V_{j+1}\,,\,\widetilde{V}_j\,]$, where the orthonormal vectors in $\widetilde{V}_j \in \mathbb{R}^{n\times p}$ are orthogonal to the columns of V_{j+1}.

We introduce $G_j \in \mathbb{R}^{(j+1)\times p}$ and $F_j \in \mathbb{R}^{p\times p}$ which are composed of the coefficients of AW_p with respect to the basis of $\mathcal{V}_{j+1}$ and the subspace of $\mathcal{V}_{j+1}^{\perp}$, respectively:

$$G_j = V_{j+1}^{\top} A W_p, \qquad F_j = \widetilde{V}_j^{\top} A W_p\ .$$

Then the iterate $x^{(j)} \in \mathcal{S}_{p,j}$ is given by $x^{(j)} = [\,V_j\,,\,W_p\,]\,y^{(j)}$, where

$$y^{(j)} = \mathrm{argmin}_y \left\| \left[\begin{array}{cc} H_j & G_j \\ 0 & F_j \end{array}\right] y - \left[\begin{array}{c} V_{j+1}^{\top} \\ \widetilde{V}_j^{\top} \end{array}\right] b \right\|_2^2 .$$

Algorithm: R³GMRES

1. Set $v_1 = Ab/\|Ab\|_2$, $V_1 := v_1$, $G_0 := v_1^\top A W_p$, and $j := 1$.
2. Use the Arnoldi process to obtain v_{j+1} and h_j such that $AV_j = V_{j+1} H_j$, where

   $V_{j+1} := [\, V_j \,,\, v_{j+1} \,]$ and $H_j := \begin{bmatrix} H_{j-1} & h_j \\ 0 & \end{bmatrix} \in \mathbb{R}^{(j+1)\times j}$ (with $H_1 = h_1$).
3. Compute $G_j = \begin{bmatrix} G_{j-1} \\ v_{j+1}^\top A W_p \end{bmatrix} \in \mathbb{R}^{(j+1)\times p}$.
4. Orthonormalize AW_p with respect to V_{j+1} to obtain $\widetilde{V}_j$.
5. Compute $F_j = \widetilde{V}_j^\top A W_p$.
6. Solve $\min_y \left\| \begin{bmatrix} H_j & G_j \\ 0 & F_j \end{bmatrix} y - \begin{bmatrix} V_{j+1}^\top \\ \widetilde{V}_j^\top \end{bmatrix} b \right\|_2^2$ to obtain $y^{(j)}$.
7. Then $x^{(j)} = [\, V_j \,,\, W_p \,]\, y^{(j)}$.
8. Stop, or set $j := j+1$ and return to step 2.

Recomputation of $\widetilde{V}_j$ and F_j in each step; but p is small!

Implementation Details – I

Efficiency: update the orthogonal factorization:

$$\begin{bmatrix} H_j & G_j \\ 0 & F_j \end{bmatrix} = Q \begin{bmatrix} T_j^{(11)} & T_j^{(12)} \\ 0 & T_j^{(22)} \\ 0 & 0 \end{bmatrix},$$

$T_j^{(11)} \in \mathbb{R}^{j\times j}$ and $T_j^{(22)} \in \mathbb{R}^{p\times p}$ are upper triangular, Q is orthogonal.

Update $T_j^{(11)}$ via Givens transf. as in (RR)GMRES algorithms; the rotations are also applied to G_j and the right-hand side, i.e., to $V_{j+1}^{\top} b$.

At this stage we have an intermediate system:

$$\begin{bmatrix} T_j^{(11)} & \text{intermediate} & \\ 0 & F_j & \widetilde{V}_j^{\top} b \end{bmatrix} = \left[\begin{array}{ccc|cc||c} \times & \times & \times & \times & \times & \times \\ & \times & \times & \times & \times & \times \\ & & \times & \times & \times & \times \\ & & & \color{blue}{\times} & \color{blue}{\times} & \color{blue}{\times} \\ \hline & & & \times & \times & \times \\ & & & \times & \times & \times \end{array}\right] \begin{array}{l} \\ \\ \\ \leftarrow \color{blue}{\text{save row } j+1} \\ \\ \\ \end{array}$$

Implementation Details – II

To complete the orthogonal reduction, we apply an orthogonal transformation that involves the bottom $p+1$ rows of the system and produces a system of the form:

$$
\left[\begin{array}{ccc|cc||c}
\times & \times & \times & \times & \times & \times \\
 & \times & \times & \times & \times & \times \\
 & & \times & \times & \times & \times \\
 & & & * & * & * \\
\hline
 & & & & * & * \\
 & & & & & *
\end{array}\right]
$$

where $*$ denotes an element that has changed. Note that $T_j^{(22)}$ in this example consists of the elements in rows 4–5 and columns 4–5.

Implementation Details – III

Next iteration: Arnoldi produces a new column of H_j in the (1,1)-block. This block is then reduced to upper triangular form:

$$
\left[\begin{array}{cccc|cc||c}
\otimes & \otimes & \otimes & \times & \otimes & \otimes & \otimes \\
 & \otimes & \otimes & \times & \otimes & \otimes & \otimes \\
 & & \otimes & \times & \otimes & \otimes & \otimes \\
 & & & \times & \otimes & \otimes & \otimes \\
 & & & \times & \times & \times & \times \\
\hline
 & & & & \times & \times & \times \\
 & & & & \times & \times & \times
\end{array}\right]
\rightarrow
\left[\begin{array}{cccc|cc||c}
\otimes & \otimes & \otimes & \star & \otimes & \otimes & \otimes \\
 & \otimes & \otimes & \star & \otimes & \otimes & \otimes \\
 & & \otimes & \star & \otimes & \otimes & \otimes \\
 & & & * & * & * & * \\
 & & & & * & * & * \\
\hline
 & & & & \times & \times & \times \\
 & & & & \times & \times & \times
\end{array}\right]
$$

$\otimes$ are from the intermediate system of the previous it., $\times$ are new.

Elements $\star$ are updated by means of the stored Givens transf. from the previous its., and $*$ are transformed by the new Givens rotation.

This is followed by an orthogonal transformation involving the bottom $p+1$ rows of the system, as before.

Implementation Details – IV

Next iteration: Arnoldi produces a new column of H_j in the (1,1)-block. This block is then reduced to upper triangular form:

$$
\left[\begin{array}{cccc|cc||c}
\otimes & \otimes & \otimes & \times & \otimes & \otimes & \otimes \\
 & \otimes & \otimes & \times & \otimes & \otimes & \otimes \\
 & & \otimes & \times & \otimes & \otimes & \otimes \\
 & & & \times & {\color{blue}\otimes} & {\color{blue}\otimes} & {\color{blue}\otimes} \\
 & & & \times & \times & \times & \times \\
\hline
 & & & & \times & \times & \times \\
 & & & & \times & \times & \times
\end{array}\right]
\rightarrow
\left[\begin{array}{cccc|cc||c}
\otimes & \otimes & \otimes & \star & \otimes & \otimes & \otimes \\
 & \otimes & \otimes & \star & \otimes & \otimes & \otimes \\
 & & \otimes & \star & \otimes & \otimes & \otimes \\
 & & & * & * & * & * \\
 & & & & * & * & * \\
\hline
 & & & & \times & \times & \times \\
 & & & & \times & \times & \times
\end{array}\right]
$$

$\otimes$ are from the intermediate system of the previous it., $\times$ are new.

In the previous iteration ($j = 3$), row $j + 1 = 4$ of the intermediate system was *overwritten* to obtain triangular form.

Therefore we must save this row, so we can insert it again in the system at the beginning of the next iteration ($j = 4$), before the Givens rotation is applied.

The Work in R^{3}GMRES

Consider the additional work in R^{3}GMRES, compared to RRGMRES where the work in j iterations is $O(j^2 n)$ flops.

In each R^{3}GMRES iteration the additional work is dominated by:

1. orthonormalization of the columns of $\widetilde{V}_j$ to v_{j+1}: $2pn$ flops,
2. computation of the new F_j: $2p^2 n$ flops (assuming AW_p is stored),
3. application of an orthogonal transformation that involves the bottom right $(p+1) \times p$ submatrix: $\approx 2p^3$ flops.

Hence, the additional work in j iterations is about $2jp(p+1)n$ flops.

Numerical Examples

Setting up the test problems:

1. Generate noise-free system: $A\,x_{\text{exact}} = b_{\text{exact}}$.
2. Add noise: $b = b_{\text{exact}} + e$ where e is a random vector of Gaussian white noise scaled such that $\|e\|_2/\|b_{\text{exact}}\|_2 = \eta$.
3. We show best solution within the iterations plus:
   - relative error $\|x_{\text{exact}} - x^{(j)}\|_2/\|x_{\text{exact}}\|_2$,
   - relative residual norm $\|b - A\,x^{(j)}\|_2/\|b\|_2$.

We compare combinations of the following algorithms:

- CGLS is the implementation from Regularization Tools.
- PCGLS is the subspace-preconditioned CGLS algorithm from Regularization Tools, $L \approx$ second derivative.
- RRGMRES is the implementation from Regularization Tools.
- ARRGMRES is our implementation of Augmented RRGMRES.
- $\mathsf{R}^3\mathsf{GMRES}$ is our new algorithm.

Large Component in Augment. Subspace

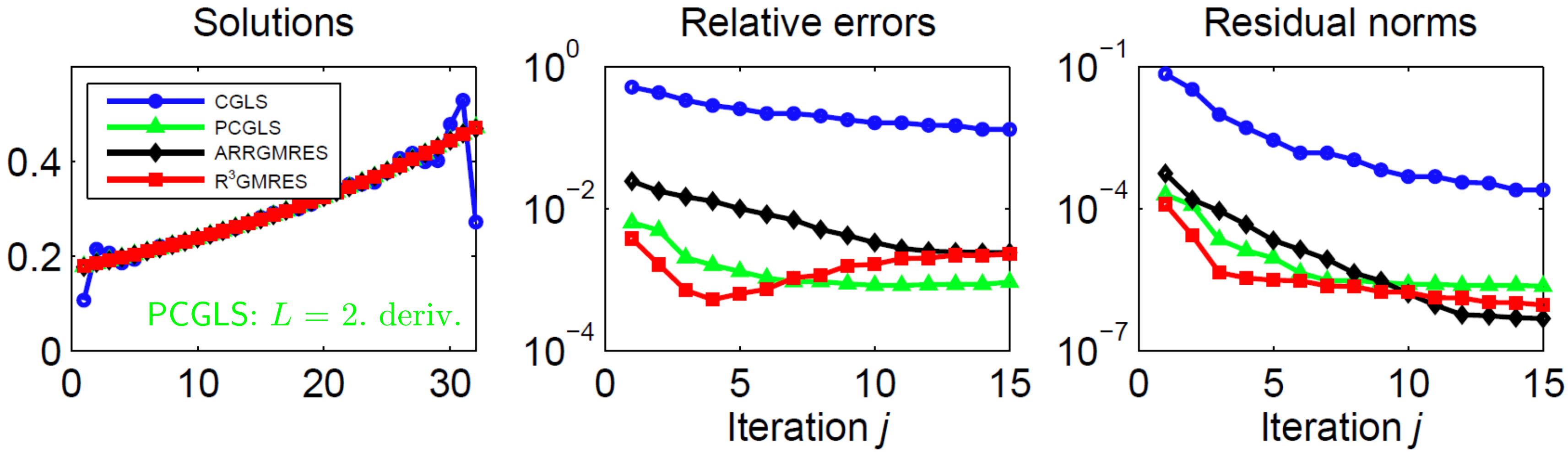
Test problem `deriv2(n,2)`, $n = 32$, relative noise level $\eta = 10^{-5}$.

$$\mathcal{W}_2 = \mathrm{span}\{w_1, w_2\}, \quad w_1 = (1, 1, \ldots, 1)^\top, \quad w_2 = (1, 2, \ldots, n)^\top.$$

For this problem

$$\|W_2 W_2^\top x_{\mathrm{exact}}\|_2 / \|x_{\mathrm{exact}}\|_2 = 0.99 \ ,$$
$$\|(I - W_2 W_2^\top) x_{\mathrm{exact}}\|_2 / \|x_{\mathrm{exact}}\|_2 = 0.035 \ ;$$

we only need to spend effort in capturing the small component in $\mathcal{W}_2^\perp$.

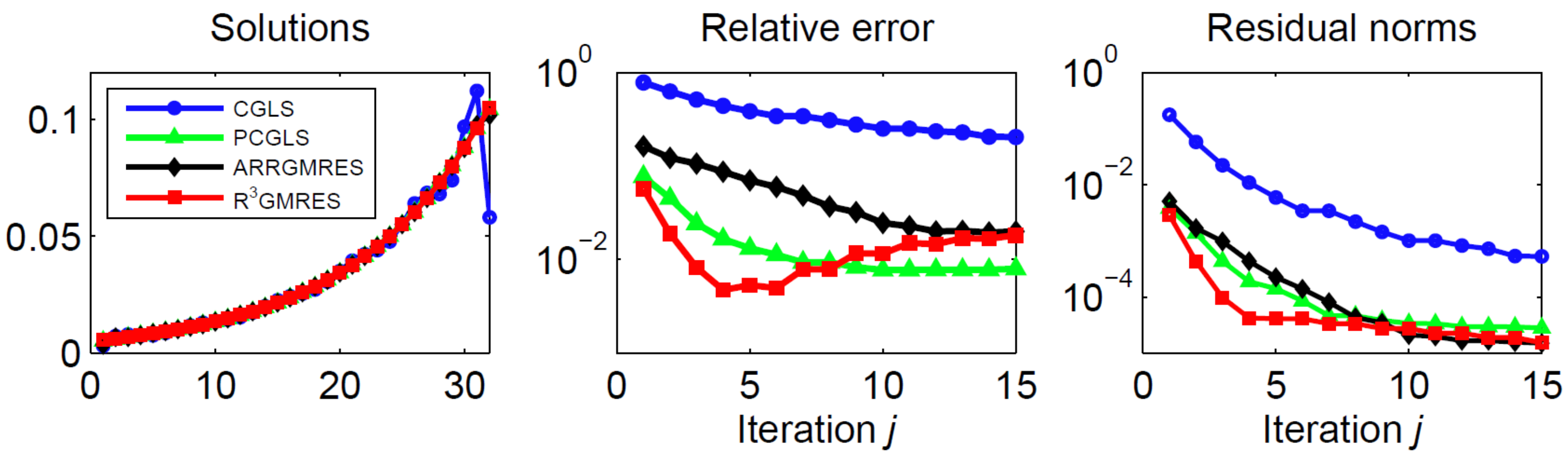


Fix Boundary Conditions

Same problem as before, except a modified exact solution:

```
x = x.^3
```

The new exact solution has a large first derivative at the right endpoint.



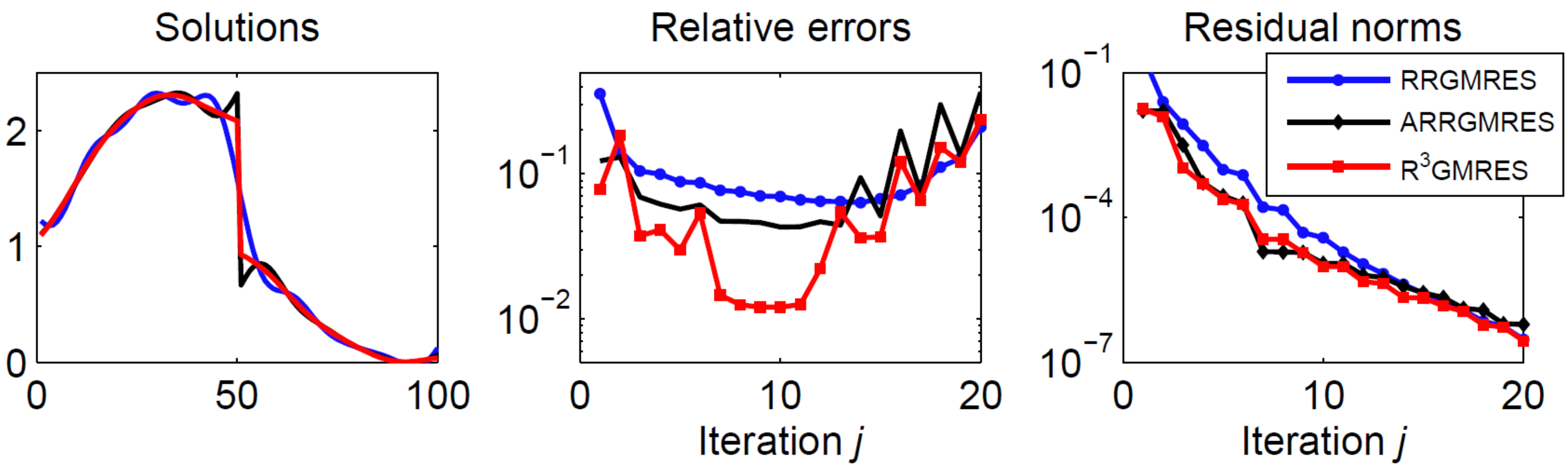
Here $\mathcal{W}_2$ compensates for the "incorrect" or "incompatible" boundary conditions implicit in A, by allowing the regularized solutions to have nonzero values and nonzero derivatives at the endpoints.

Capture a Discontinuity

Test problem **gravity(n)**, $n = 100$, $\eta = 10^{-3}$, exact sol. changed to include a single discontinuity between elements $\ell = 50$ and $\ell + 1 = 51$.

Augmentation matrix W_2 allows us to represent this discontinuity:

$$w_1 = \begin{bmatrix} \text{ones}(\ell, 1) \\ \text{zeros}(n-\ell, 1) \end{bmatrix}, \quad w_2 = \begin{bmatrix} \text{zeros}(\ell, 1) \\ \text{ones}(n-\ell, 1) \end{bmatrix}.$$

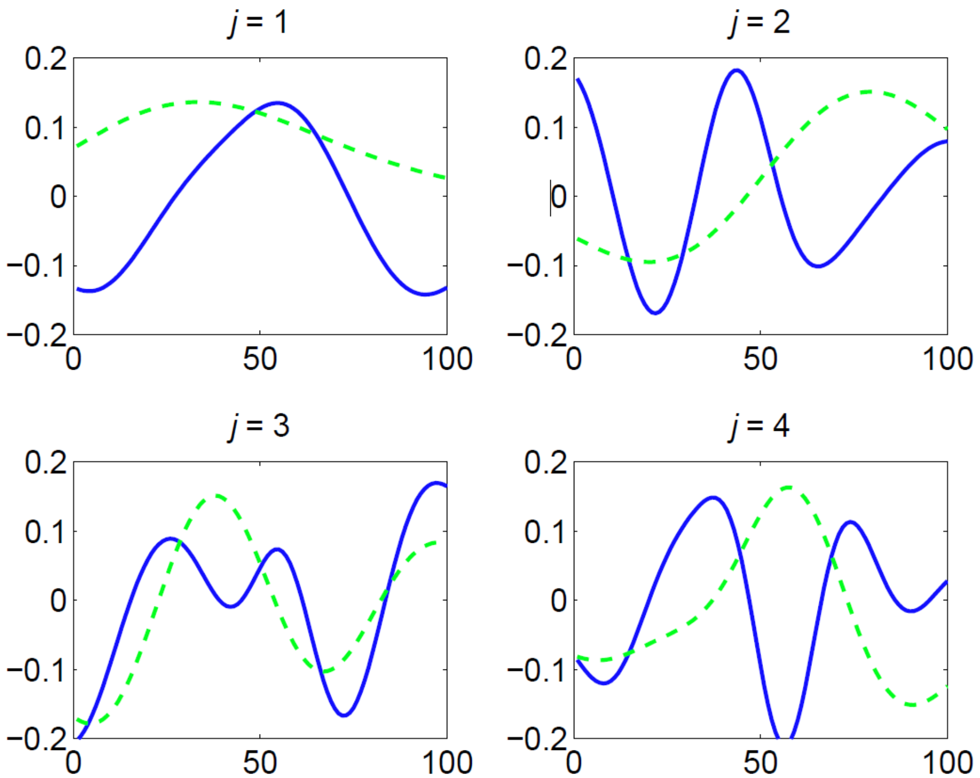


Error history for R^{3}GMRES is not smooth; not an error, since it is only the residual norm that has guaranteed monotonic behavior.

Capture a Discontinuity – More Insight

Why is R^3 much better than ARRGMRES? Look at the basis vectors:

- $\bar{v}_{p+1}, \bar{v}_{p+1}, \ldots$ basis for $\mathcal{K}_j\big((I - V_p V_p^T) A, (I - V_p V_p^T) A b\big)$.
- $v_1, v_2, \ldots$ basis for $\mathcal{K}_j(A, Ab)$.



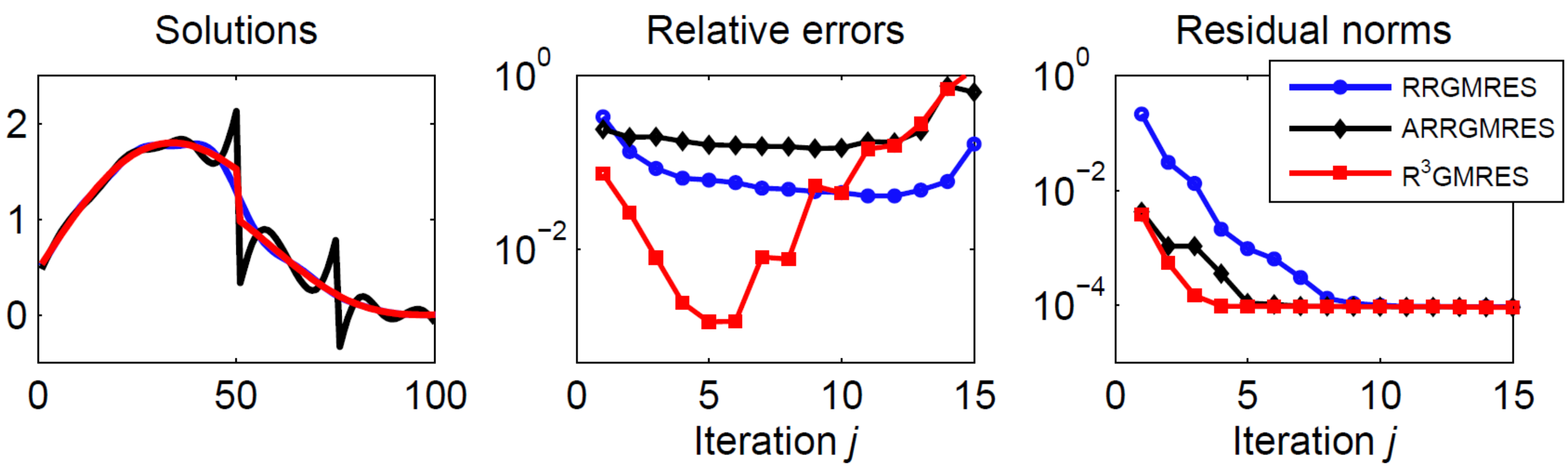
The first two smooth components in R^{3}GMRES, represented by v_1 and v_2, are missing from the basis for ARRGMRES.

Handling a Potential Discontinuity

What happens if we include a discontinuity in $\mathcal{W}_p$ that is not present in the exact solution?

I.e., our prior information tells us about *potential* discontinuities, but not all of them may be present in the given problem.

In this example there is *one* discontinuity in the solution, but *two* in $\mathcal{W}_p$ between elements 50–51 and 75–76. The noise level is $\eta = 10^{-4}$.



Only R^3GMRES captures the single discontinuity correctly.

Conclusions

- We consider how to augment the Krylov subspace.
- Focus here on RRGMRES.
- We developed an efficent algorihtm R^{3}GMRES.
- Numerical examples demonstrate the advantage of R^{3}GMRES.
- Future work:
  - When is it necessary to do the MGS twice?
  - How to augment the Krylov subspace for CGLS?