The Augmented Block-Cimmino Distributed Method

Improvements to the Conjugate Gradient accelerated Block Cimmino Method

Mohamed Zenadi[1], Ronan Guivarch[1] Iain S. Duff[2] Daniel Ruiz[1],

IRIT - ENSEEIHT[1], CERFACS and Rutherford Appleton Laboratory[2]

September 7, 2012

Block Cimmino : The basic facts

- Block row projection method [Elfving (Numer. Math. 1980)]

Partitioning the system $Ax = b$

$$\begin{pmatrix} A_1 \\ A_2 \\ \vdots \\ A_p \end{pmatrix} x = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_p \end{pmatrix}$$

Block Cimmino : The basic facts

- Block row projection method [Elfving (Numer. Math. 1980)]

The Block Cimmino Iteration

$$\begin{aligned} \delta_i^{(k)} &= A_i^+ b_i - P_{\mathcal{R}(A_i^T)} x^{(k)} \\ x^{(k+1)} &= x^{(k)} + \nu \sum_{i=1}^{p} \delta_i^{(k)} \end{aligned}$$

where: $A_i^+ = A_i^T \left(A_i A_i^T\right)^{-1}$ and $P_{\mathcal{R}(A_i^T)} = A_i^+ A_i$

Block Cimmino : The basic facts

- Block row projection method [Elfving (Numer. Math. 1980)]
- CG acceleration: iteration matrix is $\sum_{i=1}^{p} A_i^+ A_i = \sum_{i=1}^{p} P_{\mathcal{R}(A_i^T)}$

Acceleration

Apply CG to solve the SPD system

$$\sum_{i=1}^{p} A_i^+ A_i \; x = \sum_{i=1}^{p} A_i^+ b_i$$

Block Cimmino : The basic facts

- Block row projection method [Elfving (Numer. Math. 1980)]
- CG acceleration: iteration matrix is $\sum_{i=1}^{p} A_i^+ A_i = \sum_{i=1}^{p} P_{\mathcal{R}(A_i^T)}$
- Offers a natural parallelism

Acceleration

Solve independently using MUMPS the systems for each partition

$$\begin{bmatrix} I & A_i^T \\ A_i & 0 \end{bmatrix} \begin{bmatrix} u_i \\ v_i \end{bmatrix} = \begin{bmatrix} 0 \\ b_i - A_i x \end{bmatrix}$$

$$\begin{aligned} \text{where : } u_i &= A_i^+ (b_i - A_i x) \\ &= \delta_i \end{aligned}$$

Block Cimmino : The basic facts

- Block row projection method [Elfving (Numer. Math. 1980)]
- CG acceleration: iteration matrix is $\sum_{i=1}^{p} A_i^+ A_i = \sum_{i=1}^{p} P_{\mathcal{R}(A_i^T)}$
- Offers a natural parallelism
- Can also exploit 2nd and 3rd levels of parallelism (sparsity structure, BLAS3 Kernels)

Acceleration

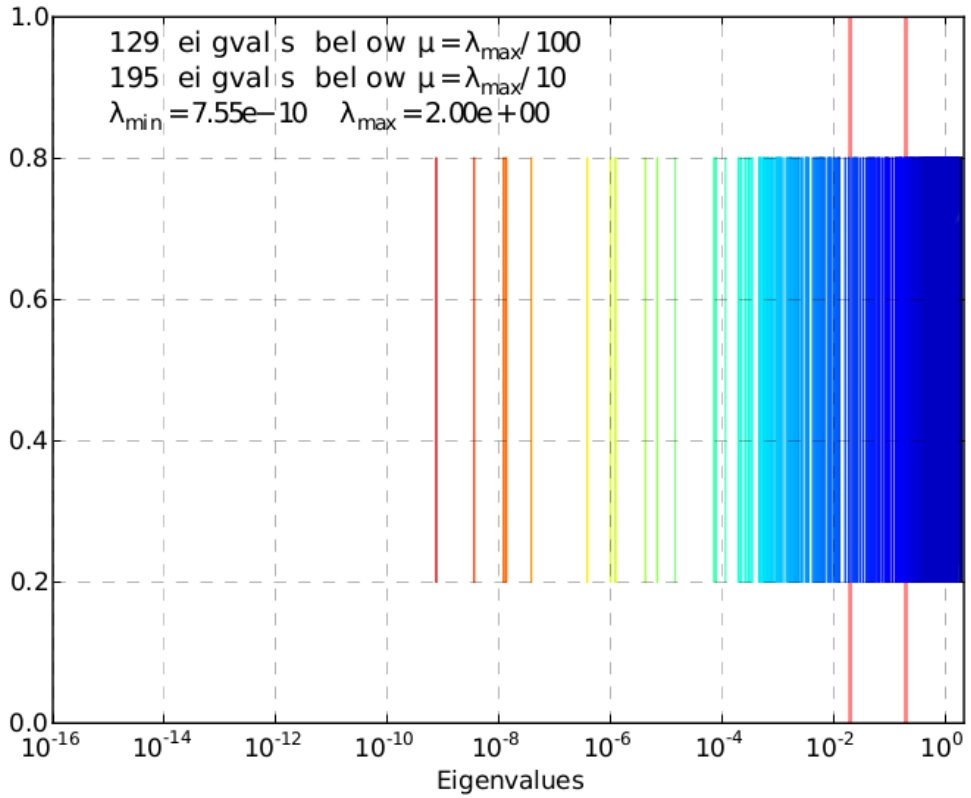
Solve independently using MUMPS the systems for each partition

$$\begin{bmatrix} I & A_i^T \\ A_i & 0 \end{bmatrix} \begin{bmatrix} u_i \\ v_i \end{bmatrix} = \begin{bmatrix} 0 \\ b_i - A_i x \end{bmatrix}$$

$$\begin{aligned} \text{where : } u_i &= A_i^+ (b_i - A_i x) \\ &= \delta_i \end{aligned}$$

Block Cimmino : Numerical considerations

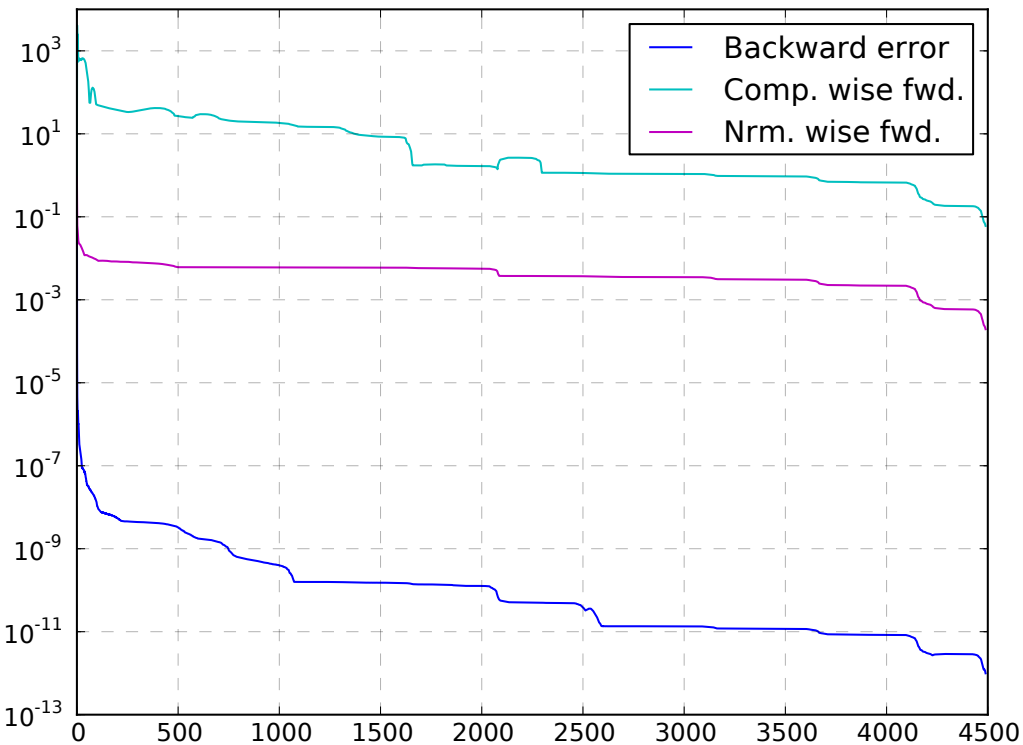
- Block Cimmino iteration matrix exhibits in general good eigenvalue clustering around 1



[bayer01 - 57k - 16 parts] Smallest eigenvalues values of the iteration matrix

Block Cimmino : Numerical considerations

- Block Cimmino iteration matrix exhibits in general good eigenvalue clustering around 1
- However, combination of ill-conditioning with clusters of few eigenvalues at the extremes implies CG convergence with plateaus



`[bayer01 - 57k - 16 parts]` Backward error iteration profile $\omega_k = \|r_k\| / (\|A\| \, \|x_k\| + \|b\|)$

Block Cimmino : Numerical considerations

To improve the convergence one can:

- Preprocess the matrix (scaling, permutation, partitioning strategy) to improve the orthogonality between the partitions [Drummond, Duff, Guivarch, Ruiz, and Zenadi (Precond'11)]
- Use Stabilized Block CG to reduce the length of the plateaus [Arioli, Duff, Ruiz, and Sadkane (SIMAX 1995)]

For successive solves with he same matrix :

- Combine Chebyshev filters (as preconditioners) with conjugate gradient [Golub, Ruiz, Touhami (SIMAX 2007)]

Block Cimmino : Numerical considerations

To improve the convergence one can:

- Preprocess the matrix (scaling, permutation, partitioning strategy) to improve the orthogonality between the partitions [Drummond, Duff, Guivarch, Ruiz, and Zenadi (Precond'11)]
- Use Stabilized Block CG to reduce the length of the plateaus [Arioli, Duff, Ruiz, and Sadkane (SIMAX 1995)]

For successive solves with he same matrix :

- Combine Chebyshev filters (as preconditioners) with conjugate gradient [Golub, Ruiz, Touhami (SIMAX 2007)]

However,

- Effectiveness is problem dependent
- For very large problems, difficult to control explicitly the spread of the intermediate eigenvalues in the iteration matrix (length of plateaus in convergence, memory issues for Krylov basis extraction)

A path to orthogonality

Objectives

- Control explicitly the behavior of the method
- A better and explicit control of the memory needed
- Maintain the efficiency in the case of multiple solves

A path to orthogonality

Objectives

- Control explicitly the behavior of the method
- A better and explicit control of the memory needed
- Maintain the efficiency in the case of multiple solves

How

- Enforce numerical orthogonality between partitions by adding extra variables and constraints
- Extract a condensed smaller subsystem (similar to Schur complement techniques) that can be reused for efficient further solves

A path to orthogonality

Objectives

- Control explicitly the behavior of the method
- A better and explicit control of the memory needed
- Maintain the efficiency in the case of multiple solves

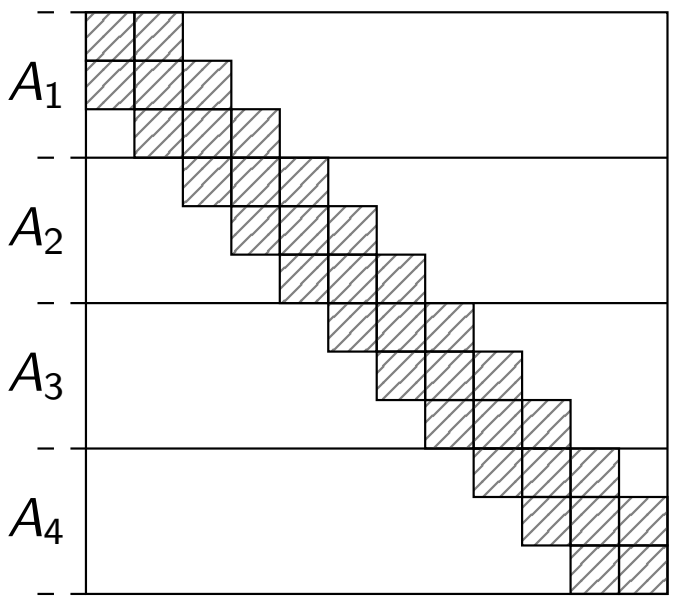
How

- Enforce numerical orthogonality between partitions by adding extra variables and constraints
- Extract a condensed smaller subsystem (similar to Schur complement techniques) that can be reused for efficient further solves

$\Rightarrow$ **A**ugmented **B**lock **C**immino **D**istributed solver (`ABCD solver`)

The augmentation process

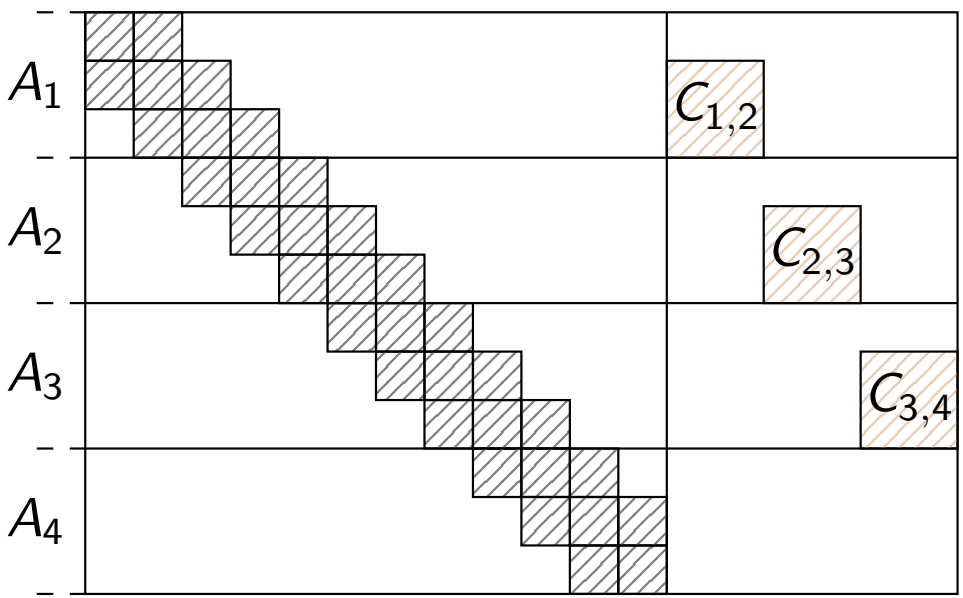
- Partition the matrix with respect to its structure (not necessarily in block-tridiagonal form)



`Illustrative example`

The augmentation process

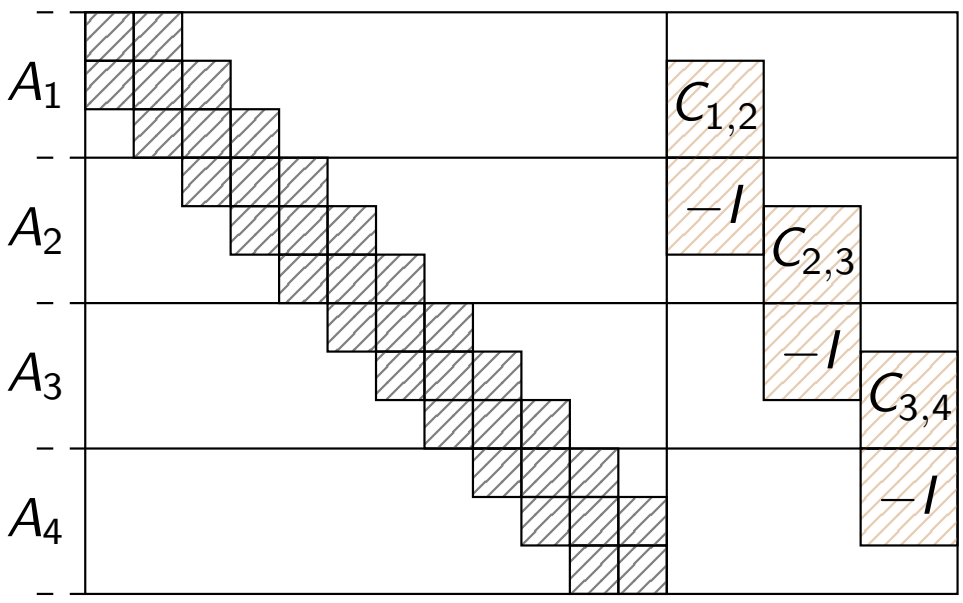
- Partition the matrix with respect to its structure (not necessarily in block-tridiagonal form)
- For each pair of partitions ($j > i$), expand with $C_{i,j} = A_i A_j^T$,



`Illustrative example`

The augmentation process

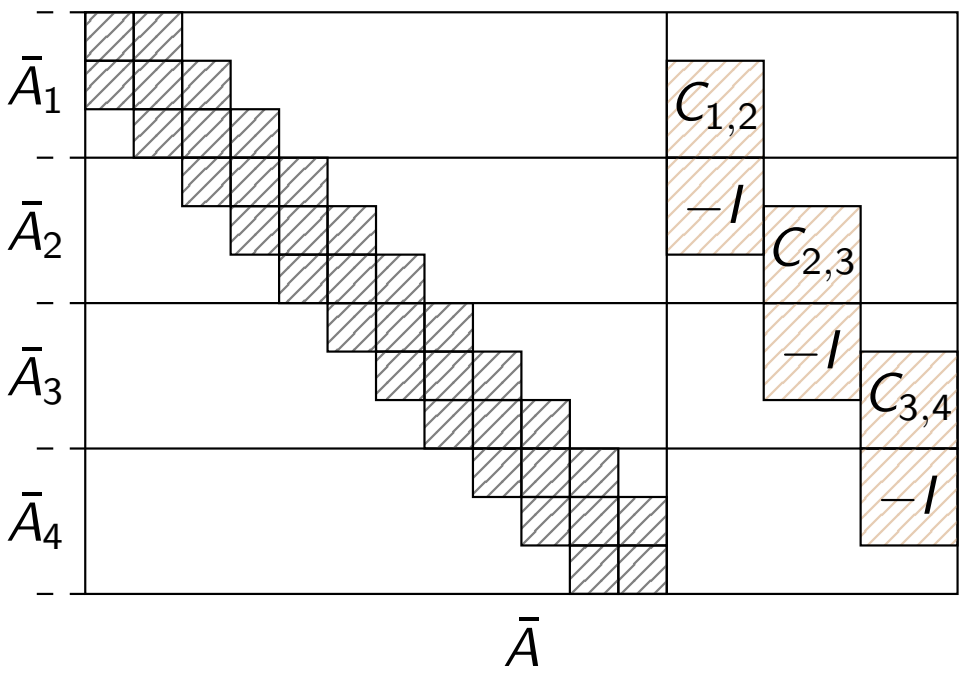
- Partition the matrix with respect to its structure (not necessarily in block-tridiagonal form)
- For each pair of partitions ($j > i$), expand with $C_{i,j} = A_i A_j^T$, and enforce numerical orthogonality



`Illustrative example`

The augmentation process

- Partition the matrix with respect to its structure (not necessarily in block-tridiagonal form)
- For each pair of partitions ($j > i$), expand with $C_{i,j} = A_i A_j^T$, and enforce numerical orthogonality to obtain $\bar{A} = \begin{bmatrix} A & C \end{bmatrix}$



`Illustrative example`

The augmentation process

- Partition the matrix with respect to its structure (not necessarily in block-tridiagonal form)
- For each pair of partitions ($j > i$), expand with $C_{i,j} = A_i A_j^T$, and enforce numerical orthogonality to obtain $\bar{A} = \begin{bmatrix} A & C \end{bmatrix}$
- Add extra constraints to build an equivalent linear system :

$$\begin{bmatrix} A & C \\ 0 & I \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} b \\ 0 \end{bmatrix},$$

where $y = 0$ ensures the same solution x.

The augmentation process

- Partition the matrix with respect to its structure (not necessarily in block-tridiagonal form)
- For each pair of partitions ($j > i$), expand with $C_{i,j} = A_i A_j^T$, and enforce numerical orthogonality to obtain $\bar{A} = \begin{bmatrix} A & C \end{bmatrix}$
- Add extra constraints to build an equivalent linear system :

$$\begin{bmatrix} A & C \\ 0 & I \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} b \\ 0 \end{bmatrix},$$

where $y = 0$ ensures the same solution x.

Problem

the extra partition $Y = \begin{bmatrix} 0 & I \end{bmatrix}$, linked to the constraints equations, is not orthogonal to the previous partitions in $\bar{A} = \begin{bmatrix} A & C \end{bmatrix}$.

The augmentation process

To enforce this orthogonality, we project the column vectors Y^T onto the null space of $\bar{A} = \begin{bmatrix} A & C \end{bmatrix}$ (orthogonal complement of $\mathcal{R}\left(\bar{A}^T\right)$) :

$$W^T = (I - P)\, Y^T,$$

where (as a result of the enforced orthogonality) :

$$P = P_{\mathcal{R}(\bar{A}^T)} = P_{\bigoplus_{i=1}^{p} \mathcal{R}(\bar{A}_i^T)} = \sum_{i=1}^{p} P_{\mathcal{R}(\bar{A}_i^T)}$$

The augmentation process

To enforce this orthogonality, we project the column vectors Y^T onto the null space of $\bar{A} = \begin{bmatrix} A & C \end{bmatrix}$ (orthogonal complement of $\mathcal{R}\left(\bar{A}^T\right)$) :

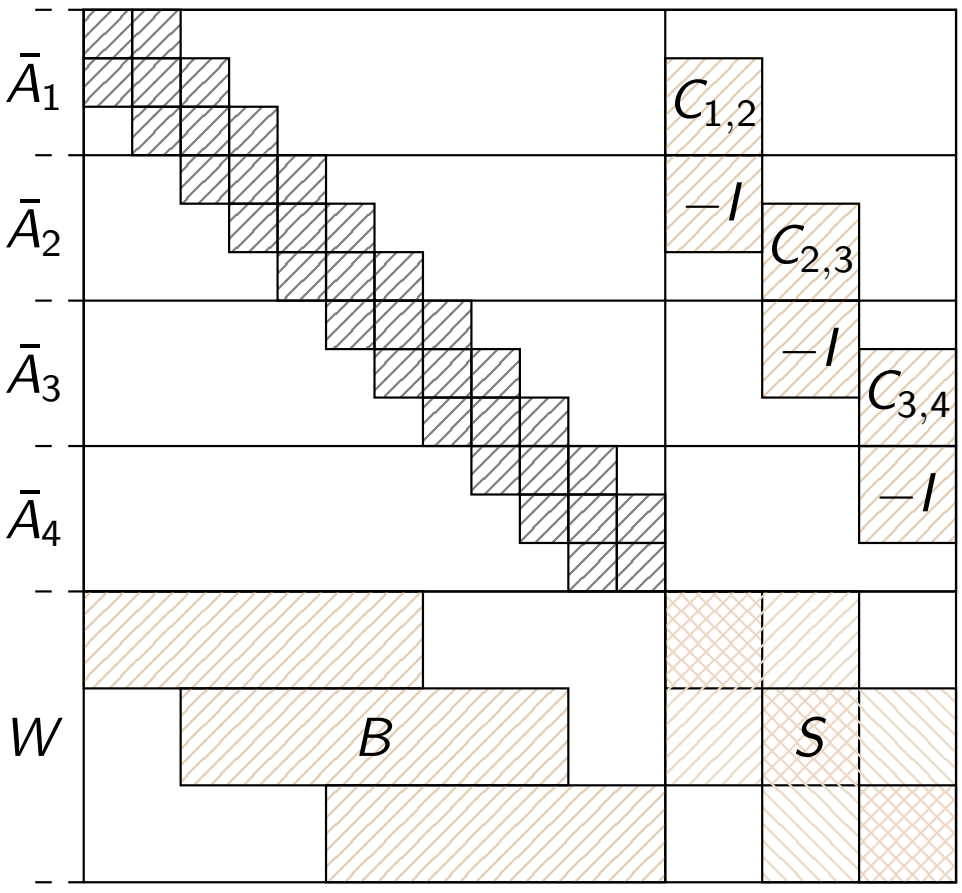
$$W^T = (I - P)\, Y^T,$$

where (as a result of the enforced orthogonality) :

$$P = P_{\mathcal{R}(\bar{A}^T)} = P_{\bigoplus_{i=1}^{p} \mathcal{R}(\bar{A}_i^T)} = \sum_{i=1}^{p} P_{\mathcal{R}(\bar{A}_i^T)}$$

We finally obtain $\begin{bmatrix} A & C \\ B & S \end{bmatrix}$, where $\begin{bmatrix} B & S \end{bmatrix} = W$, an augmented matrix with mutually numerically orthogonal partitions

The augmentation process



Illustrative example

The augmentation process

To keep the consistency within the solution of the new system :

$$\begin{bmatrix} A & C \\ B & S \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} b \\ f \end{bmatrix}$$

we compute the right hand side f as :

$$f \quad = \quad \begin{bmatrix} B & S \end{bmatrix} \begin{bmatrix} x \\ 0 \end{bmatrix} \quad = \quad Y\,(I - P) \begin{bmatrix} x \\ 0 \end{bmatrix}$$

The augmentation process

To keep the consistency within the solution of the new system :

$$\begin{bmatrix} A & C \\ B & S \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} b \\ f \end{bmatrix}$$

we compute the right hand side f as :

$$\begin{aligned} f &= \begin{bmatrix} B & S \end{bmatrix} \begin{bmatrix} x \\ 0 \end{bmatrix} &&= Y(I-P) \begin{bmatrix} x \\ 0 \end{bmatrix} \\ &&&= -YP \begin{bmatrix} x \\ 0 \end{bmatrix} \qquad (\text{since } Y = \begin{bmatrix} 0 & I \end{bmatrix}) \\ &&&= -Y\bar{A}^{+}\bar{A} \begin{bmatrix} x \\ 0 \end{bmatrix} \\ f &= -Y\bar{A}^{+}b \end{aligned}$$

Since all the partitions in the new equivalent linear system

$$\begin{bmatrix} A & C \\ B & S \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} b \\ f \end{bmatrix}$$

are mutually numerically orthogonal, the Cimmino iteration matrix becomes the `Identity` matrix,

Implicit Direct Solver

Since all the partitions in the new equivalent linear system

$$\begin{bmatrix} A & C \\ B & S \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} b \\ f \end{bmatrix}$$

are mutually numerically orthogonal, the Cimmino iteration matrix becomes the `Identity` matrix, and the solution can be directly obtained as :

$$\begin{aligned} \begin{bmatrix} x \\ y \end{bmatrix} &= \bar{A}^{+} b + W^{+} f \\ &= \bar{A}^{+} b - W^{+} Y \bar{A}^{+} b \\ &= \sum_{i=1}^{p} \bar{A}_i^{+} b_i - W^{+} Y \sum_{i=1}^{p} \bar{A}_i^{+} b_i \end{aligned}$$

Computational Ingredients

Knowing that $W = \begin{bmatrix} B & S \end{bmatrix} = Y\,(I - P)$, with $Y = \begin{bmatrix} 0 & I \end{bmatrix}$, we have :

Computational Ingredients

Knowing that $W = \begin{bmatrix} B & S \end{bmatrix} = Y\,(I - P)$, with $Y = \begin{bmatrix} 0 & I \end{bmatrix}$, we have :

$$\begin{aligned} WW^T &= Y\,(I-P)\,(I-P)^T\,Y^T \\ &= Y\,(I-P)^2\,Y^T \\ &= Y\,(I-P)\,Y^T \\ &= \begin{bmatrix} B & S \end{bmatrix} Y^T \\ &= S \end{aligned}$$

Computational Ingredients

Knowing that $W = \begin{bmatrix} B & S \end{bmatrix} = Y\,(I - P)$, with $Y = \begin{bmatrix} 0 & I \end{bmatrix}$, we have :

$$\begin{aligned} WW^T &= Y\,(I-P)\,(I-P)^T\,Y^T \\ &= Y\,(I-P)^2\,Y^T \\ &= Y\,(I-P)\,Y^T \\ &= \begin{bmatrix} B & S \end{bmatrix} Y^T \\ &= S \end{aligned}$$

Therefore $\boxed{S = Y\,(I-P)\,Y^T}$ and is SPD

Computational Ingredients

Knowing that $W = \begin{bmatrix} B & S \end{bmatrix} = Y\,(I - P)$, with $Y = \begin{bmatrix} 0 & I \end{bmatrix}$, we have :

$$\begin{aligned} WW^T &= Y\,(I-P)\,(I-P)^T\,Y^T \\ &= Y\,(I-P)^2\,Y^T \\ &= Y\,(I-P)\,Y^T \\ &= \begin{bmatrix} B & S \end{bmatrix} Y^T \\ &= S \end{aligned}$$

Therefore $\boxed{S = Y\,(I-P)\,Y^T}$ and is SPD

and the pseudo inverse $W^+ = W^T(WW^T)^{-1}$ is given by

$$\begin{aligned} W^+ &= W^T S^{-1} \\ W^+ &= (I-P)\,Y^T S^{-1} \end{aligned}$$

The solution is thus given by :

$$\begin{bmatrix} x \\ y \end{bmatrix} = \bar{A}^{+} b + W^{+} f$$
$$= \bar{A}^{+} b - (I - P) Y^{T} S^{-1} Y \bar{A}^{+} b$$

Computational Ingredients

The solution is thus given by :

$$\begin{bmatrix} x \\ y \end{bmatrix} = \bar{A}^{+} b + W^{+} f$$
$$= \bar{A}^{+} b - (I - P)\, Y^{T} S^{-1} Y \bar{A}^{+} b$$

which can be computed through the 4 following steps :

Computational Ingredients

The solution is thus given by :

$$\begin{bmatrix} x \\ y \end{bmatrix} = \bar{A}^+ b + W^+ f$$
$$= \bar{A}^+ b - (I - P)\, Y^T S^{-1} Y \bar{A}^+ b$$

which can be computed through the 4 following steps :

- Build $w = \bar{A}^+ b$ and then by simple restriction set $f = -Yw$

Computational Ingredients

The solution is thus given by :

$$\begin{bmatrix} x \\ y \end{bmatrix} = \bar{A}^{+} b + W^{+} f$$
$$= \bar{A}^{+} b - (I - P)\, Y^{T} S^{-1} Y \bar{A}^{+} b$$

which can be computed through the 4 following steps :

- Build $w = \bar{A}^{+} b$ and then by simple restriction set $f = -Yw$
- Solve $Sz = f$ (S should be small enough)

Computational Ingredients

The solution is thus given by :

$$\begin{bmatrix} x \\ y \end{bmatrix} = \bar{A}^{+} b + W^{+} f$$
$$= \bar{A}^{+} b - (I - P)\, Y^{T} S^{-1} Y \bar{A}^{+} b$$

which can be computed through the 4 following steps :

- Build $w = \bar{A}^{+} b$ and then by simple restriction set $f = -Yw$
- Solve $Sz = f$ (S should be small enough)
- Expand z and then project it onto the null space of $\bar{A}$ viz.

$$u = (I - P)\, Y^{T} z$$

Computational Ingredients

The solution is thus given by :

$$\begin{bmatrix} x \\ y \end{bmatrix} = \bar{A}^{+} b + W^{+} f$$
$$= \bar{A}^{+} b - (I - P)\, Y^{T} S^{-1} Y \bar{A}^{+} b$$

which can be computed through the 4 following steps :

- Build $w = \bar{A}^{+} b$ and then by simple restriction set $f = -Yw$
- Solve $Sz = f$ (S should be small enough)
- Expand z and then project it onto the null space of $\bar{A}$ viz.
$$u = (I - P)\, Y^{T} z$$
- Then sum $w + u$ to obtain the solution $\begin{bmatrix} x \\ y \end{bmatrix}$ (where $y = 0$)

Computational Ingredients

The solution is thus given by :

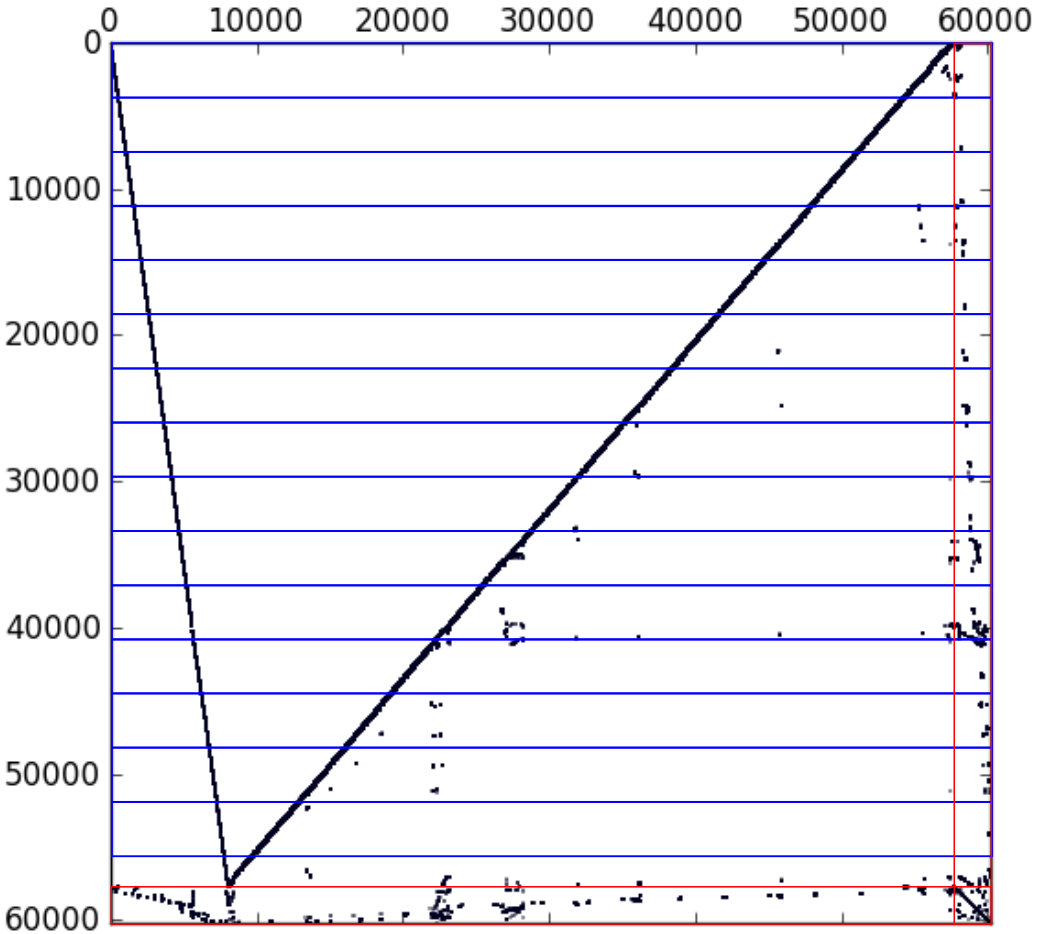
$$\begin{bmatrix} x \\ y \end{bmatrix} = \bar{A}^{+} b + W^{+} f$$
$$= \bar{A}^{+} b - (I - P)\, Y^{T} S^{-1} Y \bar{A}^{+} b$$

which can be computed through the 4 following steps :

- Build $w = \bar{A}^{+} b$ and then by simple restriction set $f = -Yw$
- Solve $Sz = f$ (S should be small enough)
- Expand z and then project it onto the null space of $\bar{A}$ viz.
$$u = (I - P)\, Y^{T} z$$
- Then sum $w + u$ to obtain the solution $\begin{bmatrix} x \\ y \end{bmatrix}$ (where $y = 0$)

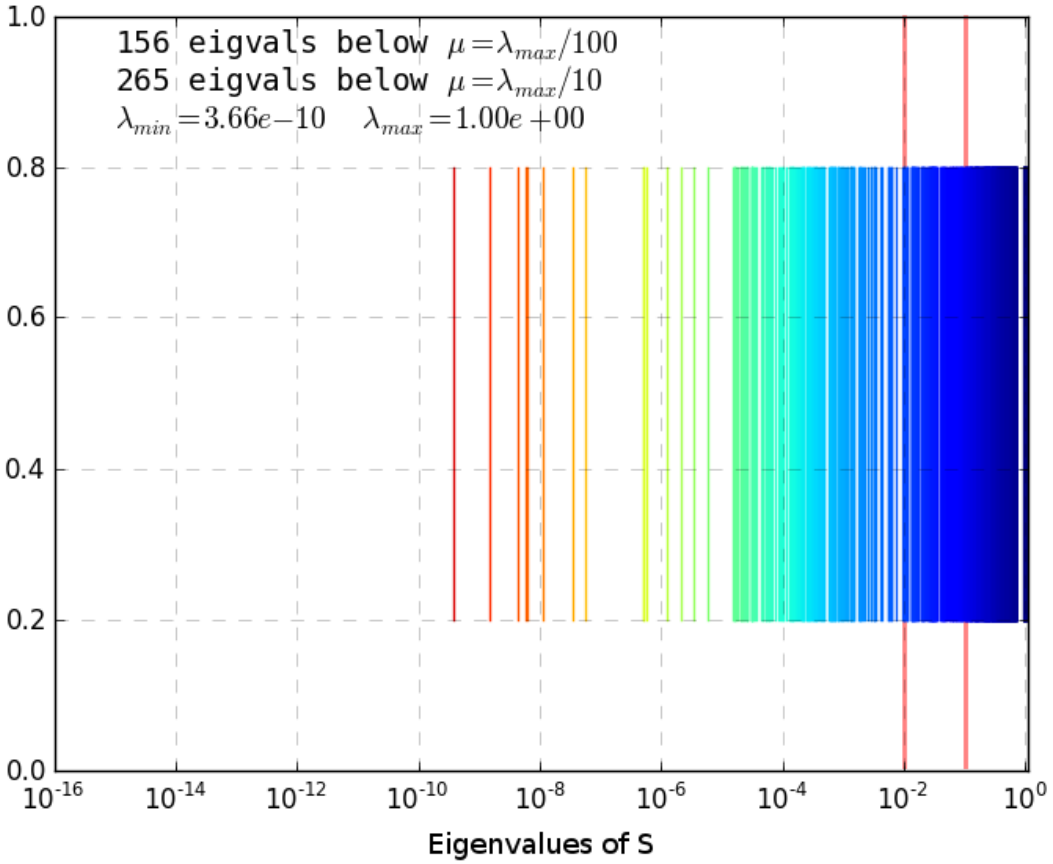
Note that we don't need to build B, only S is used

Example : bayer01 (57k) - 16 partitions



Solution Quality	
$\frac{\|x^*-x\|}{\|x^*\|}$	$3.05e-09$
$\frac{\|r\|}{\|b\|}$	$5.23e-14$

Example : bayer01 (57k) - 16 partitions



Solution Quality	
$\frac{\|x^* - x\|}{\|x^*\|}$	$3.05e-09$
$\frac{\|r\|}{\|b\|}$	$5.23e-14$

Several Issues

- The size of S is problem dependent :
  - sparsity structure (preprocessing, permutations...)
  - number and size of partitions
  - interconnections between partitions

Matrix	**Size**	**#Part.**	**Size of S**
`gre_1107`	1107	5	1227
`bayer01`	57735	16	2951
`lhr34c`	35152	8	10108

Several Issues

- The size of S is problem dependent :
  - sparsity structure (preprocessing, permutations...)
  - number and size of partitions
  - interconnections between partitions
- The size of S can be reduced with the alternative use of the transpose of the augmentation process

Matrix	**Size**	**#Part.**	**Size of S**	**Reduced S**
`gre_1107`	1107	5	1227	1138
`bayer01`	57735	16	2951	2469
`lhr34c`	35152	8	10108	1803

Several Issues

- The size of S is problem dependent :
  - sparsity structure (preprocessing, permutations...)
  - number and size of partitions
  - interconnections between partitions
- The size of S can be reduced with the alternative use of the transpose of the augmentation process
- The ill-conditioning of S can be as close as that of the original iteration matrix, but the density of eigenvalues will be less

Matrix	**Size**	**#Part.**	**Size of S**	**Reduced S**	**Cond. Nb.**
`gre_1107`	1107	5	1227	1138	$1.50e+13$
`bayer01`	57735	16	2951	2469	$4.62e+9$
`lhr34c`	35152	8	10108	1803	$2.23e+12$

Several Issues

- The size of S is problem dependent :
  - sparsity structure (preprocessing, permutations...)
  - number and size of partitions
  - interconnections between partitions
- The size of S can be reduced with the alternative use of the transpose of the augmentation process
- The ill-conditioning of S can be as close as that of the original iteration matrix, but the density of eigenvalues will be less
- Open choice : build S and factorize it, or use S implicitly (MV products) in an iterative process (CG, S is SPD)

Matrix	**Size**	**#Part.**	**Size of S**	**Reduced S**	**Cond. Nb.**
`gre_1107`	1107	5	1227	1138	$1.50e+13$
`bayer01`	57735	16	2951	2469	$4.62e+9$
`lhr34c`	35152	8	10108	1803	$2.23e+12$

Relaxation of the augmentation process

To better control the size of S, we propose to drop entries in the contributing blocks $C_{i,j}$ to reduce the number of added columns.
Target:

- A reduced size of S with respect to the size of A : better control of memory requirements
- Build and factorize S : takes care of the ill-conditioning in S, factors can be reused for efficient further solves

Relaxation of the augmentation process

To better control the size of S, we propose to drop entries in the contributing blocks $C_{i,j}$ to reduce the number of added columns.

Target:

- A reduced size of S with respect to the size of A : better control of memory requirements
- Build and factorize S : takes care of the ill-conditioning in S, factors can be reused for efficient further solves

Issues:

- The augmented partitions $\bar{A}_i$ lose *"partly"* their mutual numerical orthogonality
- $(I - P)$ is no longer explicitly available, and must be recovered via an iterative process
- The iteration matrix in this process is $\sum_{i=1}^{p} P_{\mathcal{R}(\bar{A}_i^T)}$

Inner iterations for $(I - P)$

- To recover the effect of $(I - P)$ onto any vector z, we use CG to solve the symmetric semi-positive definite linear system

$$\sum_{i=1}^{p} \bar{A}_i^+ \bar{A}_i \; x = \sum_{i=1}^{p} \bar{A}_i^+ \bar{A}_i z$$

- $\sum_{i=1}^{p} \bar{A}_i^+ \bar{A}_i$ corresponds to the Block Cimmino iteration matrix on the first set of augmented partitions, and has the same kernel as that of $\bar{A}$
- CG on consistent symmetric semi-positive definite linear systems computes the minimum norm solution : $x = Pz$

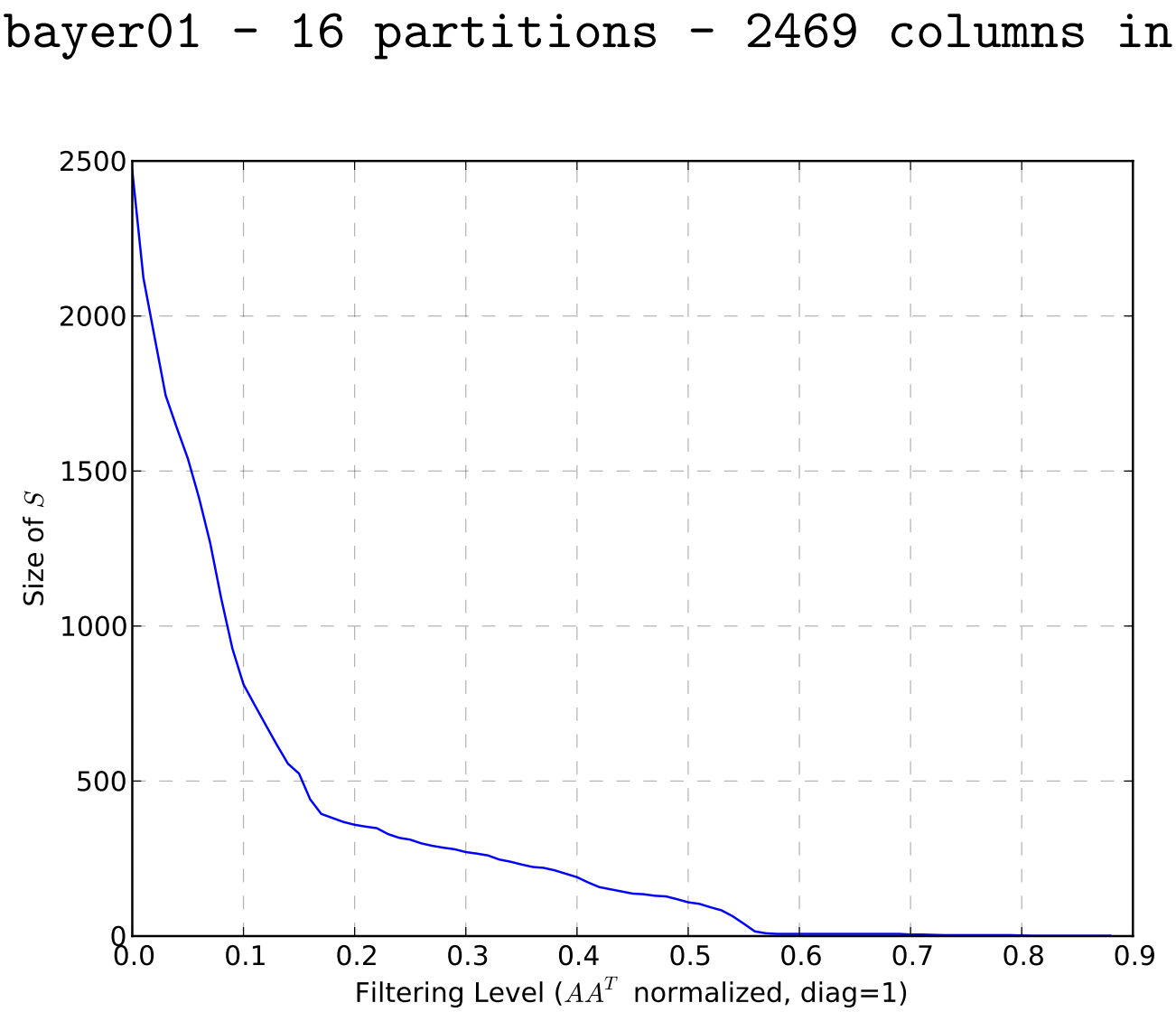
Inner iterations for $(I - P)$

- To recover the effect of $(I - P)$ onto any vector z, we use CG to solve the symmetric semi-positive definite linear system

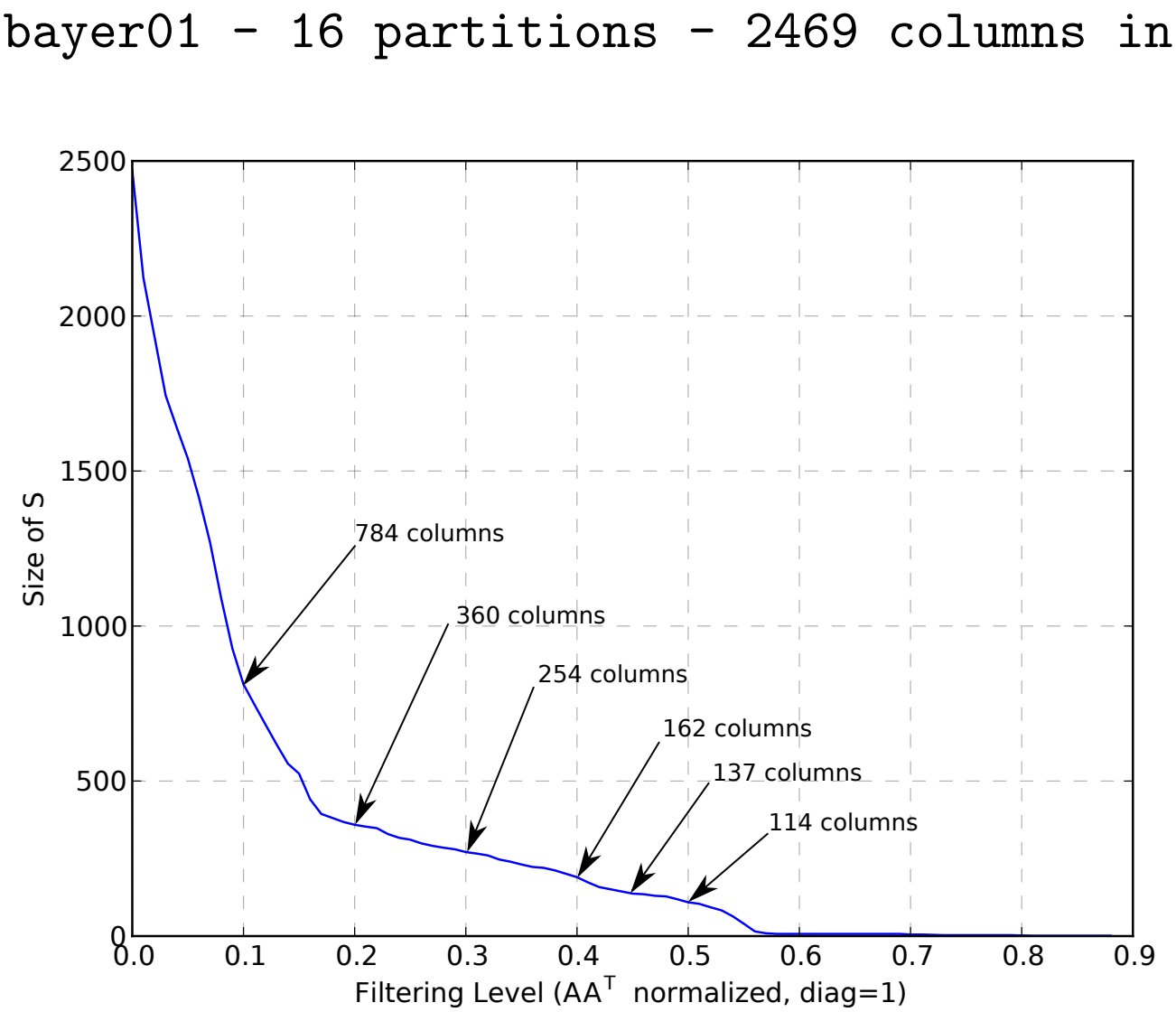
$$\sum_{i=1}^{p} \bar{A}_i^+ \bar{A}_i \, x = \sum_{i=1}^{p} \bar{A}_i^+ \bar{A}_i z$$

- $\sum_{i=1}^{p} \bar{A}_i^+ \bar{A}_i$ corresponds to the Block Cimmino iteration matrix on the first set of augmented partitions, and has the same kernel as that of $\bar{A}$
- CG on consistent symmetric semi-positive definite linear systems computes the minimum norm solution : $x = Pz$
- Possible numerical stability issues : investigate alternative techniques such as MINRES-QLP `[Choi, Paige, and Saunders (SISC 2011)]`

Example of C filtering



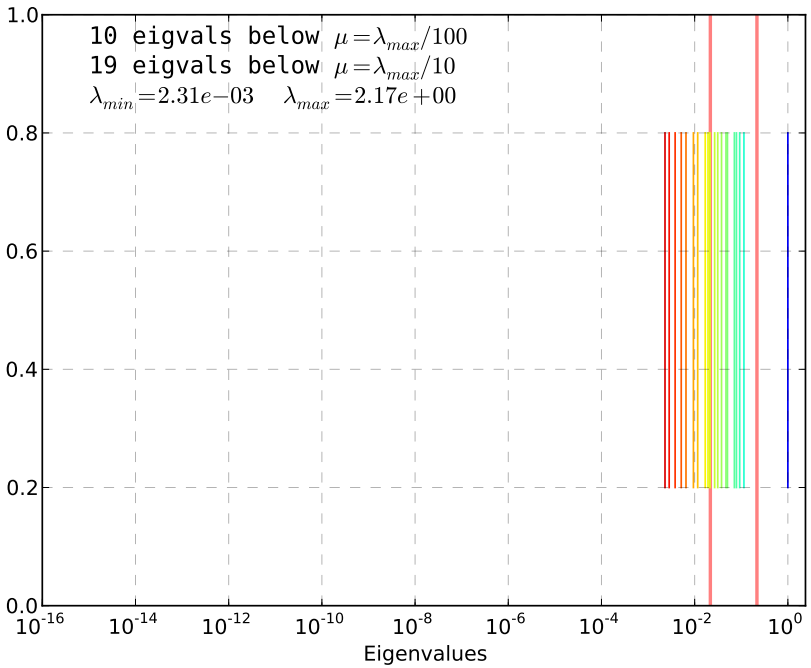
Example of C filtering



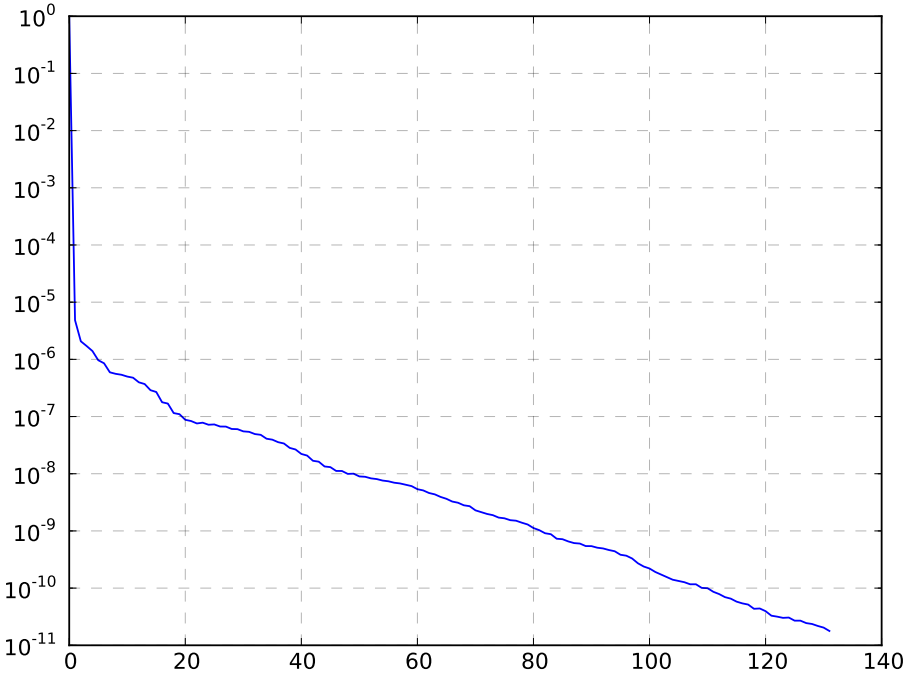
Example of C filtering

bayer01 | 16 partitions | Filter C at 0.10 | 784 cols

Ritz values of the iteration matrix



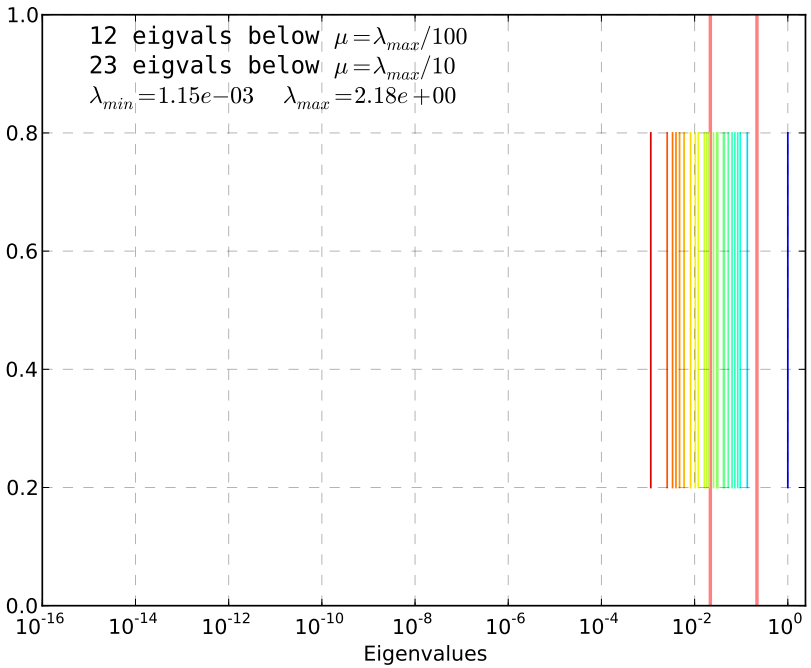
Backward error iteration profile of $w = \bar{A}^{+} b$



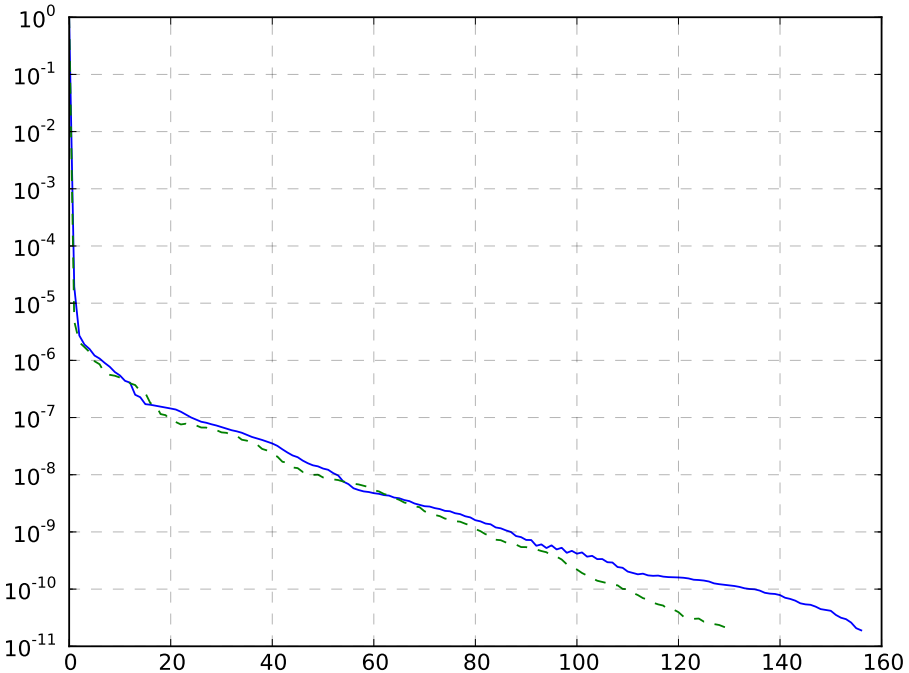
Example of C filtering

bayer01 | 16 partitions | Filter C at 0.20 | 360 cols

Ritz values of the iteration matrix



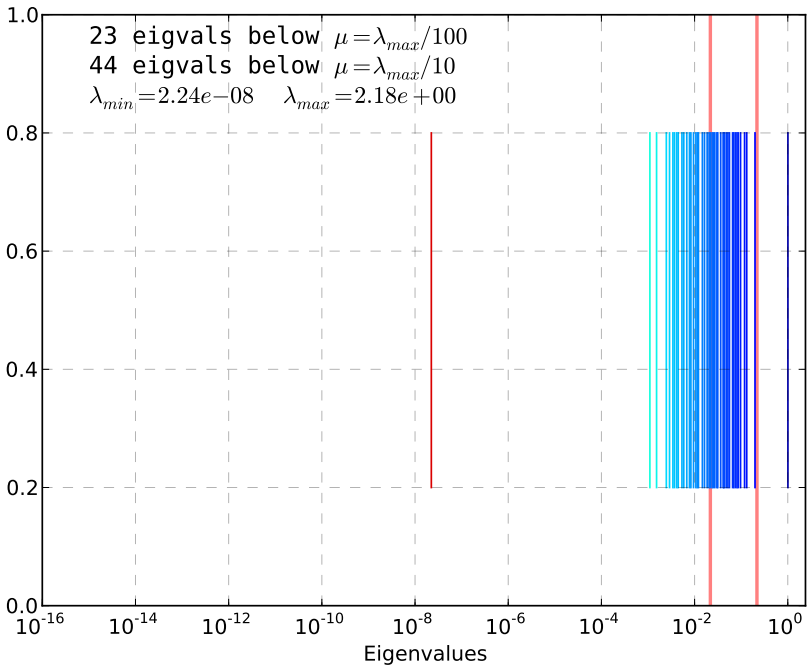
Backward error iteration profile of $w = \bar{A}^{+} b$



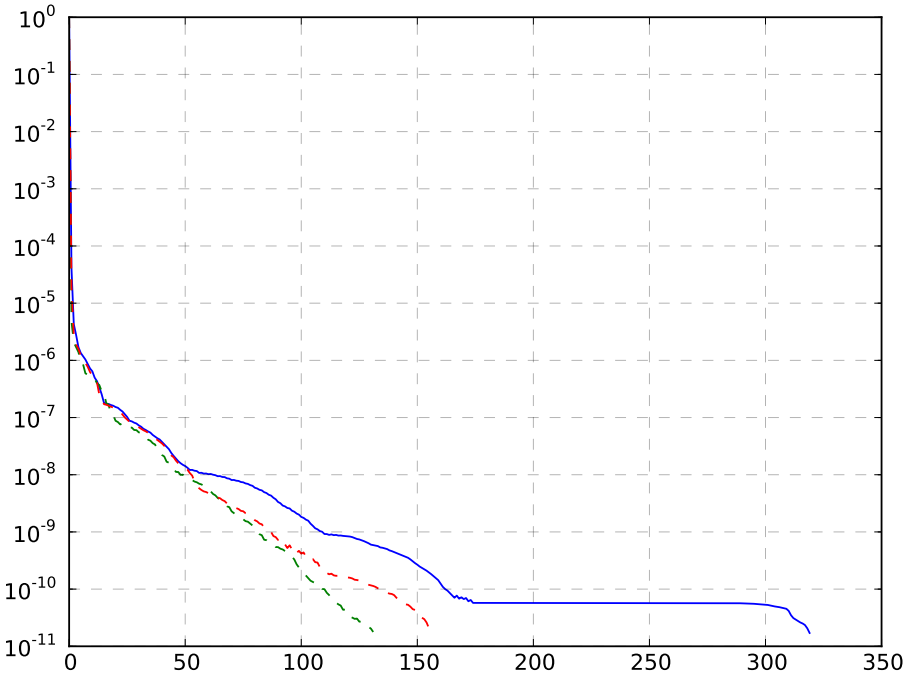
Example of C filtering

bayer01 | 16 partitions | Filter C at 0.30 | 254 cols

Ritz values of the iteration matrix



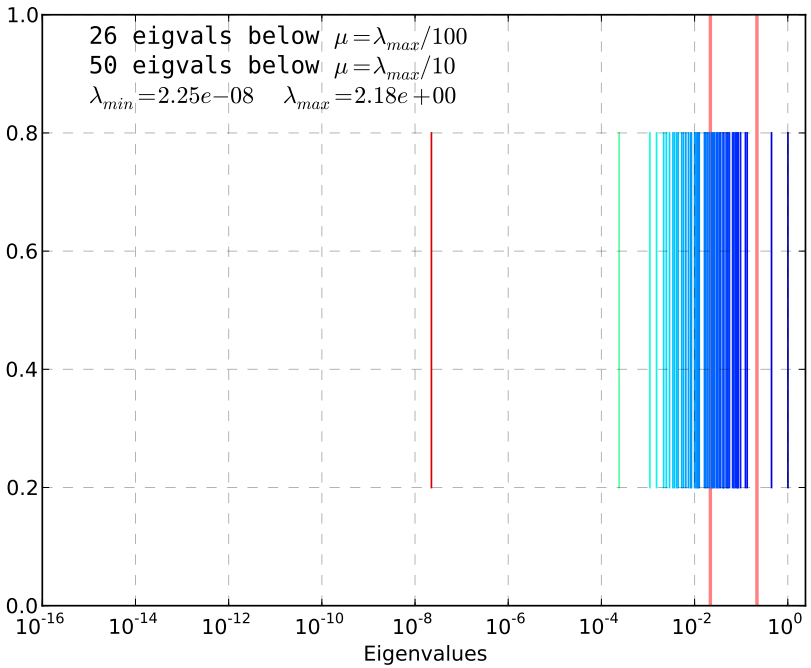
Backward error iteration profile of $w = \bar{A}^{+} b$



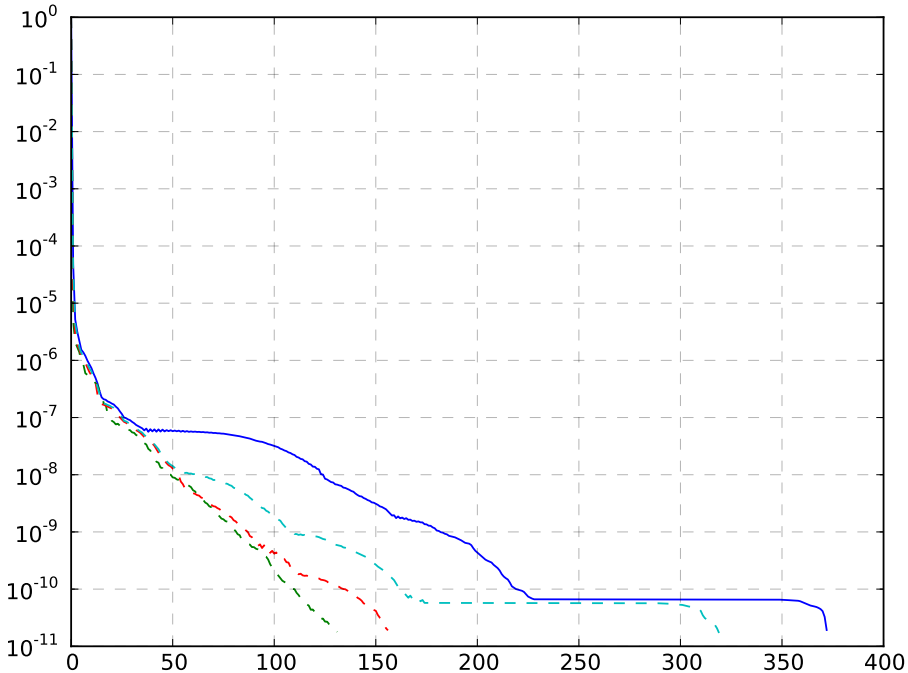
Example of C filtering

bayer01 | 16 partitions | Filter C at 0.35 | 201 cols

Ritz values of the iteration matrix



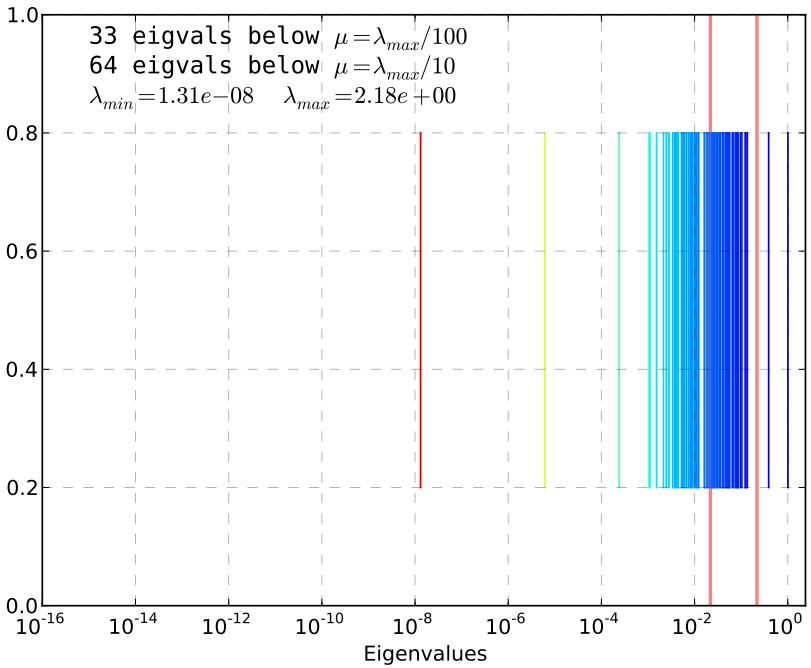
Backward error iteration profile of $w = \bar{A}^{+} b$



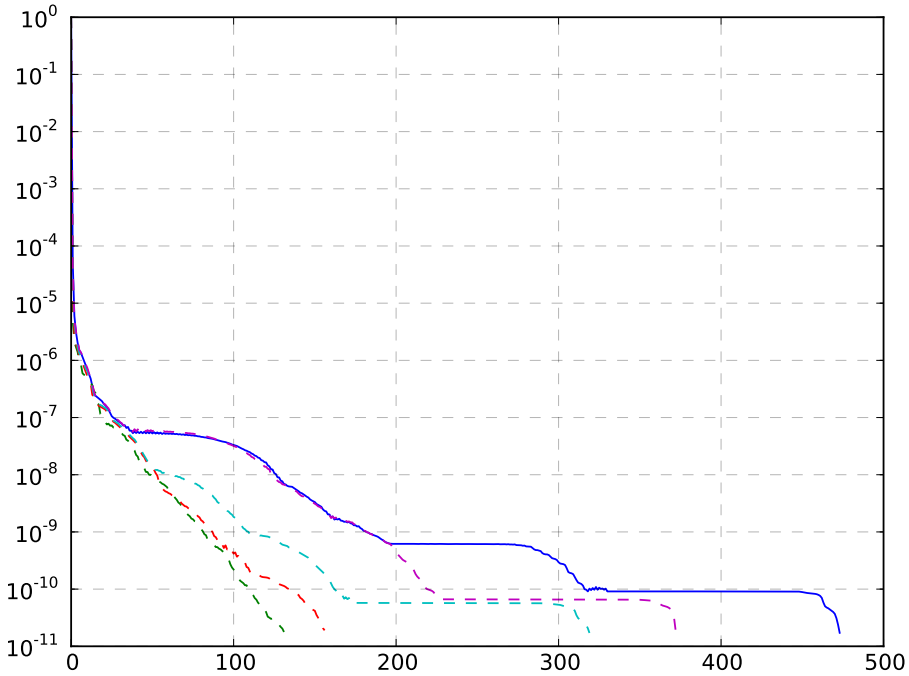
Example of C filtering

bayer01 | 16 partitions | Filter C at 0.4 | 162 cols

Ritz values of the iteration matrix



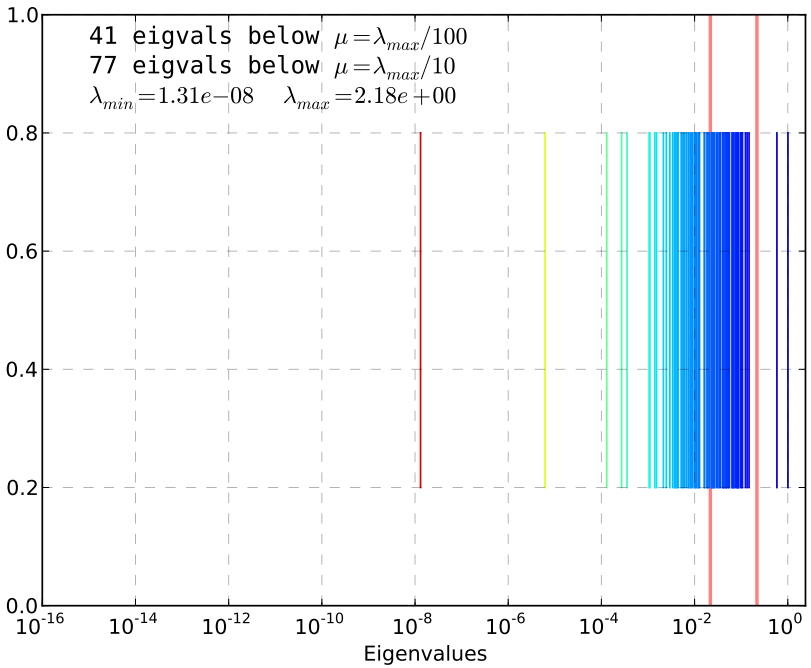
Backward error iteration profile of $w = \bar{A}^{+} b$



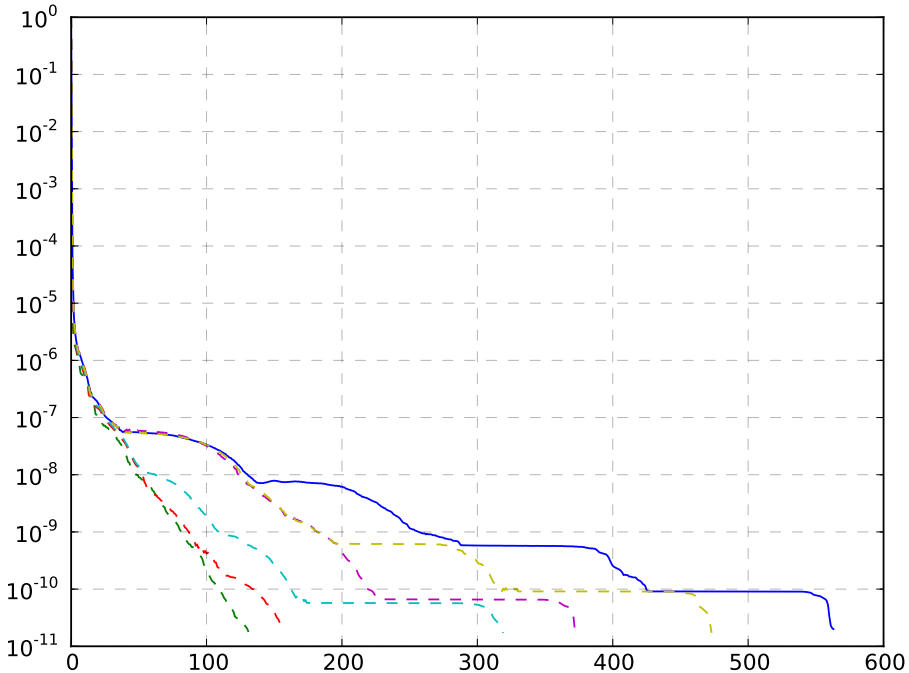
Example of C filtering

bayer01 | 16 partitions | Filter C at 0.45 | 137 cols

Ritz values of the iteration matrix



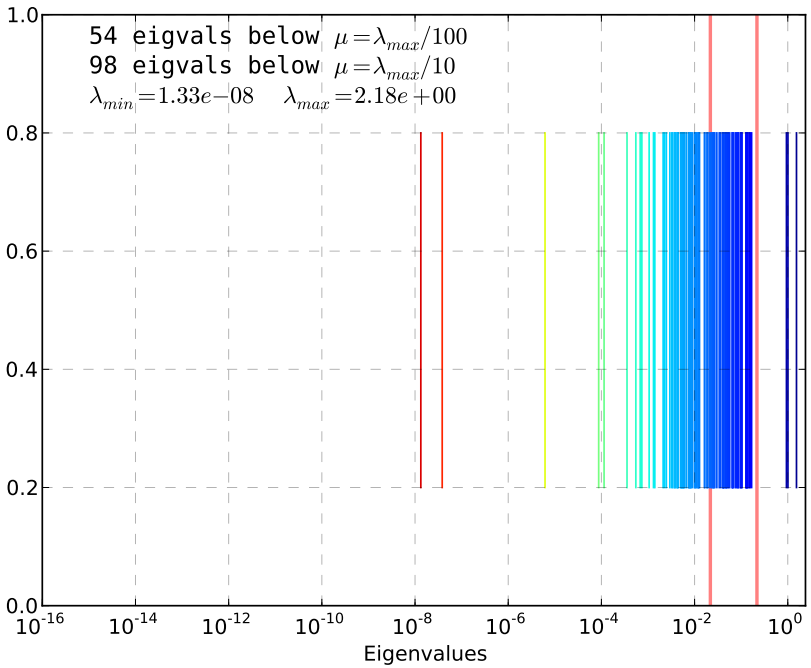
Backward error iteration profile of $w = \bar{A}^{+} b$



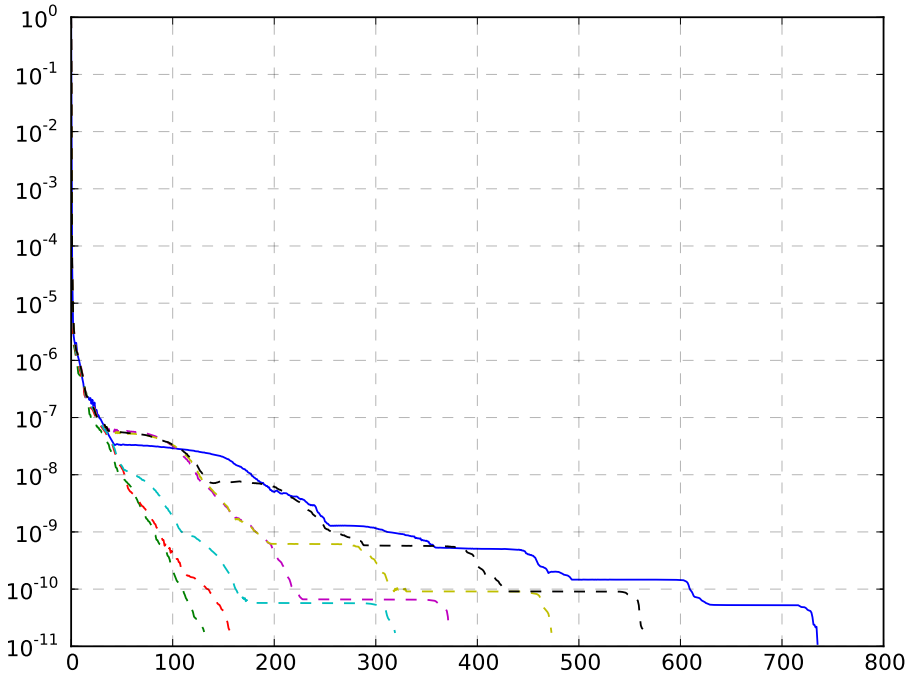
Example of C filtering

bayer01 | 16 partitions | Filter C at 0.5 | 114 cols

Ritz values of the iteration matrix

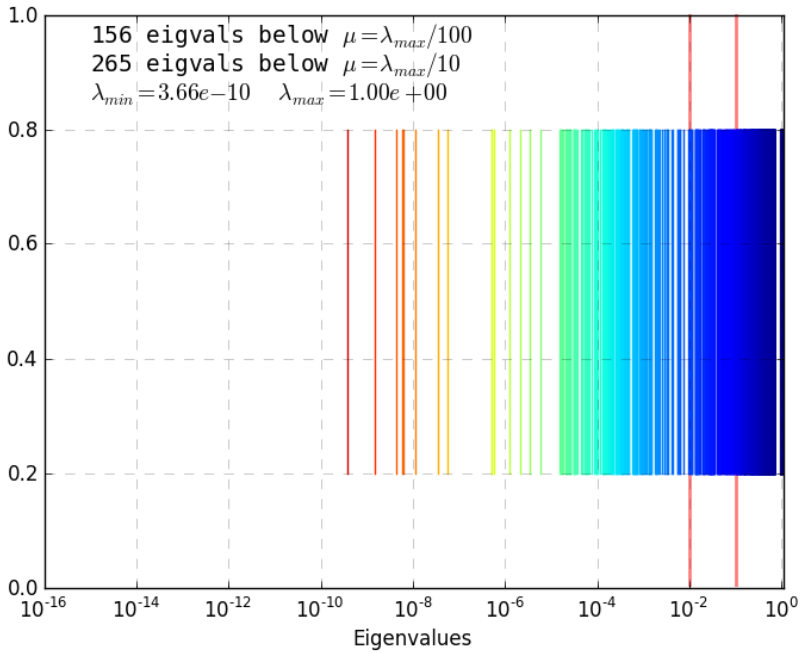


Backward error iteration profile of $w = \bar{A}^{+} b$

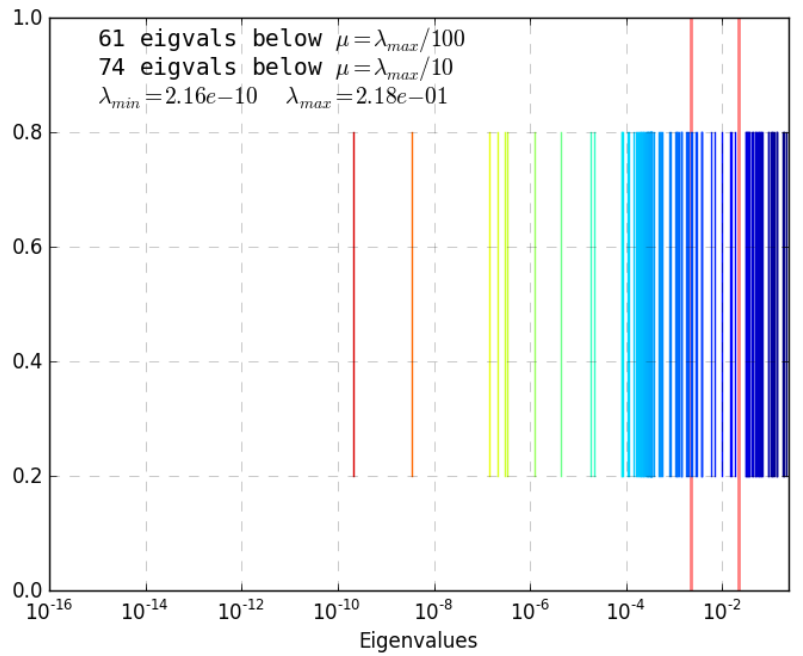


Example of C filtering

Eigenvalues of S without filtering of C (2469 cols)



Eigenvalues of S with filtering of C at 0.5 (114 cols)



Example of C filtering : summary

bayer01 - 16 partitions

	0	0.1	0.3	0.4	0.45
Size of S	2469	784	254	162	137
Nb. Iter. $w = \bar{A}^{+}b$	1	155	334	491	577
Med./Max Iter. S	1	24/43	33/64	51/121	59/160
$\|r\|/\|b\|$	$5e-14$	$3e-9$	$1e-7$	$1e-9$	$2e-9$
$\|x^* - x\|_\infty/\|x^*\|_\infty$	$3e-9$	$4e-6$	$4e-5$	$9e-6$	$7e-5$

lhr34c - 8 partitions

0	0.1	0.3
1803	1299	465
1	44	192
1	24/43	166/496
$2e-7$	$9e-3$	$7e-2$
$2e-2$	$1e-1$	$3e-1$

gre_1107 - 5 partitions

0	0.1	0.45
1138	904	86
1	9	122
1	9/10	127/133
$1e-10$	$9e-8$	$1e-7$
$9e-5$	$2e-2$	$4e-2$

Open Issues / Future Work

- Build simultaneously, and in parallel, multiple columns of S
- Investigate iteration solution without building S
- Sparse approximation of S as a preconditioner
- Investigate numerically stable alternatives to CG for semi-positive definite systems
- Analyze the benefit in the case of multiple (successive) RHS
- Investigate memory requirements and trade-offs with respect to computational work
- Chebyshev filters to precondition Pz computation
- Extensive analysis of the parallel implementation of the method

Thank you for your attention!

email:
mohamed.zenadi@enseeiht.fr