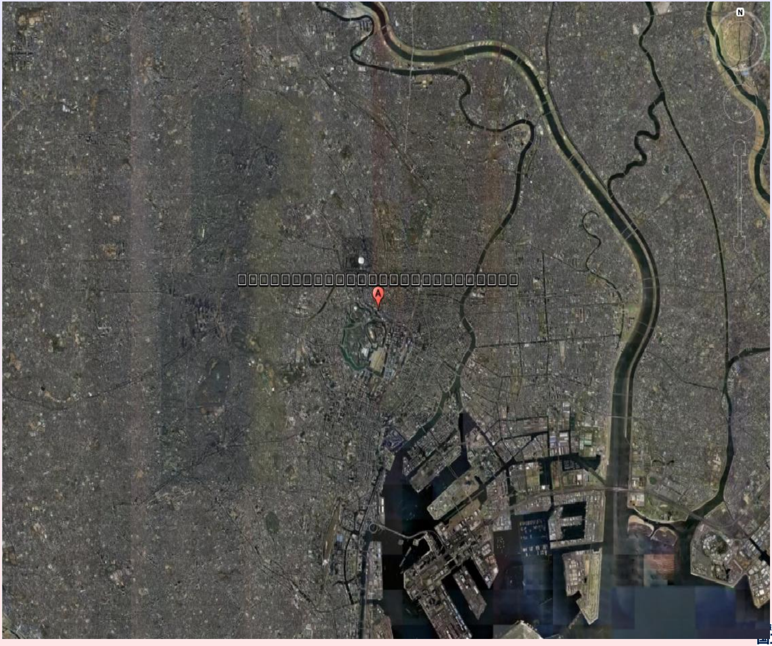
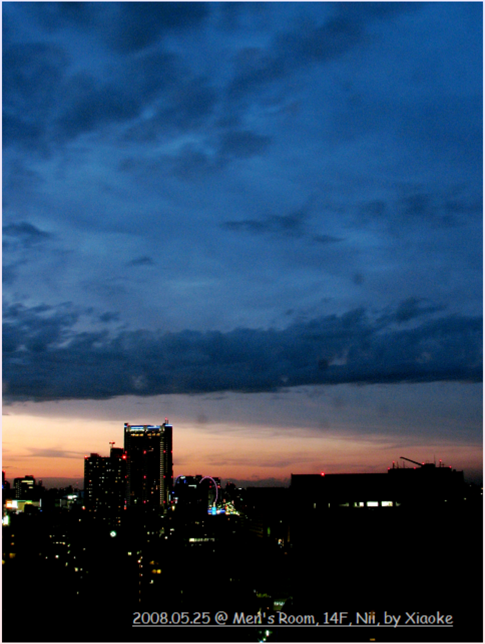


・システム研究機構
国立情報学研究所
National Institute of Informatics

情報・システム研究機構
国立情報学研究所
National Institute of Informatics


Photo by Syouka F8.0 1/40


2008.05.25 @ lobby, 14F, Nii, by Xiaoke


2008.05.25 @ Men's Room, 14F, Nii, by Xiaoke

Greville's Method for Preconditioning Least Squares Problems

Xiaoke Cui[1] Ken Hayami[1,2]

1. Department of Informatics
   The Graduate University for Advanced Studies (Sokendai), Japan
2. National Institute of Informatics, Tokyo, Japan

2008-07-08

Outline

Outline

Rank-One Update for Rectangular Matrices

Let $B \in \mathbb{R}^{m \times n}$, $c \in \mathbb{R}^m$ and $d \in \mathbb{R}^n$.
Given $B^\dagger \Rightarrow (B + c * d^T)^\dagger = \tilde{B}^\dagger =?$
(Campbell and Meyer, Generalized Inverses of Linear Transformations)

Given $A \in \mathbb{R}^{m \times n}$, then

$$A_0 = 0_{m \times n} \Rightarrow \quad A_i = [a_1, \ldots, a_i, 0 \ldots, 0] \quad \Rightarrow A = A_n$$
$$A_0^\dagger = 0_{n \times m} \Rightarrow \quad A_i^\dagger = (A_{i-1} + a_i e_i^T)^\dagger \quad \Rightarrow A^\dagger = A_n^\dagger =?$$

Greville's Method

Assume we already has $A_{i-1}^{\dagger}$, and $A_i = A_{i-1} + a_i e_i^T$,

- If $a_i \notin \mathcal{R}(A_{i-1})$, $k_i = A_{i-1}^{\dagger} a_i$, $u_i = a_i - A_{i-1} k_i$
  $A_i^{\dagger} = A_{i-1}^{\dagger} + (e_i - k_i) u_i^{\dagger}$

- If $a_i \in \mathcal{R}(A_{i-1})$, $f_i = 1 + \|k_i\|_2^2$
  $A_i^{\dagger} = A_{i-1}^{\dagger} + \frac{1}{f_i}(e_i - k_i) k_i^T A_{i-1}^{\dagger}$

$$u_i = a_i - A_{i-1} k_i = (I - A_{i-1} A_{i-1}^{\dagger}) a_i = P_{\mathcal{R}(A_{i-1})^{\perp}} a_i$$

This is called Greville's Method.

(T.N.E. Greville, Some applicatoins of the pseudoinverse of a matrix, SIAM Review, Vol.2, pp. 15-22, 1960)

Greville's Method

Assume we already has $A_{i-1}^{\dagger}$, and $A_i = A_{i-1} + a_i e_i^T$,

- If $a_i \notin \mathcal{R}(A_{i-1})$, $k_i = A_{i-1}^{\dagger} a_i$, $u_i = a_i - A_{i-1} k_i$
  $A_i^{\dagger} = A_{i-1}^{\dagger} + (e_i - k_i) u_i^{\dagger}$
- If $a_i \in \mathcal{R}(A_{i-1})$, $f_i = 1 + \|k_i\|_2^2$
  $A_i^{\dagger} = A_{i-1}^{\dagger} + \frac{1}{f_i}(e_i - k_i) k_i^T A_{i-1}^{\dagger}$

$$u_i = a_i - A_{i-1} k_i = (I - A_{i-1} A_{i-1}^{\dagger}) a_i = P_{\mathcal{R}(A_{i-1})^{\perp}} a_i$$

This is called Greville's Method.

(T.N.E. Greville, Some applicatoins of the pseudoinverse of a matrix, SIAM Review, Vol.2, pp. 15-22, 1960)

A Simple Algorithm

Simply, we can have the following algorithm for $A \in \mathbb{R}^{m \times n}$,

Algorithm: Greville's Method Algorithm

1. *set* $A_0^\dagger = 0$
2. *for* $i = 1 : n$
3. $\quad k_i = A_{i-1}^\dagger a_i$
4. $\quad u_i = a_i - A_{i-1} k_i$
5. $\quad$ *if* $\|u_i\| \neq 0$
6. $\quad\quad f_i = \|u_i\|_2^2$
7. $\quad\quad v_i = u_i$
8. $\quad$ *else*
9. $\quad\quad f_i = 1 + \|k_i\|_2^2$
10. $\quad\quad v_i = (A_{i-1}^\dagger)^T k_i$
11. $\quad$ *end if*
12. $\quad A_i^\dagger = A_{i-1}^\dagger + \frac{1}{f_i}(e_i - k_i) v_i^T$
13. *end for*
14. *get* $A^\dagger$.

Outline

A Factorization for $A^\dagger$

We update $A_i^\dagger$ in this way,

$$A_i^\dagger = A_{i-1}^\dagger + \frac{1}{f_i}(e_i - k_i)v_i^T, \quad i = 1, \ldots, n \tag{1}$$

and we have

$$A^\dagger = \sum_{i=1}^{n} \frac{1}{f_i}(e_i - k_i)v_i^T. \tag{2}$$

We denote

$$K = [k_1, \ldots, k_n]_{n\times n}, \tag{3}$$
$$V = [v_1, \ldots, v_n]_{m\times n}, \tag{4}$$
$$F = \text{Diag}\{f_1, \ldots f_n\}_{n\times n}. \tag{5}$$

$$A^\dagger = (I - K)F^{-1}V^T \tag{6}$$

A Factorization for $A^\dagger$

Theorem

Let $A \in \mathbb{R}^{m \times n}$. *Using the above notations,*

$$A^\dagger = (I - K)F^{-1}V^T. \tag{7}$$

- K *is a strict upper triangular matrix.*
- F *is a diagonal matrix with positive diagonal elements.*
- *If* A *is full column rank, then*

$$\begin{aligned} V &= A(I-K) & (8) \\ A^\dagger &= (I-K)F^{-1}(I-K)^T A^T. & (9) \end{aligned}$$

Proof.

$$\begin{aligned} k_i &= A_{i-1}^\dagger a_i = [a_1, \ldots, a_{i-1}, 0, \ldots, 0]^\dagger a_i \\ &= \begin{bmatrix} [a_1, \ldots, a_{i-1}]^\dagger \\ 0 \end{bmatrix} a_i = \begin{bmatrix} k_{i,1} & \cdots & k_{i,i-1} & 0 & \cdots & 0 \end{bmatrix}^T \end{aligned}$$

□

A Factorization for $A^\dagger$

Proof.

If A is full column rank,

$$\begin{aligned}
v_i &= u_i \\
&= a_i - A_{i-1}k_i \\
&= a_i - Ak_i \\
&= A(e_i - k_i) \\
V &= A(I - K) \\
A^\dagger &= (I-K)F^{-1}(I-K)^T A^T \\
(A^T A)^{-1} &= (I-K)F^{-1}(I-K)^T.
\end{aligned}$$

□

Outline

A Global Preconditioning Algorithm

We want a preconditioner, not $A^\dagger$.

Algorithm: Global Greville Preconditioning Algorithm

1. *set* $M_0 = 0$
2. *for* $i = 1 : n$
3. $\quad \tilde{k}_i = M_{i-1} a_i$ *perform numerical droppings*
4. $\quad \tilde{u}_i = a_i - A_{i-1} \tilde{k}_i$
5. $\quad$ *if* $\|\tilde{u}_i\| \neq 0$
6. $\quad\quad \tilde{f}_i = \|\tilde{u}_i\|_2^2$
7. $\quad\quad \tilde{v}_i = \tilde{u}_i$
8. $\quad$ *else*
9. $\quad\quad \tilde{f}_i = 1 + \|\tilde{k}_i\|_2^2$
10. $\quad\quad \tilde{v}_i = M_{i-1}^T \tilde{k}_i$
11. $\quad$ *end if*
12. $\quad M_i = M_{i-1} + \frac{1}{\tilde{f}_i}(e_i - \tilde{k}_i)\tilde{v}_i^T$ *perform numerical droppings*
13. *end for*
14. *get* $M \triangleq M_n \approx A^\dagger$.

When $\tilde{k}_i$ is sparse, the rank one update in Line 12 is very cheap.

A Factorization for M

For M from Global Greville Preconditioning Algorithm,

$$\begin{aligned} A^\dagger &= (I - K)F^{-1}V^T \\ M &= (I - \tilde{K})\tilde{F}^{-1}\tilde{V}^T, \end{aligned}$$

where

- $\tilde{K}$ is a strict upper triangular matrix,
- $\tilde{F}$ is a diagonal matrix with positive diagonal elements.

Outline

A Vector-wise Preconditioning Algorithm

If we do not want to form $A_i^\dagger$ or M_i explicitly, there are some places that we need to take care of.

Algorithm: Greville's Method Algorithm

1. *set* $A_0^\dagger = 0$
2. *for* $i = 1 : n$
3. $\quad k_i = A_{i-1}^\dagger a_i$
4. $\quad u_i = a_i - A_{i-1} k_i$
5. $\quad$ *if* $\|u_i\| \neq 0$
6. $\quad\quad f_i = \|u_i\|_2^2$
7. $\quad\quad v_i = u_i$
8. $\quad$ *else*
9. $\quad\quad f_i = 1 + \|k_i\|_2^2$
10. $\quad\quad v_i = (A_{i-1}^\dagger)^T k_i$
11. $\quad$ *end if*
12. $\quad A_i^\dagger = A_{i-1}^\dagger + \frac{1}{f_i}(e_i - k_i) v_i^T$
13. *end for*
14. *get* $A^\dagger$.

A Vector-wise Preconditioning Algorithm

$$A^\dagger = \sum_{i=1}^{n} \frac{1}{f_i}(e_i - k_i)v_i^T \Rightarrow A_i^\dagger = \sum_{p=1}^{i} \frac{1}{f_p}(e_p - k_p)v_p^T.$$

Therefore, when $\|u_i\| = 0$,

$$\begin{aligned} v_i &= (A_{i-1}^\dagger)^T k_i \\ &= \left(\sum_{p=1}^{i-1} \frac{1}{f_p}(e_p - k_p)v_p^T\right)^T k_i \\ &= \sum_{p=1}^{i-1} \frac{1}{f_p} v_p (e_p - k_p)^T k_i \end{aligned}$$

A Vector-wise Preconditioning Algorithm

Algorithm: Greville's Method Algorithm

1. *set* $A_0^{\dagger} = 0$
2. *for* $i = 1 : n$
3. $\quad k_i = A_{i-1}^{\dagger} a_i$
4. $\quad u_i = a_i - A_{i-1} k_i$
5. $\quad$ *if* $\|u_i\| \neq 0$
6. $\quad\quad f_i = \|u_i\|_2^2$
7. $\quad\quad v_i = u_i$
8. $\quad$ *else*
9. $\quad\quad f_i = 1 + \|k_i\|_2^2$
10. $\quad\quad v_i = (A_{i-1}^{\dagger})^T k_i$
11. $\quad$ *end if*
12. $\quad A_i^{\dagger} = A_{i-1}^{\dagger} + \frac{1}{f_i}(e_i - k_i) v_i^T$
13. *end for*
14. *get* $A^{\dagger}$.

A Vector-wise Preconditioning Algorithm

For k_i,

$$\begin{aligned} k_i &= A_{i-1}^{\dagger} a_i \\ &= (A_{i-2}^{\dagger} + (e_{i-1} - k_{i-1}) \frac{1}{f_{i-1}} v_{i-1}^T) a_i \\ &= A_{i-2}^{\dagger} a_i + (e_{i-1} - k_{i-1}) \frac{1}{f_{i-1}} v_{i-1}^T a_i \end{aligned}$$

Hence, we have a pyramid,

$$k_n = A_{n-1}^{\dagger} a_n$$

$$\nearrow \qquad\qquad\qquad \nwarrow$$

$$A_{n-2}^{\dagger} a_n \qquad\qquad k_{n-1} = A_{n-2}^{\dagger} a_{n-1}$$

$$\nearrow \quad \nwarrow \qquad\qquad \nearrow \quad \nwarrow$$

$$A_{n-3}^{\dagger} a_n \quad k_{n-2} = A_{n-3}^{\dagger} a_{n-2} \quad A_{n-3}^{\dagger} a_{n-1} \quad k_{n-2} = A_{n-3}^{\dagger} a_{n-2}$$

$$\cdots\cdots\ .$$

We need to compute $k_{i,j} \triangleq A_i^{\dagger} a_j, \ j = i+1, \ldots, n$.

A Vector-wise Preconditioning Algorithm

Algorithm: Vector-wise Greville's Method Algorithm

set $K = 0_{n\times n}$
for $i = 1 : n$
    $u_i = a_i - A_{i-1}k_i$
    if $\|u_i\| \neq 0$
        $f_i = \|u_i\|_2^2$
        $v_i = u_i$
    else
        $f_i = \|k_i\|_2^2 + 1$
        $v_i = \sum_{p=1}^{i-1} \frac{1}{f_p} v_p (e_p - k_p)^T k_i$
    end if
    for $j = i+1, \ldots, n$
        $k_j = k_j + \frac{v_i^T a_j}{f_i}(e_i - k_i)$
    end for
end for
$K = [k_1, \ldots, k_n]$, $F = \mathrm{Diag}\{f_1, \ldots, f_n\}$, $V = [v_1, \ldots, v_n]$.

A Vector-wise Preconditioning Algorithm

Algorithm: Vector-wise Greville Preconditioning Algorithm

set $\tilde{K} = 0_{n \times n}$
for $i = 1 : n$
    $u = a_i - A_{i-1}\tilde{k}_i$
    if $\|u_i\| \neq 0$
        $\tilde{f}_i = \|u_i\|_2^2$
        $\tilde{v}_i = u_i$
    else
        $\tilde{f}_i = \|\tilde{k}_i\|_2^2 + 1$
        $\tilde{v}_i = \sum_{p=1}^{i-1} \frac{1}{\tilde{f}_p} \tilde{v}_p (e_p - \tilde{k}_p)^T \tilde{k}_i$
    end if
    for $j = i+1, \ldots, n$
        $\tilde{k}_j = \tilde{k}_j + \frac{\tilde{v}_i^T a_j}{\tilde{f}_i}(e_i - \tilde{k}_i)$, *perform numerical droppings on* $\tilde{k}_j$
    end for
end for
$\tilde{K} = \left[\tilde{k}_1, \ldots, \tilde{k}_n\right]$, $\tilde{F} = \mathrm{Diag}\left\{\tilde{f}_1, \ldots, \tilde{f}_n\right\}$, $\tilde{V} = [\tilde{v}_1, \ldots, \tilde{v}_n]$.

Outline

Greville Preconditioning Algorithm and RIF

For the above algorithm, we consider a special case: when A is full column rank.

Algorithm: Vector-wise Grev. Pre.

1. *set* $\tilde{K} = 0_{n \times n}$
2. *for* $i = 1 : n$
3. $\quad \tilde{u}_i = a_i - A_{i-1}\tilde{k}_i$
4. $\quad \tilde{f}_i = \|\tilde{u}\|_2^2$
5. $\quad$ *for* $j = i+1, \ldots, n$
6. $\quad\quad \tilde{k}_j = \tilde{k}_j + \frac{\tilde{u}_i^T a_j}{\tilde{f}_i}(e_i - \tilde{k}_i)$
7. $\quad\quad$ *perform numerical droppings on* $\tilde{k}_j$
8. $\quad$ *end for*
9. *end for*
10. *get* $\tilde{K}$, $\tilde{F}$ *and* $\tilde{U} = A(I - \tilde{K})$.

国立情報学研究所 National Institute of Informatics

Greville Preconditioning Algorithm and RIF

For the above algorithm, we consider a special case: when A is full column rank.

Algorithm: Vector-wise Grev. Pre.

1. *set* $\tilde{K} = 0_{n \times n}$
2. *for* $i = 1 : n$
3. $\quad \tilde{u}_i = a_i - A_{i-1}\tilde{k}_i \qquad \Rightarrow \tilde{u}_i = A(e_i - \tilde{k}_i)$
4. $\quad \tilde{f}_i = \|\tilde{u}\|_2^2$
5. $\quad$ *for* $j = i+1, \ldots, n$
6. $\quad\quad \tilde{k}_j = \tilde{k}_j + \frac{\tilde{u}_i^T a_j}{\tilde{f}_i}(e_i - \tilde{k}_i) \Rightarrow e_j - \tilde{k}_j = e_j - \tilde{k}_j - \frac{\tilde{u}_i^T a_j}{\tilde{f}_i}(e_i - \tilde{k}_i)$
7. $\quad\quad$ *perform numerical droppings on* $\tilde{k}_j$
8. $\quad$ *end for*
9. *end for*
10. *get* $\tilde{K}$, $\tilde{F}$ *and* $\tilde{U} = A(I - \tilde{K})$.

Denote $e_j - \tilde{k}_j$ as z_j.

Greville Preconditioning Algorithm and RIF

For the above algorithm, we consider a special case: when A is full column rank.

Algorithm: Vector-wise Grev. Pre.

1. *set* $\tilde{K} = 0_{n\times n}$
2. *for* $i = 1 : n$
3. $\quad \tilde{u}_i = a_i - A_{i-1}\tilde{k}_i \qquad \Rightarrow \tilde{u}_i = A(e_i - \tilde{k}_i)$
4. $\quad \tilde{f}_i = \|\tilde{u}\|_2^2$
5. $\quad$ *for* $j = i+1, \ldots, n$
6. $\qquad \tilde{k}_j = \tilde{k}_j + \frac{\tilde{u}_i^T a_j}{\tilde{f}_i}(e_i - \tilde{k}_i) \Rightarrow e_j - \tilde{k}_j = e_j - \tilde{k}_j - \frac{\tilde{u}_i^T a_j}{\tilde{f}_i}(e_i - \tilde{k}_i)$
7. $\qquad$ *perform numerical droppings on* $\tilde{k}_j$
8. $\quad$ *end for*
9. *end for*
10. *get* $\tilde{K}$, $\tilde{F}$ *and* $\tilde{U} = A(I - \tilde{K})$.

Denote $e_j - \tilde{k}_j$ as z_j.

Greville Preconditioning Algorithm and RIF

We consider a special case: **when A is full column rank**.

Algorithm: Vector-wise Grev. Pre.

1. *set* $\tilde{Z} = I_{n\times n}$
2. *for* $i = 1 : n$
3. $\quad \tilde{u}_i = Az_i$
4. $\quad \tilde{f}_i = \|\tilde{u}\|_2^2$
5. $\quad$ *for* $j = i+1, \ldots, n$
6. $\quad\quad \theta_{i,j} = \frac{\tilde{u}_i^T a_j}{\tilde{f}_i} = \frac{(z_i, e_j)_{A^TA}}{(z_i, z_i)_{A^TA}}$
7. $\quad\quad z_j = z_j - \theta_{i,j} z_i$
8. $\quad\quad$ *perform droppings on* z_j
9. $\quad$ *end for*
10. *end for*
11. *get* $\tilde{Z}$, $\tilde{F}$ *and* $\tilde{U} = A(I - \tilde{K})$.

$$z_j = e_j - \tilde{k}_j.$$

Theorem

When A is full rank,

Grev. Pre. Algorithm

$\Updownarrow$

RIF

Greville Preconditioning Algorithm and RIF

We consider a special case: **when A is full column rank**.

Algorithm: Vector-wise Grev. Pre.

1. *set* $\tilde{Z} = I_{n\times n}$
2. *for* $i = 1 : n$
3. $\quad \tilde{u}_i = Az_i$
4. $\quad \tilde{f}_i = \|\tilde{u}\|_2^2$
5. $\quad$ *for* $j = i+1, \ldots, n$
6. $\qquad \theta_{i,j} = \frac{\tilde{u}_i^T a_j}{\tilde{f}_i} = \frac{(z_i, e_j)_{A^TA}}{(z_i, z_i)_{A^TA}}$
7. $\qquad z_j = z_j - \theta_{i,j} z_i$
8. $\qquad$ *perform droppings on* z_j
9. $\quad$ *end for*
10. *end for*
11. *get* $\tilde{Z}$, $\tilde{F}$ *and* $\tilde{U} = A(I - \tilde{K})$.

$$z_j = e_j - \tilde{k}_j.$$

Theorem

When A is full rank,

Grev. Pre. Algorithm

$\Updownarrow$

RIF

Outline

Square Case

For square problems,

$$Ax = b, \qquad A \in \mathbb{R}^{n \times n}, b \in \mathbb{R}^n. \tag{10}$$

- **construct a matrix** $M \approx A$

$$\begin{aligned} \text{left preconditioning} \quad & M^{-1}Ax = M^{-1}b \\ \text{right preconditioning} \quad & AM^{-1}y = b. \end{aligned}$$

- **construct a matrix** $M \approx A^{-1}$

$$\begin{aligned} \text{left preconditioning} \quad & MAx = Mb \\ \text{right preconditioning} \quad & AMy = b. \end{aligned}$$

Rectangular Case

For least squares problems,

$$\min_{x \in R^n} \|b - Ax\|_2 \qquad A \in R^{m\times n} \; b \in R^m$$

precondition it $\Downarrow$

$$\min_{x \in R^n} \|Mb - MAx\|_2 \qquad A \in R^{m\times n}, \; b \in R^m,$$

and then applying a Krylov subspace solver to solve preconditioned problems.

We hope,

- $\|I - MA\|$ is small, **or**
- $M \approx A^\dagger$, which is the Moore-Penrose Inverse of A.

$$\min_{x\in\mathbb{R}^n}\|b - Ax\|_2 \Leftrightarrow \min_{x\in\mathbb{R}^n}\|Mb - MAx\|_2?$$

Lemma

$$\|b - Ax^T\|_2 = \min_{x\in\mathbb{R}^n}\|b - Ax\|_2$$

and

$$\|Mb - MAx^T\|_2 = \min_{x\in\mathbb{R}^n}\|Mb - MAx\|_2$$

are equivalent for all $b \in \mathbb{R}^m$, if and only if $\mathcal{R}(A) = \mathcal{R}(M^T)$.

(K. Hayami, J-F Yin, and T. Ito, GMRES methods for least squares problem, National Institute of Informatics Technical Report, NII-2007-09E, 2007.)

$\min_{x\in\mathbb{R}^n}\|b - Ax\|_2 \Leftrightarrow \min_{x\in\mathbb{R}^n}\|Mb - MAx\|_2$?

For full rank case,

$$\begin{aligned} M &= (I - \tilde{K})\tilde{F}^{-1}(I - \tilde{K})^T A^T \\ &= CA^T \\ \mathcal{R}(M^T) &= \mathcal{R}(A), \end{aligned}$$

where C is a nonsingular matrix.

Theorem

When A is full column rank, we have $\mathcal{R}(M^T) = \mathcal{R}(A)$, hence

$$\min_{x\in\mathbb{R}^n}\|b - Ax\|_2 \Leftrightarrow \min_{x\in\mathbb{R}^n}\|Mb - MAx\|_2 \tag{11}$$

$\min_{x \in \mathbb{R}^n} \|b - Ax\|_2 \Leftrightarrow \min_{x \in \mathbb{R}^n} \|Mb - MAx\|_2$?

For full rank case,

$$\begin{aligned} M &= (I - \tilde{K})\tilde{F}^{-1}(I - \tilde{K})^T A^T \\ &= CA^T \\ \mathcal{R}(M^T) &= \mathcal{R}(A), \end{aligned}$$

where C is a nonsingular matrix.

Theorem

When A is full column rank, we have $\mathcal{R}(M^T) = \mathcal{R}(A)$, hence

$$\min_{x \in \mathbb{R}^n} \|b - Ax\|_2 \Leftrightarrow \min_{x \in \mathbb{R}^n} \|Mb - MAx\|_2 \tag{11}$$

$\min_{x \in \mathbb{R}^n} \|b - Ax\|_2 \Leftrightarrow \min_{x \in \mathbb{R}^n} \|Mb - MAx\|_2$?

For general case, we update M_i by,

$$M_i = M_{i-1} + \frac{1}{f_i}(e_i - \tilde{k}_i)\tilde{v}_i^T, \tag{12}$$

from which we can see every row of M_i is a linear combination of $\tilde{v}_i^T$. Thus,

$$\mathcal{R}(M_i^T) = \text{span}\{\tilde{v}_1, \ldots, \tilde{v}_i\}. \tag{13}$$

Notice

$$\tilde{v}_i = \begin{cases} A(e_i - \tilde{k}_i) \in \mathcal{R}(A_i) & \text{if } a_i \notin \mathcal{R}(A_{i-1}) \\ (A_{i-1}^\dagger)^T k_i \in \mathcal{R}(A_{i-1}) & \text{if } a_i \in \mathcal{R}(A_{i-1}) \end{cases} \tag{14}$$

Hence, generally we have,

$$\mathcal{R}(M^T) = \mathcal{R}(A). \tag{15}$$

Theorem

For $A \in \mathbb{R}^{m\times n}$, assume the algorithm can detect all the linear independence, we have $\mathcal{R}(M^T) = \mathcal{R}(A)$, hence

$$\min_{x \in \mathbb{R}^n} \|b - Ax\|_2 \Leftrightarrow \min_{x \in \mathbb{R}^n} \|Mb - MAx\|_2 \tag{16}$$

$\min_{x\in\mathbb{R}^n}\|b - Ax\|_2 \Leftrightarrow \min_{x\in\mathbb{R}^n}\|Mb - MAx\|_2$?

For general case, we update M_i by,

$$M_i = M_{i-1} + \frac{1}{f_i}(e_i - \tilde{k}_i)\tilde{v}_i^T, \tag{12}$$

from which we can see every row of M_i is a linear combination of $\tilde{v}_i^T$. Thus,

$$\mathcal{R}(M_i^T) = \text{span}\{\tilde{v}_1, \ldots, \tilde{v}_i\}. \tag{13}$$

Notice

$$\tilde{v}_i = \begin{cases} A(e_i - \tilde{k}_i) \in \mathcal{R}(A_i) & \text{if } a_i \notin \mathcal{R}(A_{i-1}) \\ (A_{i-1}^\dagger)^T k_i \in \mathcal{R}(A_{i-1}) & \text{if } a_i \in \mathcal{R}(A_{i-1}) \end{cases} \tag{14}$$

Hence, generally we have,

$$\mathcal{R}(M^T) = \mathcal{R}(A). \tag{15}$$

Theorem

For $A \in \mathbb{R}^{m\times n}$, assume the algorithm can detect all the linear independence, we have $\mathcal{R}(M^T) = \mathcal{R}(A)$, hence

$$\min_{x\in\mathbb{R}^n}\|b - Ax\|_2 \Leftrightarrow \min_{x\in\mathbb{R}^n}\|Mb - MAx\|_2 \tag{16}$$

$\min_{x\in\mathbb{R}^n} \|b - Ax\|_2 \Leftrightarrow \min_{x\in\mathbb{R}^n} \|Mb - MAx\|_2$?

For general case, we update M_i by,

$$M_i = M_{i-1} + \frac{1}{f_i}(e_i - \tilde{k}_i)\tilde{v}_i^T, \tag{12}$$

from which we can see every row of M_i is a linear combination of $\tilde{v}_i^T$. Thus,

$$\mathcal{R}(M_i^T) = \text{span}\{\tilde{v}_1, \ldots, \tilde{v}_i\}. \tag{13}$$

Notice

$$\tilde{v}_i = \begin{cases} A(e_i - \tilde{k}_i) \in \mathcal{R}(A_i) & \text{if } a_i \notin \mathcal{R}(A_{i-1}) \\ (A_{i-1}^\dagger)^T k_i \in \mathcal{R}(A_{i-1}) & \text{if } a_i \in \mathcal{R}(A_{i-1}) \end{cases} \tag{14}$$

Hence, generally we have,

$$\mathcal{R}(M^T) = \mathcal{R}(A). \tag{15}$$

Theorem

For $A \in \mathbb{R}^{m\times n}$, assume the algorithm can detect all the linear independence, we have $\mathcal{R}(M^T) = \mathcal{R}(A)$, hence

$$\min_{x\in\mathbb{R}^n} \|b - Ax\|_2 \Leftrightarrow \min_{x\in\mathbb{R}^n} \|Mb - MAx\|_2 \tag{16}$$

$\min_{x\in\mathbb{R}^n}\|b - Ax\|_2 \Leftrightarrow \min_{x\in\mathbb{R}^n}\|Mb - MAx\|_2$?

For general case, we update M_i by,

$$M_i = M_{i-1} + \frac{1}{f_i}(e_i - \tilde{k}_i)\tilde{v}_i^T, \tag{12}$$

from which we can see every row of M_i is a linear combination of $\tilde{v}_i^T$. Thus,

$$\mathcal{R}(M_i^T) = \text{span}\{\tilde{v}_1, \ldots, \tilde{v}_i\}. \tag{13}$$

Notice

$$\tilde{v}_i = \begin{cases} A(e_i - \tilde{k}_i) \in \mathcal{R}(A_i) & \text{if } a_i \notin \mathcal{R}(A_{i-1}) \\ (A_{i-1}^\dagger)^T k_i \in \mathcal{R}(A_{i-1}) & \text{if } a_i \in \mathcal{R}(A_{i-1}) \end{cases} \tag{14}$$

Hence, generally we have,

$$\mathcal{R}(M^T) = \mathcal{R}(A). \tag{15}$$

Theorem

For $A \in \mathbb{R}^{m\times n}$, assume the algorithm can detect all the linear independence, we have $\mathcal{R}(M^T) = \mathcal{R}(A)$, hence

$$\min_{x\in\mathbb{R}^n}\|b - Ax\|_2 \Leftrightarrow \min_{x\in\mathbb{R}^n}\|Mb - MAx\|_2 \tag{16}$$

How to Solve the Preconditioned Problem?

Now, we have to solve

$$\min_{x \in \mathbb{R}^n} \|Mb - MAx\|_2. \tag{17}$$

Consider using GMRES to solve $MAx = Mb$:

- $MAx = Mb$ is consistent or not?
  If $\mathcal{R}(M^T) = \mathcal{R}(A)$, there is a nonsingular matrix C so that $A = M^T C$.

$$\mathcal{R}(MA) = \mathcal{R}(MM^T C) = \mathcal{R}(MM^T) = \mathcal{R}(M).$$

- Will GMRES break down without finding a solution?

How to Solve the Preconditioned Problem?

Now, we have to solve

$$\min_{x \in \mathbb{R}^n} \|Mb - MAx\|_2. \tag{17}$$

Consider using GMRES to solve $MAx = Mb$:

- $MAx = Mb$ is consistent or not?
  If $\mathcal{R}(M^T) = \mathcal{R}(A)$, there is a nonsingular matrix C so that $A = M^T C$.

$$\mathcal{R}(MA) = \mathcal{R}(MM^T C) = \mathcal{R}(MM^T) = \mathcal{R}(M).$$

- Will GMRES break down without finding a solution?

How to Solve the Preconditioned Problem?

Now, we have to solve

$$\min_{x \in \mathbb{R}^n} \|Mb - MAx\|_2. \tag{17}$$

Consider using GMRES to solve $MAx = Mb$:

- $MAx = Mb$ is consistent or not?
  If $\mathcal{R}(M^T) = \mathcal{R}(A)$, there is a nonsingular matrix C so that $A = M^T C$.

  $$\mathcal{R}(MA) = \mathcal{R}(MM^T C) = \mathcal{R}(MM^T) = \mathcal{R}(M).$$

- Will GMRES break down without finding a solution?

Break Down?

Let $A \in \mathbb{R}^{m\times n}$, and rank($A$)$= r$. Assume,

$$\mathcal{R}(A) = \text{span}\{a_1, \ldots, a_r\}. \tag{18}$$

Again, we have

$$\tilde{v}_i = \begin{cases} A(e_i - \tilde{k}_i) \in \mathcal{R}(A_i) & \text{if } a_i \notin \mathcal{R}(A_{i-1}) \\ (A_{i-1}^{\dagger})^T k_i \in \mathcal{R}(A_{i-1}) & \text{if } a_i \in \mathcal{R}(A_{i-1}) \end{cases}, \tag{19}$$

which implies

$$\begin{aligned} \mathcal{R}(\tilde{V}) &= \text{span}\{\tilde{v}_1, \ldots, \tilde{v}_r\} \\ \tilde{V} &= [\tilde{v}_1, \ldots, \tilde{v}_r, \tilde{v}_{r+1}, \ldots, \tilde{v}_n] \\ &= [\tilde{U}, \tilde{U}\tilde{H}] \\ \tilde{U} &= A(I - \tilde{K}) \begin{bmatrix} I_{r\times r} \\ 0 \end{bmatrix}. \end{aligned}$$

Again, here we also need to assume the algorithm can detect all the linear independence.

Break Down?

Let $A \in \mathbb{R}^{m \times n}$, and rank($A$)$= r$. Assume,

$$\mathcal{R}(A) = \text{span}\{a_1, \ldots, a_r\}. \quad (18)$$

Again, we have

$$\tilde{v}_i = \begin{cases} A(e_i - \tilde{k}_i) \in \mathcal{R}(A_i) & \text{if } a_i \notin \mathcal{R}(A_{i-1}) \\ (A_{i-1}^{\dagger})^T k_i \in \mathcal{R}(A_{i-1}) & \text{if } a_i \in \mathcal{R}(A_{i-1}) \end{cases}, \quad (19)$$

which implies

$$\begin{aligned} \mathcal{R}(\tilde{V}) &= \text{span}\{\tilde{v}_1, \ldots, \tilde{v}_r\} \\ \tilde{V} &= [\tilde{v}_1, \ldots, \tilde{v}_r, \tilde{v}_{r+1}, \ldots, \tilde{v}_n] \\ &= [\tilde{U}, \tilde{U}\tilde{H}] \\ \tilde{U} &= A(I - \tilde{K}) \begin{bmatrix} I_{r \times r} \\ 0 \end{bmatrix}. \end{aligned}$$

Again, here we also need to assume the algorithm can detect all the linear independence.

Break Down?

Let $A \in \mathbb{R}^{m \times n}$, and rank($A$)$= r$. Assume,

$$\mathcal{R}(A) = \text{span}\{a_1, \ldots, a_r\}. \tag{18}$$

Again, we have

$$\tilde{v}_i = \begin{cases} A(e_i - \tilde{k}_i) \in \mathcal{R}(A_i) & \text{if } a_i \notin \mathcal{R}(A_{i-1}) \\ (A_{i-1}^{\dagger})^T k_i \in \mathcal{R}(A_{i-1}) & \text{if } a_i \in \mathcal{R}(A_{i-1}) \end{cases}, \tag{19}$$

which implies

$$\begin{aligned} \mathcal{R}(\tilde{V}) &= \text{span}\{\tilde{v}_1, \ldots, \tilde{v}_r\} \\ \tilde{V} &= [\tilde{v}_1, \ldots, \tilde{v}_r, \tilde{v}_{r+1}, \ldots, \tilde{v}_n] \\ &= [\tilde{U}, \tilde{U}\tilde{H}] \\ \tilde{U} &= A(I - \tilde{K}) \begin{bmatrix} I_{r \times r} \\ 0 \end{bmatrix}. \end{aligned}$$

Again, here we also need to assume the algorithm can detect all the linear independence.

Break Down?

$$
\begin{aligned}
\tilde{V} &= [\tilde{U}_r, \tilde{U}_r \tilde{H}] \\
&= \tilde{U}_r \begin{bmatrix} I_{r\times r} & \tilde{H} \end{bmatrix} \\
&= A(I-\tilde{K}) \begin{bmatrix} I_{r\times r} \\ 0 \end{bmatrix} \begin{bmatrix} I_{r\times r} & \tilde{H} \end{bmatrix} \\
&= A(I-\tilde{K}) \begin{bmatrix} I_{r\times r} & \tilde{H} \\ 0 & 0 \end{bmatrix}.
\end{aligned}
$$

Hence,

$$
M = (I-\tilde{K})\tilde{F}^{-1} \begin{bmatrix} I_{r\times r} & 0 \\ \tilde{H}^T & 0 \end{bmatrix} (I-\tilde{K})^T A^T. \tag{20}
$$

If there is no numerical dropping, M will be the Moore-Penrose inverse of A,

$$
A^{\dagger} = (I-K)F^{-1} \begin{bmatrix} I_{r\times r} & 0 \\ H^T & 0 \end{bmatrix} (I-K)^T A^T. \tag{21}
$$

Comparing Equation (20) and Equation (21), we can have

$$
\mathcal{R}(M) = \mathcal{R}(A^{\dagger}) = \mathcal{R}(A^T). \tag{22}
$$

Breakdown Free

Theorem

Let $A \in \mathbb{R}^{m\times n}$*, and* $\operatorname{rank}(A) = r$*. Assume that all the linear independence is detected. Then*

$$\mathcal{R}(M^T) = \mathcal{R}(A) \ \textit{ and } \ \mathcal{R}(M) = \mathcal{R}(A^T),$$

which implies,

- $\min_{x\in\mathbb{R}^n} \|b - Ax\|_2 \Leftrightarrow \min_{x\in\mathbb{R}^n} \|Mb - MAx\|_2 \Leftrightarrow MAx = Mb$
- *GMRES can obtain a least squares solution to* $MAx = Mb$ *without breakdown for any* b *and* x_0*.*

Proof.

$$\begin{array}{rcll}
\text{Breakdown free} & \Leftrightarrow & \mathcal{N}(MA) = \mathcal{N}(A^T M^T) \text{ (Brown, Walker, 1997)} & \\
 & \Leftrightarrow & \mathcal{R}(MA) = \mathcal{R}(A^T M^T) & \\
\mathcal{R}(A^T M^T) \subseteq \mathcal{R}(M) & \& & \mathcal{R}(MA) = \mathcal{R}(M) & \Rightarrow \supseteq \\
\mathcal{R}(MA) \subseteq \mathcal{R}(A^T) & \& & \mathcal{R}(A^T M^T) = \mathcal{R}(A^T) & \Rightarrow \subseteq
\end{array}$$

$\square$

Outline

Numerical Results

Table: Information on the matrix

Name	Origin	m	n	rank	density
B	illc1850	1850	712	712	0.6%
C	random	712	10	10	full
A	[B, BC]	1850	722	712	2%

$b = A[1, \ldots, 1]^T$, we stop the iteration when

$$\|A^T(b - Ax)\|_2 \leq 10^{-8} \cdot \|A^T b\|_2.$$

We judge a_i is linear dependent to $\{a_1, \ldots, a_{i-1}\}$ when

$$\|\tilde{u}_i\|_2 \leq 10^{-4} \|A_{i-1}\|_F \|a_i\|_2. \tag{23}$$

Numerical Results

Rank deficient columns are,

$$\begin{array}{ccccc} 713 & 714 & 715 & 716 & 717 \\ 718 & 719 & 720 & 721 & 722\text{'} \end{array} \tag{24}$$

τ_k	rank(V)	deficiency detected	ITS
10^{-3}	712	0, RIF	206
10^{-4}	712	0, RIF	48
10^{-5}	712	0, RIF	27
10^{-6}	712	RIF	10
10^{-6}	712	$713 \sim 722$	10
N.E.			119

The linear dependency is perfectly detected, but no improvement is made. Why ?

- The linearly dependent columns are in the end of A.

Numerical Results

lp_cycle, from the Florida University Sparse Matrices Collection, where zero rows are omitted.

Table: Information on the matrix

Name	m	n	rank	density	rank deficiency	cond
lp_cycle^T	3371	1890	1875	0.3%	15	10^7

$b = A[1, \ldots, 1]^T$, we stop the iteration when

$$\|A^T(b - Ax)\|_2 \leq 10^{-8} \cdot \|A^T b\|_2.$$

We judge a_i is linear dependent to $\{a_1, \ldots, a_{i-1}\}$ when

$$\|\tilde{u}_i\|_2 \leq 10^{-6} \|A_{i-1}\|_F \|a_i\|_2. \tag{25}$$

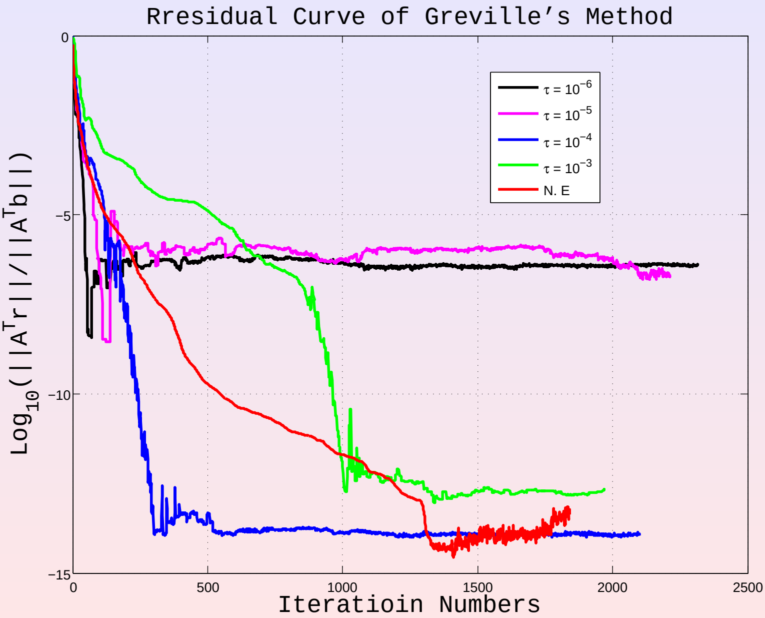
Numerical Results

Rank deficient columns are,

$$\begin{array}{ccccc} 182 & 184 & 216 & 237 & 253 \\ 717 & 754 & 961 & 1221 & 1239, \\ 1260 & 1261 & 1278 & 1640 & 1859 \end{array} \qquad (26)$$

τ_k	rank(V)	deficiency	ITS
10^{-1}	NaN 1875	182th $-1260, -1261, -1278$	RIF 1676
10^{-2}	NaN 1875	182th $-1260, -1261, -1278$	RIF 1376
10^{-3}	NaN 1875	182th $-1260, -1261, -1278$	RIF 905
10^{-4}	NaN 1875	182th $-1239, -1278$	RIF 204
10^{-5}	NaN 1873	182th $+1537, +1545$	RIF 111
10^{-6}	NaN 1873	182th $+1537, +1545$	RIF 56
N.E.			374

Numerical Results



Conclusion

- We proposed a preconditioner that can work with rank-deficient matrices.
- We proved the equivalence and the breakdown free under certain assumptions.