Notes complémentaires : Fonctions de plusieurs variables

Vincent Souveton

2022-2023

Les fonctions de plusieurs variables servent à modéliser une quantité, scalaire ou vectorielle, qui dépend d'un ou de plusieurs paramètres. Par exemple, l'aire d'un rectangle, une grandeur scalaire, dépend à la fois de sa petite et de sa grande longueur. De même, les coordonnées d'un mobile évoluant dans le plan, une grandeur vectorielle, varient au cours du temps. Il s'agit donc dans ce chapitre de généraliser certaines notions bien connus pour les fonctions de la variable réelle à valeurs dans l'ensemble des réels. On cherche entre autres à répondre aux questions suivantes :

- Comment représenter graphiquement ces fonctions de plusieurs variables ?
- Est-il possible d'étudier leurs variations ? La notion de dérivée a-t-elle toujours un sens ?
- Peut-on interpréter géométriquement certaines de leurs propriétés ?

Ces considérations sont à la base de ce qu'on appelle le calcul différentiel. Dans ces notes complémentaires, nous allons illustrer ces notions à travers l'étude d'un exemple simple. L'objectif final est l'implémentation d'un célèbre algorithme d'optimisation : la descente de gradient.

1 Étude d'une fonction

On considère la fonction $f : \mathbf{R}^2 \to \mathbf{R}$ définie par $f(x, y) = x^2 + 2y^2$. Commençons par examiner certaines propriétés de la fonction f. Il s'agit d'une fonction (continue) de deux variables à valeurs dans l'ensemble des réels. Sa représentation graphique est donc une surface, comme illustré avec Geogebra :

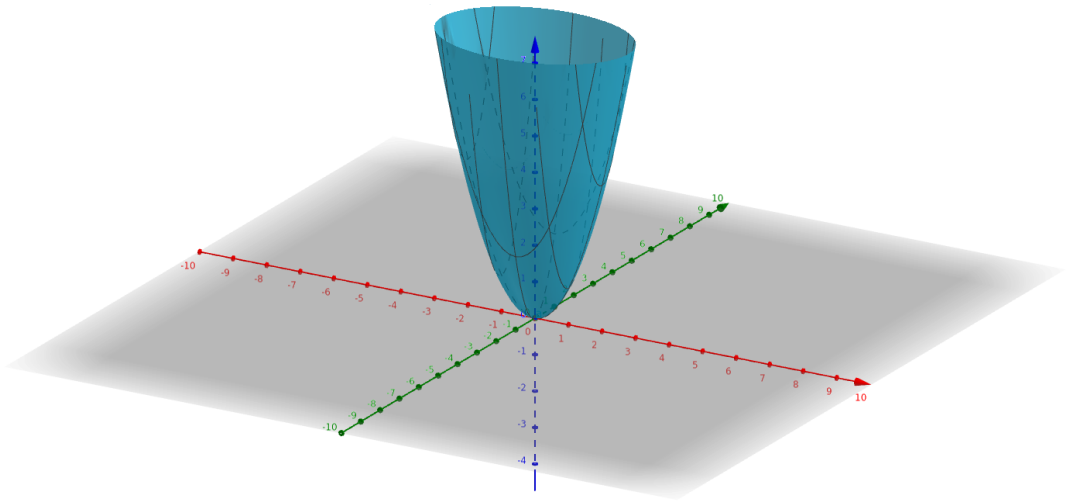

Figure 1: Représentation graphique de la fonction de deux variables $f(x, y) = x^2 + 2y^2$.

Ses lignes de niveaux $L_a := \{(x, y) \in \mathbf{R}^2 \ ; \ f(x, y) = a\}$ sont définies de la façon suivante :

$$(x, y) \in L_a \iff x^2 + 2y^2 = a$$

- si $a < 0$, alors $L_a = \emptyset$;
- si $a = 0$, alors $L_a = \{(0, 0)\}$;
- si $a > 0$, alors $x^2 + 2y^2 = a \iff \left(\frac{x}{\sqrt{a}}\right)^2 + \left(\frac{y}{\sqrt{a/2}}\right)^2 = 1$ et L_a est une ellipse.

Cela est confirmé visuellement par la représentation de quelques unes de ces lignes avec un script Python :

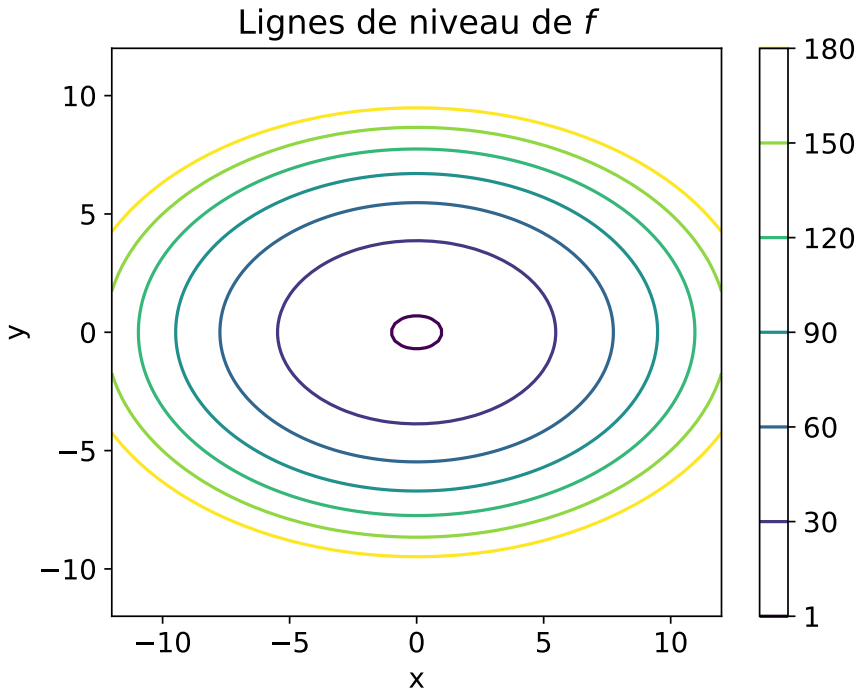

Figure 2: Représentation de quelques lignes de niveau de f. Les points appartenant à une même ligne de couleur ont même image par la fonction.

2 Recherche théorique du minimum global

L'optimisation est la branche des mathématiques qui cherche à minimiser (ou maximiser) une quantité, la plupart du temps modélisée par une fonction de plusieurs variables, avec éventuellement des contraintes. Ici, on veut trouver le minimum global de la fonction f, c'est à dire à résoudre :

$$\min_{(x,y)\in\mathbf{R}^2} f(x, y)$$

La résolution de ce problème est immédiate. Dans notre cas, les graphiques de la partie précédente illustrent clairement que l'on a un minimum global unique en $(0, 0)$ et qu'il vaut 0. Et en effet, pour tout $(x, y) \in \mathbf{R}^2$, on a $f(x, y) \geq 0$, par définition de la fonction comme somme de deux carrés. Or, $f(0, 0) = 0$. Cela prouve que $(0, 0)$ est un minimum global. Ce minimum est de plus unique car pour tout $(x, y) \neq (0, 0)$, on a $f(x, y) > 0$. On a donc prouvé l'existence et l'unicité d'un minimum global pour f. Ce genre de raisonnement particulier est parfaitement adapté à des fonctions simples comme c'est le cas ici.

On peut aussi démontrer des résultats plus généraux qui font appel à des propriétés globales sur la fonction à minimiser. Souvent, il est commode de se ramener à l'optimisation d'une fonction convexe car on dispose de résultats d'existence et d'unicité de minima pour de telles fonctions. En particulier, une fonction strictement convexe est une fonction qui vérifie, pour tous $u, v \in \mathbf{R}^2$ et $t \in]0,1[$:

$$f(u+tv) < tf(u) + (1-t)f(v)$$

On a alors le théorème suivant, qui s'applique précisément à notre fonction (avec $d = 2$) :

Théorème 1. *Soit $f : \mathbf{R}^d \to \mathbf{R}$, $d \geq 1$, une fonction continue. Si :*

1. *$f(x,y) \to +\infty$ quand $||(x,y)|| \to +\infty$;*
2. *f est strictement convexe ;*

alors il existe un unique minimum global de f sur $\mathbf{R}^d$.

Preuve.

- Existence : On peut réécrire la condition 1 plus spécifiquement sous la forme :

  $$\exists r \in \mathbf{R} \text{ t.q. } ||x|| > r \Longrightarrow f(x) \geq f(0)$$

  Ainsi, l'inf des $f(u)$ sur $\mathbf{R}^d$ est en fait l'inf des $f(v)$ sur une boule fermée bornée, donc compacte car on est en dimension finie d. Or, une fonction continue sur un compact y atteint son minimum.

- Unicité : Supposons qu'il existe u et v deux minima distincts de la fonction tels que $f(u) = f(v) = m$. Alors, par stricte convexité de la fonction,

  $$f\left(\frac{u+v}{2}\right) < \frac{1}{2}f(u) + \frac{1}{2}f(v) = m$$

  ce qui est absurde, car cela voudrait dire qu'on a trouvé un nouveau minimum, différent de u et de v. Donc il ne peut exister qu'un seul miminum.

On a ainsi prouvé l'existence et l'unicité du minimum avec ces hypothèses. La preuve d'existence utilise des arguments de compacité que vous aborderez plus tard dans votre cursus. La preuve d'unicité, elle, n'utilise rien de compliqué et constitue une jolie application de la définition de convexité. □

3 Recherche numérique du minimum global

Nous allons maintenant exploiter les propriétés géométriques de la fonction f afin de construire un algorithme qui converge vers le minimum de la fonction. Intuitivement, comment, partant d'un point quelconque sur la surface, est-il posible de *glisser* le long de la surface jusqu'à atteindre le minimum global ? On sait que le vecteur gradient pointe localement vers la direction de pente maximale. L'idée de l'algorithme, attribuée historiquement à Cauchy, est donc toute simple : il s'agit de se déplacer, à chaque étape, dans la direction opposée à ce vecteur. On espère alors atteindre le minimum global de la fonction en dévalant sa surface représentative, suivant ses pentes les plus abruptes. Il faut donc définir un point de départ $(x_0, y_0) \in \mathbf{R}^2$, le pas de temps $\alpha \in \mathbf{R}$ qui donne la durée de chaque étape et un critère d'arrêt $N \in \mathbf{N}^*$. Une fois ces paramètres choisis, l'algorithme s'écrit de la façon suivante en pseudo-code :

Algorithme 1 Descente de gradient

Require: $(x_0, y_0) \in \mathbf{R}^2$, $\alpha \in \mathbf{R}_+^*$, $N \in \mathbf{N}^*$
  $(x,y)^0 \leftarrow (x_0, y_0)$
  for $i \in \{1, ..., N\}$ **do**
    $(x,y)^i \leftarrow (x,y)^{i-1} - \alpha \nabla f\left((x,y)^{i-1}\right)$
  end for

En Python, cela correspond au code suivant, sachant qu'ici $\nabla f(x,y) = \begin{pmatrix} \frac{\partial f}{\partial x}(x,y) \\ \frac{\partial f}{\partial y}(x,y) \end{pmatrix} = \begin{pmatrix} 2x \\ 4y \end{pmatrix}$:

```
import numpy as np
import matplotlib.pyplot as plt

def grad(X):
    Y = np.copy(X)
    Y[0] = 2*X[0]
    Y[1] = 4*X[1]
    return Y

def gradient_descent(start,alpha,N):
    vector = start
    P = [vector]
    for _ in range(N):
        diff = -alpha * grad(vector)
        vector = vector + diff
        P.append(vector)
    return P

P = gradient_descent(np.array([10.,11.]),0.1,25)
px = [P[i][0] for i in range(len(P))]
py = [P[i][1] for i in range(len(P))]

plt.figure()
xlist = np.linspace(-12.0, 12.0, 100)
ylist = np.linspace(-12.0, 12.0, 100)
X, Y = np.meshgrid(xlist, ylist)
Z = X**2 + 2*Y**2
fig,ax=plt.subplots(1,1)
cp = ax.contour(X, Y, Z, [1,30,60,90,120,150,180])
fig.colorbar(cp) # Add a colorbar to a plot
ax.set_title('Descente de gradient')
ax.set_xlabel('x')
ax.set_ylabel('y')
plt.scatter(px,py, linewidth=0.01, c='red')
plt.text(px[0], py[0], 'start', c='red', va='bottom', ha='center')
plt.text(px[-1], py[-1], 'end', c='red', va='bottom', ha='center')
plt.show()
```

En exécutant le script, on observe nettement la convergence vers le minimum global :

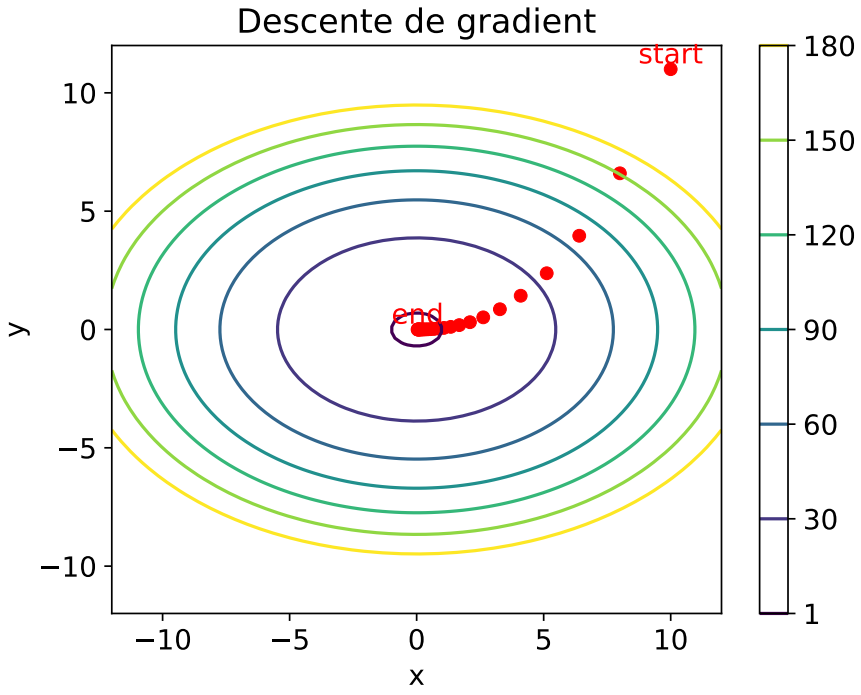

Figure 3: Mise en œuvre de l'algorithme de descente de gradient.

Au bout de 25 étapes, on converge en effet vers $(3, 8.10^{-2}; 3, 1.10^{-}5)$, qui est proche du vrai minimum global $(0; 0)$. Une des principales limitations de cet algorithme est qu'il peut se retrouver piégé dans un minimum local, si la fonction n'est pas convexe. Aussi, il est important de bien régler ses paramètres, notamment le pas de temps. Certaines variantes proposent un pas qui s'ajuste au cours du temps afin de garantir une exploration optimale.