
Projet d'analyse numérique

A. Introduction

Nous allons, au cours de ce projet, nous intéresser à la recherche numérique des extrema d'une fonction dans un premier temps à une variable, puis à plusieurs variables à valeurs réelles.

B. Optimisation d'une fonction d'une variable réelle

Dans cette partie, nous allons construire et tester un algorithme permettant de déterminer une valeur approchée du minimum d'une fonction définie sur un intervalle $[a,b]$.

Dans un premier temps, nous utiliserons deux types de méthodes :

- Méthode par balayage à pas constant : on subdivise l'intervalle $[a,b]$ en N intervalles de même longueur $\Delta x = \frac{b-a}{N}$. On cherche ensuite le minimum des valeurs prises par la fonction en chacun des points de la subdivision
- Méthodes par balayage aléatoire : On génère de façon aléatoire $N+1$ valeurs comprises entre a et b . On cherche ensuite le minimum des valeurs prises par la fonction en ces valeurs.

Soit la fonction f définie sur $[0;3]$ tel que :

$$f(x) = x^3 - 3x^2 + 2x + 5$$

Etude théorique

Calculons dans un premier temps la dérivée de la fonction f :

$$\frac{df}{dx} = 3x^2 - 6x + 2$$

Cette dérivée à deux racines :

$$- \quad x_1 = 1 - \frac{1}{\sqrt{3}}$$

$$- \quad x_2 = 1 + \frac{1}{\sqrt{3}}$$

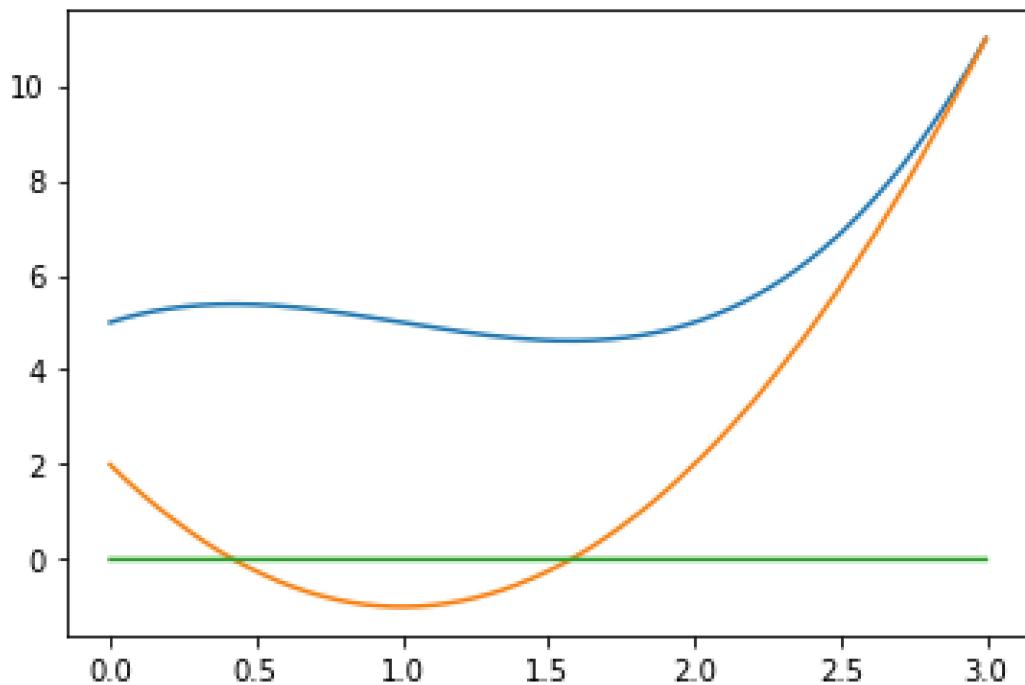
D'où le tableau de variation suivant :

x	0	$1 - \frac{1}{\sqrt{3}}$	$1 + \frac{1}{\sqrt{3}}$	3
$\frac{df}{dx}$	+	0	-	0
f(x)	5	5,38	4,615099	11

Le minimum de la fonction sur $[0;3]$ est obtenu pour $x_2 = 1 + \frac{1}{\sqrt{3}}$

On a donc $\min(f(x)) = 4,615099$ et $\max(f(x)) = f(3) = 11$

- Représentation graphique



La courbe bleue représente $f(x)$ et la courbe verte $f'(x)$. On remarque bien que le minimum de $f(x)$ est d'environ 4,5 et que le maximum est d'environ 11, ce qui confirme les valeurs calculées précédemment.

2. Etude numérique

Nous allons maintenant chercher à obtenir ce résultat par des méthodes numériques, en commençant par la méthode du balayage à pas constant et du balayage aléatoire décrits précédemment.

Afin de pouvoir comparer les différents résultats, nous fixons N à 100 pour les deux méthodes.

	Balayage à pas constant	Balayage aléatoire	Valeur réelle
Minimum obtenu	4,615379	4,624059	4,6150998

On remarque que la méthode de balayage à pas constant est plus proche de la valeur réelle.

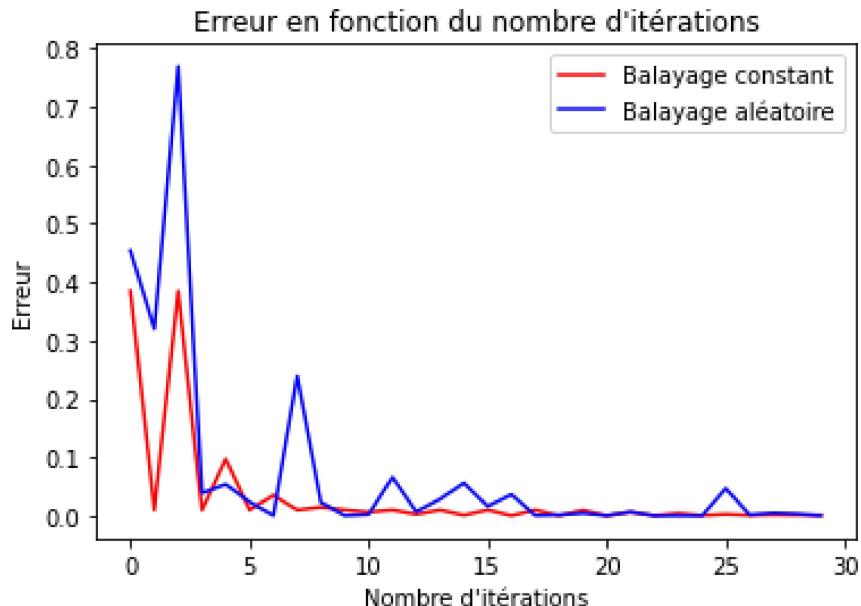
Cependant, la méthode de balayage aléatoire repose, comme son nom l'indique, sur du hasard. On peut supposer qu'il soit possible d'obtenir un résultat plus précis par cette méthode à N fixé, mais que cela repose sur de la chance.

Afin de mieux pouvoir comparer ces méthodes, nous allons tracer les courbes d'erreurs.

3. Courbes d'erreurs

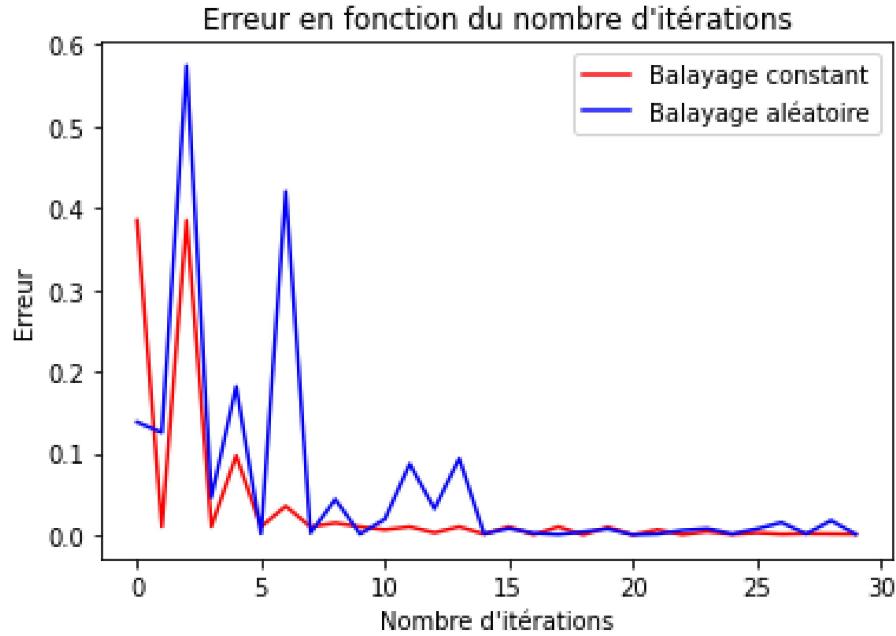
Afin de les tracer, nous allons chercher les valeurs obtenues par l'algorithme en faisant varier N de 1 à 30 (abscisse).

En ordonnée se trouve alors l'erreur absolue (valeur obtenue par l'algorithme - valeur réelle).



On remarque que la courbe d'erreur de la méthode du balayage à pas aléatoire possède des "pics" qui sont dû au caractère aléatoire de la méthode. Globalement, la méthode de balayage à pas constant est plus fiable, puisque lorsque N augmente, la méthode converge vers 0, et ce, plus vite que l'autre méthode.

Traçons une nouvelle fois les courbes afin d'étudier le comportement de la méthode à balayage aléatoire.



On obtient alors un graph différent, il est donc complexe de se fier au résultat étant donné sa nature aléatoire.

La méthode par balayage à pas constant semble donc plus fiable et efficace.

4. Calcul du maximum

On définit un nouveau programme en se basant sur la méthode par balayage à pas constant : on subdivise l'intervalle $[a,b]$ en N intervalles de même longueur $\Delta x = \frac{b-a}{N}$. On cherche ensuite le maximum des valeurs prises par la fonction en chacun des points de la subdivision.

Toujours avec $N=100$, on trouve $\max(f(x)) = 10,999999$ soit quasiment 11, la valeur théorique.

5. Méthode du gradient 1D

On teste maintenant une nouvelle méthode, appelée méthode du gradient 1D.

- On se fixe $u < 0$ assez petit. On considère alors la suite définie par :

- $x_0 \in [a, b]$
- $x_{n+1} = x_n + u * f'(x_n)$

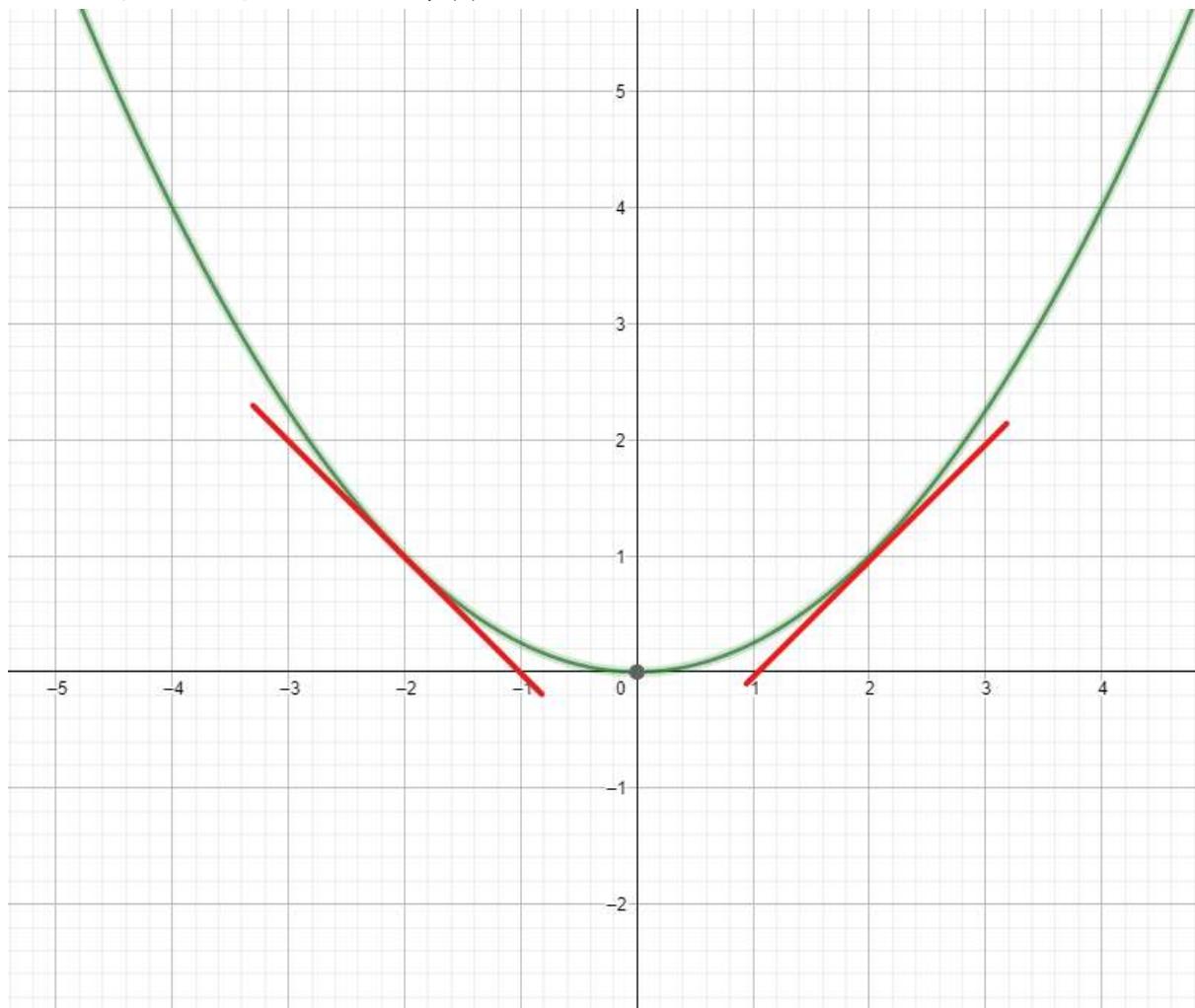
Après quelques itérations, on approche le minimum de f par $f(x_n)$

Si la fonction f est croissante, cela signifie par définition que sa dérivée f' est positive. Or comme u est négatif, $u * f'(x_n) < 0$. Ainsi, on a donc $x_{n+1} < x_n$.

Inversement, si la fonction f est décroissante, sa dérivée f' est négative, donc $u * f'(x_n) > 0$ et $x_{n+1} > x_n$

Ainsi, la suite $(x_n)_n$ se stabilise dans les minimums locaux.

Prenons par exemple la fonction $f(x) = 0,25x^2$ définie sur \mathbb{R}



Si l'on se trouve du côté gauche de l'axe des ordonnées ($x < 0$), la fonction est décroissante, sa tangente a donc une pente négative, $u * f'(x_n) > 0$ et donc $x_{n+1} > x_n$.

Ainsi, on se rapproche de l'axe des ordonnées et donc de $x = 0$

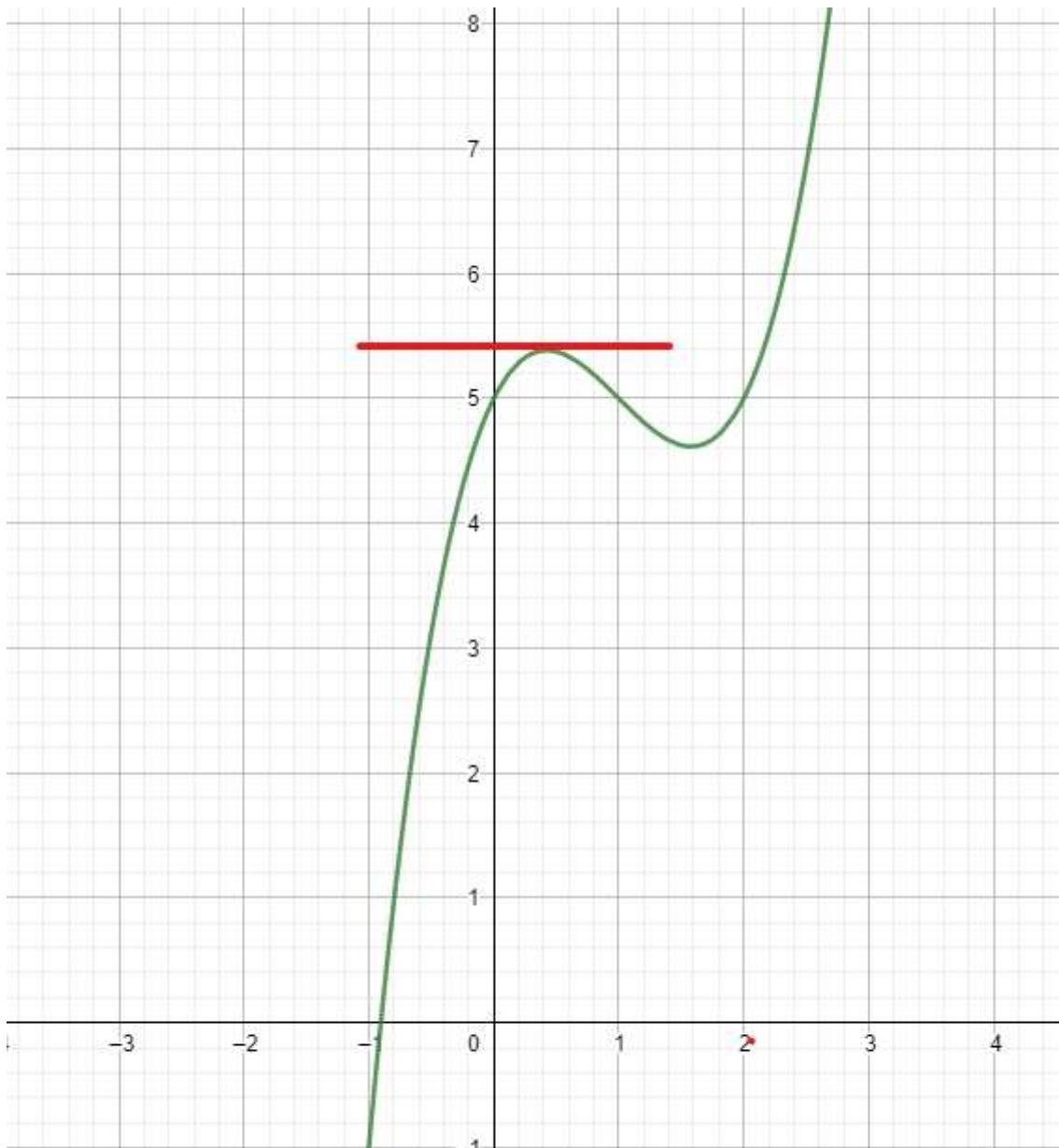
A l'inverse, si l'on se trouve du côté droit de l'axe des ordonnées ($x > 0$), la fonction est croissante, sa tangente a une pente positive, $u * f'(x_n) < 0$ et donc $x_{n+1} < x_n$. Ainsi, on se rapproche de l'axe des ordonnées et donc de $x = 0$

Ainsi, si la fonction possède un minimum, l'algorithme convergera vers l'antécédent du minimum de la fonction sur l'intervalle donné.

6. Test de la méthode du gradient 1D

Pour tester cette méthode, il faut choisir avec attention la valeur de x_0

En effet, rappelons la représentation graphique de la fonction f , mais cette fois-ci sur \mathbb{R} .



D'après le tableau de variation réalisé précédemment, f est croissante jusqu'à $x = 1 - \frac{1}{\sqrt{3}}$,

$$f(1 - \frac{1}{\sqrt{3}}) = 5,38$$

De plus la fonction f tend vers $-\infty$ lorsque x tend vers $-\infty$.

Ainsi, d'après l'algorithme, si f est croissante, la pente de sa tangente est positive. Or comme u est négatif, $u * f'(x_n) < 0$. On a donc $x_{n+1} < x_n$, l'algorithme va donc faire se "déplacer" x_n vers la gauche, il deviendra alors de plus en plus petit et il tendra vers $-\infty$

Il faut donc prendre $x_0 > 1 - \frac{1}{\sqrt{3}}$ pour que l'algorithme converge vers le minimum sur $[0,3]$, car f est décroissante à partir de cette valeur.

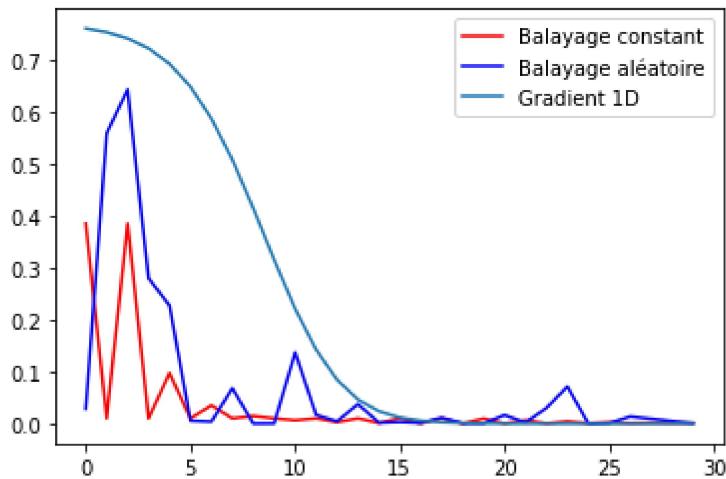
Ainsi, en prenant $x_0 = 0,5$, toujours avec $N = 100$, l'algorithme converge vers $f(x_n) = 4,6150998$

Comparaison avec les autres méthodes

	Balayage à pas constant	Balayage aléatoire	gradient 1D	Valeur réelle
Minimum obtenu	4,615379	4,624059	4,6150998	4,6150998

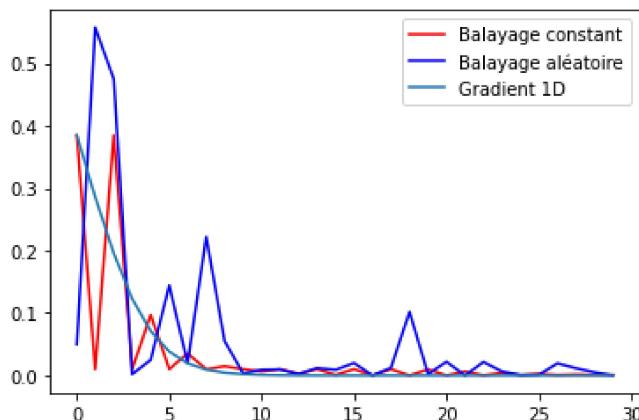
Ainsi, on remarque que la méthode du gradient 1D permet d'obtenir un résultat plus proche de la valeur réelle que les autres méthodes.

Traçons à nouveau les courbes d'erreurs :



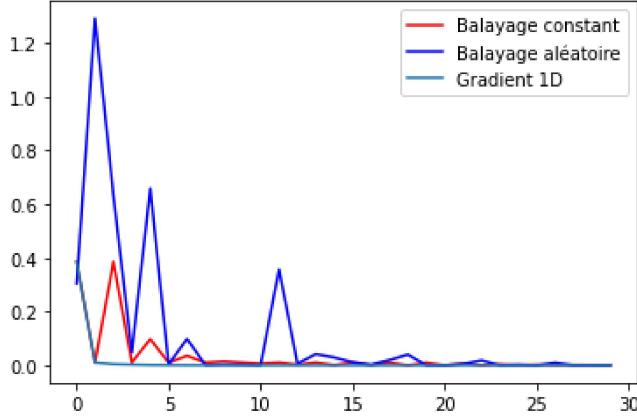
On utilise le gradient avec $u = -0,1$, on remarque que pour une petite valeur de N, la méthode du gradient 1D est moins précise que les deux autres. Cependant, pour N plus grand ($N \approx 15$ sur le graphique), la méthode du gradient est la plus précise.

En prenant $x_0 = 1$, on obtient :



Dans ce cas, la méthode du gradient 1D converge plus vite, cela s'explique par le fait que 1 est plus proche de l'antécédent du minimum que $x_0 = 0.5$ utilisé précédemment.

En prenant cette fois-ci $u = 0.5$, avec $x_0 = 1$, on obtient :



L'algorithme converge encore plus vite.

En conclusion, la méthode du gradient 1D possède des avantages mais aussi des inconvénients. En effet, les paramètres "x0" et "u" influent grandement sur le résultat de la méthode :

- une mauvaise valeur de x0 empêchera l'algorithme de converger
- le choix de u et de x0 influe grandement sur la vitesse de convergence de l'algorithme

7. Amélioration de la méthode du gradient 1D

Afin d'améliorer la méthode, on peut essayer de déterminer à chaque étape la longueur de pas optimale, c'est-à-dire la longueur de pas optimale.

On définit la fonction $\varphi(t) = f(x_n + t \frac{df}{dx}(x_n))$

Calculons ensuite sa dérivée $\frac{d\varphi}{dt}$

Rappel : dérivée de $g \circ f$

$$(f \circ g)' = f'(g(x)) * g'(x)$$

$$\text{En posant } g(t) = x_n + t \frac{df}{dt}(x_n)$$

- Calcul de $\frac{d\varphi}{dt}$

$$\text{On a donc } \frac{d\varphi}{dt} = \frac{df}{dt}(x_n + t \frac{df}{dt}(x_n)) * \frac{df}{dt}(x_n)$$

On évalue ensuite cette dérivée en 0 :

$$\frac{d\varphi}{dt}(0) = \frac{df}{dt}(x_n) * \frac{df}{dt}(x_n) = (\frac{df}{dt}(x_n))^2$$

- Calcul de $\frac{d^2\varphi}{dt^2}$

$$\frac{d^2\varphi}{dt^2} = \frac{d}{dt} (\frac{d}{dt} f(x_n + t \frac{df}{dt}(x_n))) * \frac{df}{dt}(x_n)$$

$$\frac{d^2\varphi}{dt^2} = \frac{d^2f}{dt^2}(x_n + t \frac{df}{dt}(x_n)) * (\frac{df}{dt}(x_n))^2$$

Qu'on évalue ensuite aussi en 0

$$\frac{d^2\varphi(0)}{dt^2} = \frac{d^2f}{dt^2}(x_n) * (\frac{df}{dt}(x_n))^2$$

Approchons maintenant $\frac{d\varphi}{dt}$ par son développement limité à l'ordre 1 en 0.

Rappel : Développement limité à l'ordre 1

Si f admet un développement limité à l'ordre 1 en x_0 alors on a :

$$f(x) = f(x_0) + f'(x_0)(x - x_0) + (x - x_0)\varepsilon(x)$$

$$\text{D'où } \frac{d\varphi}{dt} = \frac{d\varphi}{dt}(0) + t \frac{d^2\varphi}{dt^2}(0) + x\varepsilon(x)$$

$$\frac{d\varphi}{dt} = \frac{df}{dt}(x_n) * t \frac{df}{dt}(x_n) + \frac{d^2f}{dt^2}(x_n) * (\frac{df}{dt}(x_n))^2(x) + x\varepsilon(x)$$

On établit alors le tableau de variation de φ :

t	$-\frac{1}{f''(x_n)}$		
$\frac{df}{dx}$	-	0	-
$f(x)$			

Si on prend $t = -\frac{1}{f''(x_n)}$, on obtient alors un pas optimal.

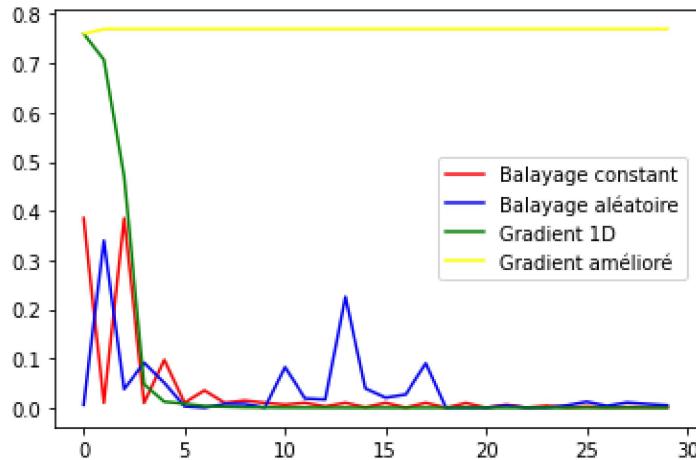
On retrouve alors la méthode de Newton.

Reprendons alors notre fonction $f(x) = x^3 - 3x^2 + 2x + 5$

On a $f''(x) = 6x - 6 = 6(x - 1)$.

En prenant $-\frac{1}{f''(x_n)}$ comme pas, on remarque une erreur : si $x_n = 1$, alors la fonction n'est pas définie.

Si l'on prend $x_0 = 0.5$, on obtient :

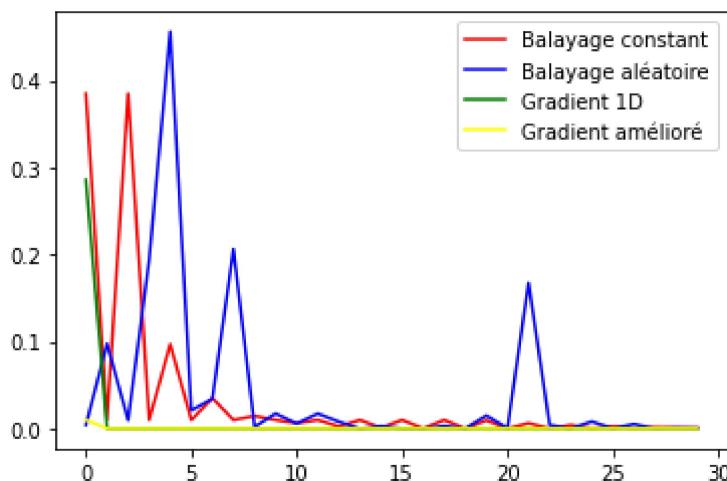


On remarque que le gradient amélioré n'a pas convergé. La valeur de x_n trouvée en fin

d'algorithme est $x_n = 0.42264973081037427 \simeq 1 - \frac{1}{\sqrt{3}}$. Or $f(1 - \frac{1}{\sqrt{3}}) = 5.38$

La méthode a en réalité convergé vers l'extremum le plus proche de l'image de x_0 , il s'agit, pour $x_0 = 0.5$, d'un maximum local (voir tableau de variation). La méthode du gradient amélioré n'a donc pas convergé vers la solution voulue.

Si l'on prend $x_0 = 1.1$, on obtient :



L'algorithme a cette fois-ci convergé vers la solution voulue, et ce bien plus vite que les autres méthodes.

En conclusion, la nouvelle méthode du gradient permet de trouver à la fois les maximums et les minimums, en choisissant avec soin la valeur de x_0 . De plus, elle converge plus vite que les autres méthodes, mais nécessite tout de même de calculer la dérivée seconde de la fonction à étudier.

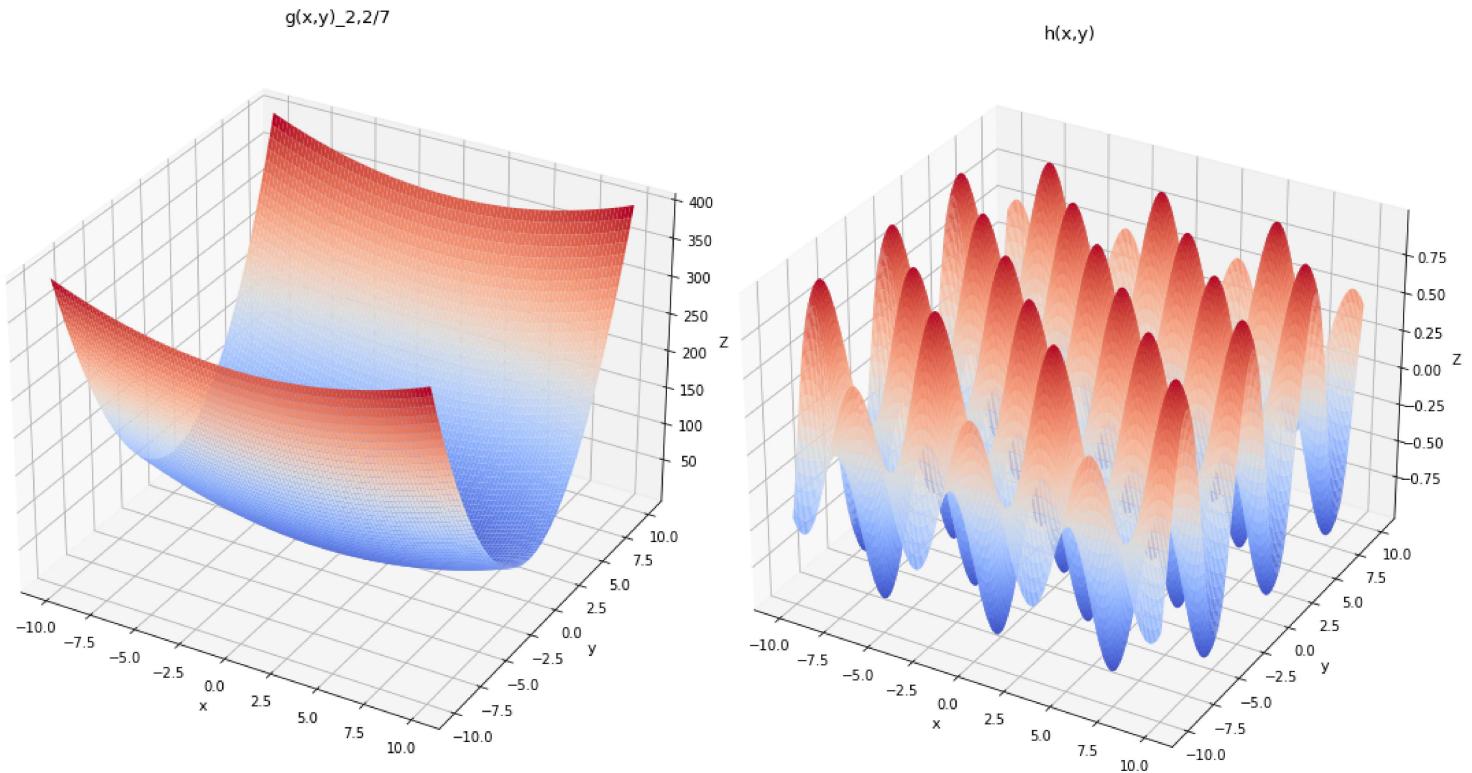
C. Optimisation d'une fonction de deux variables réelles

Soit les fonctions g et h tel que :

- $g_{a,b}(x,y) = \frac{x^2}{a} + \frac{y^2}{b}$
- $h(x,y) = \cos(x)\sin(y)$

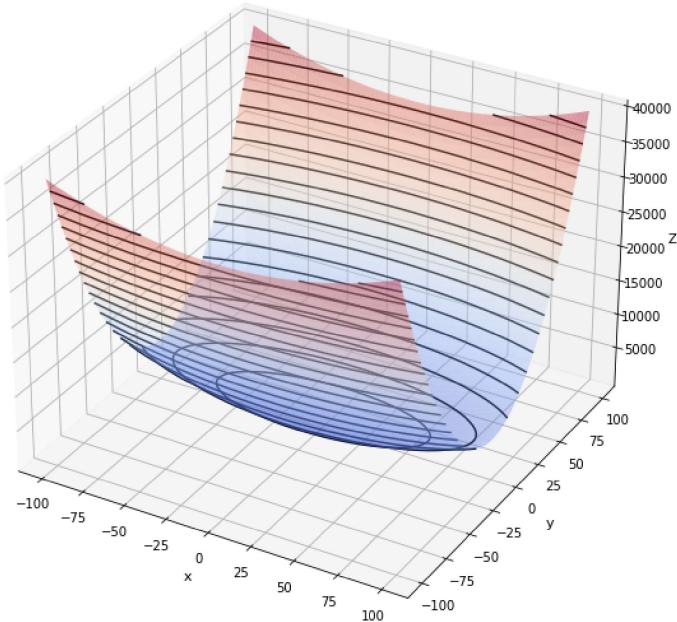
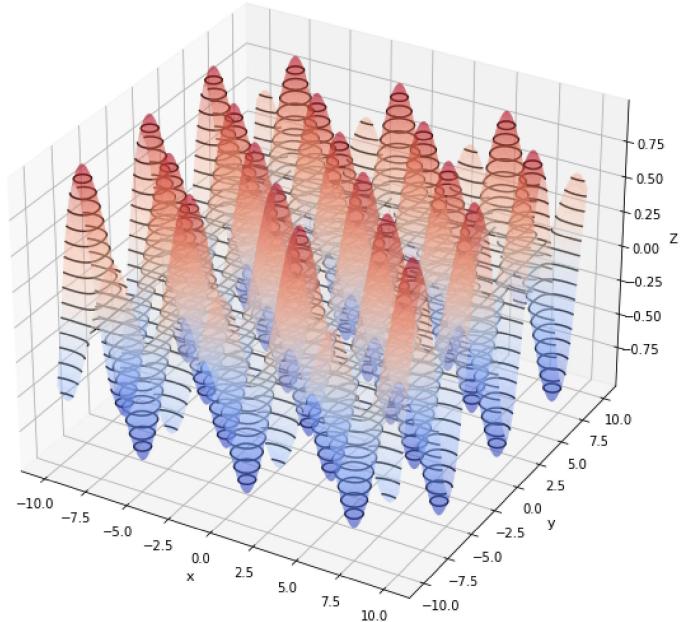
1. Représentation graphique

Représentons graphiquement ces fonctions, avec $a = 2$ et $b = \frac{2}{7}$:

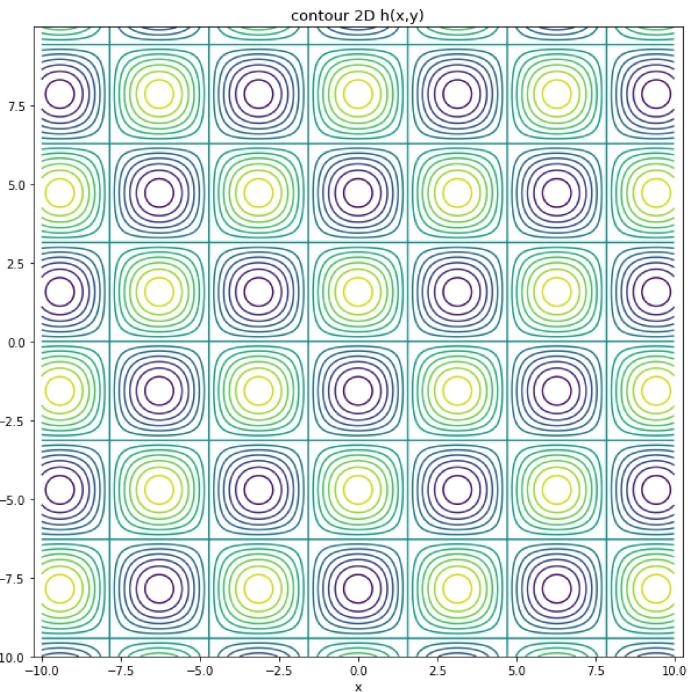
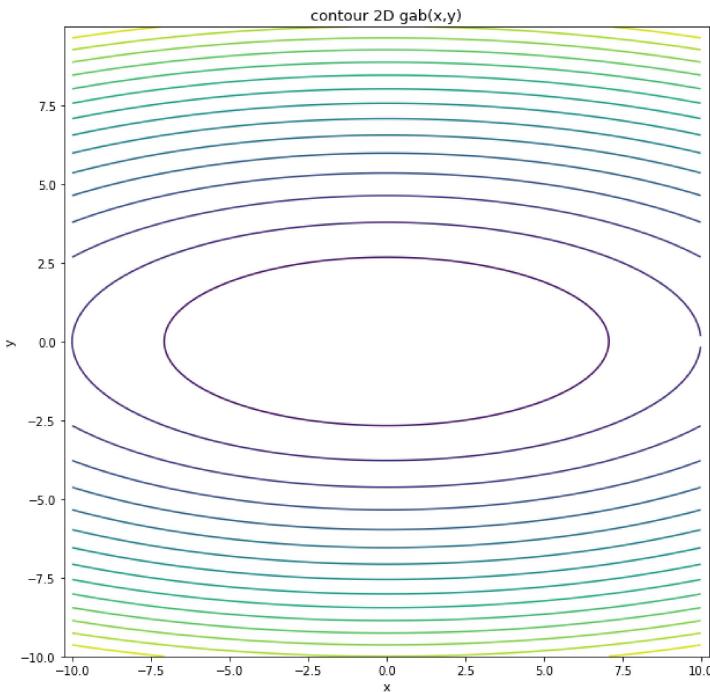


2. Courbes de niveau

Ainsi que leurs courbes de niveau (3D) :

contour $g(x,y)$ 2,2/7contour $h(x,y)$ 

Ainsi que leurs courbes de niveau (2D) :



Grâce à ces différents graphiques, on peut remarquer que :

- Pour la fonction g :

On peut lire graphiquement grâce aux courbes de niveau que le minimum se trouve aux alentours de 0. Après rapide analyse, on trouve effectivement que le minimum de $g_{a,b}$ est

obtenu pour le couple $(0; 0)$ et est 0. Cette fonction peut se voir comme la somme de deux paraboles : $g_{a,b}(x, y) = g_a(x) + g_b(y)$ avec $g_a(x) = \frac{x^2}{a}$ et $g_b(y) = \frac{y^2}{b}$

Donc, peu importe le signe de x et y , si $|x|$ ou $|y|$ est grand, l'image du couple par la fonction sera grande : elle tend vers l'infini. La fonction n'a pas de maximum.

- Pour la fonction h :

Son maximum est 1 et son minimum est -1. Ces valeurs sont obtenues de manières récurrentes. En effet, on trouve une alternance de minimum et de maximum sur les graphiques. Or, on sait que les fonctions cosinus et sinus sont périodiques et comprises entre -1 et 1.

Donc, après une rapide analyse, on obtient $\max(h(x, y))$ lorsque $\cos(x) = 1$ et $\sin(y) = 1$ ou $\cos(x) = -1$ et $\sin(y) = -1$. On en déduit donc que $\max(h(x, y))$ est obtenu pour les couples $(x = 2k\pi ; y = \frac{\pi}{2} + 2k'\pi)$

ou $(x = \pi + 2k\pi ; y = -\frac{\pi}{2} + 2k'\pi)$ avec k et k' des entiers.

De même, $\min(h(x, y))$ lorsque $\cos(x) = -1$ et $\sin(y) = 1$ ou $\cos(x) = 1$ et $\sin(y) = -1$. On en déduit donc que $\min(h(x, y))$ est obtenu pour les couples $(x = \pi + 2k\pi ; y = \frac{\pi}{2} + 2k'\pi)$ ou $(x = 2k\pi ; y = -\frac{\pi}{2} + 2k'\pi)$ avec k et k' des entiers

C'est donc l'alternance de ces 4 couples périodiques qui crée les différents pics sur le graphique.

3. Calcul des gradients

Définition du gradient

En tout point (x, y) le vecteur $\nabla f(x, y)$ est perpendiculaire à la surface de niveau passant par ce point. De plus, il est dirigé suivant la direction de variation la plus rapide de f , dans le sens des valeurs croissantes de f .

$$\text{Rappel : } \nabla f(x, y) = \left[\frac{df}{dx}; \frac{df}{dy} \right]^t$$

Gradient de $g_{a,b}(x, y)$

$$\frac{dg_{a,b}}{dx} = \frac{d}{dx} \left(\frac{x^2}{a} + \frac{y^2}{b} \right) = \frac{2x}{a}$$

$$\frac{dg_{a,b}}{dy} = \frac{d}{dy} \left(\frac{x^2}{a} + \frac{y^2}{b} \right) = \frac{2y}{b}$$

$$\nabla g_{a,b}(x, y) = \left[\frac{2x}{a}, \frac{2y}{b} \right]^t$$

Soit, avec $a = 2$ et $b = \frac{2}{7}$:

$$\nabla g_{2, \frac{2}{7}}(x, y) = [x; 7y]^t$$

Gradient de $h(x, y)$

$$\frac{dh}{dx} = \frac{d}{dx} (\cos(x)\sin(y)) = -\sin(x)\sin(y)$$

$$\frac{dh}{dy} = \frac{d}{dy} (\cos(x)\sin(y)) = \cos(x)\cos(y)$$

$$\nabla h(x, y) = [-\sin(x)\sin(y); \cos(x)\cos(y)]^t$$

4. Calcul de gradients et de leurs normes en quelques points

Norme du gradient

Le gradient étant un vecteur, on calcule sa norme en utilisant simplement la formule de la norme euclidienne.

$$\|\nabla f(x, y)\| = \left\| \left[\frac{df}{dx}, \frac{df}{dy} \right]^t \right\| = \sqrt{\frac{df}{dx}^2 + \frac{df}{dy}^2}$$

Evaluations en quelques points

- $g_{2, \frac{2}{7}}(x, y)$

x	y	$\frac{dg_{2, \frac{2}{7}}}{dx} = x$	$\frac{dg_{2, \frac{2}{7}}}{dy} = 7y$	$\ \nabla g_{2, \frac{2}{7}}(x, y)\ $
0	0	0	0	0
7	1,5	7	10.5	12.619429464123963
10	10	10	70	70.71067811865476
-10	-10	-10	-70.0	70.71067811865476
-54	20	-54	140.0	150.05332385522155
42	58	42	406.0	408.16663263917104
100	-26	100	-182.0	207.66318884193223

- $h(x, y)$

x	y	$\frac{dh}{dx} = -\sin(x)\sin(y)$	$\frac{dh}{dy} = \cos(x)\cos(y)$	$\ \nabla h(x, y)\ $
0	0	0	1	1
7	1,5	-0.6553408384880418	0.05332893580321819	0.6575071026111531
10	10	-0.295958969093304	0.704041030906696	0.7637181971034437
-10	-10	-0.295958969093304	0.704041030906696	0.7637181971034437
-54	20	-0.5101438083076915	-0.3384264664769138	0.6121921090366359
42	58	0.9099891763055343	-0.04767030401785035	0.9112369389343142
100	-26	-0.3861333986607683	0.5578507404915458	0.6784515091207932

5. Méthode du gradient à pas constant

Elle consiste à fixer un pas u et un point de départ (x_0, y_0) et à calculer les termes :

$$(x_{n+1}, y_{n+1}) = (x_n, y_n) + u \nabla f(x_n, y_n)$$

On arrête le calcul lorsque le nombre d'itération est trop grand (la méthode n'a pas convergé) ou lorsque $\|\nabla f(x, y)\| < \varepsilon$, avec ε une précision donnée. Dans ce cas, on a sans doute un point critique.

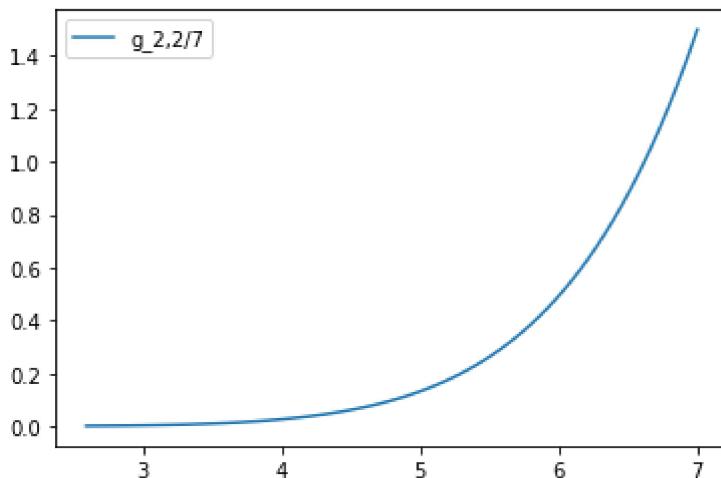
En d'autres termes, on utilise le gradient de sorte à se déplacer de manière itérative dans la direction de la descente la plus raide telle que définie par $u \nabla f(x_n, y_n)$ avec $u < 0$.

L'algorithme fonctionne donc selon la même logique que le gradient 1D de la partie B, mais en utilisant cette fois-ci des fonctions à plusieurs variables.

6. Représentation graphique du gradient à pas constant

- $g_{2,\frac{2}{7}}(x, y)$ avec $(x_0; y_0) = (7; 1.5)$

- $N = 100$ et $u = -0.01$

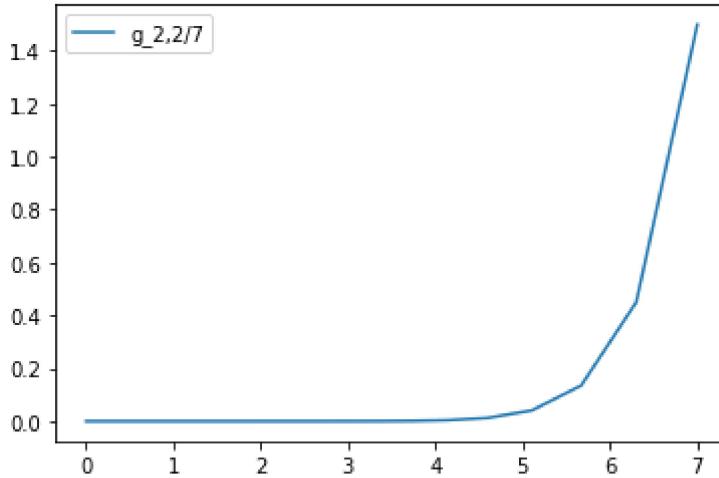


Le couple donné en fin d'algorithme est : $(2.56222639; 1.05775750e^{-3})$

$$g_{2,\frac{2}{7}}(2.56222639; 1.05775750e^{-3}) = 3.2825059499983427$$

Le couple obtenu n'est pas satisfaisant, on obtient un minimum trop grand par rapport à la valeur réelle.

- $N = 100$ et $u = -0.1$



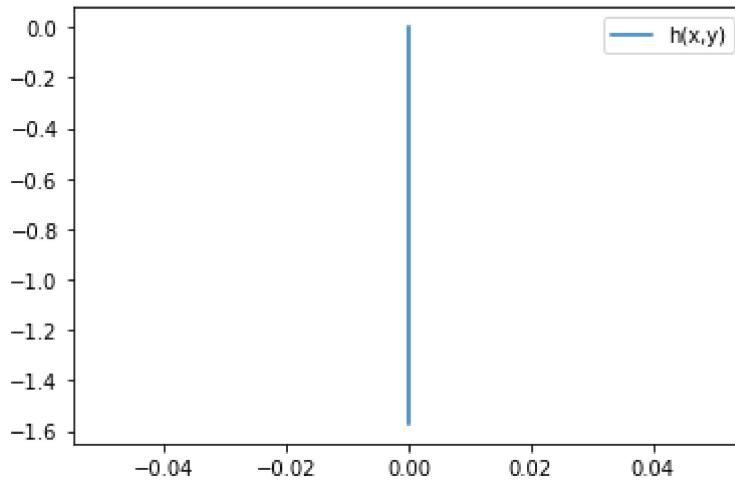
Le couple donné en fin d'algorithme est : $(1.85929792e^{-4}; 7.73066281e^{-53}) \approx (0; 0)$
 $g_{2,\frac{2}{7}}(1.85929792e^{-4}; 7.73066281e^{-53}) = 1.728494381620552e^{-8} \approx 0$

On trouve cette fois-ci une valeur satisfaisante : bien qu'elle ne soit pas exacte, il s'agit d'une bonne valeur approchée.

On remarque en comparant au graphique précédent que le choix de u influe également fortement sur le résultat, tout comme avec le gradient 1D de la partie B. En effet, lorsque l'on calcule $u\nabla f(x_n, y_n)$, avec u négatif, on se déplace dans le sens inverse des variations de la fonction. Plus u est petit, plus on se déplacera vite dans ce sens, expliquant ainsi la différence de vitesse de convergence entre les deux essais.

De plus, prendre N plus grand améliorera la précision du résultat, mais n'influera pas sur la vitesse de convergence.

- $h(x, y)$ avec $(x_0; y_0) = (0; 0)$
 - $N = 100$ et $u = -0.01$



Le couple final renvoyé par l'algorithme est $(0; -1.57070284)$

En évaluant la fonction, on a $h(0; -1.57070284) = -0.9999999956304325 \simeq -1$. L'algorithme a bien convergé vers une des solutions possibles, à savoir la solution la plus proche du point initial $(0; 0)$.

En effet, il s'agit d'une solution de la forme $(x = 2k\pi; y = -\frac{\pi}{2} + 2k'\pi)$ avec $k = 0$ et $k' = 0$.

$$-\frac{\pi}{2} \simeq -1.57070284$$

L'autre solution possible serait $(x = \pi + 2k\pi; y = \frac{\pi}{2} + 2k'\pi)$ soit $(\pi; \frac{\pi}{2})$

Or $\|(0; -\frac{\pi}{2})\| < \|(\pi; \frac{\pi}{2})\|$. $(0; -\frac{\pi}{2})$ est bien la solution la plus proche de $(0; 0)$.

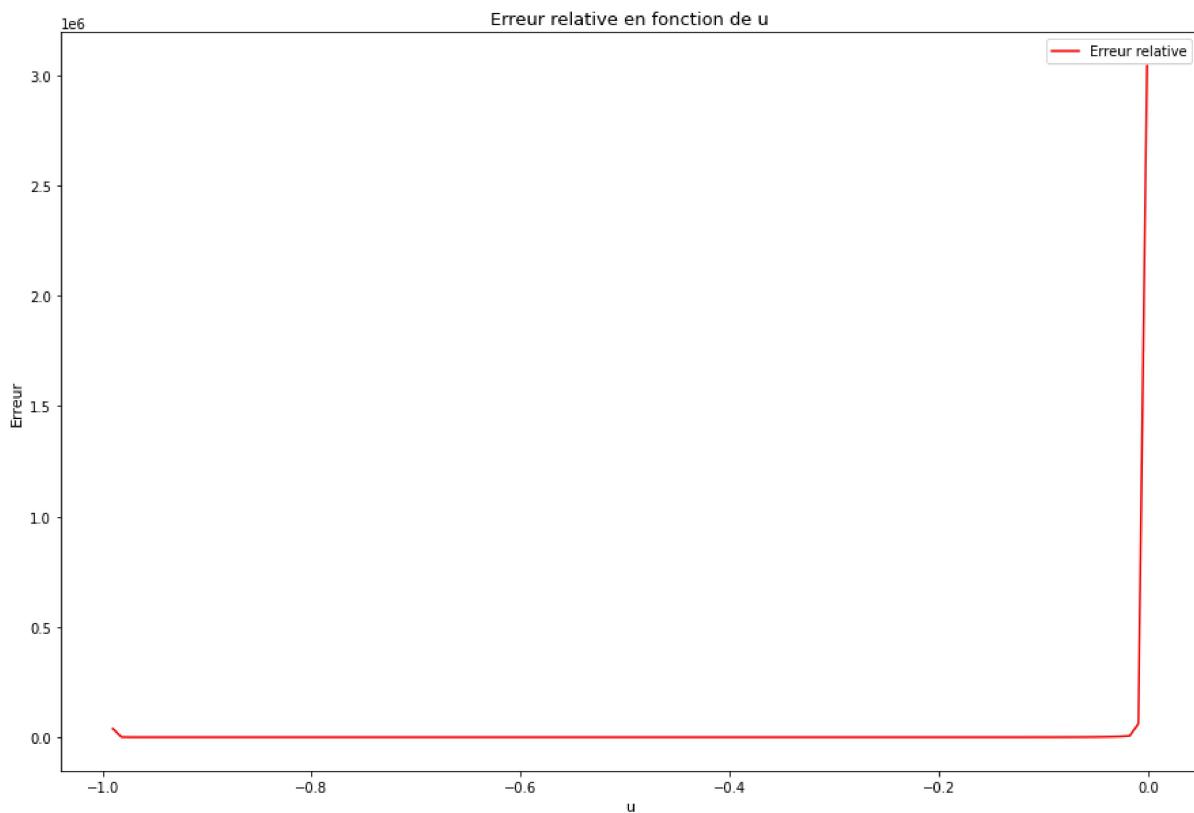
Ainsi, l'algorithme ne modifie pas la valeur de x , puisqu'elle est initialement à la valeur voulue, expliquant pourquoi on obtient une droite d'abscisse 0 sur la courbe ci-dessus.

7. Erreur relative

Etudions maintenant la fonction $g_{1,20}(x, y) = \frac{x^2}{1} + \frac{y^2}{20}$

$$\nabla g_{1,20}(x, y) = \left[2x; \frac{2y}{20} \right]^t$$

On sait par ailleurs que le min de $g_{1,20}(x, y)$ est atteint pour $g_{1,20}(0, 0) = 0$ et l'erreur relative est donnée par la formule : $\frac{(valeur \text{ r\'eelle} - valeur \text{ obtenue})}{valeur \text{ r\'eelle}}$ Or la valeur réelle du minimum est ici de 0, par conséquent on l'approximera par $\epsilon = 10^{-5}$. On peut ainsi tracer la courbe d'erreur en fonction de $u \in [-0.99; -0.001]$, avec $N = 120$, N étant le nombre d'itération :

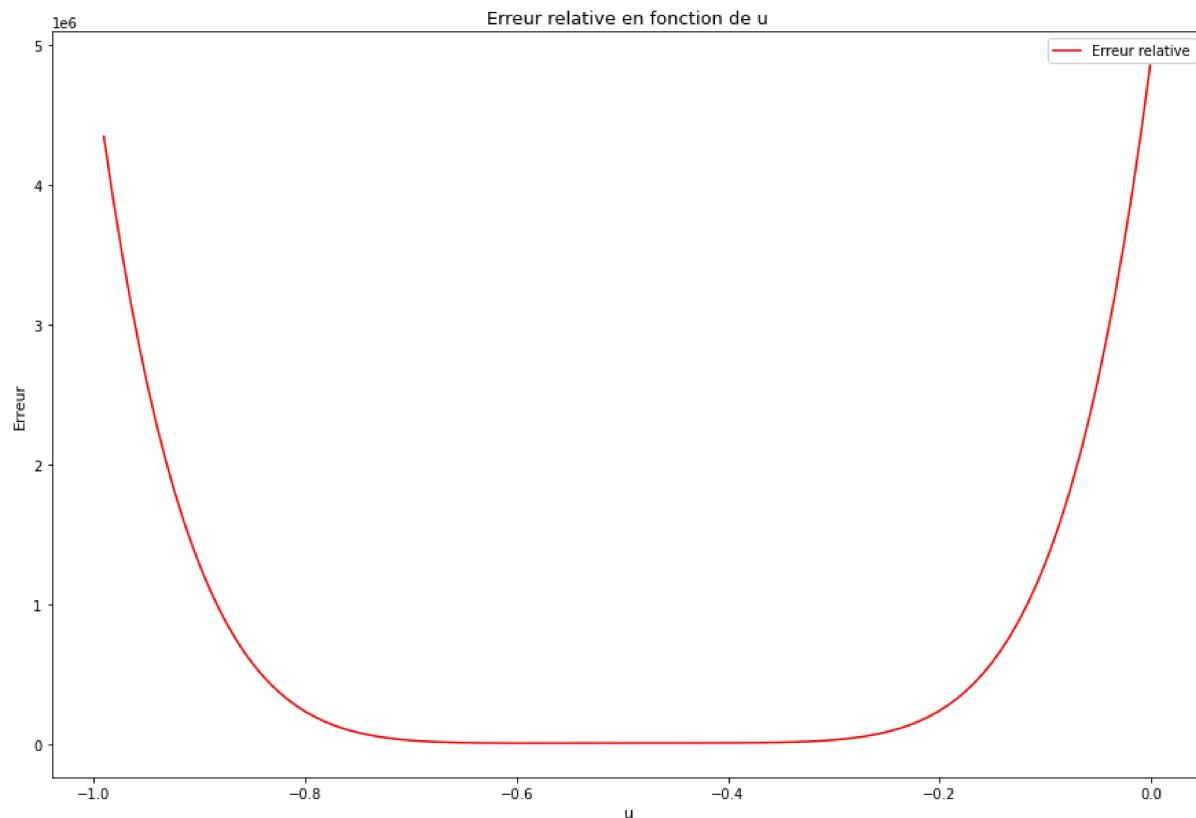


On sait que prendre une grande valeur de N améliore la précision du résultat. Or dans cette partie, on veut étudier la vitesse de convergence de la méthode en fonction du pas u. On essaye donc de minimiser ici l'effet de N, c'est pourquoi on choisit une petite valeur de N.

On remarque sur le graphique que l'erreur relative devient élevée lorsque u tend vers 0 et vers -1. Cela s'explique par le fait que l'algorithme n'a pas le temps de converger vers la solution voulue, il a été limité par N.

Cependant, l'erreur relative est faible - quasi-nulle pour u compris entre -0.95 et -0.05. Cela signifie que pour ces valeurs de u, la méthode du gradient à pas constant a convergé plus vite.

Prenons maintenant N=3:



L'erreur relative est maintenant faible entre -0.7 et -0.3 environ : les valeurs non comprises dans cette intervalle ne permettent pas à l'algorithme de converger vers la solution voulue en seulement 3 itérations.

On peut donc conclure qu'il existe un pas u optimale qui permet à la fonction de converger rapidement vers la solution voulue.

8 - 9. Méthode du gradient amélioré

On modifie maintenant la méthode du gradient à pas constant. On cherche maintenant le plus grand pas possible dans la direction du gradient pour se rapprocher de l'extremum. On fixe un pas u et un point de départ (x_0, y_0) , puis, à chaque nouvelle itération, on calcule les termes :

- $F_1 = f((x_n, y_n)) + ku\nabla f(x_n, y_n)$

- $F_2 = f((x_n, y_n) + (k + 1)u \nabla f(x_n, y_n))$

Lors de la recherche d'un minimum, on augmente k tant que $F_2 < F_1$ car on se rapproche du minimum. On utilise ensuite le même critère d'arrêt que la méthode précédente.

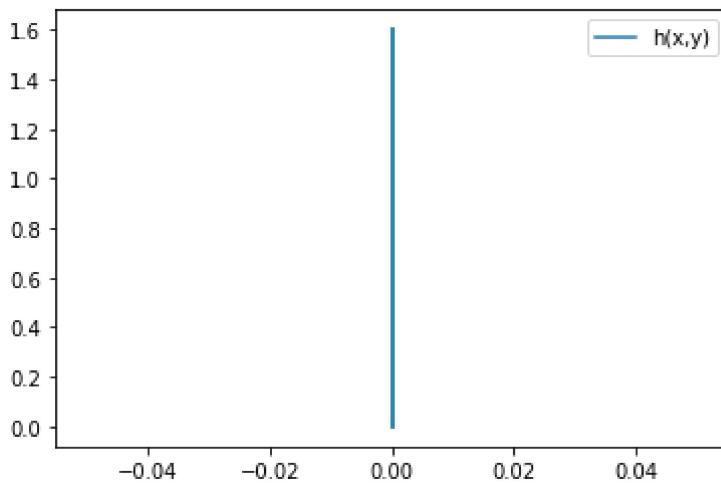
Lors de la recherche d'un maximum, on prend u positif et on augmente k tant que $F_2 > F_1$

- Recherche de maximum

- $g_{2, \frac{2}{7}}(x, y)$ avec $(x_0; y_0) = (7; 1.5)$

On obtient une erreur en fin d'algorithme, cela s'explique simplement par le fait que la fonction n'a pas de maximum.

- $h(x, y)$ avec $(x_0; y_0) = (0; 0)$



Les solutions possibles sont de la forme :

$$\cos(x) = 1 \text{ et } \sin(y) = 1 \text{ avec } (x = 2k\pi; y = \frac{\pi}{2} + 2k'\pi)$$

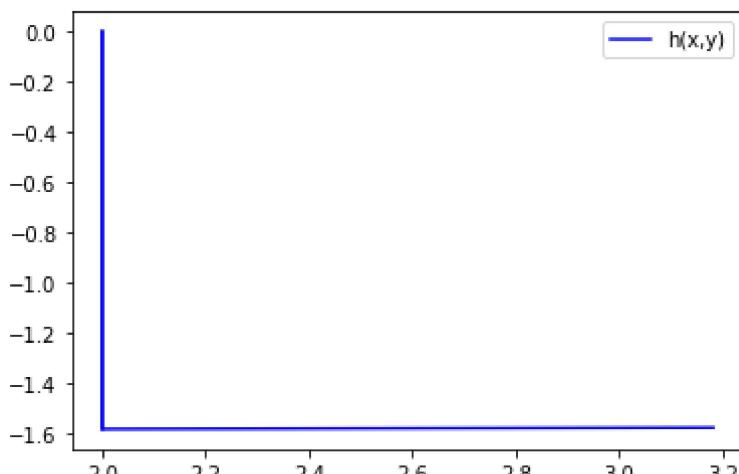
$$\text{ou } \cos(x) = -1 \text{ et } \sin(y) = -1 \text{ avec } (x = \pi + 2k\pi; y = -\frac{\pi}{2} + 2k'\pi)$$

Comme nous avons pris $(x_0; y_0) = (0; 0)$, l'antécédent-solution le plus proche serait : $(x = 0; \frac{\pi}{2})$.

On obtient alors à nouveau une droite d'abscisse 0, avec y qui tend vers $\frac{\pi}{2}$.

- $h(x, y)$ avec $(x_0; y_0) = (2; 0)$

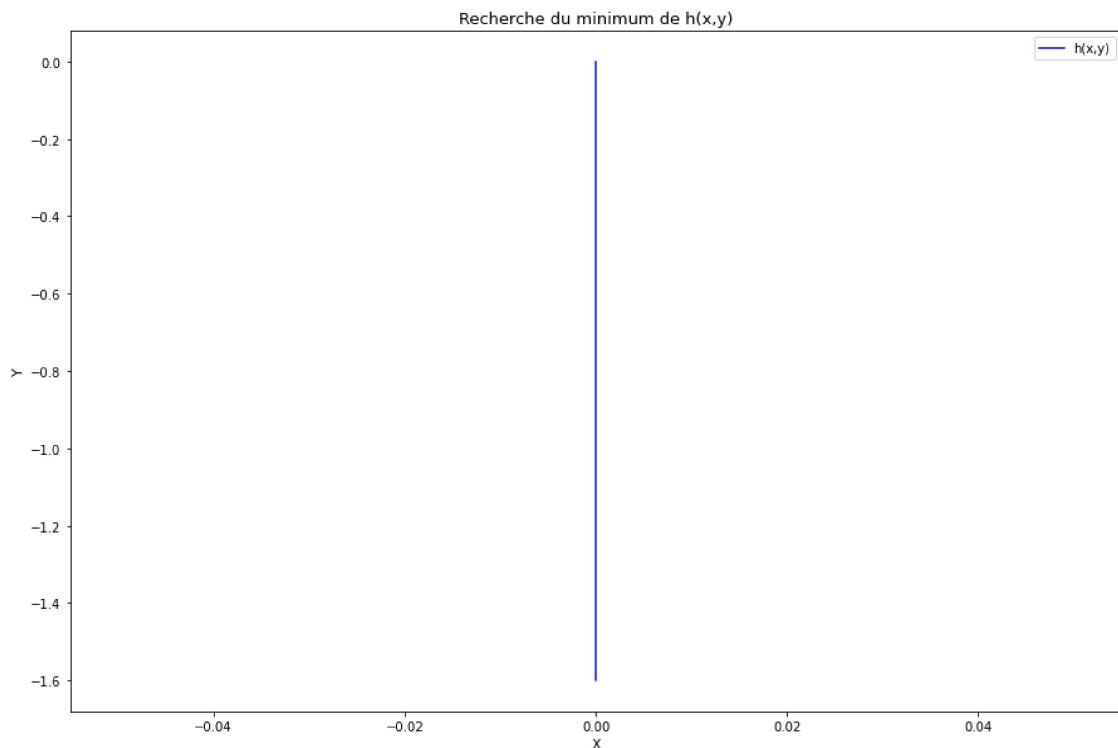
Essayons d'obtenir une autre solution : on prend de nouvelles valeurs $(x_0; y_0)$, et on obtient



On remarque alors que l'algorithme fait d'abord converger y vers la valeur voulue, à savoir $-\frac{\pi}{2}$, puis fait converger x vers π . Nous sommes ainsi parvenus à obtenir un autre maximum en modifiant le couple initial.

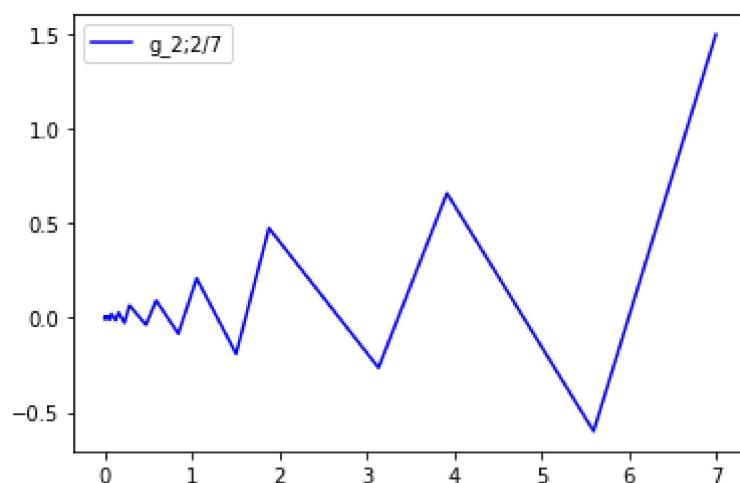
- Recherche de minimum

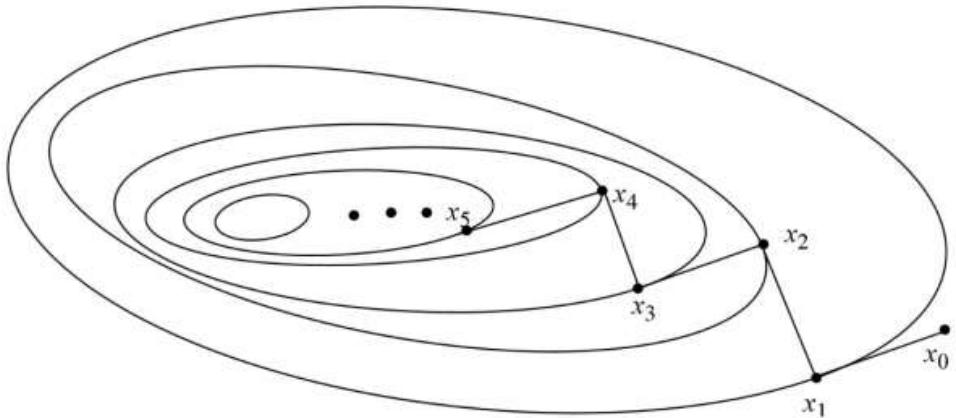
- $h(x, y)$ avec avec $(x_0; y_0) = (0; 0)$



L'algorithme converge bien vers une des solution voulue, à savoir $(0; \frac{\pi}{2})$

- $g_{2,\frac{2}{7}}(x, y)$ avec avec $(x_0; y_0) = (7; 1.5)$ avec $N = 100$

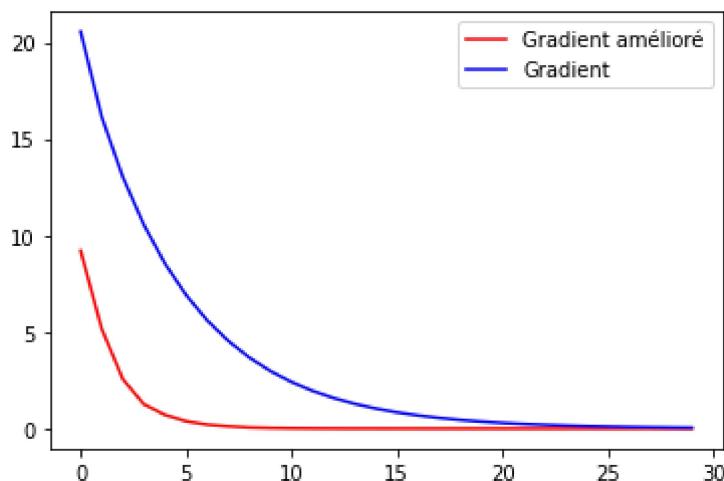




On obtient le couple $(5.58399455e^{-5}; - 7.20208374e^{-6})$

D'où $g_{2,\frac{2}{7}}(5.58399455e^{-5}; - 7.20208374e^{-6}) \approx 0$. L'algorithme a donc bien convergé vers le minimum de la fonction $g_{2,\frac{2}{7}}$.

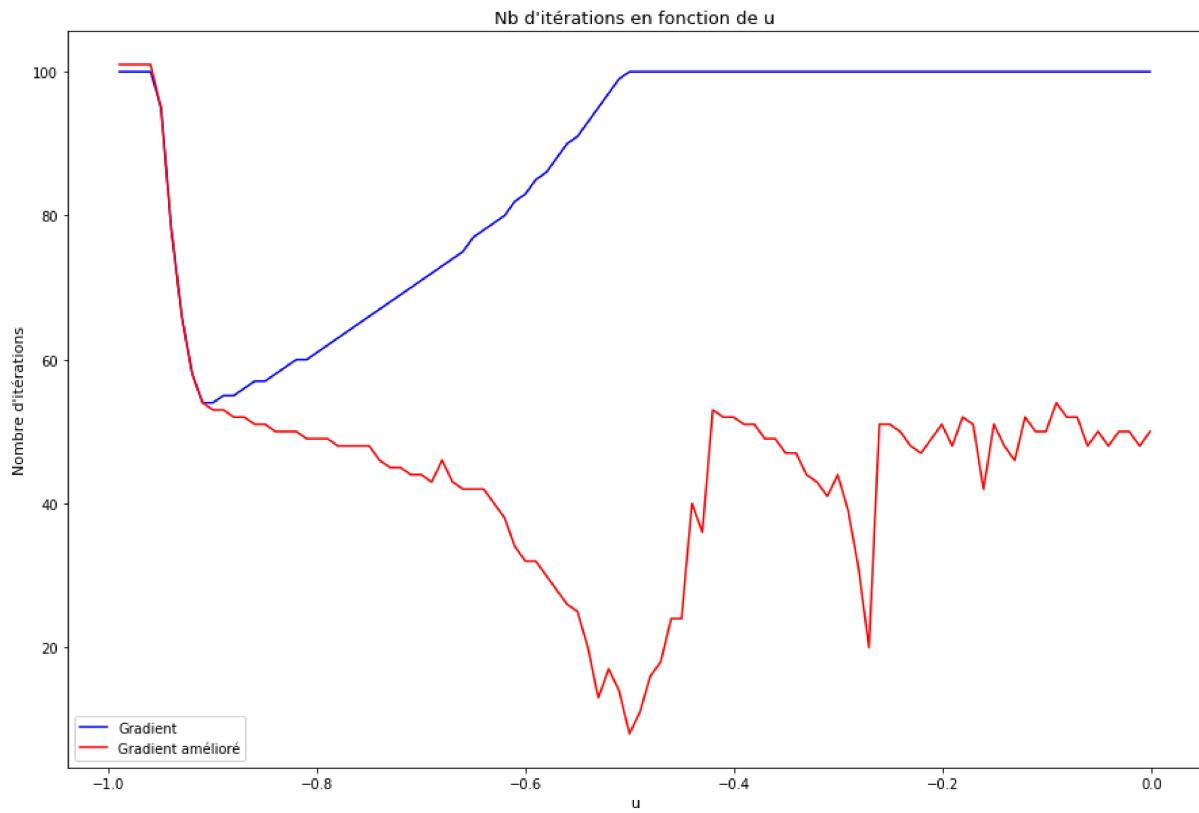
Traçons maintenant les courbes d'erreur absolue afin de comparer la nouvelle méthode avec celle décrite dans la question 5.



On remarque alors que le gradient amélioré converge effectivement plus vite que la méthode du gradient de la partie 5. Il faut environ $N = 5$ pour que le gradient amélioré donne une solution convenable, contre environ 20 pour la première version du programme. Le nouveau gradient est environ 4 fois plus rapide pour ces valeurs de u et ce couple.

10. Comparaison du nombre d'itération

On modifie les méthodes du gradient et du gradient amélioré de manière à ce qu'elles renvoient le nombre d'itérations nécessaires à la détermination du minimum de la fonction g . On regarde ensuite l'évolution de ce nombre d'itérations en fonction de la valeur de la variable $u \in [-0.99 ; -0.001]$:



On remarque alors plusieurs choses, dans un premier temps lorsque u est proche de -0.99 la convergence nécessite plus de 100 itérations, quand u augmente on voit alors que les deux méthodes sont sensiblement équivalentes jusqu'à environ -0.9 à partir de ce moment là la performance du gradient amélioré devient bien meilleure que celle du gradient. En particulier aux alentours de -0.5 où le gradient amélioré converge particulièrement rapidement tandis que le gradient quant à lui nécessite plus d'une centaine d'itérations pour converger. Pour des valeurs de u supérieures à -0.5 le gradient amélioré nécessite ~60 itérations tandis que le gradient normal en prend toujours plus de 100. En regardant cet exemple, on peut conclure que le gradient amélioré est au moins supérieur au gradient normal en termes de vitesse de convergence.

D. Calcul de l'inverse d'une matrice

Soit A une matrice symétrique définie positive, $b \in R^n$ et Φ la fonction définie sur R^n par
 $\Phi = y^T * Ay - 2y^T b$

L'algorithme de gradient à pas optimal est donné par :

- $y_{k+1} = y_k - p_k G(y_k)$
- $p_k = \frac{\|G(y_k)\|^2}{2G(y_k)^T AG(y_k)}$ si $G(y_k) \neq 0$

- $p_k = 0$ sinon

où $G(y_k) = 2(Ay - b)$ est le gradient de Φ au point y .

On peut montrer que la résolution de $Ax = b$ est équivalente à la détermination de $x \in R^n$ minimisant Φ .

1. N=1

1.

$$\Phi = y^T * Ay - 2y^T b$$

On écrit $y^T = Y = y$

$$\Phi = AY^2 - 2Yb$$

En dérivant selon y on obtient $\Phi = 2AY - 2b = G(Y)$

En dérivant une seconde fois, on obtient : $\Phi = 2a$

$$\text{Comme } G(y_k) \neq 0, p_k = \frac{1}{2a} = \frac{1}{\Phi(y)^{''}}$$

$$\text{Donc notre algorithme devient } y_{k+1} = y_k - \frac{\Phi(y)^{'}}{\Phi(y)^{''}}.$$

On retrouve alors l'algorithme de Newton de la question 7 de la partie B.

2. Programme de calcul

Dans le programme que nous avons réalisé nous avons choisis arbitrairement de définir Y_n comme un array remplis de 3 de longueur n (le résultat ne dépendant pas de la valeur de Y_n). A est une matrice définie symétrique positive de taille n et b un array de taille 2.

$$A = \begin{pmatrix} 12 & 5 \\ 5 & 12 \end{pmatrix} \quad b = (-6, 9)$$

3. Test de l'algorithme

Pour $n = 2$:

Posons A et b tels que :

$$A = \begin{pmatrix} 12 & 5 \\ 5 & 12 \end{pmatrix} \quad b = (-6, 9)$$

on trouve manuellement :

$$12x + 5y = -6 \quad \Rightarrow \quad x = -117/119 \sim -0.9839$$

$$5x + 12y = 9 \quad \Rightarrow \quad y = 138/119 \sim 1.1597$$

et on obtient en utilisant l'algorithme :

$$[-0.98319285 \quad 1.15966406]$$

Pour n=3 :

Posons A et b tels que :

$$A = \begin{pmatrix} 12 & 5 & 1 \\ 5 & 12 & 1 \\ 1 & 1 & 12 \end{pmatrix} \quad b = (-6, 9, 1)$$

on trouve manuellement :

$$\begin{aligned} 12x + 5y + z &= -6 & \Rightarrow & x = -698/707 \sim -0.98727 \\ 5x + 12z &= 9 & \Rightarrow & y = 817/707 \sim 1.1556 \\ x + 12z &= 1 & \Rightarrow & z = 7/101 \sim 0.069307 \end{aligned}$$

et on obtient en utilisant l'algorithme :

$$[-0.98726986 \quad 1.1555863 \quad 0.06930686]$$

Conclusion :

L'algorithme semble converger vers la solution d'un système de n équations avec une précision satisfaisante.

E. Application à des problèmes de transfert de la chaleur

On s'intéresse ici à l'équation différentielle suivante :

$$(\star) \quad \forall x \in]0, 1[\quad -\frac{d^2T}{dx^2}(x) + c(x)T(x) = f(x); \quad T(0) = a, \quad T(1) = b,$$

avec a et b deux réels. Notre but consiste à déterminer une fonction T qui soit une solution approchée de cette équation (en admettant qu'une telle solution existe).

1. On considère une barre cylindrique dont les parois latérales sont parfaitement calorifugées

On suppose que depuis un temps très long la température des extrémités est maintenue aux valeurs suivantes :

- $T(0)=a$
- $T(1)=b$

On prend ici $a = 500K$ et $b = 350K$.

On est ici en régime permanent, on a donc $c = 0$ et $f = 0$

Le système devient :

$$-\frac{d^2T}{dx^2} = 0$$

On approxime alors $\frac{d^2T}{dx^2}$ par $\frac{T_2 - 2T_1 + T_0}{dx^2}$

$$\text{D'où } \frac{T_2 - 2T_1 + T_0}{dx^2} = 0$$

.

.

.

$$\frac{d^2T}{dx^2} = -\frac{T_{i+1} - 2T_i + T_{i-1}}{dx^2} = 0$$

.

.

.

$$\frac{T_{N+1} - 2T_N + T_{N-1}}{dx^2} = 0$$

Or $T_{N+1} = b$ et $T_0 = a$

D'où, sous forme matricielle $\frac{1}{dx^2} * \text{tridiag}(-1, 2, -1) * (U_1, \dots, U_N)^T = (\frac{a}{dx^2}, 0, \dots, 0, \frac{b}{dx^2})^T$

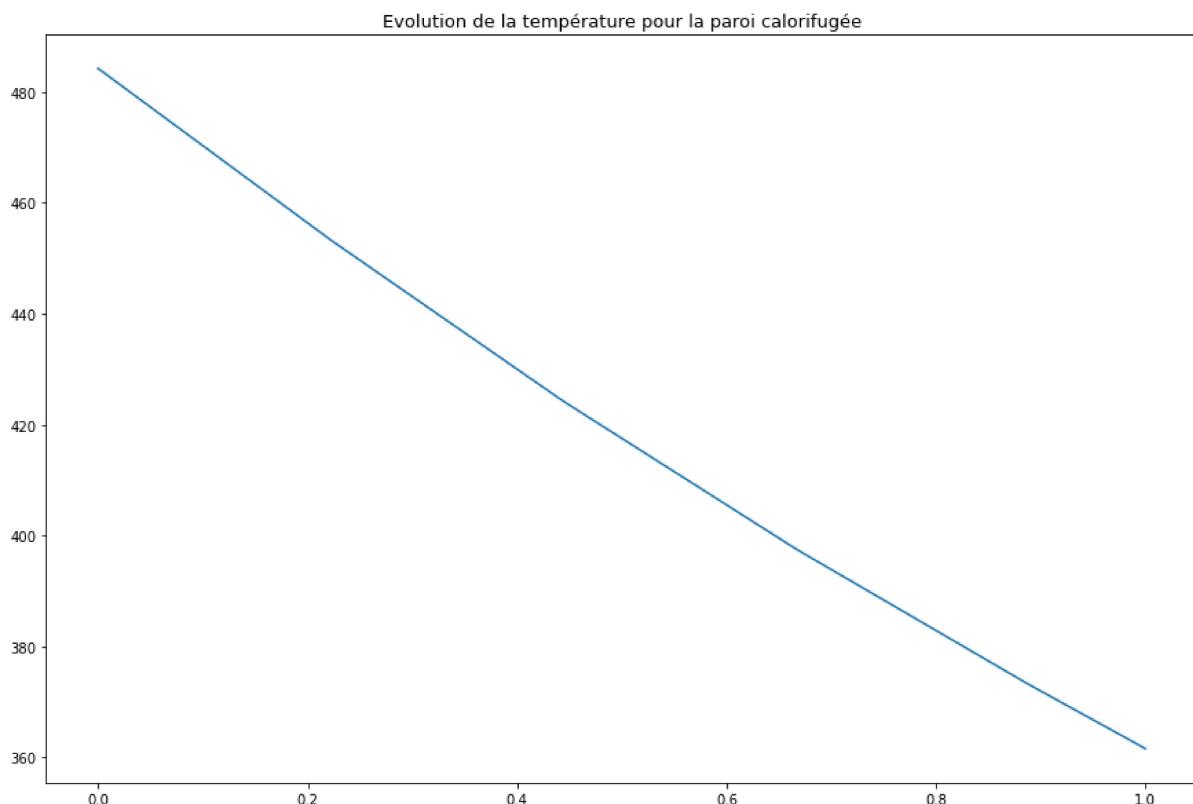
Soit $\frac{1}{dx^2} AY = \frac{1}{dx^2} b$

D'où $Y = A^{-1}b$

Avec $A = \text{tridiag}(-1, 2, -1)$ et $b(a, 0, \dots, 0, b)^T$

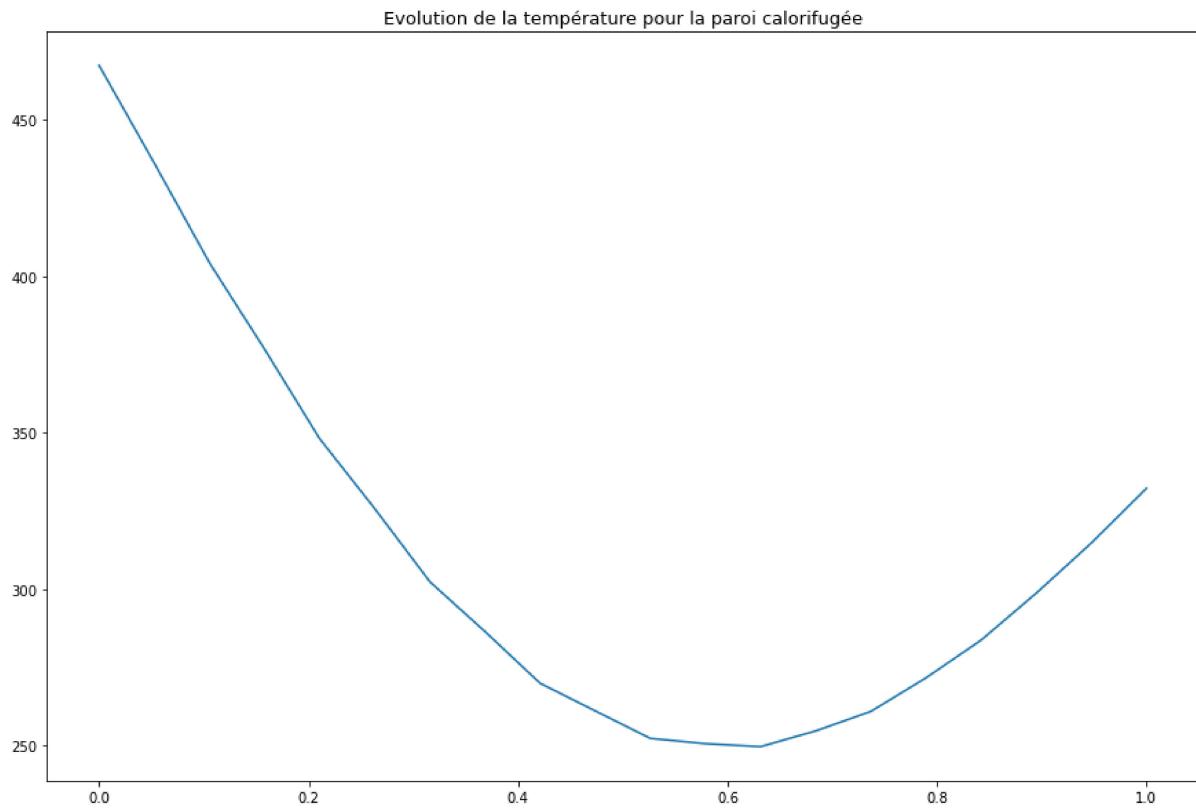
On utilise alors l'algorithme de la partie précédente permettant de résoudre un système.

On obtient, pour $N = 10$

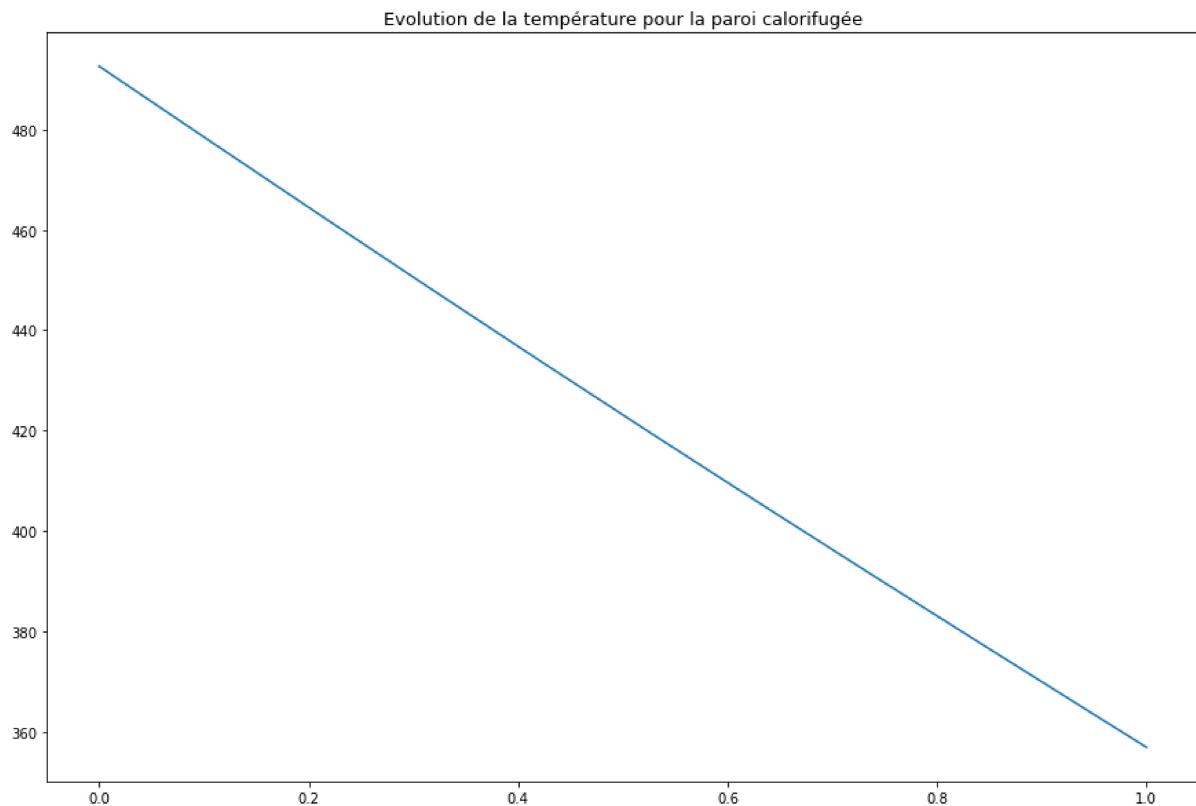


On obtient bien une droite.

En revanche, en augmentant N, on obtient :



On remarque alors que l'algorithme n'a pas convergé vers la bonne solution, on augmente alors le nombre d'itération maximal et on obtient, pour N=20 et m = 500.



2. On considère maintenant que les parois ne sont plus parfaitement calorifugées

Maintenant c et f ne sont plus nuls

$$\text{On a } c = \frac{2h}{\lambda R} \text{ et } f = \frac{2hT_a}{\lambda R}$$

De plus, on a $D = \sqrt{\frac{\lambda R}{2h}} = \sqrt{0.1}$, $b = 350K$ et $T_a = 300K$

$$\text{D'où } c = \left(\frac{1}{D}\right)^2 \text{ et } f = c * T_a$$

Calculons à nouveau le schéma d'Euler.

$$T_0 = a$$

$$-T_1'' + CT_1 = f$$

.

.

$$-T_i'' + CT_i = f$$

.

.

$$-T_N'' + CT_N = f$$

D'où

$$T_0 = a$$

$$-\frac{(T_2 - 2T_1 + T_0)}{dx^2} + CT_1 = f$$

.

.

$$-\frac{(T_{i+1} - 2T_i + T_{i-1})}{dx^2} + CT_i = f$$

.

.

$$-\frac{(T_{N+1} - 2T_N + T_{N-2})}{dx^2} + CT_N = f$$

D'où $A = \text{tridiag}(-1, 2, -1)$

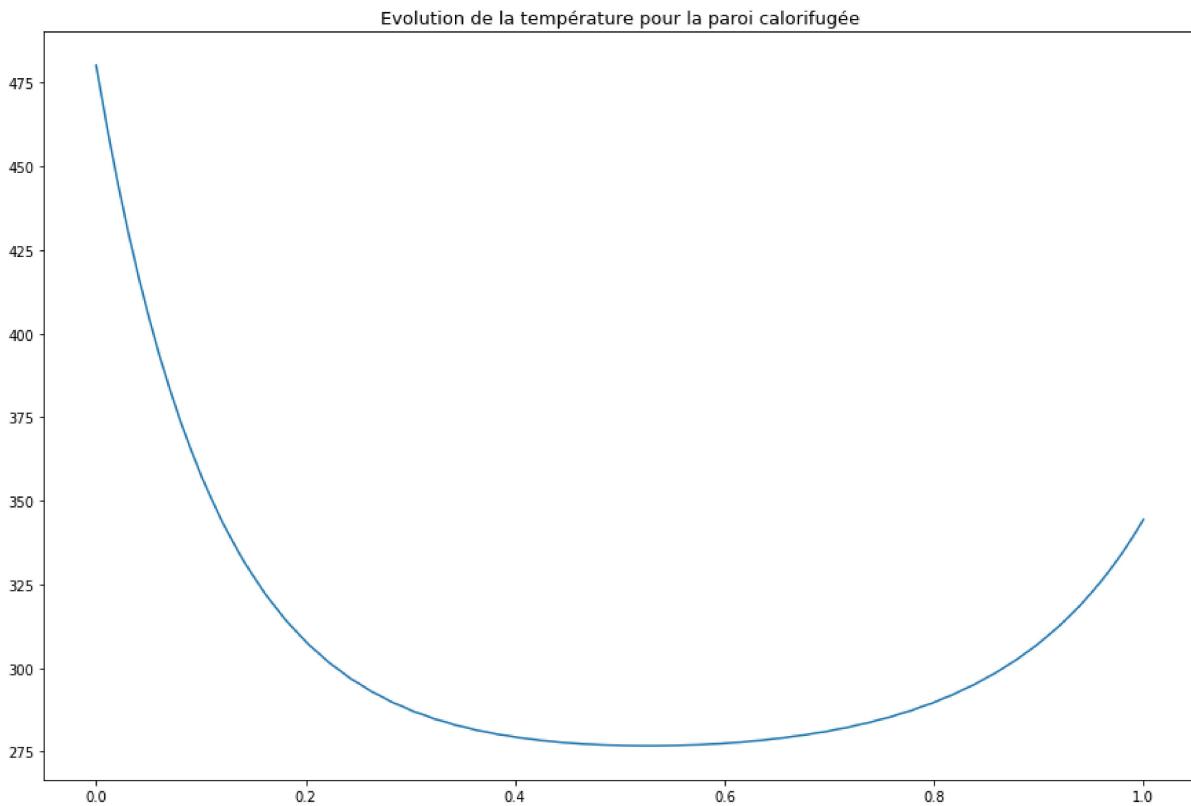
$$B = (dx^2 * f * a, f, \dots, f, dx^2 * f * b)^T$$

$$C = \text{diag}(c)$$

On en déduit le système suivant :

$$(A + dx^2 * C)Y = B$$

On obtient la solution suivante :



F. Pour aller plus loin

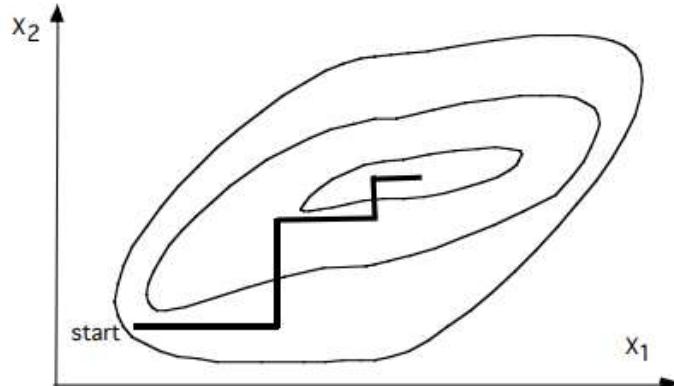
Méthodes numériques pour minimiser une fonction

Recherche sur grille et aléatoire :

Pour illustrer l'accroissement de complexité pour des espaces multidimensionnels, la recherche sur grille peut fournir un exemple. En effet, pour localiser un minimum à 1% de l'espace d'une variable (fonction unidimensionnelle), la recherche sur grille demande 100 évaluations ; si la fonction est 10-dimensionnelle, le nombre d'évaluation requis est de 1020. Donc, cette méthode est inapplicable au-delà de 2 dimensions.

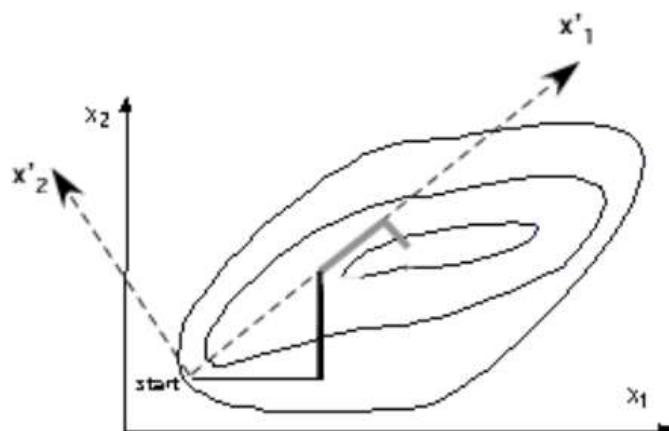
Recherche par variation d'un seul paramètre :

Puisque la condition pour avoir un minimum (avec n variables x_i) est l'annulation de toutes les dérivées partielles $\partial f / \partial x_i$, il est naturel d'essayer d'annuler chacune des dérivées séparément, l'une après l'autre. C'est la vieille méthode de variation du paramètre unique, qui recherche un minimum par rapport à une variable à la fois. Bien sûr, une fois un minimum trouvé sur x_2 , il est probable qu'on ne sera plus au minimum en x_1 , et on devra recommencer le processus, mais il est en général convergent. Si la fonction est une vallée étroite et allongée, la méthode est lente. Un tel comportement est en général inacceptable quand il y a beaucoup de dimensions.



Méthode de Rosenbrock :

Cette méthode commence par des minimisations utilisant les variations des paramètres uniques comme ci-dessus. Quand un cycle complet a été fait sur tous les paramètres, un nouvel ensemble d'axes orthogonaux est défini en prenant comme premier axe le vecteur reliant le point de départ à celui d'arrivée. Cet axe pointe dans la direction des améliorations précédentes et on espère que ce sera une bonne direction pour les futures recherches. Dans le cas de la vallée étroite vue ci dessus, il devrait pointer plus ou moins dans la direction de la vallée et éviter le comportement en zig zag. Le prochain cycle de minimisation par variation de paramètre unique est accompli en utilisant des multiples des axes nouvellement définis comme variables. Cette méthode se comporte généralement bien, elle est stable et capable de suivre des vallées étroites, mais l'efficacité diminue quand le nombre de variables augmente, probablement car le nouvel axe est basé sur des points trop éloignés pour qu'il pointe le long d'une « hypervallée ».



Méthodes de tir

Concernant les méthodes de tir nous avons trouvé les ressources suivantes :

https://www.grenoble-sciences.fr/pap-ebook/grivet/sites/grivet/files/exercices/ch12/ch12ex1_methode_du_tir.pdf

<https://savoirs.usherbrooke.ca/bitstream/handle/11143/4917/MR83667.pdf?sequence=1&isAllowed=y>