Cours : Résoudre $Ax = b$ en utilisant les Gradients Conjugués

Basé sur la référence suivante (disponible en ligne)

Shewchuk, J. R. (1994). An introduction to the conjugate gradient method without the agonizing pain.

Minimisation d'une forme quadratique

Forme quadratique

Soit $A \in \mathbb{R}^{n \times n}$ une matrice réelle symétrique définie positive (SDP) et $x \in \mathbb{R}^n \neq 0$. On a par définition du caractère SDP de A :

$$x^T A x > 0.$$

Note : La matrice A que vous avez calculé dans le devoir 4 qui correspond à l'opérateur Laplacien muni de conditions aux limites de Dirichlet est SDP. On peut le prouver en utilisant la propriété de coercivité de la forme bilinéaire correspondant au problème variationnel continu

$$a(u,u) = \int_\Omega |\nabla u|^2 \, d\Omega \;\; \geq \gamma \|u\|_2^2$$

On a

$$a(x,x) = x^T A x \geq \gamma \|x\|_2^2 > 0$$

avec x le vecteur des inconnues de votre problème d'éléments finis.

Forme quadratique

Soit la forme quadratique suivante (A est toujours SDP) :

$$f(x) = \frac{1}{2}x^T Ax - b^T x \tag{1}$$

avec $b \in \mathbb{R}^n$. On prouve que la solution x de $Ax = b$ *minimise la forme quadratique* (1). Soit e une perturbation. On a

$$f(x+e) = \frac{1}{2}x^T Ax + \frac{1}{2}e^T Ae + \frac{1}{2}e^T Ax + \frac{1}{2}x^T Ae - b^T x - b^T e.$$

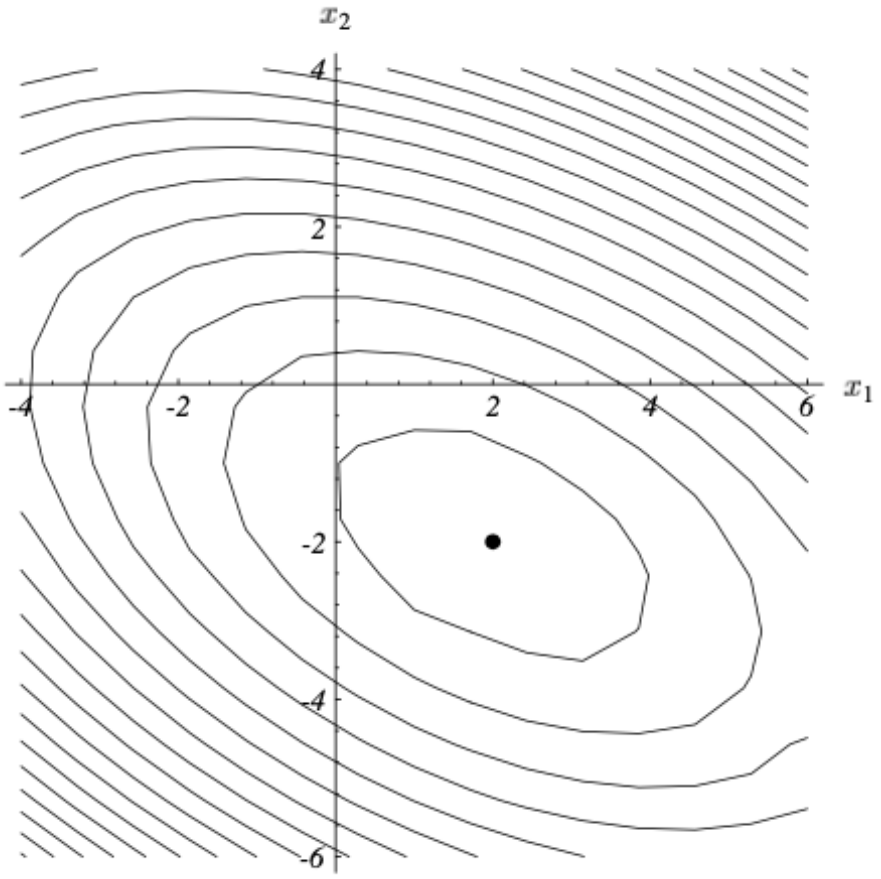
En utilisant la symétrie de A

$$f(x+e) = \underbrace{\frac{1}{2}x^T Ax - b^T x}_{f(x)} + \frac{1}{2}e^T Ae + e^T \underbrace{Ax}_{b} - b^T e = f(x) + \frac{1}{2}e^T Ae.$$

Si on perturbe x qui vérifie $Ax = b$, alors f grandit. La solution x correspond donc à un minimum de f. Ce minimum est unique car f est *convexe* : $f((1-t)x + ty) \leq tf(x) + (1-t)f(y)$.

Fil rouge : $A = \begin{pmatrix} 3 & 2 \\ 2 & 6 \end{pmatrix}$, $b = \begin{pmatrix} 2 \\ -8 \end{pmatrix}$

$$f(x_1, x_2) = \frac{3}{2}x_1^2 + 3x_2^2 + 2x_1x_2 - 2x_1 + 8x_2.$$



Algorithme de la plus grande pente

Utiliser la plus grande pente

Soit un point $x_{(0)}$ quelconque de $\mathbb{R}^n$.

$$f(x_{(0)}) = \frac{1}{2} x_{(0)}^T A x_{(0)} - b^T x_{(0)}.$$

En chaque point $x_{(0)}$ je peux calculer

$$-\nabla f = b - A x_{(0)} = r$$

qu'on appelle le *résidu*. Si, à partir de $x_{(0)}$, j'avance dans la direction inverse du gradient (i.e. le long de r), la fonction f décroit (au moins au voisinage de $x_{(0)}$). On peut donc tenter de minimiser f dans cette direction (problème à une variable α) :

$$f(x_{(0)} + \alpha r) = \frac{1}{2} x_{(0)}^T A x_{(0)} - b^T x_{(0)} + \alpha^2 \frac{1}{2} r^T A r - b^T r + \alpha r^T A x_{(0)} - b^T x_{(0)} - \alpha r^T x_{(0)}.$$

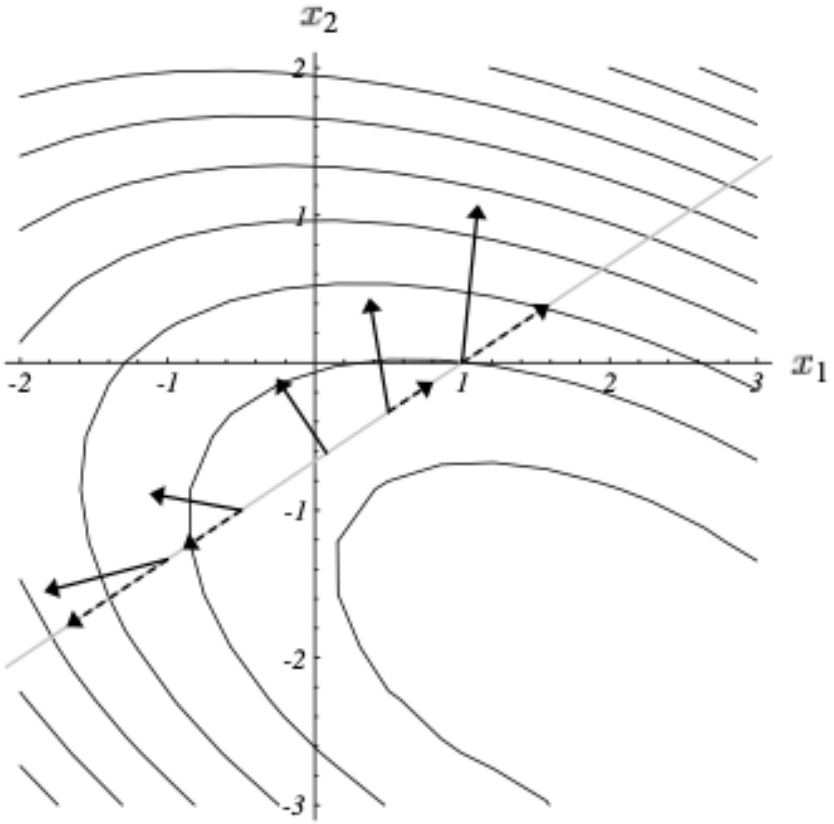
On écrit $\frac{df(x_{(0)}+\alpha r)}{d\alpha} = 0$ pour obtenir

$$\alpha = \frac{r^T x_{(0)} - r^T A x_{(0)}}{r^T A r} = \frac{r^T r}{r^T A r}.$$

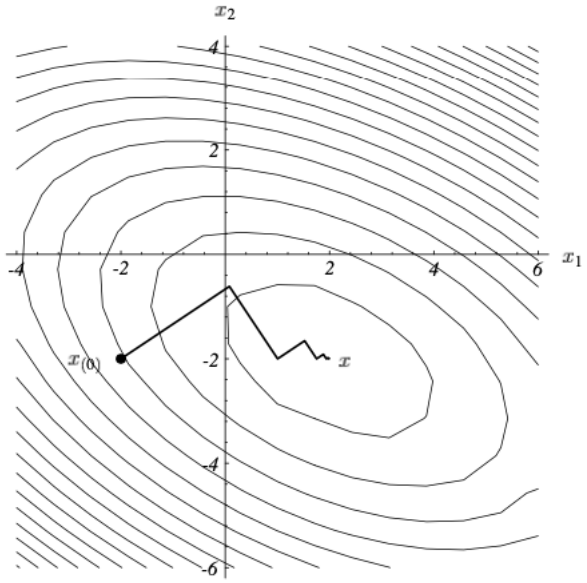
Utiliser la plus grande pente

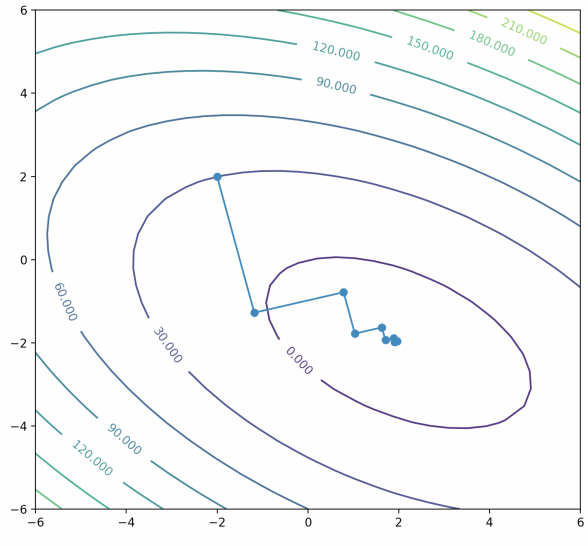
Au point $x_{(0)} + \alpha r$, on a

$$\nabla f|_{x_{(0)}+\alpha r} \cdot r = 0$$



Méthode de la plus grande pente





Faire jusqu'à convergence $\frac{\|r_{(i)}\|}{\|b\|} \leq \epsilon$:

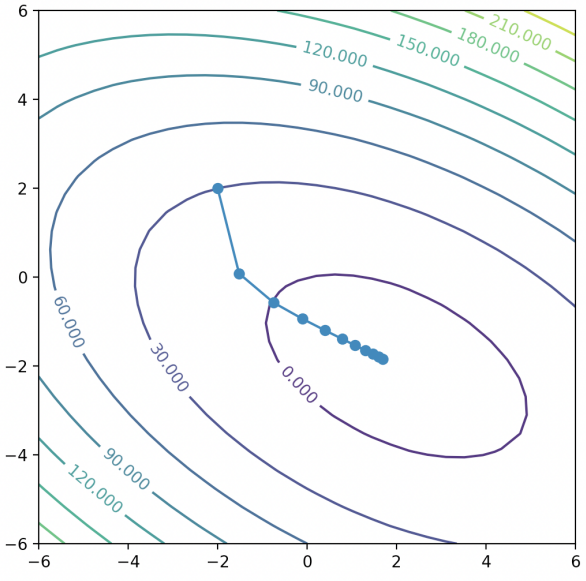
$$r_{(i)} = b - Ax_{(i)},$$

$$\alpha_{(i)} = \frac{r_{(i)}^T r_{(i)}}{r_{(i)}^T A r_{(i)}},$$

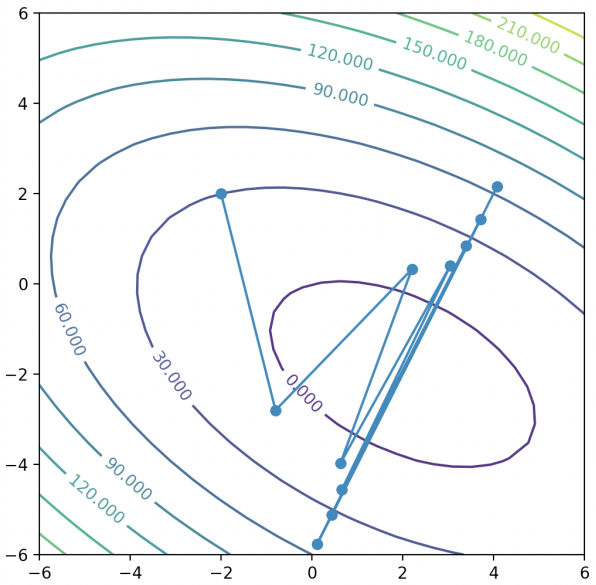
$$x_{(i+1)} = x_{(i)} + \alpha_{(i)} r_{(i)}.$$

Exemple avec PYTHON/NUMPY/MATPLOTLIB

Descente de gradient



$\alpha = 0.12$



$\alpha = 0.30$

Choisir α et faire jusqu'à convergence

$$r_{(i)} = b - Ax_{(i)},$$

$$x_{(i+1)} = x_{(i)} + \alpha_{(i)} r_{(i)}.$$

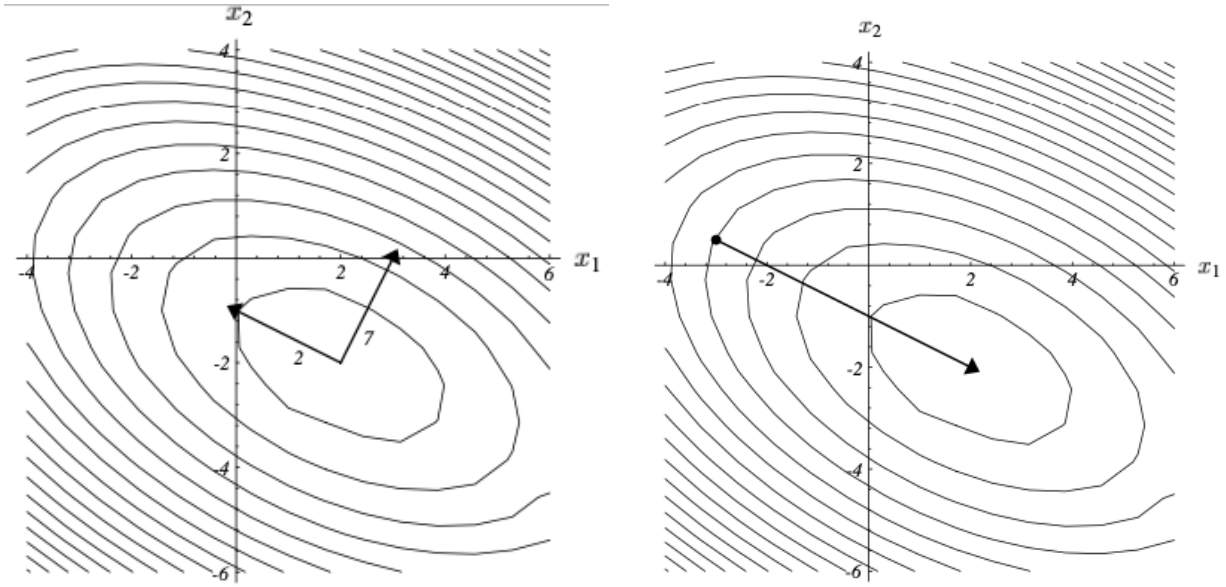
Exemple avec PYTHON/NUMPY/MATPLOTLIB

$$A = \begin{pmatrix} 3 & 2 \\ 2 & 6 \end{pmatrix} = \begin{pmatrix} -2/\sqrt{5} & 1/\sqrt{5} \\ 1/\sqrt{5} & 2/\sqrt{5} \end{pmatrix} \begin{pmatrix} 2 & 0 \\ 0 & 7 \end{pmatrix} \begin{pmatrix} -2/\sqrt{5} & 1/\sqrt{5} \\ 1/\sqrt{5} & 2/\sqrt{5} \end{pmatrix}$$

Si $r_{(i)} = Ax_{(i)} - b = v$ est aligné avec un vecteur propre v (valeur propre λ) de A, alors

$$\alpha_{(i)} = \frac{r_{(i)}^T r_{(i)}}{r_{(i)}^T A r_{(i)}} = \frac{1}{\lambda} \quad \rightarrow \quad x_{(i+1)} = x_{(i)} + \frac{v}{\lambda} \quad \rightarrow \quad Ax_{(i+1)} = \underbrace{Ax_{(i)}}_{b-v} + v = b$$

et donc $x_{(i+1)}$ est la solution !



Convergence de la méthode de la plus grande pente

A est diagonalisable car symétrique $\rightarrow$ ses vecteurs propres $\{v_0 \dots v_{n-1}\}$ forment une base orthonormale de $\mathbb{R}^n$. Le vecteur d'erreur à l'étape i se calcule comme

$$e_{(i)} = x - x_{(i)} = \sum_j \xi_j v_j \quad , \quad r_{(i)} = b - Ax_{(i)} = -Ae_{(i)} = A\sum_j \xi_j v_j = \sum_j \xi_j \lambda_j v_j.$$

Il est assez simple (voir page 15 de la référence) de calculer des égalités du genre

$$\|e_{(i)}\|^2 = \sum_j \xi_j^2 \quad , \quad e_{(i)}^T A e_{(i)} = \sum_j \xi_j^2 \lambda_j \quad , \quad r_{(i)}^T A r_{(i)} = \sum_j \xi_j^2 \lambda_j^3 \quad ,$$

En utilisant ces égalités, et en tenat compte que $e_{(i+1)} = e_{(i)} + \alpha_{(i)} r_{(i)}$ on trouve (éq 25 de la référence)

$$\|e_{(i+1)}\|_A^2 = e_{(i+1)}^T A e_{(i+1)}^T = \|e_{(i)}\|_A^2 \omega^2$$

avec

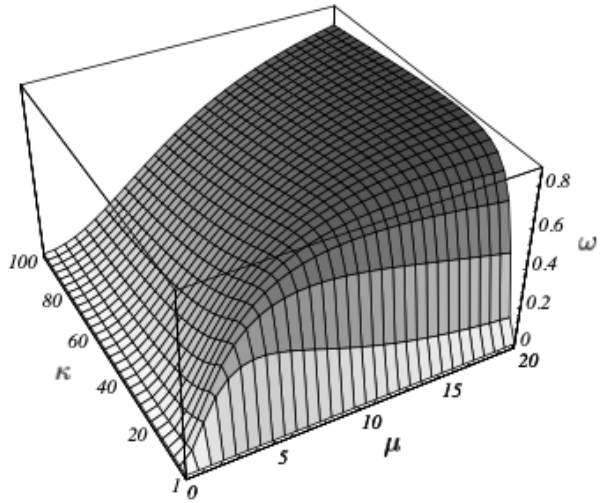
$$\omega^2 = 1 - \frac{(\sum_j \xi_i^2 \lambda_j^2)^2}{(\sum_j \xi_i^2 \lambda_j^3)(\sum_j \xi_i^2 \lambda_j)}.$$

Convergence de la méthode de la plus grande pente

$$\|e_{(i+1)}\|_A^2 = \|e_{(i)}\|_A^2 \omega^2$$

On considère le cas $n = 2$ avec $\lambda_1 \geq \lambda_2$ et le nombre de conditionnement $\kappa = \lambda_1/\lambda_2 \geq 0$. On définit $\mu = \xi_2/\xi_1$ qui dépend du "point de départ" de l'algorithme. Le paramètre μ est la pente de l'erreur $e_{(i)}$ dans le système de coordonnées définit par les vecteurs propres. On trouve dans la référence

$$\omega^2 = 1 - \frac{(\kappa^2 + \mu^2)^2}{(\kappa + \mu^2)(\kappa^3 + \mu^2)}.$$



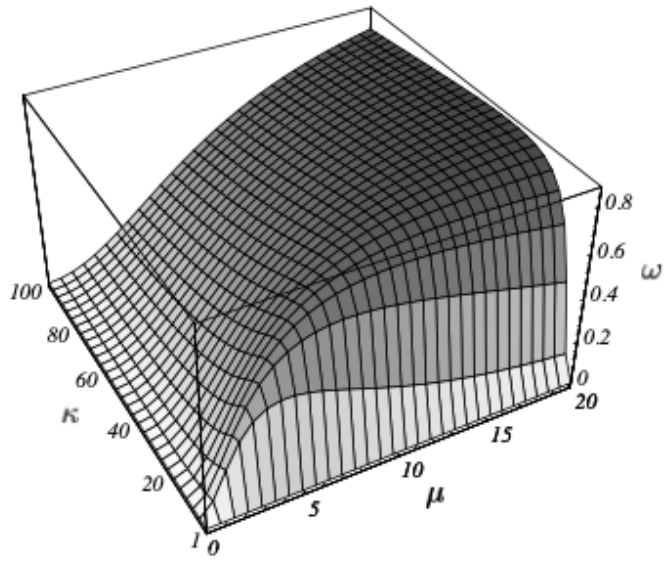
Convergence de la méthode de la plus grande pente

$$\omega^2 = 1 - \frac{(\kappa^2 + \mu^2)^2}{(\kappa + \mu^2)(\kappa^3 + \mu^2)}.$$

Si $\kappa = 1$, $\omega = 0$ et convergence directe.

Si $\mu = 0$ ou $\mu \to \infty$, $\omega = 0$ convergence directe.

Si $\mu = \pm\kappa$, ω^2 est maximal et convergence très mauvaise (zig-zag).



Convergence de la méthode de la plus grande pente

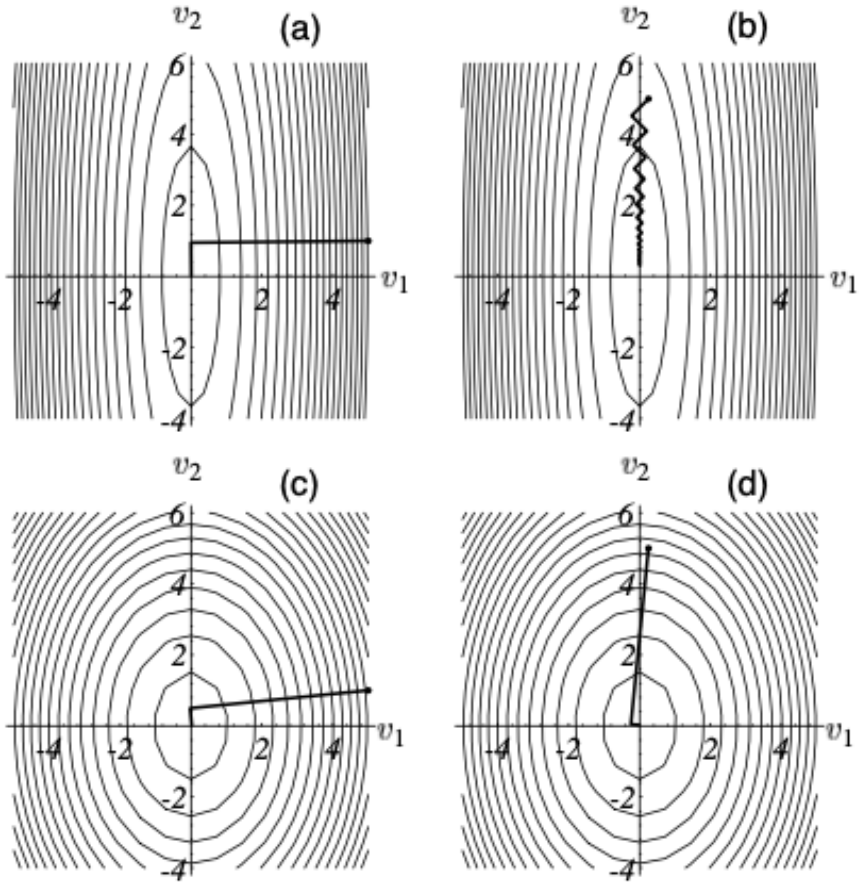


Figure 18: These four examples represent points near the corresponding four corners of the graph in Figure 17. (a) Large κ, small μ. (b) An example of poor convergence. κ and μ are both large. (c) Small κ and μ. (d) Small κ, large μ.

Convergence de la méthode de la plus grande pente

$$\omega^2 = 1 - \frac{(\kappa^2 + \mu^2)^2}{(\kappa + \mu^2)(\kappa^3 + \mu^2)}.$$

Si $\mu = \pm\kappa$, ω^2 est maximal et convergence très mauvaise (zig-zag). On a donc

$$\omega \leq \frac{\kappa - 1}{\kappa + 1}$$

et donc

$$\|e_{(i+1)}\|_A = \|e_{(0)}\|_A \left(\frac{\kappa - 1}{\kappa + 1}\right)^i.$$

Cette relation est vraie pour n'importe quel n !

Le nombre de conditionnement de votre matrice A est proportionnel à $(L/h)^2$ avec L la taille de votre problème et h la taille de maille (taille des éléments). Typiquement, $L/h \approx 100$ et donc $\kappa \approx 10^4$ et $\omega \approx 0,9998$.

En 100 itérations, l'erreur initiale $\|e_{(0)}\|_A$ serait réduite (dans le pire des cas i.e. avec un mauvais point de départ) de 2 % !

Gradients conjugués

Une idée

Observations : la méthode de la plus grande pente choisit souvent plusieurs fois la même direction. Elle est lente (mauvais taux de convergence) mais elle converge. Le nombre d'itérations peut être $\gg n$ alors qu'une itération coûte un produit matrice vecteur ($2n^2$ opérations). Cette méthode est clairement beaucoup plus coûteuse qu'une factorisation $\rightarrow$ inutile en partique pour résoudre $Ax = b$.

Idée : trouver n directions orthogonales entre elles $\{d_{(0)} \dots d_{(n-1)}\}$:

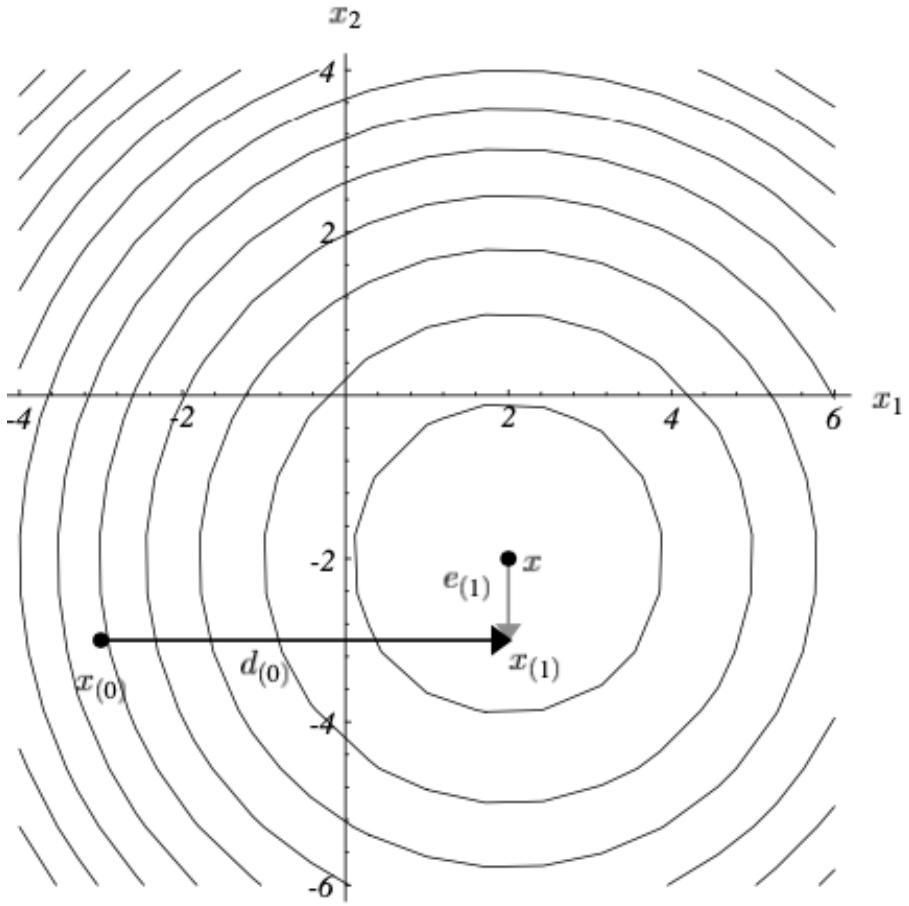
$$x_{(i+1)} = x_{(i)} + \alpha_{(i)} d_{(i)} \quad \rightarrow \quad x_{(i+1)} = x_{(0)} + \sum_{j=1}^{i} \alpha_j d_{(j)}$$

et choisir $\alpha_{(i)}$ pour que $e_{(i+1)}$ soit orthogonal à $d_{(i)}$ ce qui implique que $d_{(i)}$ ne sera plu jamais "visité". On aurait donc

$$e_{(i+1)} = e_{(i)} + \alpha_{(i)} d_{(i)} \quad \rightarrow \quad \alpha_{(i)} = -\frac{d_{(i)}^T e_{(i)}}{d_{(i)}^T d_{(i)}}$$

Problème : pour cela il faudrait connaître $e_{(i)} = x - x_{(i)}$ et donc il faudrait connaître x...

Directions Conjuguées

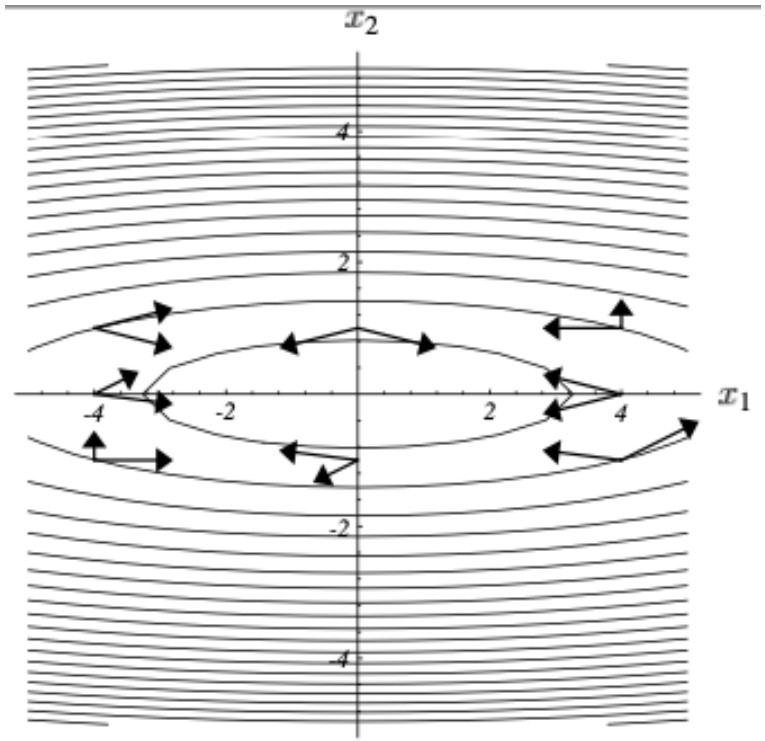


Directions Conjuguées

Au lieu de choisir des directions orthogonales, nous allons choisir des directions A-orthogonales :

$$d_{(i)}^T A d_{(j)} = 0.$$

La matrice A SDP définit un produit scalaire et on peut utiliser ce produit pour définir la notion de A-orthogonlaité.



(a)

Directions Conjuguées

On veut donc

$$d_{(i)}^T A e_{(i+1)} = 0 \quad \rightarrow \quad d_{(i)}^T A(e_{(i)} + \alpha_{(i)} d_{(i)}^T) = 0 \quad \rightarrow \alpha_{(i)} = -\frac{d_{(i)}^T A e_{(i)}}{d_{(i)}^T A d_{(i)}}.$$

On n'a pas l'air beaucoup plus avancés ici... mais en fait si

$$A e_{(i)} = A(x - x_{(i)}) = b - A x_{(i)} = r_{(i)}$$

et donc

$$\alpha_{(i)} = -\frac{d_{(i)}^T r_{(i)}^T}{d_{(i)}^T A d_{(i)}}.$$

Notez ici que c'est la même formule que la plus grande pente si on remplace les d par des r.

Il reste un problème : *comment trouver les* $d_{(i)}$? On a bien une solution : utiliser Gram-Schmidt (modifié) mais le coût ($4/3n^3$ opérations pour GS ou QR) est un peu plus cher qu'une décomposition LU ($2/3n^3$ opérations pour LU).

Damned !

Gram Schmidt conjugué : *First trick, les* $\alpha_{(i)}$

Gram Schmidt : soit $\{u_{(0)} \dots u_{(n-1)}\}$, on choisit $d_{(0)} = u_{(0)}$ et on forme successivement

$$d_{(i)} = u_{(i)} + \sum_{k=0}^{i-1} \beta_{ik} d_{(k)},$$

les β_{ik} étant choisis pour assurer que $d_{(i)}$ est orthogonal au SEV $\text{span}\{d_{(0)} \dots d_{(i-1)}\} = \text{span}\{u_{(0)} \dots u_{(i-1)}\}$

$$\beta_{ij} = -\frac{u_{(i)}^T A d_{(j)}^T}{d_{(j)}^T A d_{(j)}}.$$

Nous allons montrer que seul $\beta_{i,i-1}$ est non nul !

Gram Schmidt conjugué : *First trick, les* $\alpha_{(i)}$

$$\text{Soit} \qquad \mathcal{D}_i = \text{span}\{d_{(0)} \dots d_{(i-1)}\} \subset \mathbb{R}^n$$

l'espace vectoriel engendré par les i premières directions conjuguées. On appelle ce genre d'espace un *espace de Krylov* car il est engendré par $d_{(0)}, Ad_{(0)}, A^2 d_{(0)} \dots, A^{i-1} d_{(0)}$.

Soit $e_{(0)} = \sum_{i=0}^{n-1} \delta_j d_{(j)}$. Il est possible de calculer δ_k car les $d_{(j)}$ son A-orthogonaux :

$$d_{(k)}^T A e_{(0)} = \delta_k d_{(k)}^T A d_{(k)} \quad \rightarrow \quad \delta_k = \frac{d_{(k)}^T A e_{(0)}}{d_{(k)}^T A d_{(k)}}.$$

Ici arrive le premier "truc". On ajoute un terme nul dans la dernière équation

$$\delta_k = \frac{d_{(k)}^T A \overbrace{\left(e_{(0)} + \sum_{i=0}^{k-1} \alpha_{(i)} d_{(i)}\right)}^{e_{(k)}}}{d_{(k)}^T A d_{(k)}} = -\alpha_k.$$

Gram Schmidt conjugué : *First trick, les* $\alpha_{(i)}$

Les itérations conjuguées s'écrivent

$$x_{(i)} = x_{(0)} + \sum_{j=0}^{i-1} \alpha_j d_{(j)}$$

ce qui se traduite en terme d'erreur

$$e_{(i)} = e_{(0)} + \sum_{j=0}^{i-1} \alpha_j d_{(j)} = \sum_{j=i}^{n-1} \alpha_j d_{(j)}.$$

La différence $e_{(i)} = x - x_{(i)}$ vit dans l'espace $\mathbb{R}^n \setminus \mathcal{D}_i$: la solution à l'étape i est la meilleur possible au sens de la norme $\|\cdot\|_A$.

On vient donc de prouver formellement qu'au bout de n itérations, on aura trouvé x !

Ce n'est pas tout à fait vrai : tout processus d'orthogonalisation est entâché par des erreurs d'arrondis ce qui implique que $d_{(i+1)}$ n'est pas exactement dans le complément orthogonal de $\mathcal{D}_i$.

Gram Schmidt conjugué : *Second trick, les* β_{ij}

Écrivons l'algorithme tel que nous l'avons maintenant

$r_{(0)} = b$, $x_{(0)} = 0$, faire jusqu'à convergence

$$\alpha_{(i)} = -\frac{d_{(i)}^T r_{(i)}^T}{d_{(i)}^T A d_{(i)}}$$

$$x_{(i+1)} = x_{(i)} + \alpha_{(i)} d_{(i)}$$

$$r_{(i+1)} = b - A x_{(i+1)} = r_{(i)} - \alpha_{(i)} A d_{(i)}$$

$$d_{(i+1)} = r_{(i+1)} + \sum_{k=0}^{i} \underbrace{-\frac{r_{(i+1)}^T A d_{(k)}^T}{d_{(k)}^T A d_{(k)}}}_{\beta_{ik}} d_{(k)}.$$

Ca marche, mais c'est coûteux à cause du calcul des β_{ij} !

$$\beta_{ij} = -\frac{r_{(i)}^T A d_{(j)}^T}{d_{(j)}^T A d_{(j)}}$$

La relation d'optimalité vue dans le "first trick" dit que l'erreur s'exprime

$$e_{(i)} = \sum_{k=i}^{n-1} \alpha_k d_{(k)} \quad \rightarrow \quad d_{(j)}^T A e_{(i)} = d_{(j)}^T r_{(i)} = 0 \quad \text{si} \quad j < i.$$

Le résidu à l'étape i est donc (simplement) orthogonal à toutes les directions de recherche précédentes ($j < i$).

Les directions de recherche $d_{(i)}$ étant construites à partir des résidus $r_{(i)}$, on a

$$r_{(i)}^T r_{(j)} = 0 \quad i \neq j.$$

On a donc

$$r_{(i)}^T r_{(j+1)} = r_{(i)}^T r_{(j)} - \alpha_j r_{(i)}^T A d_{(j)} \quad \rightarrow \quad \boxed{r_{(i)}^T A d_{(j)} = \frac{1}{\alpha_j}(r_{(i)}^T r_{(j+1)} - r_{(i)}^T r_{(j)})}$$

qui n'est non nul que si $i = j$ ou $i = j + 1$!

Gram Schmidt conjugué :*Second trick, les* β_{ij}

Le "Second trick" :

$$\begin{aligned} d_{(i+1)} &= r_{(i+1)} - \sum_{k=0}^{i} \frac{r_{(i+1)}^T A d_{(k)}}{d_{(k)}^T A d_{(k)}} d_{(k)} \\ &= r_{(i+1)} - \frac{r_{(i+1)}^T A d_{(i)}^T}{d_{(i)}^T A d_{(i)}} d_{(i)} \\ &= r_{(i+1)} + \frac{r_{(i+1)}^T r_{(i+1)}^T}{\alpha_i d_{(i)}^T A d_{(i)}} d_{(i)}. \end{aligned} \tag{2}$$

Dernière simplification (promis !) :

$$\alpha_i A d_{(i)} = r_{(i)} - r_{(i+1)} \quad \rightarrow d_{(i)}^T A d_{(i)} = r_{(i)}^T A d_{(i)} = -\frac{1}{\alpha_{(i)}} r_{(i)} r_{(i)}^T.$$

On trouve (ouf) la formule calssique

$$\boxed{d_{(i+1)} = r_{(i+1)} + \frac{r_{(i+1)}^T r_{(i+1)}}{r_{(i)}^T r_{(i)}} d_{(i)}.}$$

Le gradient conjugué

$r_{(0)} = d_{(0)} = b$, $x_{(0)} = 0$, faire $i = 0, 1, 2, 3 \ldots$ jusqu'à convergence

$$z_{(i)} = A d_{(i)}$$

$$\alpha_{(i)} = -\frac{d_{(i)}^T r_{(i)}^T}{d_{(i)}^T z_{(i)}}$$

$$x_{(i+1)} = x_{(i)} + \alpha_{(i)} d_{(i)}$$

$$r_{(i+1)} = r_{(i)} - \alpha_{(i)} z_{(i)}$$

$$d_{(i+1)} = r_{(i+1)} + \frac{r_{(i+1)}^T r_{(i+1)}}{r_{(i)}^T r_{(i)}} d_{(i)}.$$

Le gradient conjugué

En pratique, on effectue à chaque itération la seule opération coûteuse de l'algorithme, le calcul de $z_{(i)} = Ad_{(i)}$.

Ce calcul peut se faire sans calculer explicitement A : connaissant $d_{(i)}$.

Votre programme d'assemblage éléments finis calcule la matrice de raideur locale d'un élément, K, la multiplie par le vecteur local d contenant les composantes de la direction de recherche $d_{(i)}$ relative à cet élément et l'assemble dans $z_{(i)}$!

Convergence du gradient conjugué

$$\|e_{(i+1)}\|_A \leq 2\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^i \|e_{(0)}\|_A.$$

Si vous voulez une preuve mieux foutue que celle de la référence, *Trefethen, L. N., & Bau III, D. (1997). Numerical linear algebra (Vol. 50). Siam.*

Ce qui vient d'être écrit peut sembler contre intuitif car on devrait avoir

$$\|e_{(n-1)}\|_A = 0$$

ce qui est vrai aussi car

$$0 = \|e_{(n-1)}\|_A \leq 2\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^{n-1} \|e_{(0)}\|_A$$

. Si on considère encore une $\kappa \approx 10^4$, 100 itérations de gradient conjugué multiplieront $2\left(\frac{100-1}{100+1}\right)^{100} = 0.26$ ce qui fait une réduction d'au moins 74% de l'erreur. Avec 200 itérations, on réduit l'erreur de 98 % !