Résolution de systèmes linéaires

Gérard MEURANT
CEA/DIF
France
(gerard.meurant@cea.fr)
Roscoff, Juin 2005

June 12, 2005

- Méthodes directes
- Méthodes de Krylov (matrices non symétriques)
- Méthode du gradient conjugué (matrices symétriques définies positives)
- Préconditioneurs

○ Non traité : méthodes multigrilles, mais

○ Préconditionnement multigrille algébrique

- Non traité : problèmes de point-selle

$$\mathcal{A} = \begin{pmatrix} A & B^T \\ B & 0 \end{pmatrix}$$

Voir article de revue de Benzi, Golub et Liesen (Acta Numerica 2006)

disponible sur le site Web de M. Benzi (Emory University)

- Effleuré : parallélisme

Méthodes directes de résolution de systèmes linéaires

Gérard MEURANT
CEA/DIF
France
(gerard.meurant@cea.fr)
Roscoff, Juin 2005

June 9, 2005

$$A = \begin{pmatrix} a_{1,1} & \dots & a_{1,n} \\ \vdots & \vdots & \vdots \\ a_{n,1} & \dots & a_{n,n} \end{pmatrix}$$

A grande et creuse, b donné

On veut résoudre

$$Ax = b$$

On suppose $\det(A) \neq 0$

Elimination de Gauss $\equiv$ factorisation LU

Proposition

Si la factorisation $A = LU$ existe, elle est unique

Théorème

Une matrice non singulière A a une factorisation LU unique ssi tous les mineurs principaux de A sont non nuls

$$A\begin{pmatrix} 1 & 2 & \dots & k \\ 1 & 2 & \dots & k \end{pmatrix} \neq 0, \quad k = 1, \dots, n$$

avec

$$A\begin{pmatrix} i_1 & i_2 & \dots & i_p \\ k_1 & k_2 & \dots & k_p \end{pmatrix} = \begin{vmatrix} a_{i_1,k_1} & a_{i_1,k_2} & \dots & a_{i_1,k_p} \\ a_{i_2,k_1} & a_{i_2,k_2} & \dots & a_{i_2,k_p} \\ \vdots & \vdots & & \vdots \\ a_{i_p,k_1} & a_{i_p,k_2} & \dots & a_{i_p,k_p} \end{vmatrix}$$

Théorème

Soit A une matrice non singulière. Il existe une matrice de permutation P tq

$$PA = LU$$

où L est triangulaire inférieure (diagonale unité) et U est triangulaire supérieure

La matrice P correspond à des permutations de ligne (pivotage partiel)

$$Ax = b$$

est transformé en

$$PAx = LUx = Pb$$

que l'on résout par

$$Ly = Pb, \quad Ux = y$$

en résolvant deux systèmes triangulaires

On obtient la solution "exacte" du système

Pivotage complet

On permute les lignes et les colonnes, en cherchant le $\max_{i,j} |a_{i,j}^{(k)}|$

$$PAQ = LU$$

où P et Q sont des matrices de permutation

Il existe d'autres techniques de pivotage (Rook's)

Nombre d'opérations : $\frac{2}{3}n(n^2-1)$ multiplications et additions et $n-1$ divisions pour une matrice dense

Solutions des systèmes triangulaires : $2n(n-1)+n$ opérations

Systèmes définis positifs

Lemme

Soit A symétrique et définie positive

$$A = \begin{pmatrix} A_{1,1} & A_{2,1}^{\mathrm{T}} \\ A_{2,1} & A_{2,2} \end{pmatrix}$$

où les blocs $A_{1,1}$ et $A_{2,2}$ sont carrés. Alors,

$$S_{2,2} = A_{2,2} - A_{2,1} A_{1,1}^{-1} A_{2,1}^{\mathrm{T}}$$

est symétrique et définie positive

$S_{2,2}$ s'appelle le complément de Schur (de $A_{2,2}$ dans A)

On applique le lemme avec

$$A = \begin{pmatrix} a_{1,1} & a_1^{\mathrm{T}} \\ a_1 & B_1 \end{pmatrix}$$

Si A est définie positive, A_2 l'est aussi

Théorème

Une matrice A a une factorisation $A = LDL^{\mathrm{T}}$, avec L triangulaire inférieure à diagonale unité et D diagonale, $\mathrm{diag}(D) > 0$, ssi A est symétrique et définie positive

H–matrices

• Une matrice A est réductible ssi $\exists$ une matrice de permutation P tq

$$P^{-1}AP = \begin{pmatrix} D_1 & 0 \\ F & D_2 \end{pmatrix}$$

D_1 et D_2 carrées

Une matrice qui n'est pas réductible est dite irréductible

• – A est à diagonale dominante (par lignes)

$$|a_{i,i}| \geq \sum_{j=1, j\neq i}^{n} |a_{i,j}|, \ \forall i$$

– A est strictement à diagonale dominante (par lignes)

$$|a_{i,i}| > \sum_{j=1, j\neq i}^{n} |a_{i,j}|, \ \forall i$$

– A est à diagonale dominante irréductible (par lignes) si, (a) A est irréductible, (b) A est à diagonale dominante (par lignes) et (c) $\exists\ i_0$ tq

$$|a_{i_0,i_0}| > \sum_{j=1, j\neq i_0}^{n} |a_{i_0,j}|$$

• A est une M–matrice ssi $a_{i,j} \leq 0$ pour $i \neq j$ et $A^{-1} \geq 0$

Si A est une M–matrice symétrique, A est définie positive

Soit A, on définit $M(A)$ comme la matrice d'éléments $m_{i,j}$ tq

$$m_{i,i} = |a_{i,i}|, \quad m_{i,j} = -|a_{i,j}|,\ \forall i, j,\ i \neq j$$

• A est une H–matrice ssi $M(A)$ est une M–matrice

• Une matrice A est à diagonale dominante généralisée (par ligne) si $\exists\ d$ avec $d_i > 0, \forall i$ tq

$$|a_{i,i}|d_i \geq \sum_{j=1, j\neq i}^{n} |a_{i,j}|d_j, \ \ \forall i$$

A est à diagonale strictement dominante généralisée (par ligne) si

$$|a_{i,i}|d_i > \sum_{j=1, j\neq i}^{n} |a_{i,j}|d_j, \ \ \forall i.$$

Théorème

A est une M–matrice ssi $a_{i,j} \leq 0, \forall i \neq j$ et A est à diagonale strictement dominante généralisée

Théorème

A est une H–matrice ssi A est à diagonale strictement dominante généralisée

Soit

$$\Omega_B = \{A | B \leq M(A)\}$$

Théorème

Soit B une M–matrice. $\forall\ A \in \Omega_B$

$$A = LU$$

où L est triangulaire inférieure à diagonale unité et U est triangulaire supérieure. En particulier, pour toute H-matrice, il existe une factorisation LU

A l'étape $k+1$ de l'élimination

$$A_{k+1} = \begin{pmatrix} a_{1,1}^{(k+1)} & \cdots & \cdots & \cdots & \cdots & a_{1,n}^{(k+1)} \\ & \ddots & & & & \vdots \\ & & a_{k,k}^{(k+1)} & \cdots & \cdots & a_{k,n}^{(k+1)} \\ & & 0 & a_{k+1,k+1}^{(k+1)} & \cdots & a_{k+1,n}^{(k+1)} \\ & & \vdots & \vdots & & \vdots \\ & & 0 & a_{n,k+1}^{(k+1)} & \cdots & a_{n,n}^{(k+1)} \end{pmatrix}$$

$$\begin{aligned} a_{i,j}^{(k+1)} &= a_{i,j}^{(k)} - \frac{a_{i,k}^{(k)} a_{k,j}^{(k)}}{a_{k,k}^{(k)}}, \quad k+1 \leq i \leq n, \quad k \leq j \leq n \\ a_{i,j}^{(k+1)} &= a_{i,j}^{(k)}, \quad 1 \leq i \leq k, \quad 1 \leq j \leq n \\ & \qquad \text{et } k+1 \leq i \leq n, \quad 1 \leq j \leq k-1 \end{aligned}$$

Systèmes creux

On ne stocke que les éléments non nuls

Il existe de nombreux schémas de stockage. Le plus utilisé est

$$A = \begin{pmatrix} a_1 & 0 & 0 & a_2 \\ a_3 & a_4 & a_5 & 0 \\ 0 & a_6 & a_7 & 0 \\ a_8 & 0 & 0 & a_9 \end{pmatrix}$$

	1	2	3	4	5	6	7	8	9
AA	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9
JA	1	4	1	2	3	2	3	1	4
IA	1	3	6	8	10				

Autre schéma (listes liées)

$$A = \begin{pmatrix} a_1 & 0 & 0 & a_2 \\ a_3 & a_4 & a_5 & 0 \\ 0 & a_6 & a_7 & 0 \\ a_8 & 0 & 0 & a_9 \end{pmatrix}$$

	1	2	3	4	5	6	7	8	9
`AA`	a_3	a_2	a_9	a_1	a_4	a_8	a_6	a_5	a_7
`JA`	1	4	4	1	2	1	2	3	3
`IPA`	5	0	0	2	8	3	9	0	0
`IA`	4	1	7	6					

Remplissage

$$a_{i,j}^{(k+1)} = a_{i,j}^{(k)} - \frac{a_{i,k}^{(k)} a_{k,j}^{(k)}}{a_{k,k}^{(k)}}$$

$$A = \begin{pmatrix} x & x & 0 & x & 0 \\ x & x & x & 0 & 0 \\ 0 & x & x & 0 & x \\ x & 0 & 0 & x & 0 \\ 0 & 0 & x & 0 & x \end{pmatrix}$$

$$A_2 = \begin{pmatrix} x & x & 0 & x & 0 \\ 0 & x & x & \bullet & 0 \\ 0 & x & x & 0 & x \\ 0 & \bullet & 0 & x & 0 \\ 0 & 0 & x & 0 & x \end{pmatrix}$$

$$A_3 = \begin{pmatrix} x & x & 0 & x & 0 \\ 0 & x & x & \bullet & 0 \\ 0 & 0 & x & \bullet & x \\ 0 & 0 & \bullet & x & 0 \\ 0 & 0 & x & 0 & x \end{pmatrix}$$

$$A_4 = \begin{pmatrix} x & x & 0 & x & 0 \\ 0 & x & x & \bullet & 0 \\ 0 & 0 & x & \bullet & x \\ 0 & 0 & 0 & x & \bullet \\ 0 & 0 & 0 & \bullet & x \end{pmatrix}$$

$$A_5 = \begin{pmatrix} x & x & 0 & x & 0 \\ 0 & x & x & \bullet & 0 \\ 0 & 0 & x & \bullet & x \\ 0 & 0 & 0 & x & \bullet \\ 0 & 0 & 0 & 0 & x \end{pmatrix}$$

$$L = \begin{pmatrix} x & & & & \\ x & x & & & \\ 0 & x & x & & \\ x & \bullet & \bullet & x & \\ 0 & 0 & x & \bullet & x \end{pmatrix}$$

Le remplissage dépend de la numérotation

$$A = \begin{pmatrix} x & x & x & x \\ x & x & & \\ x & & x & \\ x & & & x \end{pmatrix}$$

$$L = \begin{pmatrix} x & & & \\ x & x & & \\ x & \bullet & x & \\ x & \bullet & \bullet & x \end{pmatrix}$$

En numérotant autrement

$$PAP^{\mathrm{T}} = \begin{pmatrix} x & & & x \\ & x & & x \\ & & x & x \\ x & x & x & x \end{pmatrix}$$

Pas de remplissage

Plus de remplissage = plus de stockage et plus d'opérations

But : minimiser le remplissage

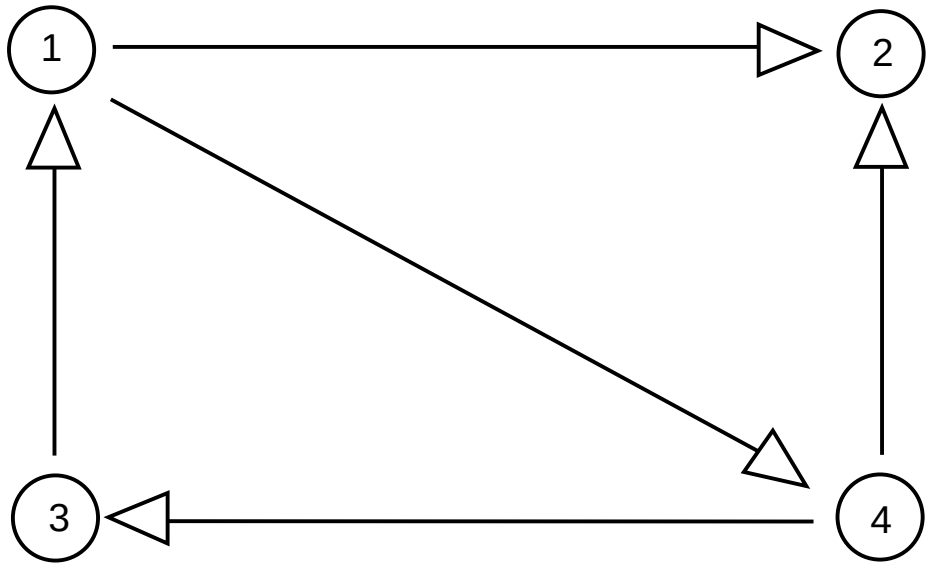
Définitions sur les graphes

$$G = (X, E)$$

X sommets et E arcs

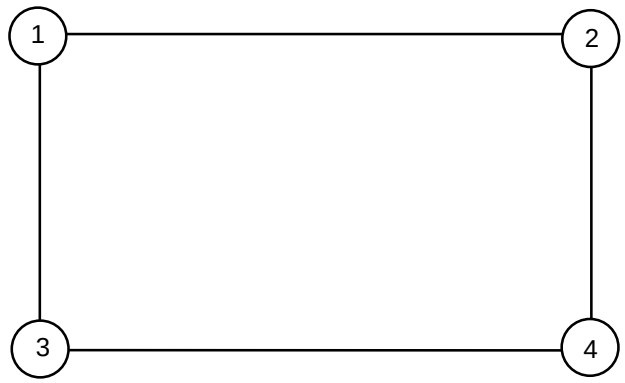
arc de i à j si $a_{i,j} \neq 0$

$$A = \begin{pmatrix} x & x & 0 & x \\ 0 & x & 0 & 0 \\ x & 0 & x & 0 \\ 0 & x & x & x \end{pmatrix}$$



Graphe orienté

$$A = \begin{pmatrix} x & x & x & 0 \\ x & x & 0 & x \\ x & 0 & x & x \\ 0 & x & x & x \end{pmatrix}$$



Graphe non orienté

Deux nœuds x et y sont adjacents si $\{x, y\} \in E$

$$\text{Adj}(y) = \{x \in X |\ x \quad \text{est adjacent à} \quad y\}$$

Si $Y \subset X$

$$\text{Adj}(Y) = \{x \in X |\ x \in \text{Adj}(y), x \notin Y, y \in Y\}$$

$$\deg(x) = |\text{Adj}(x)|$$

Soient x et $y \in X$, un chemin de longueur l de x à y est

$$\{\nu_1, \nu_2, \ldots, \nu_{l+1}\}$$

tq $x = \nu_1$, $y = \nu_{l+1}$ et $\{\nu_i, \nu_{i+1}\} \in E$, $1 \leq i \leq l$

La distance $d(x, y)$ entre 2 nœuds x et y est la longueur du plus court chemin entre x et y

L'excentricité d'un nœud est

$$e(x) = \max\{d(x, y) | y \in X\}$$

Le diamètre δ de G est

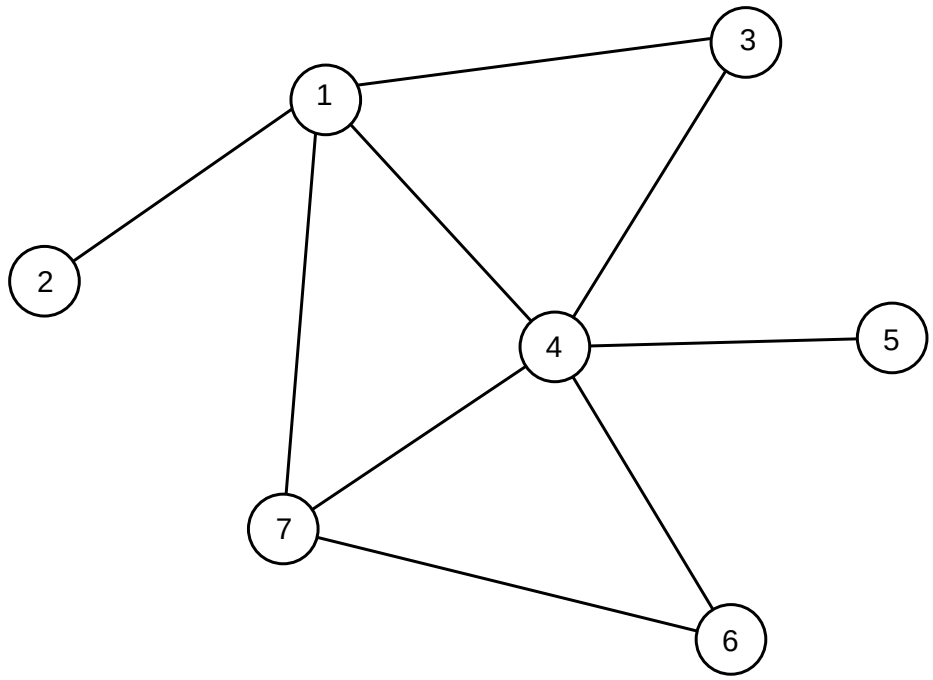
$$\delta(G) = \max\{e(x) | x \in X\}$$

Soient $G^{(i)}$, $G^{(1)} = G$ les graphes correspondant aux étapes de l'élimination de Gauss

Théorème

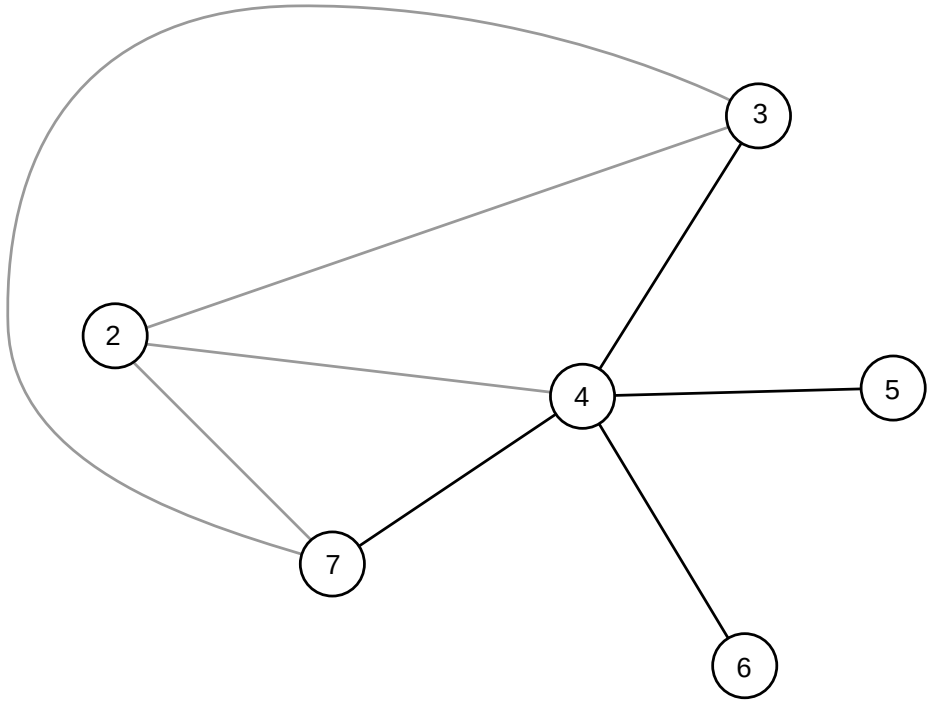
$G^{(i+1)}$ est obtenu d'après $G^{(i)}$ en supprimant x_i du graphe et tous les arcs incidents et en ajoutant des arcs tq tous les voisins de x_i soient connectés 2 à 2

$$A = \begin{pmatrix} x & x & x & x & & & x \\ x & x & & & & & \\ x & & x & x & & & \\ x & & x & x & x & x & x \\ & & & x & x & & \\ & & & x & & x & x \\ x & & & x & & x & x \end{pmatrix}$$

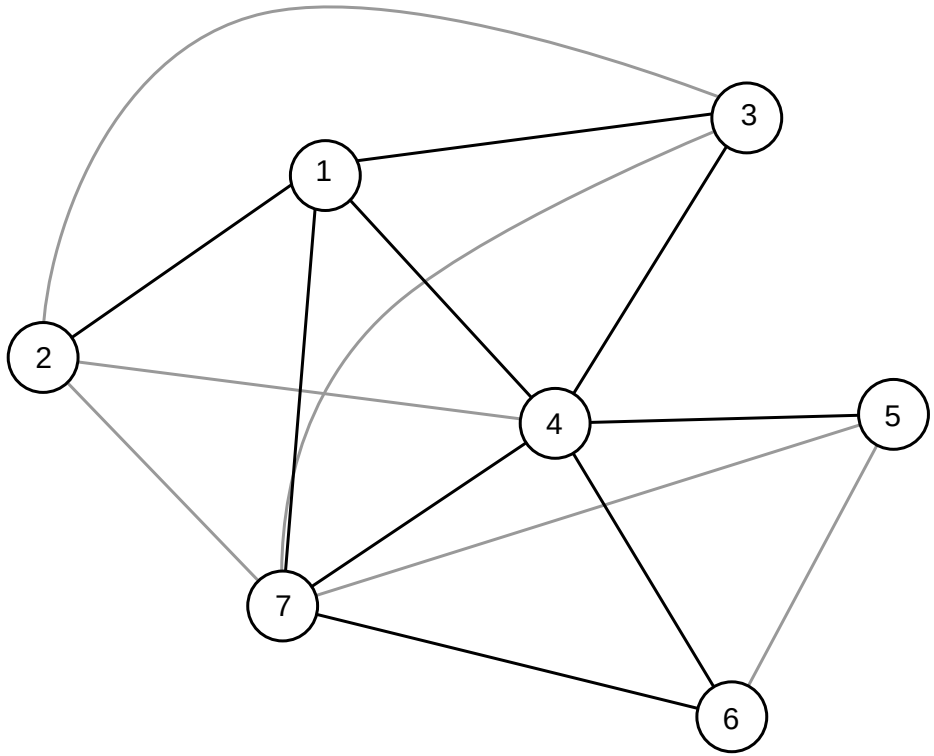


Graphe de A

$$A^{(2)} = \begin{pmatrix} x & x & x & x & & & x \\ x & x & \bullet & \bullet & & & \bullet \\ x & \bullet & x & x & & & \bullet \\ x & \bullet & x & x & x & x & x \\ & & & x & x & & \\ & & & x & & x & x \\ x & \bullet & \bullet & x & & x & x \end{pmatrix}$$



Graphe de $A^{(2)}$



Graphe de A avec remplissages

On peut renuméroter cette matrice pour ne pas avoir de remplissages

$$A' = PAP^{\mathrm{T}} = \begin{pmatrix} x & & & & x & & \\ & x & & & & x & \\ & & x & & x & x & \\ & & & x & & x & x \\ x & & x & & x & x & x \\ & x & x & x & x & x & x \\ & & & x & x & x & x \end{pmatrix}$$

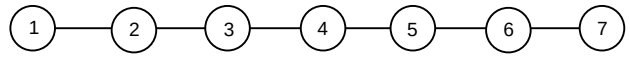
L'arbre d'élimination de A est un graphe avec n nœuds tq le nœud p est le père de j, ssi

$$p = \min\{i | i > j,\, l_{i,j} \neq 0\}$$

où L est tel que $A = LL^T$

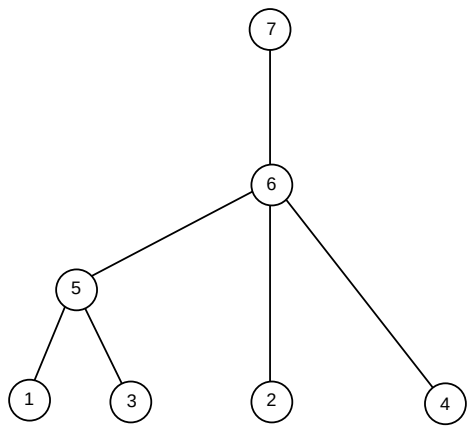
p est l'indice du premier élément non nul dans la colonne j de L

$T(A)$ est



$T(A)$

et $T(A')$



$T(A)$

Caractérisation du remplissage

Soient $S \subset X$ et $x \in X, x \notin S$, x est atteignable depuis $y \notin S$ à travers S s'il existe un chemin $(y, v_1, \ldots, v_k, x)$ de y à x tq $v_i \in S$, $i = 1, \ldots, k$

$Reach(y, S) = \{x | x \notin S, x$ est atteignable depuis y à travers $S\}$

Théorème

Soit $k > j$, il existe un remplissage entre x_j et x_k ssi

$$x_k \in Reach(x_j, \{x_1, \ldots, x_{j-1}\})$$

Matrice symétrique déf pos : on peut renuméroter avant élimination

Matrice non symétrique : renumérotation pendant l'élimination (stabilité)

Regardons le cas symétrique déf pos (ou H–matrice)

Réduction de la largeur de bande

$f_i(A) = \min\{i|\ a_{i,j} \neq 0\}$

$f_i(A)$ est l'indice de la colonne avec le premier élément non nul de la ligne i

$\beta_i(A) = i - f_i(A)$ est la largeur de bande de la ligne i

La largeur de bande de A est

$$\beta(A) = \max_i\{\beta_i(A),\, 1 \leq i \leq n\}$$

$$band(A) = \{(i,j)|0 < i - j \leq \beta(A),\, i \geq j\}$$

$Env(A) = \{(i,j)|\ 0 < i - j \leq \beta_i(A),\, i \geq j\}$ est l'enveloppe A. Le profil de A, $P_r(A)$ est

$$P_r(A) = |Env(A)| = \sum_{i=1}^{n} \beta_i(A).$$

Théorème

Soit $Fill(A) = \{(i,j)|\ i > j,\ a_{i,j} = 0,\ l_{i,j} \neq 0\}$

$$Fill(A) \subset Env(A)$$

On veut minimiser la largeur de bande et/ou le remplissage

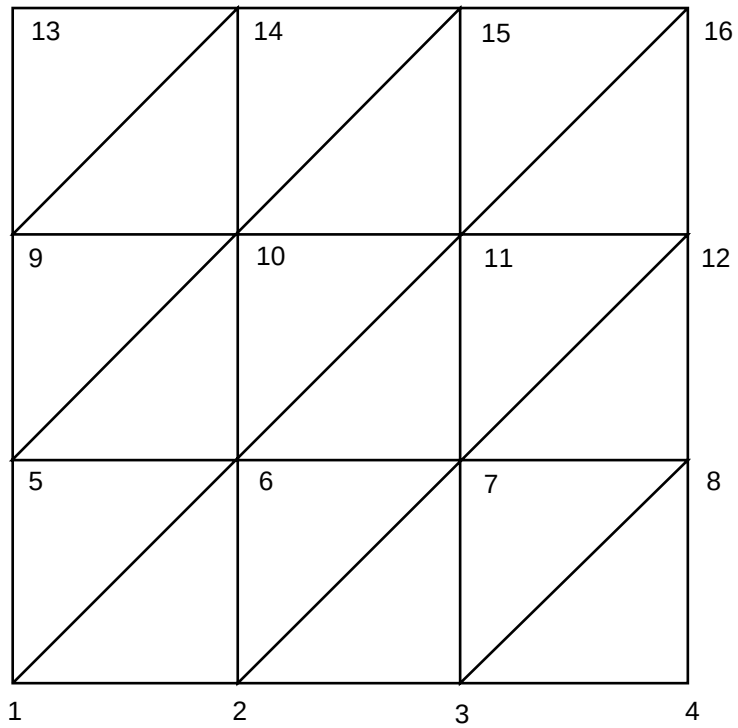
Cuthill–MccKee (CM)

1) on choisit un nœud de départ

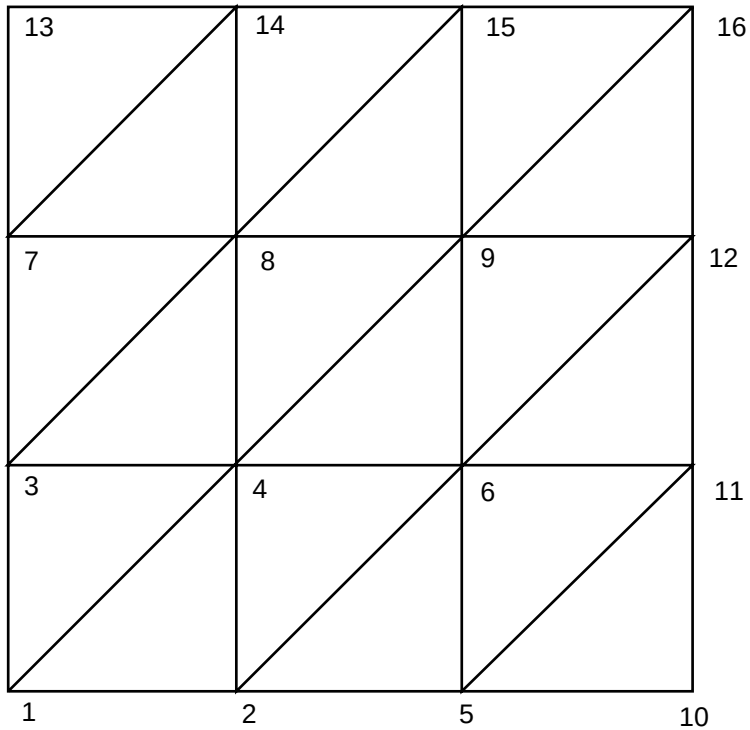
2) pour $i = 1, \ldots, n-1$ on numérote tous les voisins (non numérotés) de x_i dans $G(A)$ par degré croissant

3) on modifie les degrés des nœuds restants

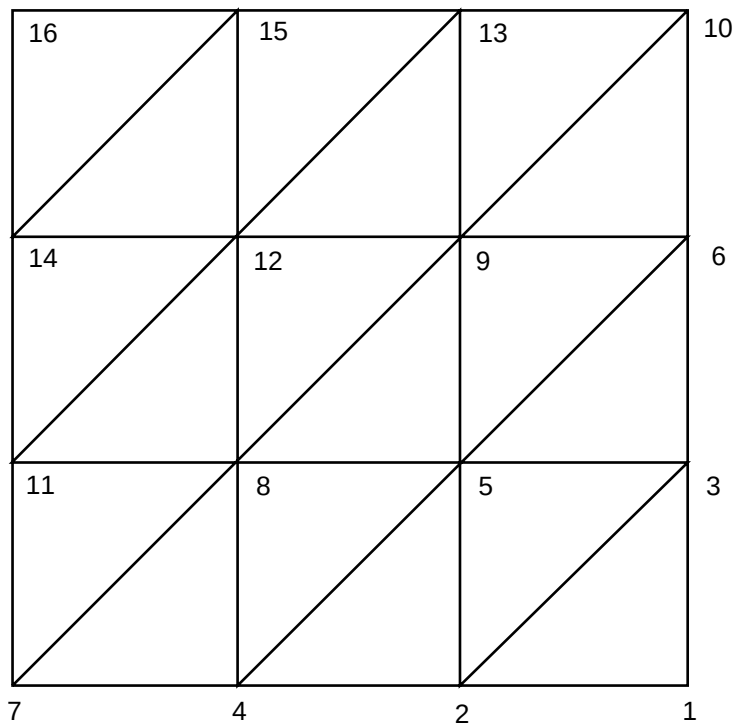
Exemple : numérotation initiale



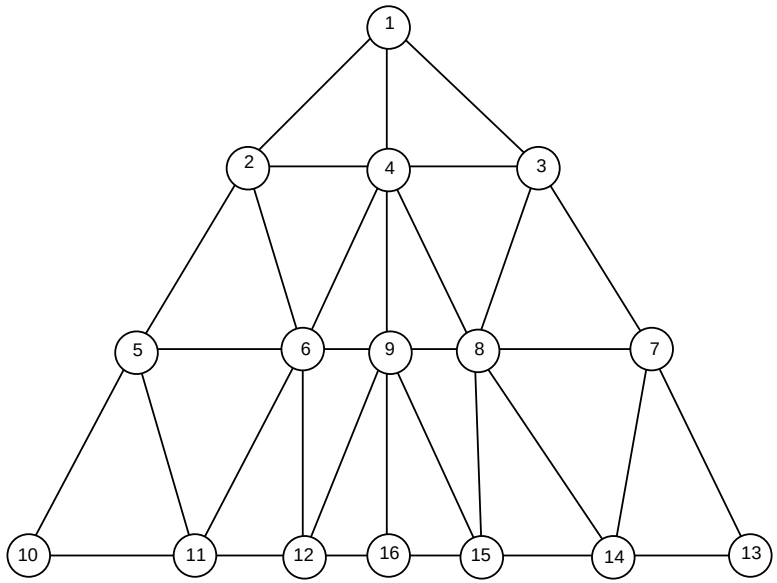
nœud de départ : 1, 38 remplissages



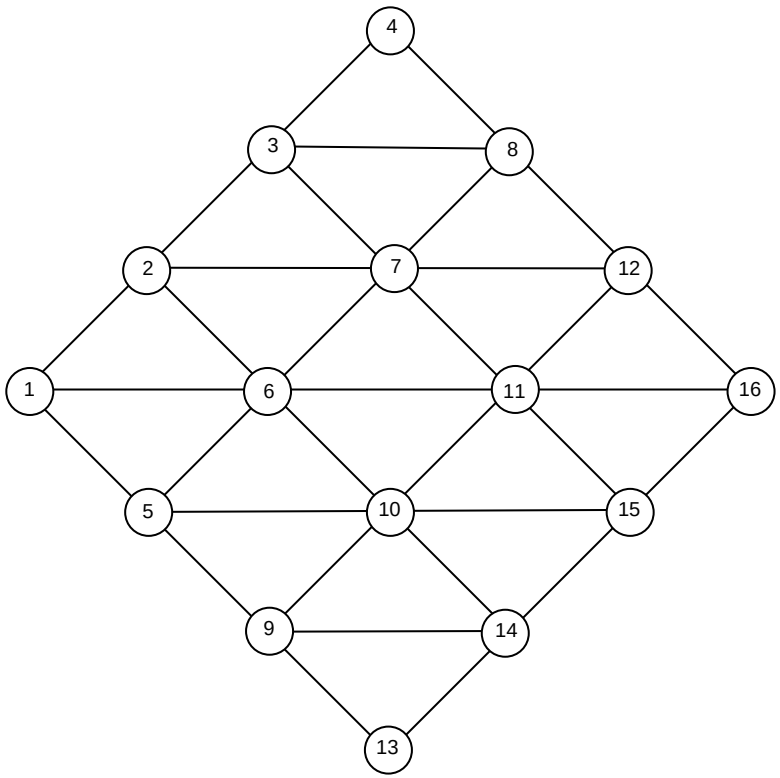
nœud de départ : 4, 14 remplissages



Structure de niveau, première numérotation



Structure de niveau, deuxième numérotation



Bon choix de départ : nœud périphérique (difficile à trouver)

Algorithme GPS (Gibbs, Poole & Stockmeyer)

1) on choisit un nœud de départ r

2) on construit la structure de niveau $\mathcal{L}(r)$

$$\mathcal{L}(r) = \{L_0(r), \ldots, L_{e(r)}(r)\}$$

3) on trie les nœuds $x \in L_{e(r)}(r)$ par degré croissant

4) pour tous les nœuds $x \in L_{e(r)}(r)$ par degré croissant on construit $\mathcal{L}(x)$. Si la hauteur de $\mathcal{L}(x)$ est plus grande que la hauteur de $\mathcal{L}(r)$, on choisit x comme nœud de départ $(r = x)$ et on va en 2)

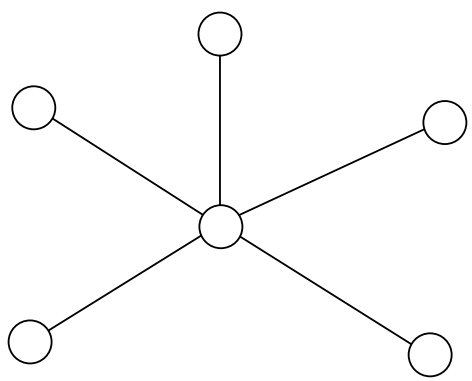
Reverse Cuthill Mc–Kee (RCM)

1) nœud de départ pseudo–périphérique (GPS)

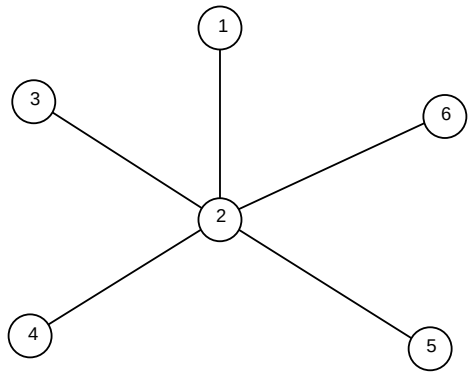
2) on génére la numérotation CM

3) on inverse la numérotation. Soit $x_1, \ldots, x_n$ donnée par CM, on choisit $\{y_i\}$ tq $y_i = x_{n+i-1}$, $i = 1, \ldots, n$

Exemple :



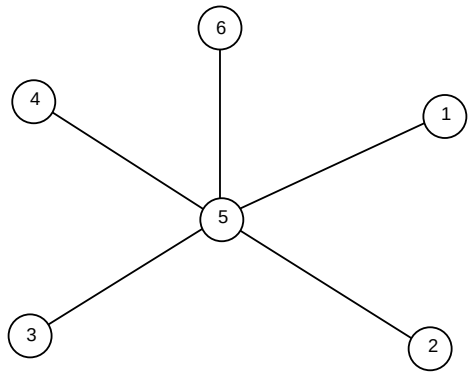
CM :



Matrice L :

$$L = \begin{pmatrix} x & & & & & \\ x & x & & & & \\ & x & x & & & \\ & x & \bullet & x & & \\ & x & \bullet & \bullet & x & \\ & x & \bullet & \bullet & \bullet & x \end{pmatrix}$$

RCM :



Matrice L :

$$L = \begin{pmatrix} x & & & & & \\ & x & & & & \\ & & x & & & \\ & & & x & & \\ x & x & x & x & x & \\ & & & & x & x \end{pmatrix}$$

Théorème

On a

$$|Env(A_{RCM})| \leq |Env(A_{CM})|$$

Lemme

Si $\forall i > 1$, $f_i < i$, l'enveloppe $Env(A)$ se remplit complètement

Donc

$$|Fill(A_{RCM})| \leq |Fill(A_{CM})|$$

Algorithme du degré minimum

1) dans $G^{(i)}$, trouver un nœud x_i tq

$$\deg(x_i) = \min_{y \in X_i} \{\deg(y)\}$$

2) on forme $G^{(i+1)}$ en éliminant x_i

3) si $i+1 < n$ on va en 1) avec $i \leftarrow i+1$

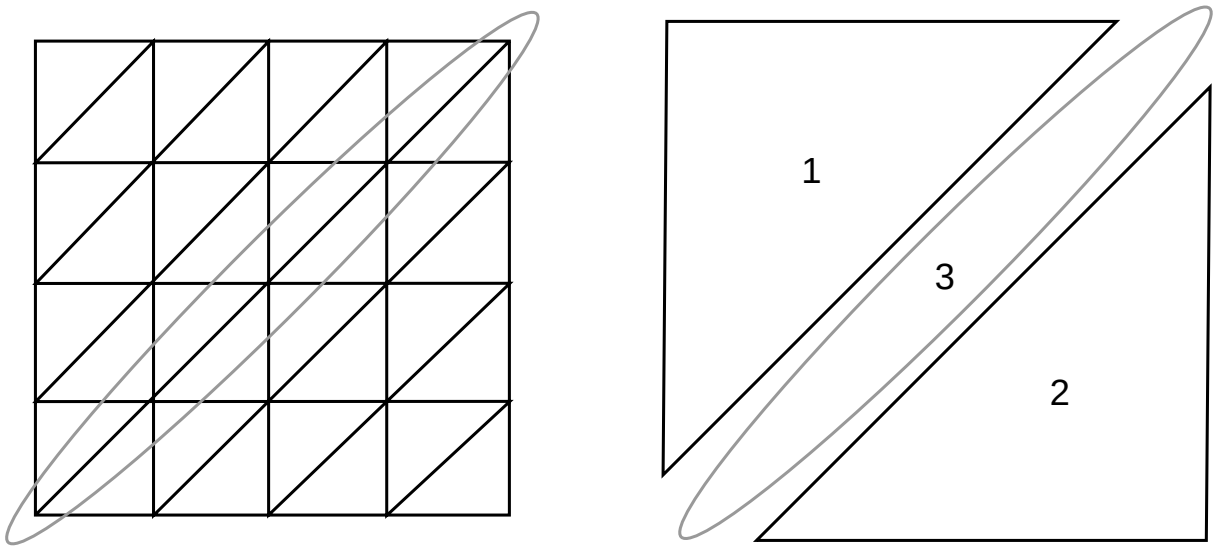
Pb : on a souvent des ex–aequo et il faut une stratégie de "tie–break"

Le résultat dépend beaucoup du choix

Ce qui coûte cher, c'est l'actualisation des degrés.
Différentes modifs ont été proposées pour améliorer çà

La dernière en date (Amestoy, Davis & Duff) est d'utiliser des bornes sur les degrés

Dissection emboîtée

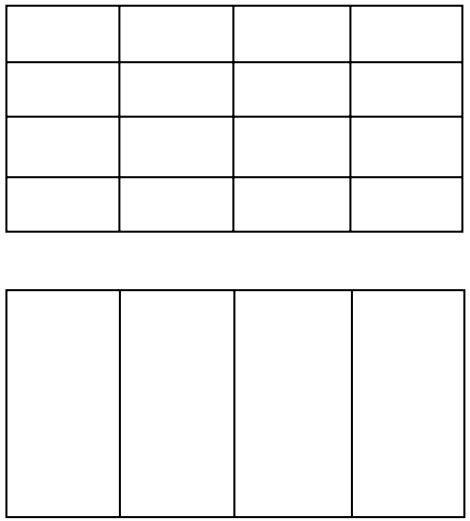


$$A = \begin{pmatrix} A_1 & 0 & A_{3,1}^{\mathrm{T}} \\ 0 & A_2 & A_{3,2}^{\mathrm{T}} \\ A_{3,1} & A_{3,2} & A_3 \end{pmatrix}$$

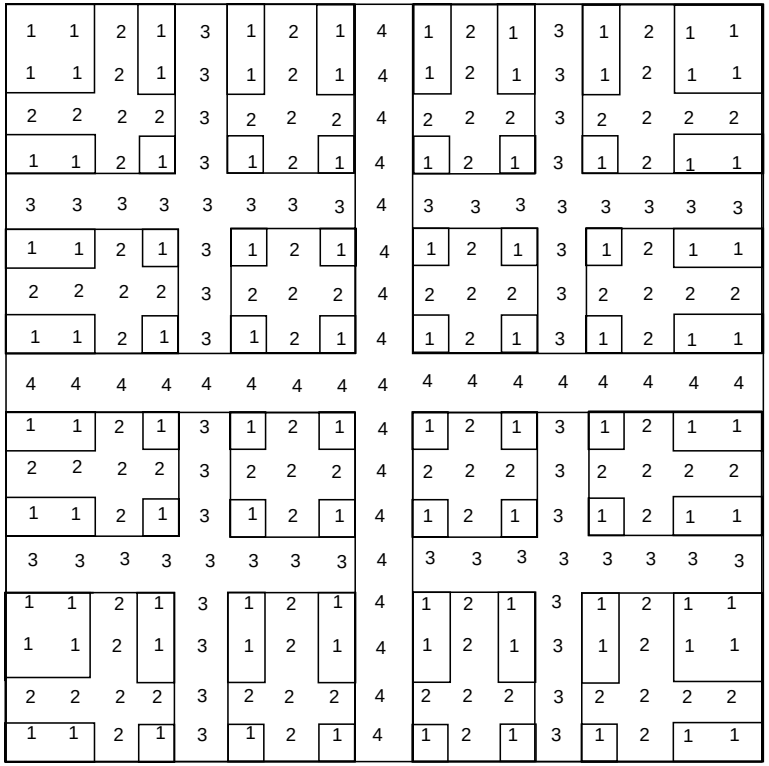
$$L = \begin{pmatrix} L_1 & & \\ 0 & L_2 & \\ L_{3,1} & L_{3,2} & L_3 \end{pmatrix}$$

Conséquence du théorème de caractérisation du remplissage

Il existe deux façons de découper un maillage régulier :



Exemple sur un maillage régulier :



Pour des problèmes quelconques, on considère le graphe de la matrice et on utilise des algorithmes de partitionnement de graphe

Méthode multifrontale

$$A = \begin{pmatrix} D & B^{\mathrm{T}} \\ B & C \end{pmatrix}$$

$$= \begin{pmatrix} L_D & 0 \\ BL_D^{-\mathrm{T}} & I \end{pmatrix} \begin{pmatrix} I & 0 \\ 0 & C - BD^{-1}B^{\mathrm{T}} \end{pmatrix} \begin{pmatrix} L_D^{\mathrm{T}} & L_D^{-1}B^{\mathrm{T}} \\ 0 & I \end{pmatrix}$$

$$D = L_D L_D^{\mathrm{T}}$$

Le complément de Schur $C - BD^{-1}B^{\mathrm{T}}$ est la prochaine matrice à factoriser. Soit D d'ordre $j-1$,

$$BD^{-1}B^{\mathrm{T}} = (BL_D^{-\mathrm{T}})(L_D^{-1}B^{\mathrm{T}}) = \sum_{k=1}^{j-1} \begin{pmatrix} l_{j,k} \\ \vdots \\ l_{n,k} \end{pmatrix} \begin{pmatrix} l_{j,k} & \dots & l_{n,k} \end{pmatrix}$$

Théorème

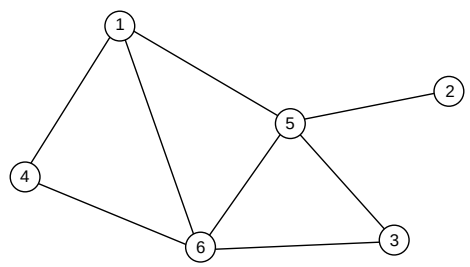
Si le nœud k est un descendant du nœud j dans $T(A)$, alors la structure de $(l_{j,k}, \ldots, l_{n,k})^{\mathrm{T}}$ est contenue dans la structure de $(l_{j,j}, \ldots, l_{n,j})^{\mathrm{T}}$

Si $l_{j,k} \neq 0$, $k < j$, le nœud k est un descendant du nœud j dans $T(A)$

Exemple :

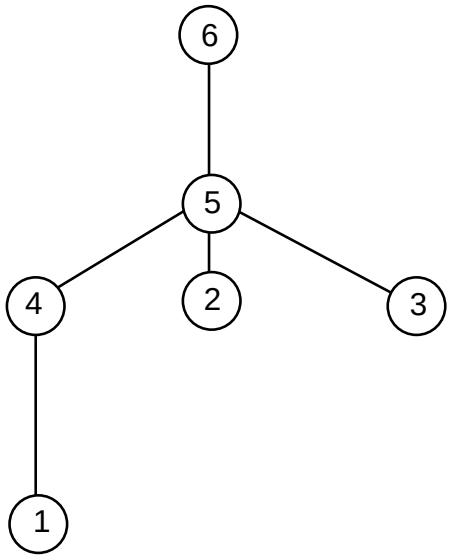
$$A = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{pmatrix} x & & & x & x & x \\ & x & & & x & \\ & & x & & x & x \\ x & & & x & & x \\ x & x & x & & x & x \\ x & & x & x & x & x \end{pmatrix} \end{array}$$

Graphe de A :



$$L = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{pmatrix} x & & & & & \\ & x & & & & \\ & & x & & & \\ x & & & x & & \\ x & x & x & \bullet & x & x \\ x & & x & x & x & x \end{pmatrix} \end{array}.$$

Arbre d'élimination $T(A)$:



On peut éliminer 1,2 et 3 indépendamment

Si on élimine 1, on peut se restreindre à

$$F_1 = \begin{array}{c} \\ 1 \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 4 & 5 & 6 \end{array} \\ \begin{pmatrix} a_{1,1} & a_{1,4} & a_{1,5} & a_{1,6} \\ a_{4,1} & & & \\ a_{5,1} & & & \\ a_{6,1} & & & \end{pmatrix} \end{array}$$

Ca crée des contributions dans la matrice réduite

$$\bar{U}_4 = \begin{array}{c} \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{ccc} 4 & 5 & 6 \end{array} \\ \begin{pmatrix} x & \bullet & x \\ \bullet & x & x \\ x & x & x \end{pmatrix} \end{array}$$

On peut éliminer 2

$$F_2 = \begin{array}{c c} & \begin{array}{cc} 2 & 5 \end{array} \\ \begin{array}{c} 2 \\ 5 \end{array} & \begin{pmatrix} a_{2,2} & a_{2,5} \\ a_{5,2} & \end{pmatrix} \end{array}$$

$$\bar{U}_5^2 = \begin{array}{cc} & 5 \\ 5 & \begin{pmatrix} x \end{pmatrix} \end{array}$$

On élimine 3,

$$F_3 = \begin{array}{cc} & \begin{array}{ccc} 3 & 5 & 6 \end{array} \\ \begin{array}{c} 3 \\ 5 \\ 6 \end{array} & \begin{pmatrix} a_{3,3} & a_{3,5} & a_{3,6} \\ a_{5,3} & & \\ a_{6,3} & & \end{pmatrix} \end{array}$$

$$\bar{U}_5^3 = \begin{array}{cc} & \begin{array}{cc} 5 & 6 \end{array} \\ \begin{array}{c} 5 \\ 6 \end{array} & \begin{pmatrix} x & x \\ x & x \end{pmatrix} \end{array}$$

Pour éliminer 4, il faut tenir compte du résultat de l'élimination de 1

$$F_4 = \begin{array}{c} \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{ccc} 4 & 5 & 6 \end{array} \\ \begin{pmatrix} a_{4,4} & 0 & a_{4,6} \\ 0 & & \\ a_{6,4} & & \end{pmatrix} \end{array} + \bar{U}_4$$

$$\bar{U}_5^4 = \begin{array}{c} \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{cc} 5 & 6 \end{array} \\ \begin{pmatrix} x & x \\ x & x \end{pmatrix} \end{array}$$

Pour éliminer 5, il faut tenir compte des contributions de 2,3 et 4. Il faut étendre les contributions au bon ensemble d'indices

$$\bar{U}_5 = \bar{U}_5^2 \circ \bar{U}_5^3 \circ \bar{U}_5^4$$

$$F_5 = \begin{pmatrix} a_{5,5} & a_{5,6} \\ a_{6,5} & 0 \end{pmatrix} + \bar{U}_5$$

On n'a travaillé que sur des petites matrices pleines (!)

Matrices creuses non symétriques

Pb : pivotage

On ne peut pas déterminer la numérotation a priori

Structures de données plus compliquées

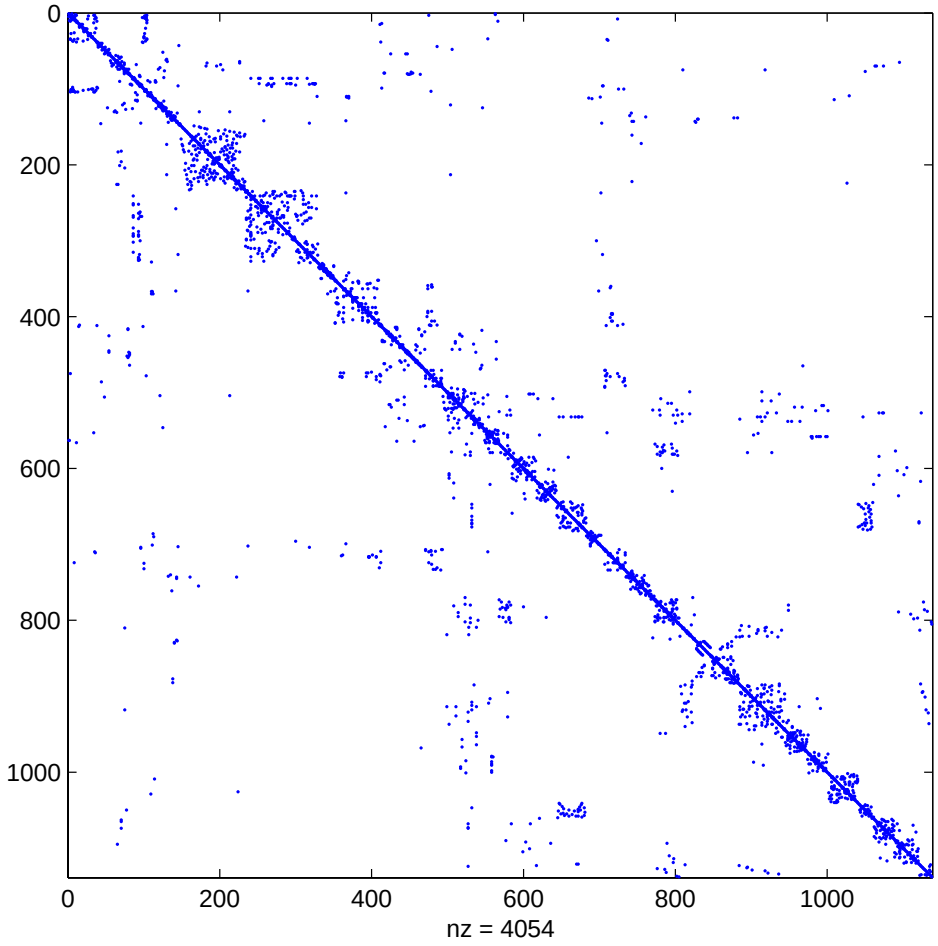
Pour choisir le pivot, il faut relaxer un petit peu, sinon on ne contrôle pas le remplissage

Critère de Markowitz

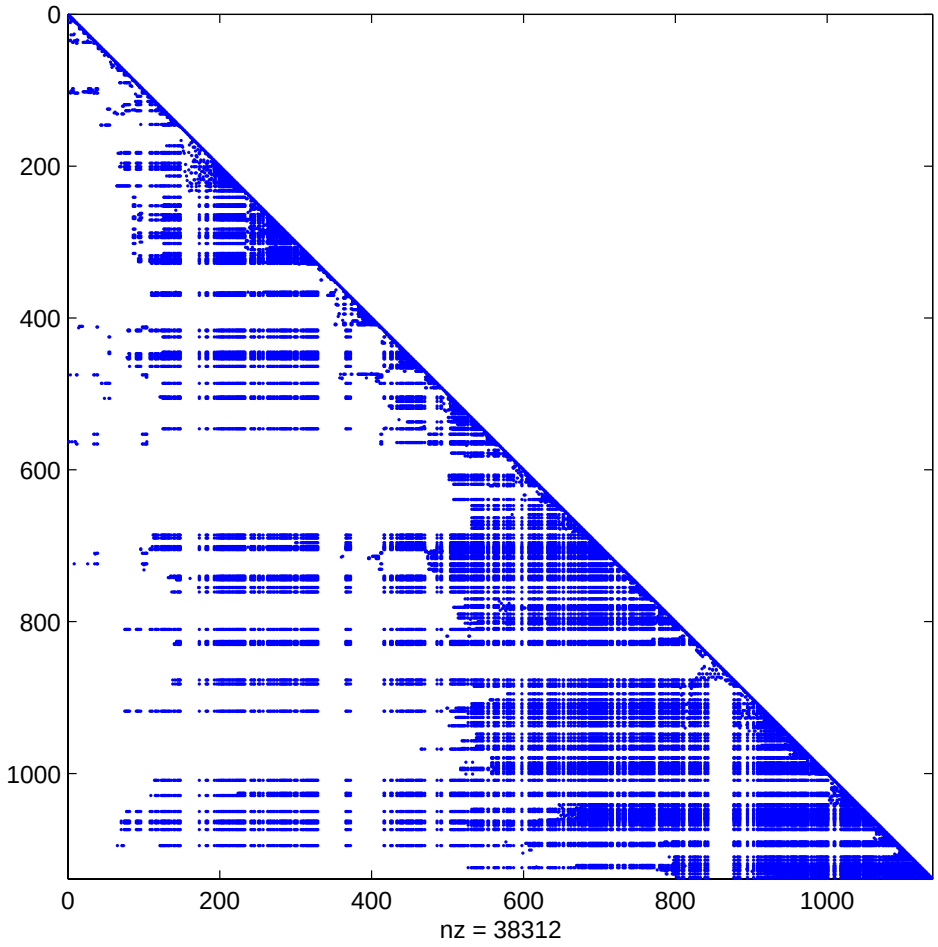
$$|a_{i,j}^{(k)}| \geq u \max_l |a_{l,j}^{(k)}|$$

On choisit le candidat qui minimise

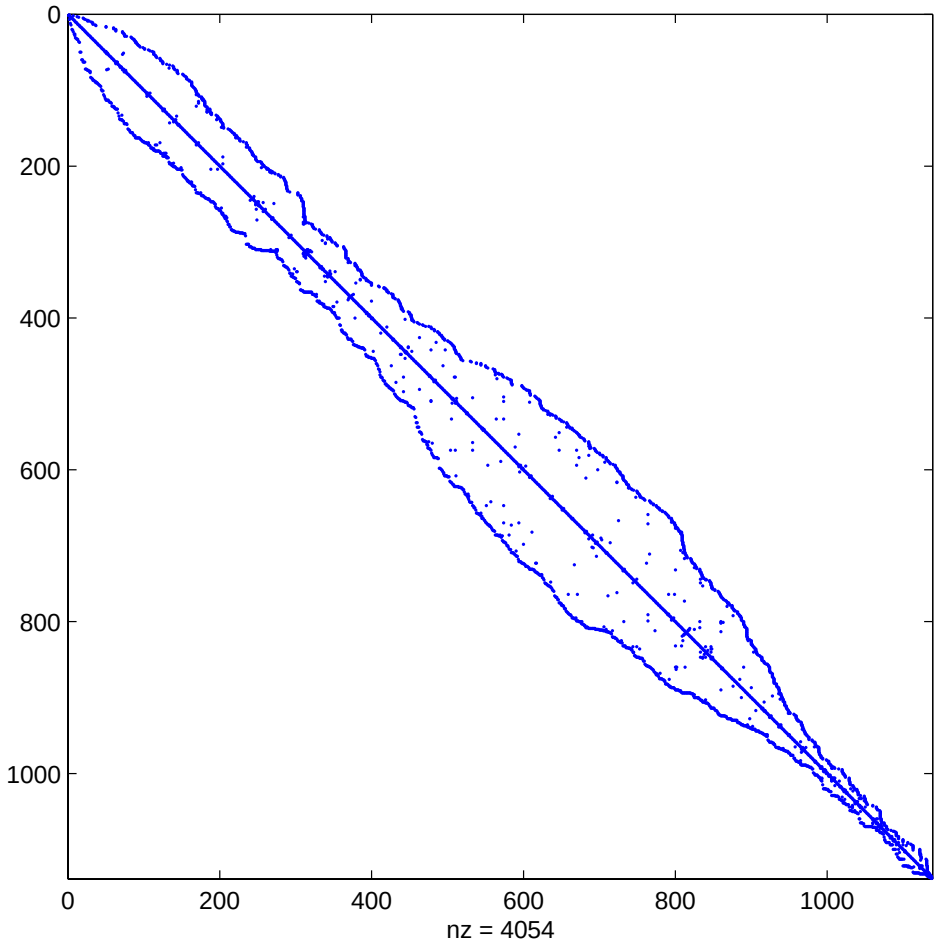
$$(r_i^{(k)} - 1)(c_j^{(k)} - 1)$$



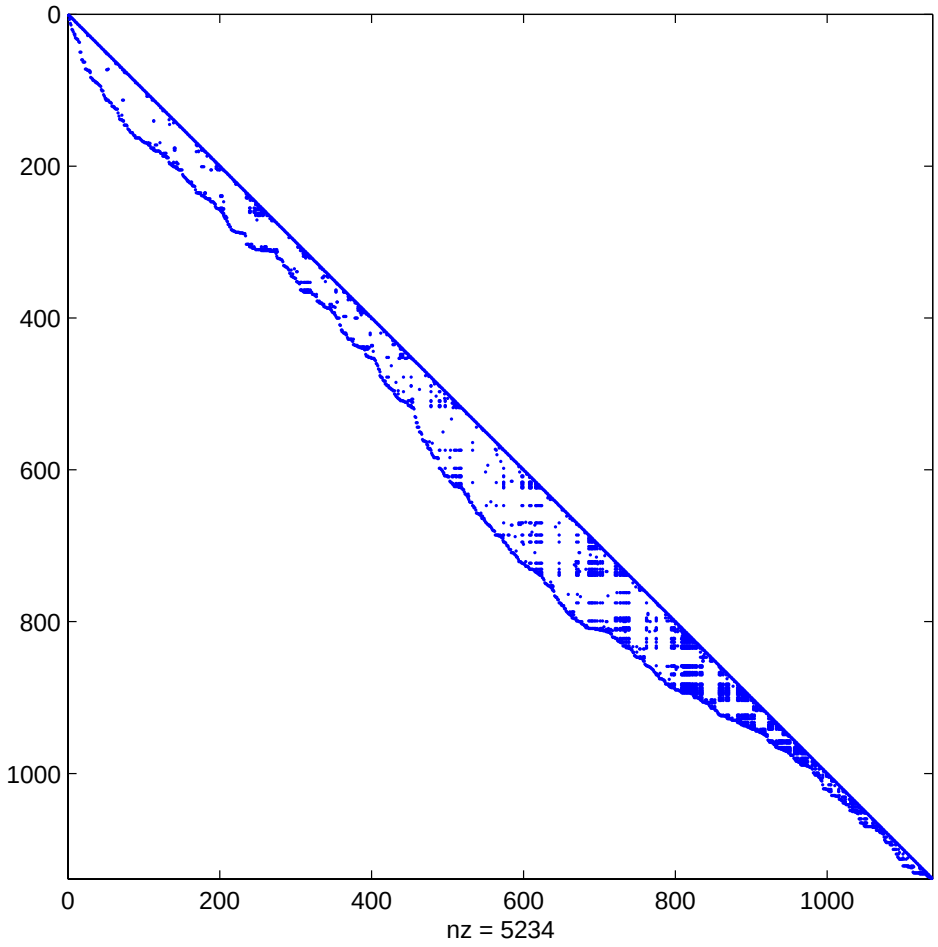
Matrice 1138-bus



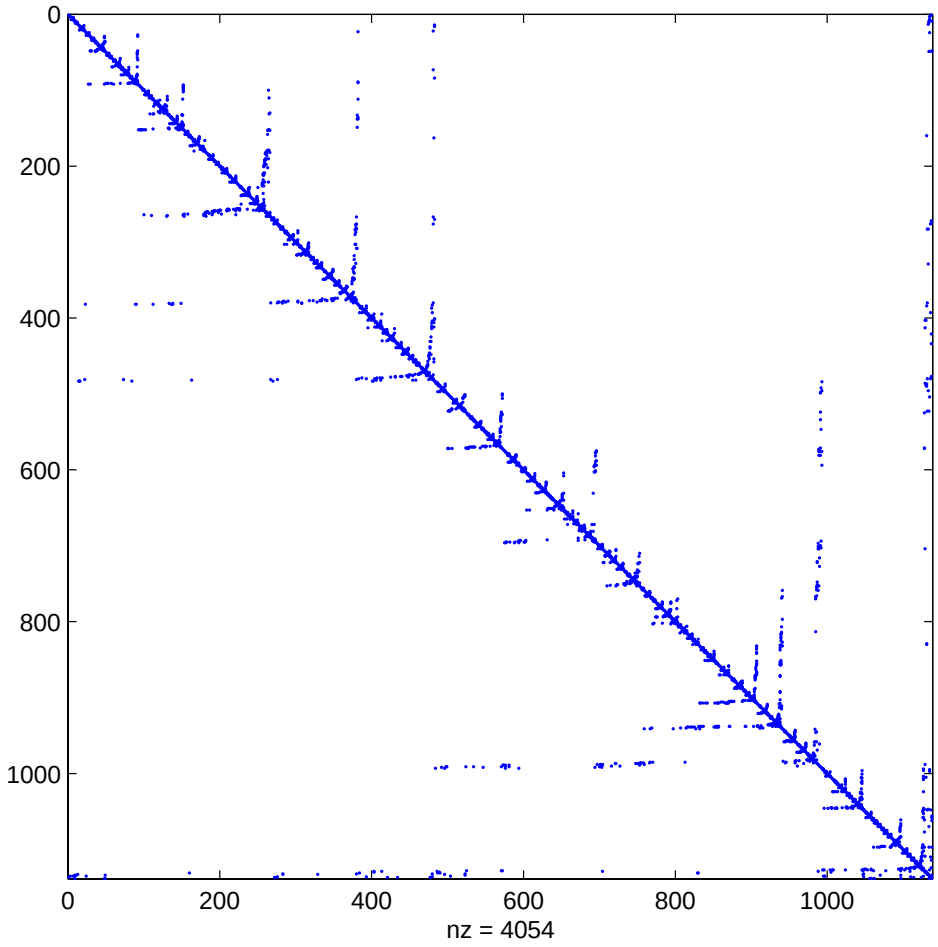
Matrice 1138-bus, facteur L, nz=38312



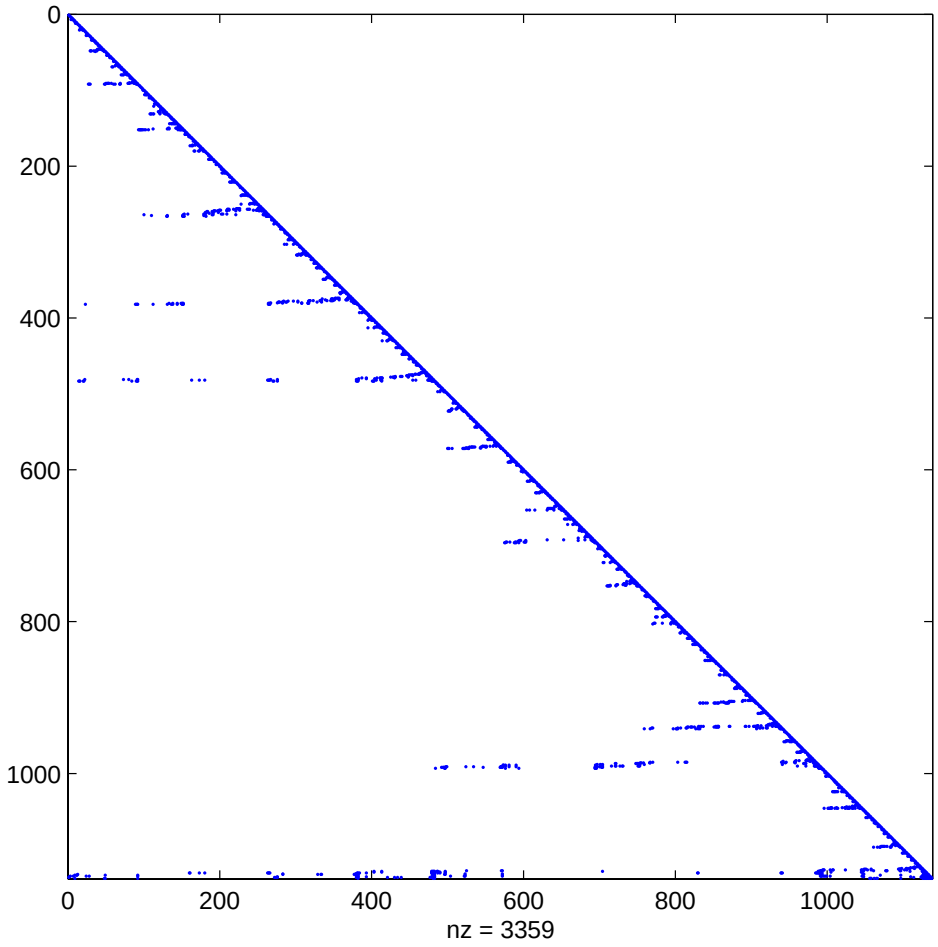
Matrice 1138-bus, RCM



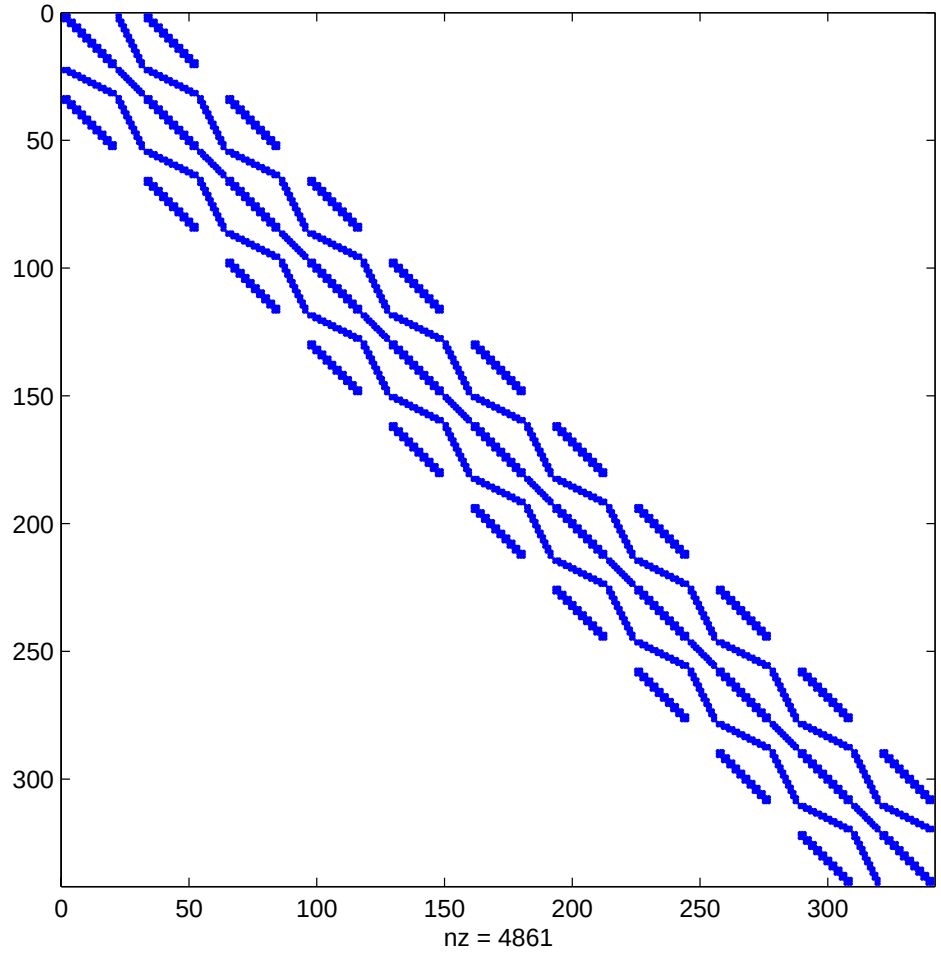
Matrice 1138-bus, RCM facteur L, nz=5234



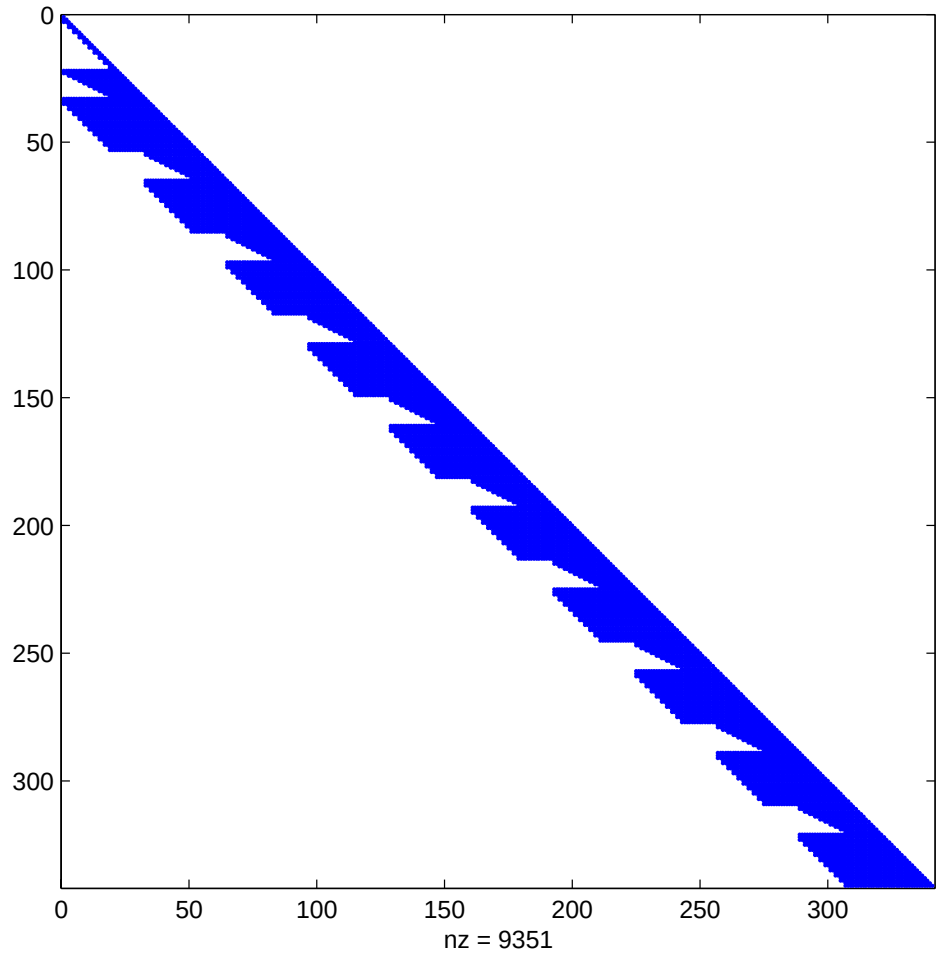
Matrice 1138-bus, MD



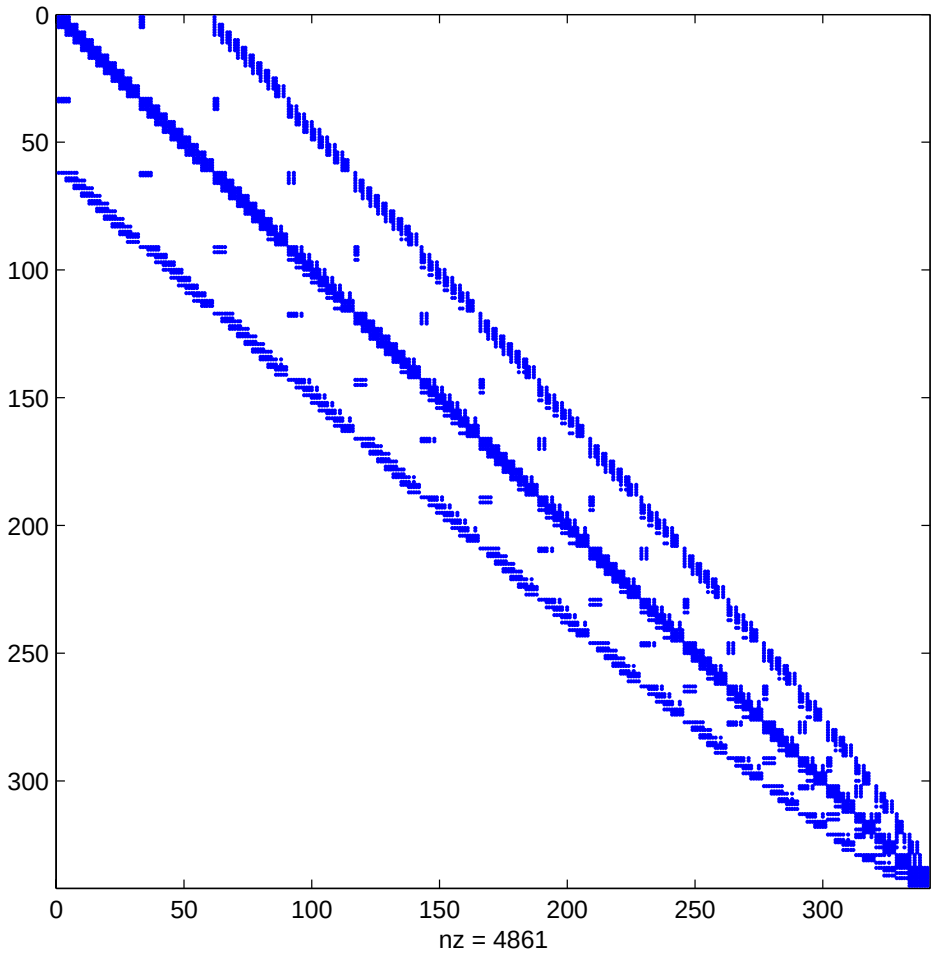
Matrice 1138-bus, MD facteur L, nz=3359



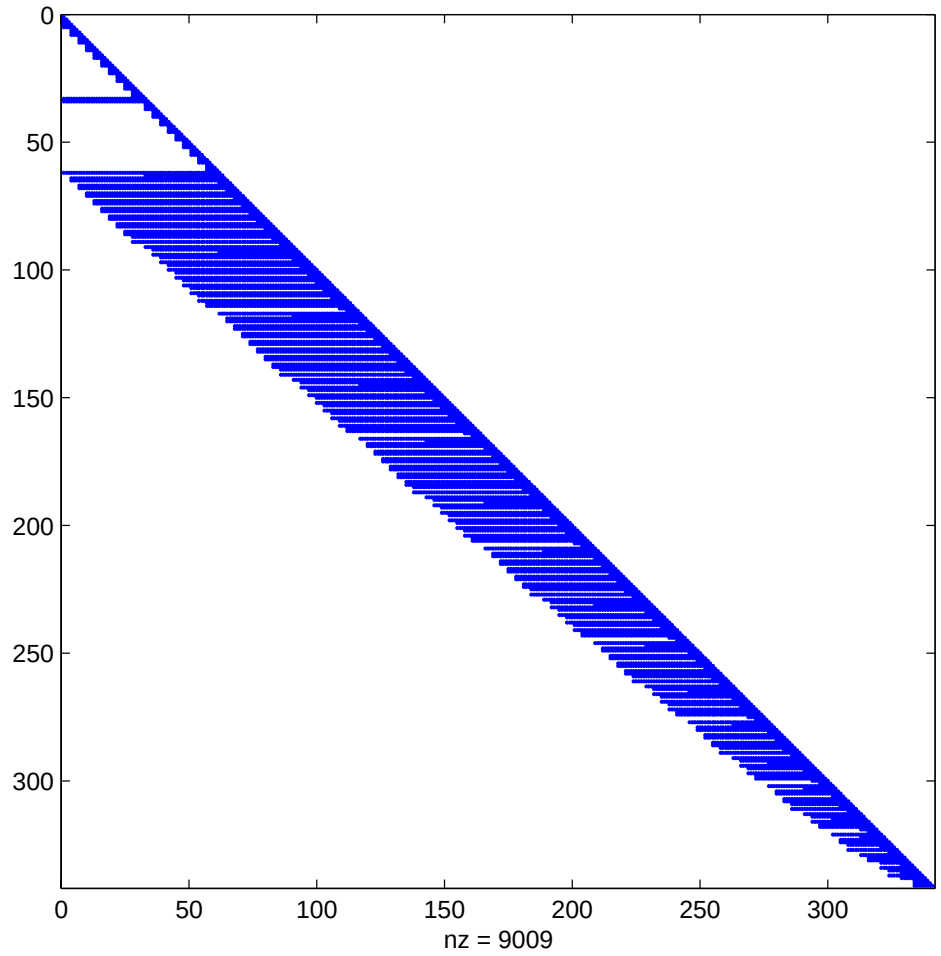
Matrice Wathen (Eléments finis)



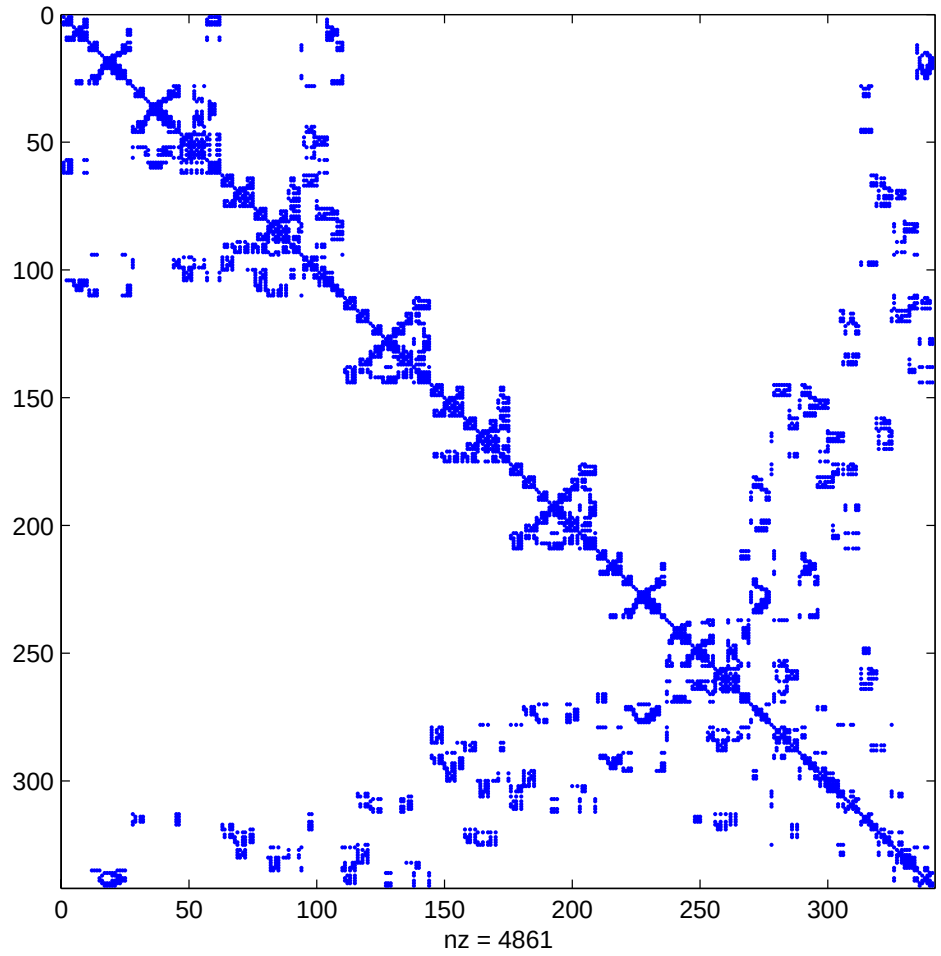
Matrice Wathen, facteur L, nz=9351



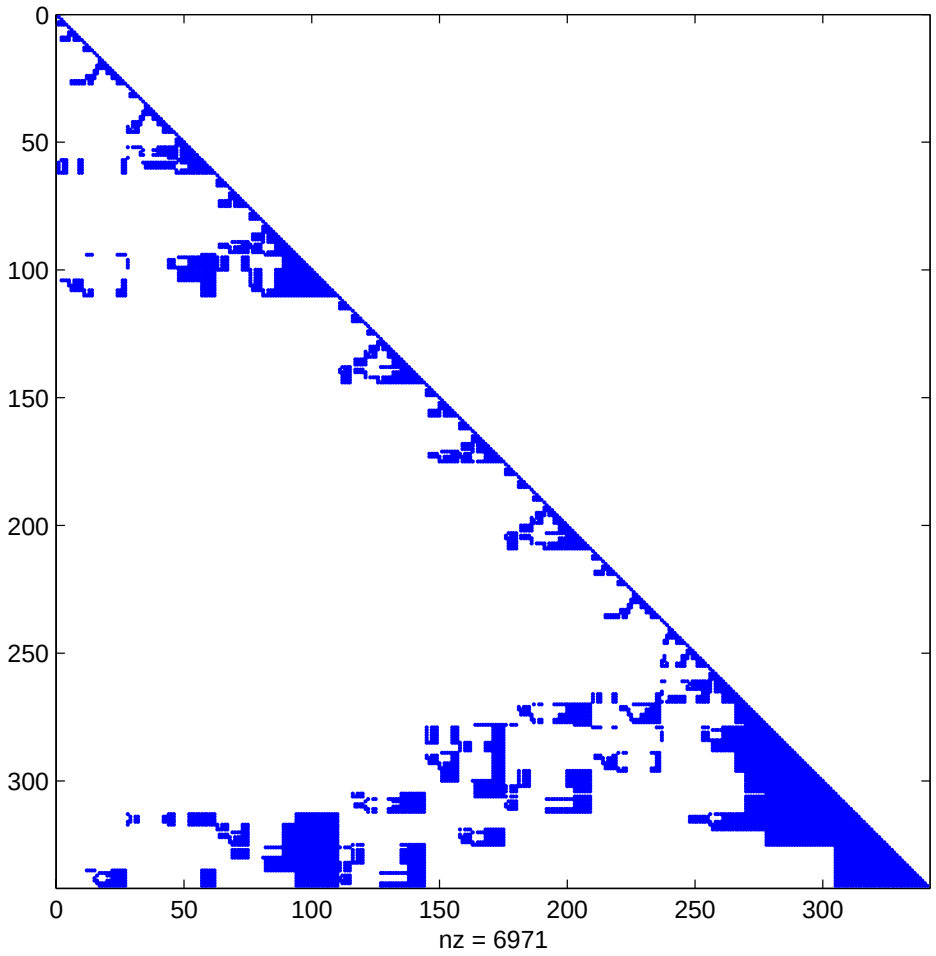
Matrice Wathen, RCM



Matrice Wathen, RCM facteur L, nz=9009



Matrice Wathen, MD



Matrice Wathen, MD facteur L, nz=6971

Autre problème : Parallélisme

- Le calcul de L et U et la résolution ne sont pas naturellement parallèles
- Utilisation de renumérotation (dissection + MD)
- Parallélisme dans l'arbre d'élimination
- Parallélisme à deux niveaux multifrontal

Il est difficile d'écrire un code (parallèle) efficace pour matrices creuses

- Utilisation de logiciels disponibles (sur Internet)

• MUMPS (*http://www.enseeiht.fr/lima/apo/MUMPS*)

developpé par P. Amestoy et ses collègues

MUMPS est un code direct multifrontal parallèle (F90, basé sur MPI)

Résout des systèmes symétriques ou non symétriques, réels ou complexes (+ décomp incomplètes)

Voir aussi (*http://www.cerfacs.fr*)

• SuperLU (*http://crd.lbl.gov/ ∼xiaoye/SuperLU*)

développé par S. Li, J. Demmel et J. Gilbert (écrit en C)

Résout des systèmes généraux non symétriques

Il existe des versions pour calculateurs séquentiels, parallèles à mémoire partagée et à mémoire distribuée

• PASTIX (*http://dept-info.labri.u-bordeaux.fr/∼ramet/pastix/main.html*)

développé à l'université de Bordeaux

Résout des systèmes symétriques et non symétriques à structure symétrique (pivotage statique)

- PSPASES

(*http://www-users.cs.umn.edu/∼mjoshi/pspases*)

développé à University of Minnesota par A. Gupta, M. Joshi, G. Karypis et V. Kumar

Résout des systèmes symétriques définis positifs

• HSL (auparavant the Harwell Subroutine Library) (*http://www.cse.clrc.ac.uk/nag/hsl*)

développé par I.S. Duff et ses collaborateurs

HSL est un produit commercial mais peut être utilisé gratuitement par les universitaires

Contient de nombreuses routines pour résoudre différents types de systèmes (en particulier symétriques indéfinis)

J.H. WILKINSON

The algebraic eigenvalue problem, Oxford University Press, 1965

I.S. DUFF, A.M. ERISMAN & J.K. REID

Direct methods for sparse matrices, Oxford University Press, 1986

G.H. GOLUB & C. VAN LOAN

Matrix computations, Johns Hopkins University Press, 1989

N.J. HIGHAM

Accuracy and stability of numerical algorithms, SIAM, 1996 (second edition 200?)

G. MEURANT

Computer solution of large linear systems, North–Holland, 1999

Krylov methods

Gérard MEURANT
CEA/DIF
Roscoff 2005

June 7, 2005

Aleksei N. Krylov, 1863–1945

Initial residual $r^0 = b - Ax^0$

Krylov space $\mathcal{K}_k$ of order k based on A and r^0

$$span\{r^0, Ar^0, \ldots, A^{k-1}r^0\}$$

$$x^k \in x^0 + \mathcal{K}_k$$

If V_k is a matrix whose columns are basis vectors $v^j, j = 1, \ldots, k$ of the Krylov space

$$x^k = x^0 + V_k z^k$$

Two basic kinds of Krylov methods:

1) Orthogonal residual (OR) methods for which

$$(r^k)^T V_k = 0$$

2) Minimum residual (MR) methods minimizes the l_2 norm of the residual $r^k = b - Ax^k$

$$(r^k)^T A V_k = 0$$

or solving a least squares problem

Examples of OR methods are CG for SPD (symmetric positive definite) matrices and FOM (Full Orthogonal Method) for general matrices

Examples of MR methods are MINRES (MINimum RESidual) for symmetric indefinite matrices and GMRES (Generalized Minimum RESidual) for general matrices

There are strong relations between OR and MR methods

OR methods may break down for non SPD matrices

Generally for stability reasons one uses an orthogonal basis of the Krylov space: Arnoldi process (1951)

Basis vectors are computed recursively by Gram–Schmidt starting from $v^1 = r^0/\|r^0\|$

The algorithm for computing column j of V_k is

$$h_{i,j} = (Av^j, v^i), \quad i = 1, \ldots, j$$

$$\bar{v}^j = Av^j - \sum_{i=1}^{j} h_{i,j} v^i$$

$$h_{j+1,j} = \|\bar{v}^j\|$$

$$v^{j+1} = \frac{\bar{v}^j}{h_{j+1,j}}$$

Generally, for stability reasons one prefers using the modified Gram–Schmidt form of the algorithm

$$w^j = Av^j$$

and for $i = 1, \ldots, j$

$$h_{i,j} = (w^j, v^i), \quad w^j = w^j - h_{i,j} v^i$$

Both algorithms are the same in exact arithmetic, but MGS is more stable and less parallel

$$AV_k = V_k H_k + h_{k+1,k} v^{k+1} (e^k)^T$$

H_k is an upper Hessenberg matrix with elements $h_{i,j}$

$$AV_k = V_{k+1} \tilde{H}_k$$

with

$$\tilde{H}_k = \begin{pmatrix} H_k \\ h_{k+1,k} (e^k)^T \end{pmatrix}$$

We also have

$$V_k^T A V_k = H_k$$

GMRES (Saad and Schulz 1986) minimizes the l_2 norm of the residual vector $b - Ax^k$

$$\| \, \|r^0\| e^1 - \tilde{H}_k z^k \|$$

This is solved by incrementally computing a QR decomposition of $\tilde{H}_k$ using Givens rotations

In FOM we have to solve linear systems

$$H_k z^k = \|r^0\| e^1$$

This is also done using QR factorizations (see next slides)

Drawback: the storage grows with the iteration number since we have to store the basis vectors v^j

Solution: restart every m iterations $\rightarrow$ GMRES(m)

More details

$$x^k = x^0 + V_k z^k$$

$$V_k^T A V_k z^k = H_k z^k = V_k^T (r^0 - r^k) = V_k^T r^0$$

If $v^1 = r^0 / \|r^0\|$,

$$V_k^T r^0 = \begin{pmatrix} \|r^0\| \\ 0 \\ \vdots \\ 0 \end{pmatrix} = \|r^0\| e^1$$

This is FOM or Arnoldi's method

Other possibility: we minimize the norm of the residual

$$r^k = b - Ax^k$$

$$r^0 = \|r^0\| V_{k+1} e^1$$

$$\begin{aligned}
\|r^k\| &= \|b - Ax^k\| \\
&= \|r^0 - AV_k z^k\| \\
&= \| \, \|r^0\| V_{k+1} e^1 - V_{k+1} \tilde{H}_k z^k \| \\
&= \| \, \|r^0\| e^1 - \tilde{H}_k z^k \|
\end{aligned}$$

This is a least squares problem with a (small) Hessenberg matrix

QR decomposition of $\tilde{H}_k$

Use Givens rotations to zero subdiagonal elements

$$\begin{pmatrix} x & x & \cdots & & x \\ & \ddots & & & \vdots \\ & & \ddots & & \\ & & & x & x \\ & & & 0 & r \\ & & & 0 & h \end{pmatrix}$$

We multiply by a (orthogonal) rotation matrix

$$\begin{pmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & & c_j & -s_j \\ & & & s_j & c_j \end{pmatrix}$$

$$c_j = \frac{r}{\sqrt{r^2+h^2}}, \qquad s_j = -\frac{h}{\sqrt{r^2+h^2}}$$

$$Q_k \tilde{H}_k = R_k$$

$$r_{k+1,k+1} = c_k r - s_k h = (r^2+h^2)/\sqrt{r^2+h^2} = \sqrt{r^2+h^2} > 0$$

There are some relationships between the iterates of FOM x_F^k and those of GMRES x_G^k

Let s_k and c_k be the rotation on rows k and $k+1$

$$\frac{\|r_G^k\|}{\|r_G^{k-1}\|} = |s_k|$$

$$\|r_G^k\| = |c_k| \; \|r_F^k\|$$

$$\|r_G^k\| = \sqrt{1 - \frac{\|r_G^k\|}{\|r_G^{k-1}\|}} \|r_F^k\|$$

If s_k is small, GMRES reduces significantly the norm of the residual and the same is true for FOM

If FOM breaks down, then GMRES stagnates

If A is similar to a diagonal matrix ($A = X\Lambda X^{-1}$), then

$$\|r^k\| \leq \epsilon_k \|X\| \|X^{-1}\| \|r^0\|$$

where ϵ_k is the minimum of $\max_\lambda |p_k(\lambda)|$ over polynomials p_k of degree k that satisfy the constraint $p_k(0) = 1$ and λ is an eigenvalue of A

If the symmetric part of A is positive definite

$$\|r^k\| \leq \left(1 - \frac{\lambda_{min}(A_S)}{\sigma_{max}(A)}\right)^k \|r^0\|$$

where $\sigma_{max}(A) = \lambda_{max}(A^T A)$ is the largest singular value of A

If the matrix A is diagonalizable by a matrix X

$$\|r_G^{k+l}\| \leq \frac{F_k}{|c_k|}\kappa(X)\epsilon_l\|r_G^k\|$$

$$F_k = \frac{|\sigma_1^{(k)}|}{|\lambda_1|} \max_{\lambda_j \neq \lambda_1} \left| \frac{\lambda_j - \lambda_1}{\lambda_j - \sigma_1^{(k)}} \right|$$

where λ_j are the eigenvalues of A, $\sigma_1^{(k)}$ is the smallest eigenvalue of H_k and $\epsilon_l = \min_q \max_{\lambda_i \neq \lambda_1} |q(\lambda_i)|$, q being a polynomial of degree less than l and such that $q(0) = 1$

So far, there is no satisfying convergence theory for non normal matrices ($A^H A \neq AA^H$ or $A^T A \neq AA^T$ for the real case)

Extensions to GMRES

Flexible GMRES (Saad) Variable preconditioner

x^0 be given, $r^0 = b - Ax^0$

$$v^1 = \frac{r^0}{\|r^0\|}, \quad f = \|r^0\| e^1$$

for $k = 1, \ldots$

$$M_k z^k = v^k, \quad w = Az^k$$

for $i = 1, \ldots, k$

$$h_{i,k} = (w, v^i), \quad w = w - h_{i,k} v^i$$

end i

$$h_{k+1,k} = \|w\|, \quad v^{k+1} = \frac{w}{h_{k+1,k}}$$

Apply the rotations of iterations 1 to $k-1$ on $(h_{1,k} \dots h_{k+1,k})^T$

Compute the rotation $R_{k+1,k}$ to eliminate $h_{k+1,k}$, $f = R_{k+1,k} f$, solve the triangular system for y^k

$$x^k = x^0 + Z_k y^k$$

where $Z_k = [z^1 \cdots z^k]$

end k

For non symmetric problems:

minimization $\neq$ orthogonalization

short recurrences $\longrightarrow$ no orthogonality of residuals

Non symmetric Lanczos algorithm

two starting vectors v^1 and $\tilde{v}^1$ such that $(v^1, \tilde{v}^1) = 1$,
$v^{-1} = \tilde{v}^{-1} = 0$

for $j = 0, 1, \ldots$

$$z^k = Av^k - \delta_k v^k - \eta_k v^{k-1}$$

$$w^k = A^T \tilde{v}^k - \delta_k \tilde{v}^k - \tilde{\eta}_k \tilde{v}^{k-1}$$

$$\delta_k = (\tilde{v}^k, Av^k)$$

η_k and $\tilde{\eta}_k$ are chosen such that

$$\eta_{k+1}\tilde{\eta}_{k+1} = (z^k, w^k)$$

$$v^{k+1} = \frac{z^k}{\tilde{\eta}_{k+1}}$$

$$\tilde{v}^{k+1} = \frac{w^k}{\eta_{k+1}}$$

The vectors v^k and $\tilde{v}^k$ are bi–orthogonal

$$T_k = \begin{pmatrix} \delta_1 & \eta_2 & & & \\ \tilde{\eta}_2 & \delta_2 & \eta_3 & & \\ & \ddots & \ddots & \ddots & \\ & & \tilde{\eta}_{k-1} & \delta_{k-1} & \eta_k \\ & & & \tilde{\eta}_k & \delta_k \end{pmatrix}$$

$$V_k = [v^1 \ \cdots v^k], \quad \tilde{V}_k = [\tilde{v}^1 \cdots \tilde{v}^k]$$

$$AV_k - V_k T_k = \tilde{\eta}_{k+1} v^{k+1} (e^k)^T$$

$$A^T \tilde{V}_k - \tilde{V}_k T_k^T = \eta_{k+1} \tilde{v}^{k+1} (e^k)^T$$

This can also be written as

$$AV_k = V_{k+1}\bar{T}_k$$

$$\tilde{V}_k A V_k = T_k$$

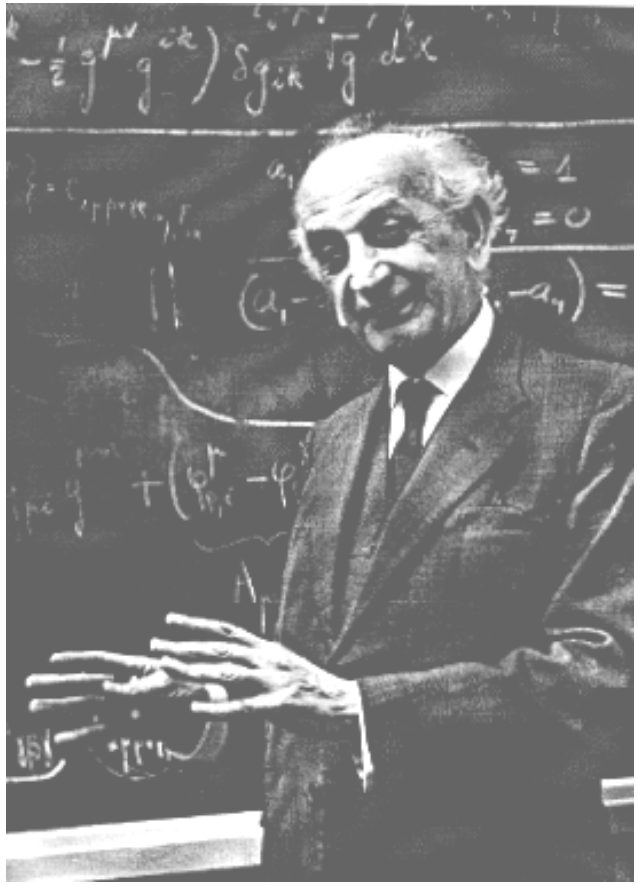
where

$$\bar{T}_k = \begin{pmatrix} T_k \\ \tilde{\eta}_{k+1}(e^k)^T \end{pmatrix}$$

is a $k+1 \times k$ matrix

$$(\tilde{v}^i, v^j) = 0, i \neq j, \quad i, j = 1, \ldots, k$$

The vectors $v^1, \ldots, v^k$ span $K_k(A, v^1)$ and $\tilde{v}^1, \ldots, \tilde{v}^k$ span $K_k(A^T, \tilde{v}^1)$



Cornelius Lanczos, 1893–1974

Construct bi–orthogonal sequences r^k and $\tilde{r}^k$ st

$$(\tilde{r}^k, r^l) = 0, \quad k \neq l$$

Biconjugate gradient (BiCG)

$$r^{k+1} = r^k - \alpha_k A p^k$$

$$\tilde{r}^{k+1} = \tilde{r}^k - \alpha_k A^T \tilde{p}^k$$

$$p^{k+1} = r^k + \beta_{k+1} p^k$$

$$\tilde{p}^{k+1} = \tilde{r}^k + \beta_{k+1} \tilde{p}^{k+1}$$

$$\alpha_k = \frac{(\tilde{r}^k, r^k)}{(\tilde{p}^k, A p^k)}, \quad \beta_{k+1} = \frac{(\tilde{r}^{k+1}, r^{k+1})}{(\tilde{r}^k, r^k)}$$

$$\Longrightarrow (\tilde{r}^k, r^l) = 0$$

$$(\tilde{p}^k, A p^l) = 0$$

Problems:

1) degeneracies $(\tilde{r}^k, r^k) = 0$

Remedies: Look Ahead algorithms (Freund, Brezinski)

2) erratic behavior of norm residuals

Evolution: CGS

Improvements: BiCGSTAB, BiCGSTAB(l) (Van Der Vorst and co)

Let ϕ_k and θ_k be polynomials of degree less or equal to k defined by

$$\phi_{k+1}(A) = \phi_k(A) - \alpha_k A \theta_k(A)$$

$$\theta_{k+1}(A) = \phi_{k+1}(A) + \beta_{k+1} \theta_k(A)$$

Then the vectors defined in BiCG satisfy

$$r^k = \phi_k(A)r^0, \quad p^k = \theta_k(A)r^0$$

$$\tilde{r}^k = \phi_k(A^T)r^0, \quad \tilde{p}^k = \theta_k(A^T)r^0$$

$$(r^k, \tilde{r}^k) = (\phi_k(A)r^0, \phi_k(A^T)r^0) = (\phi_k(A)^2 r^0, r^0)$$

$$\phi_{k+1}^2(t) = \phi_k^2(t) - 2\alpha_k t\theta_k(t)\phi_k(t) + \alpha_k^2 t^2 \theta_k^2(t)$$

$$\phi_k(t)\theta_k(t) = \phi_k^2(t) + \beta_k \phi_k(t)\theta_{k-1}(t)$$

The CGS algorithm (Sonneveld):

Let x^0 be given, $r^0 = b - Ax^0$, $q^0 = p^0 = r^0$, for $k = 0, 1, \ldots$

$$\alpha_k = \frac{(r^k, r^0)}{(Aq^k, r^0)}$$

$$u^k = p^k - \alpha_k A q^k$$

$$x^{k+1} = x^k + \alpha_k (p^k + u^k)$$

$$r^{k+1} = r^k - \alpha_k A (p^k + u^k)$$

$$\beta_{k+1} = \frac{(r^{k+1}, r^0)}{(r^k, r^0)}$$

$$p^{k+1} = r^{k+1} + \beta_{k+1} u^k$$

$$q^{k+1} = p^{k+1} + \beta_{k+1} (u^k + \beta_{k+1} q^k)$$

For the BiCG method

$$p^k = r^{k-1} - \beta_k p^{k-1}, \quad x^{k+1} = x^k + \alpha_k p^k, \quad r^{k+1} = r^k - \alpha_k A p^k$$

with r^k and Ap^k being orthogonal to $K_k(A^T, \tilde{r}^0)$

For any polynomial ψ_k of degree k and leading coefficient θ_k,

$$\beta_k = \frac{\theta_{k-1}}{\theta_k} \frac{\rho_k}{\sigma_{k-1}}, \quad \alpha_k = \frac{\rho_k}{\sigma_k}$$

where

$$\rho_k = (r^k, \psi_k(A^T)\tilde{r}^0) = (y^k, \tilde{r}^0)$$

with $y^k = \psi_k(A) r^k$

$$\sigma_k = (Ap^k, \psi_k(A^T)\tilde{r}^0)$$

We are free to choose the polynomial ψ_k as long as

$$\psi_0(A^T)\tilde{r}^0, \ldots, \psi_{k-1}(A^T)\tilde{r}^0$$

span the Krylov space $K_k(A^T, \tilde{r}^0)$

H. Van der Vorst looked for residuals of the form

$$r^k = \psi_k(A)\phi_k(A)r^0$$

where ϕ is the BiCG polynomial

For ψ he chose a product of degree one minimal residual polynomials in order to smooth the residuals

$$\psi_{k+1}(t) = (1 - \omega_k t) \cdots (1 - \omega_1 t)$$

and the polynomial ψ_k can be computed by the simple recurrence

$$\psi_{k+1}(t) = (1 - \omega_k t)\psi_k(t)$$

There are recurrences for $\psi_k \phi_k$ and $\psi_k \theta_k$

$$r^k = \psi_k(A)\phi_k(A)r^0$$

$$p^k = \psi_k(A)\theta_k(A)r^0$$

Parameters ω_k are chosen as to minimize the Euclidean norm of the residual

BiCGSTAB algorithm

Let x^0 be given, $r^0 = b - Ax^0$, $p^0 = r^0$, $\tilde{r}^0$ arbitrary for $k = 0, 1, \ldots$

$$\alpha_k = \frac{(r^k, \tilde{r}^0)}{(Ap^k, \tilde{r}^0)}$$

$$s^k = r^k - \alpha_k A p^k$$

$$\omega_k = \frac{(As^k, s^k)}{(As^k, As^k)}$$

$$x^{k+1} = x^k + \alpha_k p^k + \omega_k s^k$$

$$r^{k+1} = s^k - \omega_k A s^k$$

$$\beta_{k+1} = \frac{(r^{k+1}, \tilde{r}^0)}{(r^k, \tilde{r}^0)} \frac{\alpha_k}{\omega_k}$$

$$p^{k+1} = r^{k+1} + \beta_{k+1}(p^k - \omega_k A p^k)$$

Extension:

BiCGSTAB(l) algorithm

Let $k = -l$, x^0 and $\tilde{r}^0$ be given, $r^0 = b - Ax^0$, $u^{-1} = 0$, $\alpha = 0$, $\omega = 1$.

Then we repeat until convergence

$$k = k + l$$

$$\hat{u}^0 = u^{k-1}, \quad \hat{r}^0 = r^k, \quad \hat{x}^0 = x^k$$

$$\rho_0 = -\omega\rho_0$$

$$\text{for } j = 0, \ldots, l-1$$

$$\rho_1 = (\hat{r}^j, \tilde{r}^0), \quad \beta = \beta_{k+j} = \alpha\frac{\rho_1}{\rho_0}, \quad \rho_0 = \rho_1$$

$$\hat{u}^i = \hat{r}^i - \beta\hat{u}^i, i = 0, \ldots, j$$

$$\hat{u}^{j+1} = A\hat{u}^j$$

$$\gamma = (\hat{u}^{j+1}, \tilde{r}^0), \qquad \alpha = \alpha_{k+j} = \frac{\rho_0}{\gamma}$$

$$\hat{r}^i = \hat{r}^i - \alpha\hat{u}^{i+1}, i = 0, \ldots, j$$

$$\hat{r}^{j+1} = A\hat{r}^j, \qquad \hat{x}^0 = \hat{x}^0 + \alpha\hat{u}^0$$

end j

for $j = 1, \ldots, l$

$$\tau_{i,j} = \frac{(\hat{r}^j, \hat{r}^i)}{\sigma_i}, \quad \hat{r}^j = \hat{r}^j - \tau_{i,j}\hat{r}^i, i = 1, \ldots, j-1$$

$$\sigma_j = (\hat{r}^j, \hat{r}^j), \quad \gamma'_j = \frac{(\hat{r}^0, \hat{r}^j)}{\sigma_j}$$

end j

$$\gamma_l = \gamma'_l, \quad \omega = \gamma_l$$

$$\gamma_j = \gamma'_j - \sum_{i=j+1}^{l} \tau_{i,j}\gamma_i, j = l-1, \ldots, 1$$

$$\tilde{\gamma}_j = \gamma_{j+1} - \sum_{i=j+1}^{l-1} \tau_{i,j}\gamma_{i+1}, j = 1, \ldots, l-1$$

$$\hat{x}^0 = \hat{x}^0 + \gamma_1 \hat{r}^0, \quad \hat{r}^0 = \hat{r}^0 - \gamma_1' \hat{r}^l, \quad \hat{u}^0 = \hat{u}^0 - \gamma_l \hat{u}^l$$

for $j = 1, \ldots, l-1$

$$\hat{u}^0 = \hat{u}^0 - \gamma_j \hat{u}^j$$

$$\hat{x}^0 = \hat{x}^0 + \tilde{\gamma}_j \hat{r}^j$$

$$\hat{r}^0 = \hat{r}^0 - \gamma_j' \hat{r}^j$$

end j

$$u^{k+l-1} = \hat{u}^0, \quad r^{k+l} = \hat{r}^0, \quad x^{k+l} = \hat{x}^0$$

Quasi Minimal Residual (QMR)

Use the non symmetric Lanczos algorithm

$$AV_k = V_{k+1}\bar{T}_k$$

$$\|r^k\| = \|b - Ax^k\|$$

$$= \|b - AV_k y^k\|$$

$$= \| \, \|b\| V_{k+1} e^1 - V_{k+1}\bar{T}_k y^k \|$$

Unfortunately V_{k+1} is not orthogonal

QMR minimizes

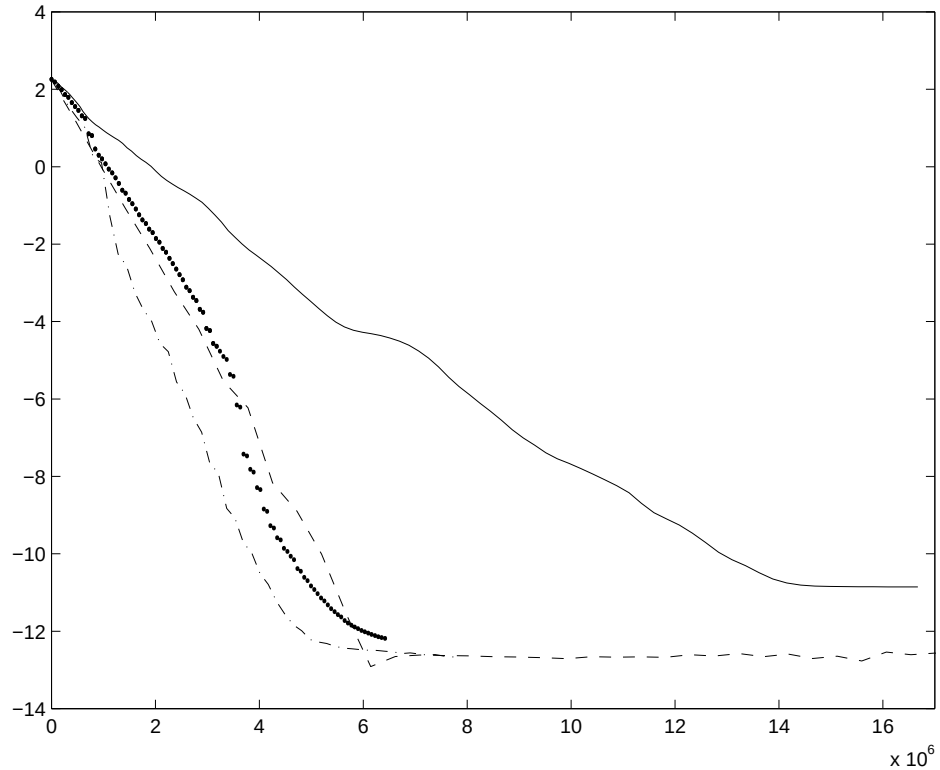
$$\| \, \|b\| e^1 - \bar{T}_k y \|$$

As a numerical example for nonsymmetric problems, we solve a convection–diffusion equation

$$-\Delta u + 2P\frac{\partial u}{\partial x} = f$$

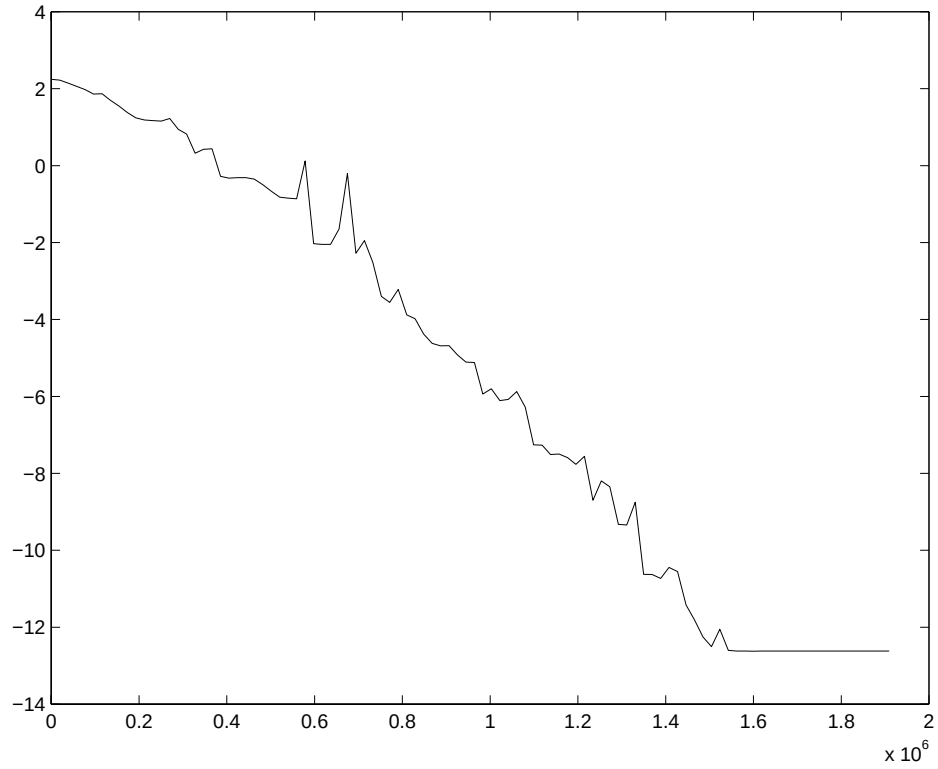
in the unit square with Dirichlet boundary conditions where $P = e^{2(x^2+y^2)}$

We use an upwind scheme for the first order derivative



Convection–diffusion problem, GMRES(m)

nb of flops, GMRES (solid line), GMRES(20) (dashed line), GMRES(10) (dot–dashed line) and GMRES(5) (dotted line)



Convection–diffusion problem, BiCGstab

nb of flops

B. FISCHER

Polynomial based iteration methods for symmetric linear systems, Wiley Teubner, 1996

A. GREENBAUM

Iterative methods for solving linear equations, SIAM, 1997

G. MEURANT

Computer solution of large linear systems, North–Holland, 1999

Y. SAAD

Iterative methods for sparse linear systems, PWS Publishing Company, 1996, second edition SIAM

H.A VAN DER VORST

Iterative Krylov methods for large linear systems, Cambridge University Press, 2003

The Conjugate Gradient method

Gérard MEURANT
CEA/DIF
Roscoff 2005

June 4, 2005

CG was developed independently by Magnus Hestenes and Eduard Stiefel in the US and in Switzerland at the beginning of the fifties. Their famous joint paper was published in 1952

Magnus Hestenes, 1906–1991

Eduard Stiefel, 1909–1978

Suppose A is symmetric positive definite (SPD)

Lanczos $\equiv$ symmetric Arnoldi

$$AV_k = V_k T_k + G_k, \quad G_k = \begin{pmatrix} 0 & \eta_{k+1} v^{k+1} \end{pmatrix}$$

$$x^k = x^0 + V_k y^k, \quad T_k y^k = \|r^0\| e^1$$

Using the LU decomposition of T_k and change of variables one obtains the Conjugate Gradient algorithm

CG algorithm

x^0 given and $r^0 = b - Ax^0$:

for $k = 0, 1, \ldots$ until convergence do,

$$\beta_k = \frac{(r^k, r^k)}{(r^{k-1}, r^{k-1})}, \ \beta_0 = 0$$

$$p^k = r^k + \beta_k p^{k-1}$$

$$\gamma_k = \frac{(r^k, r^k)}{(Ap^k, p^k)}$$

$$x^{k+1} = x^k + \gamma_k p^k$$

$$r^{k+1} = r^k - \gamma_k A p^k$$

Relations with Lanczos coefficients

$$\alpha_k = \frac{1}{\gamma_{k-1}} + \frac{\beta_{k-1}}{\gamma_{k-2}}, \qquad \beta_0 = 0, \qquad \gamma_{-1} = 1$$

$$\eta_{k+1} = \frac{\sqrt{\beta_k}}{\gamma_{k-1}}$$

Properties

$$(r^i, r^j) = 0, \qquad i \neq j$$

$$(p^i, Ap^j) = 0, \qquad i \neq j$$

The A–norm of the error

$$A\epsilon^k = r^k$$

Therefore,

$$\|\epsilon^k\|_A^2 = (A\epsilon^k, \epsilon^k) = (A^{-1}r^k, r^k) = (r^k)^T A^{-1} r^k$$

The Lanczos algorithm is very tightly linked to Gauss quadrature

$$A = Q\Lambda Q^T$$

Consider

$$u^T f(A) u, \quad f(A) = Q f(\Lambda) Q^T$$

$$I[f] = u^T f(A) u = \int_a^b f(\lambda)\ d\alpha(\lambda)$$

where the measure α is piecewise constant if $y = Q^T u$

$$\alpha(\lambda) = \begin{cases} 0 & \text{if } \lambda < a = \lambda_1 \\ \sum_{j=1}^{i} y_j^2 & \text{if } \lambda_i \leq \lambda < \lambda_{i+1} \\ \sum_{j=1}^{n} y_j^2 & \text{if } b = \lambda_n \leq \lambda \end{cases}$$

We can use Gauss, Gauss–Radau and Gauss–Lobatto to obtain bounds

$$\int_a^b f(\lambda)\ d\alpha(\lambda) = \sum_{j=1}^{N} w_j f(t_j) + \sum_{k=1}^{M} v_k f(z_k) + R[f]$$

$[w_j]_{j=1}^N, [v_k]_{k=1}^M$ and $[t_j]_{j=1}^N$ are unknowns and $[z_k]_{k=1}^M$ are prescribed

$$R[f] = \frac{f^{(2N+M)}(\eta)}{(2N+M)!} \int_a^b \prod_{k=1}^{M} (\lambda - z_k) \left[\prod_{j=1}^{N} (\lambda - t_j) \right]^2 d\alpha(\lambda), \quad a < \eta < b$$

$M = 0$, Gauss rule

$M = 1$ and either $z_1 = a$ or $z_1 = b$, Gauss–Radau

$M = 2$ and $z_1 = a, z_2 = b$, Gauss–Lobatto

The polynomials $p_1(\lambda), p_2(\lambda), \ldots$ that are orthonormal with respect to α are the Lanczos polynomials

$$(v^k, v^l) = \int_a^b p_k(\lambda) p_l(\lambda_l)\, d\alpha(\lambda)$$

$$\lambda p(\lambda) = T_k p(\lambda) + \eta_{k+1} p_{k+1}(\lambda) e^k$$

where

$$p(\lambda)^T = [p_1(\lambda)\ p_2(\lambda) \cdots\ p_k(\lambda)]$$

For Gauss, the nodes t_j are the Ritz values $\theta_j^{(k)}$ and the weights are the squares of the first elements of the normalized eigenvectors

For $f(x) = 1/x$, Gauss gives a lower bound of the integral

$$\sum_{l=1}^{k} w_l f(t_l) = (e^1)^T f(T_k) e^1$$

To obtain Gauss–Radau and Gauss–Lobatto we should modify the matrix T_k to get the prescribed eigenvalues

For CG we have

$$\|\epsilon^k\|_A^2 = \|r^0\|^2 [(T_n^{-1} e^1, e^1) - (T_k^{-1} e^1, e^1)]$$

and

$$\|\epsilon^k\|_A^2 = \|r^0\|^2 \left[\sum_{j=1}^{n} \frac{[(z_{(n)}^j)_1]^2}{\lambda_j} - \sum_{j=1}^{k} \frac{[(z_{(k)}^j)_1]^2}{\theta_j^{(k)}} \right]$$

where $z_{(k)}^j$ is the jth eigenvector of T_k

Other formulas can be obtained

$$\|\epsilon^k\|_A^2 = \|r^k\|^2[\gamma_k + \beta_{k+1}\gamma_{k+1} + \beta_{k+1}\beta_{k+2}\gamma_{k+2} + \cdots + \beta_{k+1}\cdots\beta_{n-1}\gamma_{n-1}]$$

$$\|\epsilon^k\|_A^2 = \sum_{j=k}^{n-1} \gamma_j \|r^j\|^2$$

$$\|\epsilon^k\|_A^2 = -\eta_{k+1}\|r^0\|^2(e^k, T_k^{-1}e^1)(T_n^{-1}e^1, f^{k+1})$$

where f^{k+1} is a vector whose all components are zero except for the $k+1$ st one which is 1

For the l_2 norm of the error we have

$$\|\epsilon^k\|^2 = \|r^0\|^2[(e^1, T_n^{-2}e^1) - (e^1, T_k^{-2}e^1)] +$$

$$+(-1)^k 2\eta_{k+1}\frac{\|r^0\|}{\|r^k\|}(e^k, T_k^{-2}e^1)\|\epsilon^k\|_A^2$$

and

$$\|\epsilon^k\|^2 = \|r^0\|^2[(e^1, T_n^{-2}e^1) - (e^1, T_k^{-2}e^1)] - 2\frac{(e^k, T_k^{-2}e^1)}{(e^k, T_k^{-1}e^1)}\|\epsilon^k\|_A^2$$

$$\|r^k\|^2 = \sum_{j=1}^{n} \prod_{i=1}^{k} \left(1 - \frac{\lambda_j}{\theta_i^{(k)}}\right)^2 (\bar{r}_j^0)^2$$

$$\|\epsilon^k\|^2 = \sum_{j=1}^{n} \prod_{i=1}^{k} \left(\frac{1}{\lambda_j} - \frac{1}{\theta_i^{(k)}}\right)^2 (\bar{r}_j^0)^2$$

$$\|\epsilon^k\|_A^2 = \sum_{j=1}^{n} \prod_{i=1}^{k} \left(\frac{1}{\sqrt{\lambda_j}} - \frac{\sqrt{\lambda_j}}{\theta_i^{(k)}}\right)^2 (\bar{r}_j^0)^2$$

Optimality of CG

Consider all the iterative methods that can be written as

$$\overline{x}^{k+1} = \overline{x}^0 + Q_k(A)\overline{r}^0, \quad \overline{x}^0 = x^0, \quad \overline{r}^0 = b - A\overline{x}^0$$

where Q_k is a kth degree polynomial

Of all these methods, CG is the one which minimizes $\|\epsilon^k\|_A$ at each iteration

$$\|\epsilon^k\|_A^2 \leq \max_{1 \leq i \leq n} (R_k(\lambda_i))^2 \|\epsilon^0\|_A^2$$

for all polynomials R_k of degree k such that $R_k(0) = 1$

$$\|\epsilon^k\|_A^2 \leq 4\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^{2k}\|\epsilon^0\|_A^2$$

where $\kappa = \frac{\lambda_n}{\lambda_1}$

Choose

$$R_k(\lambda) = \frac{C_k(\frac{\lambda_1+\lambda_n-2\lambda}{\lambda_n-\lambda_1})}{C_k(\frac{\lambda_1+\lambda_n}{\lambda_n-\lambda_1})}$$

Bounds for the A–norm of the error

$$\|\epsilon^{k-d}\|_A^2 \simeq \|r^0\|^2((T_k^{-1})_{(1,1)} - (T_{k-d}^{-1})_{(1,1)})$$

$$s_{k-d} = \|r^0\|^2(b_k - b_{k-d})$$

$$(T_{k+1}^{-1})_{1,1} = (T_k^{-1})_{1,1} + \frac{\eta_{k+1}^2 (t_k t_k^T)_{1,1}}{\alpha_{k+1} - \eta_{k+1}^2 (t_k)_k}, \qquad t_k = T_k^{-1} e^k$$

$$(t_k)_1 = (-1)^{k-1} \frac{\eta_2 \cdots \eta_k}{\delta_1 \cdots \delta_k}, \qquad (t_k)_k = \frac{1}{\delta_k}$$

$$b_k = b_{k-1} + f_k, \quad f_k = \frac{\eta_k^2 c_{k-1}^2}{\delta_{k-1}(\alpha_k \delta_{k-1} - \eta_k^2)} = \frac{c_k^2}{\delta_k}$$

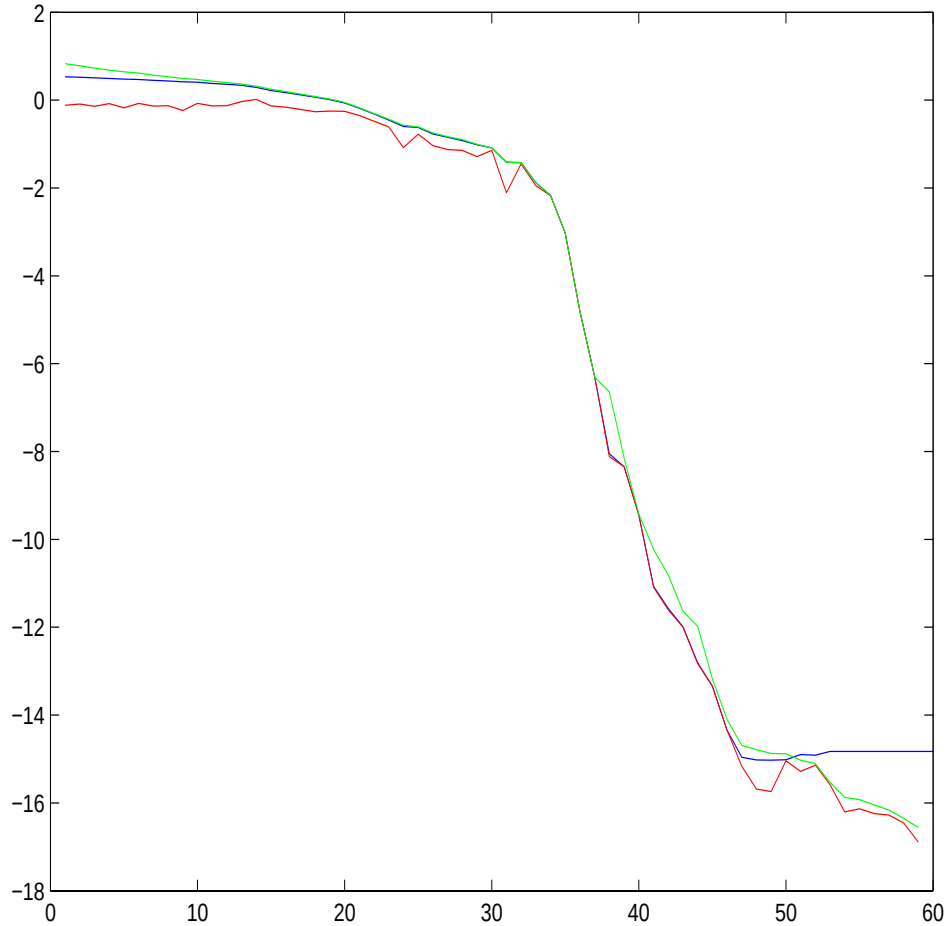
$$c_k = \frac{\eta_2 \cdots \eta_k}{\delta_1 \cdots \delta_{k-1}} = \frac{\|r^{k-1}\|}{\|r^0\|}, \quad \gamma_{k-1} = 1/\delta_k$$

This gives

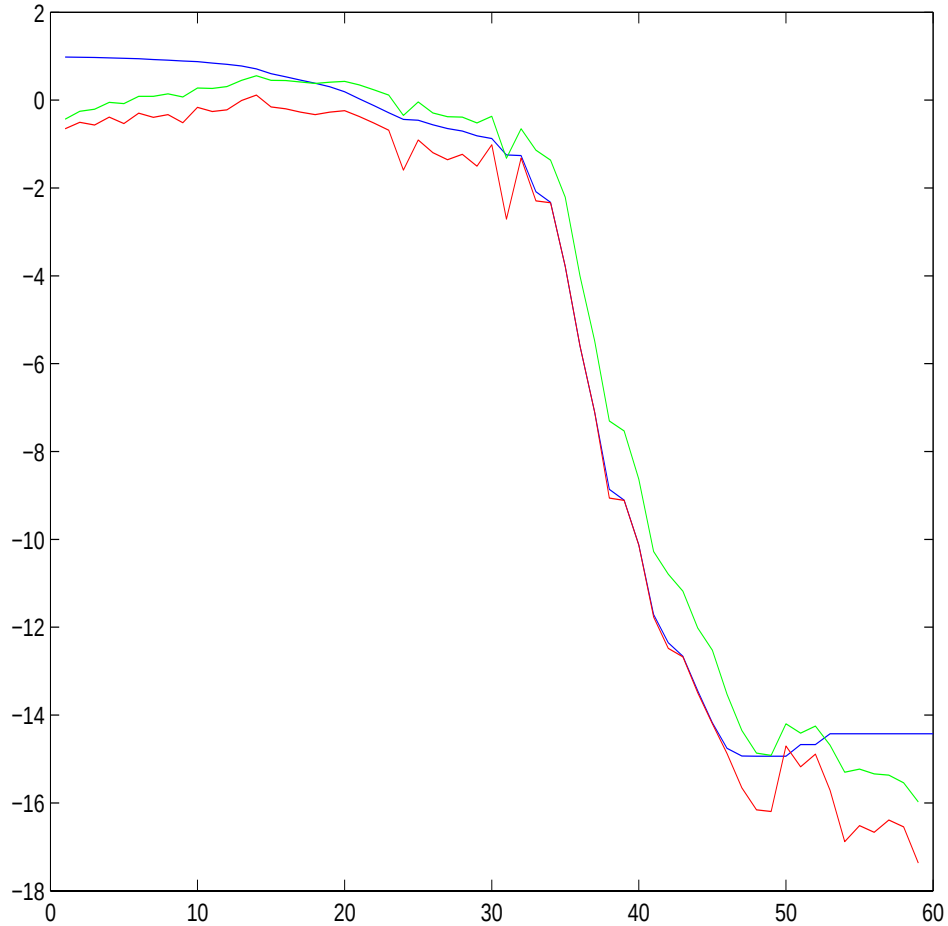
$$f_k = \gamma_{k-1} \|r^{k-1}\|^2 / \|r^0\|^2$$

So, it's enough to sum some f_k's

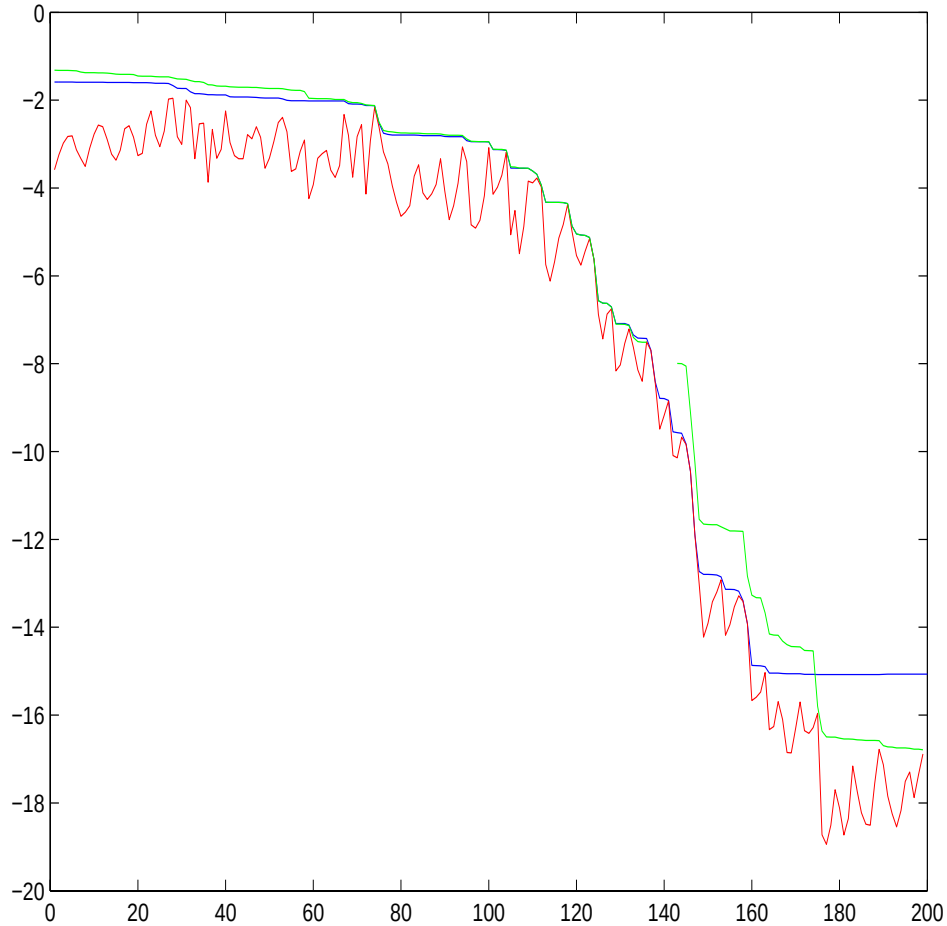
EXAMPLES



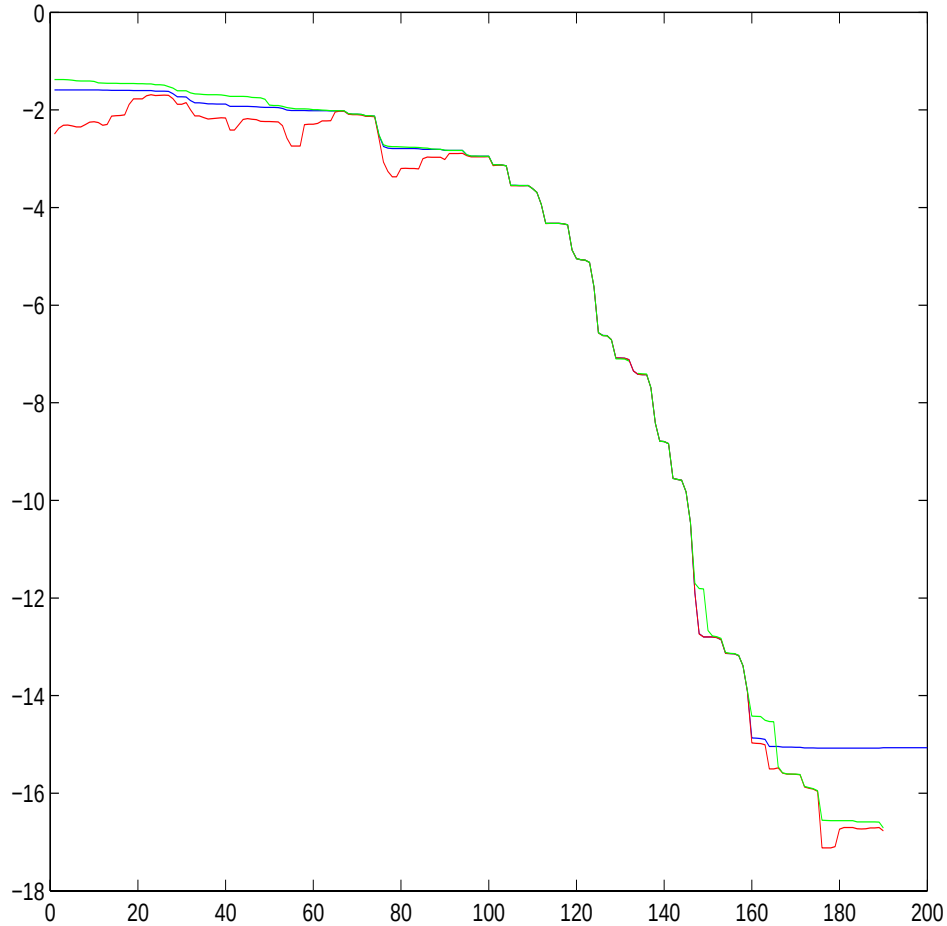
Strakos30: $\log_{10}$ of A–norm of the error (blue), Gauss estimate (red), Gauss–Radau estimate (green), $d = 1$



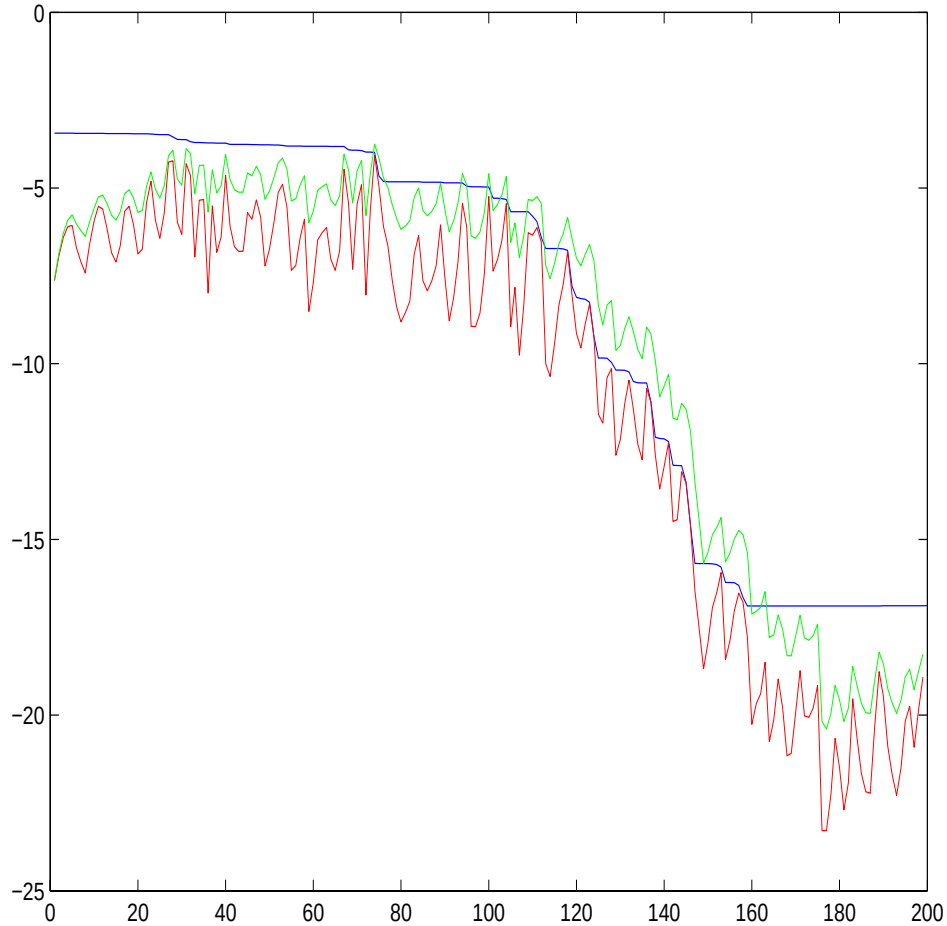
Strakos30: $\log_{10}$ of l_2 norm of the error (blue), lower estimate (red), upper estimate (green), $d = 1$



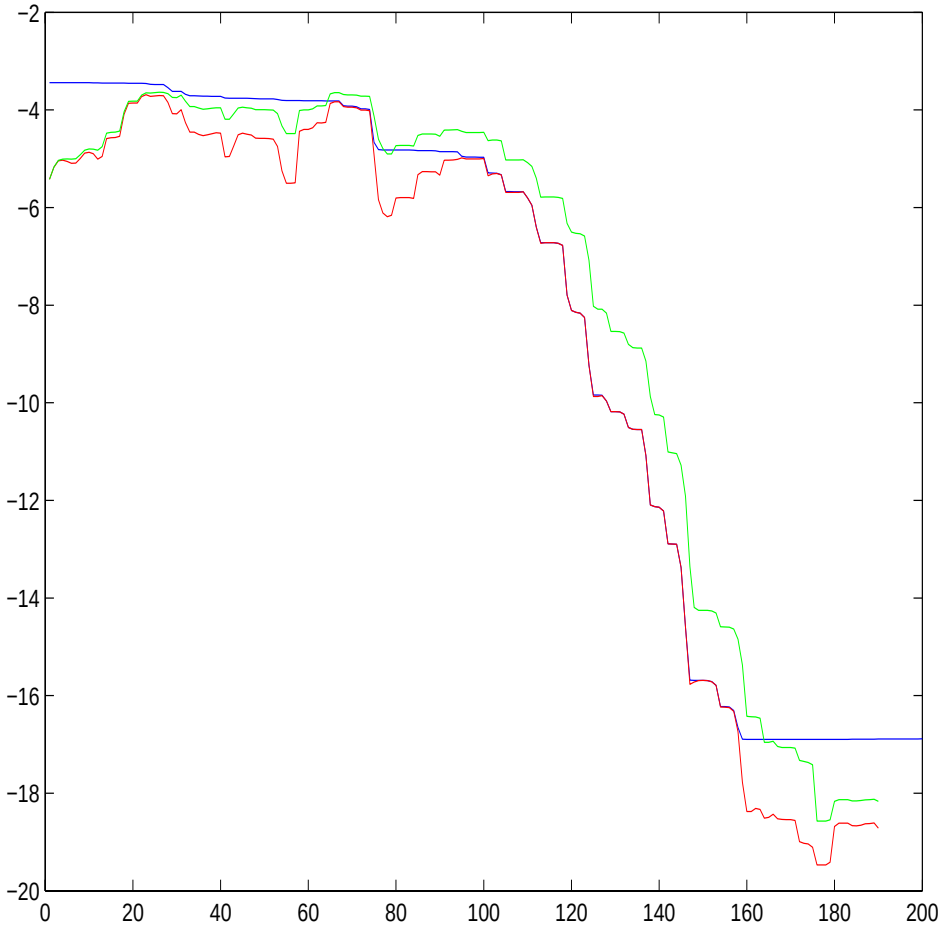
Bcsstk01: $\log_{10}$ of A–norm of the error (blue), Gauss estimate (red), Gauss–Radau estimate (green), $d = 1$



Bcsstk01: $\log_{10}$ of A–norm of the error (blue), Gauss estimate (red), Gauss–Radau estimate (green), $d = 10$



Bcsstk01: $\log_{10}$ of l_2 norm of the error (blue), lower estimate (red), upper estimate (green), $d = 1$



Bcsstk01: $\log_{10}$ of l_2 norm of the error (blue), lower estimate (red), upper estimate (green), $d = 10$

The Conjugate Residual method

(CR) is constructed by requiring the residuals to be A–orthogonal,

$$(r^i, Ar^j) = 0, i \neq j$$

The iterates x^k minimize the l_2–norm of the residual

$$\|b - Ax^k\| = \min\{\|b - Ay\|\}, \quad y \in x^0 + K_k(A, r^0)\}$$

$r^0 = b - Ax^0$

For $k = 0, 1, \ldots$

$$\beta_k = \frac{(r^k, Ar^k)}{(r^{k-1}, Ar^{k-1})}, \qquad \beta_0 = 0$$

$$p^k = r^k + \beta_k p^{k-1}$$

$$Ap^k = Ar^k + \beta_k Ap^{k-1}$$

$$\gamma_k = \frac{(r^k, Ar^k)}{(Ap^k, Ap^k)}$$

$$x^{k+1} = x^k + \gamma_k p^k$$

$$r^{k+1} = r^k - \gamma^k Ap^k$$

This algorithm is well defined if A is positive definite

Symmetric indefinite systems

When A is symmetric but not positive definite the Cholesky factorization may not exist

Paige and Saunders proposed using an LQ factorization of T_k with Q orthogonal

The CG solution is obtained by

$$T_k y^k = \eta_1 e^1, \quad x_{CG}^k = Q_k y^k$$

$$T_k = L_k Z_k, \quad Z_k^T Z_k = I$$

with L_k being lower triangular

The matrix Z_k is constructed as the product of of plane rotations

$$T_k = \begin{pmatrix} \delta_1 & \eta_2 & & & \\ \eta_2 & \delta_2 & \eta_3 & & \\ & \ddots & \ddots & \ddots & \\ & & \eta_{k-1} & \delta_{k-1} & \eta_k \\ & & & \eta_k & \delta_k \end{pmatrix}$$

Let us look at the first steps of the reduction considering T_4

$$\bar{T}_4 = \begin{pmatrix} \delta_1 & \eta_2 & & \\ \eta_2 & \delta_2 & \eta_3 & \\ & \eta_3 & \delta_3 & \eta_4 \\ & & \eta_4 & \delta_4 \end{pmatrix}$$

We zero the element in position $(1, 2)$

$$Z_{1,2} = \begin{pmatrix} c_1 & s_1 & & \\ s_1 & c_2 & & \\ & & 1 & \\ & & & 1 \end{pmatrix}$$

To annihilate the $(1,2)$ element, we must have: $s_1\delta_1 = c_1\eta_2$

Let $\gamma_1 = \sqrt{\delta_1^2 + \eta_2^2}$, we take $s_1 = \eta_2/\gamma_1$, $c_1 = \delta_1/\gamma_1$

$$\bar{T}_4 Z_{1,2} = \begin{pmatrix} \gamma_1 & & & \\ \omega_2 & \bar{\gamma}_2 & \eta_3 & \\ \pi_3 & \bar{\omega}_3 & \delta_3 & \\ & & \eta_4 & \delta_4 \end{pmatrix}$$

with $\bar{\gamma}_2 = (\eta_2\delta_1\delta_2)/\gamma_1$, $\pi_3 = s_1\eta_3$, $\bar{\omega}_3 = -c_1\eta_3$

We have created a fill–in in position $(3,1)$

For the next step, we multiply by

$$Z_{2,3} = \begin{pmatrix} 1 & & & \\ & c_2 & s_2 & \\ & s_2 & -c_2 & \\ & & & 1 \end{pmatrix}$$

$$T_4 Z_{1,2} Z_{2,3} = \begin{pmatrix} \gamma_1 & & & \\ \omega_2 & \gamma_2 & & \\ \pi_3 & \omega_3 & \bar{\gamma}_3 & \\ & \pi_4 & \bar{\omega}_4 & \delta_4 \end{pmatrix}$$

$\gamma_2 = \sqrt{\gamma_2^2 + \eta_3^2}$, $s_2 = \eta_3/\gamma_2$, $c_2 = \bar{\gamma}_2/\gamma_2$,

$\bar{\gamma}_3 = (\eta_3\bar{\omega}_3 - \bar{\gamma}_2\delta_3)/\gamma_2$, $\bar{\omega}_4 = -c_2\eta_4$, $\pi_4 = s_2\eta_4$

Now, the process is clear: $Z_{j,j+1}$ differs from the identity only in the elements $z_{j,j} = -z_{j+1,j+1} = c_j = \cos\theta_j$, $z_{j,j+1} = z_{j+1,j} = s_j = \sin\theta_j$ and

$$\bar{T}_k Z_{1,2} \cdots Z_{k-1,k} = T_k Z_k^T = \bar{L}_k = \begin{pmatrix} \gamma_1 & & & & \\ \omega_2 & \gamma_2 & & & \\ \pi_3 & \omega_3 & \gamma_3 & & \\ & \ddots & \ddots & \ddots & \\ & & \pi_k & \omega_4 & \bar{\gamma}_k \end{pmatrix}$$

The last rotation is defined by

$$\gamma_k = \sqrt{\bar{\gamma}_k^2 + \eta_{k+1}^2}, \qquad s_k = \frac{\eta_{k+1}}{\gamma_k}, \qquad c_k = \frac{\bar{\gamma}_k}{\gamma_k}$$

$\bar{\omega}_{k+2} = -c_k \eta_{k+2}$, $\omega_{k+1} = \bar{\omega}_{k+1} c_k + s_k \delta_{k+1}$

$\pi_{k+2} = s_k \eta_{k+2}$

Let us define $\bar{L}_k$ as being identical to L_k except for the (k,k) element which is replaced by γ_k

$$\overline{W}_k = [w^1 \cdots w^{k-1}\, \bar{w}^k] = [W_{k-1}\, \bar{w}^k] = Q_k Z_k^T$$

$$\bar{z}^k = (\zeta_1, \ldots, \zeta_{k-1}, \bar{\zeta}_k)^T = ((z^{k-1})^T, \bar{\zeta}_k)^T = Z_k y^k$$

With these notations

$$L_k \bar{z}^k = \eta_1 e^1, \quad x_{CG}^k = \bar{W}_k \bar{z}^k$$

As we have $L_k \bar{z}^k = \eta_1 e^1$ and $\bar{L}_k z^k = \eta_1 e^1$ we get

$$\zeta_k = \bar{\zeta}_k \frac{\bar{\gamma}_k}{\gamma_k} = \bar{\zeta}_k c_k$$

By looking at the last two columns of the matrix equality
$\bar{W}_{k+1} = Q_{k+1} Z_{k+1}^T$
as $\bar{W}_{k+1} = [W_k \, \bar{w}^{k+1}]$

$$[\bar{w}^k \, q^{k+1}] \begin{pmatrix} c_k & s_k \\ s_k & -c_k \end{pmatrix} = [w^k \, \bar{w}^{k+1}], \quad \bar{w}^1 = q^1$$

We do not want to compute x_{CG}^k at each iteration since L_k can be singular

However, $\bar{L}_k$ is non–singular as long as $\eta_{k+1} \neq 0$, so Z^k is always well defined

$$x_S^k = W_k z^k$$

$$x_S^k = x_S^{k-1} + \zeta_k w^k$$

w^k is obtained by applying the plane rotation to $\bar{w}^k$ and q^{k+1}

The CG iterate can be obtained (if it exists) through

$$x_{CG}^{k+1} = x_S^k + \bar{\zeta}_{k+1} \bar{w}^{k+1}$$

$r^0 = b - Aw^0,\ d_1 = \|r^0\|,\ q^1 = r^0/d_1,\ \delta_1 = (q^1, Aq^1),\ \theta = \delta_1,\ \nu = 0,$
$\bar{q} = Aq^1 - \delta_1 q^1,\ w = Aq^1,\ \eta_2 = \|\bar{q}\|,\ q^2 = \bar{q}/\eta_2,\ \bar{\gamma} = \delta_1,\ \bar{\omega} = \eta_2,$

for $k = 2, \ldots$

$$\delta_k = (Aq^k, q^k)$$

$$\bar{q} = Aq^k - \delta_k q^k - \eta_k q^{k-1}$$

$$\eta_{k+1} = \|\bar{q}\|, \quad q^{k+1} = \bar{q}/\eta_{k+1}$$

$$\gamma = \sqrt{\bar{\gamma}^2 + \eta_k^2}, c = \bar{\gamma}/\gamma, s = \eta_k/\gamma$$

$$\omega = c\bar{\omega} + s\delta_k, \bar{\gamma} = s\bar{\omega} - c\delta_k, \pi = s\eta_{k+1}, \bar{\omega} = -c\eta_{k+1}$$

$$\zeta = \theta/\gamma, \sigma = c\zeta, \tau = s\zeta$$

$$x^{k+1} = x^k + (\sigma w^k + \tau q^{k+1})$$

$$w^{k+1} = s w^k + c q^{k+1}$$

$$\theta = \nu - \omega\zeta, \quad \nu = -\pi\zeta$$

The minimal residual method

We use the Lanczos vectors to derive an implementation of the CR algorithm when the matrix A could be indefinite

$$x^k = x^0 + Q_k y$$

$$\|r^k\| = \|b - A(x^0 + Q_k y)\| = \|r^0 - AQ_k y\|$$

$$= \|Q_{k+1}(\eta_1 e^1 - \tilde{T}_k y)\| = \|\eta_1 e^1 - \tilde{T}_k y\|$$

Then we use a QR factorization of the $(k+1) \times k$ matrix $\tilde{T}_k$

$$Q_k^T A^2 Q_k y^k = Q_k^T A b, \quad x^k = Q_k y^k$$

$$Q_k^T A^2 Q_k = \bar{T}_k^2 + \eta_{k+1}^2 e^k (e^k)^T$$

$$Q_k^T A b = \eta_1 \bar{T}_k e^1$$

Using the LQ factorization of $\bar{T}_k$, we have
$\bar{T}_k^2 + \eta_{k+1}^2 e^k (e^k)^T = L_k L_k^T$

$$L_k L_k^T y^k = \eta_1 \bar{L}_k Z_k e^1$$

But $\bar{L}_k$ differs only from L_k in the (k,k) element

We can write $\bar{L}_k = L_k D_k$ where D_k is equal to the identity matrix of order k except for the (k,k) element which is c_k

$$L_k^T y^k = \eta_1 D_k Z_k e^1$$

Let us denote $t^k = \eta_1 D_k Z_k e^1$

Since $Z_k = Z_{k-1,k} \cdots Z_{1,2}$

$$t_1^k = \eta_1 c_1, \quad t_j^k = \eta_1 s_1 s_2 \cdots s_{j-1} c_j, \; j = 2, \ldots, k$$

Let $V_k = Q_k L_k^{-T}$

V_k can be computed column by column starting from the first one and

$$x_M^k = Q_k y^k = Q_k L_k^{-T} L_k^T y^k = V_k t^k$$

$$x_M^k = Q_k L_k^{-T} (\eta_1 D_k Z_k e^1)$$

Non symmetric systems

Simplest method: normal equations

$$Ax = b$$

$$A^T A x = A^T b$$

and we use CG, but

$$\kappa(A^T A) = \kappa(A)^2$$

Variant:

$$A A^T y = b$$

$$x = A^T y$$

Preconditioners

Gérard MEURANT
CEA/DIF
Roscoff 2005

June 1, 2005

Instead of

$$Ax = b$$

solve

$$M^{-1}Ax = M^{-1}b$$

or

$$M^{-\frac{1}{2}}AM^{-\frac{1}{2}}y = M^{-\frac{1}{2}}b$$

in the symmetric case

"M^{-1}" A is supposed to have better properties than A for convergence of the iterative methods

Preconditioners

Suppose A SPD large and sparse

- M SPD
- M sparse
- M easy and cheap to compute
- $Mz = r$ easy to solve
- "good" eigenvalue distribution for $M^{-1}A$

- Constructing good preconditioners is more art than science
  - Computing M must be parallel
  - Solving $Mz = r$ must be parallel

Simplest idea: (Jacobi)

$$M = D = diag(A)$$

SSOR (Axelsson)

$$A = D + L + L^T$$

$$M = \frac{1}{\omega(2-\omega)}(D + \omega L)D^{-1}(D + \omega L^T)$$

For the Poisson equation

$$\kappa(A) = O(\frac{1}{h^2})$$

$$\exists\ \omega_{opt}\ \text{tq}\ \kappa(M^{-1}A) = O(\frac{1}{h})$$

Pb: how to choose ω ? In practice $\omega = 1$

Incomplete Cholesky decomposition

André–Louis Cholesky, 1875–1918

(. . . , Meijerink & Van der Vorst (1977),. . .)

The (complete) Cholesky outer product factorization

$$A = \bar{L}\bar{\Sigma}\bar{L}^T$$

The first step is

$$\bar{L}_1 = \begin{pmatrix} 1 & 0 \\ \bar{l}_1 & I \end{pmatrix}, \quad \bar{\Sigma}_1 = \begin{pmatrix} a_{1,1} & 0 \\ 0 & \bar{A}_2 \end{pmatrix}$$

and

$$A = \begin{pmatrix} a_{1,1} & \bar{a}_1^T \\ \bar{a}_1 & \bar{B}_1 \end{pmatrix} = \bar{L}_1\bar{\Sigma}_1 L_1^T$$

$$\bar{l}_1 = \frac{\bar{a}_1}{a_{1,1}}$$

$$\bar{A}_2 = \bar{B}_1 - \frac{1}{a_{1,1}} \bar{a}_1 \bar{a}_1^T$$

$$\bar{A}_2 = \begin{pmatrix} \bar{a}_{2,2}^{(2)} & \bar{a}_2^T \\ \bar{a}_2 & \bar{B}_2 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ \bar{l}_2 & I \end{pmatrix} \begin{pmatrix} \bar{a}_{2,2}^{(2)} & 0 \\ 0 & \bar{A}_3 \end{pmatrix} \begin{pmatrix} 1 & \bar{l}_2^T \\ 0 & I \end{pmatrix}$$

$$\bar{L}_2 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & \bar{l}_2 & I \end{pmatrix}$$

$$\bar{\Sigma}_1 = \begin{pmatrix} a_{1,1} & 0 \\ 0 & \bar{A}_2 \end{pmatrix} = \bar{L}_2 \begin{pmatrix} a_{1,1} & 0 & 0 \\ 0 & \bar{a}_{2,2}^{(2)} & 0 \\ 0 & 0 & \bar{A}_3 \end{pmatrix} \bar{L}_2^T = \bar{L}_2 \bar{\Sigma}_2 \bar{L}_2^T$$

After two steps, we have $A = \bar{L}_1 \bar{L}_2 \bar{\Sigma}_2 \bar{L}_2^T \bar{L}_1^T$

$$\bar{L}_1 \bar{L}_2 = \begin{pmatrix} 1 & 0 \\ \bar{l}_1 & \begin{pmatrix} 1 & 0 \\ \bar{l}_2 & I \end{pmatrix} \end{pmatrix}$$

Finally, if all the pivots are non–zero,

$$A = \bar{L}_1 \cdots \bar{L}_{n-1} \bar{\Sigma} \bar{L}_{n-1}^T \cdots \bar{L}_1^T$$

This works if A is SPD

Let $G = \{(i,j),\ i > j\}$ be a given set of indices

$$A = L\Sigma L^T - R$$

L being lower triangular with $l_{i,i} = 1$ and Σ diagonal

Construct L s.t. that $l_{i,j} = 0$ if $(i,j) \notin G$

$$A = A_1 = \begin{pmatrix} a_{1,1} & a_1^T \\ a_1 & B_1 \end{pmatrix} = \begin{pmatrix} a_{1,1} & b_1^T \\ b_1 & B_1 \end{pmatrix} - \begin{pmatrix} 0 & r_1^T \\ r_1 & 0 \end{pmatrix} = M_1 - R_1$$

$$a_1 = b_1 - r_1$$

$$(b_1)_i = 0, \text{ if } (i,1) \notin G \Rightarrow (r_1)_i = -(a_1)_i$$

$$(b_1)_i = (a_1)_i, \text{ if } (i,1) \in G \Rightarrow (r_1)_i = 0$$

Then,

$$M_1 = \begin{pmatrix} 1 & 0 \\ l_1 & I \end{pmatrix} \begin{pmatrix} a_{1,1} & 0 \\ 0 & A_2 \end{pmatrix} \begin{pmatrix} 1 & l_1^T \\ 0 & I \end{pmatrix} = L_1 \Sigma_1 L_1^T$$

$$l_1 = \frac{b_1}{a_{1,1}}$$

$$A_2 = B_2 - \frac{1}{a_{1,1}} b_1 b_1^T$$

Then, we use the same process on A_2

$$A_2 = \begin{pmatrix} a_{2,2}^{(2)} & a_2^T \\ a_1 2 & B_2 \end{pmatrix} = \begin{pmatrix} a_{2,2}^{(2)} & b_2^T \\ b_2 & B_2 \end{pmatrix} - \begin{pmatrix} 0 & r_2^T \\ r_2 & 0 \end{pmatrix} = M_2 - R_2$$

where b_2 is obtained from a_2 by zeroing the elements $(i, 2)$ whose indices do not belong to G

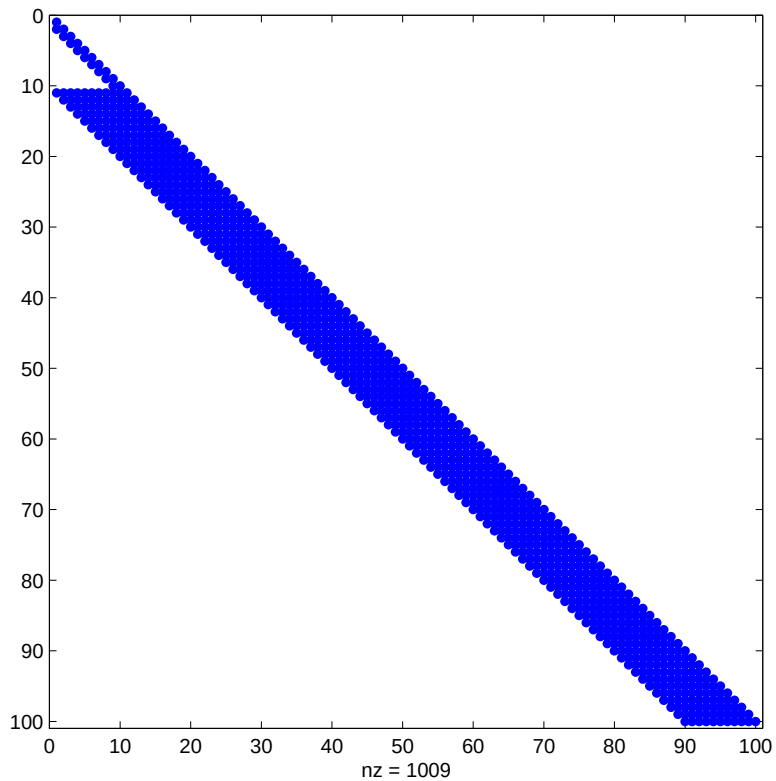
$$L_2 = \begin{pmatrix} 1 & 0 \\ 0 & \begin{pmatrix} 1 & 0 \\ l_2 & I \end{pmatrix} \end{pmatrix}$$

$$A = L_1 L_2 \Sigma_2 L_2^T L_1^T - L_1 \begin{pmatrix} 0 & 0 \\ 0 & R_2 \end{pmatrix} L_1^T - R_1$$

$$L_1 L_2 = \begin{pmatrix} 1 & 0 \\ l_1 & \begin{pmatrix} 1 & 0 \\ l_2 & I \end{pmatrix} \end{pmatrix} \quad \text{and } L_1 \begin{pmatrix} 0 & 0 \\ 0 & R_2 \end{pmatrix} L_1^T = \begin{pmatrix} 0 & 0 \\ 0 & R_2 \end{pmatrix}$$

This algorithm has constructed a complete decomposition of $M = A + R$

For the Poisson equation



In the Cholesky decomposition of A, $A = \tilde{L}\tilde{L}^T$, we get fill–in

Incomplete Cholesky (IC) is easy for 5 diagonal matrices

$$M = L \; D^{-1} \; L^T$$

If

$$A = (c_i, b_i, a_i, b_{i+1}, c_{i+m})$$

$$L = (\bar{c}_i, \bar{b}_i, d_i, 0, 0), \;\; D = (0, 0, d_i, 0, 0)$$

Then,

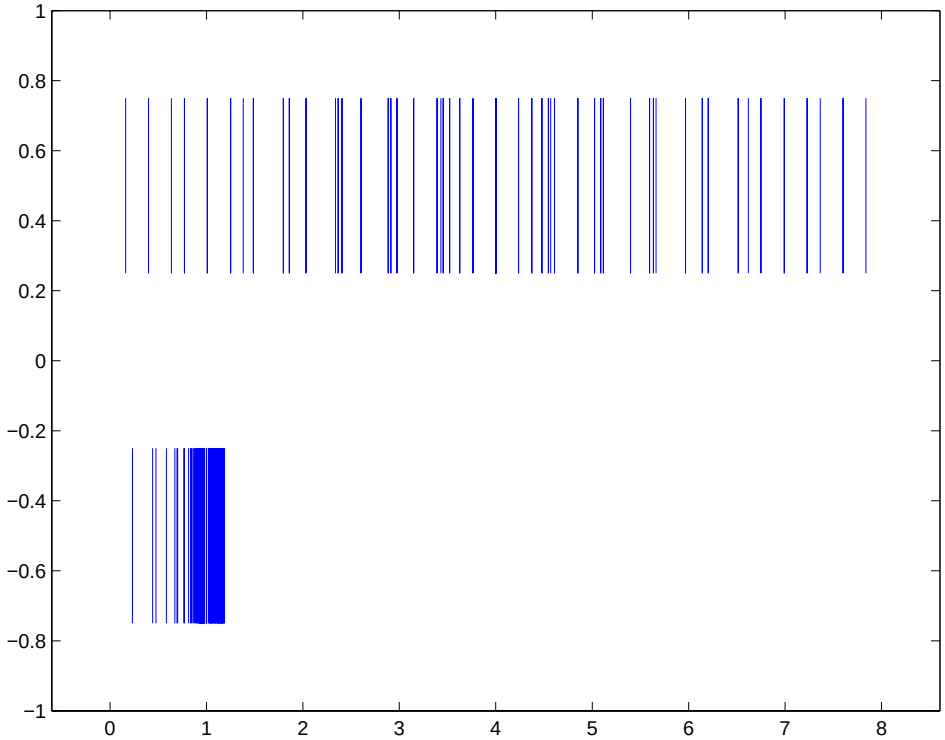
$$\bar{c}_i = c_i, \quad \bar{b}_i = b_i$$

$$d_i = a_i - \frac{b_i^2}{d_{i-1}} - \frac{c_i^2}{d_{i-m}}$$

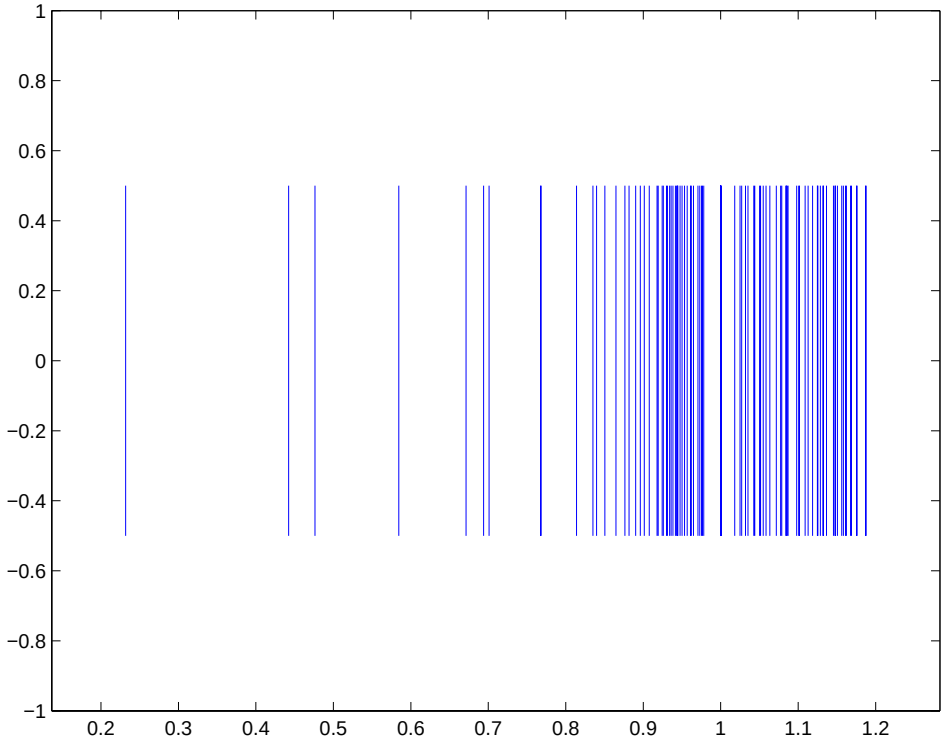
This is IC(1,1). For the Poisson equation

$$\kappa(M^{-1}A) = O(\frac{1}{h^2})$$

but a "good" distribution of eigenvalues



Poisson, $m = 10$, A and $IC(1,1)(A)$



Poisson, $m = 10$, $IC(1,1)(A)$

We can use other orderings of the unknowns

$$A_P = PAP^T$$

Orderings can have a very large impact on convergence

- Lichnewsky 1984 (nested dissection)
- Simon 1985
- Duff–Meurant 1989 (numerical experiments)

Theoretical explanation:

- V. Eijkhout 1990
- S. Doi 1990

$$M = LDL^T = A + R$$

Ordering examples

- ROW (row)
- CM (Cuthill–Mc Kee)
- MIND (Minimum degree)
- RB (Red–Black)
- ND (Nested dissection)
- VDV2 (Van der Vorst)

Table 1: Poisson 30$\times$ 30 mesh

ordering	nit	nb of fills	nb R	$\|R\|_F^2$
ROW	23	24389	841	142.5
CM	23	16675	841	142.5
MIND	39	7971	1582	467.3
RB	38	12853	1681	525.5
ND	25	15228	1012	157.1
VDV2	20	17413	841	140.7

nb elements in L : 2639

Table 2: Anisotropic problem $a = 100, b = 1$

ordering	nit	nb of fills	nb R	$\|R\|_F^2$
ROW	9	24389	841	$0.12\ 10^4$
CM	9	16675	841	$0.12\ 10^4$
MIND	48	7971	1582	$0.18\ 10^7$
RB	47	12853	1681	$0.21\ 10^7$
ND	26	15228	1012	$0.43\ 10^6$
VDV2	9	17413	841	$0.11\ 10^4$

Doi et Eijkhout theory explain these results

Situation is different if we keep some fills

Table 3: Poisson with 1 level of fill

ordering	nit	nb of fill	nb R	nb L	$\|R\|_F^2$
ROW	17	24389	1653	3481	24.7
CM	17	16675	1653	3481	24.7
MIND	23	7971	2467	4222	38.81
RB	16	12853	2016	4321	16.47
ND	19	15228	2187	3652	35.34
VDV2	17	17413	1651	3481	25.20

Table 4: Anisotropic problem with 1 level of fill

ordering	nit	nb of fill	nb R	nb L	$\|R\|_F^2$
ROW	8	24389	1653	3481	823
CM	8	16675	1653	3481	844
MIND	27	7971	2467	4222	$0.22\ 10^6$
RB	8	12853	2016	4321	806
ND	23	15228	2187	3652	$0.18\ 10^6$
VDV2	8	17413	1651	3481	795

Why do we get better results with RB when we keep more fill–ins?

If

$$A = LDL^T$$

$$\|A\|_F = \left(\sum_{i,j} a_{i,j}^2 \right)^{1/2}$$

$$\|A\|_F^2 = trace(A^T A) = trace(AA^T)$$

$$\|L\sqrt{D}\|_F^2 = trace(LDL^T) = trace(A)$$

If $A_P = PAP^T$

$$A_P = L_P D_P^{-1} L_P^T$$

then

$$\|PAP^T\|_F = \|A\|_F$$

and

$$\|L_P \sqrt{D_P^{-1}}\|_F = \|L\sqrt{D^{-1}}\|_F = \sqrt{trace(A)}, \quad \forall P$$

If we have a few fills, they are large. This is what happens with RB

Modified preconditioners

(Dupont, Kendall & Rachford)

$$M = LD^{-1}L^T = A + R$$

Modify D s.t. $rowsum(R) = 0$ or ch^2

For a 5 diagonal matrix:

$$d_i = (1 + ch^2)a_i - \frac{b_{i-1}(b_{i-1} + c_{i-1})}{d_{i-1}}$$

$$- \frac{c_{i-m}(c_{i-m} + b_{i-m})}{d_{i-m}}$$

For Poisson

$$\kappa(M^{-1}A) = O(\frac{1}{h})$$

instead of $O(1/h^2)$

This can be generalized to non symmetric matrices: ILU without pivoting

However, beware of instabilities

Polynomial preconditioners

$$M^{-1} = P_k(A) = \sum_{i=0}^{k} \alpha_i A^i$$

P_k polynomial of degree k

Eigenvalues of $M^{-1}A$ are $P_k(\lambda_i)\lambda_i$

We would like $P_k(\lambda)\lambda$ to be close to 1 on $[\lambda_{min}, \lambda_{max}] \subset [a, b]$

Neumann series

$$A = D - L - L^T$$

$$A = D^{1/2}(I - D^{-1/2}(L + L^T)D^{-1/2})D^{1/2}$$

$$A^{-1} = D^{-1/2}(I - D^{-1/2}(L + L^T)D^{-1/2})^{-1}D^{-1/2}$$

We have

$$\rho(I - D^{-1}A) = \rho(D^{-1}(L + L^T)) < 1$$

We choose

$$M^{-1} = D^{-1/2}[I + D^{-1/2}(L + L^T)D^{-1/2}]D^{-1/2}$$

$$= D^{-1} + D^{-1}(L + L^T)D^{-1}$$

or more terms (odd) summed to D^{-1}

MINMAX polynomial

(Johnson, Michelli & Paul)

$$q_{k+1}(\lambda) = p_k(\lambda)\lambda$$

$$\mathcal{Q}_k = \{\text{polynomials of deg } k, \text{ positive, value 0 at 0 }\}$$

Eigenvalues of A in $[a, b]$, minimize over $\mathcal{Q}_k$

$$cond(q) = \frac{sup_{\lambda \in [a,b]} q_{k+1}(\lambda)}{inf_{\lambda \in [a,b]} q_{k+1}(\lambda)}$$

Solution:

$$q_{k+1}(\lambda) = 1 - \frac{C_{k+1}(\mu(\lambda))}{C_{k+1}(\mu(0))}$$

with $\mu(\lambda) = \frac{2\lambda - b - a}{b - a}$ and C_k Chebychev polynomial

Least squares polynomial

(Saad)

Look for s, polynomial which minimizes

$$\int_a^b (1 - \lambda s(\lambda))^2 w(\lambda)\ d\lambda$$

w is a weight

Usual choice:

$$w(\lambda) = (b - \lambda)^\alpha (\lambda - a)^\beta$$

$\alpha \geq \beta \geq -1/2$: Jacobi polynomial

In practice $\alpha = \beta = -\frac{1}{2}$ (Chebychev) or $\alpha = \beta = 0$ (Legendre)

Drawbacks:

- we need a and b

- when the degree is large ($\geq$ 10 ou 20) applying the polynomial can be difficult because of rounding errors (32 bits) – instability of Horner's scheme

- Use recurrences (orthogonal polynomials)

Approximate inverses

- Huckle et Grote (1994)
- Gould et Scott (1995)
- Chow et Saad (1994–1995)
- Benzi (1995–1996)

We want $M^{-1}A$ "to look like" I

We compute $C = M^{-1}$ to minimize

$$\|AC - I\| \text{ or } \|CA - I\|$$

Generally one takes the Frobenius norm:

$$\|AC - I\|_F^2 = \sum_{k=1}^{n} \|(AC - I)e_k\|^2,$$

e_k k–th column of I

We minimize the l_2 norms

$$\|Ac_k - e_k\|, \ k = 1, \ldots, n$$

n independent least squares problems (parallel)

Generally A^{-1} is dense, how to choose the sparsity structure of c_k?

Let $\hat{c}_k$ be the vector of the non zero elements of c_k

Let $\hat{A}_k$ be the matrix whose columns are those of A with indices $G_k = \{j | (c_k)_j \neq 0\}$ and whose rows i are such that $\exists a_{i,j} \neq 0, j \in G_k$

$$\min_{\hat{c}_k} \|\hat{A}_k \hat{c}_k - \hat{e}_k\|, \quad k = 1, \ldots, n$$

These small least squares problems are solved with QR

Structure of c_k

Huckle & Grote start from G_k^0 (diagonal or same structure as A)

One solves the problem and augment G_k iteratively

At iteration p, consider the residual $r = Ac_k^p - e_k$

We want to decrease $\|r\|$

Let $\mathcal{L} = \{j|(r)_j \neq 0\}$ and $\forall l \in \mathcal{L}$ $\mathcal{N}_l = \{j|a_{l,j} \neq 0, j \notin G_k^p\}$

The candidates are chosen in

$$\cup_{l \in \mathcal{L}} \mathcal{N}_l$$

For j in this set, solve

$$\min_{\mu_j \in \Re} \|r + \mu_j A e_j\| \quad \Longrightarrow \quad \mu_j = -\frac{(r, Ae_j)}{\|Ae_j\|^2}$$

The new residual is

$$\|r\|^2 - \frac{(r, Ae_j)^2}{\|Ae_j\|^2}$$

One chooses indices that give the smallest residuals and iterate the process

This method is denoted as SPAI

Parallel implementation was considered by Deshpande, Grote, Messmer and Sawyer

Gould and Scott improved the choice of the new indices

Chow and Saad iteratively solve $Ac_j = e_j$ (which is as hard as the original problem) with a small number of iterations

They can precondition with the already computed columns
For these methods, there are conditions for C being non

singular

Remark that C is not symmetric

We can keep symmetry by computing only the lower triangular part, but this is not parallel Is C positive definite?

One can look for C as KK^T

Benzi, Meyer et Tuma approximate inverse

A SPD

If $Z = [z_1, z_2, \ldots, z_n]$ is a set of conjugate directions for A,

$$Z^T A Z = D$$

D diagonal and $A^{-1} = Z D^{-1} Z^T$

The directions are computed by Gram–Schmidt applied to $v_1, v_2, \ldots, v_n$

If $V = [v_1, v_2, \ldots, v_n] = I$, Z is upper triangular

1) $z_i^{(0)} = e_i, \quad i = 1, \ldots, n$

2) for $i = 1, \ldots, n \quad d_j^{(i-1)} = (a_i, z_j^{(i-1)}), \quad j = i, \ldots, n$ where a_i is the ith column of A

if $j \neq n$, $z_j^{(i)} = z_j^{(i-1)} - \left(\frac{d_j^{(i-1)}}{d_i^{(i-1)}} \right) z_i^{(i-1)}, \quad j = i+1, \ldots, n$

3) $z_i = z_i^{(i-1)}, d_i = d_i^{(i-1)}, \quad i = 1, \ldots, n$

To preserve the sparsity structure, fills are thrown away based on position or value (or both)

This method is known as AINV

Benzi, Meyer and Tuma show that this method is feasible for H–matrices

There exists a "robust" variant SAINV (Benzi, Cullum and Tuma)

This is generalized to non symmetric matrices by considering two sets $Z = [z_1, \ldots, z_n]$ and $W = [w_1, \ldots, w_n]$ such that

$$W^T A Z = D$$

Pb : Poisson, L-shaped region, mixed b.c.

Table 5: Comparison between IC and AINV (Benzi)

IC			AINV		
fill	nb. iter	time	fill	nb. iter	time
675	87	0.33	743	76	0.32
897	53	0.18	780	74	0.32
912	51	0.18	1135	54	0.26
1204	38	0.14	1208	47	0.18
1439	32	0.14	1300	40	0.21
1565	24	0.10	3654	22	0.14

Table 6: Comparison between SPAI and AINV (Benzi)

Matrix	SPAI			AINV		
	Its	init	t its	Its	init	t its
3DCD	40	10.63	0.111	25	1.885	0.068
ALE3D	45	30.79	0.088	43	1.446	0.094
ORSREG1	40	3.309	0.033	33	0.550	0.031
SHERMAN1	62	0.878	0.029	43	0.201	0.021
PORES3	111	0.941	0.044	75	0.127	0.038
WATT2	377	2.590	0.384	111	0.505	0.116

Algebraic multilevel preconditioners

Goal

Efficiently solve linear systems arising from PDE discretizations on Terascale parallel computers

$$Ax = b$$

A large sparse of order n

The iterative method must be (almost) scalable

We use Krylov methods. The preconditioner must be such that:

- the number of iterations is (almost) constant, when the problem size is increased
- the complexity of applying the preconditioner is proportional to n
- easy to construct and use on a parallel computer

Contents

- Recap on multilevel preconditioners
- The SLOOP library
- Update on numerical experiments
- Block extension

Recap on multilevel preconditioners

- grid $\equiv$ (sub) set of unknowns without overlap
- Algebraic MultiGrid (AMG)–like V–cycle:

Starting from the zero vector:

0– coarsest level: exact solve by Gaussian elimination or use diagonal CG, otherwise

1– do ν iterations of smoothing

2– restrict the residual r to $r_c = Rr$

3– recursively solve $A_c e_c = r_c$, $A_c = RAP$, $R = P^T$

4– interpolate e_c to $e = Pe_c$

5– add the correction e to the current iterate

6– do ν iterations of smoothing

We have to define:

- the smoother
- coarsening algorithm
- the interpolation

Suppose now A SPD

Use a partition of the graph of A to parallelize

Smoothers

- Symmetric Gauss–Seidel

parallelized by using Jacobi for the interface nodes (SGSJ)

- Incomplete Cholesky (IC) $M = LDL^T$

parallelized by ignoring dependencies between subdomains (ICp)

$$LD^{-1}L^T(x^{k+1} - x^k) = b - Ax^k$$

- Approximate inverse AINV from M. Benzi and al.

$M^{-1} = ZD^{-1}Z^T$ where Z is upper triangular and D is diagonal

$$x^{k+1} = x^k + M^{-1}(b - Ax^k)$$

Other ingredients

- Influence matrix

$$S_i^A = \{j \mid |a_{i,j}| > \tau \max_{k \neq i} |a_{i,k}|, \quad \tau < 1\}$$

Keep at least the index of the largest element in magnitude

- Coarsening algorithm (using S)

$$\mathcal{N} = F \cup C$$

- Ruge-Stuben (Wagner) RS
- LLNL algorithms (Cleary, Falgout, Henson and Jones)

- Standard interpolation algorithm (with entries of A), projection P, restriction $R = P^T$

- Coarse matrices

$$A_C = RAP$$

The SLOOP Library

Solveurs Linéaires Orientés Objet Parallèles

Parallel Object Oriented Linear Solvers

- Symmetric and non symmetric iterative solvers
- Written in C++ at CEA
- Symmetric matrices:
  - SPD: PCG (with estimates of norm of the error)

    soon SYMMLQ for indefinite matrices
  - IC(k), polynomials, (S)AINV, AMG

- Non symmetric matrices:
  - BiCGstab, BiCGStab(2), GMRES(m)
  - ILU(k), AINV, SPAI, AMG

All algorithms are parallelized as much as possible

SLOOP has been optimized since last meeting but is still under development

Probably released to the public in a few months

Update on numerical experiments on the CEA computer

5(7) point finite differences, unit square(cube), $m \times m(\times m)$ mesh

b random

$x^0 = 0$

stopping criterion $\|r^k\| \leq 10^{-10}\|r^0\|$

Mesh decomposition into squares(cubes) with $m_p^{2(3)}$ unknowns per processor

• Partition the graph of A (mesh) into non overlapping subdomains but with ghost nodes

Nodes in relation to ghost nodes are interface nodes

• A is distributed by rows

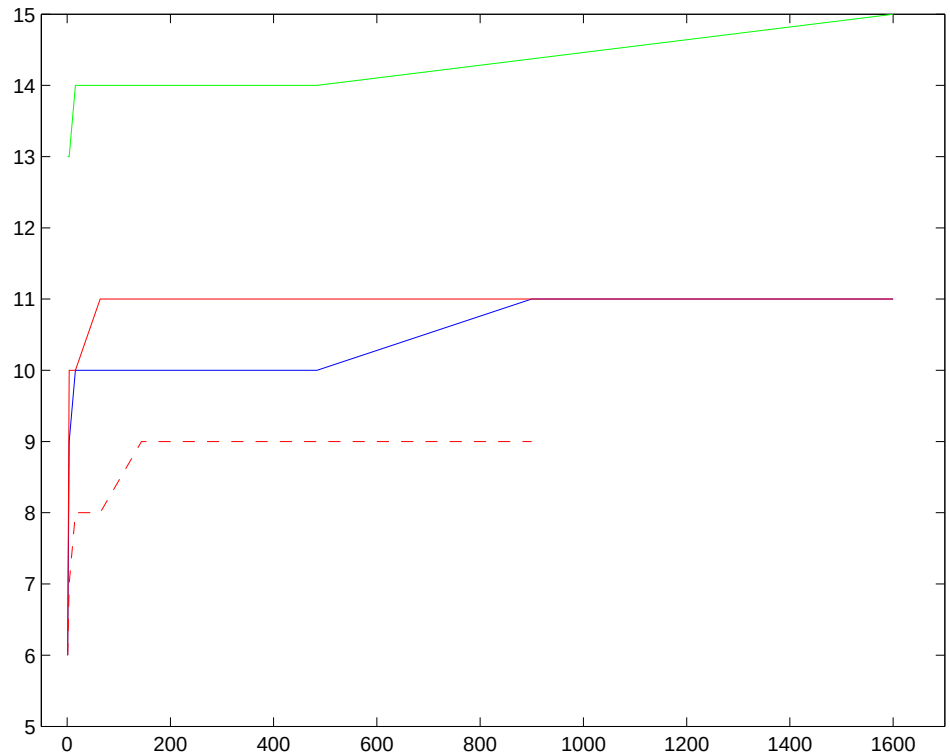
First experiment

- Poisson

$m_p = 250 \rightarrow 62500$ unknowns per processor,

$p = 1, 4, 16, 64, 144, 256, 484, 900, 1600$

- Largest problem is $\simeq 100\ 10^6$ unknowns

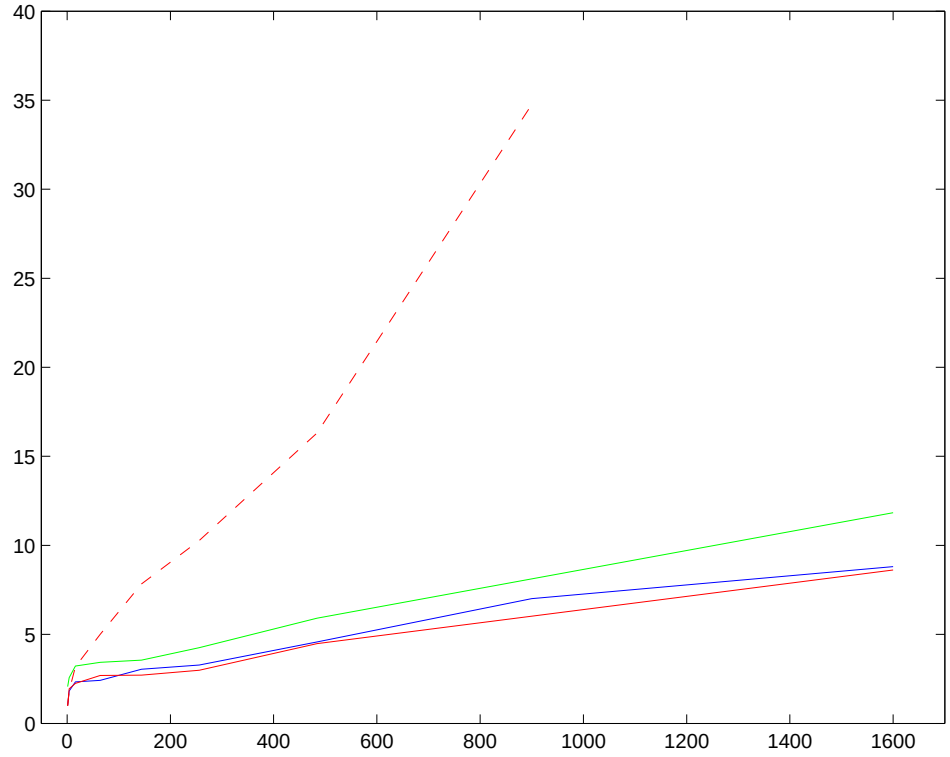


Nb of iterations for the Poisson equation as a function of p

coarsening: LLNL

blue: SGSJ, red dashed: IC, red: ICp, green: SAINVp

62500 unknowns per processor



Elapsed time (s) for the Poisson equation as a function of p

coarsening: LLNL

blue: SGSJ, red dashed: IC, red: ICp, green: SAINVp

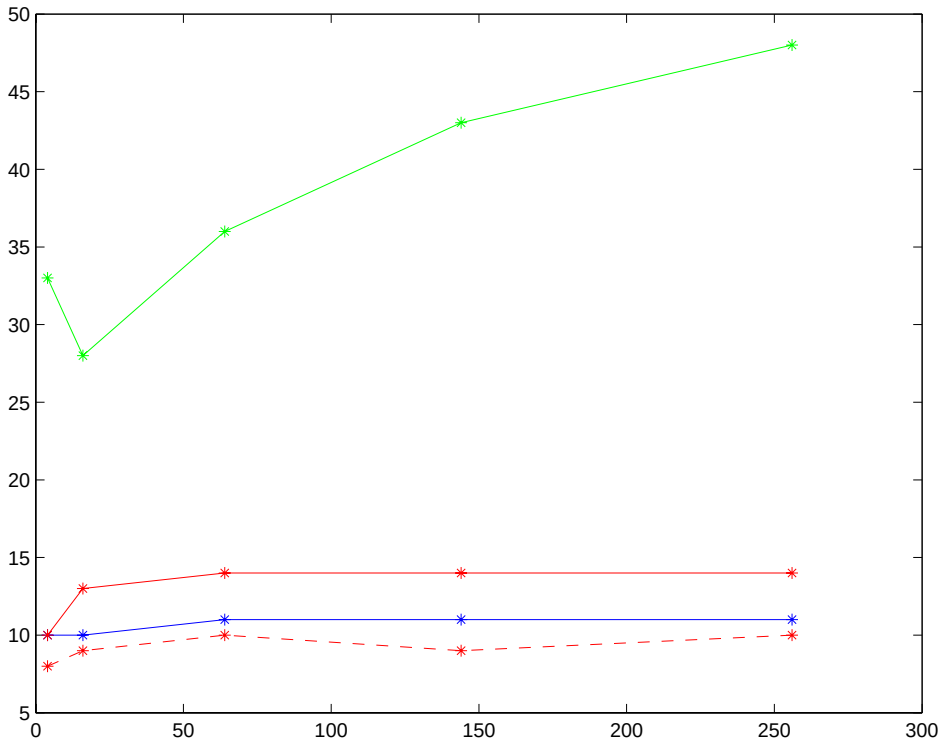
Second experiment

- Discontinuous problem

$m_p = 250 \rightarrow 62500$ unknowns per processor,

$p = 4, 16, 64, 144, 256$

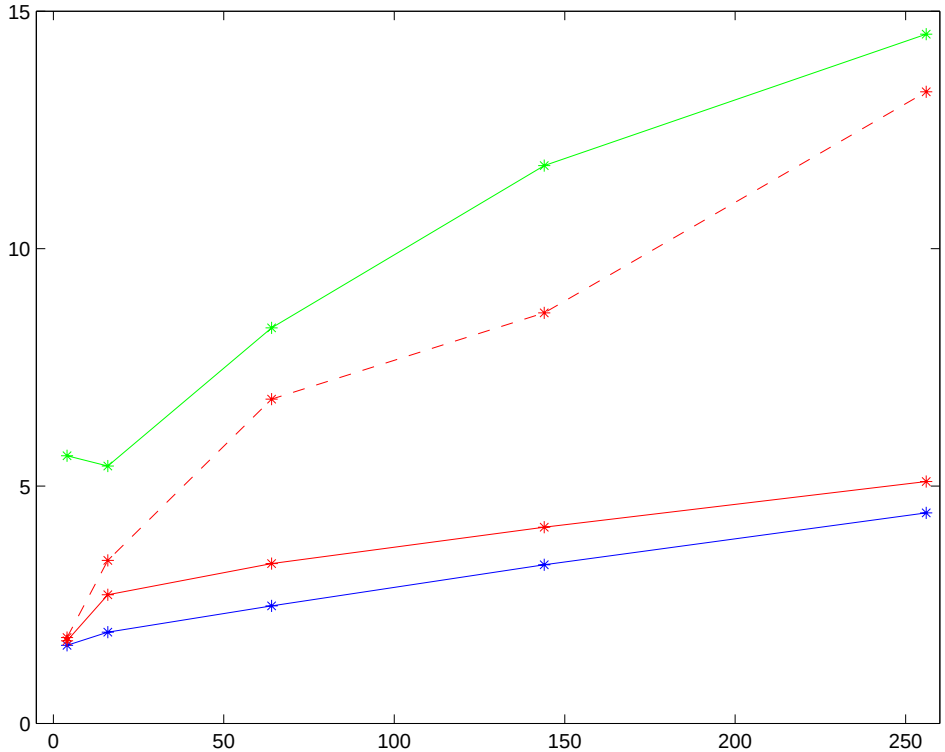
- Largest problem is $16\ 10^6$ unknowns



Nb of iterations for the discontinuous problem as a function of p

coarsening: LLNL

blue: SGSJ, red dashed: IC, red: ICp, green: SAINVp



Elapsed time (s) for the discontinuous problem as a function of p

coarsening: LLNL

blue: SGSJ, red dashed: IC, red: ICp, green: SAINVp

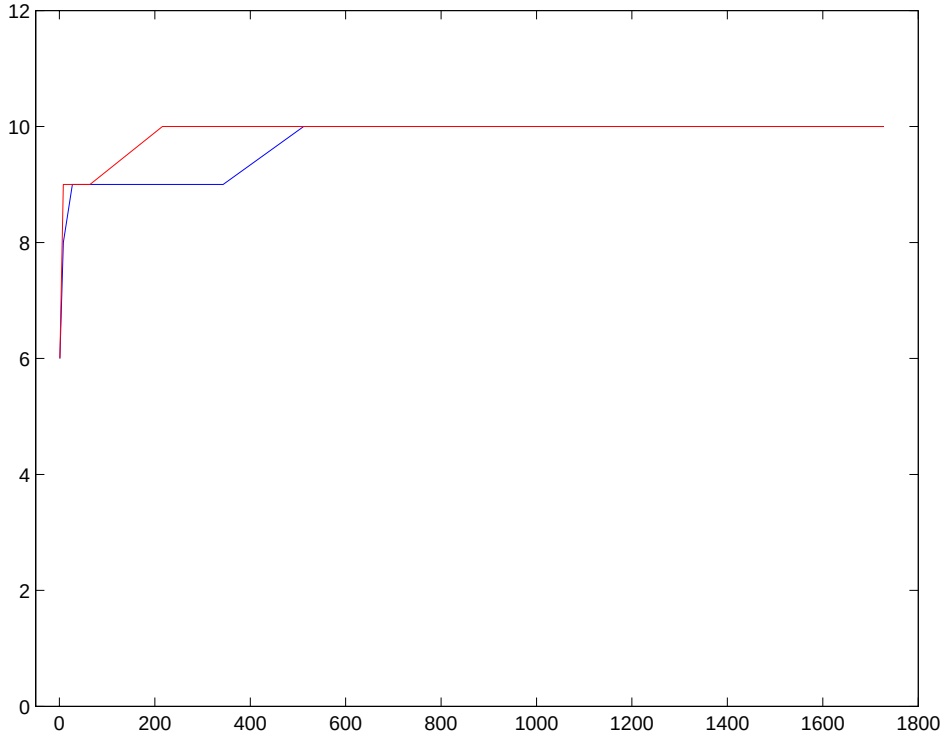
Third experiment

- 3D Poisson in $[0,1]^3$, 7 point finite differences

$m_p = 30 \rightarrow m_p^3 = 27000$ unknowns per processor,

$p = 1, 8, 27, 64, 125, 216, 343, 512, 729, 1331, 1728$

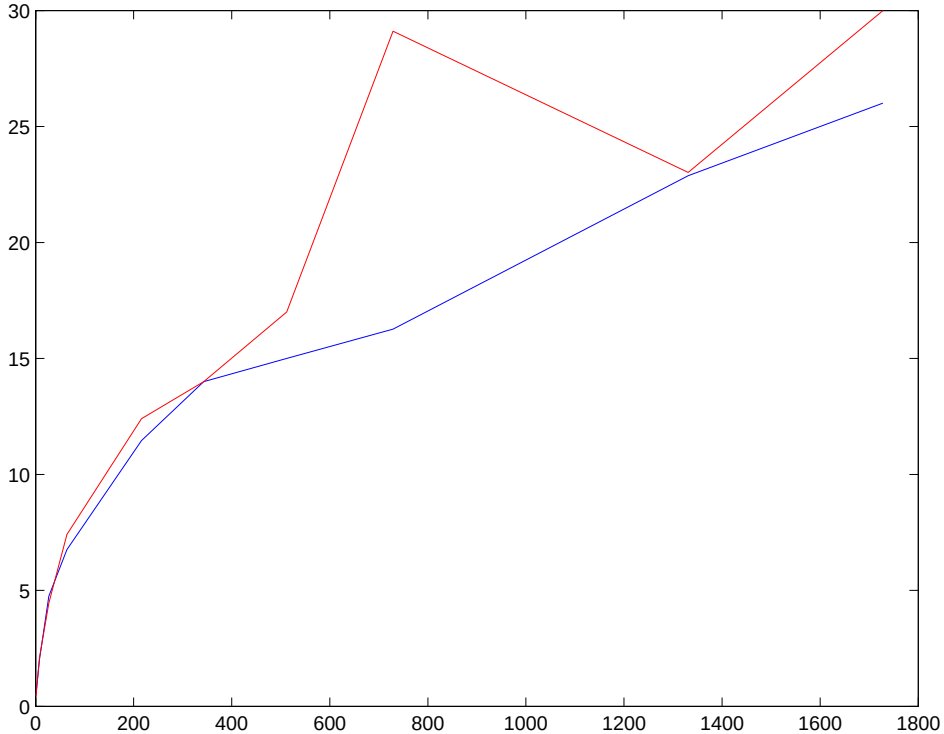
- Largest problem is $\simeq 47\ 10^6$ unknowns



Nb of iterations for the 3D Poisson equation as a function of p

coarsening: LLNL

blue: SGSJ, red: ICp



Elapsed time (s) for the 3D Poisson equation as a function of p

coarsening: LLNL

blue: SGSJ, red: ICp

Extension to block structure

Aim: solve linear systems arising from systems of PDEs involving several unknown functions

Example: three–temperature model of radiation transport

$$\rho \frac{\partial E_i(T_i)}{\partial t} = \mathrm{div}(K_i \nabla T_i) + \alpha(T_e - T_i),$$

$$\rho \frac{\partial E_e(T_e)}{\partial t} = \mathrm{div}(K_e \nabla T_e) - \alpha(T_e - T_i) - c(a\sigma_E T_e^4 - \sigma_A T_r^4),$$

$$\rho \frac{\partial a T_r^4}{\partial t} = \mathrm{div}(K_r \nabla a T_r^4) + c(a\sigma_E T_e^4 - \sigma_A T_r^4).$$

Unknowns are T_e, T_i, T_r (ρ is given)

Nonlinear system of PDES whose behavior depends on relation between diffusion and exchange terms

Using finite volumes, this gives a (non)linear system with $3 \times N$ unknowns linearized with Newton's method

One can solve the linear systems by standard (point) multilevel preconditioners

However, in some cases results are not so good

As we say in French "it is not good to interpolate carrots with turnips"

This motivates the derivation of a block extension of the algebraic multilevel preconditioner

In this example blocks are 3×3

SLOOP data structures have been extended to handle block matrices (blocks are stored as dense matrices)

$$A = \begin{pmatrix} A_{1,1} & \cdots & A_{1,q} \\ \vdots & & \vdots \\ A_{q,1} & \cdots & A_{q,q} \end{pmatrix}$$

where $A_{i,j}$ are $p \times p$ matrices

Typically $p = 2, 3$

Smoothers

◦ (symmetric) block Gauss–Seidel/Jacobi iterations

small $p \times p$ systems are solved with Gaussian elimination

◦ coming soon: block ILU

Influence matrix

Defines influences between blocks:

block I depends on block J (J influences I) if

$$\|A_{I,J}\|_F \geq \tau \max_{K \neq I} \|A_{I,K}\|_F$$

This gives the influence matrix S of order q

Coarsening

We coarsen the (block) graph of A using S with the same algorithms as for the point case

Interpolation P

Interpolation proceeds component by component:

$$e_I = \sum_{J \in C_I} W_{I,J} e_J,$$

C_I coarse nodes influencing I, $W_{I,J}$ $p \times p$ diagonal matrix, no coupling between different kinds of unknowns (carrots with carrots, turnips with turnips)

Use the same formulas as for point methods

$$R = P^T$$

$$A_{\mathsf{C}oarse} = RAP$$

It couples the unknowns

Preliminary results (sequential computations)

Small linear systems arising from the (T_i, T_e, T_r) model of radiation transport

3×3 blocks

Diagonal blocks have non diagonal entries (local coupling)

Non diagonal blocks are diagonal (discretization of diffusion operator)

Krylov solver: BiCGStab, $N = 4120$

Stopping criteria: $\|r^k\| \leq 10^{-12}\|r^0\|$

Number of iterations: point 8, block 7

Must be better when coupling terms are larger

Initialization time of block method is smaller than for point one

Conclusions

- Extension of multilevel preconditioner to block structure is working
- Needs more optimization (coarsest level solvers)
- Testing in transport codes is about to start
- Next step: parallelization of the algorithm