On some Krylov subspace methods tailored for large-scale two-by-two block linear systems

Kui Du

kuidu@xmu.edu.cn

School of Mathematical Sciences, Xiamen University

https://kuidu.github.io

joint work with Jia-Jun Fan and Fang Wang

The 22nd Annual Meeting of CSIAM, Nanjing, 2024

Outline

1. Two-by-two block linear systems
2. SPMR-SC, SPQMR-SC, nsLSQR
3. GPMR
4. Randomized Gram–Schmidt process, randomized GMRES
5. Sketched GMRES + k-truncated Arnoldi
6. Summary

Two-by-two block linear systems

- Nonsymmetric saddle-point linear systems of the form:

$$\begin{bmatrix} \mathbf{M} & \mathbf{A}^\top \\ \mathbf{B} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{b} \end{bmatrix},$$

  where $\mathbf{M} \in \mathbb{R}^{m \times m}$ is invertible, $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{n \times m}$ are nonzero, and $\mathbf{b} \in \mathbb{R}^n$ is nonzero.

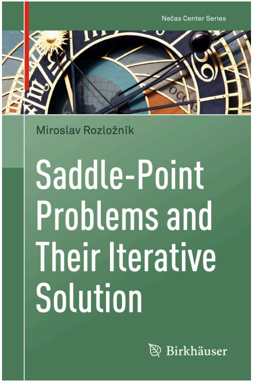
- Nonsymmetric partitioned linear systems of the form:

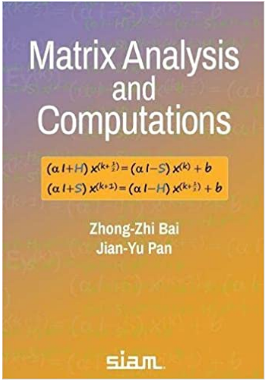
$$\begin{bmatrix} \lambda\mathbf{I} & \mathbf{A} \\ \mathbf{B} & \mu\mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} = \begin{bmatrix} \mathbf{b} \\ \mathbf{c} \end{bmatrix},$$

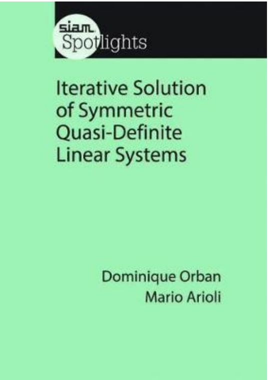
  where $\mathbf{A} \in \mathbb{R}^{m \times n}$, $\mathbf{B} \in \mathbb{R}^{n \times m}$, and $\mathbf{b} \in \mathbb{R}^m$, and $\mathbf{c} \in \mathbb{R}^n$. Note that λ and/or μ may be zero.

Review papers and books

- Michele Benzi, Gene H. Golub, and Jörg Liesen
  Numerical solution of saddle point problems.
  Acta Numerica (2005), pp. 1137.







Nonsymmetric saddle-point linear systems

- Nonsymmetric saddle-point linear systems of the form:

$$\begin{bmatrix} \mathbf{M} & \mathbf{A}^\top \\ \mathbf{B} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{b} \end{bmatrix},$$

  where $\mathbf{M} \in \mathbb{R}^{m\times m}$ is invertible.
- Monolithic methods: solving the system as a whole, for example, GMRES

  Segregated methods: exploiting the block structure, excluding the preconditioning stage, for example, SPMR, SPQMR, nsLSQR

R. Estrin and C. Greif. *SPMR: A family of saddle-point minimum residual solvers*. SISC, Vol. 40, No. 3 (2018)

K. Du, J.-J. Fan, and F. Wang. *nsLSQR: A quasi-minimum residual method for nonsymmetric saddle-point linear systems*. (2024)

Simultaneous bidiagonalization via $\mathbb{M}$-conjugacy

Algorithm Simultaneous bidiagonalization via **M**-conjugacy

Require: $\mathbf{M} \in \mathbb{R}^{m \times m}$, $\mathbf{A}$, $\mathbf{B} \in \mathbb{R}^{n \times m}$, and $\mathbf{b}$, $\mathbf{c} \in \mathbb{R}^n$

1: $\beta_1 \mathbf{v}_1 := \mathbf{c}$, $\delta_1 \mathbf{z}_1 := \mathbf{b}$

2: $\mathbf{u} = \mathbf{A}^\top \mathbf{v}_1$, $\mathbf{w} = \mathbf{M}^{-\top} \mathbf{B}^\top \mathbf{z}_1$

3: $\alpha_1 = |\mathbf{w}^\top \mathbf{u}|^{1/2}$, $\gamma_1 = \mathbf{w}^\top \mathbf{u} / \alpha_1$

4: $\mathbf{u}_1 = \mathbf{M}^{-1} \mathbf{u} / \alpha_1$, $\mathbf{w}_1 = \mathbf{w} / \gamma_1$

5: **for** $k = 1, 2, \ldots$ **do**

6: $\quad \beta_{k+1} \mathbf{v}_{k+1} := \mathbf{A} \mathbf{w}_k - \alpha_k \mathbf{v}_k$, $\delta_{k+1} \mathbf{z}_{k+1} := \mathbf{B} \mathbf{u}_k - \gamma_k \mathbf{z}_k$

7: $\quad \mathbf{u} = \mathbf{A}^\top \mathbf{v}_{k+1} - \beta_{k+1} \mathbf{M} \mathbf{u}_k$, $\mathbf{w} = \mathbf{M}^{-\top} \mathbf{B}^\top \mathbf{z}_{k+1} - \delta_{k+1} \mathbf{w}_k$

8: $\quad \alpha_{k+1} = |\mathbf{w}^\top \mathbf{u}|^{1/2}$, $\gamma_{k+1} = \mathbf{w}^\top \mathbf{u} / \alpha_{k+1}$

9: $\quad \mathbf{u}_{k+1} = \mathbf{M}^{-1} \mathbf{u} / \alpha_{k+1}$, $\mathbf{w}_{k+1} = \mathbf{w} / \gamma_{k+1}$

10: **end for**

Simultaneous bidiagonalization via $\mathbb{M}$-conjugacy

- Simultaneous bidiagonalization via $\mathbf{M}$-conjugacy:

$$
\begin{aligned}
&\mathbf{A}\mathbf{W}_k = \mathbf{V}_{k+1}\mathbf{C}_{k+1,k}, \quad \mathbf{A}^\top\mathbf{V}_{k+1} = \mathbf{M}\mathbf{U}_{k+1}\mathbf{C}_{k+1}^\top,\\
&\mathbf{B}\mathbf{U}_k = \mathbf{Z}_{k+1}\mathbf{F}_{k+1,k}, \quad \mathbf{B}^\top\mathbf{Z}_{k+1} = \mathbf{M}^\top\mathbf{W}_{k+1}\mathbf{F}_{k+1}^\top,\\
&\mathbf{W}_k^\top\mathbf{M}\mathbf{U}_k = \mathbf{V}_k^\top\mathbf{V}_k = \mathbf{Z}^\top\mathbf{Z}_k = \mathbf{I}_k,
\end{aligned}
$$

where

$$
\begin{aligned}
&\mathbf{U}_k = \begin{bmatrix}\mathbf{u}_1 & \cdots & \mathbf{u}_k\end{bmatrix}, \quad \mathbf{V}_k = \begin{bmatrix}\mathbf{v}_1 & \cdots & \mathbf{v}_k\end{bmatrix},\\
&\mathbf{W}_k = \begin{bmatrix}\mathbf{w}_1 & \cdots & \mathbf{w}_k\end{bmatrix}, \quad \mathbf{Z}_k = \begin{bmatrix}\mathbf{z}_1 & \cdots & \mathbf{z}_k\end{bmatrix},\\
&\mathbf{C}_k = \mathrm{bidiag}(\beta_i, \alpha_i), \quad \mathbf{C}_{k+1,k} = \begin{bmatrix}\mathbf{C}_k\\ \beta_{k+1}\mathbf{e}_k^\top\end{bmatrix},\\
&\mathbf{F}_k = \mathrm{bidiag}(\delta_i, \gamma_i), \quad \mathbf{F}_{k+1,k} = \begin{bmatrix}\mathbf{F}_k\\ \delta_{k+1}\mathbf{e}_k^\top\end{bmatrix}.
\end{aligned}
$$

SPMR-SC

- The kth SPMR-SC iterate is

$$\mathbf{x}_k = \mathbf{U}_k \widetilde{\mathbf{x}}_k, \quad \mathbf{y}_k = \mathbf{V}_k \widetilde{\mathbf{y}}_k,$$

where

$$\begin{bmatrix} \widetilde{\mathbf{x}}_k \\ \widetilde{\mathbf{y}}_k \end{bmatrix} = \underset{\widetilde{\mathbf{x}} \in \mathbb{R}^k,\ \widetilde{\mathbf{y}} \in \mathbb{R}^k}{\operatorname{argmin}} \left\| \begin{bmatrix} \mathbf{0} \\ \delta_1 \mathbf{e}_1 \end{bmatrix} - \begin{bmatrix} \mathbf{I}_k & \mathbf{C}_k^\top \\ \mathbf{F}_{k+1,k} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \widetilde{\mathbf{x}} \\ \widetilde{\mathbf{y}} \end{bmatrix} \right\|_2 .$$

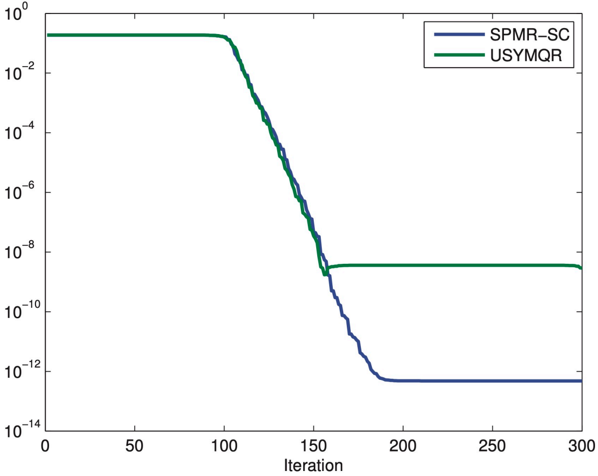
- Equivalent to USYMQR applied to the Schur complement system:

$$-\mathbf{S}\mathbf{y} = \mathbf{b}, \quad \mathbf{S} = \mathbf{B}\mathbf{M}^{-1}\mathbf{A}^\top.$$

- SPMR-SC can be more numerically stable than USYMQR when the Schur complement is ill-conditioned.

Example: an ill-conditioned Schur complement

$$\mathrm{cond}(\mathbf{A}) \approx 10^5,\ \mathrm{cond}(\mathbf{B}) \approx 10^5,\ \mathrm{cond}(\mathbf{S}) \approx 10^8$$



Simultaneous bidiagonalization via biorthogonality

Algorithm Simultaneous bidiagonalization via biorthogonality

Require: $\mathbf{M} \in \mathbb{R}^{m\times m}$, $\mathbf{A},\ \mathbf{B} \in \mathbb{R}^{n\times m}$, and $\mathbf{b},\ \mathbf{c} \in \mathbb{R}^{n}$

$\delta_1 = |\mathbf{c}^\top \mathbf{b}|^{1/2},\ \beta_1 = \mathbf{c}^\top \mathbf{b}/\beta_1$

$\mathbf{v}_1 = \mathbf{b}/\delta_1,\ \mathbf{z}_1 = \mathbf{c}/\beta_1$

$\mathbf{u} = \mathbf{A}^\top \mathbf{v}_1,\ \mathbf{w} = \mathbf{M}^{-\top}\mathbf{B}^\top \mathbf{z}_1$

$\alpha_1 = |\mathbf{w}^\top \mathbf{u}|^{1/2},\ \gamma_1 = \mathbf{w}^\top \mathbf{u}/\alpha_1$

$\mathbf{u}_1 = \mathbf{M}^{-1}\mathbf{u}/\alpha_1,\ \mathbf{w}_1 = \mathbf{w}/\gamma_1$

for $k = 1, 2, \ldots$ **do**

    $\mathbf{v} = \mathbf{B}\mathbf{u}_k - \gamma_k \mathbf{v}_k,\ \mathbf{z} = \mathbf{A}\mathbf{w}_k - \alpha_k \mathbf{z}_k$

    $\delta_{k+1} = |\mathbf{z}^\top \mathbf{v}|^{1/2},\ \beta_{k+1} = |\mathbf{z}^\top \mathbf{v}|/\delta_{k+1}$

    $\mathbf{v}_{k+1} = \mathbf{v}/\delta_{k+1},\ \mathbf{z}_{k+1} = \mathbf{z}/\beta_{k+1}$

    $\mathbf{u} = \mathbf{A}^\top \mathbf{v}_{k+1} - \beta_{k+1}\mathbf{M}\mathbf{u}_k,\ \mathbf{w} = \mathbf{M}^{-\top}\mathbf{B}^\top \mathbf{z}_{k+1} - \delta_{k+1}\mathbf{w}_k$

    $\alpha_{k+1} = |\mathbf{w}^\top \mathbf{u}|^{1/2},\ \gamma_{k+1} = \mathbf{w}^\top \mathbf{u}/\alpha_{k+1}$

    $\mathbf{u}_{k+1} = \mathbf{M}^{-1}\mathbf{u}/\alpha_{k+1},\ \mathbf{w}_{k+1} = \mathbf{w}/\gamma_{k+1}$

end for

Simultaneous bidiagonalization via biorthogonality

- Simultaneous bidiagonalization via biorthogonality:

$$\mathbf{A}\mathbf{W}_k = \mathbf{Z}_{k+1}\mathbf{C}_{k+1,k}, \quad \mathbf{A}^\top\mathbf{V}_{k+1} = \mathbf{M}\mathbf{U}_{k+1}\mathbf{C}_{k+1}^\top,$$
$$\mathbf{B}\mathbf{U}_k = \mathbf{V}_{k+1}\mathbf{F}_{k+1,k}, \quad \mathbf{B}^\top\mathbf{Z}_{k+1} = \mathbf{M}^\top\mathbf{U}_{k+1}\mathbf{F}_{k+1}^\top,$$
$$\mathbf{W}_k^\top\mathbf{M}\mathbf{U}_k = \mathbf{V}_k^\top\mathbf{Z}_k = \mathbf{I}_k,$$

where

$$\mathbf{U}_k = \begin{bmatrix}\mathbf{u}_1 & \cdots & \mathbf{u}_k\end{bmatrix}, \quad \mathbf{V}_k = \begin{bmatrix}\mathbf{v}_1 & \cdots & \mathbf{v}_k\end{bmatrix},$$
$$\mathbf{W}_k = \begin{bmatrix}\mathbf{w}_1 & \cdots & \mathbf{w}_k\end{bmatrix}, \quad \mathbf{Z}_k = \begin{bmatrix}\mathbf{z}_1 & \cdots & \mathbf{z}_k\end{bmatrix},$$
$$\mathbf{C}_k = \mathrm{bidiag}(\beta_i, \alpha_i), \quad \mathbf{C}_{k+1,k} = \begin{bmatrix}\mathbf{C}_k \\ \beta_{k+1}\mathbf{e}_k^\top\end{bmatrix},$$
$$\mathbf{F}_k = \mathrm{bidiag}(\delta_i, \gamma_i), \quad \mathbf{F}_{k+1,k} = \begin{bmatrix}\mathbf{F}_k \\ \delta_{k+1}\mathbf{e}_k^\top\end{bmatrix}.$$

SPQMR-SC

- The kth SPQMR-SC iterate is

$$\mathbf{x}_k = \mathbf{U}_k \widetilde{\mathbf{x}}_k, \quad \mathbf{y}_k = \mathbf{V}_k \widetilde{\mathbf{y}}_k,$$

  where

$$\begin{bmatrix} \widetilde{\mathbf{x}}_k \\ \widetilde{\mathbf{y}}_k \end{bmatrix} = \underset{\widetilde{\mathbf{x}} \in \mathbb{R}^k,\ \widetilde{\mathbf{y}} \in \mathbb{R}^k}{\operatorname{argmin}} \left\| \begin{bmatrix} \mathbf{0} \\ \delta_1 \mathbf{e}_1 \end{bmatrix} - \begin{bmatrix} \mathbf{I}_k & \mathbf{C}_k^\top \\ \mathbf{F}_{k+1,k} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \widetilde{\mathbf{x}} \\ \widetilde{\mathbf{y}} \end{bmatrix} \right\|_2 .$$

- Equivalent to QMR applied to the Schur complement system:

$$-\mathbf{S}\mathbf{y} = \mathbf{b}, \quad \mathbf{S} = \mathbf{B}\mathbf{M}^{-1}\mathbf{A}^\top.$$

- The convergence of SPMR-SC is monotonic, while the convergence of SPQMR-SC is erratic.

Bidiagonal-Hessenberg reduction

Algorithm Bidiagonal-Hessenberg reduction

Require: $\mathbf{M} \in \mathbb{R}^{m\times m}$, $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{n\times m}$, $\mathbf{b} \in \mathbb{R}^n$

$\mathbf{u}_1 = \mathbf{b}/\beta_1$ with $\beta_1 = \|\mathbf{b}\|_2$

$\mathbf{v} = \mathbf{A}^\top \mathbf{u}_1$, $\mathbf{v}_1 = \mathbf{M}^{-1}\mathbf{v}$, $\alpha_1 = \begin{cases} |\mathbf{v}_1^\top \mathbf{v}|^{1/2} & \text{if} \quad \mathbf{v}_1^\top \mathbf{v} \neq 0 \\ \|\mathbf{v}_1\|_2 & \text{if} \quad \mathbf{v}_1^\top \mathbf{v} = 0 \end{cases}$

$\mathbf{v}_1 = \mathbf{v}_1/\alpha_1$

for $k = 1, 2, \ldots$ **do**

    $\mathbf{u} = \mathbf{B}\mathbf{v}_k$

    for $i = 1, 2, \ldots, k$ **do**

        $h_{ik} = \mathbf{u}_i^\top \mathbf{u}$

        $\mathbf{u} = \mathbf{u} - h_{ik}\mathbf{u}_i$

    end for

    $\mathbf{u}_{k+1} = \mathbf{u}/\beta_{k+1}$ with $\beta_{k+1} = \|\mathbf{u}\|_2$

    $\mathbf{v} = \mathbf{A}^\top \mathbf{u}_{k+1} - \beta_{k+1}\mathbf{M}\mathbf{v}_k$, $\mathbf{v}_{k+1} = \mathbf{M}^{-1}\mathbf{v}$, $\alpha_{k+1} = \begin{cases} |\mathbf{v}_{k+1}^\top \mathbf{v}|^{1/2} & \text{if} \quad \mathbf{v}_{k+1}^\top \mathbf{v} \neq 0 \\ \|\mathbf{v}_{k+1}\|_2 & \text{if} \quad \mathbf{v}_{k+1}^\top \mathbf{v} = 0 \end{cases}$

    $\mathbf{v}_{k+1} = \mathbf{v}_{k+1}/\alpha_{k+1}$

end for

Bidiagonal-Hessenberg reduction

- Bidiagonal-Hessenberg reduction:

$$\mathbf{A}^\top \mathbf{U}_k = \mathbf{M}\mathbf{V}_k \mathbf{C}_k^\top, \qquad \mathbf{U}_{k+1}^\top \mathbf{U}_{k+1} = \mathbf{I}_{k+1},$$
$$\mathbf{B}\mathbf{V}_k = \mathbf{U}_{k+1}\mathbf{H}_{k+1,k} = \mathbf{U}_k \mathbf{H}_k + \beta_{k+1}\mathbf{u}_{k+1}\mathbf{e}_k^\top,$$

where

$$\mathbf{U}_k = \begin{bmatrix} \mathbf{u}_1 & \cdots & \mathbf{u}_k \end{bmatrix}, \quad \mathbf{V}_k = \begin{bmatrix} \mathbf{v}_1 & \cdots & \mathbf{v}_k \end{bmatrix},$$

$$\mathbf{C}_k = \begin{bmatrix} \alpha_1 & & & \\ \beta_2 & \alpha_2 & & \\ & \ddots & \ddots & \\ & & \beta_k & \alpha_k \end{bmatrix}, \; \mathbf{H}_{k+1,k} = \begin{bmatrix} h_{11} & \cdots & h_{1k} \\ h_{21} & \ddots & \vdots \\ & \ddots & h_{kk} \\ & & h_{k+1,k} \end{bmatrix}.$$

nsLSQR

- The kth nsLSQR iterate is

$$\begin{bmatrix} \mathbf{x}_k \\ \mathbf{y}_k \end{bmatrix} = \begin{bmatrix} \mathbf{V}_k \widetilde{\mathbf{x}}_k \\ \mathbf{U}_k \widetilde{\mathbf{y}}_k \end{bmatrix},$$

where $\widetilde{\mathbf{x}}_k$ and $\widetilde{\mathbf{y}}_k$ solve

$$\begin{bmatrix} \widetilde{\mathbf{x}}_k \\ \widetilde{\mathbf{y}}_k \end{bmatrix} = \underset{\widetilde{\mathbf{x}} \in \mathbb{R}^k,\ \widetilde{\mathbf{y}} \in \mathbb{R}^k}{\operatorname{argmin}} \left\| \begin{bmatrix} \mathbf{0} \\ \beta_1 \mathbf{e}_1 \end{bmatrix} - \begin{bmatrix} \mathbf{I}_k & \mathbf{C}_k^\top \\ \mathbf{H}_{k+1,k} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \widetilde{\mathbf{x}} \\ \widetilde{\mathbf{y}} \end{bmatrix} \right\|_2 .$$

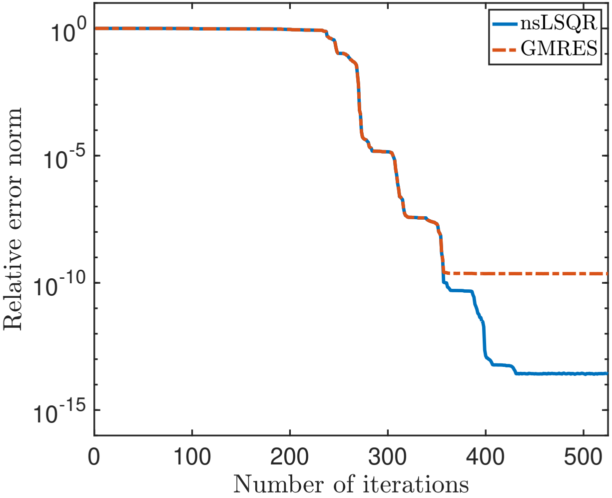
- Equivalent to GMRES applied to the Schur complement system:

$$-\mathbf{S}\mathbf{y} = \mathbf{b}, \quad \mathbf{S} = \mathbf{B}\mathbf{M}^{-1}\mathbf{A}^\top.$$

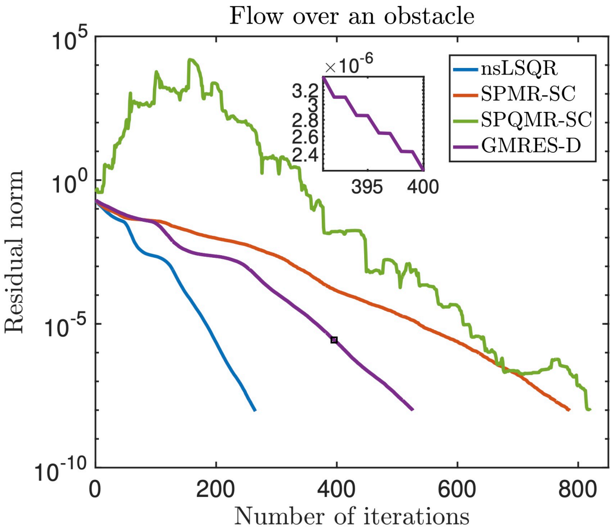
- nsLSQR can be more numerically stable than GMRES when the Schur complement is ill-conditioned.

Example: an ill-conditioned Schur complement

$$\text{cond}(\mathbf{A}) \approx 7 \times 10^3, \ \text{cond}(\mathbf{B}) \approx 7 \times 10^3, \ \text{cond}(\mathbf{S}) \approx 5 \times 10^7$$



Example: Flow over an obstacle (IFISS)



Nonsymmetric partitioned linear systems

- Nonsymmetric partitioned linear systems of the form
$$\begin{bmatrix} \lambda \mathbf{I} & \mathbf{A} \\ \mathbf{B} & \mu \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} = \begin{bmatrix} \mathbf{b} \\ \mathbf{c} \end{bmatrix}.$$
Note that λ and/or μ may be zero.
- Monolithic methods: solving the system as a whole, for example, GMRES

  Segregated methods: exploiting the block structure, excluding the preconditioning stage, for example, GPMR, GPBiLQ, GPBiCG, GPQMR

A. Montoison and D. Orban. *GPMR: An iterative method for unsymmetric partitioned linear systems*. SIMAX, Vol. 44, No. 1 (2023)

K. Du, J.-J. Fan, and F. Wang. *GPBiLQ and GPQMR: Two iterative methods for unsymmetric partitioned linear systems*. arXiv:2401.02608 (2024)

Simultaneous orthogonal Hessenberg reduction

Algorithm Simultaneous orthogonal Hessenberg reduction

Require: $\mathbf{A}$, $\mathbf{B}$, $\mathbf{b}$, $\mathbf{c}$, all nonzero

1: $\beta \mathbf{v}_1 := \mathbf{b}$, $\gamma \mathbf{u}_1 := \mathbf{c}$
2: **for** $k = 1, 2, \cdots$ **do**
3:     **for** $i = 1, 2, \cdots, k$ **do**
4:         $h_{ik} = \mathbf{v}_i^\top \mathbf{A} \mathbf{u}_k$
5:         $f_{ik} = \mathbf{u}_i^\top \mathbf{B} \mathbf{v}_k$
6:     **end for**
7:     $h_{k+1,k} \mathbf{v}_{k+1} = \mathbf{A} \mathbf{u}_k - \sum_{i=1}^{k} h_{ik} \mathbf{v}_i$
8:     $f_{k+1,k} \mathbf{u}_{k+1} = \mathbf{B} \mathbf{v}_k - \sum_{i=1}^{k} f_{ik} \mathbf{u}_i$
9: **end for**

Simultaneous orthogonal Hessenberg reduction

- Simultaneous orthogonal Hessenberg reduction

$$\begin{aligned}
\mathbf{A}\mathbf{U}_k &= \mathbf{V}_k\mathbf{H}_k + h_{k+1,k}\mathbf{v}_{k+1}\mathbf{e}_k^\top = \mathbf{V}_{k+1}\mathbf{H}_{k+1,k}, \\
\mathbf{B}\mathbf{V}_k &= \mathbf{U}_k\mathbf{F}_k + f_{k+1,k}\mathbf{u}_{k+1}\mathbf{e}_k^\top = \mathbf{U}_{k+1}\mathbf{F}_{k+1,k}, \\
\mathbf{V}_{k+1}^\top\mathbf{V}_{k+1} &= \mathbf{U}_{k+1}^\top\mathbf{U}_{k+1} = \mathbf{I}_{k+1},
\end{aligned}$$

where

$$\mathbf{U}_k = \begin{bmatrix}\mathbf{u}_1 & \mathbf{u}_2 & \cdots & \mathbf{u}_k\end{bmatrix}, \quad \mathbf{V}_k = \begin{bmatrix}\mathbf{v}_1 & \mathbf{v}_2 & \cdots & \mathbf{v}_k\end{bmatrix},$$

and

$$\mathbf{H}_{k+1,k} = \begin{bmatrix} h_{11} & \cdots & h_{1k} \\ h_{21} & \ddots & \vdots \\ & \ddots & h_{kk} \\ & & h_{k+1,k} \end{bmatrix}, \mathbf{F}_{k+1,k} = \begin{bmatrix} f_{11} & \cdots & f_{1k} \\ f_{21} & \ddots & \vdots \\ & \ddots & f_{kk} \\ & & f_{k+1,k} \end{bmatrix}.$$

GPMR

- The kth GPMR iterate is

$$\begin{bmatrix} \mathbf{x}_k \\ \mathbf{y}_k \end{bmatrix} = \underset{\mathbf{x} \in \operatorname{range}(\mathbf{V}_k),\ \mathbf{y} \in \operatorname{range}(\mathbf{U}_k)}{\operatorname{argmin}} \left\| \begin{bmatrix} \mathbf{b} \\ \mathbf{c} \end{bmatrix} - \begin{bmatrix} \lambda \mathbf{I} & \mathbf{A} \\ \mathbf{B} & \mu \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} \right\|_2 .$$

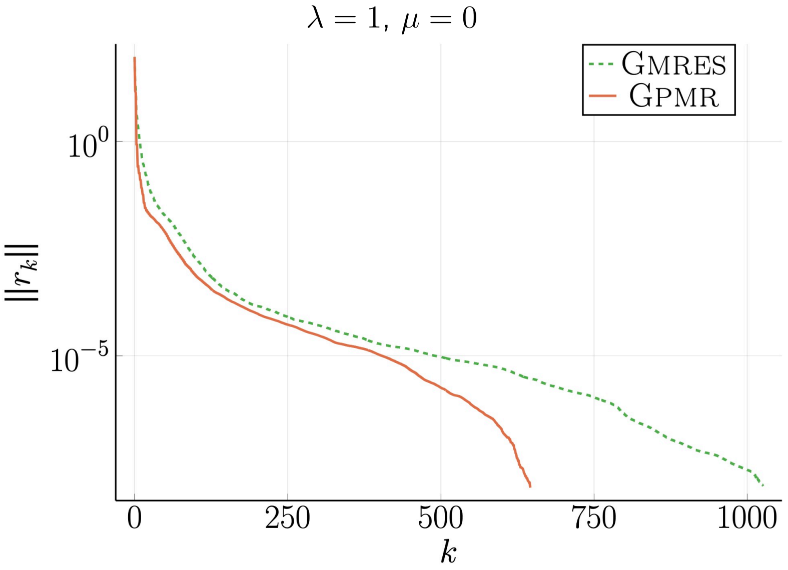
- Equivalent to Block-GMRES:

$$\begin{bmatrix} \lambda \mathbf{I} & \mathbf{A} \\ \mathbf{B} & \mu \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{x}^1 & \mathbf{x}^2 \\ \mathbf{y}^1 & \mathbf{y}^2 \end{bmatrix} = \begin{bmatrix} \mathbf{b} & \mathbf{0} \\ \mathbf{0} & \mathbf{c} \end{bmatrix} .$$

- GPMR terminates significantly earlier than GMRES on a residual-based stopping criterion with an improvement up to 50% in terms of number of iterations.

A. Montoison and D. Orban. *GPMR: An iterative method for unsymmetric partitioned linear systems*. SIMAX, Vol. 44, No. 1 (2023)

Example: $\mathbf{A} = \texttt{well1850}$, $\mathbf{B} = \texttt{illc1850}$



Randomized Gram–Schmidt process

Algorithm 2.1. RGS algorithm

Given: $n \times m$ matrix $\mathbf{W}$ and $k \times n$ matrix $\mathbf{\Theta}$, $m \le k \ll n$.
Output: $n \times m$ factor $\mathbf{Q}$ and $m \times m$ upper triangular factor $\mathbf{R}$.
for $i = 1 : m$ **do**

1. Sketch $\mathbf{w}_i$: $\mathbf{p}_i = \mathbf{\Theta}\mathbf{w}_i$. # macheps: u_{fine}
2. Solve $k \times (i-1)$ least-squares problem:
   $[\mathbf{R}]_{(1:i-1,i)} = \arg\min_{\mathbf{y}} \|\mathbf{S}_{i-1}\mathbf{y} - \mathbf{p}_i\|$. # macheps: u_{fine}
3. Compute projection of $\mathbf{w}_i$: $\mathbf{q}'_i = \mathbf{w}_i - \mathbf{Q}_{i-1}[\mathbf{R}]_{(1:i-1,i)}$. # macheps: u_{crs}
4. Sketch $\mathbf{q}'_i$: $\mathbf{s}'_i = \mathbf{\Theta}\mathbf{q}'_i$. # macheps: u_{fine}
5. Compute the sketched norm $r_{i,i} = \|\mathbf{s}'_i\|$. # macheps: u_{fine}
6. Scale vector $\mathbf{s}_i = \mathbf{s}'_i / r_{i,i}$. # macheps: u_{fine}
7. Scale vector $\mathbf{q}_i = \mathbf{q}'_i / r_{i,i}$. # macheps: u_{fine}

end for

8. (Optional) compute $\Delta_m = \|\mathbf{I}_{m\times m} - \mathbf{S}_m^{\mathrm{T}}\mathbf{S}_m\|_{\mathrm{F}}$ and $\tilde{\Delta}_m = \frac{\|\mathbf{P}_m - \mathbf{S}_m\mathbf{R}_m\|_{\mathrm{F}}}{\|\mathbf{P}_m\|_{\mathrm{F}}}$.
   Use Theorem 3.2 to certify the output. # macheps: u_{fine}

$\mathbf{W} = \mathbf{QR}$, $\mathbf{\Theta W} = \mathbf{\Theta QR}$, $(\mathbf{\Theta Q})^\top(\mathbf{\Theta Q}) = \mathbf{I}_m$, $\mathrm{cond}(\mathbf{Q})$ is small

O. Balabanov and L. Grigori. *Randomized Gram–Schmidt process with application to GMRES*. SISC, Vol. 44, No. 3 (2022)

Randomized GMRES (rGMRES) for $\mathbf{A}\mathbf{x} = \mathbf{b}$

Algorithm 4.1. RGS-Arnoldi algorithm

Given: $n \times n$ matrix $\mathbf{A}$, $n \times 1$ vector $\mathbf{b}$, $k \times n$ matrix $\mathbf{\Theta}$ with $k \ll n$, parameter m.
Output: $n \times m$ factor $\mathbf{Q}_m$ and $m \times m$ upper triangular factor $\mathbf{R}_m$.
1. Set $\mathbf{w}_1 = \mathbf{b}$.
2. Perform 1st iteration of Algorithm 2.1.

for $i = 2 : m$ **do**
  3. Compute $\mathbf{w}_i = \mathbf{A}\mathbf{q}_{i-1}$. # macheps: u_{fine}
  4. Perform ith iteration of Algorithm 2.1.

end for
5. (Optional) Compute Δ_m and $\tilde{\Delta}_m$.
   Use Proposition 4.1 to certify the output. # macheps: u_{fine}

- Let $\widehat{\mathbf{Q}}_m$ and $\widehat{\mathbf{H}}_{m+1,m}$ be the basis matrix and the Hessenberg matrix computed with Algorithm 4.1.
- The mth rGMRES iterate is $\mathbf{x}_m = \widehat{\mathbf{Q}}_m \mathbf{y}_m$ where

$$\mathbf{y}_m = \operatorname*{argmin}_{\mathbf{y}} \|\widehat{\mathbf{H}}_{m+1,m}\mathbf{y} - \widehat{r}_{11}\mathbf{e}_1\|_2.$$

sGMRES + k-truncated Arnoldi for $\mathbf{Ax} = \mathbf{f}$

- The solution $\mathbf{y}_\star$ of the overdetermined least-squares problem

$$\min_{\mathbf{y}} \|\mathbf{ABy} - \mathbf{f}\|_2$$

  yields an approximate solution $\mathbf{x_B} = \mathbf{By}_\star$ to $\mathbf{Ax} = \mathbf{f}$.
- The solution $\widehat{\mathbf{y}}$ of the sketched problem

$$\min_{\mathbf{y}} \|\mathbf{S}(\mathbf{ABy} - \mathbf{f})\|_2$$

  induces an approximate solution $\widehat{\mathbf{x}} = \mathbf{B}\widehat{\mathbf{y}}$ to $\mathbf{Ax} = \mathbf{f}$.
- sGMRES saves computational cost: $\widehat{\mathbf{x}} = \mathbf{B}\widehat{\mathbf{y}}$, columns of $\mathbf{B}$ form a basis of the Krylov subspace $\mathcal{K}_j(\mathbf{A}, \mathbf{f})$.
- k-truncated Arnoldi, skecth + precondition, for a good $\mathbf{B}$

Y. Nakatsukasa and J. A. Tropp. *Fast and accurate randomized algorithms for linear systems and eigenvalue problems*. SIMAX, Vol. 45, No. 2 (2024)

sGMRES + k-truncated Arnoldi for $\mathrm{Ax} = \mathrm{f}$

Algorithm 1.1. sGMRES + k-truncated Arnoldi.

Input: Matrix $\boldsymbol{A} \in \mathbb{C}^{n \times n}$, right-hand side $\boldsymbol{f} \in \mathbb{C}^n$, initial guess $\boldsymbol{x} \in \mathbb{C}^n$, basis dimension d, number k of vectors for truncated orthogonalization, stability tolerance $\texttt{tol} = O(u^{-1})$.

Output: Approximate solution $\hat{\boldsymbol{x}} \in \mathbb{C}^n$ to linear system (1.5) and estimated residual norm $\hat{r}_{\text{est}}$

1 **function** sGMRES
2 Draw subspace embedding $\boldsymbol{S} \in \mathbb{C}^{s \times n}$ with $s = 2(d+1)$ ▷ See subsection 2.4
3 Form residual and sketch: $\boldsymbol{r} = \boldsymbol{f} - \boldsymbol{A}\boldsymbol{x}$ and $\boldsymbol{g} = \boldsymbol{S}\boldsymbol{r}$
4 Normalize basis vector $\boldsymbol{b}_1 = \boldsymbol{r}/\|\boldsymbol{r}\|_2$ and apply matrix $\boldsymbol{m}_1 = \boldsymbol{A}\boldsymbol{b}_1$
5 **for** $j = 2, 3, 4, \ldots, d$ **do** ▷ See also subsection 5.2
6 Truncated Arnoldi: $\boldsymbol{w}_j = (\mathbf{I} - \boldsymbol{b}_{j-1}\boldsymbol{b}_{j-1}^* - \ldots - \boldsymbol{b}_{j-k}\boldsymbol{b}_{j-k}^*)\boldsymbol{m}_{j-1}$ ▷ $\boldsymbol{b}_{-i} = \boldsymbol{0}$ for $i \geq 0$
7 Normalize basis vector $\boldsymbol{b}_j = \boldsymbol{w}_j/\|\boldsymbol{w}_j\|_2$ and apply matrix $\boldsymbol{m}_j = \boldsymbol{A}\boldsymbol{b}_j$
8 Sketch reduced matrix: $\boldsymbol{C} = \boldsymbol{S}[\boldsymbol{m}_1, \ldots, \boldsymbol{m}_d]$
9 Thin QR factorization: $\boldsymbol{C} = \boldsymbol{U}\boldsymbol{T}$
10 **if** condition number $\kappa_2(\boldsymbol{T}) > \texttt{tol}$ **then** warning...
11 Either whiten $\boldsymbol{B} \leftarrow \boldsymbol{B}\boldsymbol{T}^{-1}$ or form new residual and restart ▷ See subsection 5.3
12 Solve least-squares problem: $\hat{\boldsymbol{y}} = \boldsymbol{T}^{-1}(\boldsymbol{U}^*\boldsymbol{g})$ ▷ See (3.7)
13 Residual estimate: $\hat{r}_{\text{est}} = \|(\mathbf{I} - \boldsymbol{U}\boldsymbol{U}^*)\boldsymbol{g}\|_2$ ▷ See (3.8)
14 Construct solution: $\hat{\boldsymbol{x}} = \boldsymbol{x} + [\boldsymbol{m}_1, \ldots, \boldsymbol{m}_j]\hat{\boldsymbol{y}}$

Implementation: In line 6, use double Gram–Schmidt for stability. In line 9, the QR factorization may require pivoting. In lines 11–12, apply $\boldsymbol{T}^{-1}$ via triangular substitution.

Summary

- We have presented nsLSQR for nonsymmetric saddle-point linear systems.
- nsLSQR is mathematically equivalent to GMRES applied to the corresponding Schur complement system, but may be numerically superior.
- nsLSQR usually is faster than SPMR-SC and SPQMR-SC in terms of the number of iterations, and if the iteration cost is dominated by the $\mathbf{M}$-solve rather than reorthogonalization, then nsLSQR should be the preferred method.
- The ideas of rGMRES and sGMRES can be used for GPMR and nsLSQR.
- Intelligent iterative methods for two-by-two block linear systems?

Thanks!