Automatic Differentiation and Preconditioning: Myth or Reality?

H. Martin Bücker
Institute for Scientific Computing
Department of Computer Science

RWTH AACHEN UNIVERSITY

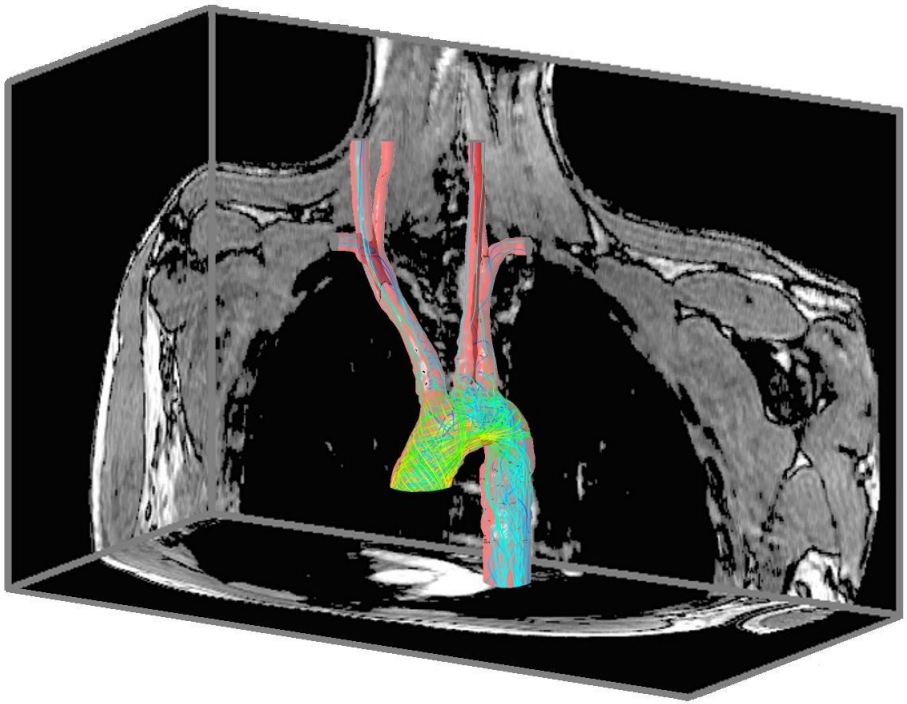
Problem

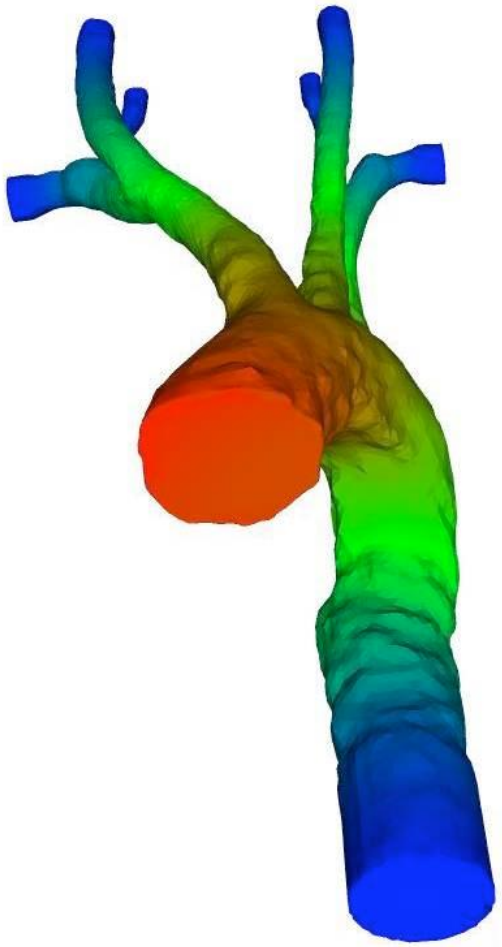
$$A\mathbf{x} = \mathbf{b} \qquad \mathbf{x}, \mathbf{b} \in \mathbb{R}^N$$

$$A = \frac{\partial \mathbf{f}}{\partial \mathbf{z}} \in \mathbb{R}^{N \times N}$$ Jacobian matrix

large, sparse,
unsymmetric, nonsingular,
a priori known pattern of nonzero elements

Flow through Aorta

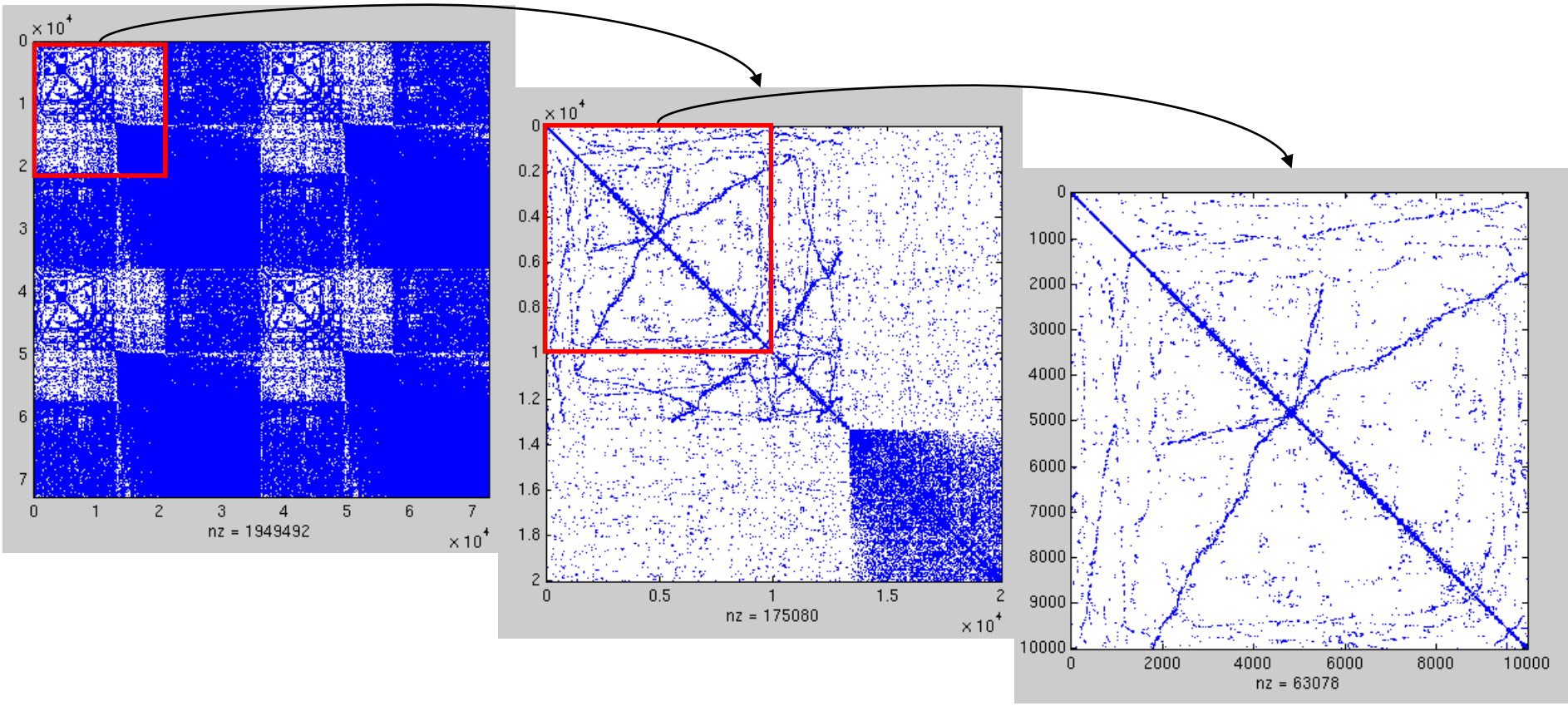




APPLIED MEDICAL ENGINEERING

CHAIR FOR COMPUTATIONAL ANALYSIS OF TECHNICAL SYSTEMS

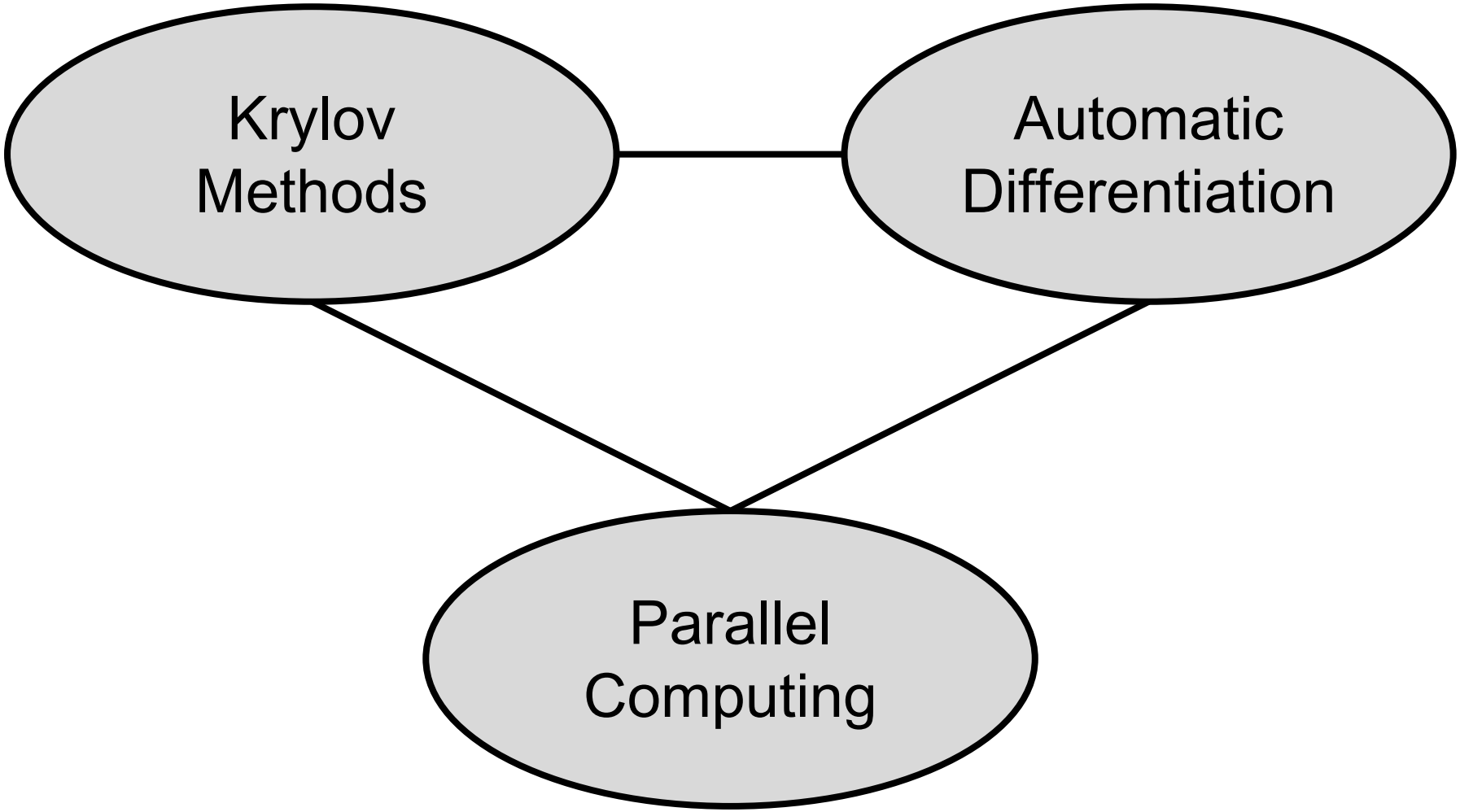
Coefficient Matrix



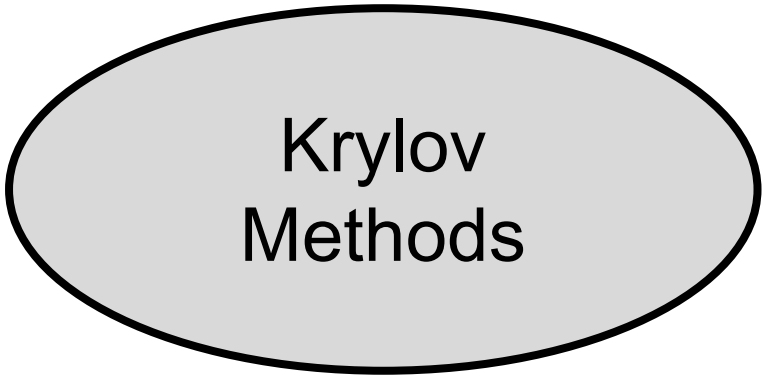
$N = 290{,}000$ (small problem)

0,04% nonzeros

Research Areas



Research Area



Iterative solution of linear systems

Krylov Subspace Methods

Solve $A\mathbf{x} = \mathbf{b}$ with $A \in \mathbb{R}^{N \times N}$ large, sparse nonsingular

Let $\mathbf{r}_0 := \mathbf{b} - A\mathbf{x}_0$

$$\mathbf{x}_n \in \mathbf{x}_0 + \mathcal{K}_n(A, \mathbf{r}_0)$$
$$\mathbf{r}_n \perp \mathcal{L}_n$$

$n = 1, 2, \ldots$

$$\mathcal{K}_n(A, \mathbf{r}_0) := \text{span}\{\mathbf{r}_0, A\mathbf{r}_0, \ldots, A^{n-1}\mathbf{r}_0\}$$

nth Krylov subspace generated by A and $\mathbf{r}_0$

Design of Krylov Subspace Methods

- Basis for spaces $\mathcal{K}_n$ and $\mathcal{L}_n$
- Definition of current iterate

$$\left\| \mathbf{r}_n \right\|_1$$

Algorithmic Ingredients

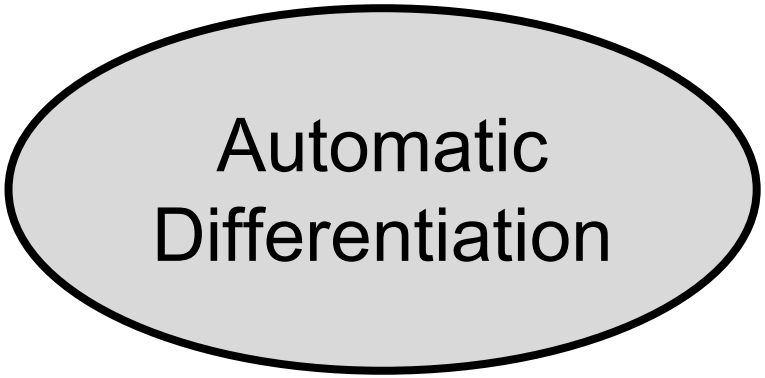
- Preconditioner $M \approx A$

$$M^{-1}A\mathbf{x} = M^{-1}\mathbf{b}$$

- Construction of M with assumption:

  Elementwise access to $A(i,j)$ possible
- Solve „simple“ systems $M\mathbf{y} = \mathbf{z}$
- Matrix-vector products $\mathbf{y} = A\mathbf{z}$ $\mathbf{y} = A^T\mathbf{z}$
- Vector-vector operations $\mathbf{y}^T\mathbf{z}$ $\alpha\mathbf{y} + \beta\mathbf{z}$

Research Area



Fast computation of exact derivatives

Automatic Differentiation (AD)

- Given: arbitrarily complex computer code evaluating some function

- Generates: new computer code evaluating function and derivatives of selected output w.r.t. selected inputs

Program transformation changing semantics

Example of Program Transformation

```
x = f(x,y,n)

  real x,y
  integer n
  do i = 1,n
    x = x*y
  enddo
```

```
[x,g_x] = g_f(x,g_x,y,g_y,n)

  real x,g_x(2),y,g_y(2)
  integer n
  do i = 1,n
    do j = 1,2
      g_x = g_x(j)*y + x*g_y[j]
    enddo
    x = x*y
  enddo
```

Looking for:

$$\frac{\partial f}{\partial x} \qquad \frac{\partial f}{\partial y}$$

Initialization: $\mathtt{g\_x} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ $\mathtt{g\_y} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$

Result: $\mathtt{g\_x} = \begin{bmatrix} \partial f / \partial x \\ \partial f / \partial y \end{bmatrix}$

AD in a Nutshell

- Code corresponds to composition of few elementary functions (+, -, *, sin, ...)
- Derivatives of elementary functions are known
- Chain rule handles composition
- Accumulation leads to final derivatives without truncation error (in exact arithmetic)

Computation of Jacobian Matrices

- Given: Program evaluating

$$\mathbf{f} : \mathbb{R}^N \longrightarrow \mathbb{R}^N \qquad \text{with runtime } T_{\mathbf{f}}(N)$$

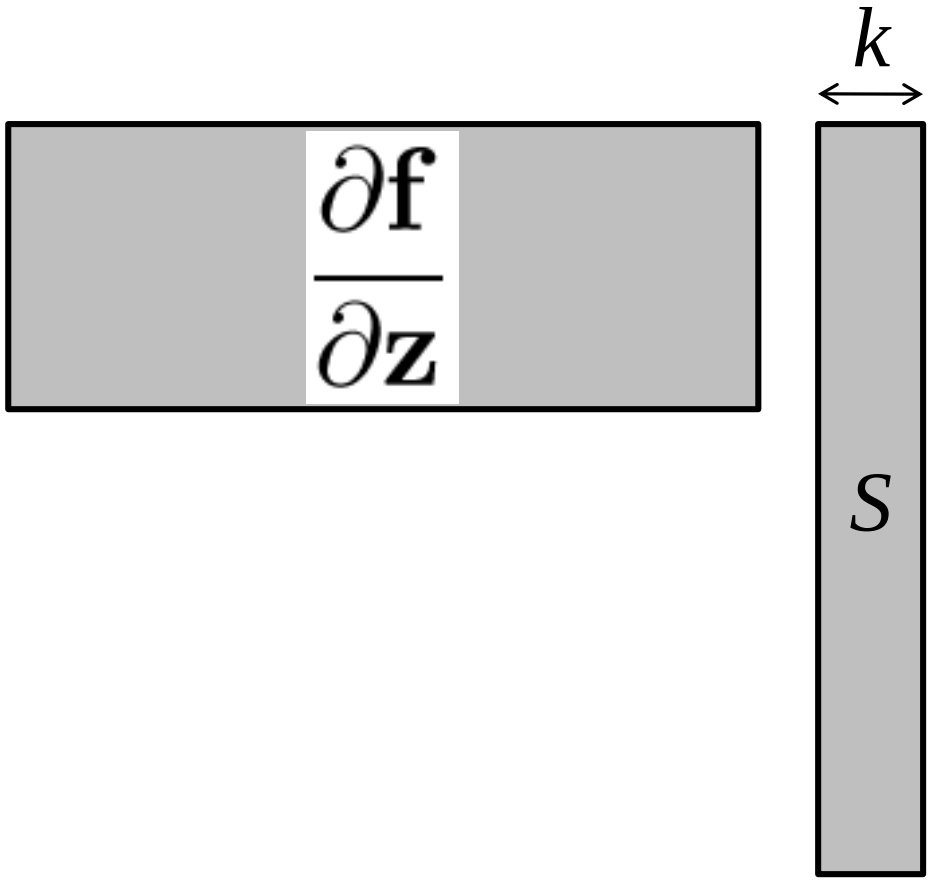
- Options: Program evaluating

a) $\dfrac{\partial \mathbf{f}}{\partial \mathbf{z}} \cdot S \qquad (N \times N) \cdot (N \times k)$

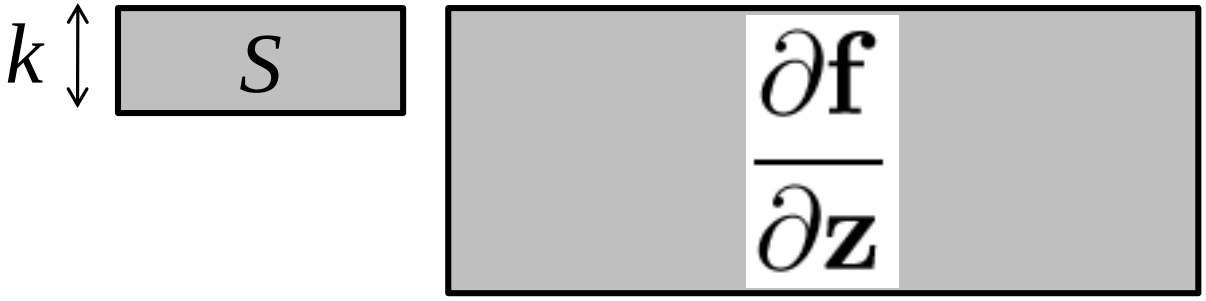
b) $S \cdot \dfrac{\partial \mathbf{f}}{\partial \mathbf{z}} \qquad (k \times N) \cdot (N \times N)$

Runtimes: $T(N, k) \approx k \cdot T_{\mathbf{f}}(N)$

Try to minimize k



Try to minimize k



Exploitation of Sparsity

$$\begin{bmatrix} 2 & 0 & 0 & 0 & 4 \\ 0 & 1 & 0 & 5 & 0 \\ 3 & 0 & 6 & 0 & 7 \end{bmatrix} \cdot \overbrace{\begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix}}^{k=3} = \begin{bmatrix} 2 & 0 & 4 \\ 1 & 0 & 5 \\ 3 & 6 & 7 \end{bmatrix}$$

$$\frac{\partial \mathbf{f}}{\partial \mathbf{z}} \cdot S$$

Exploitation of Sparsity

$$k = 2 \updownarrow \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 2 & 0 & 0 & 0 & 4 \\ 0 & 1 & 0 & 5 & 0 \\ 3 & 0 & 6 & 0 & 7 \end{bmatrix} = \begin{bmatrix} 2 & 1 & 0 & 5 & 4 \\ 3 & 0 & 6 & 0 & 7 \end{bmatrix}$$

$$S \cdot \frac{\partial \mathbf{f}}{\partial \mathbf{z}}$$

In particular, AD allows:

- Efficient exploitation of sparsity
- Efficient ($k = 1$) matrix-vector products

$$\mathbf{y} = A\mathbf{z} \qquad\qquad \mathbf{y} = A^T\mathbf{z}$$

- Access to complete rows and columns

Missing Connections: PC and AD

Preconditioning:

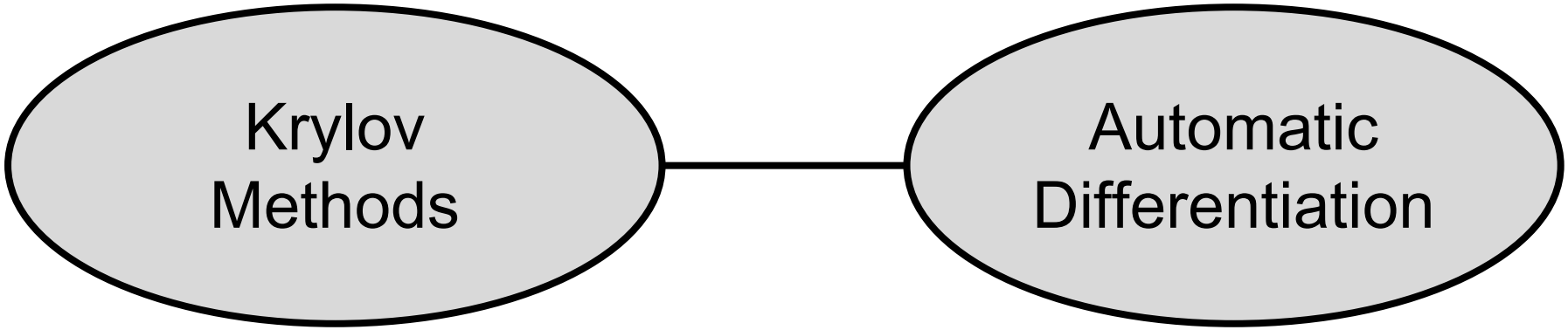
- Access to elements $A(i, j)$
- Access to chunks of rows/columns

Automatic Differentiation:

- Access to complete rows/columns
- Access to groups of complete rows/columns

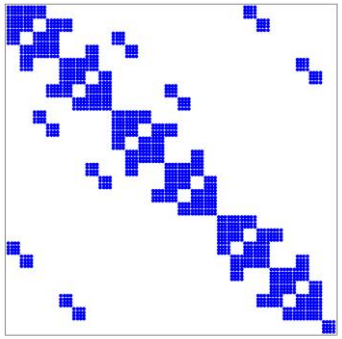
$\Rightarrow$ No work on preconditioning with AD

Connections between Areas



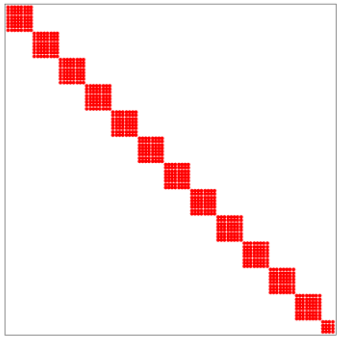
Main Idea

A

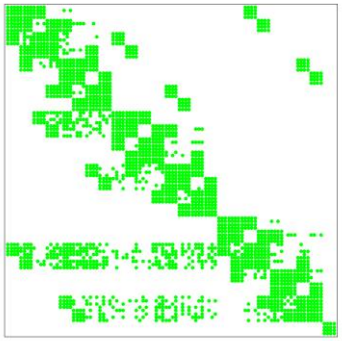


sparsification

$\tilde{A}$

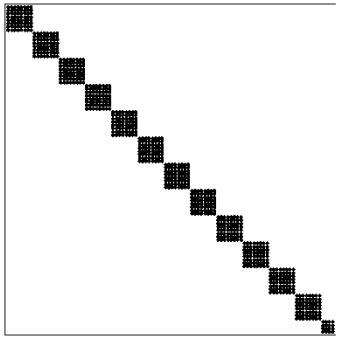


preconditioning



$M \approx A$

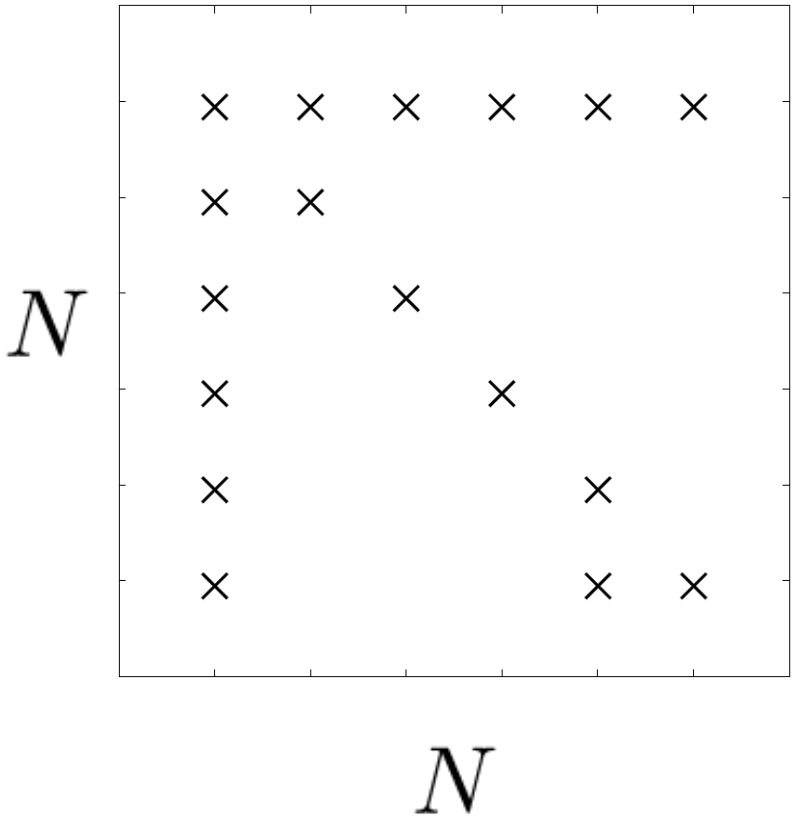
preconditioning



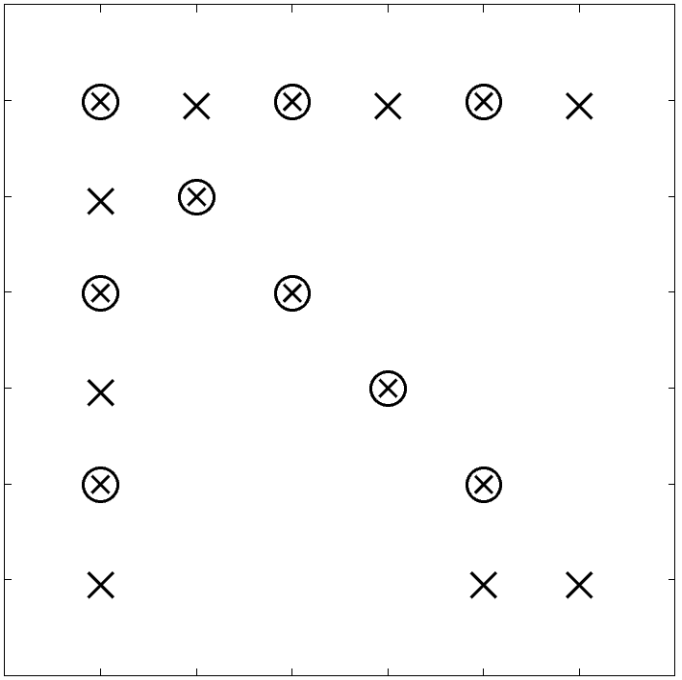
$\tilde{M} \approx \tilde{A}$

Subset of Nonzeros

Full

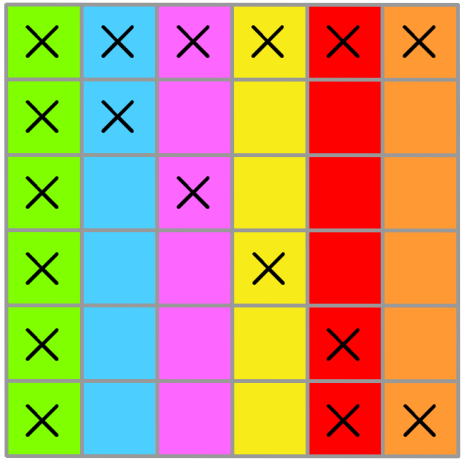
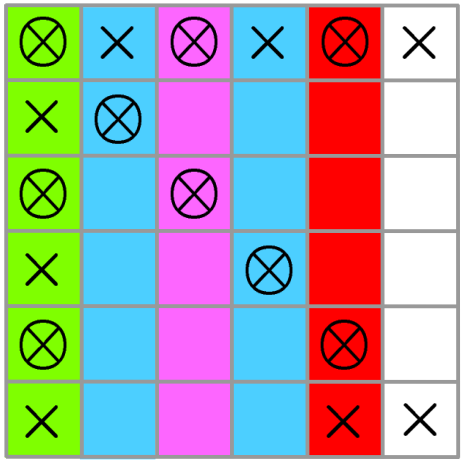
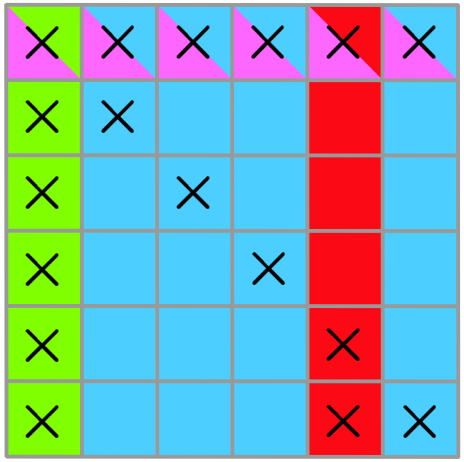
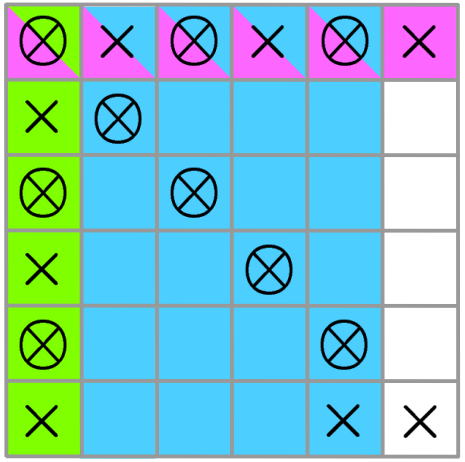


Partial



⊗ „Required Elements“

Minimal Number of Colors

	Full	Partial
Uni-directional	 $k = N$	 $k = N/2+1$
Bi-directional	 $k = 4$	 $k = 3$

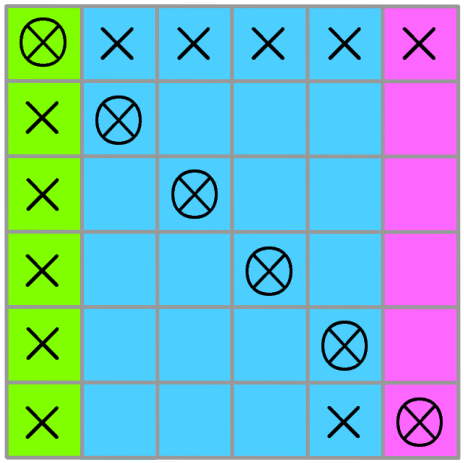
Bipartite Graph

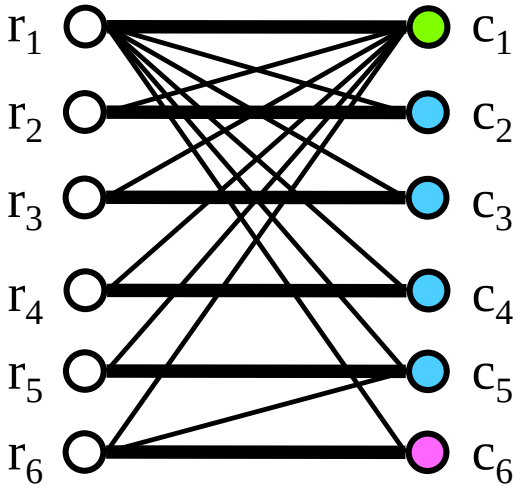
$$G(A) = (V_r, V_c, E)$$

$$V_r = \{r_i \mid 1 \leq i \leq N\}$$

$$V_c = \{c_i \mid 1 \leq i \leq N\}$$

$$E = \{(r_i, c_j) \mid A(i,j) \neq 0\}$$

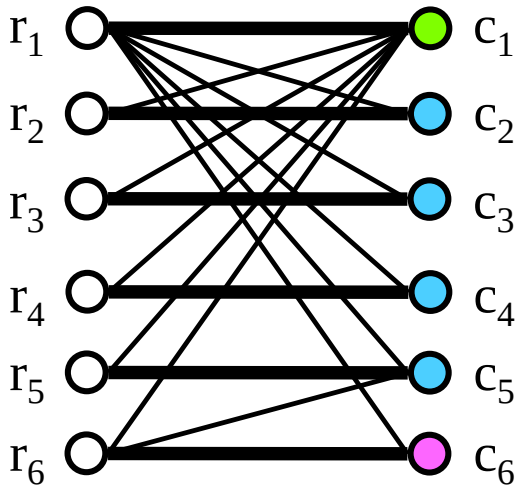




[Gebremedhin, Manne, Pothen] *SIAM Review* 2005

Diagonal Preconditioning

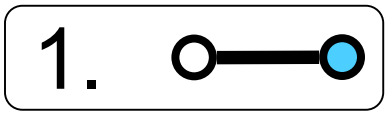
$$E_D = \{(1,1), \ldots, (N,N)\}$$



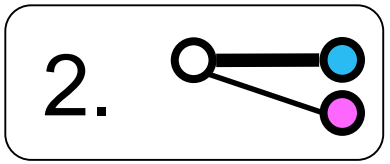
Restricted Distance-2 Coloring

- $G(A) = (V_r, V_c, E)$, Required Elements: Diagonal $E_D = \{(1,1), \ldots, (N,N)\}$
- Mapping $\phi_c : V_c \longrightarrow \{1, \ldots, k\}$ is called *distance-2 coloring restricted to* E_D

1. $\forall\, (r_i, c_i) \in E_D$ gilt $\phi_c(c_i) \neq 0$



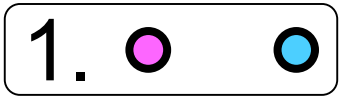
2. Distance-2 neighbors c_i and c_j have different colors, if $\exists$ path of length 2 with at least one edge in E_D



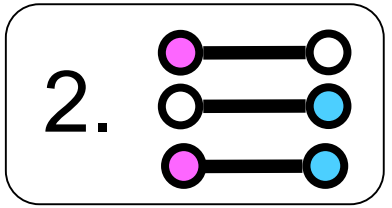
Restricted Star Bicoloring (i)

- $G(A) = (V_r, V_c, E)$, Required Elements: Diagonal $E_D = \{(1,1), \ldots, (N,N)\}$
- Mapping $\phi : V_r \cup V_c \longrightarrow \{1, \ldots, k\}$ is called *star bicoloring restricted to* E_D

1. Different colors for nodes from V_r and V_c (except 0)



2. At least one node incident to $(r_i, c_i) \in E_D$ is colored different from zero

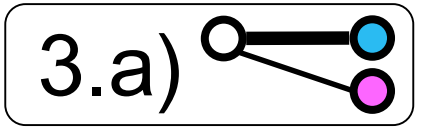


Restricted Star Bicoloring (ii)

3. $\forall\,(r_i, c_i) \in E_D$:

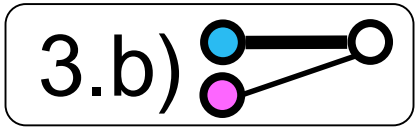
   a) If $\phi(r_i) = 0$, then

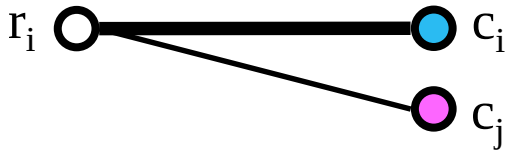
   $$\phi(c_i) \neq \phi(c_j) \quad \forall\,(c_i, r_i, c_j)$$

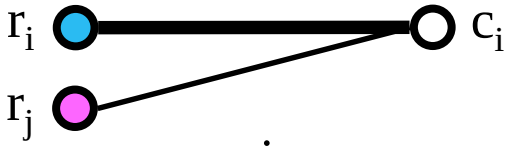


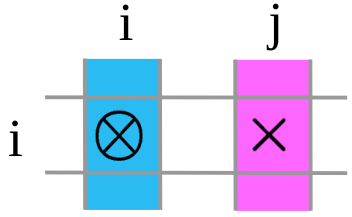
   b) If $\phi(c_i) = 0$, then

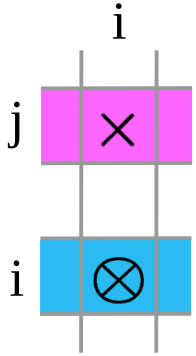
   $$\phi(r_i) \neq \phi(r_j) \quad \forall\,(r_i, c_i, r_j)$$











Restricted Star Bicoloring (iii)

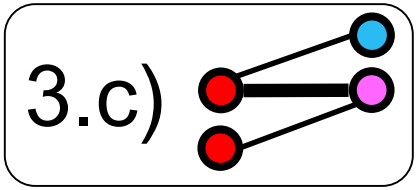
3. $\forall\,(r_i, c_i) \in E_D$:

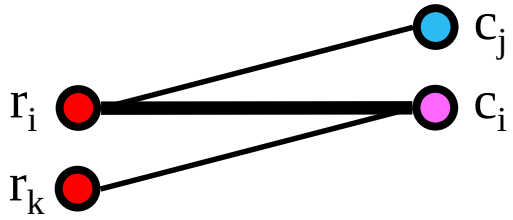
   c) If $\phi(r_i) \neq 0$ and $\phi(c_i) \neq 0$, then

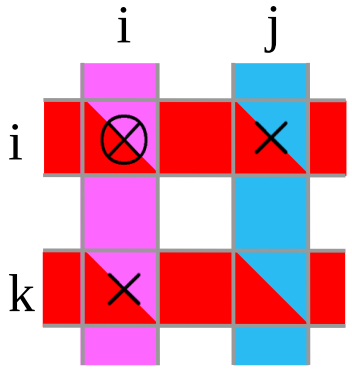
   $\forall\,(c_j, r_i, c_i, r_k)$

   either $\phi(r_i) \neq \phi(r_k)$

   or $\phi(c_i) \neq \phi(c_j)$







Bi- is no better than Unidirectional

Theorem (B, Lülfesmann, Wolf):

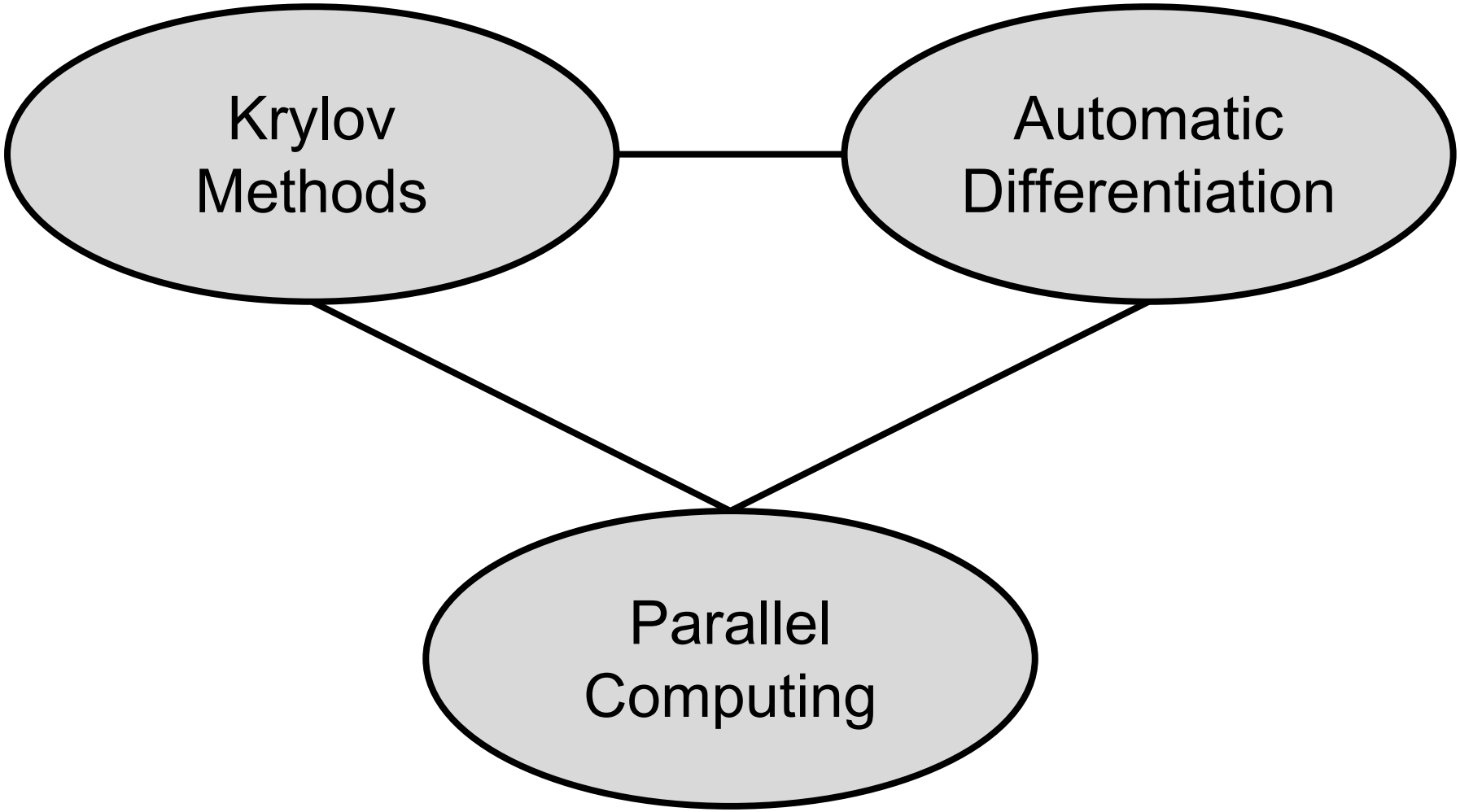
Let $G(A) = (V_r, V_c, E)$

with $E_D = \{(1,1), \ldots, (N,N)\}$.

Minimal number of colors is equal for

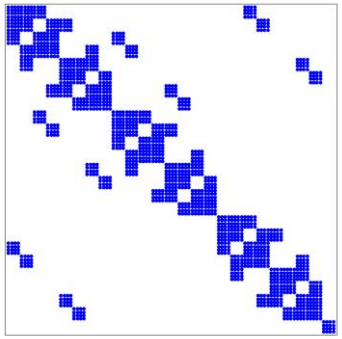
- Distance-2 coloring restricted to E_D
- Star bicoloring restricted to E_D

Research Areas



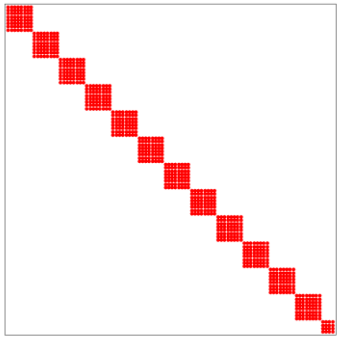
Main Idea

A

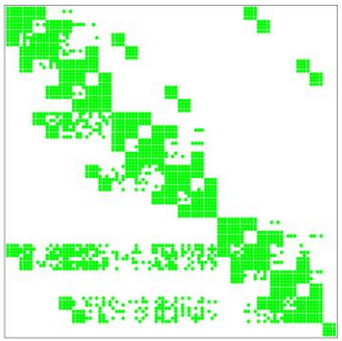


sparsification

$\tilde{A}$

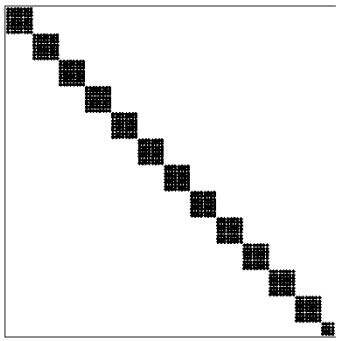


preconditioning



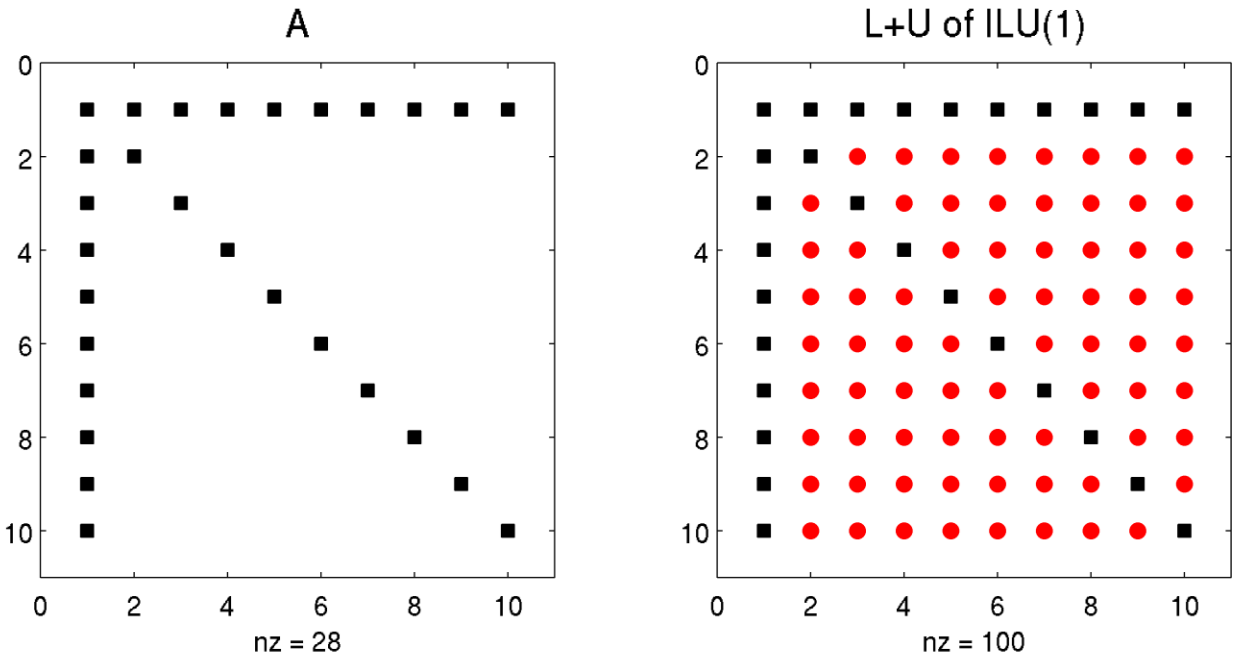
$M \approx A$

preconditioning

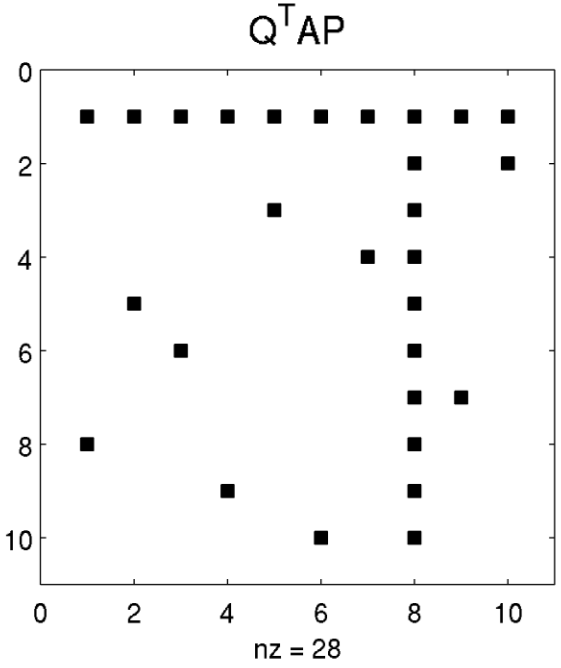


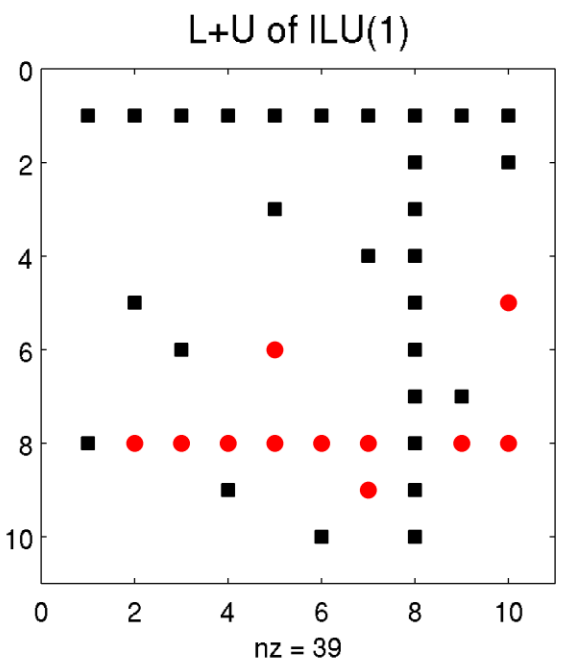
$\tilde{M} \approx \tilde{A}$

ILU(L): Influence of Ordering

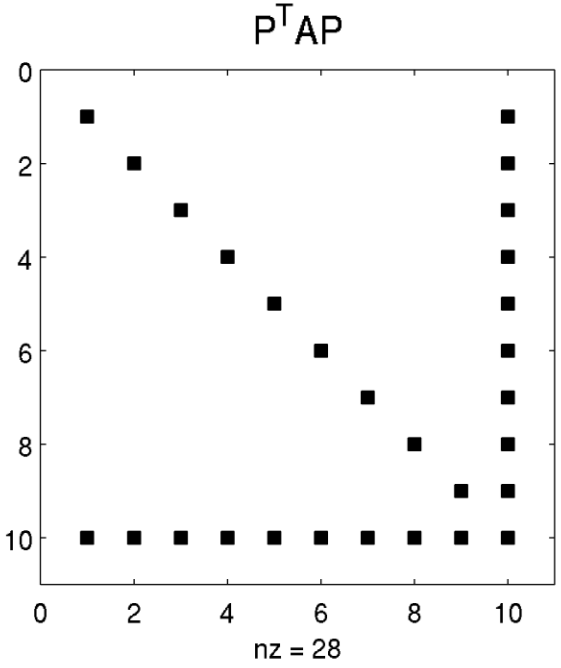


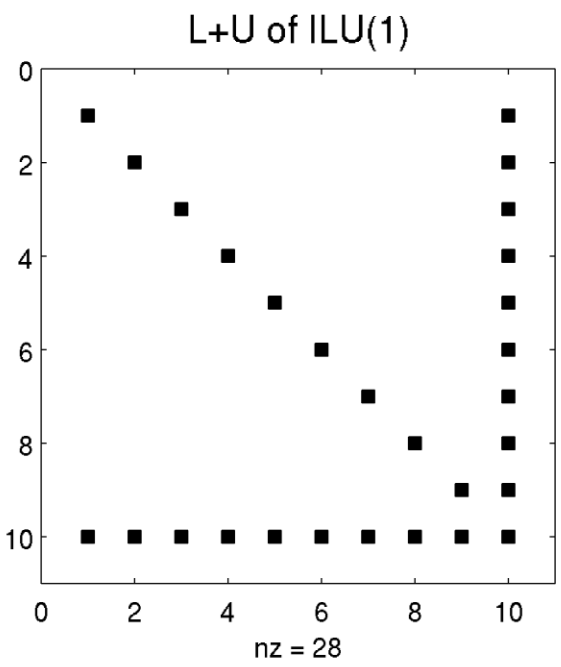
ILU(L): Influence of Ordering



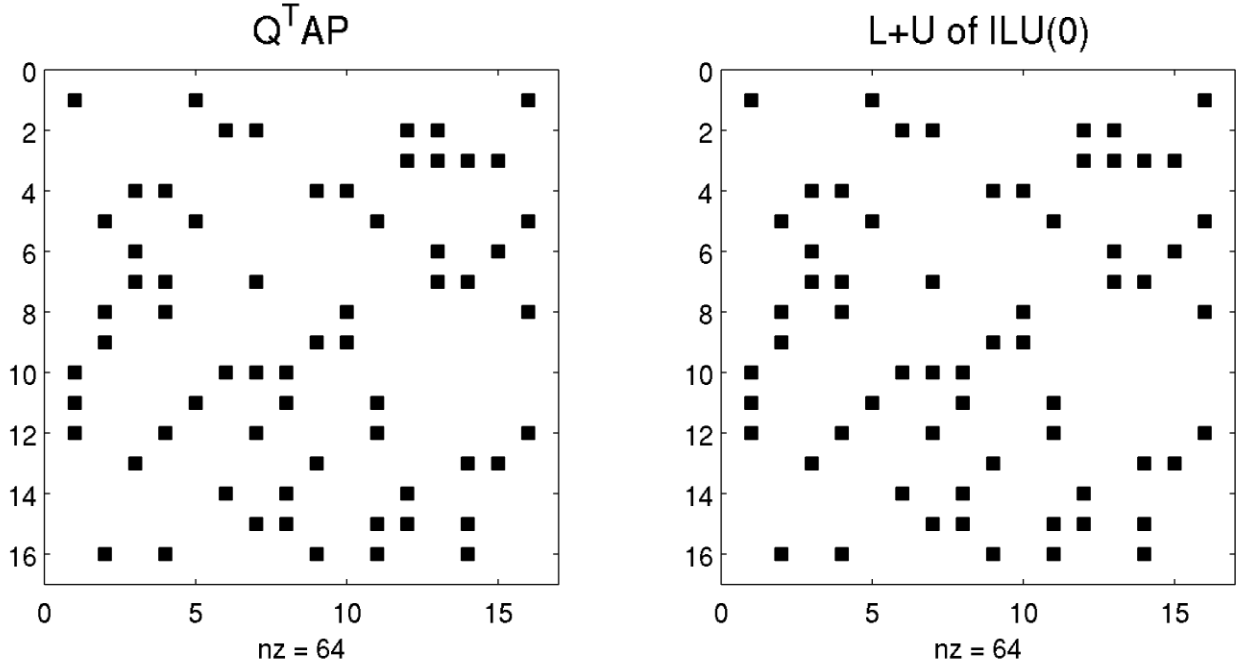


ILU(L): Influence of Ordering

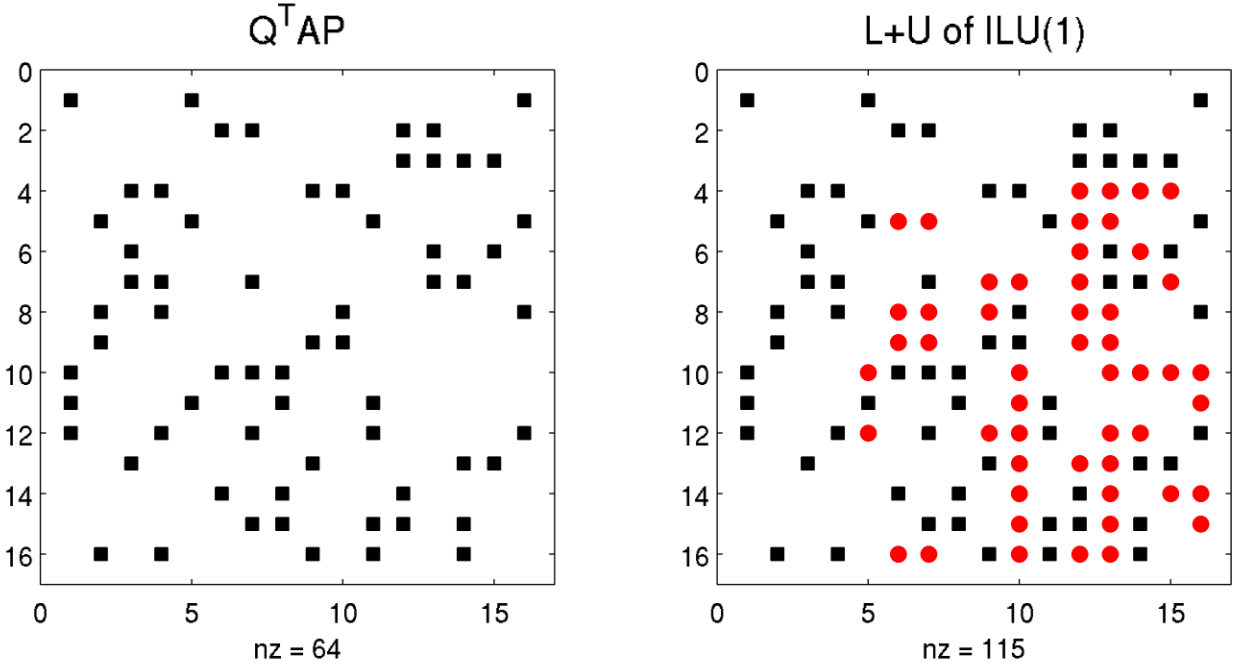




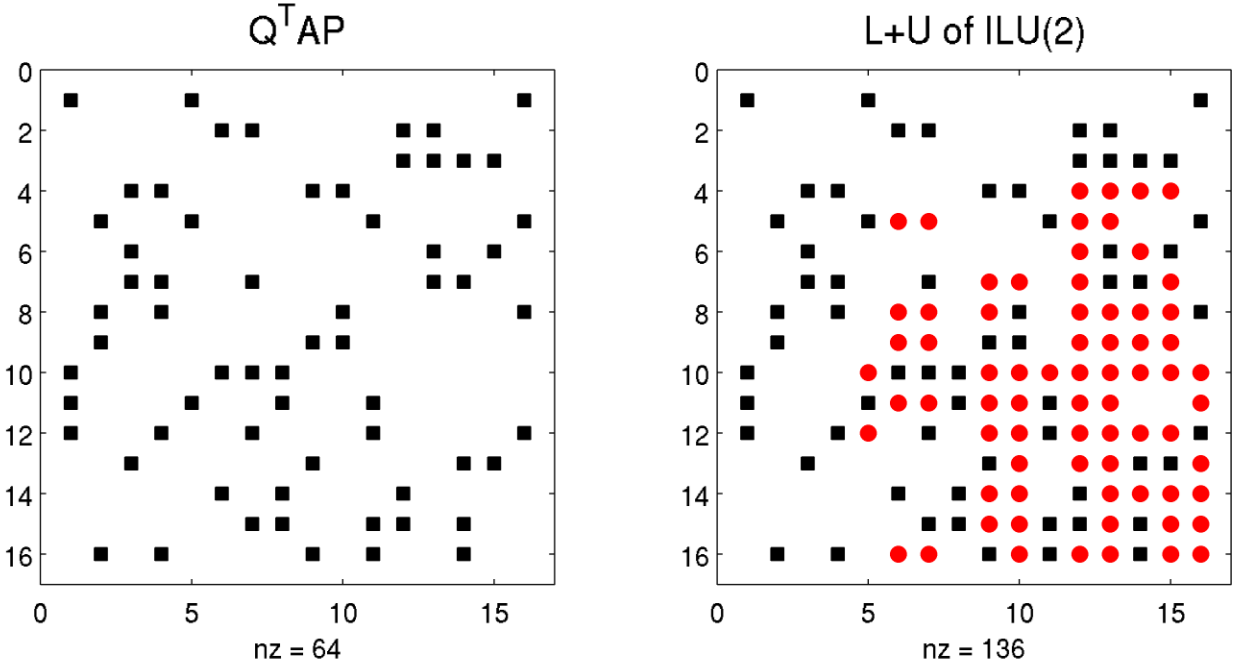
ILU(L): Influence of Level



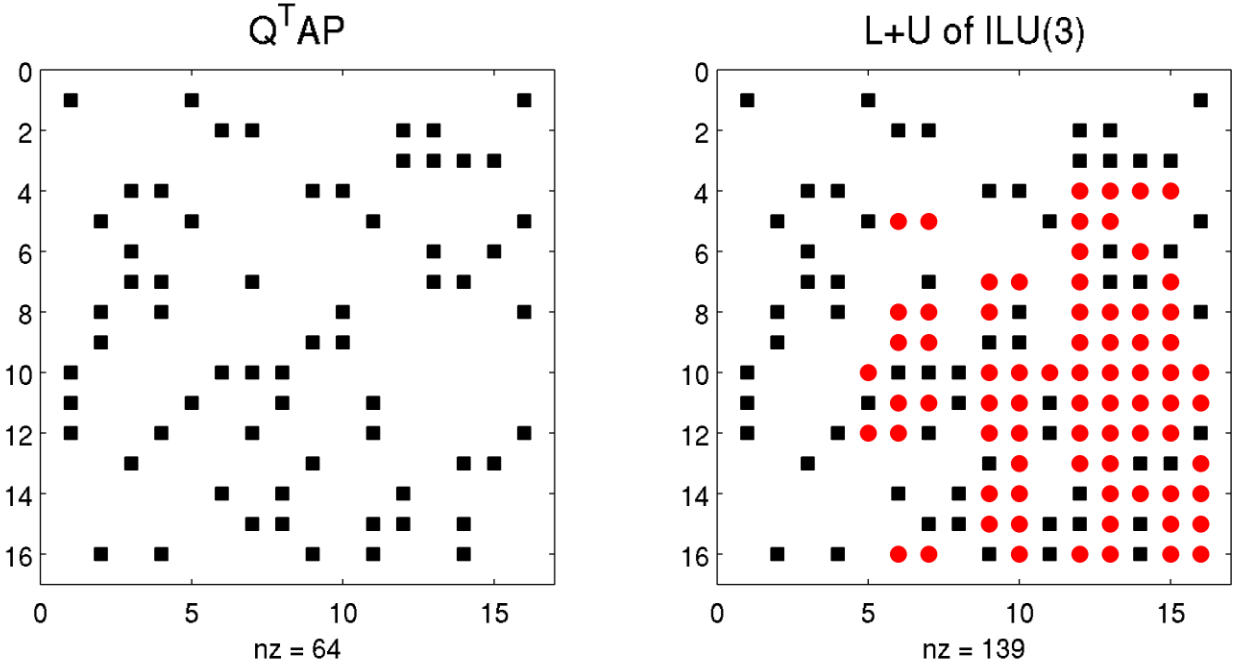
ILU(L): Influence of Level



ILU(L): Influence of Level

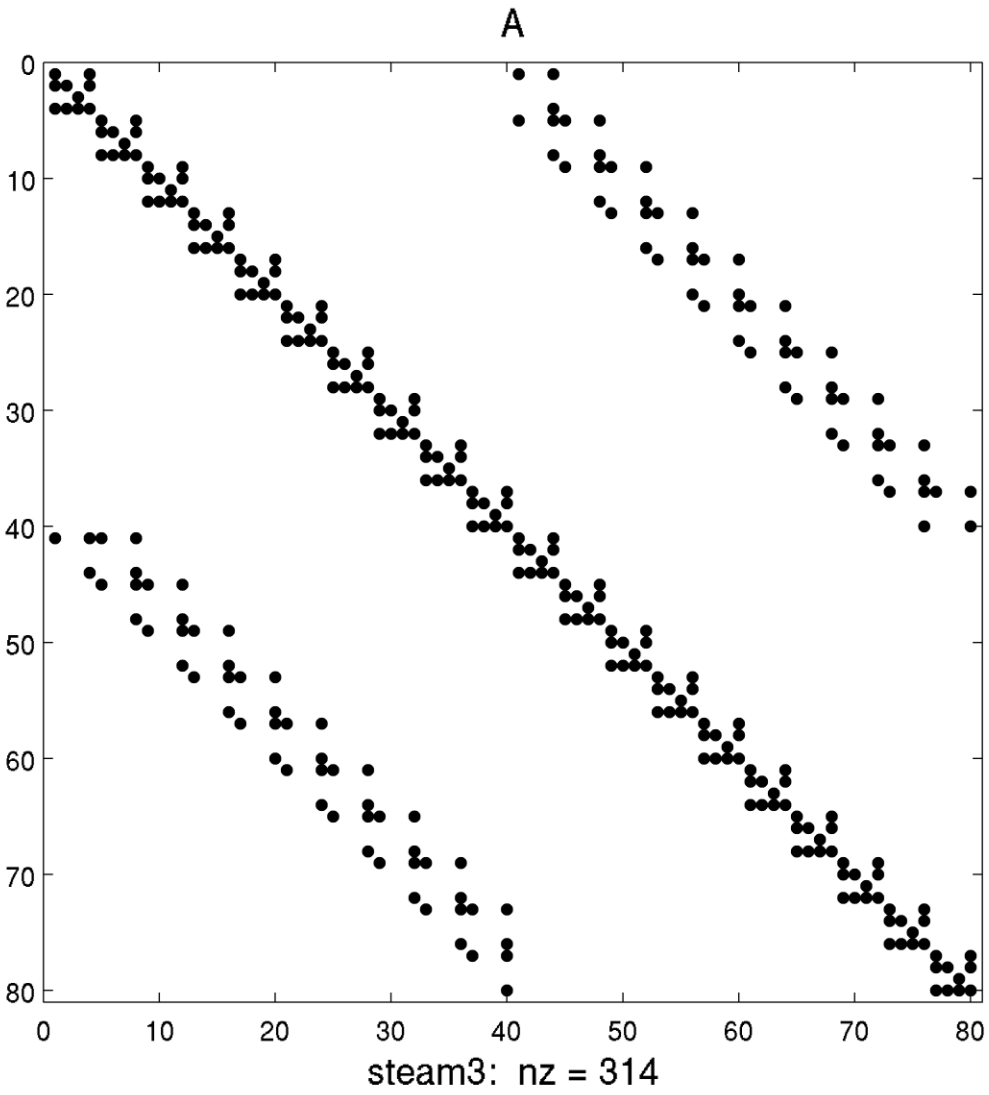


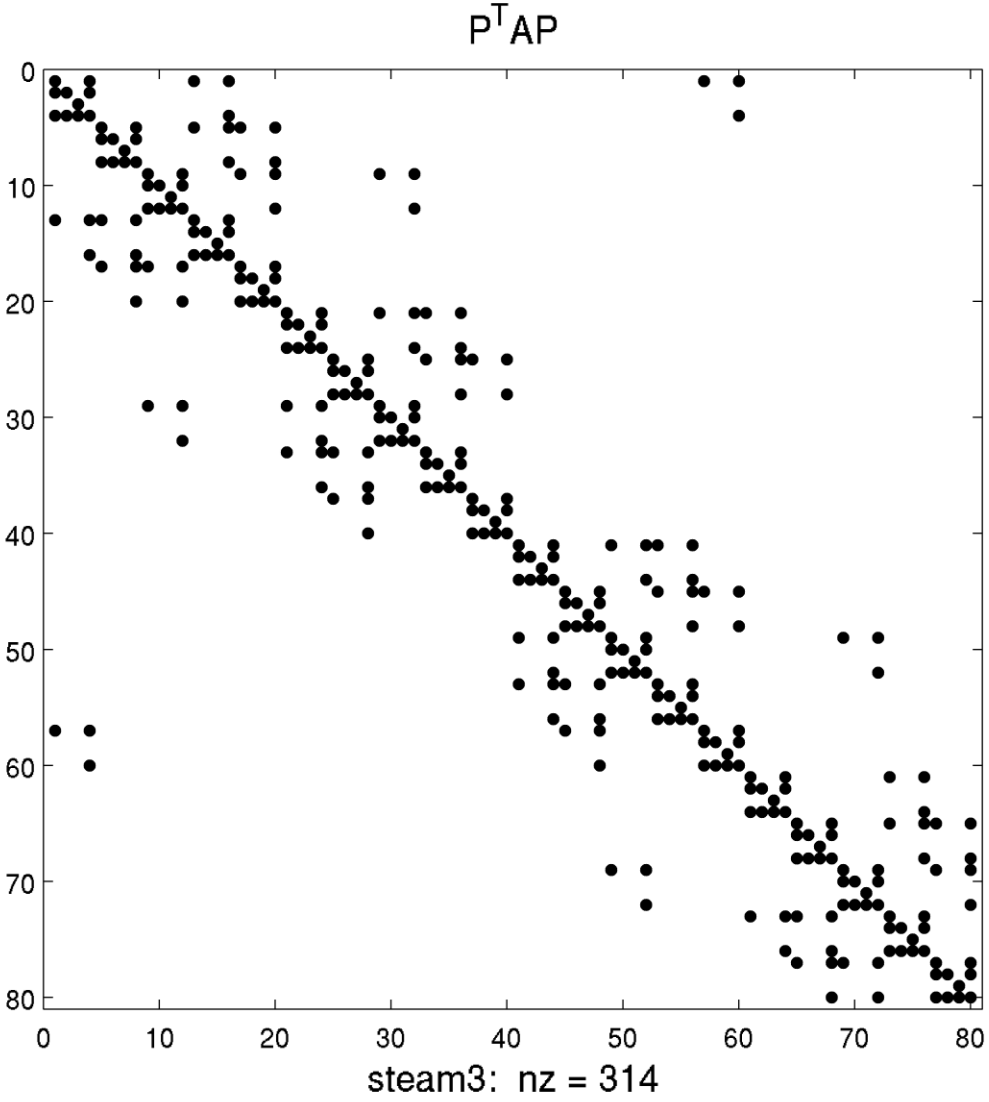
ILU(L): Influence of Level

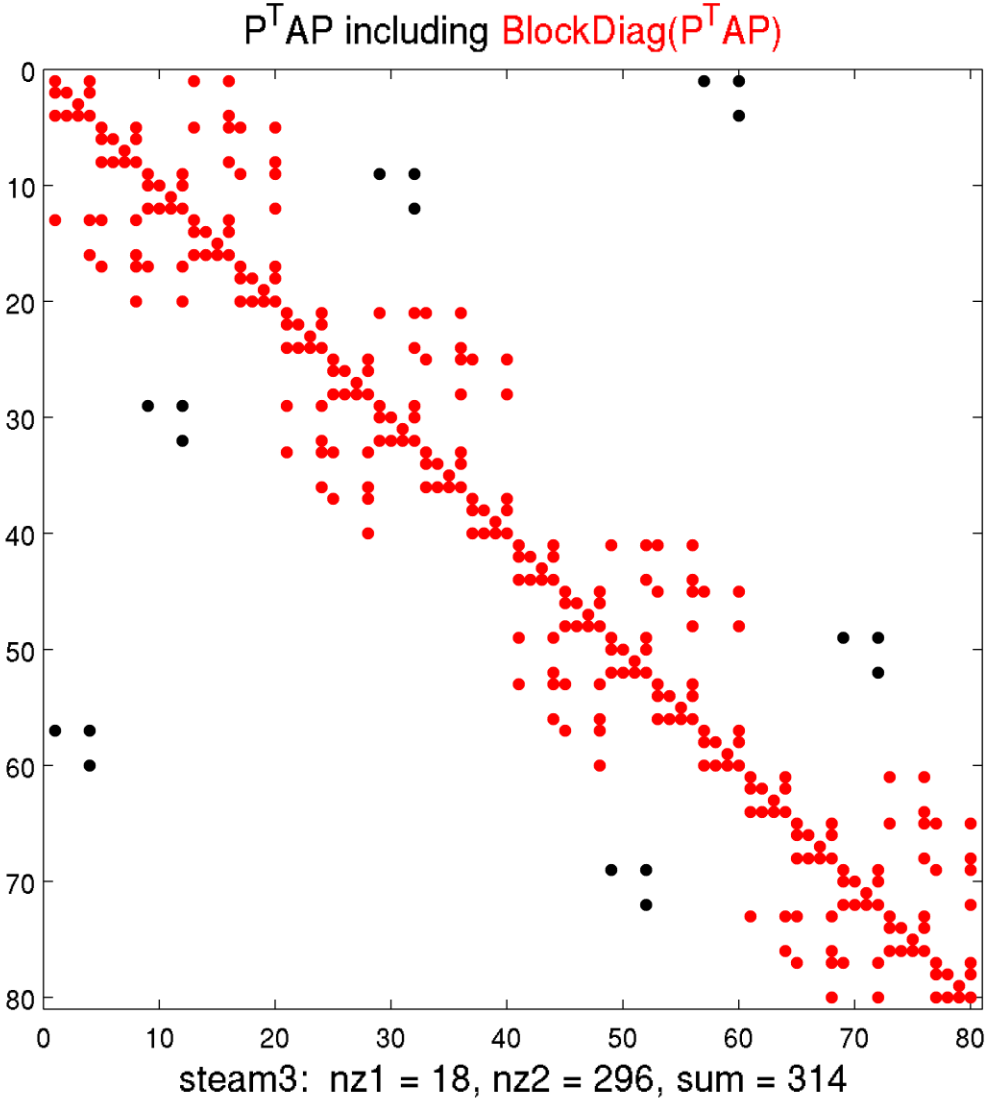


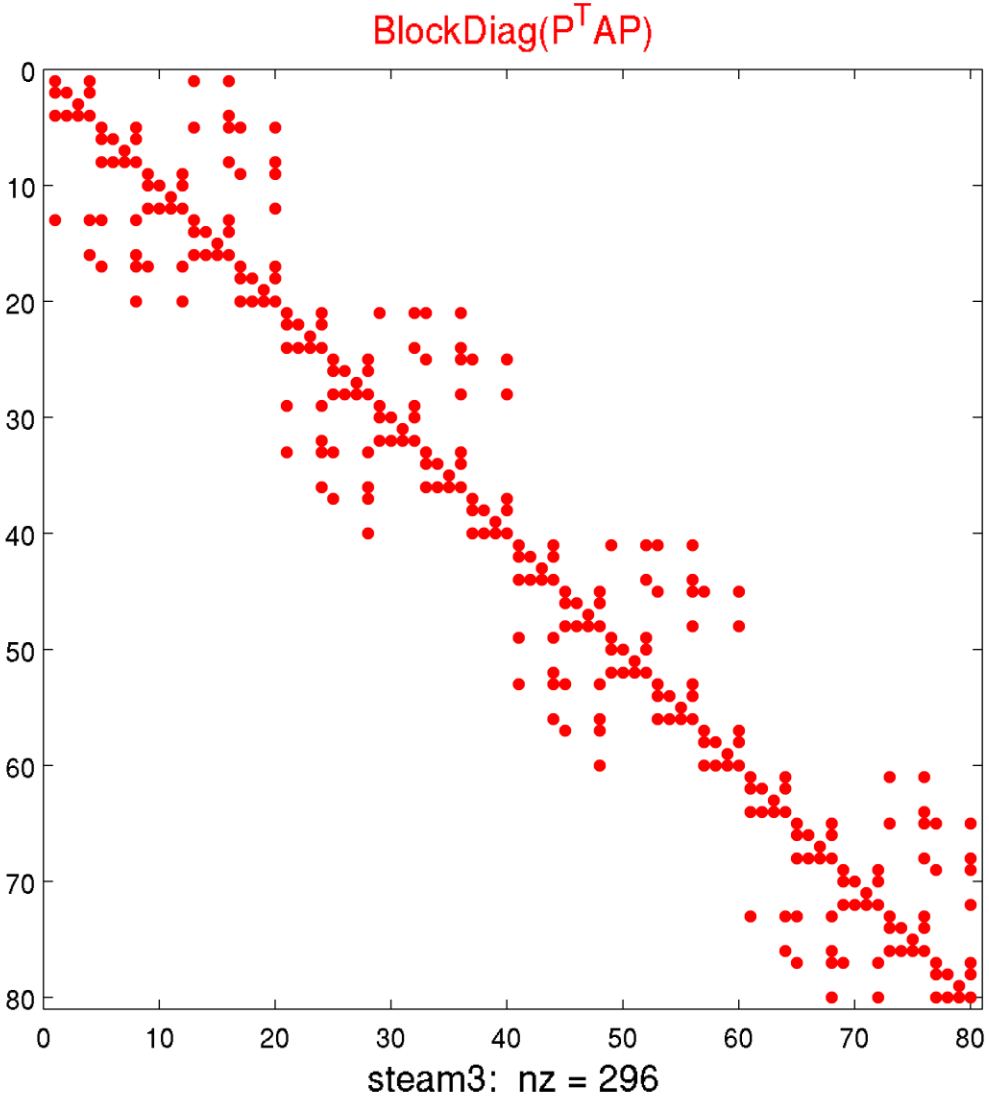
Sparsification & Parallel

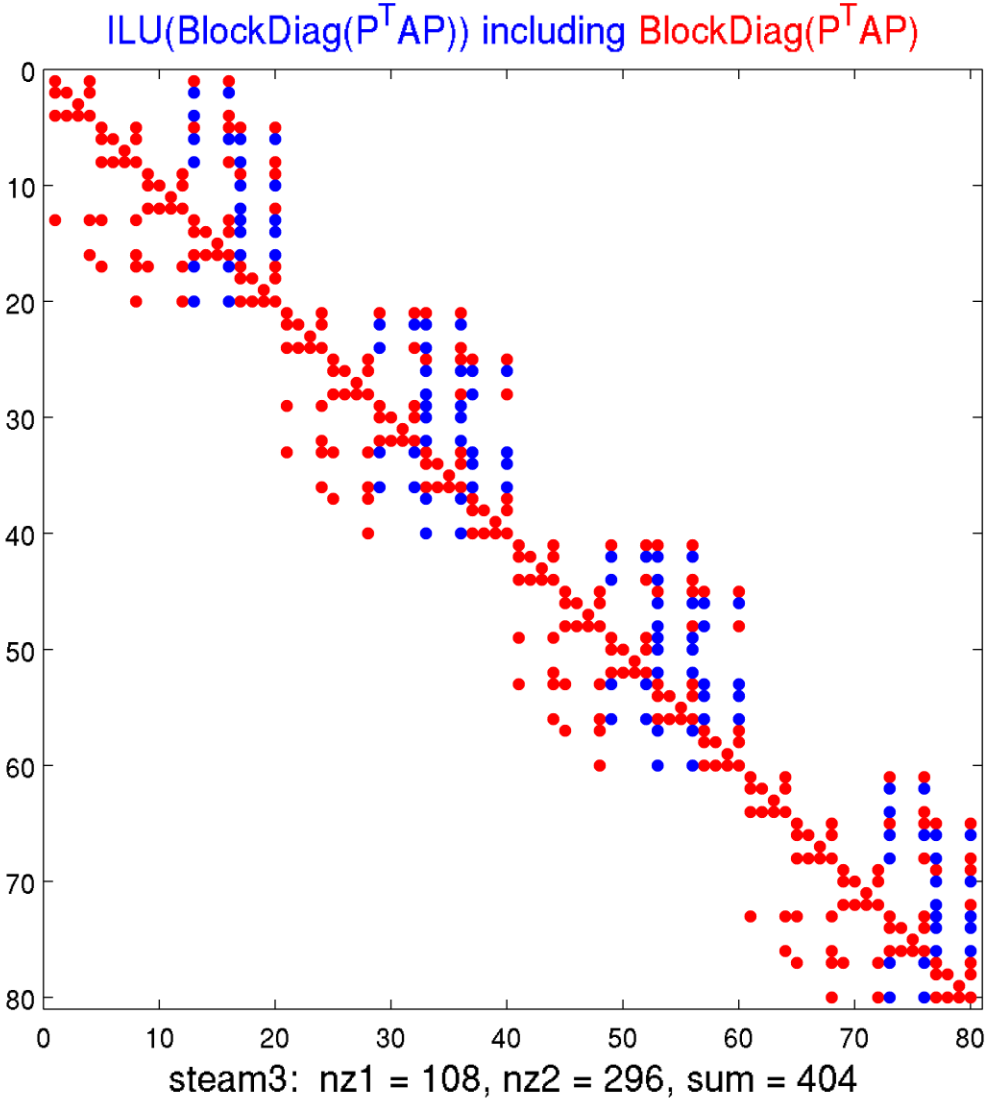
- Permute such that many nonzeros gather around diagonal
- Sparsify in form of block diagonal
- Apply ILU(L) to each block
- Advantages:
  - Parallel construction of M
  - Parallel application $M\mathbf{y} = \mathbf{z}$


A
0
10
20
30
40
50
60
70
80
0
10
20
30
40
50
60
70
80
steam3: nz = 314


P^TAP
0
10
20
30
40
50
60
70
80
0
10
20
30
40
50
60
70
80
steam3: nz = 314


P^T AP including BlockDiag(P^T AP)
0
10
20
30
40
50
60
70
80
steam3: nz1 = 18, nz2 = 296, sum = 314

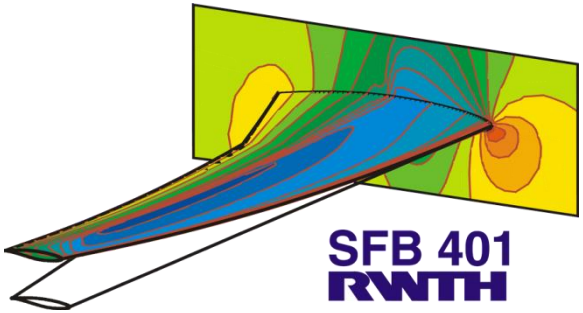

BlockDiag(P^T AP)
0
10
20
30
40
50
60
70
80
steam3: nz = 296


ILU(BlockDiag(P^T AP)) including BlockDiag(P^T AP)
0
10
20
30
40
50
60
70
80
steam3: nz1 = 108, nz2 = 296, sum = 404

QUADFLOW

Josef Ballmann
Mechanik

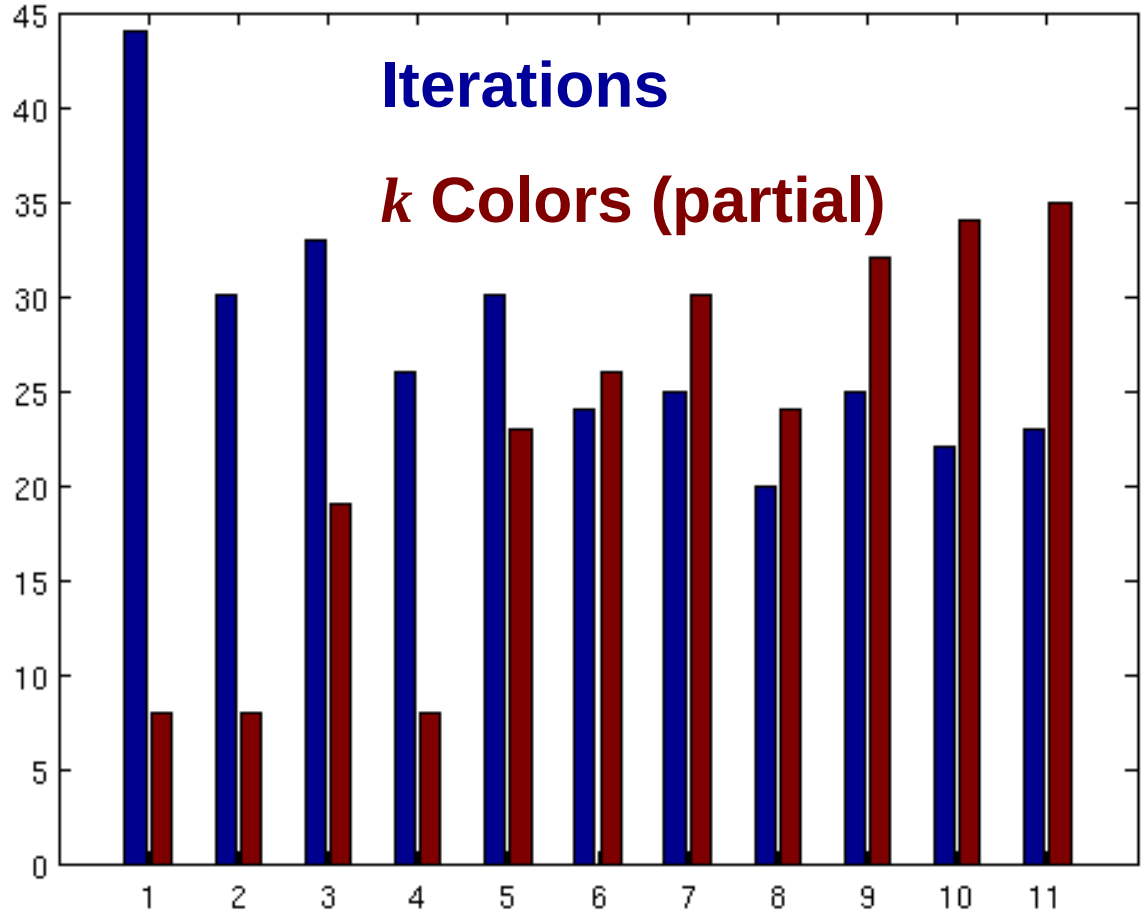
LFM RWTH RHEINISCH-WESTFÄLISCHE TECHNISCHE HOCHSCHULE AACHEN



- Finite volume
- Implicit time integration
- Unstructured grid
- Adaptation via Multiscale analysis (Wolfgang Dahmen,Siegfried Müller)

Institut für Geometrie und Praktische Mathematik igpm

Results for Blockdiagonal PC

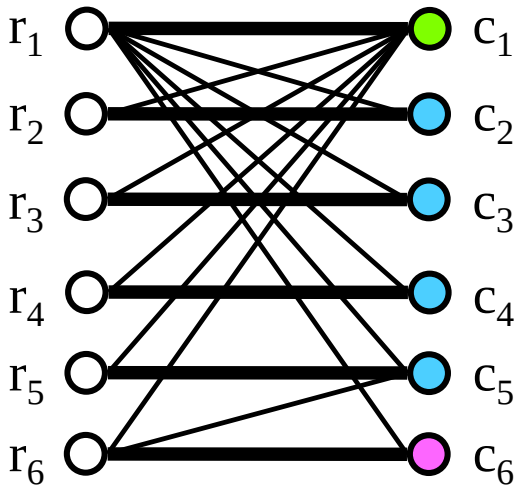


Without PC:
177 Iterations

Full:
$k = 36$

- GMRES
- ILU(0)

Missing Connection: Coloring and ILU




```
for i = 1:n
    % All entries of A are entries of F with level = 0.
    index_nnz = find( A(i,:)==1 );
    level(i,index_nnz) = 0;
    F(i,index_nnz) = 1;
    % Update phase
    for k = 1:i-1  % node (i,k) with k<i
        if F(i,k)==1
            for j = k + find( F(k,k+1:n)==1 ) % node (k,j) with k<j
                lev = level(i,k) + level(k,j) + 1;
                if(lev <= el)
                    if F(i,j)==0
                        F(i,j) = 1;
                        level(i,j) = lev;
                    else
                        level(i,j) = min(level(i,j),lev);
                    end;
                end
            end;
        end
    end
end
```

Symbolic ILU on bipartite graph

$$G(A) = (V_r, V_c, E) \qquad V_r = \{r_i \mid 1 \leq i \leq N\}$$

$$V_c = \{c_i \mid 1 \leq i \leq N\}$$

$$E = \{(r_i, c_j) \mid A(i,j) \neq 0\}$$

$$\sigma : E \longrightarrow \mathbb{N}$$

$$\sigma\big((r_i, c_j)\big) = \mathsf{lev}(i,j)$$

Path $(r_i, c_k, r_k, c_l, r_l, \ldots, c_j)$ is called *fill path from* r_i *to* c_j, if for all indices m of inner nodes the inequality $m < \min(i,j)$ holds.

Sequence of Graphs for ILU(L)

$$G_\nu = (V_r, V_c, E_\nu), \quad \nu = 0, 1, \ldots, N$$

$$G_0 = G(A) \qquad \forall (r_i, c_j) \in E_0 = E$$
$$\sigma\big((r_i, c_j)\big) = 0$$

$$G_{\nu+1} = (V_r, V_c, E_{\nu+1})$$

$$E_{\nu+1} = E_\nu \cup \{\exists \text{ Fill path } (r_i, c_k, r_k, c_j)$$
of length 3 in G_ν and
$$t(k) := \mathsf{lev}_\nu(i,k) + \mathsf{lev}_\nu(k,j) + 1 \leq L\}$$

$$\sigma\big((r_i, c_j)\big) = \mathsf{lev}_{\nu+1}(i,j) = \min_k(\mathsf{lev}_\nu(i,j), t(k))$$

„Static“ Determination of Fill-in

Lemma:

$(r_i, c_j) \in E_N$ in G_N with lev$(i, j) = k$

$\Leftrightarrow$

$\exists$ shortest fill path of length $2k + 1$ between r_i and c_j in G_0

New Algorithm (PC + AD + Par)

- Compute $R_{\text{init}} = \rho(A)$
- Compute $R_{\text{init}} \dot{\cup} F = \mathsf{SILU}(R_{\text{init}})$
- Color $\phi(R_{\text{init}}, A)$
- Compute $R_{\text{pot}} \subset A \setminus R_{\text{init}}$ such that

$$|\phi(R_{\text{init}}, A)| = |\phi(R_{\text{init}} \dot{\cup} R_{\text{pot}}, A)|$$

- Compute $R_{\text{add}} \subseteq R_{\text{pot}}$ such that

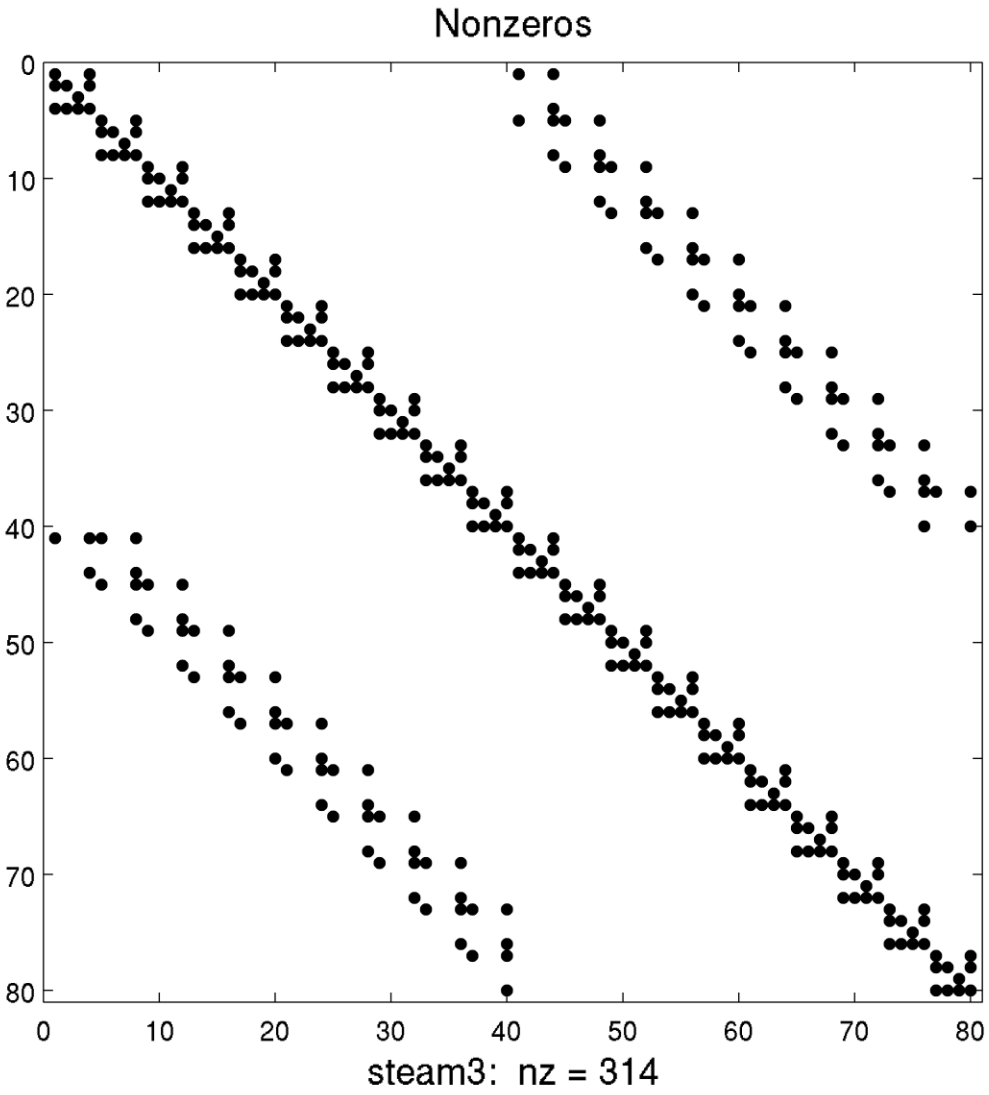
$$\mathsf{SILU}(R_{\text{init}}) \cup R_{\text{add}} = \mathsf{SILU}(R_{\text{init}} \dot{\cup} R_{\text{add}})$$

- Compute $[L, U] = \mathsf{NILU}(R_{\text{init}} \dot{\cup} R_{\text{add}})$

steam3

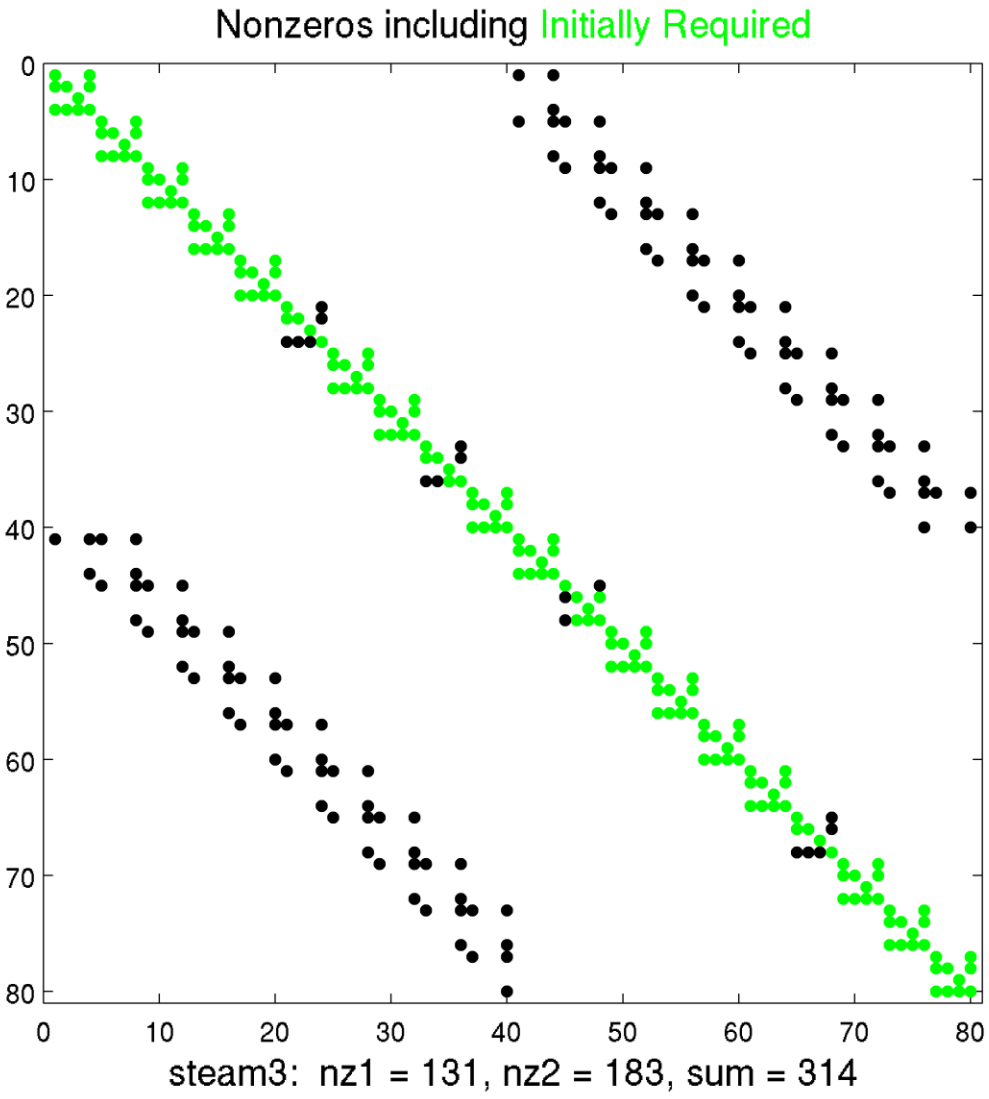
- Visualization of idea
- Small example without Fill-in

steam3


Nonzeros
0
10
20
30
40
50
60
70
80
0
10
20
30
40
50
60
70
80
steam3: nz = 314

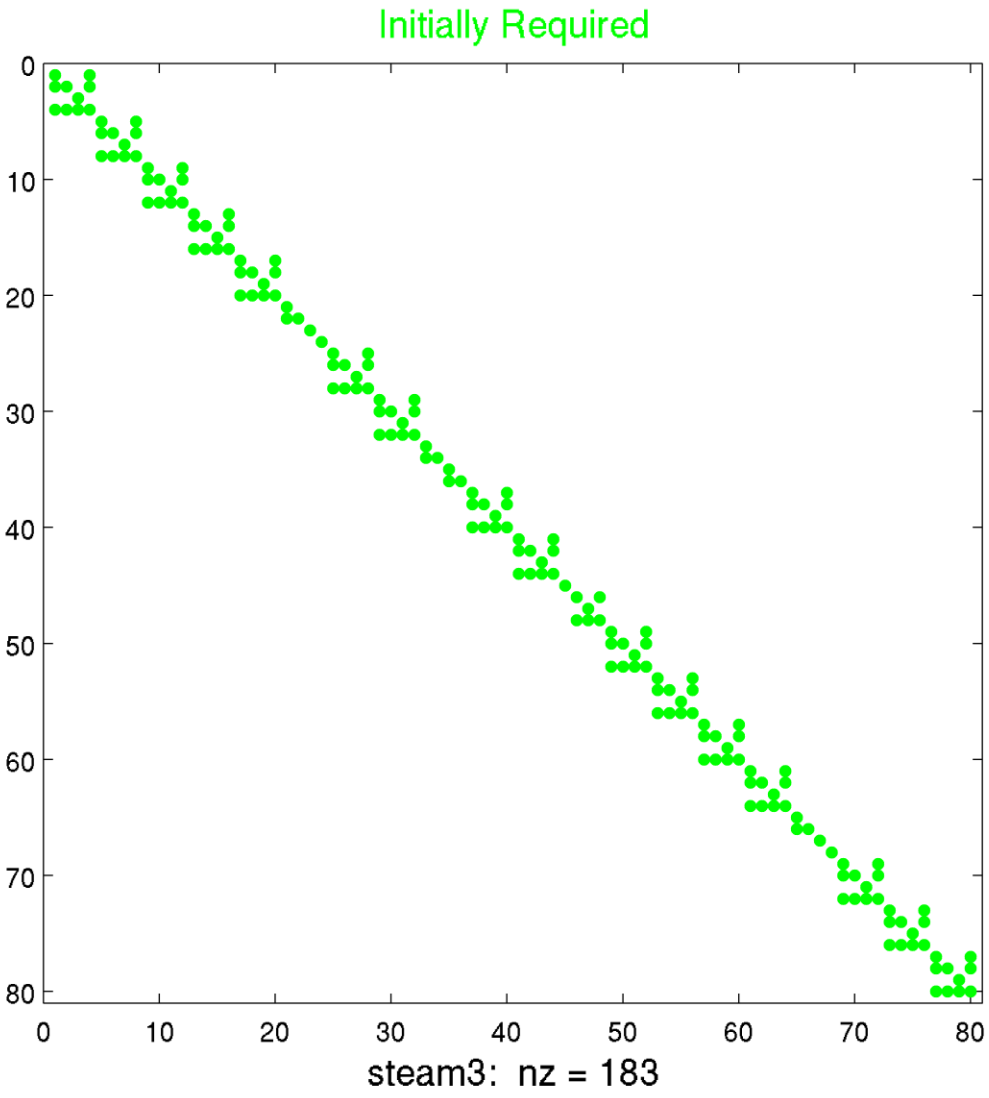
A

steam3



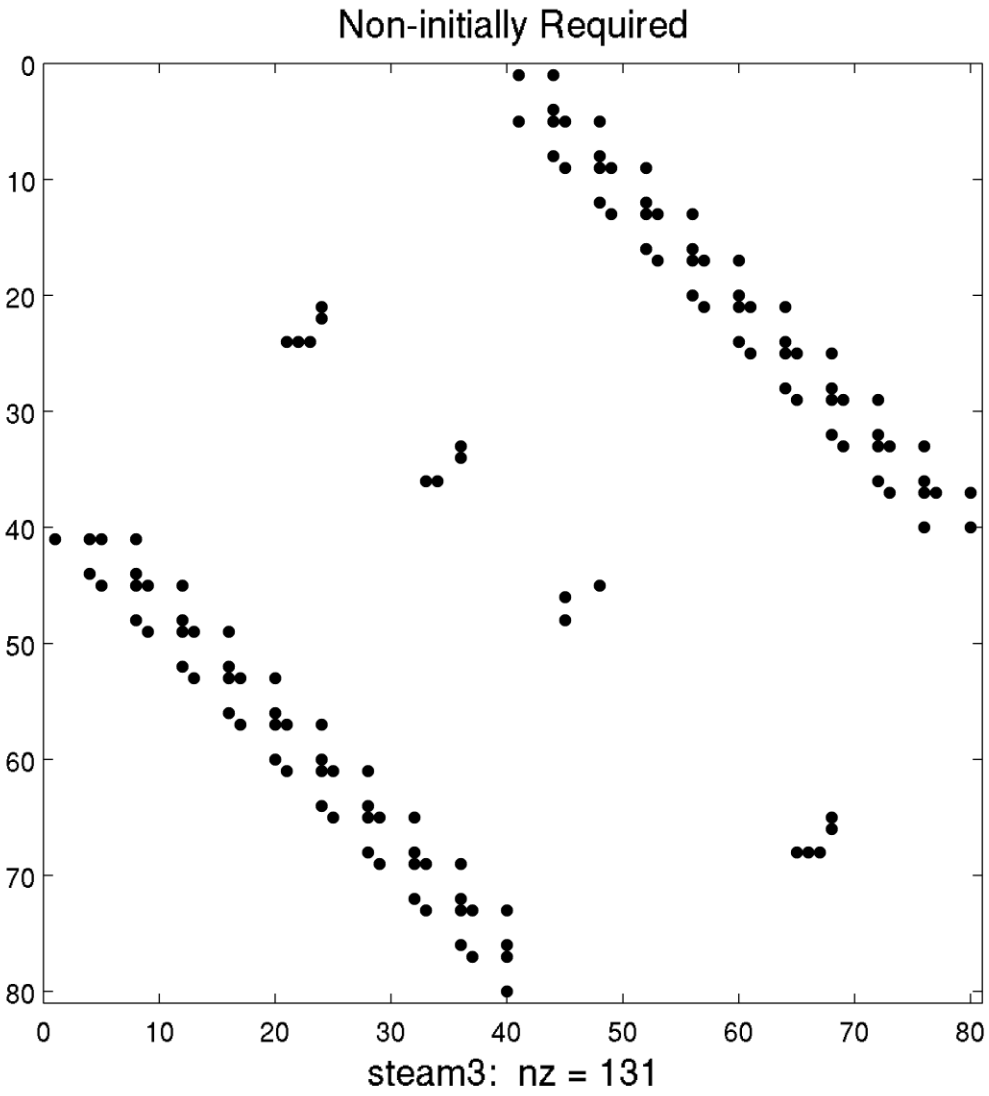
R_{init}

steam3



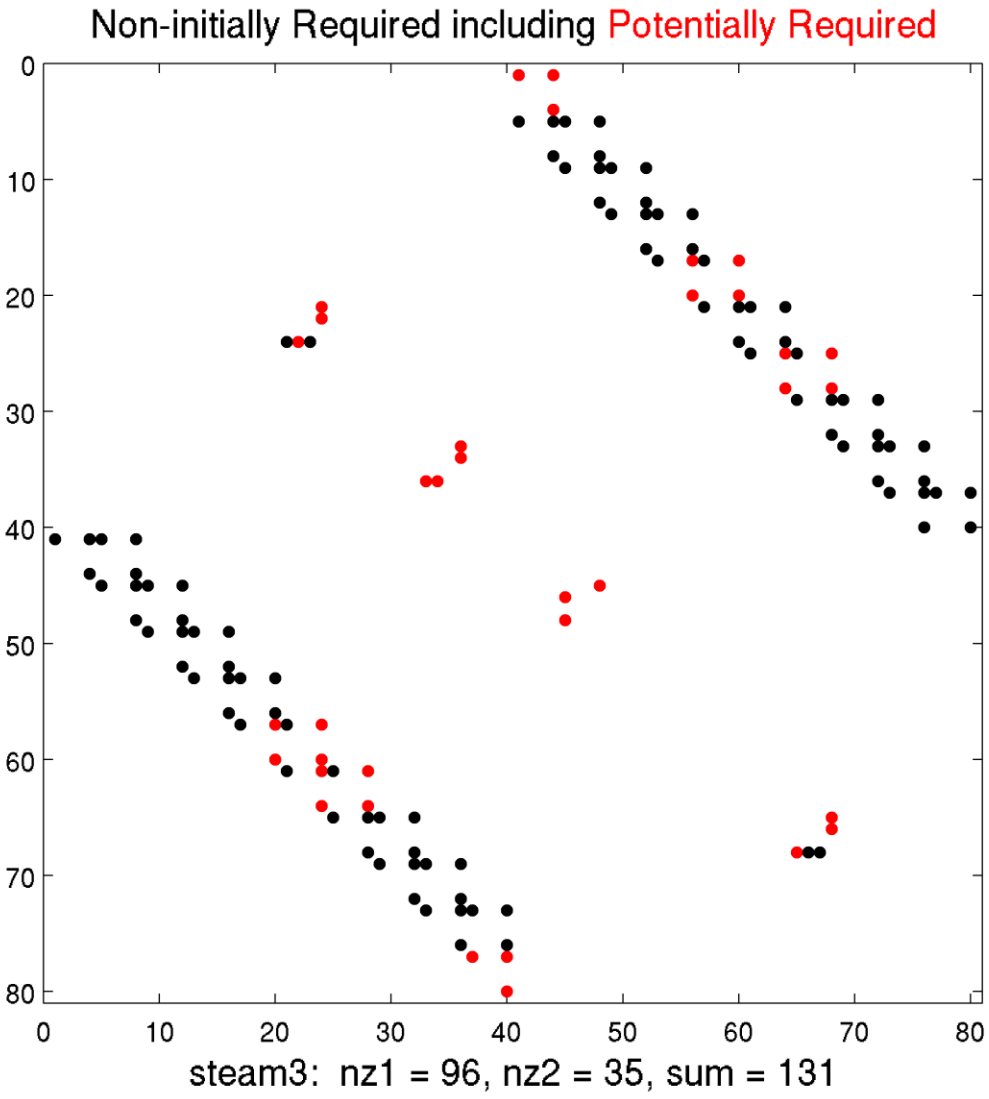
R_{init}

steam3

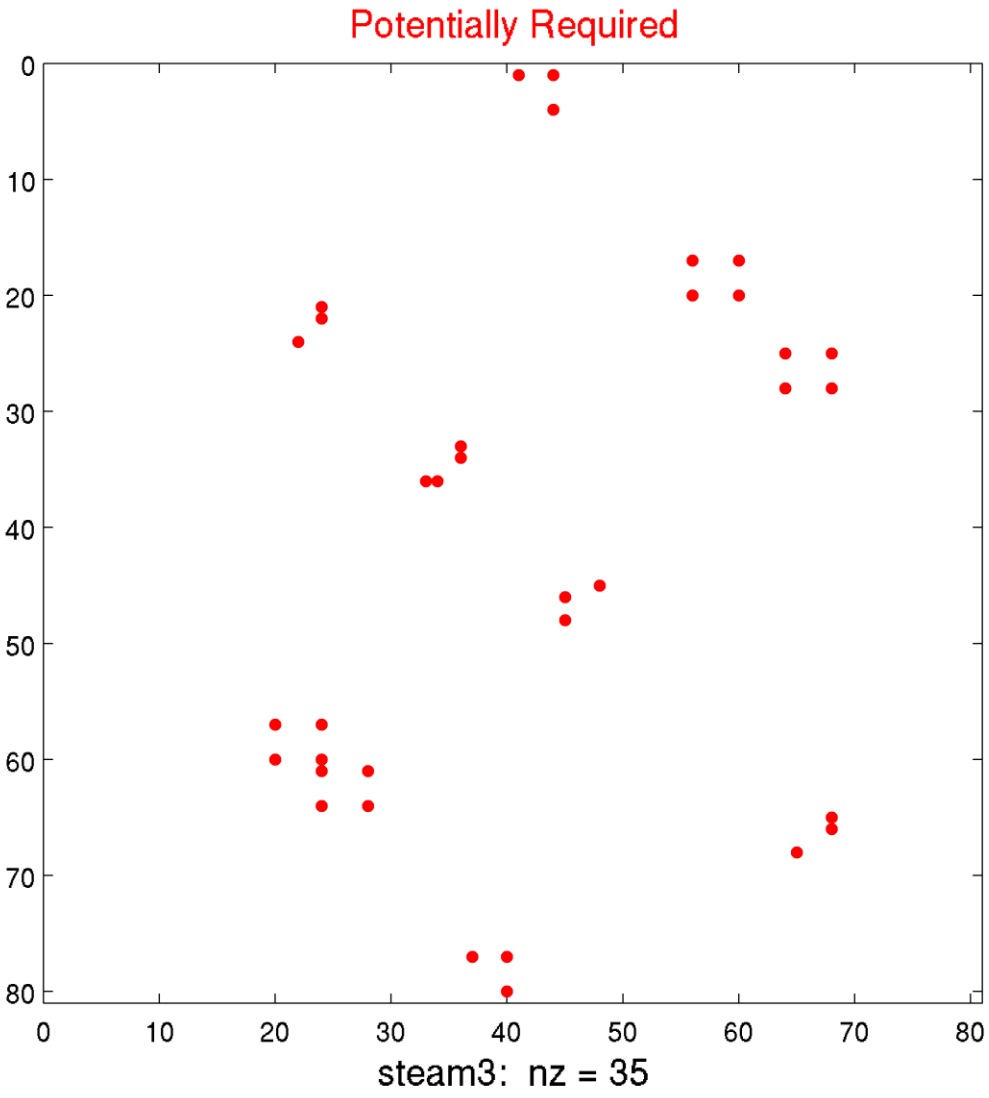


$A \setminus R_{\text{init}}$

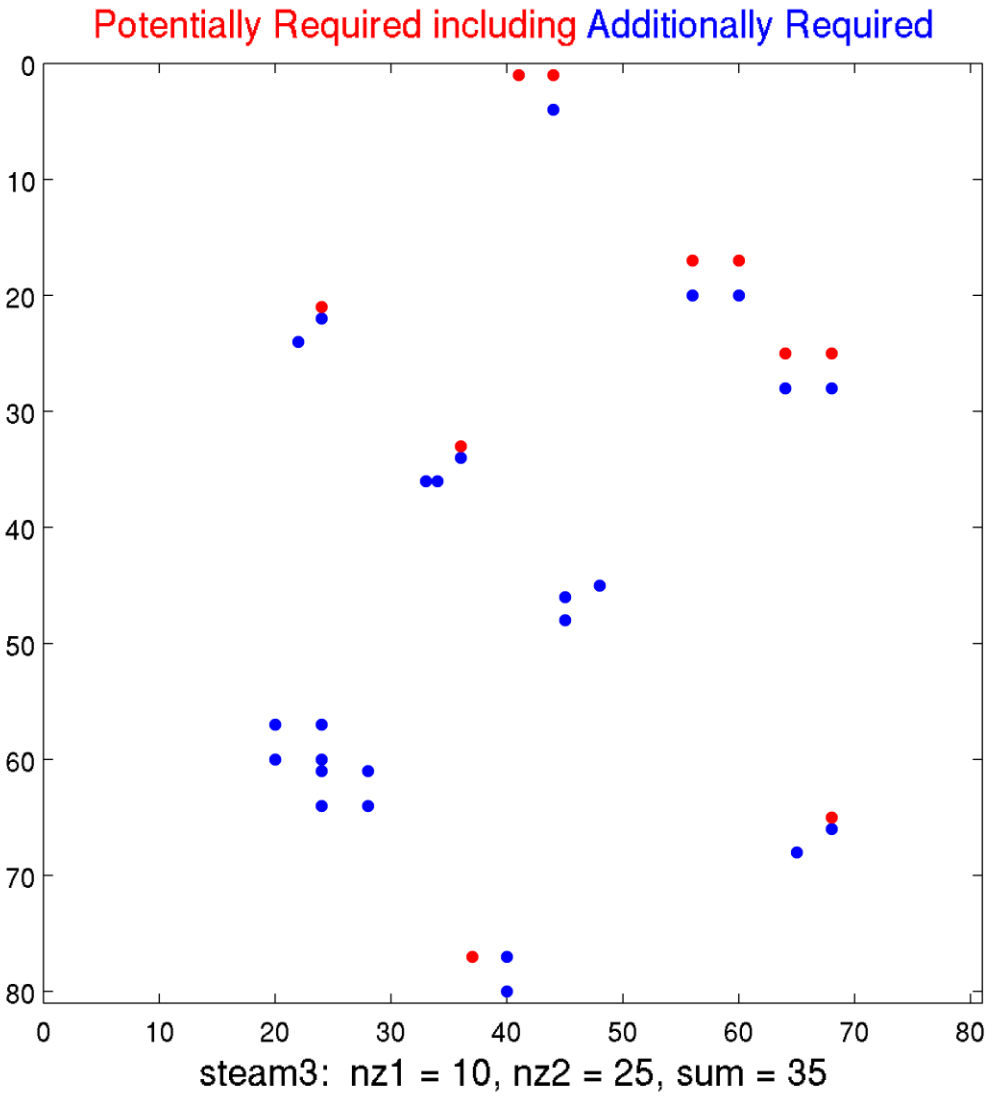
steam3



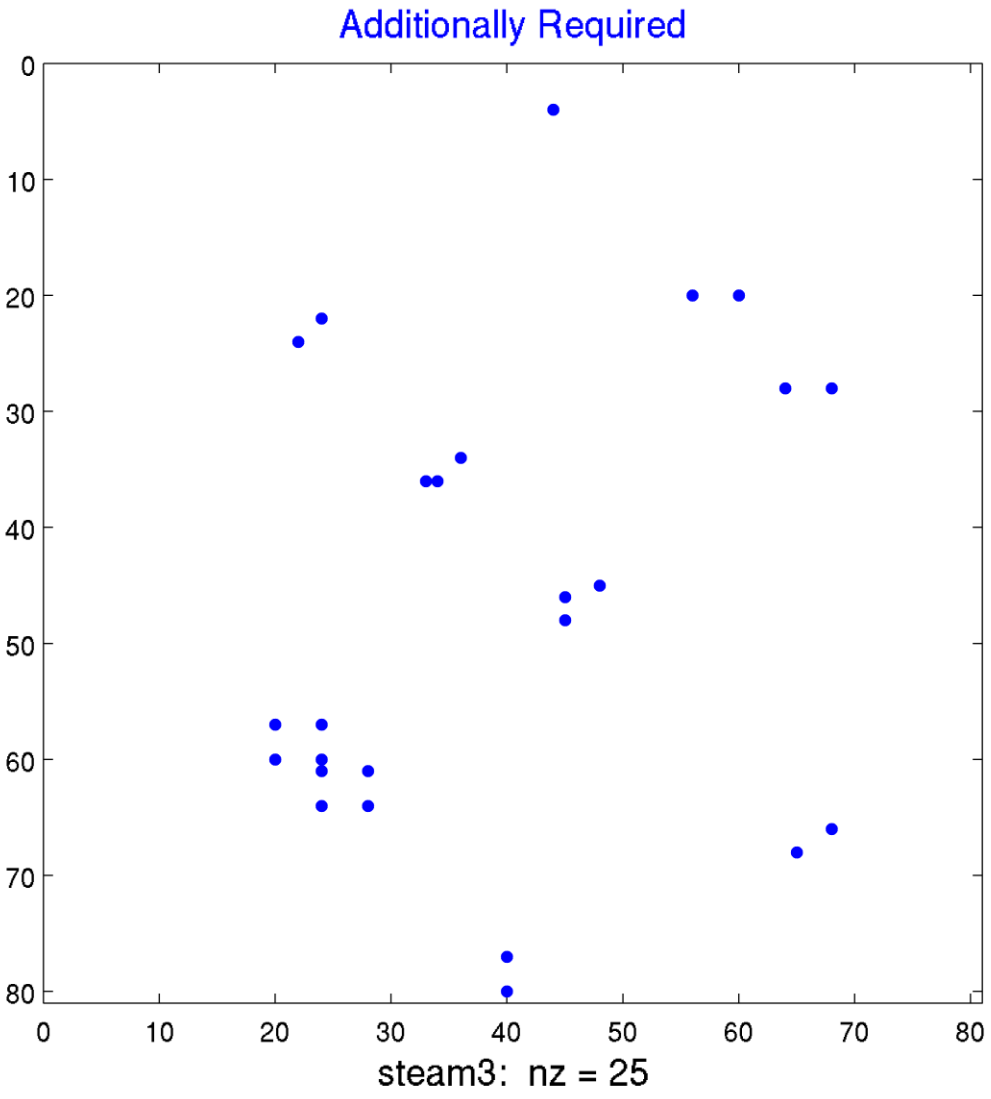
R_{pot}



R_{pot}


Potentially Required including Additionally Required
0
10
20
30
40
50
60
70
80
0
10
20
30
40
50
60
70
80
steam3: nz1 = 10, nz2 = 25, sum = 35

R_{add}

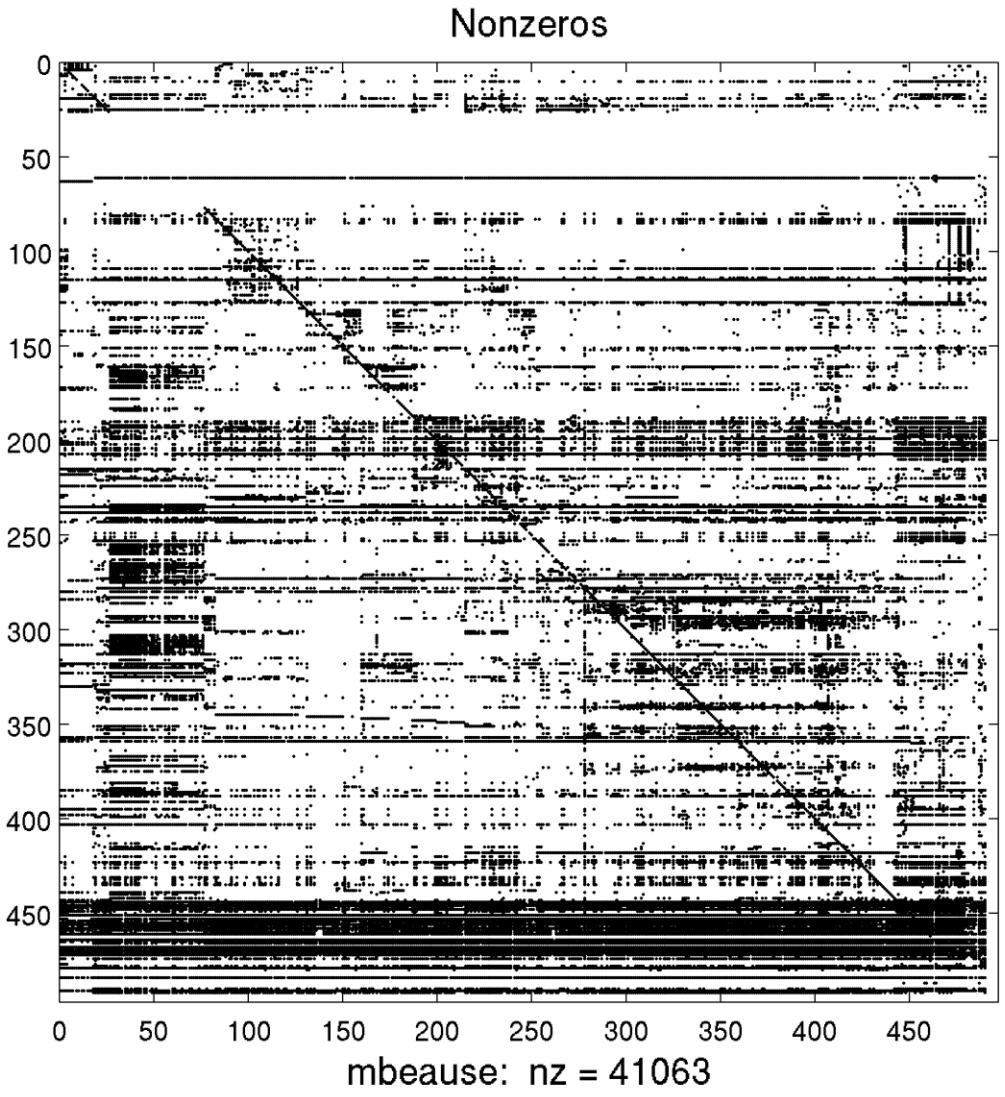

Additionally Required
0
10
20
30
40
50
60
70
80
0
10
20
30
40
50
60
70
80
steam3: nz = 25

R_{add}

mbeause

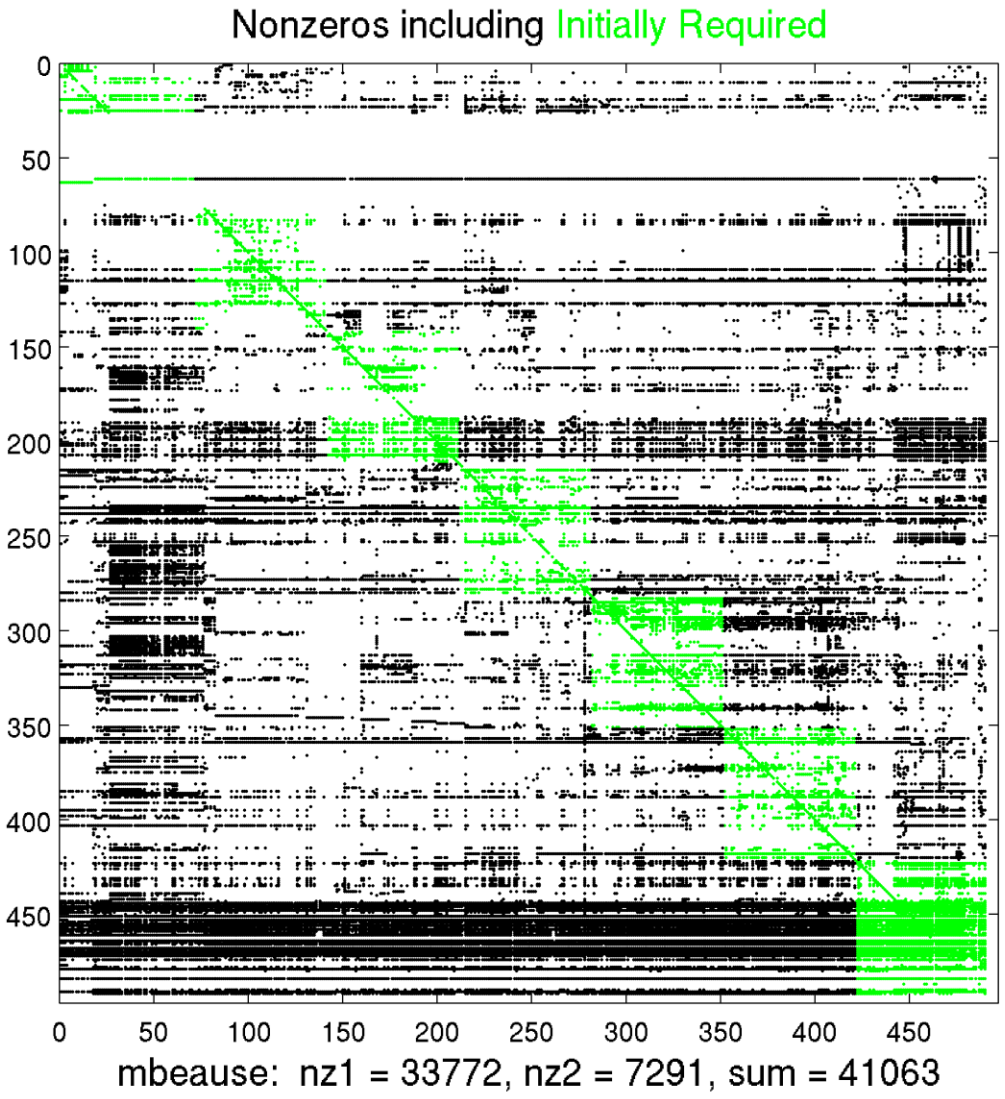
- Visualization of idea
- Larger example with Fill-in

mbeause


Nonzeros
0
50
100
150
200
250
300
350
400
450
0
50
100
150
200
250
300
350
400
450
mbeause: nz = 41063

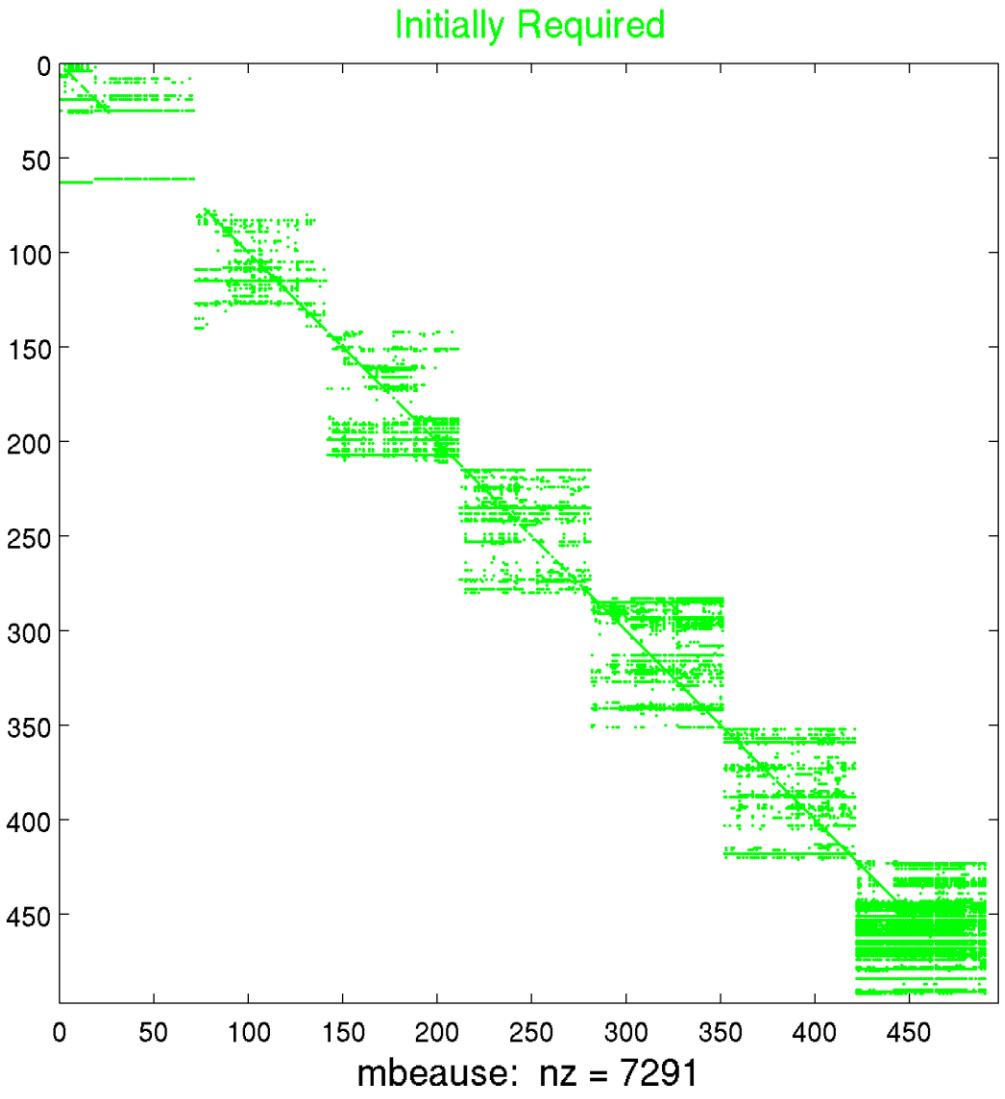
A

mbeause



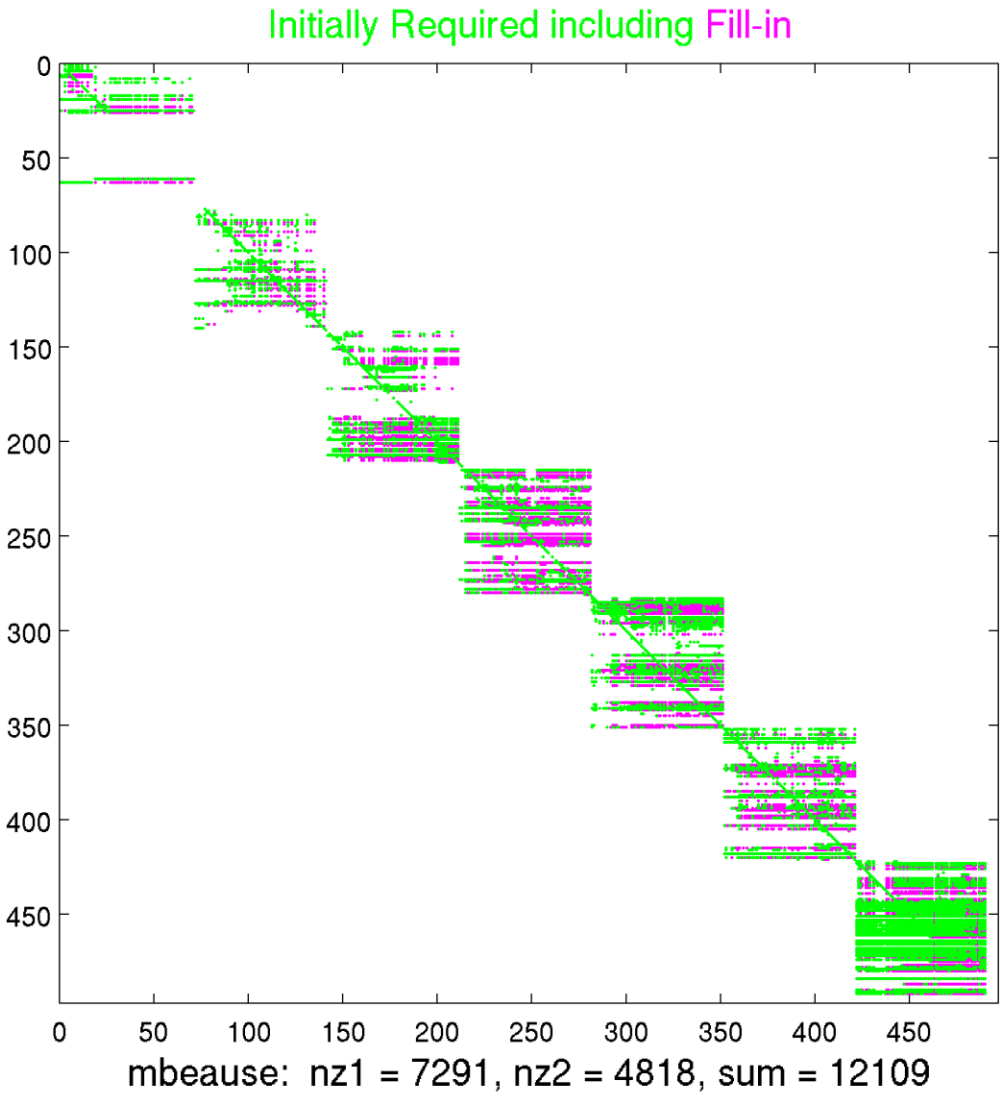
R_{init}

mbeause



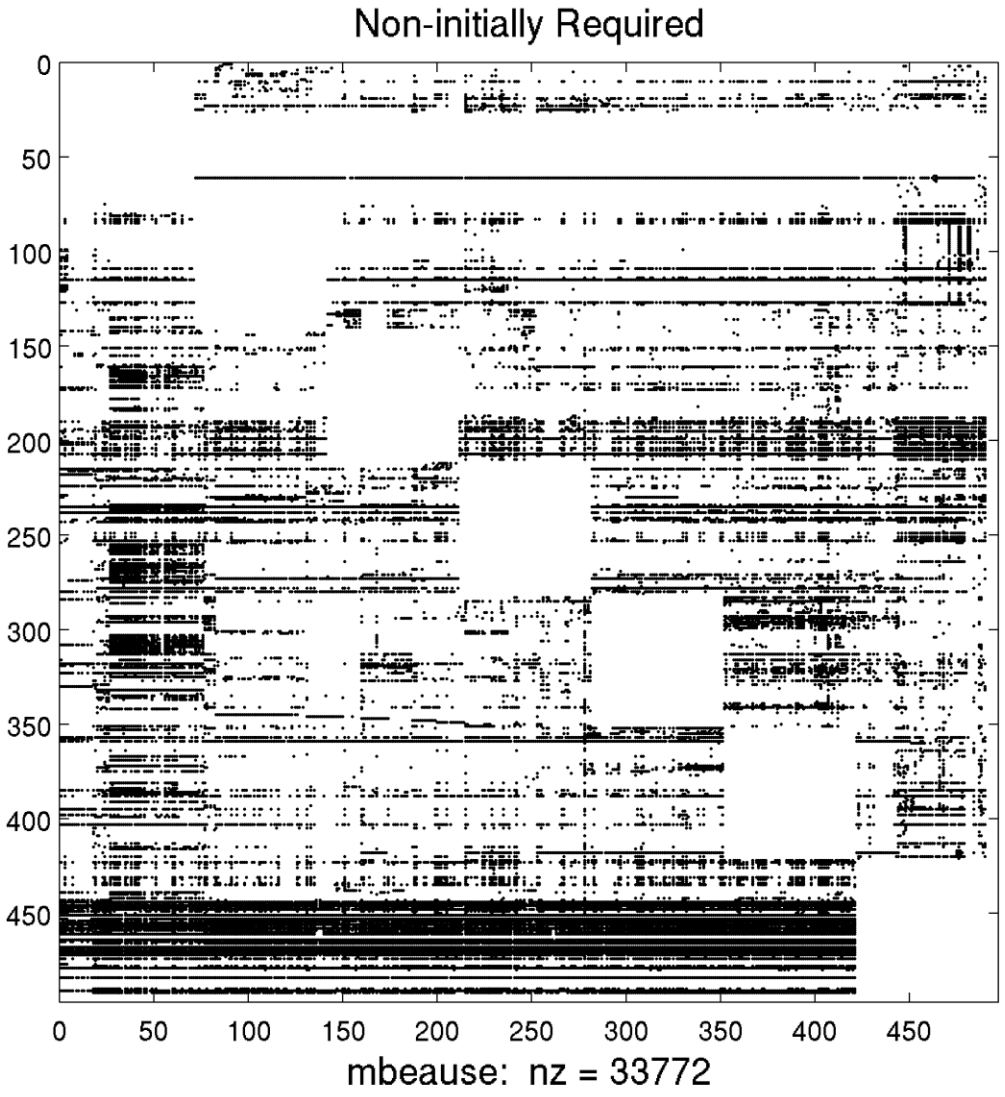
R_{init}

mbeause


Initially Required including Fill-in
0
50
100
150
200
250
300
350
400
450
0
50
100
150
200
250
300
350
400
450
mbeause: nz1 = 7291, nz2 = 4818, sum = 12109

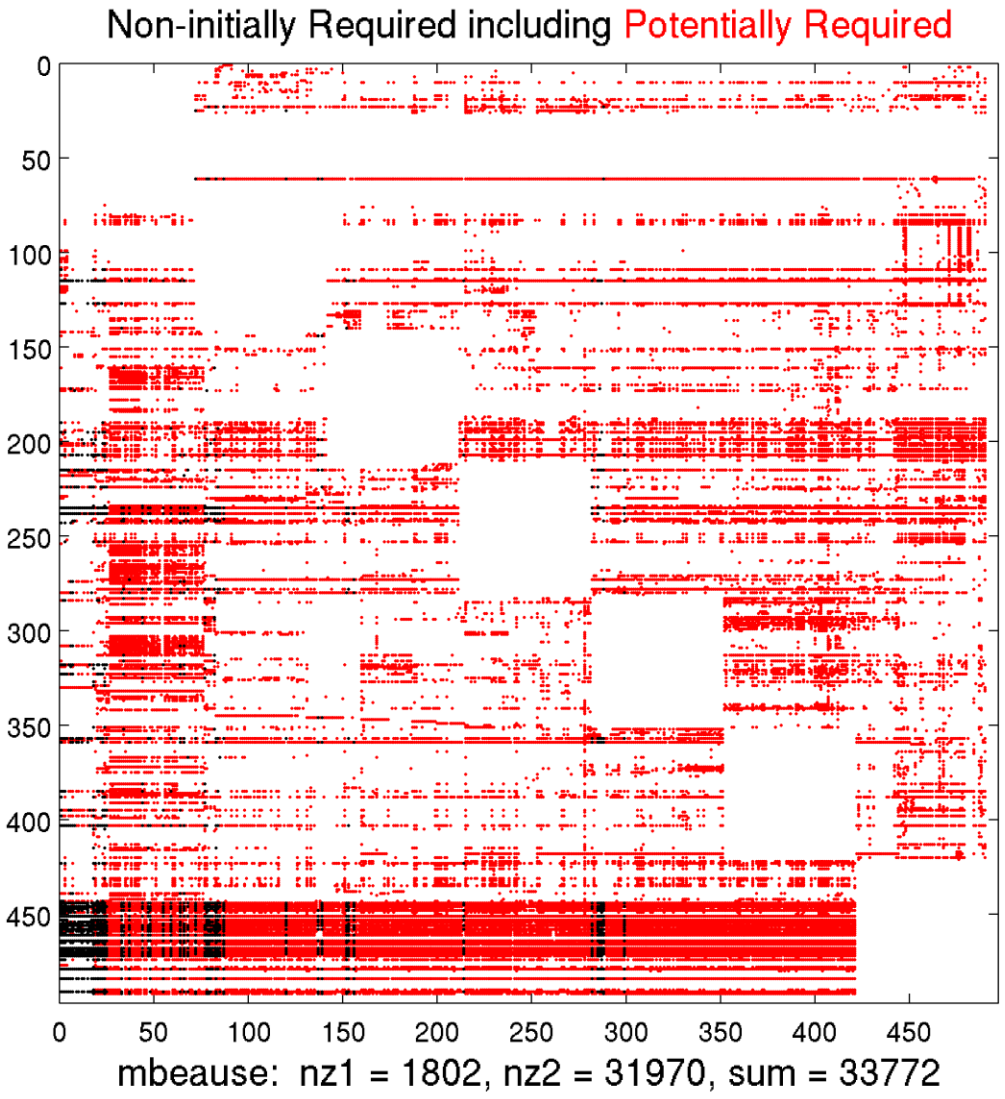
F

mbeause



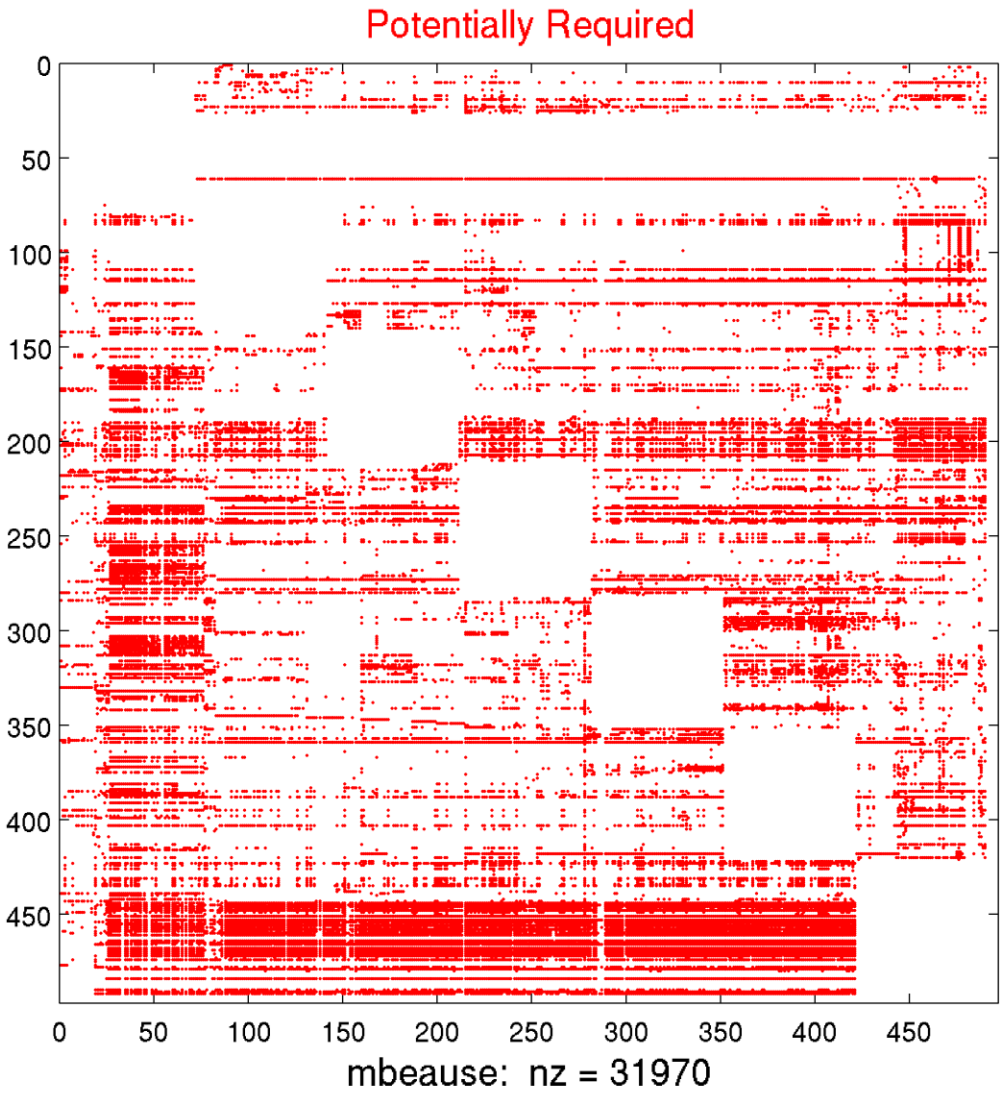
$A \setminus R_{\text{init}}$

mbeause



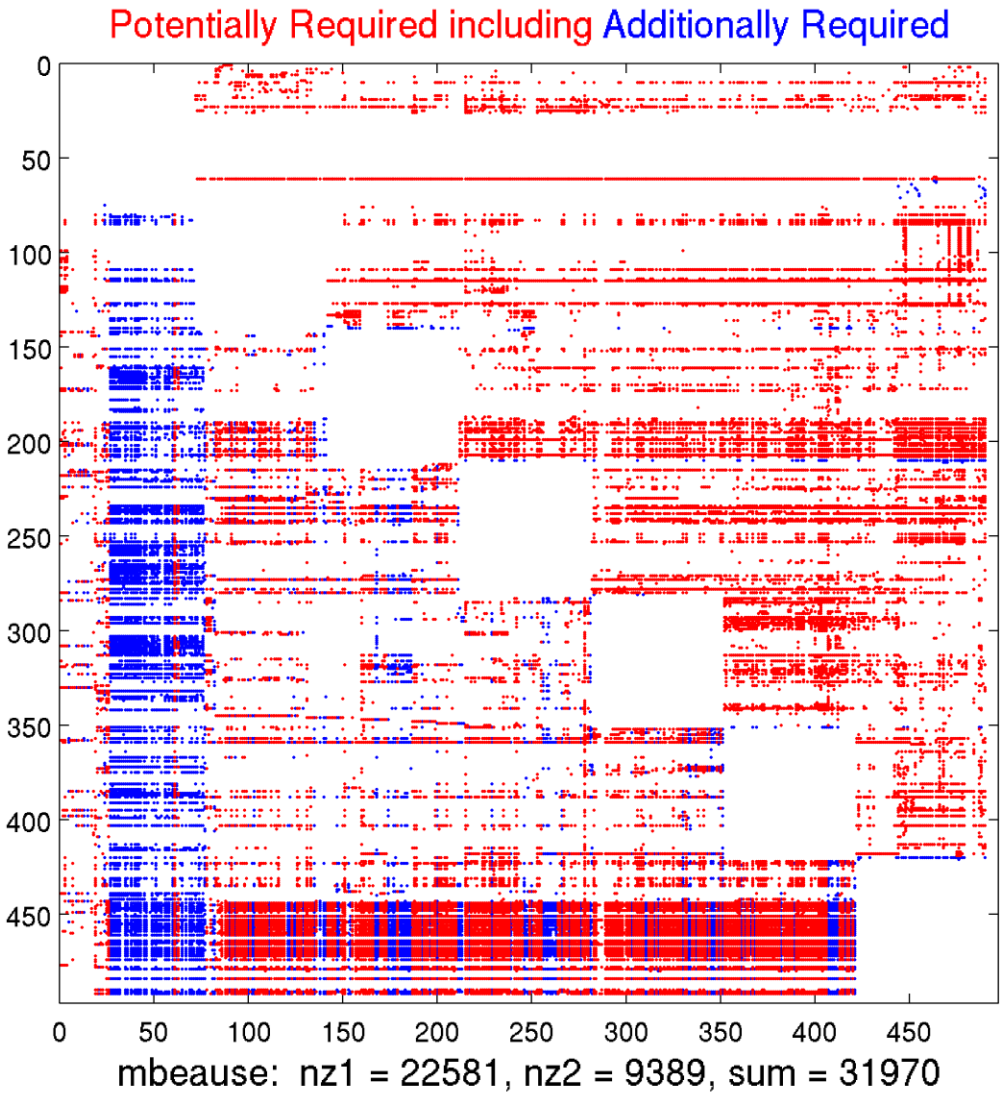
R_{pot}

mbeause



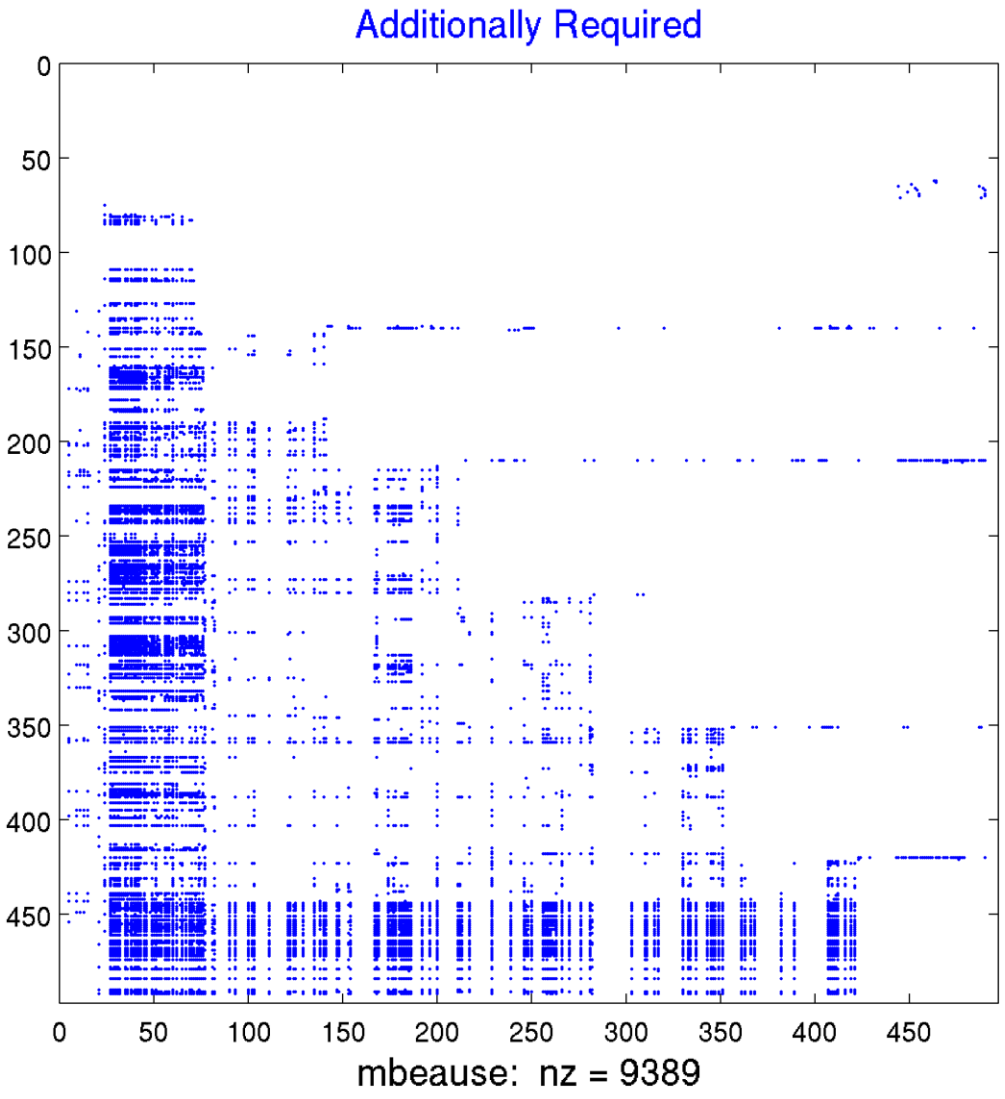
R_{pot}

mbeause



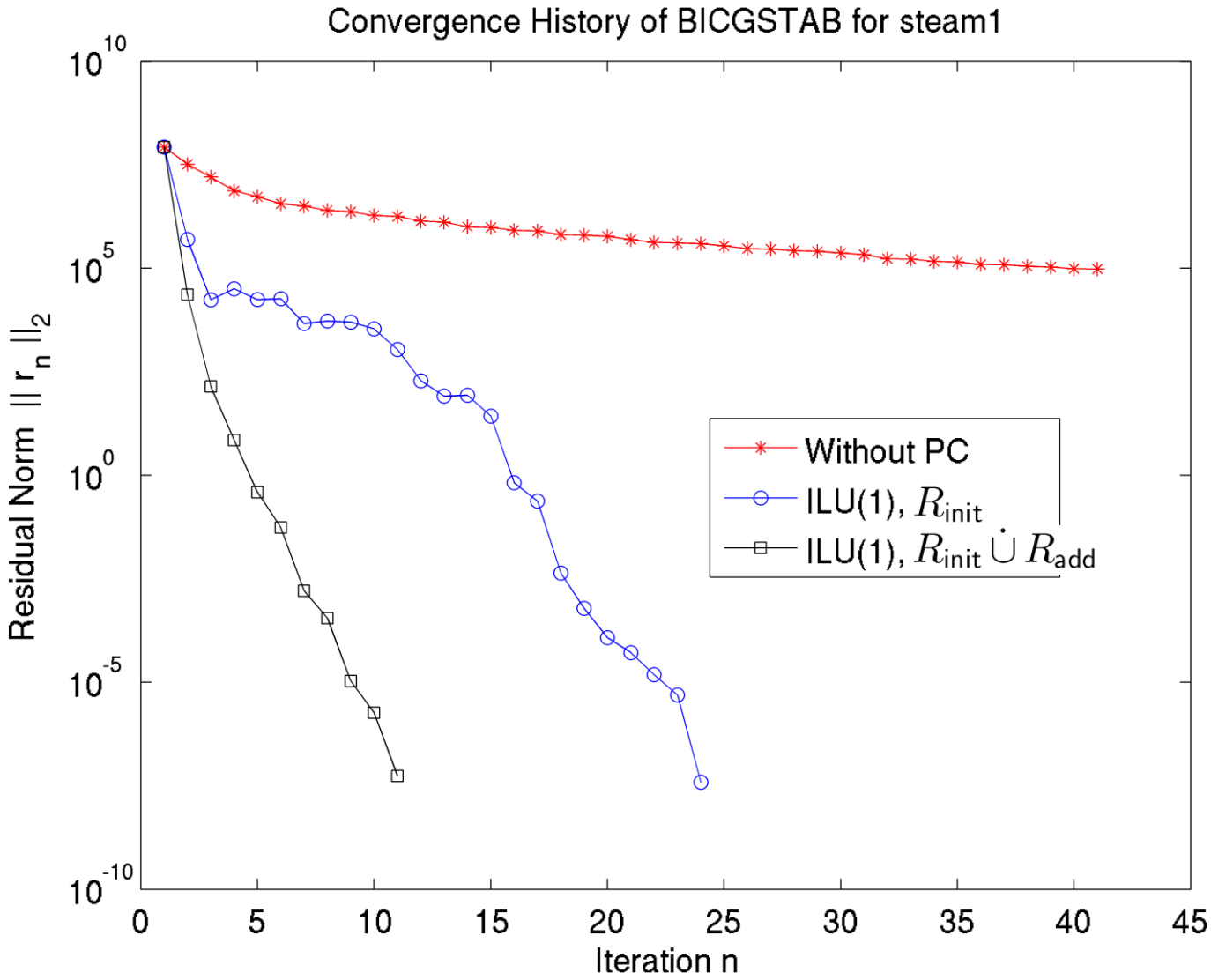
R_{add}

mbeause



R_{add}

steam1



Full:

$k = 25$

Partial:

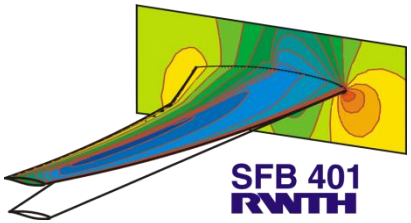
$k = 6$

Summary

- Graphs are suitable in scientific computing
- Preconditioning in connection with automatic differentiation is a new field
- Do not consider fields separately
- Use synergetic effects from different fields
- Interest in collaboration (ADiMat, …)?

Thank you for your attention!

- SFB 401: Strömungsbeeinflussung und Strömungs-Struktur-Wechsel-wirkung an Tragflügeln



- SFB 540: Modellgestützte experimentelle Analyse kinetischer Phänomene in mehrphasigen fluiden Reaktionssystemen

- Graduiertenkolleg 775: Hierarchie und Symmetrie in mathematischen Modellen

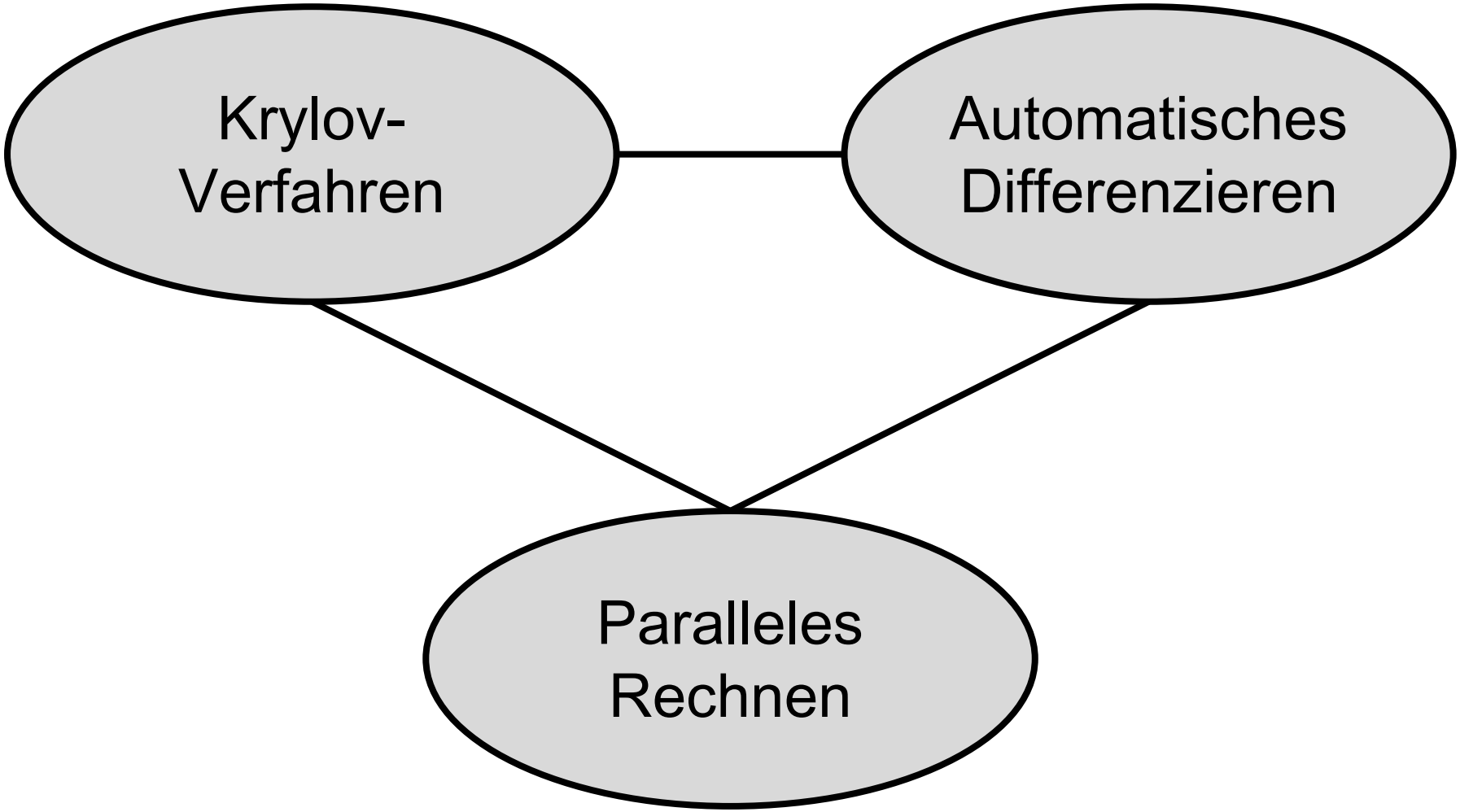
- SPP 1253: Optimierung mit partiellen Differentialgleichungen
- Exzellenz: Graduiertenschule AICES

- BMBF: Verbundprojekt MeProRisk

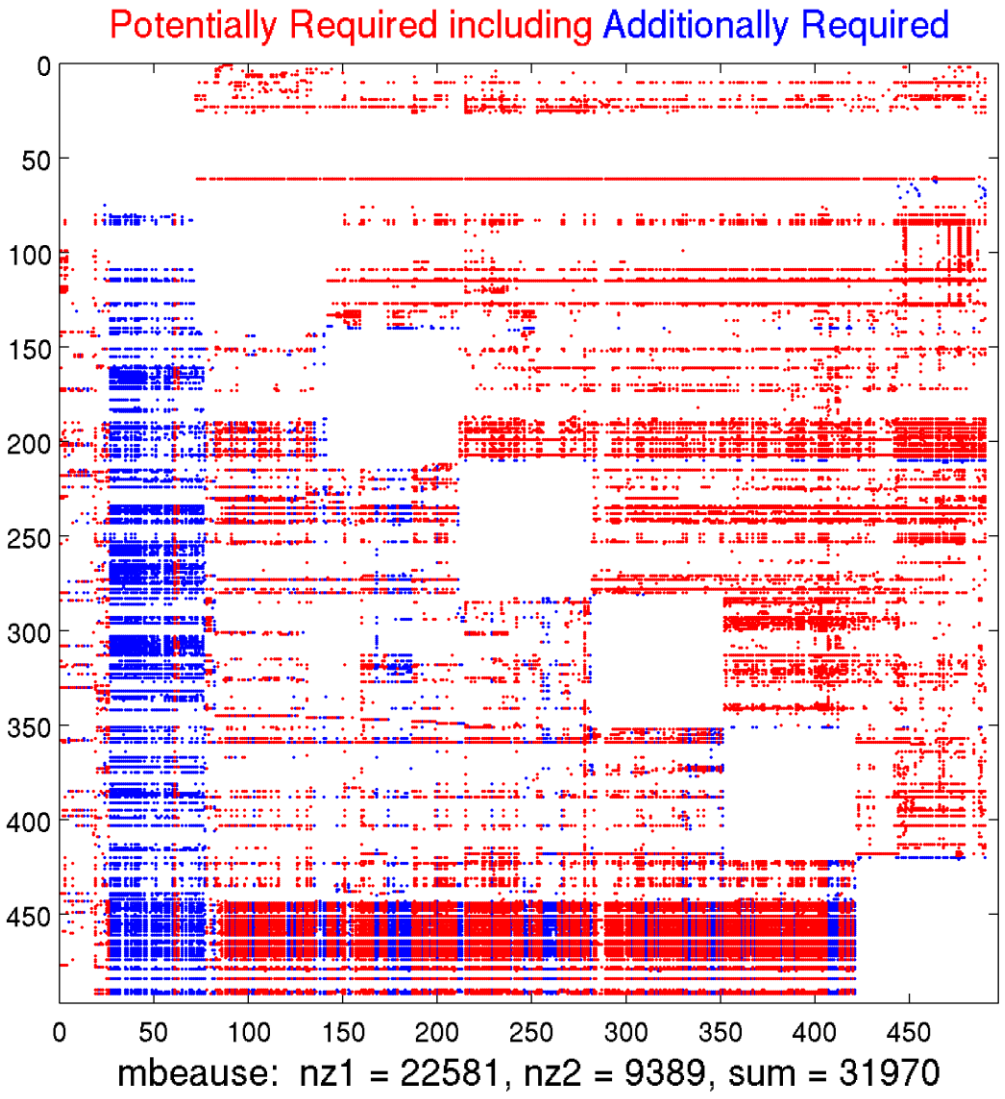
Future Work

- Choice of $R_{\text{add}} \subseteq R_{\text{pot}}$
  - More aggressive strategies
  - Parallelism i $M\mathbf{y} = \mathbf{z}$
- Einfluss der Parameter Level und Blockgröße
- Vorkonditionierung so, dass ganz R_{pot} verwendet wird
- Färbung und R_{add} kombiniert betrachten
- Algorithmen zur Färbung
- Mehr Erfahrungen mit realen Anwendungen

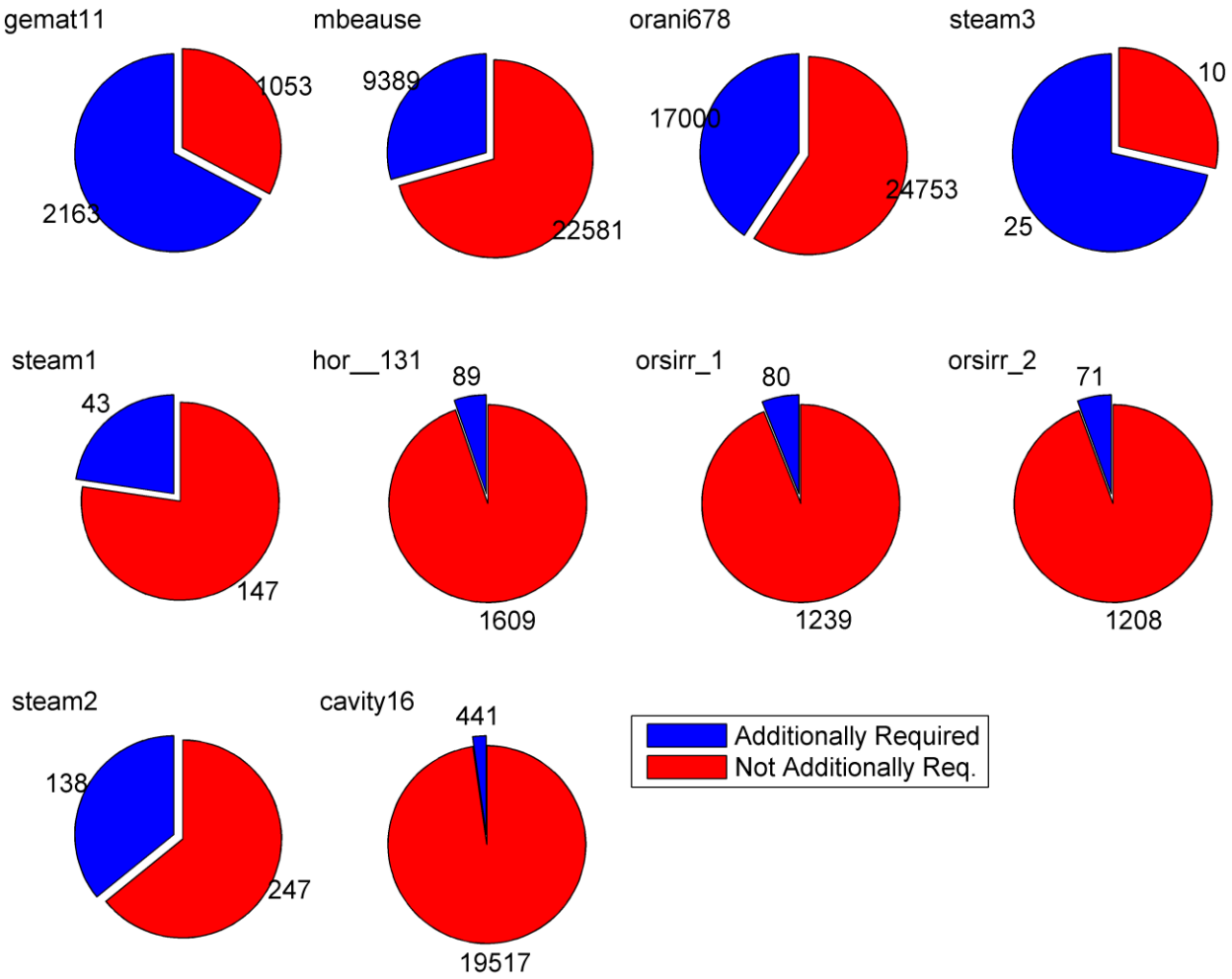
Teilgebiete



mbeause


Potentially Required including Additionally Required
0
50
100
150
200
250
300
350
400
450
0
50
100
150
200
250
300
350
400
450
mbeause: nz1 = 22581, nz2 = 9389, sum = 31970

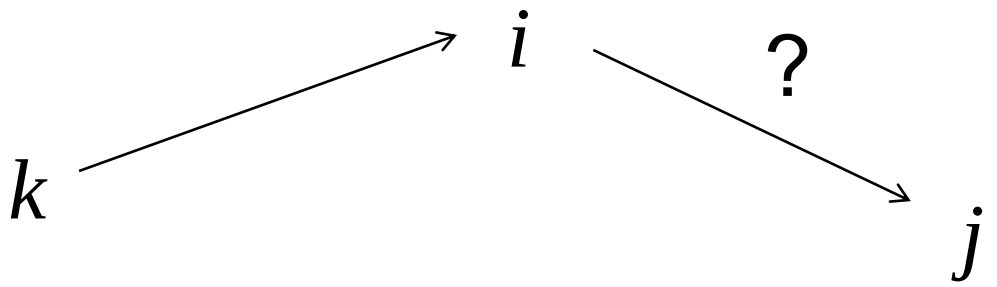
Aufteilung von R_{pot}



Schnelle Strategie (i)

zur Auswahl von $R_{\text{add}} \subseteq R_{\text{pot}}$

Betrachte Fill-in an Position (i,j) :

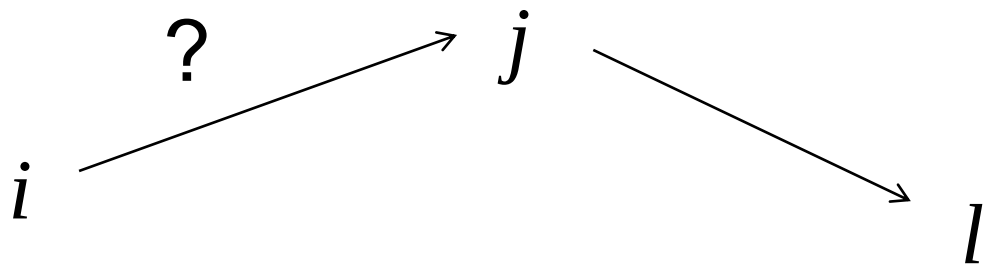


Fall 1: $i < j$
Falls es Vorgänger k von i gibt mit $k > i$,
dann Fill-in, also nicht einschließen in R_{add}

Schnelle Strategie (ii)

zur Auswahl von $R_{\text{add}} \subseteq R_{\text{pot}}$

Betrachte Fill-in an Position (i,j) :



Fall 2: $i > j$:
Falls es Nachfolger l von j gibt mit $j > l$,
dann Fill-in, also nicht einschließen in R_{add}

Sketch of proof

- Let p_{rsb} be the smallest number of colors for a RSB.
- Let p_{rd2} be the smallest number of colors for a RD2.

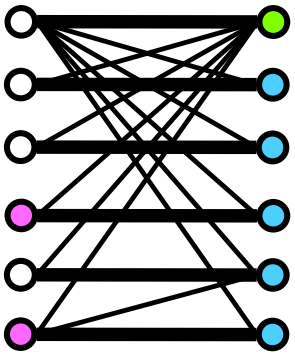
To prove $p_{rsb} = p_{rd2}$ show the following:

- Lemma 1: $p_{rsb} \geq p_{rd2}$
- Lemma 2: $p_{rd2} \geq p_{rsb}$

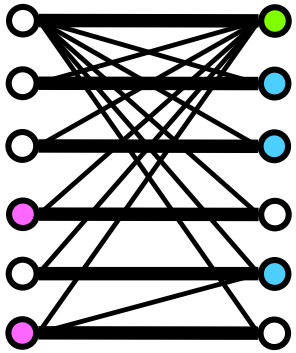
Lemma 1: $p_{rsb} \geq p_{rd2}$

We transform an RSB Φ with the fewest number of colors into an RD2 coloring Φ_c where the number of colors is not larger.

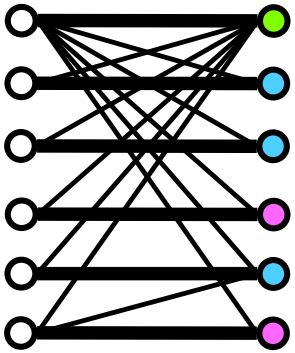
1. The mapping Φ will be transformed into an RSB Φ_* where exactly one of both incident vertices of each edge in E_D is colored with nonzero.
2. The coloring Φ_* is transformed into an RSB Φ', where all vertices $r_i \in V_r$ are colored with $\Phi'(r_i) = 0$.
3. The mapping Φ' is also an RD2 coloring.



Φ



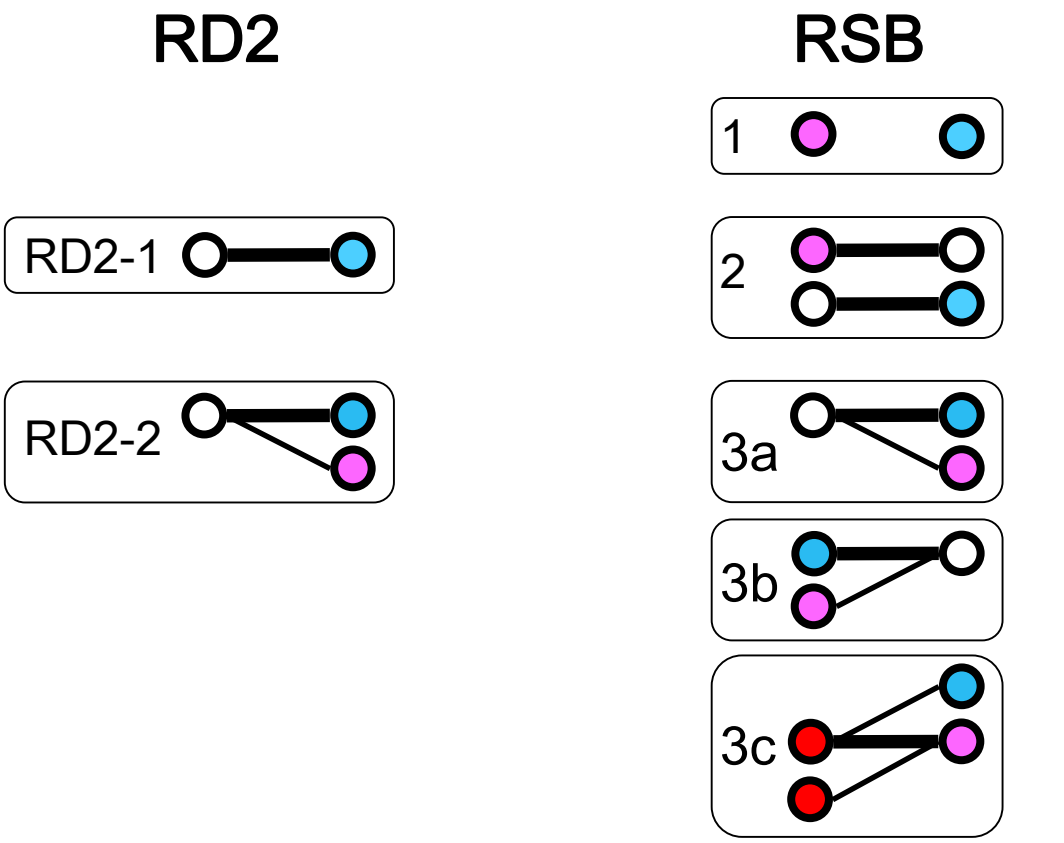
Φ_*

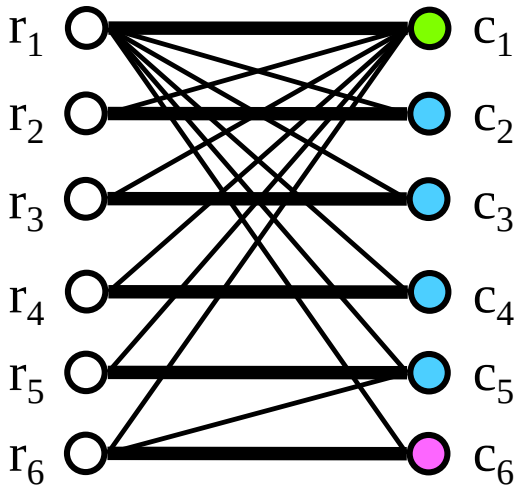


$\Phi' = \Phi_C$

Lemma 2: $p_{rd2} \geq p_{rsb}$

A given RD2 coloring with the fewest number of colors is also an RSB.

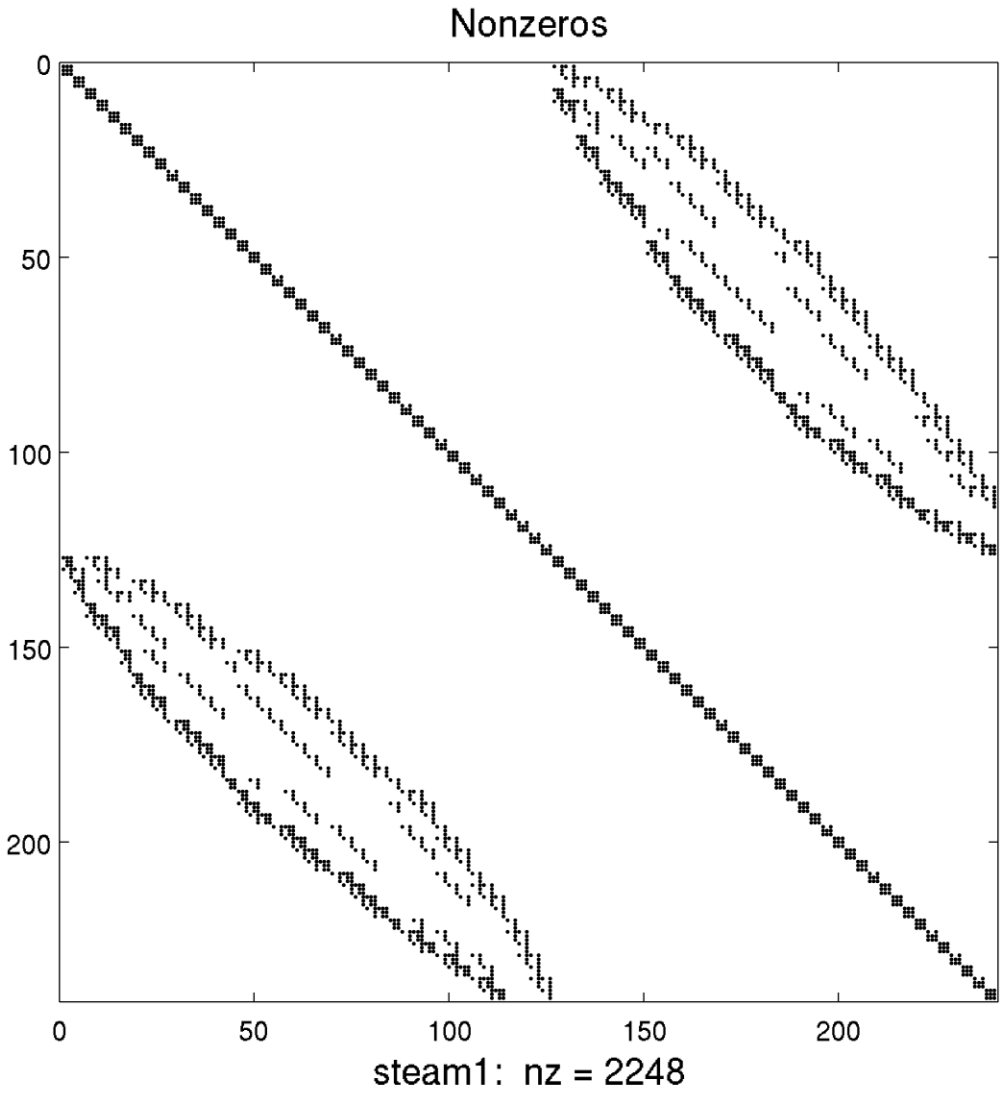




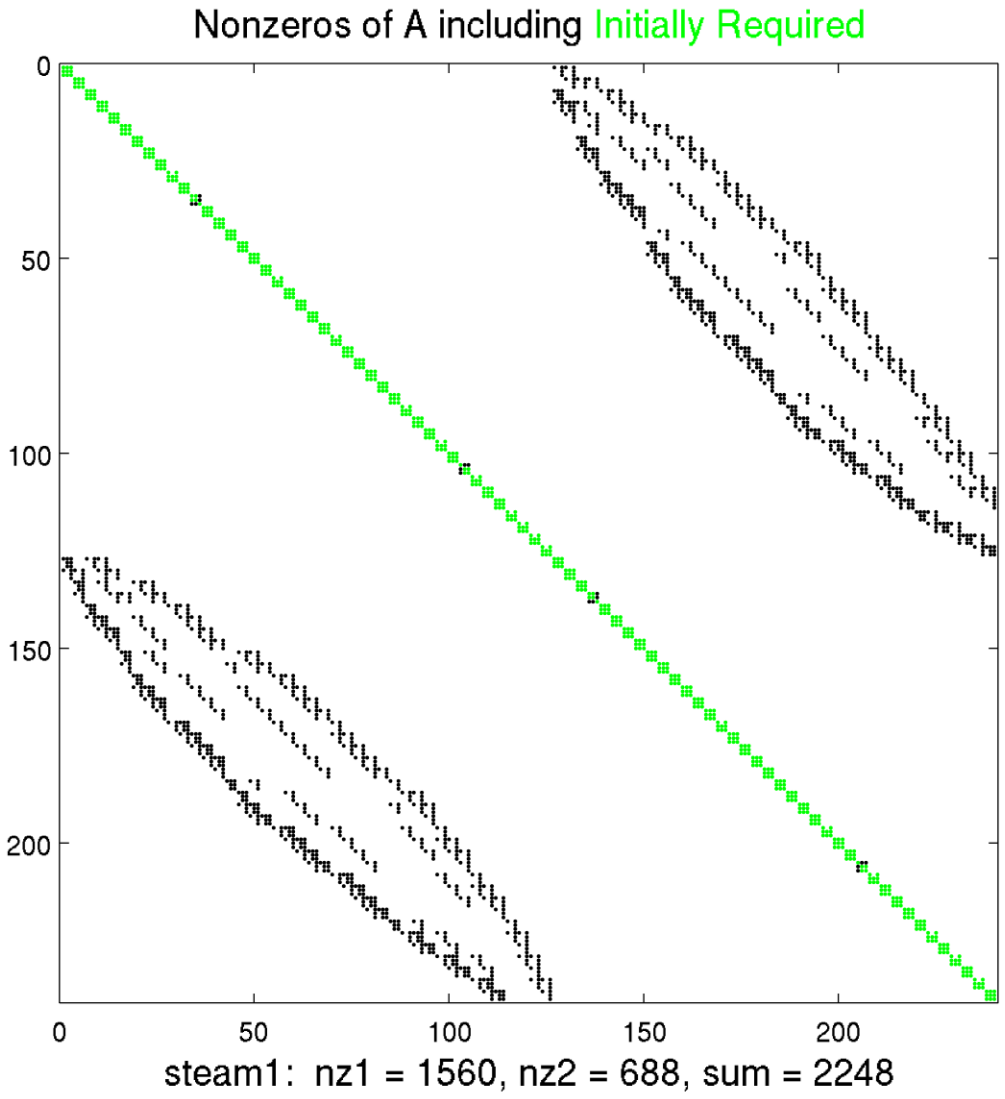
Steam1

- Visualisierung der Idee (Phänomene schlechter erkennbar)
- Allerdings ohne Fill-in

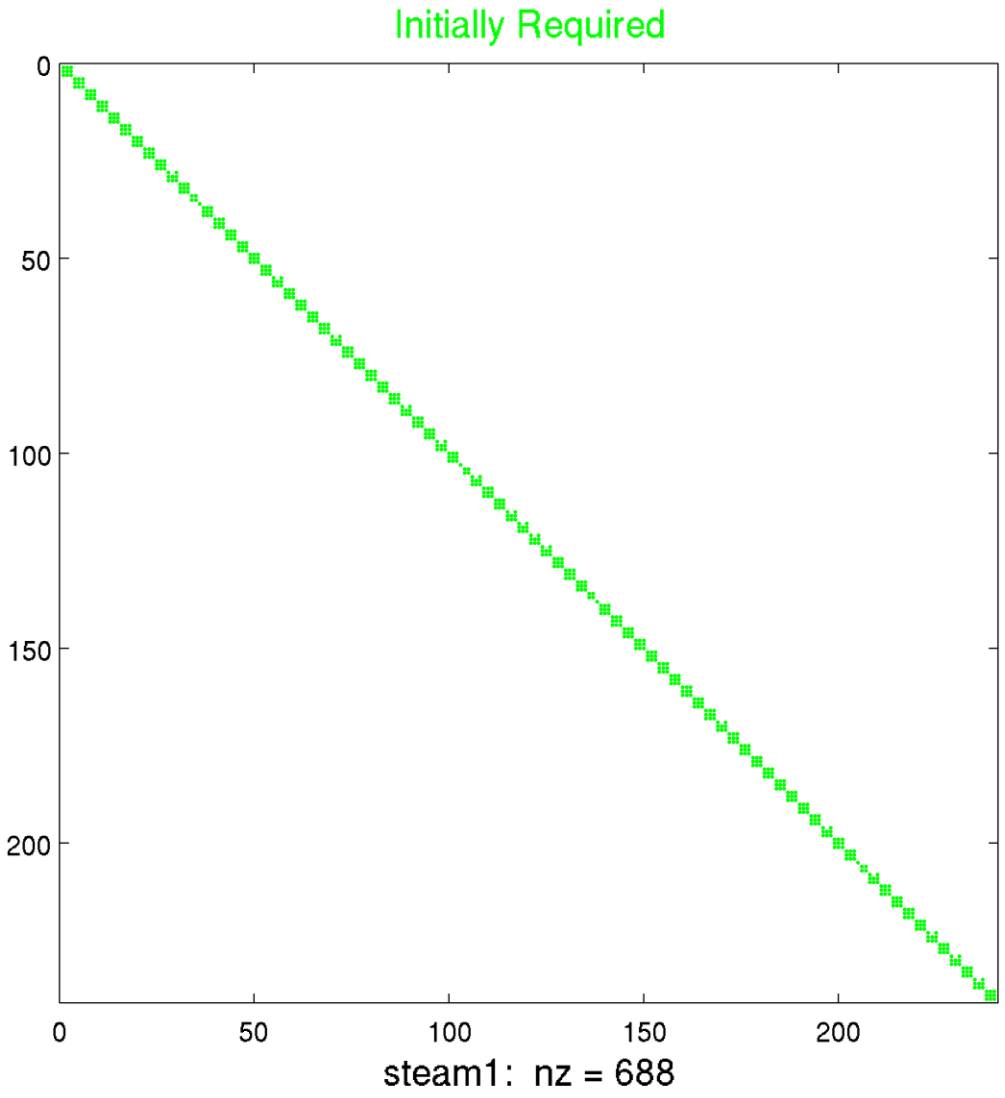
steam1



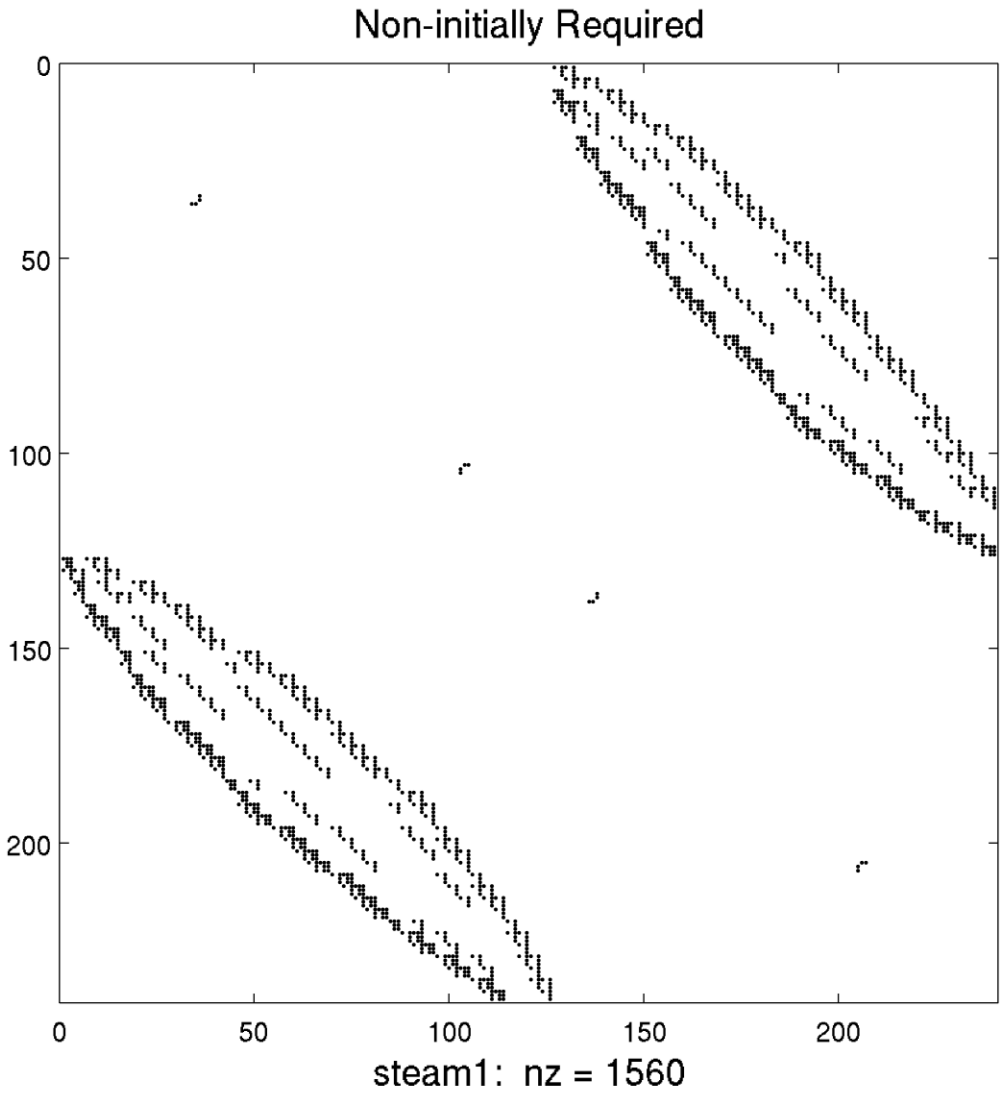
steam1


Nonzeros of A including Initially Required
0
50
100
150
200
0
50
100
150
200
steam1: nz1 = 1560, nz2 = 688, sum = 2248

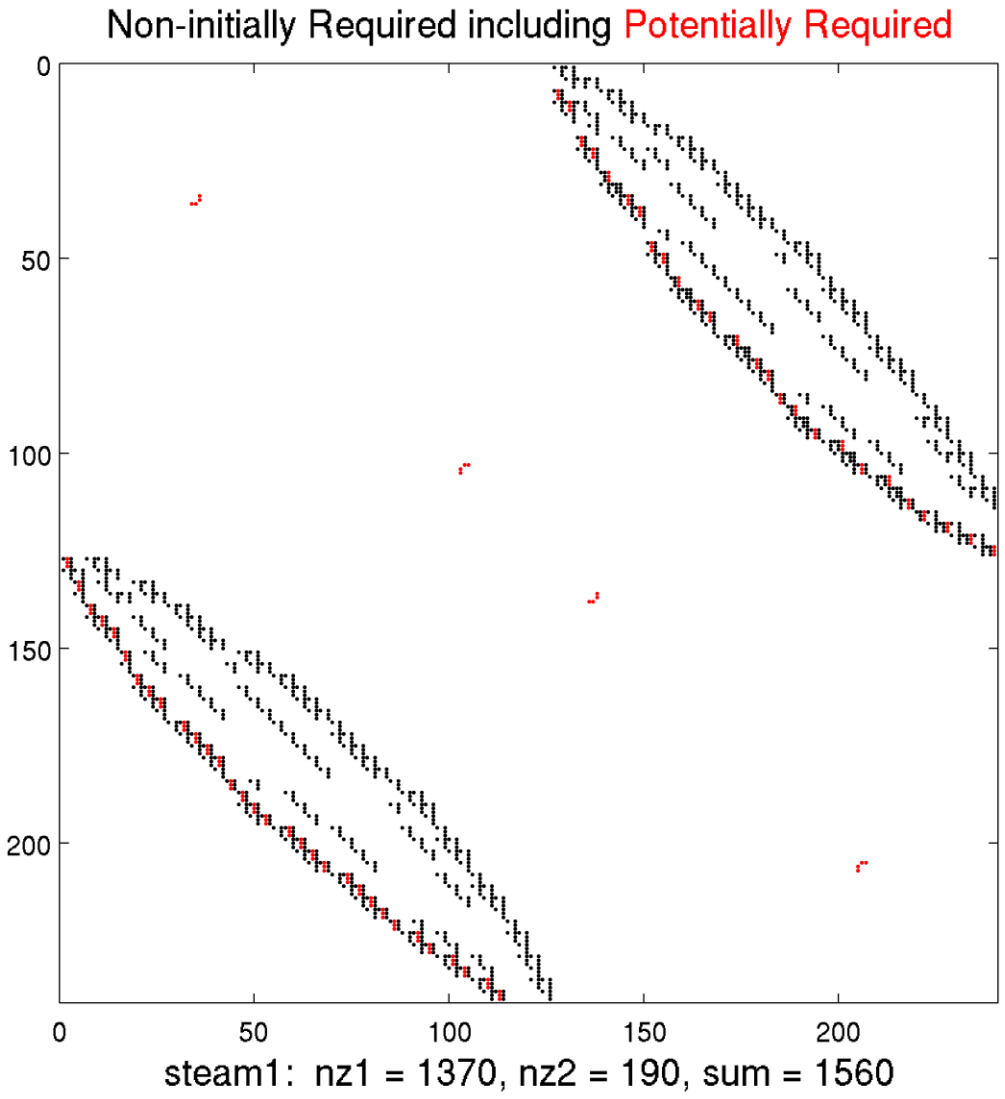
steam1


Initially Required
0
50
100
150
200
0
50
100
150
200
steam1: nz = 688

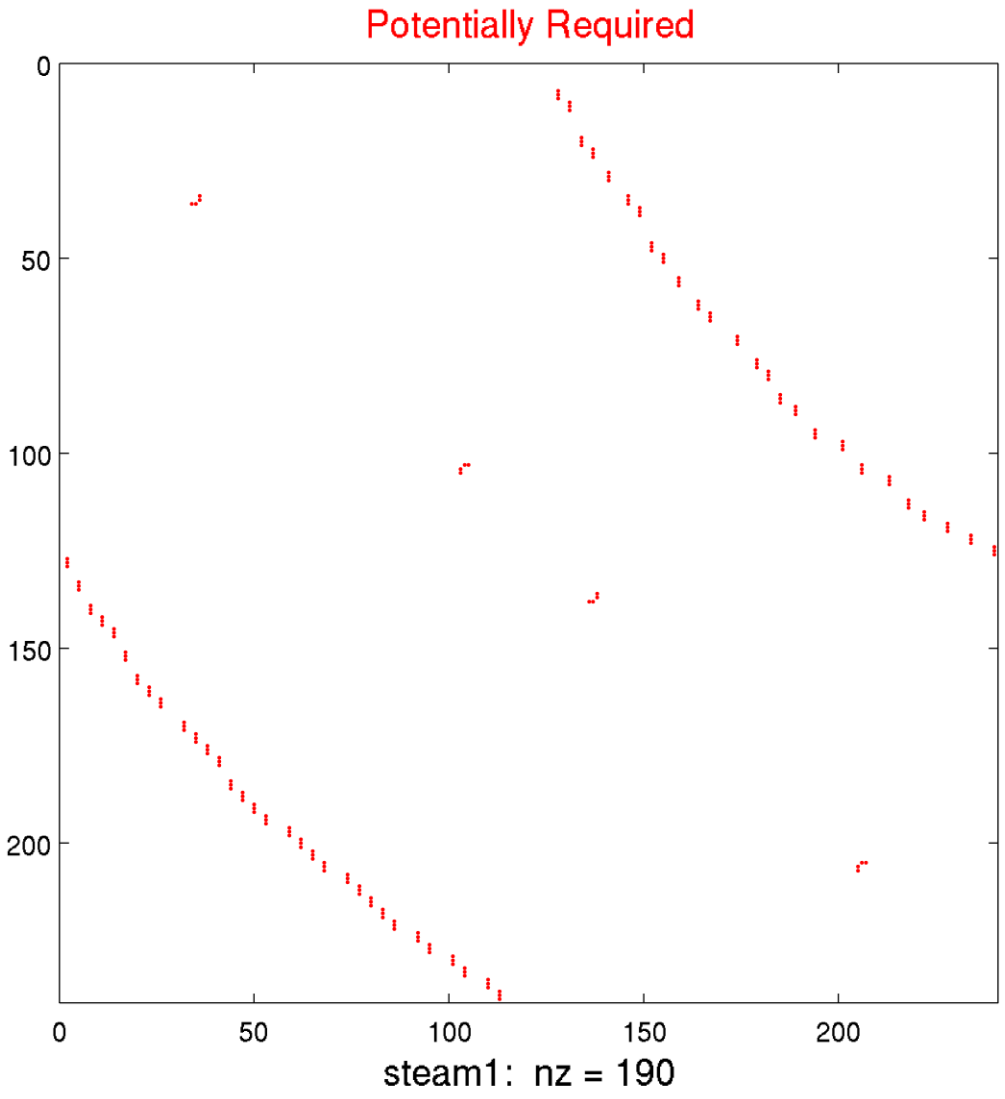
steam1


Non-initially Required
0
50
100
150
200
0
50
100
150
200
steam1: nz = 1560

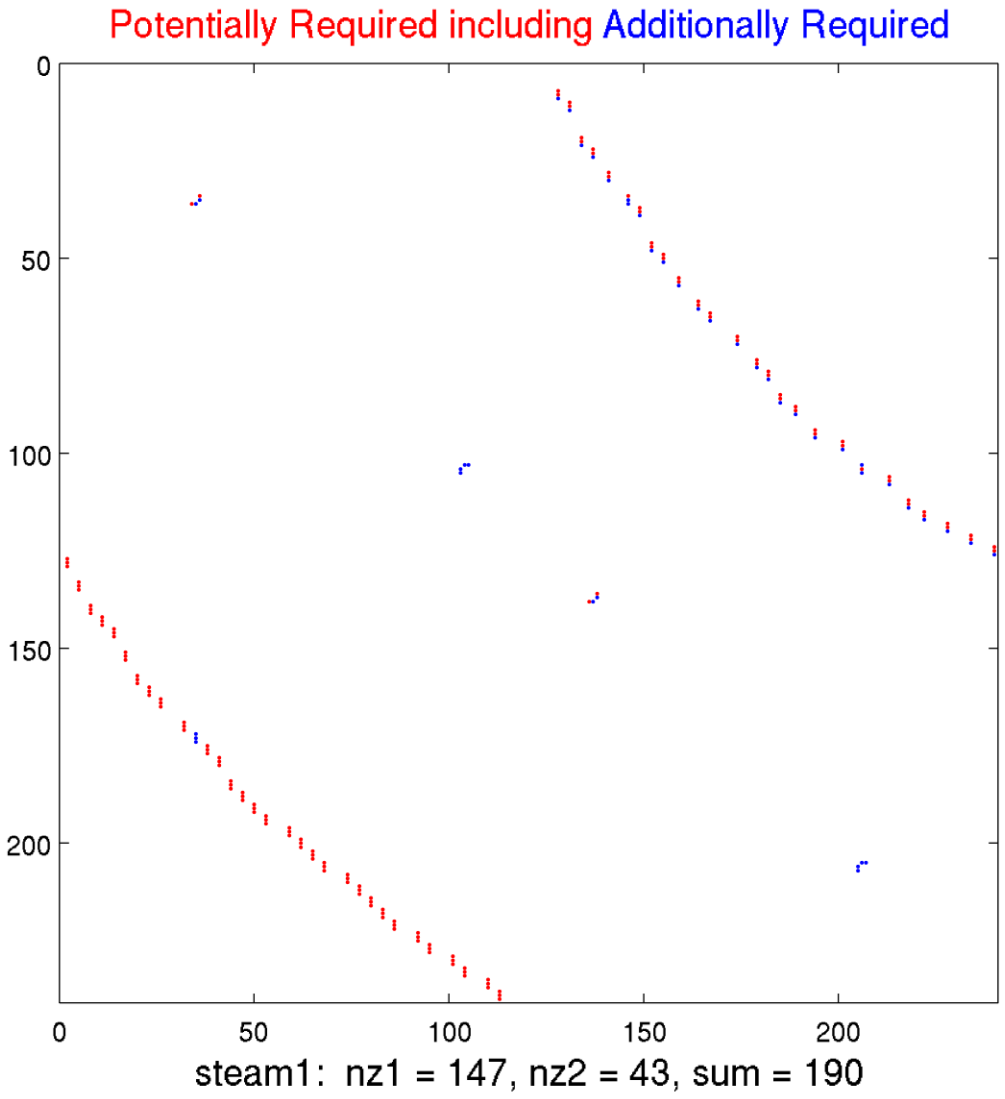
steam1


Non-initially Required including Potentially Required
0
50
100
150
200
0
50
100
150
200
steam1: nz1 = 1370, nz2 = 190, sum = 1560

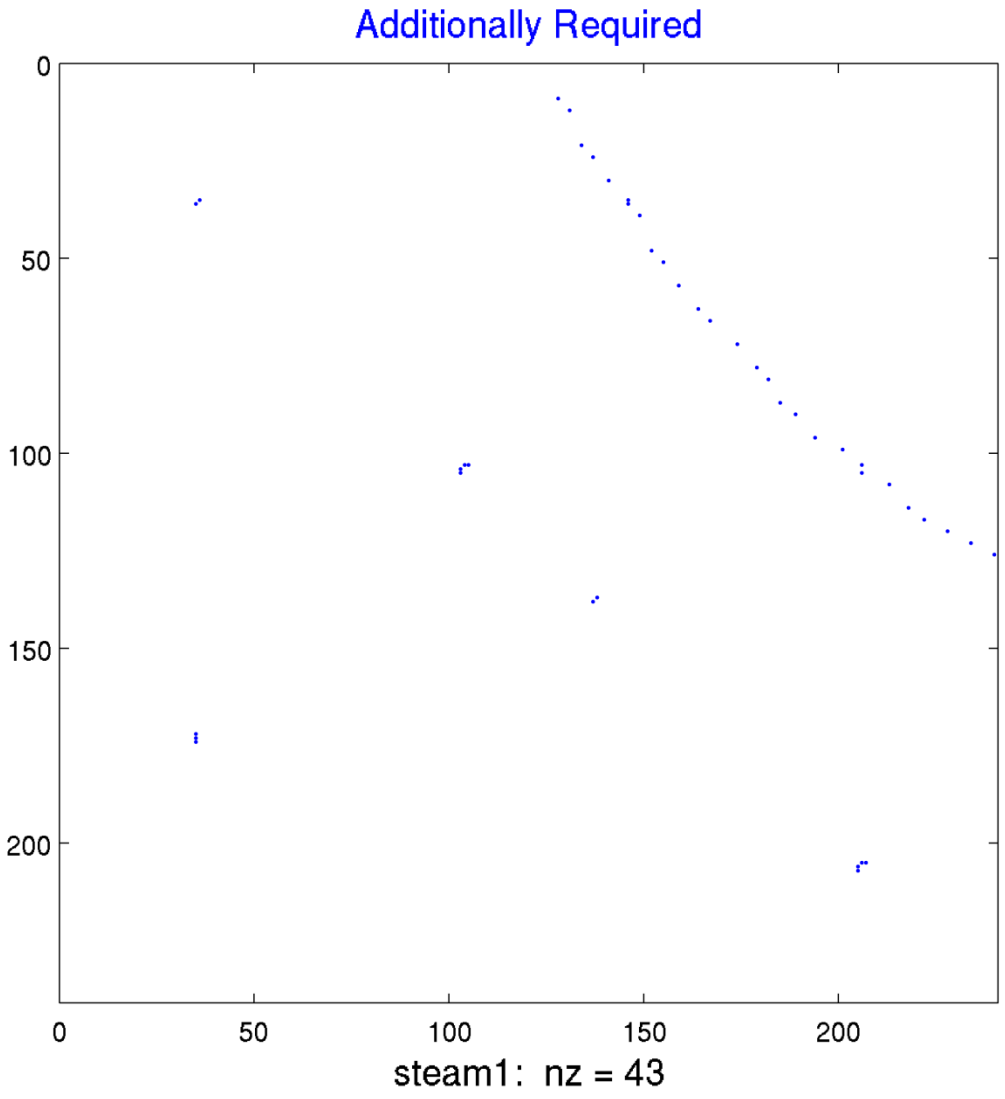
steam1


Potentially Required
0
50
100
150
200
0
50
100
150
200
steam1: nz = 190

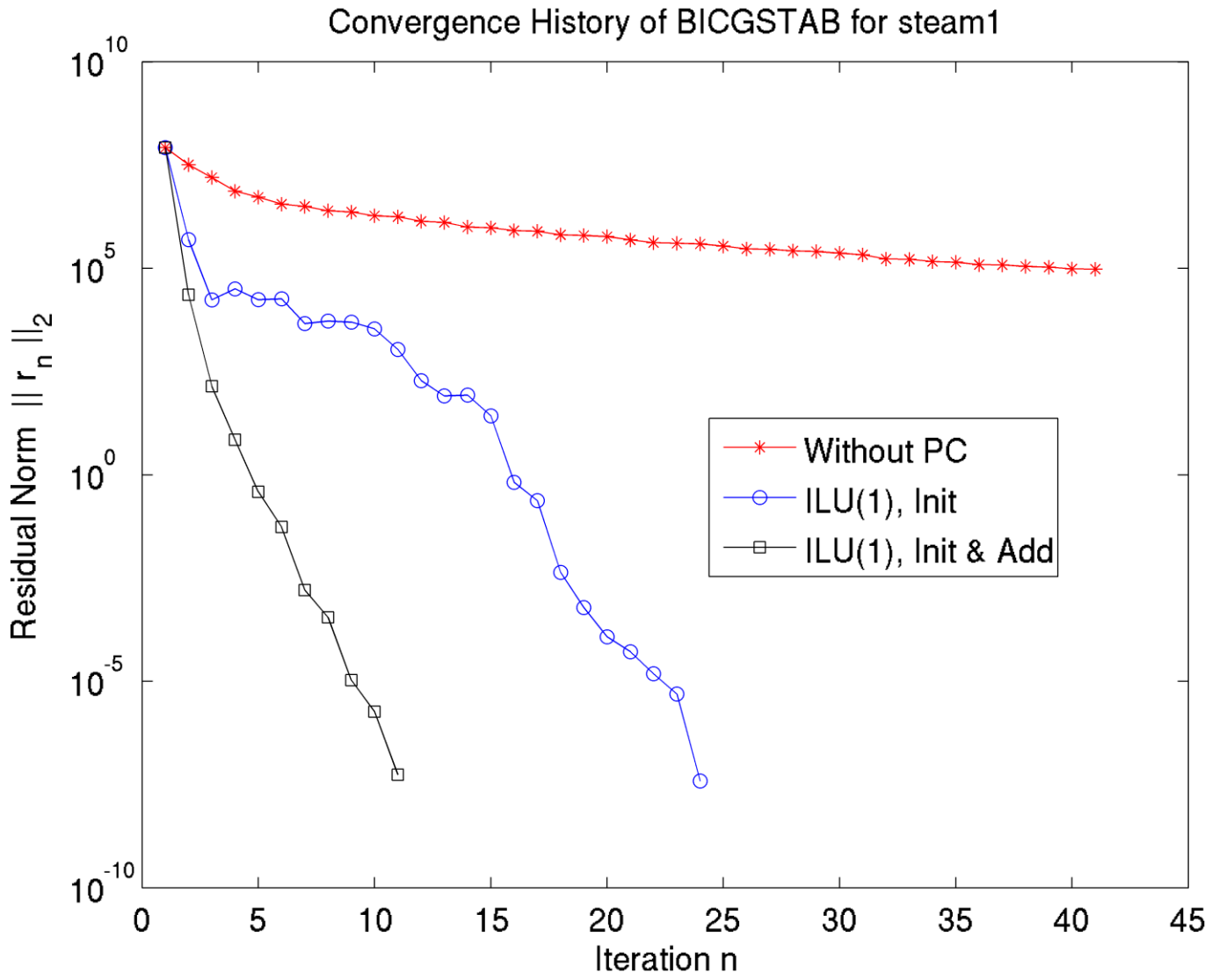
steam1


Potentially Required including Additionally Required
0
50
100
150
200
0
50
100
150
200
steam1: nz1 = 147, nz2 = 43, sum = 190

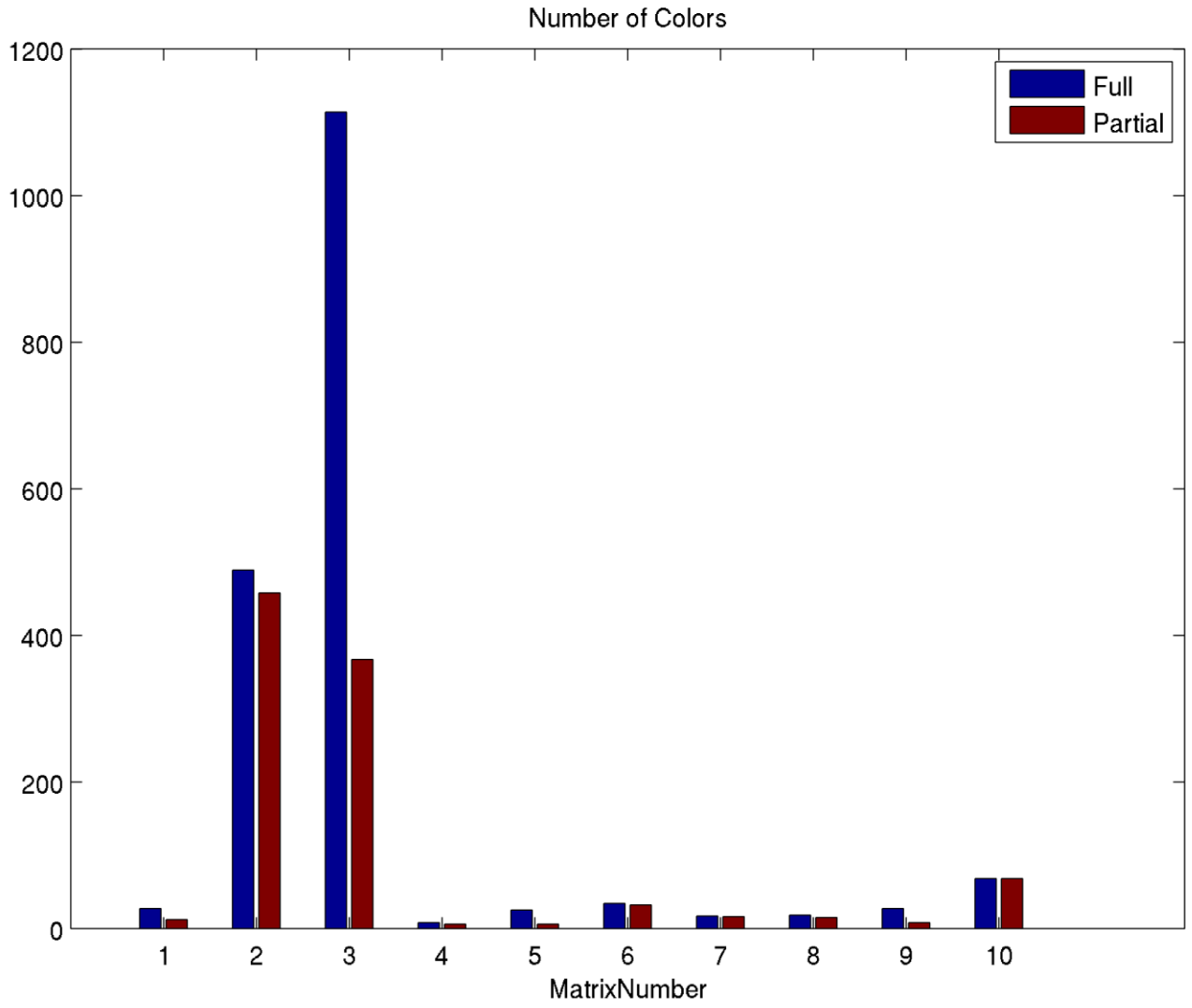
steam1


Additionally Required
0
50
100
150
200
0
50
100
150
200
steam1: nz = 43

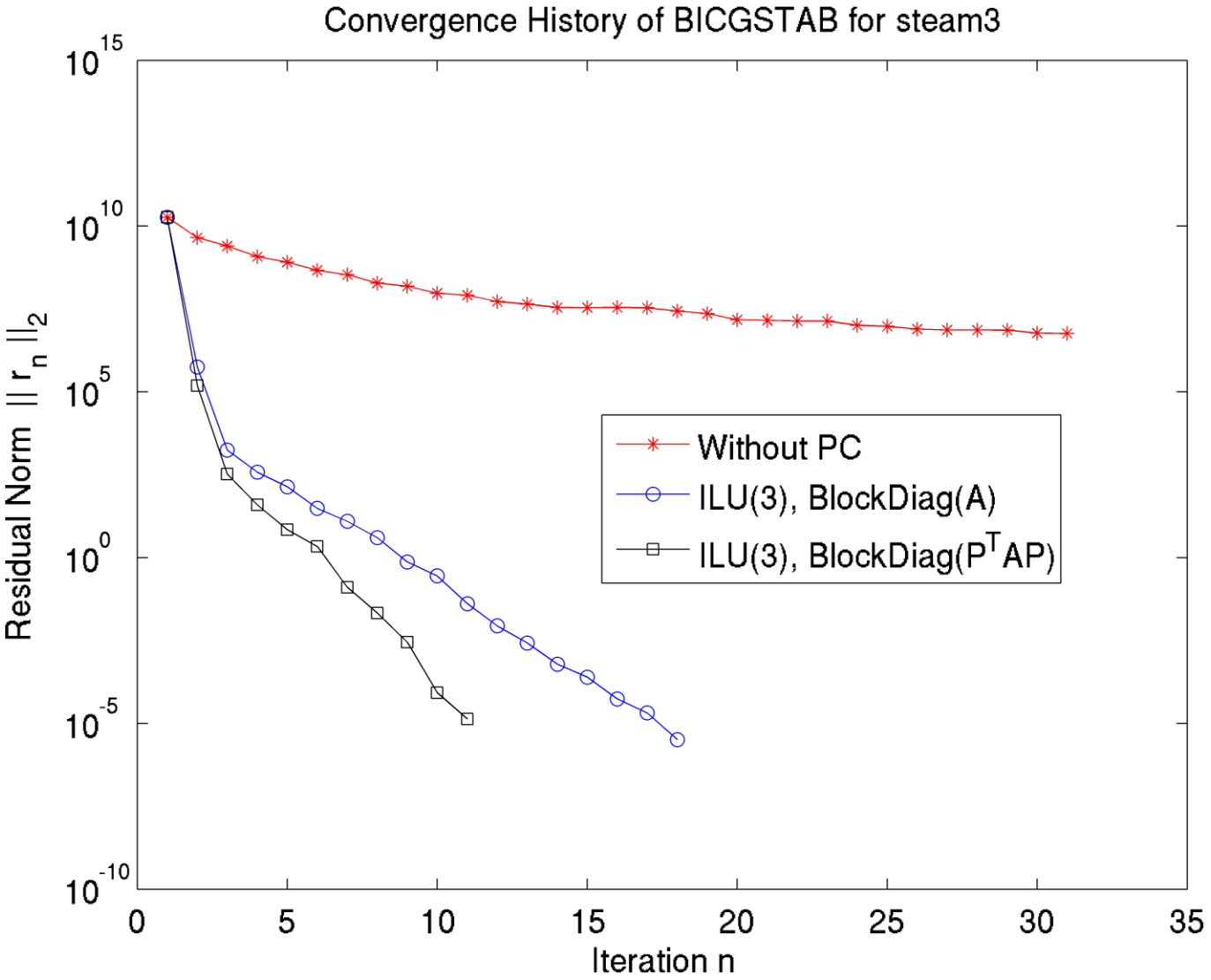
steam1


Convergence History of BICGSTAB for steam1
Residual Norm || r_n ||_2
Iteration n
Without PC
ILU(1), Init
ILU(1), Init & Add
10^10
10^5
10^0
10^-5
10^-10
0
5
10
15
20
25
30
35
40
45

Vergleich Farbanzahlen



steam3

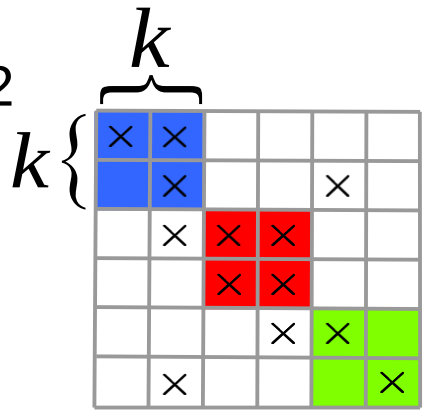

Convergence History of BICGSTAB for steam3
Residual Norm || r_n ||_2
Iteration n
Without PC
ILU(3), BlockDiag(A)
ILU(3), BlockDiag(P^T AP)
10^15
10^10
10^5
10^0
10^-5
10^-10
0
5
10
15
20
25
30
35

Block Size Identifying Problem

max$_k$ #req. elements(k)/ #colors(k)^2

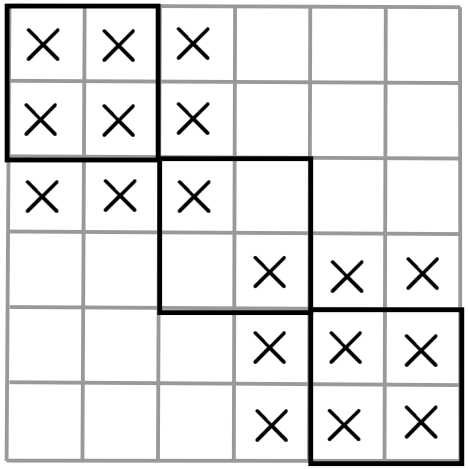
s.t. #colors(k) < c

#required elements(k) < r



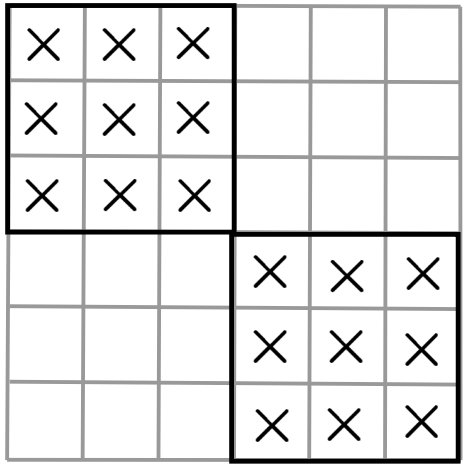
- Find block size k such that ratio of number of required elements and number of colors (squared) is maximized

Block Size?



k=2:

- #req. elements(*k*)=10
- #colors(*k*)=3



k=3:

- #req. elements(*k*)=18
- #colors(*k*)=3

QUADFLOW

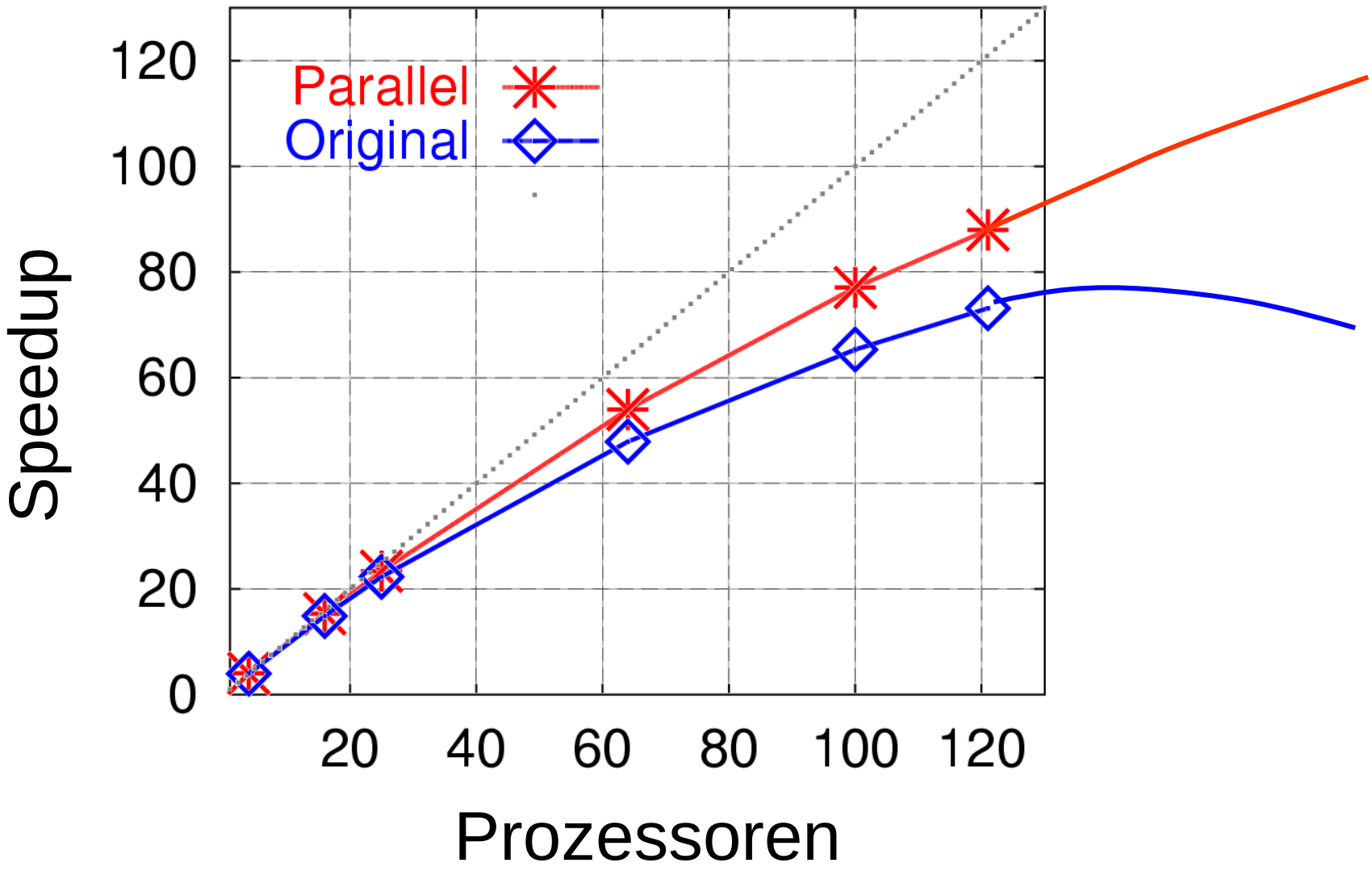
Matrix J	dim	k	#nonzeros J	#nonzeros $\tilde{J}$	#colors J	#colors $\tilde{J}$	#iterations w/o pc	#iterations w/ pc
Quadflow	1.600	4	30.720	6.400	36	8	177	26

University of Florida Sparse Matrix Collection

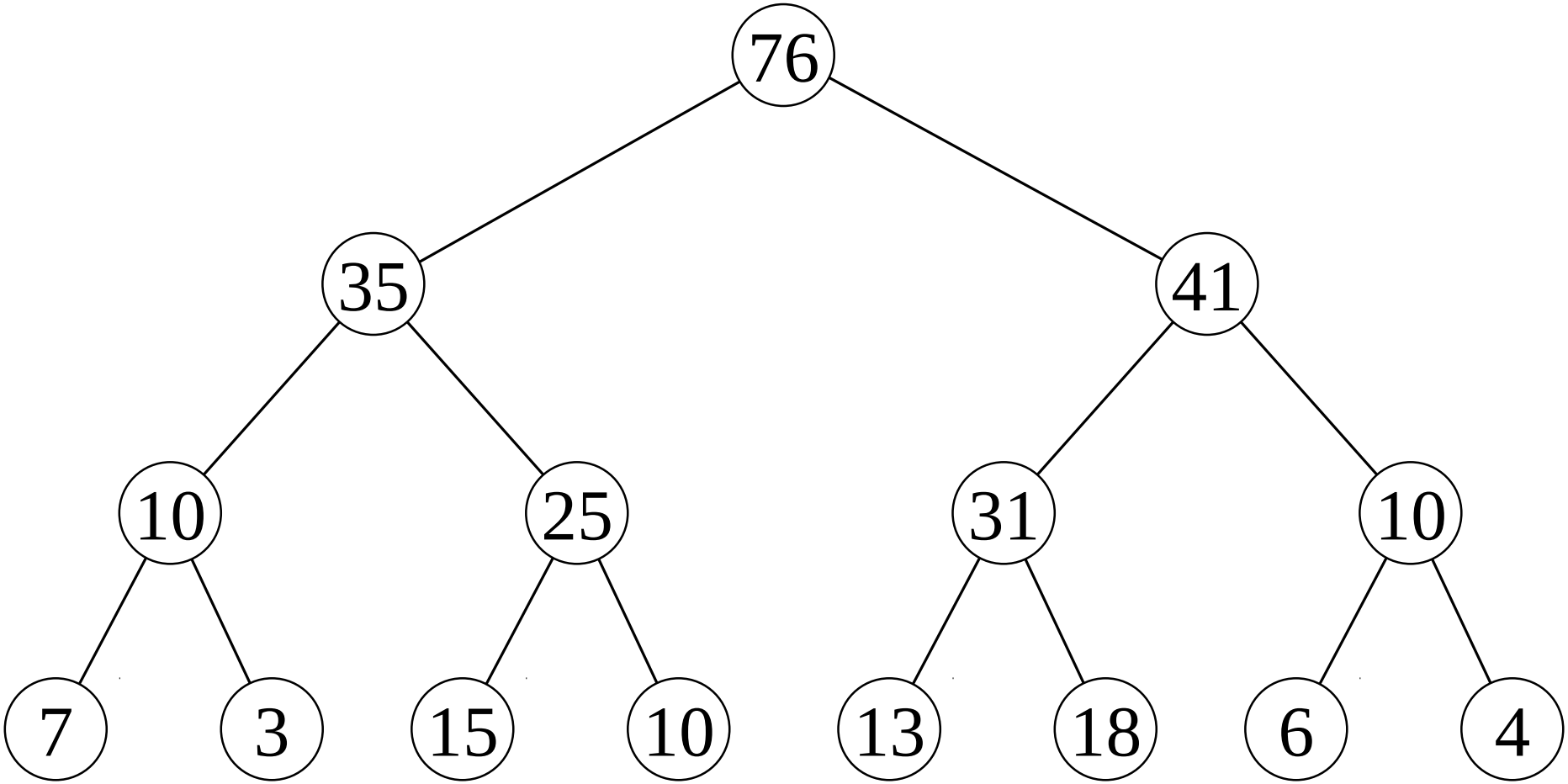
Matrix J	dim	k	#nonzeros J	#nonzeros $\tilde{J}$	#colors J	#colors $\tilde{J}$	#iterations w/o pc	#iterations w/ pc
685_bus	685	1	3.249	685	14	6	213	10
bcsstk10	1.086	5	22.070	2.810	27	12	536	1
fs_541_1	541	1	4.282	541	15	7	9	5
sherman2	1.080	6	23.094	4.950	55	12	364	1

```
for i = 1:n
    % All entries of A are entries of F with level = 0.
    index_nnz = find( A(i,:)==1 );
    level(i,index_nnz) = 0;
    F(i,index_nnz) = 1;
    % Update phase
    for k = 1:i-1  % node (i,k) with k<i
        if F(i,k)==1
            for j = k + find( F(k,k+1:n)==1 ) % node (k,j) with k<j
                lev = level(i,k) + level(k,j) + 1;
                if(lev <= el)
                    if F(i,j)==0
                        F(i,j) = 1;
                        level(i,j) = lev;
                    else
                        level(i,j) = min(level(i,j),lev);
                    end;
                end
            end;
        end
    end
end
```

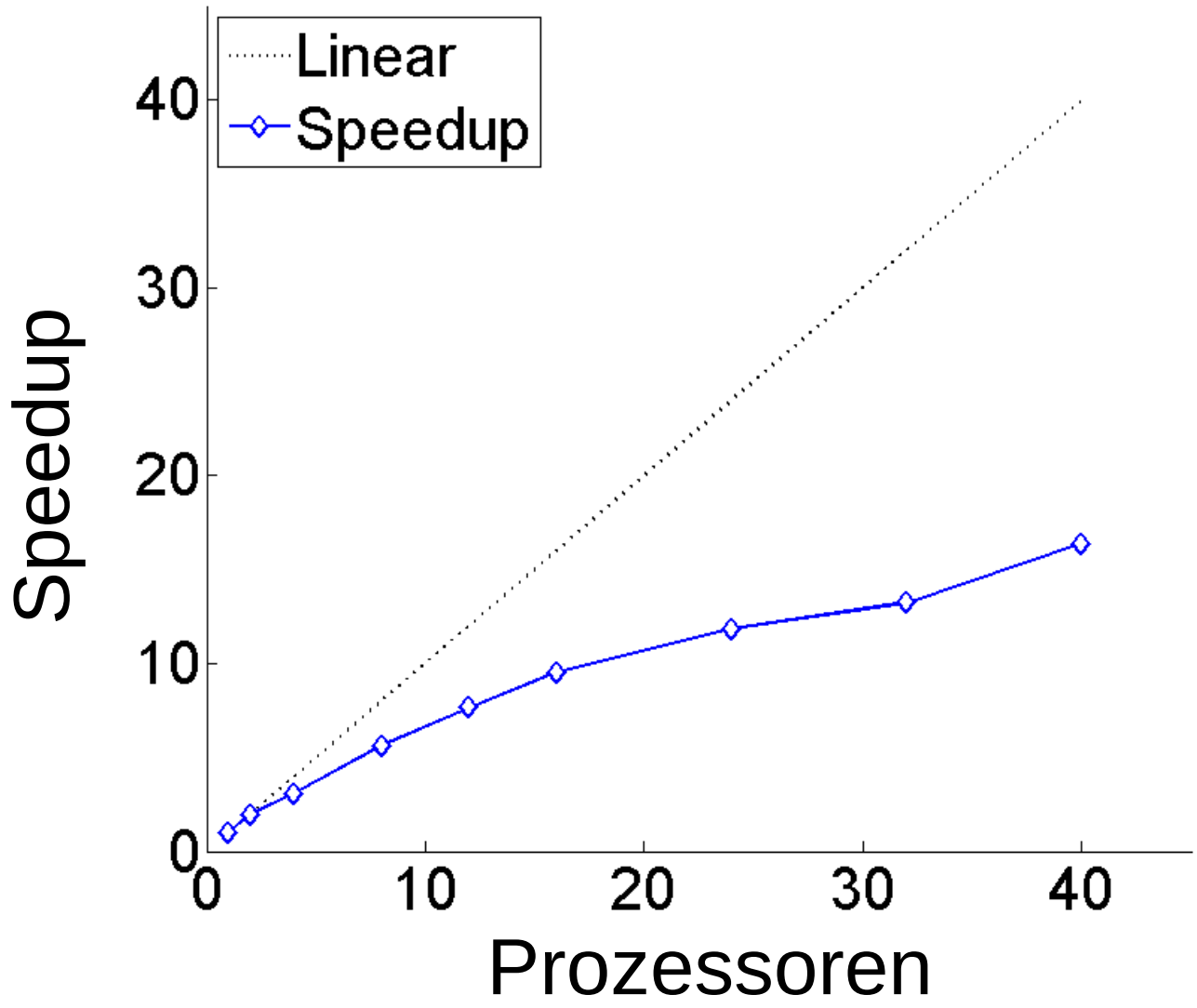
Neue parallele Varianten von QMR



Rekursives Doppeln



Zweiphasen-Problem



Zweiphasen-Strömung

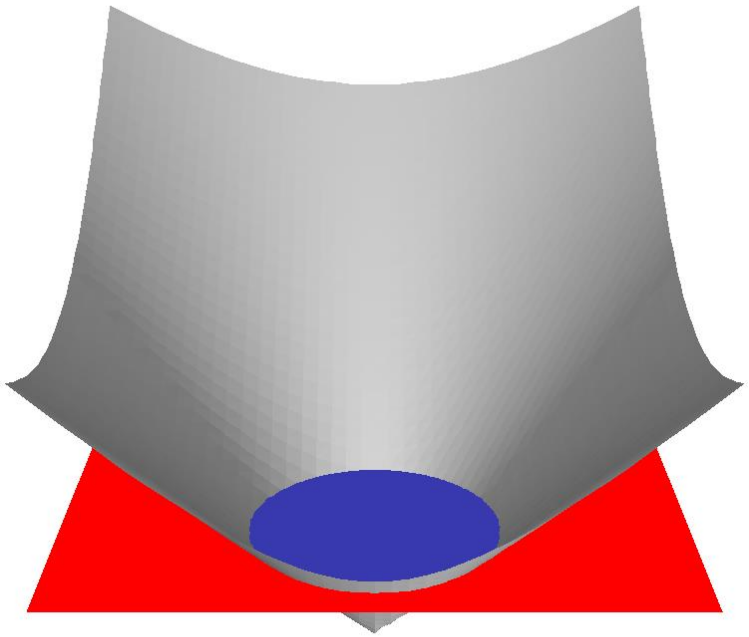
$$\rho(\varphi)\left(\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\mathbf{u}\right) = -\nabla p + \rho(\varphi)\mathbf{g} + \nabla\cdot(\mu(\varphi)\mathbf{D}(\mathbf{u})) + f_\Gamma$$

$$\nabla\cdot\mathbf{u} = 0$$

$$\frac{\partial \varphi}{\partial t} + \mathbf{u}\cdot\nabla\varphi = 0$$

Reparametrisierung:

$$\|\nabla\varphi\|_2 = 1$$



DROPS