Méthodes numériques : optimisation.
Examen du 10 mai 2017

Recommandations

Les trois exercices sont indépendants. Pour les questions demandant d'écrire du code en python, on supposera que les librairies `numpy` et `matplotlib` ont déjà été chargées, par exemple dans Jupyter avec la directive `%pylab inline`.

Les questions de chaque exercice ne sont pas censées être bloquantes pour les suivantes. N'hésitez donc pas à admettre des résultats et passer à la suite : il y a des questions de cours dans tout l'examen.

1 Changement de variable affine pour Newton-Wolfe

On se donne une fonction $f \in C^2(R^n)$, et des constantes $0 < c_1 < c_2 < 1$.

On définit $\widetilde{f}$ comme la fonction lue après changement de variable $x \mapsto \widetilde{x} = Mx + b$, où M est une matrice inversible de $M_n(\mathbb{R})$ et b un vecteur de $\mathbb{R}^n$: $\widetilde{f}(\widetilde{x}) = \widetilde{f}(Mx + b) = f(x)$. On note H_f et $H_{\widetilde{f}}$ les hessiennes de f et $\widetilde{f}$.

1. Rappeler la définition du fait que $d \in \mathbb{R}^2$ est une direction de descente au point $x \in \mathbb{R}^2$ de la fonction f, en utilisant la fonction auxiliaire $h : \mathbb{R} \to \mathbb{R}$ définie par $h(t) = f(x + td)$. Si d est une direction de descente, quels sont les deux critères pour qu'un pas α satisfasse la règle de Wolfe (pour les constantes c_1 et c_2) ? Exprimer ces critères en fonction de la fonction h et de ses dérivées en 0 et en α, et les illustrer sur le graphe de la fonction h.
2. Montrer que si d est une direction de descente pour f au point x, alors $\widetilde{d} = Md$ est une direction de descente au point $\widetilde{x} = Mx + b$ pour la fonction $\widetilde{f}$, et que si le pas α satisfait la règle de Wolfe pour f au point x dans la direction d, alors ce même pas α satisfait la règle de Wolfe pour $\widetilde{f}$ au point $\widetilde{x}$ dans la direction $\widetilde{d}$.
3. Pour $d \in \mathbb{R}^n$ arbitraire, écrire un développement de Taylor à l'ordre 2 pour $t \mapsto f(x + td)$ et $t \mapsto \widetilde{f}(\widetilde{x} + t\widetilde{d})$ au voisinage de 0, avec un reste sous la forme d'un $o(t^2)$.
   En déduire que $\nabla f(x) = M^T \nabla \widetilde{f}(Mx + b)$, et que l'on a $\langle d, M^T H_{\widetilde{f}}(\widetilde{x}) M d\rangle = \langle d, H_f(x) d\rangle$. En déduire enfin que l'on a $H_f(x) = M^T H_{\widetilde{f}}(\widetilde{x}) M$, et que donc $H_f(x)$ est symétrique définie positive si et seulement si $H_{\widetilde{f}}(\widetilde{x})$ l'est.
4. On effectue la méthode de Newton à pas variable satisfaisant la règle de Wolfe sur f en partant de x_0. Donner la relation de récurrence entre les itérées. On suppose qu'à chaque étape la direction d_k donnée par la méthode est une direction de descente au point x_k. Si l'on pose $\widetilde{x}_k = Mx_k + b$, montrer que la suite $\widetilde{x}_k$ correspond aux itérées d'une méthode de Newton partant de $\widetilde{x}_0$ pour la fonction $\widetilde{f}$, dont les directions sont bien des directions de descente et les pas satisfont bien la règle de Wolfe à chaque étape.

2 Descente de gradient et règle de Wolfe

On souhaite programmer une méthode de descente de gradient à pas variable, avec un pas qui satisfait la règle de Wolfe. On veut tester cette méthode sur la fonction f de $\mathbb{R}^2$ dans $\mathbb{R}$ définie pour $x = (u, v)$ par $f(x) = \frac{-1}{1+u^2+3v^2}$.

On définit les fonctions `f`, et `gradf` correspondant à f et ∇f, et on se donne une variable globale permettant d'évaluer le nombre total d'appels à ces fonctions. On se donne enfin $0 < c_1 < c_2 < 1$, et le code suivant de la fonction `pasWolfe`, permettant de calculer un pas satisfaisant la règle de Wolfe au point x pour la direction de descente d, en se donnant une valeur de pas initiale α_{init}, et en supposant que les valeurs de $f(x)$ et $\nabla f(x)$ ont déjà été calculées (respectivement dans les variables `h0` et `gfx`). On rappelle que la fonction `vdot` permet de calculer le produit scalaire entre deux vecteurs.

```
compteurappels=0

def f(x):
    global compteurappels
    compteurappels+=1
    denom=1+x[0]**2+3*x[1]**2
    return -1/denom

def gradf(x):
    global compteurappels
    compteurappels+=1
    denom=1+x[0]**2+3*x[1]**2
    v=array([x[0],3*x[1]])
    return 2/denom**2*v

c1=0.1
c2=0.7

```

```
def pasWolfe(x,d,ainit,h0,gfx,Nmax=30):
    hp0=vdot(gfx,d)
    c1hp0=c1*hp0
    c2hp0=c2*hp0
    amin=ainit
    hamin=f(x+amin*d)
    for i in range(Nmax):
        amax=amin
        hamax=hamin
        if hamin<=h0+amin*c1hp0:
            break
        amin=amin/2
        hamin=f(x+amin*d)
    for i in range(Nmax):
        gfxmin=gradf(x+amin*d)
        if vdot(gfxmin,d)>=c2hp0:
            return amin,hamin,gfxmin
        if hamax>h0+amax*c1hp0:
            break
        amin=amax
        hamin=hamax
        amax=2*amin
        hamax=f(x+amax*d)
    for i in range(Nmax):
        a=(amin+amax)/2
        ha=f(x+a*d)
        gfxa=gradf(x+a*d)
        if ha<=h0+amax*c1hp0:
            if vdot(gfxa,d)>=c2hp0:
                break
            else:
                amin=a
        else:
            amax=a
    return a,ha,gfxa
```

1. On suppose que f est de classe C^1, bornée inférieurement, et que $d \in \mathbb{R}^2$ est une direction de descente au point x. On se donne un $\alpha_{\text{init}} > 0$ et on suppose que l'on effectue l'algorithme correspondant aux lignes 19 à 53 (la fonction `pasWolfe`) en donnant à la variable `ainit` la valeur α_{init}, à `h0` la valeur de $h(0) = f(x)$, et à `gfx` la valeur de $\nabla f(x)$. On suppose également que tous les calculs sont exacts.
   (a) Quels sont les critères satisfaits (ou non satisfaits) par les pas $\alpha_{\min}$ et $\alpha_{\max}$ (correspondant aux valeurs des variables `amin` et `amax`) lorsqu'on sort des deux premières boucles (lignes 25 et lignes 32) avant d'avoir fait les `Nmax` itérations (donc par les instructions `break` des lignes 29 et 37), ainsi que tant que l'on est dans la dernière boucle (commençant ligne 42) ?
   (b) Expliquer en quoi les trois hypothèses principales
   — la fonction f est de classe C^1
   — la fonction f est bornée inférieurement
   — la direction d est une direction de descente
   sont chacune utilisées dans la preuve du fait que chacune des trois boucles (commençant aux lignes 25, 32 et 42) terminent avant d'avoir fait les les `Nmax` itérations, lorsque `Nmax` est suffisamment grand.
   (c) En déduire qu'il existe un pas $\alpha > 0$ satisfaisant la règle de Wolfe, et que si `Nmax` est suffisamment grand, l'algorithme renvoie un triplet dont le premier élément correspond à un tel pas α (et les deux autres éléments correspondent respectivement à $h(\alpha) = f(x+\alpha d)$ et $\nabla f(x+\alpha d)$).
2. Pourquoi a-t-on mis les variables d'entrée `h0` et `gfx` dans la fonction `pasWolfe` ? Si on suppose que la valeur de la variable `ainit` est un pas qui satisfait la règle de Wolfe au point x pour la direction d, combien d'évaluations des fonctions `f` et `gradf` sont effectuées lors de l'appel de la fonction `pasWolfe` ?
3. Compléter les deux fonctions suivantes, correspondant aux méthodes de descente de gradient en utilisant soit un pas satisfaisant la règle de Wolfe, soit un pas fixe. Elles doivent renvoyer un point correspondant à une approximation du minimum, et des listes comprenant pour chaque itération k de la méthode, les valeurs du nombre d'évaluations total des fonctions `f` et `gradf` (depuis le point initial), de la norme du gradient $\|\nabla f(x_k)\|$, et de la valeur de $f(x_k)$. On prendra comme critère d'arrêt le fait que la norme du gradient est inférieure à la tolérance donnée.

```
54  def descenteGradWolfe(x0,tol,ainit=1,Nmax=1000):
55      global compteurappels
56      compteurappels=0
57      listecompteur=[]
58      listenormes=[]
59      listef=[]
60      ...
61      return x,listecompteur,listenormes,listef
62
63  def descenteGradPasFixe(x0,tol,a=1,Nmax=1000):
64      global compteurappels
65      compteurappels=0
66      listecompteur=[]
67      listenormes=[]
68      listef=[]
69      ...
70      return x,listecompteur,listenormes,listef
```

4. Écrire en python les instructions correspondant au test de ces deux méthodes en partant du point $(2, 2)$, pour une tolérance de 10^{-5}. Afficher dans une échelle appropriée les erreurs en fonction du nombre d'évaluations, ainsi que les valeurs de $f(x_k) - f(x_*)$ en fonction du nombre d'évaluations, où $x_* = (0, 0)$ est le point de minimum de f (que l'on connaît ici à l'avance).

3 Taux de convergence de la méthode du gradient conjugué

On considère une matrice symétrique définie positive $A \in M_n(\mathbb{R})$ et un vecteur $b \in \mathbb{R}^n$. On cherche à approximer la solution x_* de l'équation $Ax + b = 0$ par la méthode du gradient conjugué. On suppose que n est grand et qu'on n'a pas les moyens d'effectuer n itérations de la méthode, donc on s'intéresse à l'estimation de l'erreur sur les premières itérations. On prend les notations $x_k, \alpha_k, r_k, p_k, \beta_k$ du cours.

1. Montrer que la norme $\|\cdot\|_A$ (définie par $\|x\|_A^2 = \langle x, Ax\rangle$) permet d'obtenir un lien entre l'erreur entre x et x_* (dans cette norme) et la valeur de la fonction $f : x \mapsto \frac{1}{2}\langle x, Ax\rangle + \langle b, x\rangle$ par rapport à sa valeur minimale, au sens où pour $x \in \mathbb{R}^n$, on a
$$f(x) - f(x_*) = \frac{1}{2}\|x - x_*\|_A^2.$$

2. On pose $e_k = x_k - x_*$. Rappeler l'interprétation de p_k et α_k en terme de méthode de descente. En déduire que $\|e_k + tp_k\|_A^2 \geqslant \|e_{k+1}\|_A^2$, avec égalité si $t = \alpha_k$.
Rappeler les relations de conjugaison entre les $(p_i)_{0\leqslant i\leqslant k}$. Montrer que si $v \in \mathrm{Vect}(p_0, \dots, p_{k-1})$, alors $\|e_k + v + tp_k\|_A^2 = \|e_k + v\|_A^2 - \|e_k\|_A^2 + \|e_k + tp_k\|_A^2$. En déduire par récurrence que e_k est un minimiseur du problème suivant
$$\inf_{e\in e_0+\mathrm{Vect}(p_0,\dots,p_{k-1})} \|e\|_A^2.$$

3. Montrer que $r_k = Ae_k$. En déduire que pour tout $k \geqslant 1$ $e_k - e_0 \in \mathrm{Vect}(Ae_0, A^2e_0, \dots, A^ke_0)$, et qu'il existe donc un polynôme $P_k \in \mathbb{R}_k[X]$ tel que $P_k(0) = 1$ et $e_k = P_k(A)e_0$. En déduire enfin que P_k est un minimiseur du problème
$$\inf_{P\in\mathbb{R}_k[X],P(0)=1} \|P(A)e_0\|_A^2.$$

4. On pose $(v_i)_{1\leqslant i\leqslant n}$ une base de vecteurs propres de A (associée aux valeurs propres $(\lambda_i)_{1\leqslant i\leqslant n}$). En décomposant e_0 dans cette base, si P est un polynôme, obtenir une expression de $P(A)e_0$, puis de $\|P(A)e_0\|_A^2$ faisant intervenir les $P(\lambda_i)$. Montrer qu'on a alors, pour tout polynôme P de $\mathbb{R}_k[X]$ tel que $P(0) = 1$, en notant Λ l'ensemble des valeurs propres de A,
$$\|e_k\|_A^2 \leqslant \max_{\lambda\in\Lambda} |P(\lambda)|^2 \|e_0\|_A^2.$$

5. *Application :* on suppose que la matrice A a une seule « petite » valeur propre $\frac{1}{\kappa} > 0$ (avec $\kappa \gg 1$) et que toutes ses autres valeurs propres sont dans $[1 - \rho, 1]$ avec $\rho \in]0, 1 - \frac{1}{\kappa}[$.
Quel est le nombre de conditionnement de la matrice A, dans le pire des cas ? À quel taux de convergence peut-on s'attendre d'après le cours ? Montrer qu'on a en fait, pour tout $k \geqslant 1$,
$$\|e_k\|_A \leqslant \kappa\rho^{k-1}\|e_0\|_A,$$
et que donc pour certaines valeurs de κ et de ρ on a une convergence plus rapide. On pourra utiliser le polynôme $P(X) = (1 - \kappa X)(1 - X)^{k-1}$.

6. * On veut démontrer le résultat donné dans le cours. On suppose donc que les valeurs propres de A sont dans $[L, \ell]$. On considère le polynôme de Tchebychev $T_k \in \mathbb{R}^k[X]$ défini par les expressions suivantes (on peut montrer, et on l'admet, que c'est effectivement un polynôme de degré k) :
$$T_k(x) = \begin{cases} \cos(k \arccos x) & \text{pour } x \in [-1, 1], \\ \frac{1}{2}[(x + \sqrt{x^2 - 1})^k + (x - \sqrt{x^2 - 1})^k] & \text{pour } |x| \geqslant 1. \end{cases}$$
En utilisant le polynôme $P(x) = \frac{T_k(\frac{L+\ell-2x}{L-\ell})}{T_k(\frac{L+\ell}{L-\ell})}$, montrer que l'on a $|P(x)| \leqslant \left|T_k(\frac{L+\ell}{L-\ell})\right|^{-1}$ si $x \in [\ell, L]$.
En déduire que l'on a donc
$$\|e_k\|_A \leqslant 2\left[\left(\frac{\sqrt{L} - \sqrt{\ell}}{\sqrt{L} + \sqrt{\ell}}\right)^k + \left(\frac{\sqrt{L} + \sqrt{\ell}}{\sqrt{L} - \sqrt{\ell}}\right)^k\right]^{-1} \|e_0\|_A \leqslant 2\left(\frac{\sqrt{L} - \sqrt{\ell}}{\sqrt{L} + \sqrt{\ell}}\right)^k \|e_0\|_A.$$