Finanziato dall'Unione europea NextGenerationEU
Ministero dell'Università e della Ricerca
Italiadomani PIANO NAZIONALE DI RIPRESA E RESILIENZA
ALMA MATER STUDIORUM UNIVERSITÀ DI BOLOGNA

Recent developments in multiterm matrix equations solvers

Valeria Simoncini

Dipartimento di Matematica
Alma Mater Studiorum - Università di Bologna
valeria.simoncini@unibo.it

Joint work with Martina Iannacito and Davide Palitta

Multiterm linear matrix equation

$$A_1 \boldsymbol{X} B_1 + A_2 \boldsymbol{X} B_2 + \ldots + A_\ell \boldsymbol{X} B_\ell = C$$

$A_i \in \mathbb{R}^{n \times n}$, $B_i \in \mathbb{R}^{m \times m}$, $\boldsymbol{X}$ unknown matrix

A sample of this equation on different problems:

- Stochastic PDEs
- PDEs on polygonal domains, IGA, spectral methods, etc
- Space-time PDEs
- All-at-once PDE-constrained optimization problem
- Bilinear control problems
- ....

A sample of computational strategies:

- Kronecker form and back on track
- Fixed point iterations (an "evergreen"...)
- Projection-type methods $\Rightarrow$ low rank approximation
- Optimization problems with fixed (low) rank approximation
- Ad-hoc problem-dependent procedures
- etc.

Multiterm linear matrix equation

$$A_1 \boldsymbol{X} B_1 + A_2 \boldsymbol{X} B_2 + \ldots + A_\ell \boldsymbol{X} B_\ell = C$$

$A_i \in \mathbb{R}^{n \times n}$, $B_i \in \mathbb{R}^{m \times m}$, $\boldsymbol{X}$ unknown matrix

A sample of this equation on different problems:

- Stochastic PDEs
- PDEs on polygonal domains, IGA, spectral methods, etc
- Space-time PDEs
- All-at-once PDE-constrained optimization problem
- Bilinear control problems
- ....

A sample of computational strategies:

- Kronecker form and back on track
- Fixed point iterations (an "evergreen"...)
- Projection-type methods $\Rightarrow$ low rank approximation
- Optimization problems with fixed (low) rank approximation
- Ad-hoc problem-dependent procedures
- etc.

Multiterm linear matrix equation

$$A_1 \boldsymbol{X} B_1 + A_2 \boldsymbol{X} B_2 + \ldots + A_\ell \boldsymbol{X} B_\ell = C$$

$A_i \in \mathbb{R}^{n \times n}$, $B_i \in \mathbb{R}^{m \times m}$, $\boldsymbol{X}$ unknown matrix

A sample of this equation on different problems:

- Stochastic PDEs
- PDEs on polygonal domains, IGA, spectral methods, etc
- Space-time PDEs
- All-at-once PDE-constrained optimization problem
- Bilinear control problems
- ....

A sample of computational strategies:

- Kronecker form and back on track
- Fixed point iterations (an "evergreen"...)
- Projection-type methods $\Rightarrow$ low rank approximation
- Optimization problems with fixed (low) rank approximation
- Ad-hoc problem-dependent procedures
- etc.

Application to dynamical systems: bilinear control problems

Deterministic bilinear dynamical system:

$$\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + \sum_{j=1}^{m} N_j \mathbf{x}(t) u_j(t) + B\mathbf{u}(t) \tag{1}$$

In this setting,

$$A\boldsymbol{X} + \boldsymbol{X}A^\top + \sum_{j=1}^{m} N_j \boldsymbol{X} N_j^\top + BB^\top = 0$$

and the matrix solution $\boldsymbol{X}$ is the controllability Gramian of the system (1)

Multiterm linear matrix equation. Classical device

$$A_1 \boldsymbol{X} B_1 + A_2 \boldsymbol{X} B_2 + \ldots + A_\ell \boldsymbol{X} B_\ell = C$$

Kronecker formulation[1]

$$\left(B_1^\top \otimes A_1 + \ldots + B_\ell^\top \otimes A_\ell\right) \boldsymbol{x} = c \qquad \Leftrightarrow \qquad \mathcal{A}\boldsymbol{x} = c$$

Iterative methods: matrix-matrix multiplications and rank truncation

(Benner, Bioli, Breiten, Bouhamidi, Chehab, Damm, Grasedyck, Jbilou, Kressner, Kuerschner, Matthies, Nagy, Onwunta, Palitta, Raydan, Robol, Stoll, Tobler, Wedderburn, Zander, ...)

Current very active area of research

Assume $\mathcal{A}$ is sym pos.def. (spd) $\Rightarrow$ CG

1

$$\text{Kronecker product}: \quad M \otimes P = \begin{bmatrix} m_{11}P & \cdots & m_{1n}P \\ \vdots & \ddots & \vdots \\ m_{n1}P & \cdots & m_{nn}P \end{bmatrix} \quad \text{and } \operatorname{vec}(AXB) = (B^\top \otimes A)\operatorname{vec}(X)$$

Multiterm linear matrix equation. Classical device

$$A_1 \boldsymbol{X} B_1 + A_2 \boldsymbol{X} B_2 + \ldots + A_\ell \boldsymbol{X} B_\ell = C$$

Kronecker formulation[1]

$$\left(B_1^\top \otimes A_1 + \ldots + B_\ell^\top \otimes A_\ell\right) \boldsymbol{x} = c \qquad \Leftrightarrow \qquad \mathcal{A}\boldsymbol{x} = c$$

Iterative methods: matrix-matrix multiplications and rank truncation

(Benner, Bioli, Breiten, Bouhamidi, Chehab, Damm, Grasedyck, Jbilou, Kressner, Kuerschner, Matthies, Nagy, Onwunta, Palitta, Raydan, Robol, Stoll, Tobler, Wedderburn, Zander, ...)

Current very active area of research

Assume $\mathcal{A}$ is sym pos.def. (spd) $\Rightarrow$ CG

[1] $\text{Kronecker product}: \; M \otimes P = \begin{bmatrix} m_{11}P & \cdots & m_{1n}P \\ \vdots & \ddots & \vdots \\ m_{n1}P & \cdots & m_{nn}P \end{bmatrix}$ and $\text{vec}(AXB) = (B^\top \otimes A)\text{vec}(X)$

CG matricization and truncation

⋆ **Matricization.** Typically,

$$x^{(k+1)} = x^{(k)} + \alpha_k p^{(k)} \in \mathbb{R}^{n^2} \qquad \Rightarrow \quad X^{(k+1)} = X^{(k)} + \alpha_k P^{(k)} \in \mathbb{R}^{n \times n}$$

⋆ **Truncation.** If $X^{(k)} = X_1^{(k)} (X_1^{(k)})^\top$ with $X_1^{(k)}$ low rank, and similarly for $P^{(k)}$, then

$$X^{(k+1)} = X_1^{(k)} (X_1^{(k)})^\top + \alpha_k P_1^{(k)} (P_1^{(k)})^\top$$

- $X^{(k+1)}$ low rank:

$$X^{(k+1)} = [X_1^{(k)}, \sqrt{\alpha_k} P_1^{(k)}] \, [X_1^{(k)}, \sqrt{\alpha_k} P_1^{(k)}]^\top \tag{2}$$

(but generally larger than at iteration k)

- Cure: Rank shrinking $[X_1^{(k)}, \sqrt{\alpha_k} P_1^{(k)}] \Rightarrow X_1^{(k+1)} \qquad X^{(k+1)} \approx X_1^{(k+1)} (X_1^{(k+1)})^\top$

Implementation: $\mathcal{T}(X^{(k+1)})$ acts on the QR-SVD of factor in (2)

Alternative truncation criteria:

♣ Fix lower threshold tolerance ♣ Fix maximum allowed rank

CG matricization and truncation

⋆ **Matricization.** Typically,

$$x^{(k+1)} = x^{(k)} + \alpha_k p^{(k)} \in \mathbb{R}^{n^2} \qquad \Rightarrow \quad X^{(k+1)} = X^{(k)} + \alpha_k P^{(k)} \in \mathbb{R}^{n \times n}$$

⋆ **Truncation.** If $X^{(k)} = X_1^{(k)} (X_1^{(k)})^\top$ with $X_1^{(k)}$ low rank, and similarly for $P^{(k)}$, then

$$X^{(k+1)} = X_1^{(k)} (X_1^{(k)})^\top + \alpha_k P_1^{(k)} (P_1^{(k)})^\top$$

- $X^{(k+1)}$ low rank:

$$X^{(k+1)} = [X_1^{(k)}, \sqrt{\alpha_k} P_1^{(k)}] \, [X_1^{(k)}, \sqrt{\alpha_k} P_1^{(k)}]^\top \tag{2}$$

(but generally larger than at iteration k)

- Cure: Rank shrinking $[X_1^{(k)}, \sqrt{\alpha_k} P_1^{(k)}] \Rightarrow X_1^{(k+1)} \qquad X^{(k+1)} \approx X_1^{(k+1)} (X_1^{(k+1)})^\top$

Implementation: $\mathcal{T}(X^{(k+1)})$ acts on the QR-SVD of factor in (2)

Alternative truncation criteria:

♣ Fix lower threshold tolerance ♣ Fix maximum allowed rank

CG matricization and truncation

⋆ **Matricization.** Typically,

$$x^{(k+1)} = x^{(k)} + \alpha_k p^{(k)} \in \mathbb{R}^{n^2} \qquad \Rightarrow \quad X^{(k+1)} = X^{(k)} + \alpha_k P^{(k)} \in \mathbb{R}^{n \times n}$$

⋆ **Truncation.** If $X^{(k)} = X_1^{(k)}(X_1^{(k)})^\top$ with $X_1^{(k)}$ low rank, and similarly for $P^{(k)}$, then

$$X^{(k+1)} = X_1^{(k)}(X_1^{(k)})^\top + \alpha_k P_1^{(k)}(P_1^{(k)})^\top$$

- $X^{(k+1)}$ low rank:

$$X^{(k+1)} = [X_1^{(k)}, \sqrt{\alpha_k} P_1^{(k)}]\, [X_1^{(k)}, \sqrt{\alpha_k} P_1^{(k)}]^\top \tag{2}$$

  (but generally larger than at iteration k)

- Cure: Rank shrinking $[X_1^{(k)}, \sqrt{\alpha_k} P_1^{(k)}] \Rightarrow X_1^{(k+1)} \qquad X^{(k+1)} \approx X_1^{(k+1)}(X_1^{(k+1)})^\top$

  Implementation: $\mathcal{T}(X^{(k+1)})$ acts on the QR-SVD of factor in (2)

Alternative truncation criteria:

♣ Fix lower threshold tolerance ♣ Fix maximum allowed rank

CG matricization and truncation

⋆ **Matricization.** Typically,

$$x^{(k+1)} = x^{(k)} + \alpha_k p^{(k)} \in \mathbb{R}^{n^2} \qquad \Rightarrow \quad X^{(k+1)} = X^{(k)} + \alpha_k P^{(k)} \in \mathbb{R}^{n \times n}$$

⋆ **Truncation.** If $X^{(k)} = X_1^{(k)}(X_1^{(k)})^\top$ with $X_1^{(k)}$ low rank, and similarly for $P^{(k)}$, then

$$X^{(k+1)} = X_1^{(k)}(X_1^{(k)})^\top + \alpha_k P_1^{(k)}(P_1^{(k)})^\top$$

- $X^{(k+1)}$ low rank:

$$X^{(k+1)} = [X_1^{(k)}, \sqrt{\alpha_k} P_1^{(k)}]\, [X_1^{(k)}, \sqrt{\alpha_k} P_1^{(k)}]^\top \tag{2}$$

(but generally larger than at iteration k)

- Cure: Rank shrinking $[X_1^{(k)}, \sqrt{\alpha_k} P_1^{(k)}] \Rightarrow X_1^{(k+1)}$ $\quad X^{(k+1)} \approx X_1^{(k+1)}(X_1^{(k+1)})^\top$

Implementation: $\mathcal{T}(X^{(k+1)})$ acts on the QR-SVD of factor in (2)

Alternative truncation criteria:

♣ Fix lower threshold tolerance ♣ Fix maximum allowed rank

CG matricization and truncation

⋆ **Matricization.** Typically,

$$x^{(k+1)} = x^{(k)} + \alpha_k p^{(k)} \in \mathbb{R}^{n^2} \qquad \Rightarrow \quad X^{(k+1)} = X^{(k)} + \alpha_k P^{(k)} \in \mathbb{R}^{n \times n}$$

⋆ **Truncation.** If $X^{(k)} = X_1^{(k)}(X_1^{(k)})^\top$ with $X_1^{(k)}$ low rank, and similarly for $P^{(k)}$, then

$$X^{(k+1)} = X_1^{(k)}(X_1^{(k)})^\top + \alpha_k P_1^{(k)}(P_1^{(k)})^\top$$

- $X^{(k+1)}$ low rank:

$$X^{(k+1)} = [X_1^{(k)}, \sqrt{\alpha_k} P_1^{(k)}]\ [X_1^{(k)}, \sqrt{\alpha_k} P_1^{(k)}]^\top \tag{2}$$

  (but generally larger than at iteration k)

- Cure: Rank shrinking $[X_1^{(k)}, \sqrt{\alpha_k} P_1^{(k)}] \Rightarrow X_1^{(k+1)}$ $\quad X^{(k+1)} \approx X_1^{(k+1)}(X_1^{(k+1)})^\top$

  Implementation: $\mathcal{T}(X^{(k+1)})$ acts on the QR-SVD of factor in (2)

Alternative truncation criteria:

♣ Fix lower threshold tolerance ♣ Fix maximum allowed rank

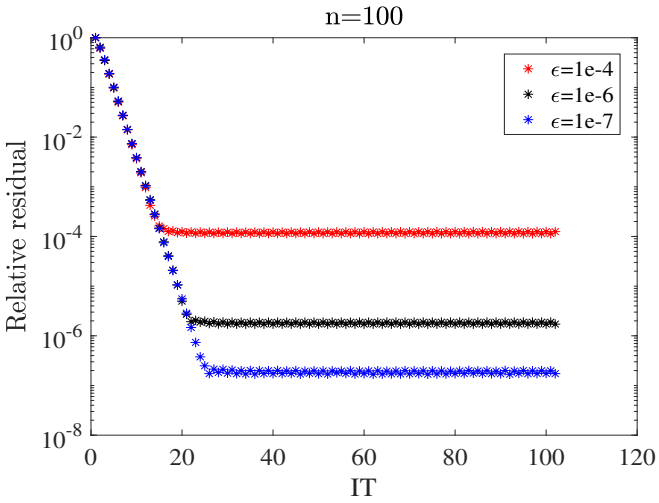
Truncated matrix-oriented CG (TCG) for Kronecker form

Input: $\mathcal{L}(\boldsymbol{X}) = A_1\boldsymbol{X}B_1 + A_2\boldsymbol{X}B_2 + \ldots + A_\ell\boldsymbol{X}B_\ell$, right-hand side $C \in \mathbb{R}^{n\times n}$ in low-rank format.
Truncation operator $\mathcal{T}$.
Output: Matrix $X \in \mathbb{R}^{n\times n}$ in low-rank format s.t. $||\mathcal{L}(X) - C||_F/||C||_F \leq tol$

1. $X_0 = 0$, $R_0 = C$, $P_0 = R_0$, $Q_0 = \mathcal{L}(P_0)$
2. $\xi_0 = \langle P_0, Q_0\rangle$, $k = 0$ $\qquad\qquad \langle X, Y\rangle = \mathrm{tr}(X^\top Y)$
3. While $||R_k||_F > tol$
4. $\quad \alpha_k = \langle R_k, P_k\rangle/\xi_k$
5. $\quad X_{k+1} = X_k + \alpha_k P_k,$ $\qquad X_{k+1} \leftarrow \mathcal{T}(X_{k+1})$
6. $\quad R_{k+1} = C - \mathcal{L}(X_{k+1}),$ $\qquad$ Optionally: $R_{k+1} \leftarrow \mathcal{T}(R_{k+1})$
7. $\quad \beta_k = -\langle R_{k+1}, Q_k\rangle/\xi_k$
8. $\quad P_{k+1} = R_{k+1} + \beta_k P_k,$ $\qquad P_{k+1} \leftarrow \mathcal{T}(P_{k+1})$
9. $\quad Q_{k+1} = \mathcal{L}(P_{k+1}),$ $\qquad$ Optionally: $Q_{k+1} \leftarrow \mathcal{T}(Q_{k+1})$
10. $\quad \xi_{k+1} = \langle P_{k+1}, Q_{k+1}\rangle$
11. $\quad k = k + 1$
12. end while

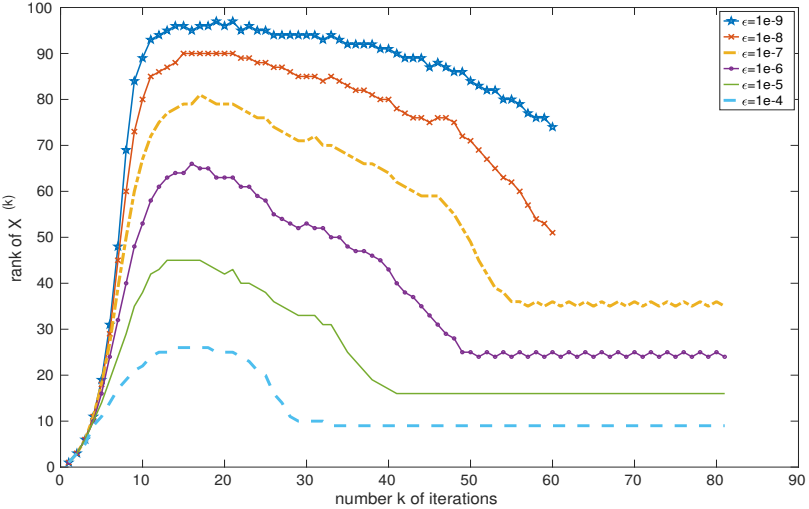
♣ Iterates kept in factored form! Kressner and Tobler, '11

Typical convergence behavior



(Hao, '20, personal comm.)

Typical iterate rank behavior



(Simoncini & Hao, '22)

Within the CG framework, can we do better?

Considerations:

1. At best, convergence as for Kronecker problem
2. Rank of iterates hard to control to maintain convergence
3. Coeffs α, β under exploited

$$\boldsymbol{p}_k = \mathrm{vec}(P_k), \boldsymbol{r}_k = \mathrm{vec}(R_k) \quad \Rightarrow \quad \{\boldsymbol{r}_0, \ldots, \boldsymbol{r}_k\}, \{\boldsymbol{p}_0, \ldots, \boldsymbol{p}_k\} \quad \text{orth prop}$$

Recalling CG basics: $\mathcal{A}\boldsymbol{x} = \boldsymbol{b}$

Problem: Minimize the convex function

$$\Phi(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T \mathcal{A}\boldsymbol{x} - \boldsymbol{b}^T \boldsymbol{x}$$

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k, \qquad \alpha_k \, s.t. \, \min_{\alpha} \Phi(\boldsymbol{x}_k + \alpha \boldsymbol{p}_k)$$

with residual and direction updates:

$$\boldsymbol{r}_{k+1} = \boldsymbol{r}_k - \mathcal{A}\boldsymbol{p}_k \alpha_k, \quad \boldsymbol{p}_{k+1} = \boldsymbol{r}_{k+1} + \boldsymbol{p}_k \beta_k.$$

Within the CG framework, can we do better?

Considerations:

1. At best, convergence as for Kronecker problem
2. Rank of iterates hard to control to maintain convergence
3. Coeffs α, β under exploited

$$\boldsymbol{p}_k = \mathrm{vec}(P_k), \boldsymbol{r}_k = \mathrm{vec}(R_k) \quad \Rightarrow \quad \{\boldsymbol{r}_0, \ldots, \boldsymbol{r}_k\}, \{\boldsymbol{p}_0, \ldots, \boldsymbol{p}_k\} \quad \text{orth prop}$$

Recalling CG basics: $\mathcal{A}\boldsymbol{x} = \boldsymbol{b}$

Problem: Minimize the convex function

$$\Phi(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T \mathcal{A}\boldsymbol{x} - \boldsymbol{b}^T \boldsymbol{x}$$

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k, \qquad \alpha_k \, s.t. \, \min_{\alpha} \Phi(\boldsymbol{x}_k + \alpha \boldsymbol{p}_k)$$

with residual and direction updates:

$$\boldsymbol{r}_{k+1} = \boldsymbol{r}_k - \mathcal{A}\boldsymbol{p}_k \alpha_k, \quad \boldsymbol{p}_{k+1} = \boldsymbol{r}_{k+1} + \boldsymbol{p}_k \beta_k.$$

The new **subspace**-CG

Warning: For the sake of the presentation, we assume a simplified form:

$$\mathcal{L}(\boldsymbol{X}) = \mathcal{L}(\boldsymbol{X})^T, \qquad \text{for any} \quad \boldsymbol{X} = \boldsymbol{X}^T$$

This assumption allows us to write a square $\boldsymbol{X}$ as $\boldsymbol{X} = XX^T$

In practice, the whole derivation holds for $\mathcal{A} = B_1 \otimes A_1 + \cdots + B_\ell \otimes A_\ell$ spd
(that is, $\mathcal{L}$ spd in the matrix inner product)
so that

$$\boldsymbol{X} \in \mathbb{R}^{n_l \times n_r}, \qquad \boldsymbol{X} = X^l (X^r)^T$$

The new **subspace**-CG

Warning: For the sake of the presentation, we assume a simplified form:

$$\mathcal{L}(\boldsymbol{X}) = \mathcal{L}(\boldsymbol{X})^T, \qquad \text{for any} \quad \boldsymbol{X} = \boldsymbol{X}^T$$

This assumption allows us to write a square $\boldsymbol{X}$ as $\boldsymbol{X} = XX^T$

In practice, the whole derivation holds for $\mathcal{A} = B_1 \otimes A_1 + \cdots + B_\ell \otimes A_\ell$ spd
(that is, $\mathcal{L}$ spd in the matrix inner product)
so that

$$\boldsymbol{X} \in \mathbb{R}^{n_l \times n_r}, \qquad \boldsymbol{X} = X^l (X^r)^T$$

The new **subspace** CG

We define $\Phi : \mathbb{R}^{n\times n} \to \mathbb{R}$,

$$\Phi(\boldsymbol{X}) = \frac{1}{2}\langle \boldsymbol{X}, \mathcal{L}(\boldsymbol{X})\rangle - \langle \boldsymbol{X}, \boldsymbol{C}\rangle$$

The new minimization problem: *Find* $\boldsymbol{X} \in \mathbb{R}^{n\times n}$ *such that*

$$\boldsymbol{X} = \arg \min_{\boldsymbol{X}\in\mathbb{R}^{n\times n}} \Phi(\boldsymbol{X})$$

with iteration

$$\boldsymbol{X}_{k+1} = \boldsymbol{X}_k + P_k \boldsymbol{\alpha}_k P_k^T$$

where $\boldsymbol{\alpha}_k \in \mathbb{R}^{s_k\times s_k}$ and $P_k \in \mathbb{R}^{n\times s_k}$

Residual and direction computation:

$$\boldsymbol{R}_{k+1} = \boldsymbol{R}_k - \mathcal{L}(P_k \boldsymbol{\alpha}_k P_k^T), \qquad \boldsymbol{P}_{k+1} = \boldsymbol{R}_{k+1} + P_k \boldsymbol{\beta}_k P_k^T,$$

where $\boldsymbol{P}_{k+1} = P_{k+1} P_{k+1}^T$

The new **subspace** CG

We define $\Phi : \mathbb{R}^{n\times n} \to \mathbb{R}$,

$$\Phi(\boldsymbol{X}) = \frac{1}{2}\langle \boldsymbol{X}, \mathcal{L}(\boldsymbol{X})\rangle - \langle \boldsymbol{X}, \boldsymbol{C}\rangle$$

The new minimization problem: *Find* $\boldsymbol{X} \in \mathbb{R}^{n\times n}$ *such that*

$$\boldsymbol{X} = \arg \min_{\boldsymbol{X}\in\mathbb{R}^{n\times n}} \Phi(\boldsymbol{X})$$

with iteration

$$\boldsymbol{X}_{k+1} = \boldsymbol{X}_k + P_k \boldsymbol{\alpha}_k P_k^T$$

where $\boldsymbol{\alpha}_k \in \mathbb{R}^{s_k\times s_k}$ and $P_k \in \mathbb{R}^{n\times s_k}$

Residual and direction computation:

$$\boldsymbol{R}_{k+1} = \boldsymbol{R}_k - \mathcal{L}(P_k \boldsymbol{\alpha}_k P_k^T), \qquad \boldsymbol{P}_{k+1} = \boldsymbol{R}_{k+1} + P_k \boldsymbol{\beta}_k P_k^T,$$

where $\boldsymbol{P}_{k+1} = P_{k+1} P_{k+1}^T$

Defining the coefficients. I

At the kth iteration:

1. Construct $\boldsymbol{\alpha}_k$ so that

$$\min_{\boldsymbol{\alpha} \in \mathbb{R}^{s_k \times s_k}} \Phi(\boldsymbol{X}_k + P_k \boldsymbol{\alpha} P_k^T)$$

2. Impose a descent direction requirement for $\boldsymbol{P}_k = P_k P_k^T$:

$$\langle \nabla \Phi(\boldsymbol{X}_k), \boldsymbol{P}_k \rangle < 0$$

3. Construct β_k so that the new direction $\boldsymbol{P}_{k+1}$ is $\mathcal{L}$-orthogonal with respect to the previous ones:

$$(P_k \otimes P_k)^T \mathrm{vec}(\mathcal{L}(\boldsymbol{P}_{k+1})) = 0$$

Defining the coefficients. II

Let $\widetilde{A}_k^{(i)} = P_k^T A_i P_k, \widetilde{B}_k^{(i)} = P_k^T B_i P_k,\ i = 1, \ldots, \ell$

- Construction of $\boldsymbol{\alpha}_k$:
  $\boldsymbol{\alpha}_k$ is the unique solution of

$$\widetilde{A}_k^{(1)} \boldsymbol{\alpha} \widetilde{B}_k^{(1)} + \ldots + \widetilde{A}_k^{(\ell)} \boldsymbol{\alpha} \widetilde{B}_k^{(\ell)} = P_k^T \boldsymbol{R}_k P_k$$

- Construction of $\boldsymbol{\beta}_k$:
  $\boldsymbol{\beta}_k$ is the unique solution of

$$\widetilde{A}_k^{(1)} \boldsymbol{\beta} \widetilde{B}_k^{(1)} + \ldots + \widetilde{A}_k^{(\ell)} \boldsymbol{\beta} \widetilde{B}_k^{(\ell)} = -P_k^T \mathcal{L}(P_k \boldsymbol{\beta}_k P_k^T) P_k$$

Defining the coefficients. II

Let $\widetilde{A}_k^{(i)} = P_k^T A_i P_k, \widetilde{B}_k^{(i)} = P_k^T B_i P_k,\ i = 1, \ldots, \ell$

- Construction of $\boldsymbol{\alpha}_k$:
  $\boldsymbol{\alpha}_k$ is the unique solution of

$$\widetilde{A}_k^{(1)} \boldsymbol{\alpha} \widetilde{B}_k^{(1)} + \ldots + \widetilde{A}_k^{(\ell)} \boldsymbol{\alpha} \widetilde{B}_k^{(\ell)} = P_k^T \boldsymbol{R}_k P_k$$

- Construction of $\boldsymbol{\beta}_k$:
  $\boldsymbol{\beta}_k$ is the unique solution of

$$\widetilde{A}_k^{(1)} \boldsymbol{\beta} \widetilde{B}_k^{(1)} + \ldots + \widetilde{A}_k^{(\ell)} \boldsymbol{\beta} \widetilde{B}_k^{(\ell)} = -P_k^T \mathcal{L}(P_k \boldsymbol{\beta}_k P_k^T) P_k$$

Making the idea practical

$$\begin{aligned} \boldsymbol{X}_{k+1} &= X_k \boldsymbol{\tau}_k X_k^T + P_k \boldsymbol{\alpha}_k P_k^T = [X_k, P_k] \boldsymbol{\tau}_{k+1} [X_k, P_k]^T \\ \boldsymbol{R}_{k+1} &= [R_0, \boldsymbol{A}_\star \bullet X_{k+1}] \boldsymbol{\rho}_{k+1} [R_0, \boldsymbol{B}_\star \bullet X_{k+1}]^T \end{aligned}$$

where $\boldsymbol{A}_\star \bullet R = [A_1 R, \ldots, A_\ell R]$

- All terms are kept in factored form
- The rank grows

$$\Rightarrow \qquad \text{Rank truncation}$$

- Computing $\boldsymbol{R}_{k+1}$ becomes too expensive (CPU time and memory)

$$\Rightarrow \qquad \text{Randomized range finder}$$

Given a target rank `maxrankR` and a Gaussian matrix $G^l \in \mathbb{R}^{n_B \times \texttt{maxrankR}}$:

$$\boldsymbol{R}_{k+1} G^l = C_1 (C_2^T G^l) - \sum_{i=1}^{\ell} A_i (X^l_{k+1} \boldsymbol{\tau}_{k+1} ((X^r_{k+1})^T (B_i G^l)))$$

(Analogously for $\boldsymbol{R}_{k+1}^T$)
Then proceed with a cheap evaluation of the reduced residual matrix

Making the idea practical

$$\begin{aligned} \boldsymbol{X}_{k+1} &= X_k \boldsymbol{\tau}_k X_k^T + P_k \boldsymbol{\alpha}_k P_k^T = [X_k, P_k] \boldsymbol{\tau}_{k+1} [X_k, P_k]^T \\ \boldsymbol{R}_{k+1} &= [R_0, \boldsymbol{A}_\star \bullet X_{k+1}] \boldsymbol{\rho}_{k+1} [R_0, \boldsymbol{B}_\star \bullet X_{k+1}]^T \end{aligned}$$

where $\boldsymbol{A}_\star \bullet R = [A_1 R, \ldots, A_\ell R]$

- All terms are kept in factored form
- The rank grows

$$\Rightarrow \qquad \text{Rank truncation}$$

- Computing $\boldsymbol{R}_{k+1}$ becomes too expensive (CPU time and memory)

$$\Rightarrow \qquad \text{Randomized range finder}$$

Given a target rank `maxrankR` and a Gaussian matrix $G^l \in \mathbb{R}^{n_B \times \texttt{maxrankR}}$:

$$\boldsymbol{R}_{k+1} G^l = C_1 (C_2^T G^l) - \sum_{i=1}^{\ell} A_i (X^l_{k+1} \boldsymbol{\tau}_{k+1} ((X^r_{k+1})^T (B_i G^l)))$$

(Analogously for $\boldsymbol{R}_{k+1}^T$)
Then proceed with a cheap evaluation of the reduced residual matrix

Making the idea practical

$$\begin{aligned} \boldsymbol{X}_{k+1} &= X_k \boldsymbol{\tau}_k X_k^T + P_k \boldsymbol{\alpha}_k P_k^T = [X_k, P_k] \boldsymbol{\tau}_{k+1} [X_k, P_k]^T \\ \boldsymbol{R}_{k+1} &= [R_0, \boldsymbol{A}_\star \bullet X_{k+1}] \boldsymbol{\rho}_{k+1} [R_0, \boldsymbol{B}_\star \bullet X_{k+1}]^T \end{aligned}$$

where $\boldsymbol{A}_\star \bullet R = [A_1 R, \ldots, A_\ell R]$

- All terms are kept in factored form
- The rank grows

$$\Rightarrow \qquad \text{Rank truncation}$$

- Computing $\boldsymbol{R}_{k+1}$ becomes too expensive (CPU time and memory)

$$\Rightarrow \qquad \text{Randomized range finder}$$

Given a target rank `maxrankR` and a Gaussian matrix $G^l \in \mathbb{R}^{n_B \times \texttt{maxrankR}}$:

$$\boldsymbol{R}_{k+1} G^l = C_1 (C_2^T G^l) - \sum_{i=1}^{\ell} A_i (X^l_{k+1} \boldsymbol{\tau}_{k+1} ((X^r_{k+1})^T (B_i G^l)))$$

(Analogously for $\boldsymbol{R}_{k+1}^T$)
Then proceed with a cheap evaluation of the reduced residual matrix

A computational experiment. I

Heat model problem: $A\boldsymbol{X} + \boldsymbol{X}A^\top + N\boldsymbol{X}N^\top + BB^\top = 0$

A: FD discr of 2D Laplace op in $(0,1)^2$
N: includes Robin b.c. $\boldsymbol{n} \cdot \nabla(x) = \delta u(x-1)$ on one side
(zero Dirichlet on the rest of boundary)
δ: parameter, $\delta \in \{0.5, 0.9\}$

Computational setting:
- Two different meshgrids: $n = 320^2, 500^2$
- Preconditioner: 8 LR-ADI iters for $\mathcal{L}_0 : C \to AX + XA^\top$
- Stopping tolerance: 10^{-6}
- Stopping criterion: Relative difference in subsequent approximate soln norms (true residual norm computed at termination)

Algorithms to be compared:
* SS-CG-determ: new method, residual matrix computed sequentially;
* SS-CG-rand'zed: new method, residual matrix computed using Randfinder
* TPCG: truncated matrix-oriented preconditioned CG (Kressner, Tobler, 2011, and others)
* SSS: Efficient fixed-point iteration $\Phi = \mathcal{L}_0^{-1} \otimes \mathcal{N}$, (Shank, Szyld, Simoncini, 2016)

A computational experiment. I

Heat model problem: $AX + XA^\top + NXN^\top + BB^\top = 0$

A: FD discr of 2D Laplace op in $(0,1)^2$
N: includes Robin b.c. $n \cdot \nabla(x) = \delta u(x-1)$ on one side
(zero Dirichlet on the rest of boundary)
δ: parameter, $\delta \in \{0.5, 0.9\}$

Computational setting:
- Two different meshgrids: $n = 320^2, 500^2$
- Preconditioner: 8 LR-ADI iters for $\mathcal{L}_0 : C \to AX + XA^\top$
- Stopping tolerance: 10^{-6}
- Stopping criterion: Relative difference in subsequent approximate soln norms (true residual norm computed at termination)

Algorithms to be compared:
* SS-CG-determ: new method, residual matrix computed sequentially;
* SS-CG-rand'zed: new method, residual matrix computed using Randfinder
* TPCG: truncated matrix-oriented preconditioned CG (Kressner, Tobler, 2011, and others)
* SSS: Efficient fixed-point iteration $\Phi = \mathcal{L}_0^{-1} \otimes \mathcal{N}$, (Shank, Szyld, Simoncini, 2016)

A computational experiment. I

Heat model problem: $A\boldsymbol{X} + \boldsymbol{X}A^\top + N\boldsymbol{X}N^\top + BB^\top = 0$

A: FD discr of 2D Laplace op in $(0,1)^2$
N: includes Robin b.c. $\boldsymbol{n} \cdot \nabla(x) = \delta u(x-1)$ on one side
(zero Dirichlet on the rest of boundary)
δ: parameter, $\delta \in \{0.5, 0.9\}$

Computational setting:
- Two different meshgrids: $n = 320^2, 500^2$
- Preconditioner: 8 LR-ADI iters for $\mathcal{L}_0 : C \to AX + XA^\top$
- Stopping tolerance: 10^{-6}
- Stopping criterion: Relative difference in subsequent approximate soln norms (true residual norm computed at termination)

Algorithms to be compared:
* SS-CG-determ: new method, residual matrix computed sequentially;
* SS-CG-rand'zed: new method, residual matrix computed using Randfinder
* TPCG: truncated matrix-oriented preconditioned CG (Kressner, Tobler, 2011, and others)
* SSS: Efficient fixed-point iteration $\Phi = \mathcal{L}_0^{-1} \otimes \mathcal{N}$, (Shank, Szyld, Simoncini, 2016)

A computational experiment. I. cont'd

Example	n_A	maxrank	SSS (iter/alloc/rank)	TPCG	SS–CG determ.	SS–CG rand'zed
HEAT1(0.5)	102400	20		61.35 (16)	– (100)	– (100)
		30		22.78 (4)	17.93 (3)	18.17 (3)
			6.15 (7/126/31)			
	250000	20		– (100)	– (100)	– (100)
		30		60.84 (4)	64.46 (4)	64.45 (4)
			18.35 (7/139/31)			
HEAT1(0.9)	102400	40		– (100)	– (100)	– (100)
		50		310.72 (26)	**58.11** (5)	58.52 (5)
		–	– (50/ /)			
	250000	50		2401.90 (93)	– (100)	– (100)
		60		936.39 (30)	**119.55** (4)	120.43 (4)
		–	– (50/ /)			

– no conv.

- Running time in seconds, and in parenthesis the number of iterations
- For SSS: no iterations / subspace total memory alloc for length n vecs / solution rank

A computational experiment. II

Parameterized diffusion equation (Biolietal, 2025)

$$-\nabla \cdot (k \nabla u) = 0 \qquad \text{in} \quad (0,1)^2$$

with homogeneous boundary conditions and semi-separable diffusion coefficient:

$$k(x,y) = \sum_{j=1}^{\ell_k} \delta_j k_{j,x}(x) k_{j,y}(y) = 1 + \sum_{j=1}^{\ell_k - 1} \frac{10^j}{j!} x^j y^j, \qquad \ell_k = 4$$

This gives

$$\sum_{j=1}^{\ell_k} \delta_j (A_{j,x} \boldsymbol{X} D_{j,y} + D_{j,y} \boldsymbol{X} A_{j,y}) = \boldsymbol{C}$$

with $\boldsymbol{C}$ rank-four nonsym matrix accounting for b.c. (total of $\ell = 8$ terms)

Algorithms to be compared:

* SS-CG-determ: new method, residual matrix computed sequentially;
* SS-CG-rand'zed: new method, residual matrix computed using Randfinder
* TPCG: truncated matrix-oriented preconditioned CG (Kressner, Tobler, 2011, and others)
* R-NLTCG: Riemannian, nonlinear CG (Bioli, Kressner, Robol, 2025)

A computational experiment. II

Parameterized diffusion equation (Biolietal, 2025)

$$-\nabla \cdot (k \nabla u) = 0 \qquad \text{in} \quad (0,1)^2$$

with homogeneous boundary conditions and semi-separable diffusion coefficient:

$$k(x,y) = \sum_{j=1}^{\ell_k} \delta_j k_{j,x}(x) k_{j,y}(y) = 1 + \sum_{j=1}^{\ell_k - 1} \frac{10^j}{j!} x^j y^j, \qquad \ell_k = 4$$

This gives

$$\sum_{j=1}^{\ell_k} \delta_j (A_{j,x} \boldsymbol{X} D_{j,y} + D_{j,y} \boldsymbol{X} A_{j,y}) = \boldsymbol{C}$$

with $\boldsymbol{C}$ rank-four nonsym matrix accounting for b.c. (total of $\ell = 8$ terms)

Algorithms to be compared:

* SS-CG-determ: new method, residual matrix computed sequentially;
* SS-CG-rand'zed: new method, residual matrix computed using Randfinder
* TPCG: truncated matrix-oriented preconditioned CG (Kressner, Tobler, 2011, and others)
* R-NLTCG: Riemannian, nonlinear CG (Bioli, Kressner, Robol, 2025)

A computational experiment. II

n type	Precond	`maxrank`	R-NLCG	TPCG	SS–CG determ.	SS–CG rand'zed
10000	$\mathcal{P}_1$	20	– (100)	– (100)	– (100)	– (100)
	$\mathcal{P}_1$	40	– (100)	– (100)	1.08 (5)	**0.92** (5)
	$\mathcal{P}_1$	60	– (100)	– (100)	2.47 (5)	**2.34** (5)
	$\mathcal{P}_2$	20	**11.25 (36)**	11.42 (38)	– (100)	– (100)
	$\mathcal{P}_2$	40	*42.97 (36)	**15.54** (33)	– (100)	– (100)
	$\mathcal{P}_2$	60	*98.62 (35)	32.39 (28)	9.59 (5)	**8.37** (5)
102400	$\mathcal{P}_1$	20	– (100)	– (100)	– (100)	– (100)
	$\mathcal{P}_1$	40	†	– (100)	18.17 (6)	**8.74** (6)
	$\mathcal{P}_1$	60	†	– (100)	23.50 (5)	**16.93** (5)
	$\mathcal{P}_2$	20	**183.44** (41)	– (100)	– (100)	– (100)
	$\mathcal{P}_2$	40	†	446.94 (47)	– (100)	– (100)
	$\mathcal{P}_2$	60	†	884.20 (26)	115.73 (3)	**101.91** (3)

– no conv. * Lower final residual norm than other methods † Out of Memory

Running time in seconds (# iter's)
Stopping tolerance $\texttt{tol} = 5 \cdot 10^{-6}$ True residual norm at termination

$\mathcal{P}_1$: one-term precond, cheap $\mathcal{P}_2$: two-term precond, expensive (fixed ADI iters)

Further considerations

What I have not told you:

- Orthogonality properties of residuals and directions
- Optimality and finite termination properties
- Preconditioning
- More experiments on a variety of application problems

Outlook:

- Experiments are very promising
- The idea can be generalized to other Krylov methods
- Tensor version under investigation

REFERENCE

Davide Palitta, Martina Iannacito, and V. Simoncini
A subspace-conjugate gradient method for linear matrix equations
pp. 1-25, Jan 2025. ArXiv 2501.02938

Happy Birthday Volker!

Dagstuhl Seminar
Theoretical and Computational Aspects of Matrix Algorithms
(Oct 12 – Oct 17, 2003)