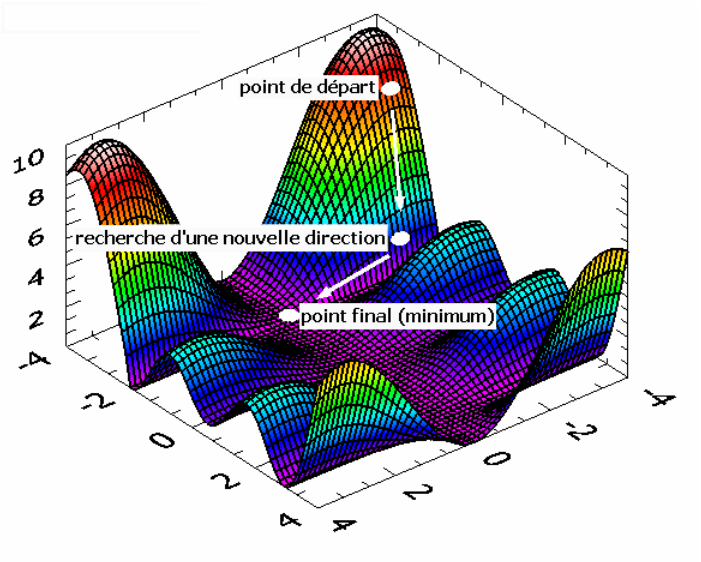
Institut national des sciences appliquées de Rouen

INSA de Rouen

Projet MMSN GM3 - Vague 3 - Sujet 4

Résolution de système linéaire par la méthode du gradient conjugué



Auteurs :
Thibaut André-Gallis
thibaut.andregallis@insa-rouen.fr
Kévin Gatel
kevin.gatel@insa- rouen.fr

Enseignant :
Bernard Gleyse
bernard.gleyse@insa-rouen.fr

4 Janvier 2021

Table des matières

Table des figures

Introduction

La méthode présentée dans ce rapport est celle du gradient conjugué. Il ne s'agit non seulement d'une des techniques les plus utiles pour résoudre des grands systèmes linéaires, mais elle peut même être adaptée de telle manière à ce qu'elle résout des problèmes d'optimisation non-linéaires. Ces deux variantes, reposant sur la même idée de base, sont respectivement appelées méthodes du gradient conjugué linéaire et non-linéaire. Dans la suite nous nous intéresserons uniquement à la méthode du gradient conjugué linéaire.

La méthode a été trouvée dans les années 50 par Magnus Hestenes et Eduard Stiefel, deux mathématiciens. Cette dernière se base sur la recherche de directions successives permettant d'atteindre la solution exacte d'un système linéaire de matrice symétrique et définie positive et représente une alternative à l'algorithme d' élimination de Gauss. Elle est même souvent préférée à cette dernière lorsque les systèmes d'équations sont de grandes tailles.

Les résultats obtenus ont été calculés avec deux machines différentes afin d'observer et de discuter des éventuelles différences. Les caractéristiques des deux ordinateurs sont indiquées dans le fichier "README".

1. Présentation du problème

1.1 Principe

La méthode du gradient conjugué linéaire est une méthode qui résout deux problèmes équivalents possédant la même solution unique. Ces problèmes sont le système d'équations linéaires

$$Ax = b$$

et le problème de minimisation suivant :

$$J(x) = (Ax, x) - 2(b, x)$$

où A est une matrice carrée symétrique définie positive de taille n, x et b deux vecteurs de taille n et (.,.) représente le produit scalaire dans $\mathbb{R}^n$.

1.2 Résolution mathématique

1.2.1 Choix de la fonctionnelle à minimiser

La solution $\overline{x}$ du problème $Ax = b$ est le vecteur pour lequel $J(x)$ atteint son minimum. On a l'expression :

$$J(\overline{x}) = -(b, A^{-1}b).$$

Posons

$$g(x) = 2(Ax - b) = -2r(x)$$

où $r(x) = b - Ax = A\overline{x} - Ax$ est le vecteur résidu du système $Ax = b$.

Si on pose $\overline{x} - x = e(x)$, on a :

$$E(x) = (Ae(x), e(x))$$

.

Il est équivalent de minimiser J ou E comme définies ci-dessus.

Puisque A est symétrique et définie positive, alors (Ax, y) est un produit scalaire et $E(x) = \|e(x)\|_A^2$, avec $\|e\|_A^2 = (Ae, e)^{\frac{1}{2}}$ norme associée à ce produit scalaire. Le minimum de E est nul et est atteint en $\overline{x}$.

$E(x)$ peut aussi s'exprimer en fonction du résidu $r(x) = A\overline{x} - Ax$:

$$E(x) = (r(x), A^{-1}r(x)).$$

Pour minimiser la fonctionnelle E, les méthodes de descente comme celle du gradient conjugué donnent x_{k+1} à partir de x_k en choisissant à la $(k+1)^{me}$ itération une direction de descente $p_k \neq 0$ (un vecteur de $\mathbb{R}^n$) et un scalaire α_k avec

$$x_{k+1} = x_k + \alpha_k p_k$$

de manière à ce que $E(x_k + 1) < E(x_k)$.

1.2.2 Choix optimal de α_k dans une direction fixée p_k

On suppose la direction p_k fixée.

Le choix local optimal de α_k est obtenu lorsqu'à chaque itération, on minimise $E(x_{k+1})$. dans la direction p_k :

$$E(x_k + \alpha_k p_k) = \min_{\alpha \in \mathbb{R}} E(x_k + \alpha p_k)$$

Son minimum est atteint pour

$$\alpha_k = \frac{(r_k, p_k)}{(Ap_k, p_k)}.$$

Lemme 1.1. *$\forall p_k \neq 0$, pour α_k optimal local, on a la relation suivante valable pour $k \geq 0$:*

$$\frac{(r_k, p_k)^2}{(Ap_k, pk)(A^{-1}r_k, r_k)} \geq \frac{1}{cond(A)} \left(\frac{r_k}{\|r_k\|_2}, \frac{p_k}{\|p_k\|_2} \right)^2$$

Ce lemme permet notamment le choix des directions de descente.

Théorème 1.2. *Pour α_k optimal local, toute direction p_k qui vérifie $\forall k \geq 0$:*

$$\left(\frac{r_k}{\|r_k\|_2}, \frac{p_k}{\|p_k\|_2} \right)^2 > 0$$

Ce théorème implique que la suite $(x_k)_{k \geq 0}$ converge vers la solution $\overline{x}$ qui minimise $E(x)$.

1.2.3 Méthode du gradient conjugué

Introduction

Recopier en gros le cours d'andré draux à partir de son introduction 2.3.1 page 44, toutes les propriétés sans démonstrations ni définitions + dire la définition 2.3.5 + rappeler l'inégalité du conditionnement + rappeler la formule de l'erreur relative + rappeler la complexité de l'algorithme

On prend les α_k = minimum local, alors $(p_{k-1}, r_k) = 0$. On cherche les p_k dans le plan (r_k, p_{k-1}).

on pose $p_k = r_k + \beta_k p_{k-1}$

$\beta_k$sera déterminé de telle façon que le facteur de réduction de l'erreur soit maximal. On a $E(x_{k+1}) = E(x_k) * (1 - \gamma_k)$. On choisit $\beta_k$pour que $\gamma_k$soit maximum.

comme $(r_k, p_k) = (r_k, r_k) + \beta_k(r_k, p_{k-1}) = ||r_k||_2^2$ (on prend $p_0 =$r$_0 (\beta_0 = 0)$ pour que la relation soit vraie $\forall k \geq 0$), γ_k sera maximum, si (Ap_k, p_k) est minimum.

$$\begin{aligned}(Ap_k, p_k) &= (A(r_k + \beta_k p_{k-1}), r_k + \beta_k p_{k-1}) \\ &= \beta_k^2 (Ap_{k-1}, p_{k-1}) + 2\beta_k (Ap_{k-1}, r_k) + (Ar_k, r_k).\end{aligned}$$

le trinôme est donc minimal pour $\beta_k = -\frac{(Ap_{k-1}, r_k)}{(Ap_{k-1}, p_{k-1})}$.

Cette valeur de $\beta_k$correspond aussi au point d'annulation de la dérivée. On obtient donc :

$$\beta_k (Ap_{k-1}, p_{k-1}) + (Ap_{k-1}, r_k) = (Ap_{k-1}, r_k + \beta_k p_{k-1}) = (Ap_{k-1}, p_k) = 0$$

Comme A est symétrique définie positive, (Au,v) est un produit scalaire $(u, v)_A$. Par conséquent deux vecteurs A-conjugués sont orthogonaux pour ce produit scalaire.

Propriété 1.2.1. *Si $r_i \neq 0$ pour i=0,...,k alors :*

i) $(r_{k+1}, r_k) = 0$ *pour* $k \geq 0$

ii) $\beta_0 = 0, \beta_k = \frac{||r_k||^2}{||r_{k-1}||^2}$ *pour* $k \geq 1$

Remarque 1.2.1. *comme* $2r_k = -g_k$,*où* $-g_k$ *est le gradient de la fonctionelle, alors :*

$$\beta_k = \frac{||g_k||^2}{||g_{k-1}||^2}$$

L'algorithme

$$\boxed{\begin{array}{l} \text{On initialise :} \begin{cases} x_0 & \\ p_0 = r_0 = b - Ax_0 \end{cases} \quad \text{P} \\ \text{our k=0,1...} \begin{cases} \alpha_k = & \frac{||r_k||^2}{(Ap_k, p_k)} \\ x_{k+1} = & x_k + \alpha_k p_k \\ r_{k+1} = & r_k - \alpha_k A p_k \\ \beta_{k+1} = & \frac{||r_{k+1}||^2}{||r_k||^2} \\ p_{k+1} = & r_{k+1} + \beta_{k+1} p_k \end{cases} \end{array}}$$

Complexité : Si on prend N le nombre d'itérations on évaluera le nombre d'opérations de cet algorithme à environ $2N^3$. Ce qui reste important en comparaison l'algorithme de la méthode de Choleski qui possède un nombre inférieur de $\frac{N^3}{3}$ pour le même type de matrice. La complexité de cet algorithme est de O(n).

Une solution à cela serait de préconditionner la matrice afin de réduire considérablement le nombres d'itérations et de cette façon on aurait alors l'une des méthodes les mieux adapté à la résolution de systèmes linéaires dont la matrice est symétrique définie positive.

Définition 1.2.1. *L'espace vectoriel* $k_k = V(r_0, ..., A^{k-1}r_0)$*est appelé espace de Krylov*

Les $r_i = 0, ..., k-1$ *forment une base orthogonale de cet espace.*

Théorème 1.3. *L'inégalité du conditionnement :*

$$\frac{||\delta X||_q}{||X||_q} \leq cond_q(A) * \frac{||\delta A||_q}{||A||_q}$$

Cette inégalité nous dit que l'erreur relative sur la solution X doit être inférieur au conditionnement de la matrice A multiplié par la norme de la perturbation δA sur la norme de A. Dans notre cas on prendra la norme infini pour q. Cette inégalité n'est présente et analysée uniquement dans le cas de résolution de systèmes avec perturbations

2. Résolution numérique

2.1 Algorithmes et langage utilisés

L'algorithme du gradient conjugué a été implémenté en Fortran. Le choix du langage a été influencé par la facilité d'écrire certaines opérations vectorielles mathématiques en Fortran. L'ensemble des variables sont déclarées en double précision afin d'avoir le plus de chiffres significatifs (précision autour de 10^{-16}). Un algorithme avec et sans perturbation de la matrice A a été implémenté avec un test d'arrêt différent chacun.

$\underline{\mathbf{x}} = \underline{\mathbf{x}}^0$
$\underline{\mathbf{r}} = A\underline{\mathbf{x}} - \underline{\mathbf{b}}$
$\underline{\mathbf{p}} = \underline{\mathbf{r}}$
tant que $\|\underline{\mathbf{r}}\| \geq \varepsilon \|\underline{\mathbf{r}}^0\|$ **faire**
  $\alpha = \dfrac{(\underline{\mathbf{r}}, \underline{\mathbf{r}})}{(A\underline{\mathbf{p}}, \underline{\mathbf{p}})}$
  $\underline{\mathbf{x}} = \underline{\mathbf{x}} - \alpha\underline{\mathbf{p}}$
  $\underline{\mathbf{r}} = \underline{\mathbf{r}} - \alpha A\underline{\mathbf{p}}$
  $\beta = \dfrac{(\underline{\mathbf{r}}, \underline{\mathbf{r}})}{\alpha(A\underline{\mathbf{p}}, \underline{\mathbf{p}})}$
  $\underline{\mathbf{p}} = \underline{\mathbf{r}} + \beta\underline{\mathbf{p}}$
fin

FIGURE 2.1 – Algorithme du gradient conjugué muni du test d'arrêt **t1**

$\underline{\mathbf{x}} = \underline{\mathbf{x}}^0$
$\underline{\mathbf{r}} = A\underline{\mathbf{x}} - \underline{\mathbf{b}}$
$\underline{\mathbf{p}} = \underline{\mathbf{r}}$
tant que $\|\underline{\mathbf{r}}\| \geq \varepsilon$ **faire**
  $\alpha = \dfrac{(\underline{\mathbf{r}}, \underline{\mathbf{r}})}{(A\underline{\mathbf{p}}, \underline{\mathbf{p}})}$
  $\underline{\mathbf{x}} = \underline{\mathbf{x}} - \alpha\underline{\mathbf{p}}$
  $\underline{\mathbf{r}} = \underline{\mathbf{r}} - \alpha A\underline{\mathbf{p}}$
  $\beta = \dfrac{(\underline{\mathbf{r}}, \underline{\mathbf{r}})}{\alpha(A\underline{\mathbf{p}}, \underline{\mathbf{p}})}$
  $\underline{\mathbf{p}} = \underline{\mathbf{r}} + \beta\underline{\mathbf{p}}$
fin

FIGURE 2.2 – Algorithme du gradient conjugué muni du test d'arrêt **t2**

Le vecteur initial x_0 a été initialisé au vecteur nul, la tolérance ε a été fixée à 10^{-10} et le vecteur b est choisi tel que la solution du problème $Ax = b$ soit un vecteur contenant uniquement des 1. En d'autres termes, b_i est la résultante des colonnes de la ligne i de la matrice A.

L'ensemble des matrices testées (à une dimension près) se trouvent en Annexes.

2.2 Résultats obtenus

Convergence des x_n, convergence des résidus, p.s. des résidus qui forment bien une base, inégalité du conditionnement...

2.2.1 Matrices sans perturbation

Les fichiers de résultats sont obtenus en entrant les données dans un fichier de données puis en donnant en entrée de l'exécutable ce fichier de donnée comme ci-dessous :

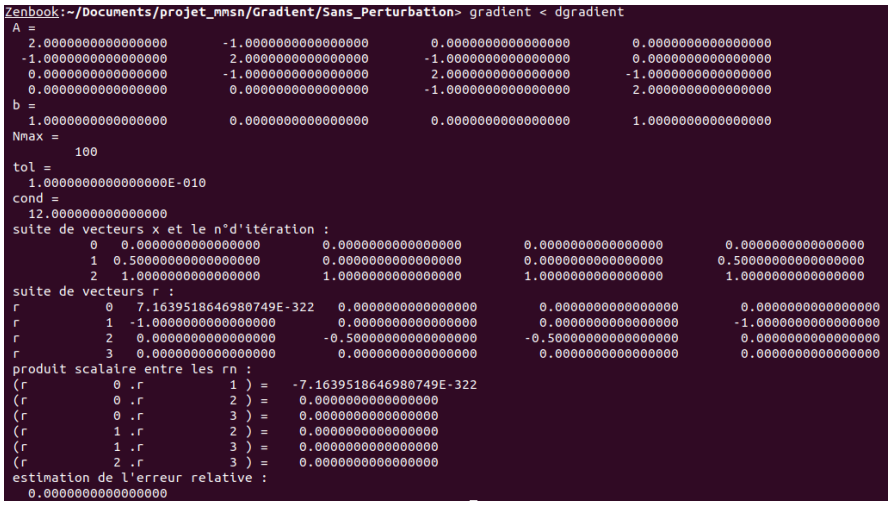


FIGURE 2.3 – Exemple de fichier de résultat avec une matrice dif_4 sans perturbation

Les propriétés des fichiers de résultat entre les deux tests étant similaires, nous décidons de travailler avec le **test d'arrêt 1** dans cette partie.

Matrices dif

En petites dimensions la suite des x_k convergent très rapidement vers la solution, la suite des résidus r_k convergent vers 0 assez rapidement et l'estimation de l'erreur relative est très basse. Les résultats en basse dimensions sont peu intéressants.

En grandes dimensions, nous observons davantage de résultats. En dimension 50, on observe une convergence de la suite x_k plus lente vers la solution.

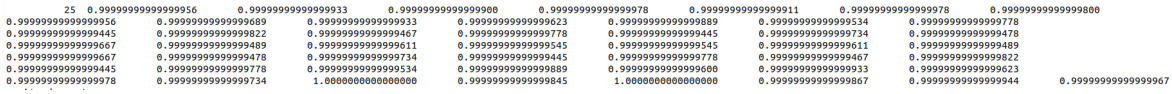


FIGURE 2.4 – Vecteur x_k à la 25^{eme} itération pour dif_{50}

De même pour la suite des résidus r_k qui elle converge à la même vitesse que les x_k.

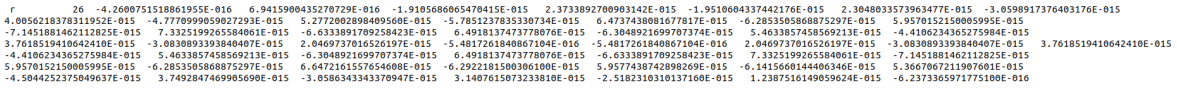


FIGURE 2.5 – Vecteur r_k à la 26^{eme} itération pour dif_{50}

On observe que les composantes du vecteur r_k sont très proche de la précision de la machine qui se trouve autour de 10^{-16}.

Les produits scalaires entre les résidus sont de l'ordre de 10^{-17} qui montrent bien l'espace vectoriel engendré par ces derniers qui forment une base orthogonale.

L'erreur relative est de l'ordre de 10^{-15} ce qui est très faible, et donc positif pour notre cas.

Même en grande dimension, pour cette matrice les convergences sont bien respectées malgré la précision machine.

Matrice elec

La matrice étant de petite taille les x_k convergent très rapidement vers la solution.

```
3  0.99999999999999978       0.99999999999999978       0.99999999999999978
```

FIGURE 2.6 – Vecteur x_k à la 3^{eme} itération pour *elec*

De même pour la suite r_k qui convergent vers le vecteur nul :

```
0   4.0019317313140970E-322   1.3235777704702403       -1.3235777704702398
1  -2.0000000000000000        2.2204460492503131E-016   0.0000000000000000
2  -2.3439490445859872       -2.4904458598726116        1.0254777070063703
3   1.3235777704702403       -1.3235777704702398       -0.18908253863860636
4   2.2204460492503131E-016   0.0000000000000000       -3.3306690738754696E-016
```

FIGURE 2.7 – Vecteurs r_k jusqu'à la 4^{eme} itération pour *elec*

En revanche certains produits scalaires sont éloignés de zéro ne formant pas une base orthogonale.

```
produit scalaire entre les rn :
(r            0 .r            1 ) =    2.9389330313161827E-016
(r            0 .r            2 ) =   -4.6535982757934571
(r            0 .r            3 ) =   -1.5015926695568318
(r            0 .r            4 ) =    4.4083995469742726E-016
(r            1 .r            2 ) =    4.6878980891719735
(r            1 .r            3 ) =   -2.6471555409404810
(r            1 .r            4 ) =   -4.4408920985006262E-016
(r            2 .r            3 ) =   -1.6653345369377348E-015
(r            2 .r            4 ) =   -8.6201392803698488E-016
(r            3 .r            4 ) =    3.5687043951696522E-016
estimation de l'erreur relative :
  2.2204460492503136E-016
```

FIGURE 2.8 – Produits scalaires entre les r_k pour *elec*

Cette observation est due aux vecteurs résidus r_2 et r_3 qui posent des problèmes d'orthogonalité.

On remarque que l'erreur relative est très basse malgré tout.

Matrice elecmodif

Cette matrice n'étant pas définie positive il est inutile de l'étudier en profondeur afin de montrer la cohérence avec la partie mathématique attendue numériquement.

Cependant ses propriétés sont étonnantes malgré sa non définie-positivité. En effet ses x_k convergent vers la solution, ses vecteurs résidus r_k convergent vers le vecteur nul.

On retrouve les mêmes produits scalaires anormaux que la matrice elec causés par les vecteurs r_2 et r_3 :

```
produit scalaire entre les rn :
(r            0 .r            1 ) =    1.0603405902531558E-015
(r            0 .r            2 ) =   -14.144747700160016
(r            0 .r            3 ) =  -0.64509701535473418
(r            0 .r            4 ) =    1.1712940538058922E-015
(r            1 .r            2 ) =   -188.57327818737690
(r            1 .r            3 ) =   -4.0118045221231409
(r            1 .r            4 ) =    4.0500935938325721E-013
(r            2 .r            3 ) =   -4.1211478674085811E-013
(r            2 .r            4 ) =   -3.1284918910098789E-013
(r            3 .r            4 ) =   -7.5602677515395840E-016
estimation de l'erreur relative :
  1.1102230246251569E-015
```

FIGURE 2.9 – Produits scalaires entre les r_k pour *elecmodif*

On remarque que l'erreur relative est également très basse.

Même si la matrice n'est pas définie positive, on observe les mêmes propriétés que sa matrice d'origine elec numériquement.

Matrices de Hilbert

En dimension 6 la suite des x_k et la suite des résidus convergent beaucoup moins rapidement que les matrices précédentes.

```
suite de vecteurs x et le n°d'itération :
          0   0.0000000000000000       0.0000000000000000       0.0000000000000000       0.0000000000000000       0.0000000000000000       0.0000000000000000
          1   1.5210871468871527       0.98892837829689528      0.75610891703865757      0.61814182888562785      0.52501404438233279      0.45728474656175461
          2  0.93489743892528543        1.1699804214389617       1.0947766113800910       0.99240407447023804      0.89838393852941589      0.81736193343418462
          3   1.0037085607344767       0.97162950690639938      1.0271946795309368        1.0292165772692474       1.0020215865016213       0.96223451189827158
          4  0.99989612104522396        1.0016605060834245      0.99453198148151856       1.0035929039545122       1.0048074940229736       0.99541943762847696
          5  0.99992551625939163        1.0010278853413876      0.99663521332752059       1.0020441617657465       1.0032623647250449       0.99701952514430747
          6   1.0000011581990764       0.99996689396291449       1.0002237793324107       0.99941847047305743       1.0006413536266310       0.99974747670367237
          7   1.0000011606244197       0.99996689313915810       1.0002237778062124       0.99941846916395971       1.0006413517907162       0.99974747536908670
suite de vecteurs r :
r           0   1.5069002198158020E-321  -1.5369050243800593E-005  -9.0324062131132285E-006  -6.6275384443997366E-006  -5.1868184528469504E-006  -4.3086303128609755E-006
r           1  -2.4499999999999997       -5.8112640307500518E-010   1.9604419696232951E-010   3.7036643133914074E-010   3.1034021953955460E-010   4.3659581235701235E-010
r           2  0.15334003223998582        6.7177246484060040E-012  -5.8795426730387274E-012   2.0355805934944045E-011  -6.7232164579370554E-011   6.5257147561399808E-011
r           3  -1.1820179993962676E-003   3.2566650697245861E-003   1.1372478250830653E-003  -5.6371249387769662E-004  -1.6548969082078452E-003  -2.3294915443311109E-003
r           4   2.4209653369403148E-006  -1.8703972592999749E-005   2.0041148098081480E-005   1.5817874204312240E-005  -1.6050791141813611E-006  -2.0280365341271249E-005
r           5  -1.0212550176866748E-009   2.1662456459224324E-008  -7.1611362412287154E-008   5.2430003776128889E-008   5.2480022909865512E-008  -5.4127413426814246E-008
r           6  -1.5369050243800593E-005  -9.0324062131132285E-006  -6.6275384443997366E-006  -5.1868184528469504E-006  -4.3086303128609755E-006  -3.7582297935742183E-006
r           7  -5.8112640307500518E-010   1.9604419696232951E-010   3.7036643133914074E-010   3.1034021953955460E-010   4.3659581235701235E-010   3.2333947615648212E-010
r           8   6.7177246484060040E-012  -5.8795426730387274E-012   2.0355805934944045E-011  -6.7232164579370554E-011   6.5257147561399808E-011  -3.1263619815678775E-011
```

FIGURE 2.10 – Convergence des suites x_k et r_k pour H_6

En effet, pour les résidus nous sommes aux alentours de 10^{-12} au lieu de 10^{-16} dans les exemples précédents. La précision tourne autour de 10^{-5} pour le vecteur x_k à partir de la 7^{me} itération.

Cependant les produits scalaires satisfont bien l'orthogonalité et l'erreur relative est environ 10^{-5} ce qui est plutôt élevé par rapport aux exemples précédents.

En dimension plus élevée, nous observons les mêmes propriétés. Une convergence des x_k et des r_k assez lente, des produits scalaires au voisinage de zéro et une erreur relative du même ordre.

Matrice lap

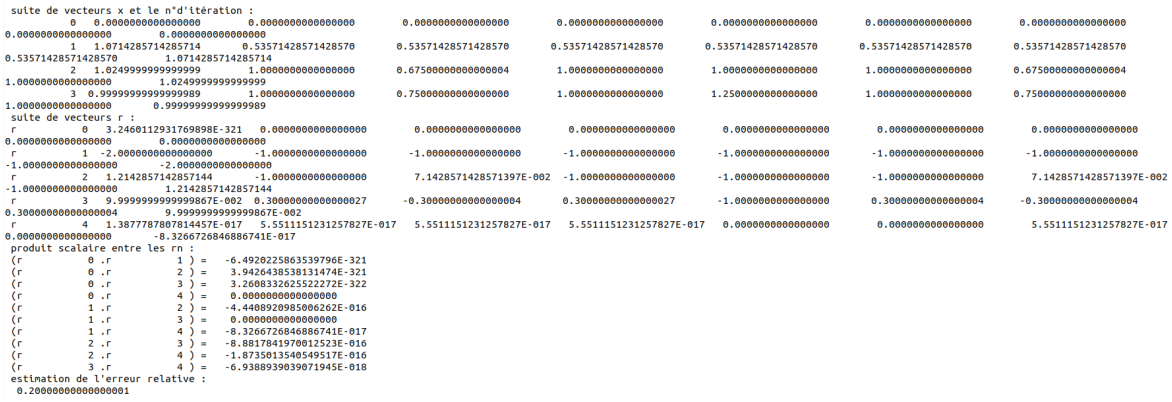


FIGURE 2.11 – Fichier résultat pour lap_3

La suite des résidus r_k convergent très rapidement vers la limite théorique. Cependant, pour la suite des x_k, certaines composantes convergent moins rapidement que d'autres ce qui explique une erreur relative bien au dessus de celles vu précédemment pour les autres matrices.

Les produits scalaires sont tous vérifiés et égals à 0 à la précision machine près. L'espace de Krylov engendré par la base orthogonale des résidus r_k pour cette matrice est bien vérifié numériquement

Matrices tri_{alpha}

Pour ces types de matrice, nous avons décidé de prendre des matrices de dimension 10 avec des α égalent à 2, 5 et 20.

Pour les 3 matrices confondues, la convergence est parfaite pour le vecteur solution x_k l'erreur relative.

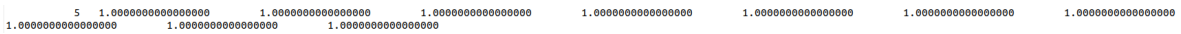


FIGURE 2.12 – Convergence parfaite de la suite x_k à la 5^{me} itération pour les matrices $tri_{alpha_{10}}$

L'erreur relative est donc nulle pour l'ensemble de ces matrices.

Nous observons également une convergence très rapide des résidus et des produits scalaires tous respectés.

```
(r          0 .r          1 ) =   -1.6549716938744235E-319
(r          0 .r          2 ) =   -3.0977915994246158E-321
(r          0 .r          3 ) =    0.0000000000000000
(r          0 .r          4 ) =    0.0000000000000000
(r          0 .r          5 ) =    0.0000000000000000
(r          0 .r          6 ) =    0.0000000000000000
(r          1 .r          2 ) =    1.2931877790833823E-012
(r          1 .r          3 ) =   -4.7073456244106637E-014
(r          1 .r          4 ) =    1.0189874118105013E-015
(r          1 .r          5 ) =   -2.8829189876273739E-017
(r          1 .r          6 ) =    1.5362043941141139E-018
(r          2 .r          3 ) =    4.1524943206194820E-017
(r          2 .r          4 ) =   -3.1405334706229287E-018
(r          2 .r          5 ) =    1.5547653189334323E-019
(r          2 .r          6 ) =   -5.7553470233283642E-021
(r          3 .r          4 ) =    7.7530540215344237E-020
(r          3 .r          5 ) =   -1.9295594602799997E-021
(r          3 .r          6 ) =    2.4974844145580805E-022
(r          4 .r          5 ) =    1.4936241102263203E-022
(r          4 .r          6 ) =   -2.9668407185213662E-024
(r          5 .r          6 ) =    8.0499081584687750E-026
```

FIGURE 2.13 – Produits scalaires entre les résidus pour W

La matrice étant très bien conditionnée, le choix du *alpha* importe peu. Il doit simplement être supérieur à 1 de manière à ce que la matrice soit définie positive.

Matrice de Wilson

Pour cette matrice, certains résidus ont du mal à converger vers zéro. En particulier celui à la deuxième itération :

```
r          0   7.1639518646980749E-322   1.4010923440210227E-003  -2.0823028461624377E-003   5.3544930608207669E-005
r          1  -32.000000000000000       -7.3619282957120002E-010  -5.2962965263469997E-010  -7.6872974222238219E-010
r          2 -0.23747079037800489      -0.15210996563573786       0.11623367697594489       0.23425429553264721
r          3  -1.5724095509345171E-003  -9.7546299276141113E-004   8.0486077895987052E-003  -6.2210097611014037E-003
r          4   1.4010923440210227E-003  -2.0823028461624377E-003   5.3544930608207669E-005   4.1646057185144858E-005
r          5  -7.3619282957120002E-010  -5.2962965263469997E-010  -7.6872974222238219E-010  -7.2549848901648072E-010
```

FIGURE 2.14 – Suite des résidus r_k pour W

Ce manque de convergence pour certains de ces vecteurs impliquent des produits scalaires très éloignés de 0 et donc des résultats numériques ne permettant pas de vérifier les résultats mathématiques théoriques avec précision.

Ce manque de convergence et ce manque de précision sont dus au conditionnement élevé de la matrice carrée de dimension 4.

```
cond =
  4488.0000000000000
```

FIGURE 2.15 – Conditionnement de la matrice de Wilson

Cependant la suite x_k converge assez rapidement vers la solution malgré le conditionnement :

```
4  0.99999999997569455       0.99999999998249978       0.99999999997462652       0.99999999997605027
```

FIGURE 2.16 – Vecteur solution x_k à partir de la 4^{me} itération pour W

Enfin l'estimation de l'erreur relative est 10^{-11} ce qui est assez faible.

2.2.2 Matrices avec perturbation

De même Les fichiers de résultats sont obtenus en entrant les données dans un fichier de données puis en donnant en entrée de l'exécutable ce fichier de donnée qui contient en plus la matrice de perturbation δA comme ci-dessous :

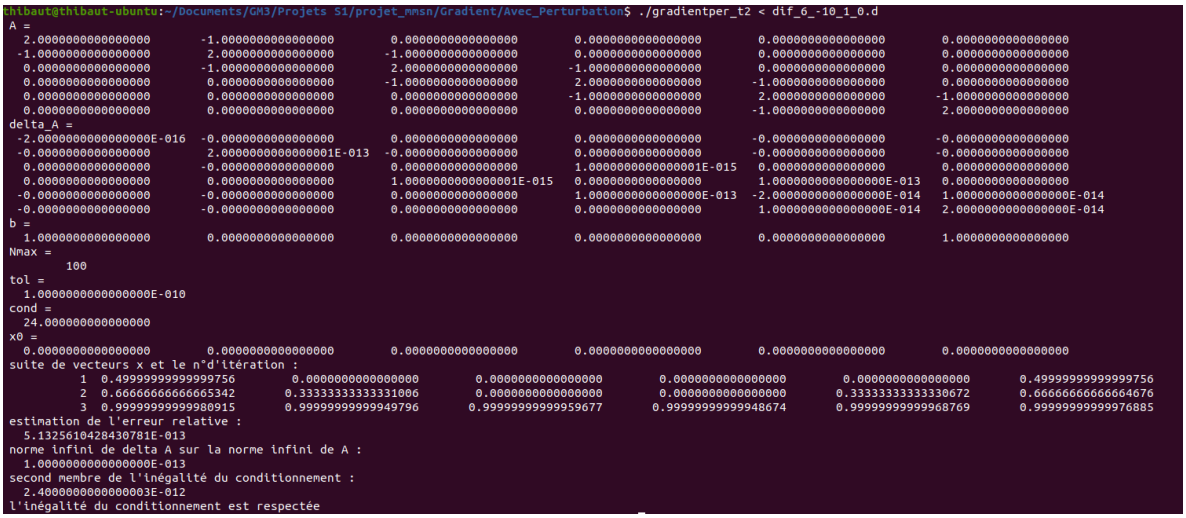


FIGURE 2.17 – Exemple de fichier de résultat avec une matrice dif_6 avec perturbation

les coefficients de δA sont formés de la façon suivante :

$$\delta_{ij} = A_{ij} * 10^{-n_{ij}} * \epsilon_{ij}$$

avec : $n_{ij} \in \{13; 14; 15\}$ $\epsilon_{ij} \in \{-1; 0; 1\}$
Les propriétés des fichiers de résultat entre les deux tests étant similaires, nous décidons de travailler avec le **test d'arrêt 2** dans cette partie.

Matrices dif

En petites dimensions la suite des x_k converge très rapidement vers la solution malgré les perturbations. On notera que pour ces matrices le conditionnements est très faible en petite dimension.
Plus intéressant observons les résultats obtenus en dimension 50 :

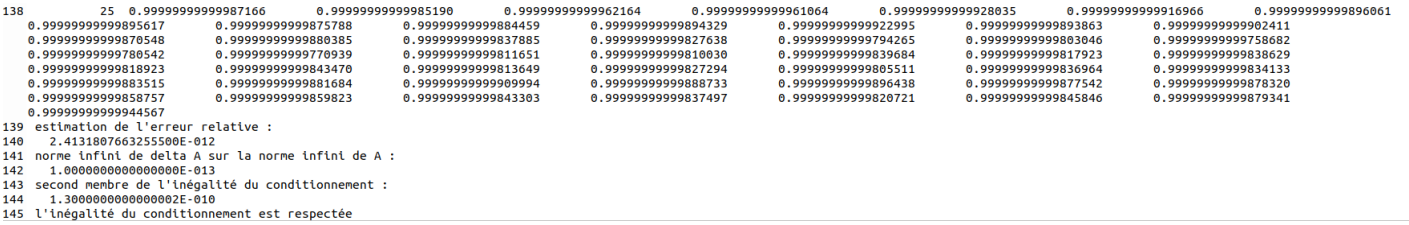


FIGURE 2.18 – Résultats obtenus pour la matrice dif en dimension 50

On observe que la suite des x_k converge au bout de la 25^{eme} itération. L'erreur relative reste très faible et l'inégalité du conditionnement est respectée.
On voit donc que même en grande dimension on converge bien et tout de même rapidement vers la solution.

Matrice elec

la matrice est petite de dimension et son conditionnement assez faible. La suite des x_k converge en seulement 3 itérations.

```
A =
   13.000000000000000       -8.000000000000000       -3.000000000000000
   -8.000000000000000        10.000000000000000       -1.000000000000000
   -3.000000000000000       -1.000000000000000        11.000000000000000
delta_A =
   -1.3000000000000001E-014  -8.000000000000006E-015  -3.000000000000003E-013
   -8.000000000000006E-015  -1.000000000000002E-014  -0.000000000000000
   -3.000000000000003E-013  -0.000000000000000        1.100000000000000E-013
b =
   2.000000000000000        1.000000000000000        7.000000000000000
Nmax =
         100
tol =
   1.000000000000000E-010
cond =
   10.936000000000000
x0 =
   0.000000000000000        0.000000000000000        0.000000000000000
suite de vecteurs x et le n°d'itération :
           1   0.22929936305732637       0.11464968152866319       0.80254777070064232
           2    1.0289378493917170       0.88885235120025841       0.98059848733972221
           3    1.0000000000000768        1.0000000000000677        1.0000000000000442
estimation de l'erreur relative :
   7.6827433304054926E-014
norme infini de delta A sur la norme infini de A :
   2.3076923076923079E-014
second membre de l'inégalité du conditionnement :
   2.5236923076923078E-013
l'inégalité du conditionnement est respectée
```

FIGURE 2.19 – Résultats obtenus pour la matrice elec

L'erreur relative est toujours très faible et l'inégalité du conditionnement est respectée, cette matrice ne pose donc pas de problème malgré les perturbations.

Matrice elecmodif

Cette matrice reste la même que la précédente mais on a mis un signe moins sur le premier élément de la matrice. Elle reste donc symétrique mais elle n'est plus définie positive car son premier mineur est négatif.

```
A =
 -13.000000000000000       -8.000000000000000       -3.000000000000000
 -8.000000000000000        10.000000000000000       -1.000000000000000
 -3.000000000000000        -1.000000000000000        11.000000000000000
delta_A =
  1.3000000000000001E-014   8.0000000000000000E-014  -3.0000000000000002E-015
  8.0000000000000000E-014   0.0000000000000000        1.0000000000000001E-015
 -3.0000000000000002E-015   1.0000000000000001E-015   0.0000000000000000
b =
 -24.000000000000000        1.0000000000000000        7.0000000000000000
Nmax =
        100
tol =
  1.0000000000000000E-010
cond =
  2.6985000000000001
x0 =
  0.0000000000000000        0.0000000000000000        0.0000000000000000
suite de vecteurs x et le n°d'itération :
          1   2.7016723610861373      -0.11256968171192239      -0.78798777198345671
          2  0.98891837250978543        1.0611889996189439       0.94056922739215487
          3   1.0000000000000087       0.99999999999999822       1.0000000000000020
estimation de l'erreur relative :
  8.6597395920761453E-015
norme infini de delta A sur la norme infini de A :
  6.1538461538461540E-015
second membre de l'inégalité du conditionnement :
  1.6606153846153847E-014
l'inégalité du conditionnement est respectée
```

FIGURE 2.20 – Résultats obtenus pour la matrice elecmodif

Malgré cela on observe de bon résultats notamment avec l'erreur relative faible et la suite des x_k qui converge aussi vite. L'inégalité du conditionnement est elle aussi respectée.

Matrices de Hilberts

Ces Matrices sont déjà plus compliqués à utiliser car elles possèdent un grand conditionnement. De plus en y ajoutant des perturbations cela n'arrange pas les choses. En dimension 4 cela se passe encore plutôt bien mais au-delà on observe des complications.

```
| A =
   1.0000000000000000       0.50000000000000000       0.33333333333333331       0.25000000000000000       0.20000000000000001       0.16666666666666666
  0.50000000000000000       0.33333333333333331       0.25000000000000000       0.20000000000000001       0.16666666666666666       0.14285714285714285
  0.33333333333333331       0.25000000000000000       0.20000000000000001       0.16666666666666666       0.14285714285714285       0.12500000000000000
  0.25000000000000000       0.20000000000000001       0.16666666666666666       0.14285714285714285       0.12500000000000000       0.11111111111111110
  0.20000000000000001       0.16666666666666666       0.14285714285714285       0.12500000000000000       0.11111111111111110       0.10000000000000001
  0.16666666666666666       0.14285714285714285       0.12500000000000000       0.11111111111111110       0.10000000000000001       9.0909090909090912E-002
delta_A =
 -1.0000000000000000E-014  -5.0000000000000002E-014   0.0000000000000000       -2.5000000000000002E-016   0.0000000000000000       -1.6666666666666664E-017
 -5.0000000000000002E-014   3.3333333333333329E-017   2.4999999999999999E-017   2.0000000000000002E-015  -1.6666666666666667E-014  -1.4285714285714284E-017
  0.0000000000000000        2.4999999999999999E-017   0.0000000000000000       -1.6666666666666667E-014  -1.4285714285714284E-017  -1.2500000000000001E-016
 -2.5000000000000002E-016   2.0000000000000002E-015  -1.6666666666666667E-014  -1.4285714285714284E-014   0.0000000000000000        0.0000000000000000
  0.0000000000000000       -1.6666666666666667E-014  -1.4285714285714284E-017   0.0000000000000000       -1.1111111111111112E-016  -1.0000000000000001E-015
 -1.6666666666666664E-017  -1.4285714285714284E-017  -1.2500000000000001E-016   0.0000000000000000       -1.0000000000000001E-015   9.0909090909090923E-017
b =
   2.4499999999999997        1.5928571428571427        1.2178571428571427        0.99563492063492065       0.84563492063492063       0.73654401154401161
Nmax =
         100
tol =
   1.0000000000000000E-010
cond =
   29070278.999740001
x0 =
   0.0000000000000000        0.0000000000000000        0.0000000000000000        0.0000000000000000        0.0000000000000000        0.0000000000000000
suite de vecteurs x et le n°d'itération :
           1   1.5210871468871943        0.98892837829692226       0.75610891703867822       0.61814182888564484       0.52501404438234722       0.45728474656176715
           2  0.93489743892529809         1.1699804214391549        1.0947766113800248       0.99240407447023415       0.89838393852941323       0.81736193343412600
           3   1.0037085607339182        0.97162950690780159        1.0271946795300804        1.0292165772696154        1.0020215865015498        0.96223451189710729
           4  0.99989612156088414         1.0016605064021324       0.99453198167278134        1.0035929041463219        1.0048074941785961        0.99541943774642183
           5  0.99989667666632553         1.0015845747150165       0.99477752352895299        1.0034102001373364        1.0046253795213433        0.99560648191018986
           6   1.0000011601291872       0.99996689361097657        1.0002237770747548       0.99941847018778440       1.0006413528577713       0.99974747488076987
           7   1.0000011607221204       0.99996689342650569        1.0002237765897999       0.99941847014994789       1.0006413521025999       0.99974747467269731
estimation de l'erreur relative :
   6.4094103372023799E-004
norme infini de delta A sur la norme infini de A :
   5.0000000000000002E-014
second membre de l'inégalité du conditionnement :
   1.4535139499870001E-006
l'inégalité du conditionnement n'est pas respectée
```

FIGURE 2.21 – Résultats obtenus pour la matrice de Hilbert en dimension 6

On observe bien une lente convergences des x_k et une erreur relative élevée. La même chose se produit en dimension supérieur mais en dimension 6 on se rend compte que l'inégalité du conditionnement n'est pas respectée étonnamment. En effet l'erreur relative est supérieure au membre de droite de l'inégalité. La différence est à 10^{-2} près.
Cela est dû au conditionnement qui est plutôt élevé et qui rend compliqué les calculs en flottants.

Matrice lap

On prendra la matrice lap en dimension 9 :

```
 suite de vecteurs x et le n°d'itération :
           1   1.0714285714285983        0.53571428571429913       0.53571428571429913       0.53571428571429913       0.53571428571429913       0.53571428571429913
0.53571428571429913        1.0714285714285983
           2   1.0250000000000379        1.0000000000000850        0.67500000000003912       1.0000000000000788        1.0000000000001439        1.0000000000000822        0.67500000000006122
1.0000000000000919        1.0250000000000365
           3   1.0000000000000409        1.0000000000000859        0.75000000000004952       1.0000000000000806        1.2500000000002067        1.0000000000000793        0.75000000000008193
1.0000000000000961        1.0000000000000415
 estimation de l'erreur relative :
  0.20000000000013229
 norme infini de delta A sur la norme infini de A :
   1.0000000000000000E-013
 second membre de l'inégalité du conditionnement :
   9.0000000000000000E-013
 l'inégalité du conditionnement n'est pas respectée
```

FIGURE 2.22 – Résultats obtenus pour la matrice lap

La solution converge très vite mais certaines coordonnées des x_k convergent mal ce qui entraîne une grosse erreur relative. De ce fait l'inégalité du conditionnement n'est une nouvelle fois pas respectée.

Matrices tri_{alpha}

Comme précédemment nous avons pris cette matrice pour des valeurs de α telles que 2,5 et 20 en dimension 10.
Pour les 3 matrices les résultats sont excellent malgré les perturbations.
Regardons plus en détail pour α=5

```
 suite de vecteurs x et le n°d'itération :
          1  0.93023537976219639       1.0148022324678505       1.0148022324678505       1.0148022324678505       1.0148022324678505       1.0148022324678505
1.0148022324678505       1.0148022324678505       1.0148022324678505       0.93023537976219639
          2  0.99871868935658181       1.0065611187590731       0.99818999980549472      0.99818999980547463      0.99818999980547374      0.99818999980546586      0.99818999980546463
0.99818999980547463       1.0065611187589634       0.99871868935648156
          3  0.99999823639562329       1.0001192005270914       0.99939722988061086      1.0002496695110190       1.0002496695110210       1.0002496695110132       1.0002496695110119
0.99939722988060276       1.0001192005269928       0.99998236395613427
          4  0.99999981775656266       1.0000013838306174       0.99999072997680261      1.0000478946208942       0.99996015558242468      0.99996015558241669      1.0000478946208857
0.99999072997679317       1.0000013838305191       0.99999981775646352
          5  0.99999999999999811       1.0000000000001081       1.0000000000000093       0.99999999999999900      0.99999999999999989      0.99999999999999201      0.99999999999999079
0.99999999999999989       1.0000000000000098       0.99999999999989897
 estimation de l'erreur relative :
   1.0813572259847856E-013
 norme infini de delta A sur la norme infini de A :
   1.0000000000000000E-013
 second membre de l'inégalité du conditionnement :
   1.5000000000000002E-013
 l'inégalité du conditionnement est respectée
```

FIGURE 2.23 – Résultats obtenus pour la matrice tri_5

On voit que la suite converge en 5 itérations, que l'erreur relative est très faible et que l'inégalité du conditionnement est respectée.
On pouvait s'attendre à de telles résultats en observant la valeur du conditionnement qui vaut 1,5

Matrice de Wilson

Le conditionnement de cette matrice reste assez élevé mais malgré tout on obtient des résultats plutôt bon :

```
A =
   10.000000000000000        7.0000000000000000        8.0000000000000000        7.0000000000000000
   7.0000000000000000        5.0000000000000000        6.0000000000000000        5.0000000000000000
   8.0000000000000000        6.0000000000000000        10.000000000000000        9.0000000000000000
   7.0000000000000000        5.0000000000000000        9.0000000000000000        10.000000000000000
delta_A =
  -1.0000000000000001E-015  -7.0000000000000005E-013  -8.0000000000000000E-014   7.0000000000000005E-014
  -7.0000000000000005E-013   5.0000000000000008E-015   0.0000000000000000        0.0000000000000000
  -8.0000000000000000E-014   0.0000000000000000        0.0000000000000000       -8.9999999999999995E-014
   7.0000000000000005E-014   0.0000000000000000       -8.9999999999999995E-014   0.0000000000000000
b =
   32.000000000000000        23.000000000000000        33.000000000000000        31.000000000000000
Nmax =
         100
tol =
   1.0000000000000000E-010
cond =
   4488.0000000000000
x0 =
   0.0000000000000000        0.0000000000000000        0.0000000000000000        0.0000000000000000
suite de vecteurs x et le n°d'itération :
           1    1.0565498281787062       0.75939518900344516       1.0895670103092907       1.0235326460481218
           2    1.1184958803925191       0.79910087804648344       1.0597648435988547       0.96309198448691369
           3    1.1206872982560552       0.80046481710810169       1.0495528583061984       0.97069307479687206
           4   0.99999999995275235       0.99999999998809541       0.99999999995629207       0.99999999996468247
           5   0.99999999999083278        1.0000000000154914       0.99999999999605982        1.0000000000022156
estimation de l'erreur relative :
   1.5491385951564677E-011
norme infini de delta A sur la norme infini de A :
   7.0000000000000005E-014
second membre de l'inégalité du conditionnement :
   3.1416000000000003E-010
l'inégalité du conditionnement est respectée
```

FIGURE 2.24 – Résultats obtenus pour la matrice de Wilson

On peut voir que malgré tout la suite converge assez vite et que l'inégalité du conditionnement est respectée.

Conclusion

Pour conclure on a pu observer que dans la grande majorité des cas l'algorithme du gradient conjugué donnait bien les résultats attendus théoriquement. La suite des x_k converge bien souvent rapidement vers la solution avec un nombre d'itérations inférieur à la dimension de la matrice A.

En ce qui concerne les matrices sans perturbations la principale difficulté qui ressort est la convergence des vecteurs r_k qui n'est pas toujours parfaite. On observe alors des produits scalaires entre des vecteurs r_k qui sont non nuls, alors que théoriquement ils forment une bas orthogonale de l'espace de Krylov et donc leur produit scalaire devrait être toujours nuls. Le problème arrive souvent lorsque les matrices sont mal conditionnées. Néanmoins la majorité des produits scalaires approches la précision machine qui est de 10^{-16}. Le fait que le calcul soit fait en flottant et non en réel peut expliquer ces légères différences avec ce qu'on attend théoriquement.

Pour les matrices avec perturbations, généralement cela ne pose pas de problèmes lorsque la matrice est bien conditionnée. On observe parfois un ralentissement dans la convergence des x_k mais rien de bien grave. Cependant quand le conditionnement devient élevé il arrive que l'inégalité du conditionnement ne soit pas respectée à la fin des itérations. La différence entre les deux membres de l'inégalité est généralement assez faible tout de même et l'utilisation des flottants à la place des réels est peut être responsable de cette erreur parfois.

La complexité de l'algorithme du gradient conjugué étant de O(n) cela en fait un très bon choix pour des problèmes d'optimisation de grande taille.
Associé avec un préconditionnement de la matrice A l'algorithme du gradient conjugué donne donc d'excellent résultats rapidement dans le cas où la matrice A est symétrique définie positive.

Ce projet nous aura permis de découvrir plus en détail une méthode de résolutions de système linéaire qu'est le gradient conjugué en la mettant en pratique à l'aide d'un ordinateur. Nous avons pu observer et analyser les résultats numériques obtenues et nous rendre compte des difficultés auxquelles nous pouvons faire face lors de l'utilisation d'un tel algorithme dans un problème d'optimisation.

Annexes

```
13.0 -8.0 -3.0
-8.0 10.0 -1.0
-3.0 -1.0 11.0
```

Matrice elec

```
-13.0 -8.0 -3.0
-8.0 10.0 -1.0
-3.0 -1.0 11.0
```

Matrice elecmodif

```
2.0 -1.0 0.0 0.0 0.0 0.0 0.0 0.0
-1.0 2.0 -1.0 0.0 0.0 0.0 0.0 0.0
0.0 -1.0 2.0 -1.0 0.0 0.0 0.0 0.0
0.0 0.0 -1.0 2.0 -1.0 0.0 0.0 0.0
0.0 0.0 0.0 -1.0 2.0 -1.0 0.0 0.0
0.0 0.0 0.0 0.0 -1.0 2.0 -1.0 0.0
0.0 0.0 0.0 0.0 0.0 -1.0 2.0 -1.0
0.0 0.0 0.0 0.0 0.0 0.0 -1.0 2.0
```

Matrice dif de dim 8

$$\begin{pmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} \\ \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} \\ \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} \end{pmatrix}.$$

Matrice de Hilbert de dim 5

```
 4 -1  0 -1  0  0  0  0  0
-1  4 -1  0 -1  0  0  0  0
 0 -1  4  0  0 -1  0  0  0
-1  0  0  4 -1  0 -1  0  0
 0 -1  0 -1  4 -1  0 -1  0
 0  0 -1  0 -1  4  0  0 -1
 0  0  0 -1  0  0  4 -1  0
 0  0  0  0 -1  0 -1  4 -1
 0  0  0  0  0 -1  0 -1  4
```

Matrice Laplacienne_3 (de dim 3^2)

```
10 1 0 0 0 0 0 0 0 0
1 10 1 0 0 0 0 0 0 0
0 1 10 1 0 0 0 0 0 0
0 0 1 10 1 0 0 0 0 0
0 0 0 1 10 1 0 0 0 0
0 0 0 0 1 10 1 0 0 0
0 0 0 0 0 1 10 1 0 0
0 0 0 0 0 0 1 10 1 0
0 0 0 0 0 0 0 1 10 1
0 0 0 0 0 0 0 0 1 10
```

Matrice tri_α de dim 10 avec $\alpha = 5$

```
10 7 8 7
7 5 6 5
8 6 10 9
7 5 9 10
```

Matrice de Wilson

Bibliographie

[1] André Draux *Analyse numérique*, poly, chapitre 2 *Les méthodes de descente.*

[2] Maria Kazakova *GM3 Analyse numérique I*, Année 2020-2021, section 1.2.4

[3] Daniel Kauth *Les méthodes de Krylov Optimisation numérique Méthodes du gradient conjugué linéaire*, chapitre 5.1, 5 novembre 2009.