Descente de gradient

version du 25/02/2021

Préambule

Cette corbeille d’exercices tutorée est aussi une ressource pour le prosit actuel. Elle a été conçue pour être abordable et avancer pas à pas, de façon appliquée, sur des notions mathématiques. Vous disposez à cet effet de la version étudiante ainsi que de la version corrigée. La version corrigée est présente pour débloquer vos réflexions et vous éviter de passer des heures bloquées sur une notion. Privilégiez donc de façon raisonnable la recherche avec la version non corrigée avant de plonger dans la correction, sous risque de pas comprendre les notions pour ce prosit et les prochaines évaluations. Une masterclass complètera vos interrogations et points de blocage, mais nécessitera que vous ayez fait l’effort de travailler les documents de cette corbeille d’exercices.

Les codes des premières parties sont dans des fichiers python. Vous pouvez les lancer :

* Soit via votre IDE préféré en prenant garde que les bibliothèques nécessaires soient bien installées (la commande pip install devra peut-être être utilisée à cette fin) ;
* Soit en mettant le code dans un notebook jupyter. Ils ont été séparés de la dernière partie qui prend déjà assez de place sur son notebook pour ne pas l’alourdir inutilement.

La partie python a une visée modélisation et algorithmique davantage que programmation en soit. Il sera pour autant intéressant d’explorer les bibliothèques python de [numpy](https://numpy.org/doc/stable/), [pandas](https://pandas.pydata.org/pandas-docs/stable/index.html) et [matplotlib](https://matplotlib.org/stable/contents.html). La meilleure façon d’apprendre à manipuler ces bibliothèques est de se confronter aux erreurs et de tenter de les résoudre en prenant les objets, méthodes, fonctions adéquates et manipuler.

Restez conscient que cette corbeille a une portée pédagogique et constitue une base de réflexion sur la modélisation de la régression linéaire. Il est évident qu’il existe des fonctions optimisées pour réaliser des descentes de gradients et régressions linéaires efficacement. Toutefois, un programmeur peut être amené à modifier des codes pour les adapter à ses données et à ses traitements, ce que cette corbeille va tenter d’initier comme réflexions. L’objet de cette corbeille est donc bien d’appréhender tous ces concepts pour savoir ce que vous faites et réfléchir à ajuster vos algorithmes lorsque vous serez confronté à un certain nombre de problèmes avec vos données.

Descriptif de cette corbeille d’exercices

Cette corbeille d’exercice tutorée a pour but de vous faire comprendre l’algorithme de descente du gradient et son application à la régression linéaire. Dans une première partie, nous nous intéresserons à réaliser quelques rappels sur les dérivées de fonctions à plusieurs variables. Dans une seconde partie, nous discuterons plus en détail de la notion de gradient et ce qu’elle permet de faire en exposant des propriétés fondamentales ainsi que des exemples applicatifs. Dans une troisième partie, nous évoquerons l’algorithme de descente de gradient pour des fonctions convexes de deux variables, avant de l’étendre pour des fonctions convexes à variables. Enfin, nous verrons comment la descente de gradient intervient pour établir une régression linéaire à deux variables que nous réaliserons. Cette dernière partie contient également une ouverture sur la régression linéaire à variables avec une approche plutôt matricielle, même si l’algorithme du gradient reste exploitable.

Indication sur ce qui est à réaliser ou non par l’étudiant

Les codes python des parties 1 et 2 vous sont donnés et sont directement exploitables.

Les codes python de la partie 3 sont plutôt à travailler avant de les regarder, leur correction est disponible pour ne pas être bloqué.

Le notebook de la dernière partie est aussi directement exploitable et se lit conjointement avec la dernière partie de ce document.

Diverses questions sont soit directement résolues, soit à compléter sur ce document ou avec les codes intervenant dans la partie 3.

Table des matières

[1 Rappels 4](#_Toc65313604)

[A. Fonctions à plusieurs variables 4](#_Toc65313605)

[B. Courbes de niveau 4](#_Toc65313606)

[C. Dérivées partielles d’ordres 1 et 2 6](#_Toc65313607)

[a. Dérivées partielles d’ordre 1 6](#_Toc65313608)

[b. Quelques exemples de dérivées partielles d’ordre 1 6](#_Toc65313609)

[c. Dérivées partielles d’ordre 2 (et supérieures) 7](#_Toc65313610)

[d. Exercices : calculs de dérivées partielles d’ordre 2 8](#_Toc65313611)

[2 Autour du Gradient 10](#_Toc65313612)

[A. Gradient 10](#_Toc65313613)

[a. Définition 10](#_Toc65313614)

[b. Points critiques et extremums 10](#_Toc65313615)

[B. Propriétés géométriques du gradient 12](#_Toc65313616)

[a. Hyperplans tangents en un point 12](#_Toc65313617)

[b. Exemple de l’ellipse 13](#_Toc65313618)

[c. Exercice : tangentes et points critiques 15](#_Toc65313619)

[C. Champ de vecteurs 16](#_Toc65313620)

[3 Descente de gradient 18](#_Toc65313621)

[A. Ressources pédagogiques 18](#_Toc65313622)

[B. Descente de gradient à deux variables 18](#_Toc65313623)

[C. Descente de gradient à variables 20](#_Toc65313624)

[4 Regressions linéaires 22](#_Toc65313625)

[A. Régression linéaire simple 22](#_Toc65313626)

[a. Fonction des moindres carrées pour deux variables 22](#_Toc65313627)

[b. Application : concentration en ozone 23](#_Toc65313628)

[B. Régression linéaire multiple 23](#_Toc65313629)

[a. Transposée d’une matrice 23](#_Toc65313630)

[b. Fonction des moindres carrées pour variables 23](#_Toc65313631)

[c. Application : attaques cardiaques 26](#_Toc65313632)

# Rappels

## Fonctions à plusieurs variables

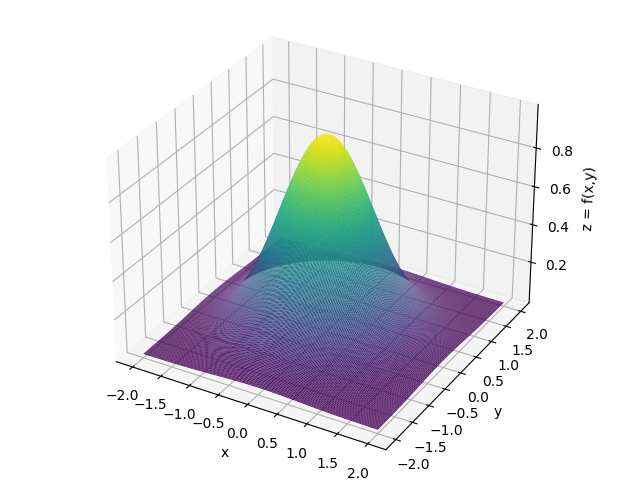
Soit un intervalle.

Une **fonction à plusieurs variables** est une fonction .

## Courbes de niveau

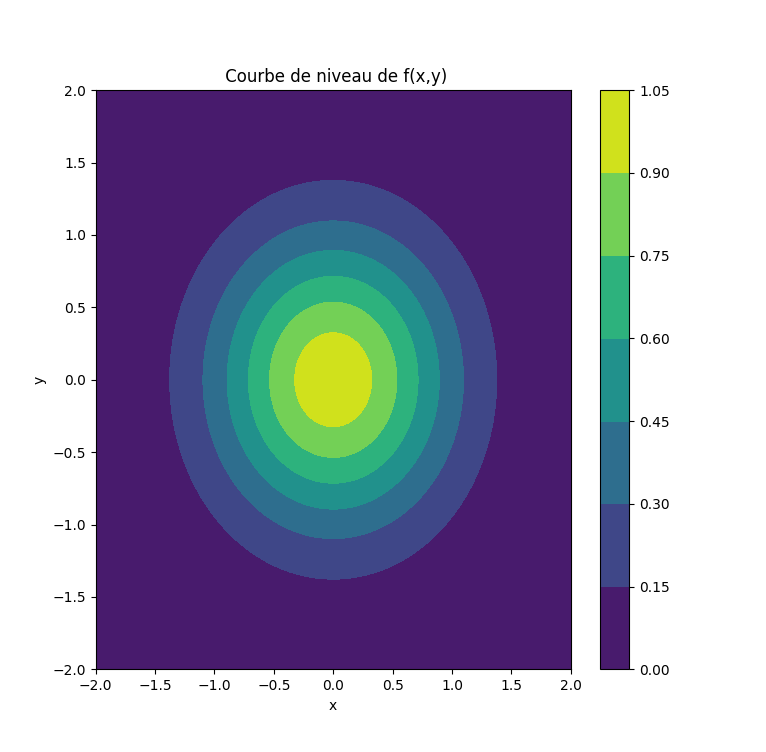
|  |
| --- |
| **Définition.** On considère une fonction à deux variables, une **courbe de niveau de**  représente un ensemble de points qui sont à une même altitude , c’est-à-dire l’ensemble . |

Les courbes de niveau sont notamment utilisées en cartographie pour représenter les différences d’altitude et de reliefs, comme en montagne. Ci-dessous un exemple de modélisation d’une montagne.



*Figure 1 – Représentation graphique de*

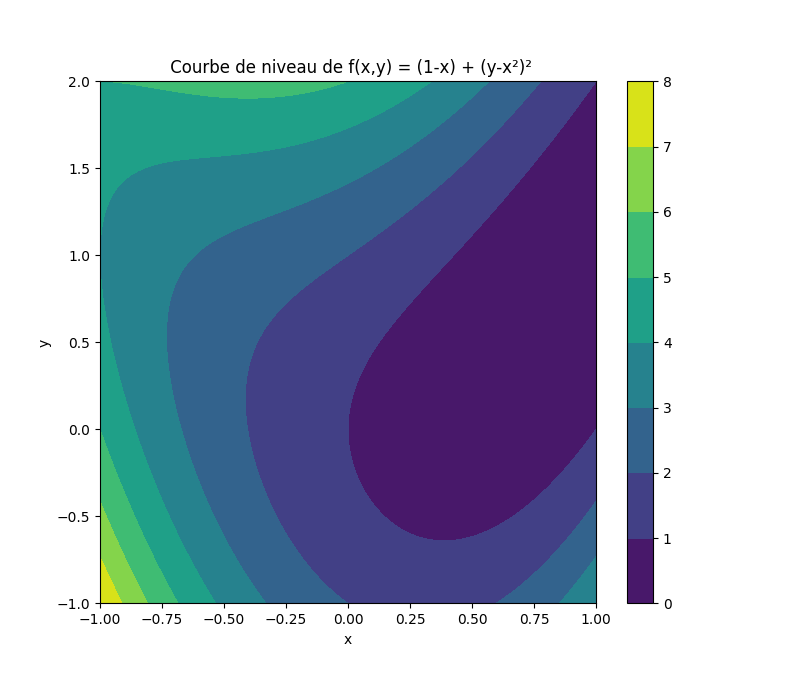
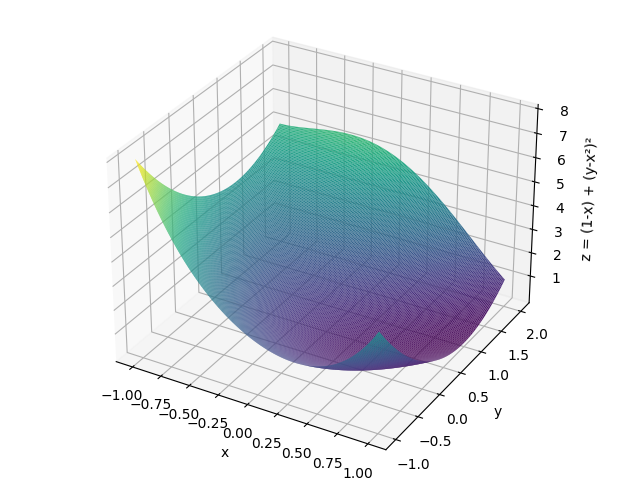
Les courbes de niveau de cette fonction s’obtiennent en projetant les couleurs du graphique vu du dessus suivant le plan . Les points sur un même intervalle d’altitude ont la même couleur, comme le montre la figure 2 suivante :



*Figure 2 – Courbes de niveau de*

On voit bien cet aspect sur la figure 2 représentant la cartographie de la montagne modélisée par la figure 1.

Ci-dessous un autre exemple de courbes de niveau.

****

*Figure 3 – Représentation graphique de et ses courbes de niveau*

## Dérivées partielles d’ordres 1 et 2

### Dérivées partielles d’ordre 1

On sait dériver une fonction à une variable (sur son domaine de dérivabilité), comme par exemple , , …

La dérivée d’une fonction à une variable se note (si elle existe)  ou . Elle est définie par :

La notion de dérivée existe aussi pour des fonctions à plusieurs variables, il s’agit du **gradient** (présenté dans la partie 2). Avant de l’introduire, nous allons définir la notion de dérivée partielle.

La dérivée partielle consiste à dériver une fonction selon l’une de ses variables et uniquement celle-là, on considère dans le calcul de la dérivée partielle que toutes les autres variables pour sont des constantes. Cela permet de réaliser la dérivée partielle de la même façon qu’une dérivée classique à une seule variable.

|  |
| --- |
| **Définition.** Soit une fonction définie de vers . Ses **dérivées partielles** sont notées et définies de la façon suivante : |

Lorsqu’on est dans , il est courant de voir les variables plutôt que .

Lorsqu’on est dans , il est courant de voir les variables plutôt que .

### Quelques exemples de dérivées partielles d’ordre 1

Les termes considérés comme variables dans les dérivées partielles suivantes seront en bleu dans un premier temps tandis que les autres resteront en noir. Cela permet de faire les dérivées partielles comme une dérivée classique à l’aide des tableaux de dérivées connus depuis le lycée.

**Déterminer les dérivées partielles de définie sur :**

**Déterminer les dérivées partielles de définie sur  :**

**Déterminer les dérivées partielles de définie sur  :**

### Dérivées partielles d’ordre 2 (et supérieures)

Une **dérivée partielle d’ordre 2 de**  est la dérivée partielle (selon une variable ) d’une dérivée partielle (selon une variable ) d’une fonction . Elle est notée : .

L’ordre de dérivation est important (et n’aboutit pas toujours au même résultat !), on dérive d’abord selon puis on dérive le résultat obtenu selon . Concrètement :

Si on considère une fonction deux fois dérivable sur , elle a donc quatre dérivées partielles d’ordre 2 si on considère toutes les possibilités de double dérivation partielle :

En général, les deux dernières sont égales si elles vérifient le théorème suivant.

**Théorème de Schwartz :** si est deux fois dérivable en , et continues en alors

Au vu de la définition des dérivées partielles d’ordre 2 :

* Si comprend variables, il y a dérivées partielles ;
* Calculer une dérivée partielle d’ordre 2 revient à calculer deux dérivées partielles d’ordre 1

### Exercices : calculs de dérivées partielles d’ordre 2

**Déterminer les dérivées partielles d’ordre 2 de définie sur  :**

*Indication :* ,

|  |  |
| --- | --- |
| Dérivée d’ordre 1 | |
| On dérive selon  : |  |
| On dérive selon  : |  |
| Dérivée d’ordre 2 | |
| On dérive selon  : |  |
| On dérive selon  : |  |
| On dérive selon  : |  |
| On dérive selon  : |  |

On vérifie bien le théorème de Schwartz sur , donc

En le remarquant avant calcul, on pourrait s’épargner le calcul de l’un des deux connaissant l’autre.

**Déterminer les dérivées partielles d’ordre 2 de définie sur**

est clairement deux fois dérivable et continue sur .

*Indication :* on utilisera les formules suivantes de dérivées composées, attention aux signes !

|  |  |
| --- | --- |
| Dérivée d’ordre 1 | |
| On dérive selon  : |  |
| On dérive selon  : |  |
| Dérivée d’ordre 2 | |
| On dérive selon  : |  |
| On dérive selon  ou selon  :  (c’est pareil grâce au théorème de Schwatz) |  |
| On dérive selon  : |  |

On peut dériver à des ordres supérieurs également, ce qui rend les calculs un peu plus longs, mais on a généralement rarement besoin d’en arriver jusque-là.

# Autour du Gradient

## Gradient

### Définition

|  |
| --- |
| **Définition.** Soit une fonction définie de vers , dérivable sur . Le **gradient** de en un point est un vecteur de dimension qui se note ou . Il est défini de la façon suivante : |

Il s’agit donc d’un vecteur comprenant toutes les dérivées partielles de .

**Remarques :**

* On se place généralement dans le plan ou l’espace, on utilise donc plutôt les notations ou .
* Le symbole s’appelle nabla. Il est assez pratique lorsque l’on manipule gradients, rotationnels, divergences et laplaciens (non abordés ici), et s’utilise comme un opérateur qu’on applique sur une fonction.
* Il arrive que des ressources ne mettent pas de flèche et notent le gradient . La flèche est simplement un indicateur pédagogique pour vous souvenir qu’il s’agit bien d’un vecteur.

Le gradient d’une fonction à plusieurs variables correspond à une extension de la notion de dérivée d’une fonction à une seule variable. Cela va notamment nous permettre d’étudier des extremums et les hyperplans tangents.

### Points critiques et extremums

|  |
| --- |
| **Question 1.** Déterminer le minimum de et en quelle valeur de il est atteint. |
| On remarque que est une somme de carrés donc . s’annule si les carrés sont nuls i.e. soit pour et . On conclut que le minimum de vaut et est atteint en . |

|  |
| --- |
| **Question 2.** Déterminer le gradient de et le point auquel il s’annule. |
| On retrouve ici le point où est minimale puisque et s’annule en ce point. |

Avec ces questions, on a potentiellement un moyen via le gradient de trouver un extremum. Mais en fait, on a eu de la chance, nous avons en réalité trouvé un point critique qui s’avère bien être un minimum.

|  |
| --- |
| **Définition.** Les points où le gradient d’une fonction à plusieurs variables s’annulent sont appelés **points critiques**. Autrement dit : |

Il peut y avoir des extremums aux points critiques (ce n’est pas toujours le cas), les points critiques sont donc des candidats pour les extrema à rechercher. On a la propriété suivante.

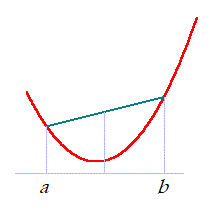
|  |
| --- |
| **Propriété.** |

On aimerait pouvoir combiner cette propriété et la définition précédente pour pouvoir affirmer que rechercher les points où le gradient s’annule permet de trouver les extremums locaux. Mais la propriété précédente n’est pas une double équivalence donc ce n’est pas toujours vrai.

Sous certaines propriétés de , on peut malgré tout avoir l’équivalence. C’est notamment le cas lorsque est **convexe.**

|  |
| --- |
| **Définition.** Une fonction est **convexe** si ses cordes (segments reliant deux points et de ) sont toujours au-dessus de sa représentation graphique entre ces deux points et . |

La figure 4 illustre un exemple de fonction convexe à une variable.

**

*Figure 4 – Fonction à une variable convexe (corde au-dessus du graphe de )*

|  |
| --- |
| **Propriété.** Si est une fonction convexe, alors : |

Remarque : on a la propriété similaire pour concave avec les maximums locaux, mais comme nous nous intéresserons par la suite à la descente de gradient, nous ne l’évoquerons pas plus en détail.

## Propriétés géométriques du gradient

### Hyperplans tangents en un point

|  |
| --- |
| **Définition.** Soit un espace vectoriel de dimension . Un **hyperplan** de est un sous-espace vectoriel de de dimension . |

Par exemple :

* Les hyperplans de sont les droites ;
* Les hyperplans de sont les plans.

|  |
| --- |
| **Propriétés.** Soient un point et une fonction à plusieurs variables. Alors :   * est orthogonal à la courbe de niveau passant par ; * indique la direction de plus forte pente vers laquelle aller pour augmenter le plus vite la valeur de . |

Pour être plus précis, est le vecteur normal à l’hyperplan tangent à en . On peut alors déterminer l’équation d’un hyperplan tangent en grâce à la propriété suivante.

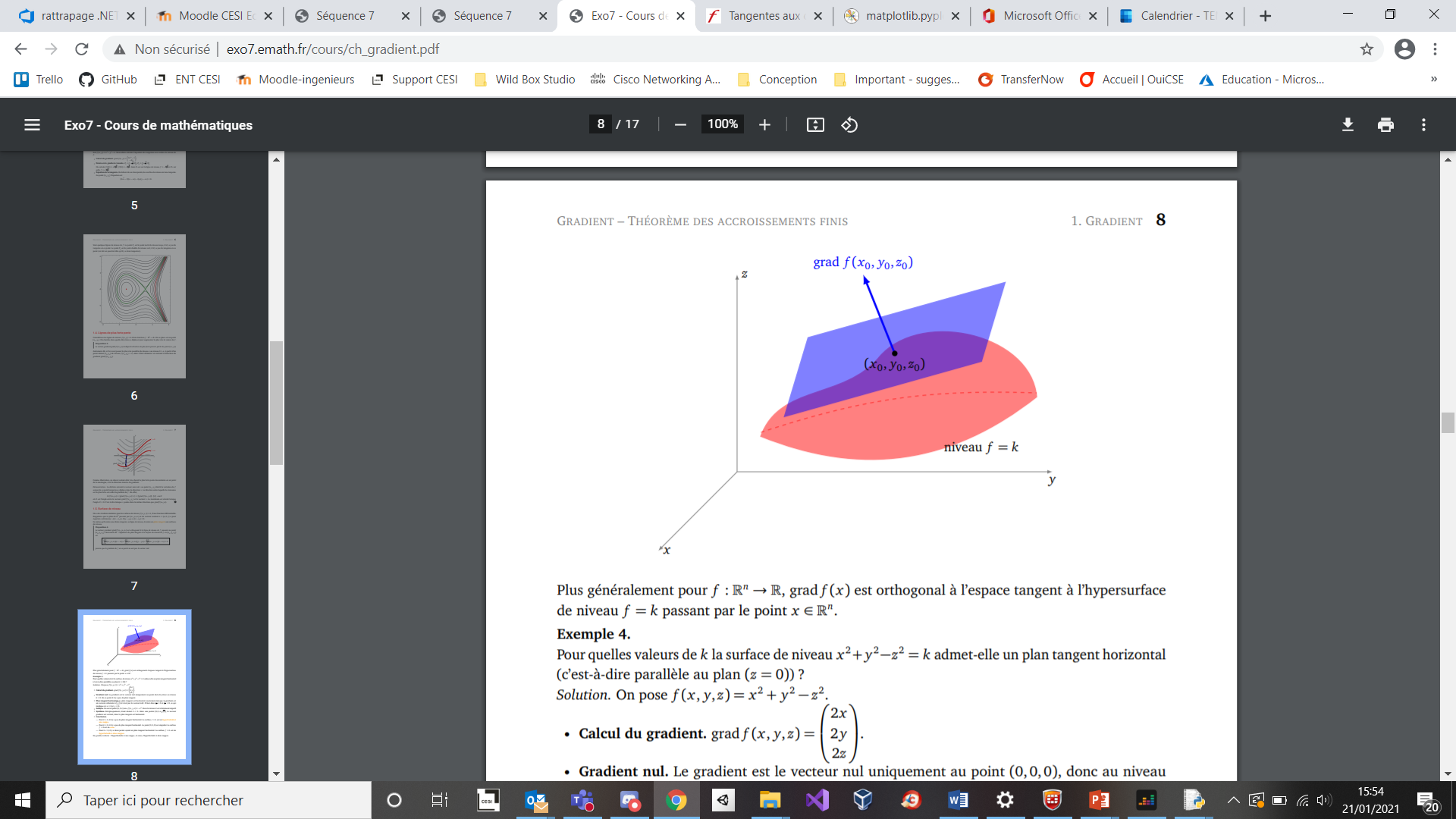
|  |
| --- |
| **Propriété.** Soit une fonction et un point. Si n’est pas un point critique, alors l’ensemble des points de l’hyperplan tangent à en vérifient . |

Si est à deux variables, il s’agit d’une tangente en un point .

Si est à trois variables, il s’agit d’un plan tangent en un point (voir figure 5).

D’après la propriété précédente, on a donc :

|  |
| --- |
| **Equation de la tangente à dans le plan :**  **Equation du plan tangent à dans l’espace :** |

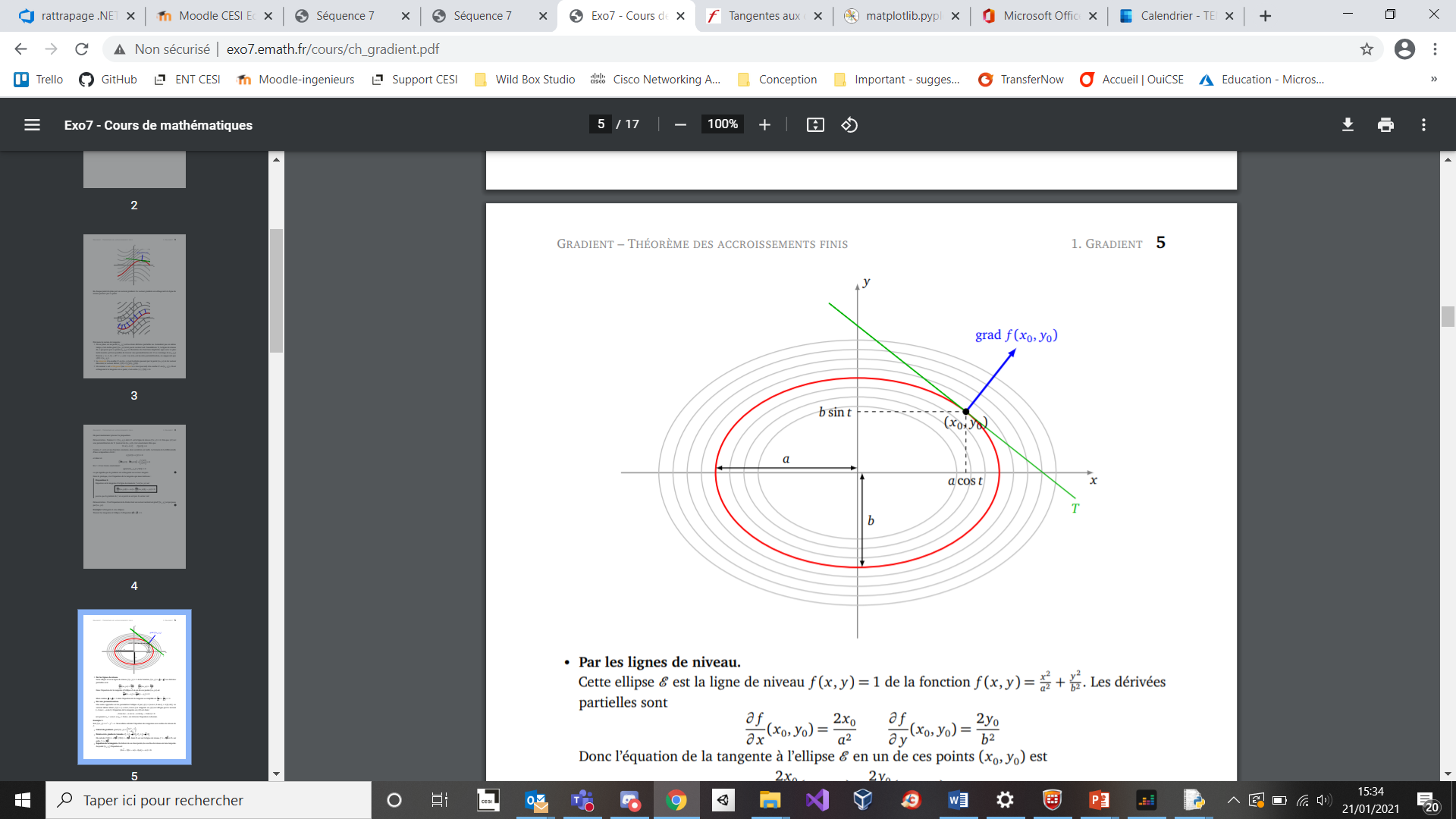


*Figure 5 – Plan tangent en un point du graphe d’une fonction à 2 variables*

**Important :** il n’y a pas de tangente ou plan tangent aux points où le gradient est le vecteur nul, il faut donc déterminer et exclure les points critiques.

### Exemple de l’ellipse

Déterminer l’équation de la tangente en tout point de l’ellipseoù *.*

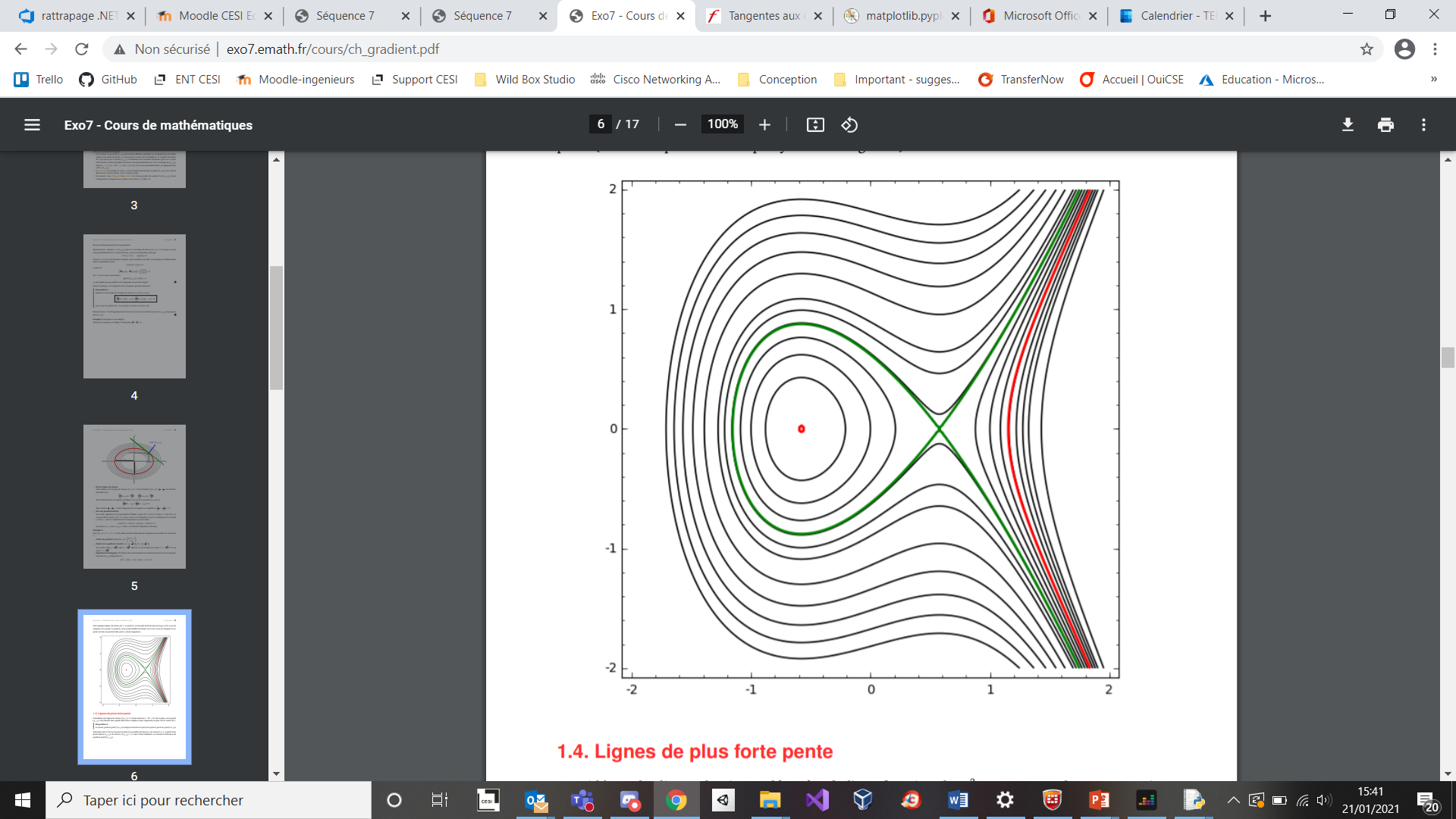
**

*Figure 6 – Courbes de niveau de l’ellipse*

|  |
| --- |
| **Calcul du gradient** |
| On s’intéresse à la ligne de niveauoù, on note . Alors |
| **Etude des points critiques** |
| or n’est pas dans donc il n’y a pas de point critique |
| **Equation de la tangente** |
| En tout point, l’équation de la tangente est  On conclue que l’équation de la tangente en tout pointest : |

### Exercice : tangentes et points critiques

Déterminer les équations des tangentes aux courbes de niveau de de et les points critiques.



*Figure 7 – Courbes de niveau de l’ellipse*

La courbe verte est une courbe de niveau, celle en rouge (décomposée en deux parties) aussi. Pour certaines valeurs, la courbe de niveau est en deux parties (courbe autour du point rouge avec courbe de même allure que la rouge à droite).

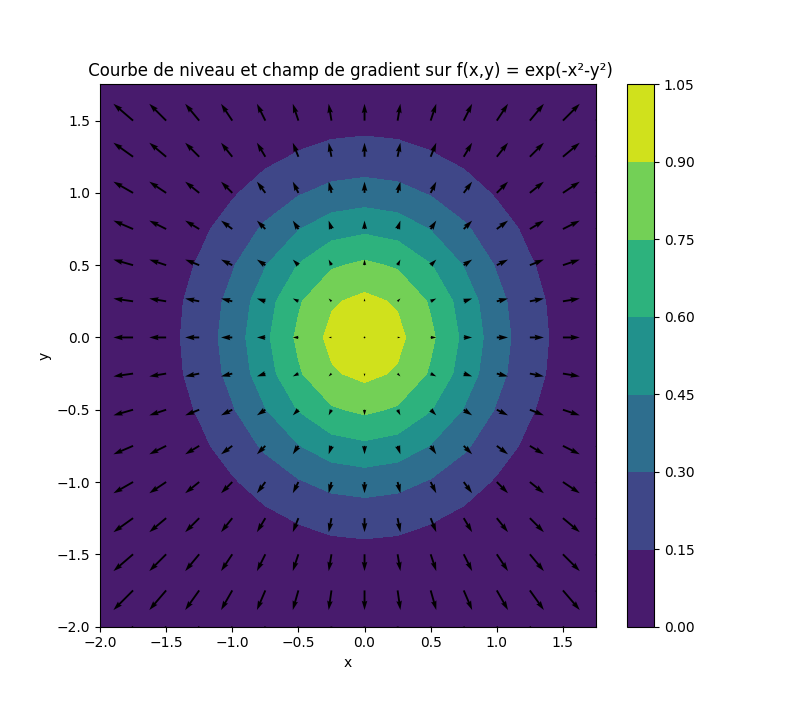
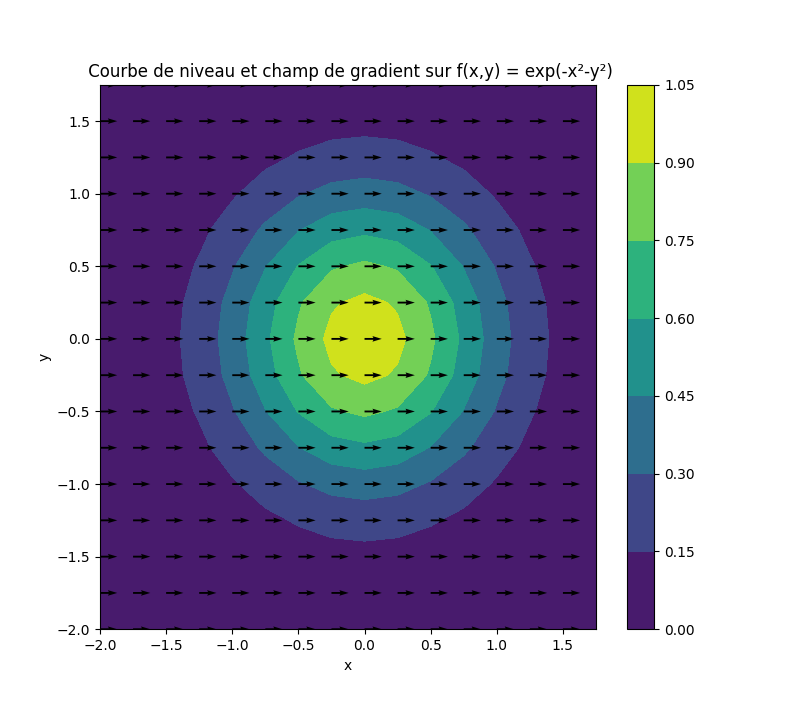
|  |
| --- |
| **Calcul du gradient** |
|  |
| **Etude des points critiques** |
| Les points critiques sont le point isolé en rouge et le point du croisement en vert. |
| **Equation de la tangente** |
| En tout point non-critique de , la tangente a pour équation : |

Si on observe plus en détail les points critiques de cette fonction, le point en rouge est un extremum local (un pic) tandis que le point du croisement en vert pourrait avoir deux tangentes ce qui n’est pas possible (et il n’est pas un extremum local).

## Champ de vecteurs

|  |
| --- |
| **Définitions.** Un **champ de vecteurs** est une fonction qui associe à chaque point de l’espace un vecteur. Un **champ de gradients** est une fonction qui associe à chaque point de l’espace pour une fonction son gradient en ce point. |

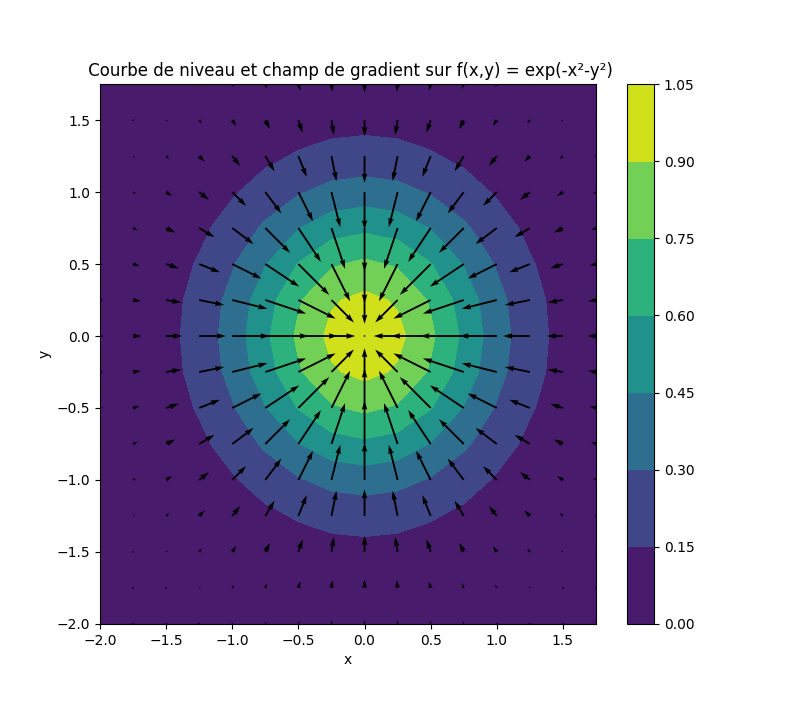
Concrètement, on ne représente pas tous les vecteurs en tous les points de l’espace sinon on ne voit rien. Sur python, la fonction *quiver* permet de les représenter comme sur la figure 8 (cf code *partie2-champs-de-vecteurs.py*).

****

*Figure 8 – Champ de vecteurs et pour*

Il n’y a pas de sens particulier aux champs de vecteurs pris au hasard. Celui avec est indépendant de . Celui avec semble se disperser à partir du sommet.

Par contre, lorsque le vecteur considéré est un gradient, le vecteur indique le sens et la direction pour augmenter le vite la valeur de . Plus la norme du gradient est élevée, plus elle indique une pente forte pour monter. C’est ce qu’illustre la figure 9 avec et .



*Figure 9 – Champ de gradient pour*

On voit bien aussi que les gradients sont orthogonaux aux courbes de niveau.

# Descente de gradient

Il est parfois compliqué de trouver un minimum algébriquement. Nous avons vu que le gradient indique par où aller pour augmenter la valeur de donc la maximiser. Si nous cherchons un minimum, il faut aller dans le sens opposé au gradient. C’est sur ce principe que repose l’algorithme de descente de gradient, qui a été proposé par Cauchy[[1]](#footnote-1).

## Ressources pédagogiques

Les deux ressources suivantes expliquent assez bien la notion de gradient. Elles vous seront utiles pour cette partie.

* Descente du gradient - principe pour des fonctions à une variable : [https://www.charlesbordet.com/fr/gradient-descent/#](https://www.charlesbordet.com/fr/gradient-descent/)
* Descente du gradient – principe et application pour la régression linéaire : <https://machinelearnia.com/descente-de-gradient/>

## Descente de gradient à deux variables

On considère l’algorithme de descente du gradient suivant pour des fonctions à 2 variables.

|  |
| --- |
| **GradientDescent(, , , , )** |
| **do**      **if**  **then**  **return**  **end if**    **while**  **return** |

Lorsque la norme du gradient est suffisamment petite (plus petite que ) pour être considérée comme nulle, c’est que le gradient peut être considéré comme le vecteur nul.

Cet algorithme sera aussi valable pour des fonctions en variables, on remplacera dans ce cas par (c’est ce qu’apporte la partie C).

Attention, cet algorithme ne garantit pas de trouver un minimum pour toute fonction . La condition sur à considérer est que ait un unique minimum local (c’est le cas lorsque est convexe). S’il y a plusieurs minimums locaux, utiliser l’algorithme de descente du gradient permettra de trouver un minimum local, mais pas le minimum global, il faut donc s’assurer que ce que l’on cherche admet un unique minimum au préalable.

**Question 3.1.** Implémenter l’algorithme *GradientDescent* en Python en ajoutant en paramètre le gradient qui devra être calculé au préalable par l’utilisateur. Les paramètres et auront pour valeurs respectives par défaut et 1000**.**

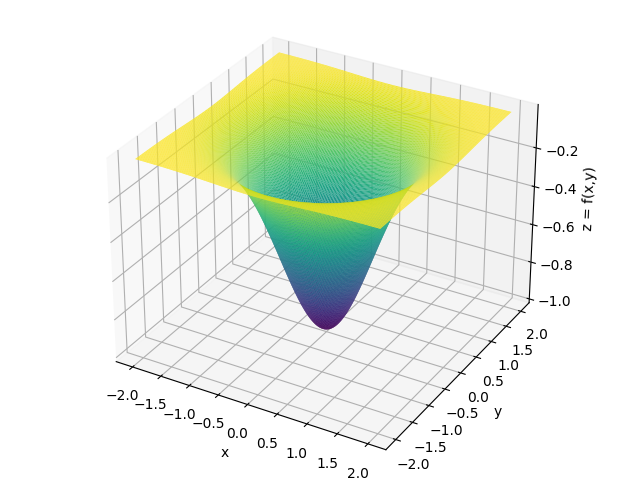
(cf code python)

Le gradient n’étant pas calculable de façon exacte sur Python, il doit être en paramètre de la fonction pour cette question. Des logiciels de calcul formel (en anglais : Computer Algebra) comme *Maple* ou *Sage* savent le faire.

**Question 3.2.** Tester *GradientDescent* sur avec le paramètre initial . Faites varier le taux d’apprentissage et le nombre d’itérations (par des facteurs 10). Que constatez-vous en terme de convergence de l’algorithme ?

Des taux d’apprentissage très petits auront une convergence très lente tandis que des taux d’apprentissage trop élevés auront une convergence trop grossière pour aboutir à un résultat. Il faudra donc observer la fonction et parfois tester divers paramètres.

**Choix du point de démarrage de l’algorithme de descente du gradient :** le choix du point de démarrage a une incidence sur le temps de convergence. Plus il est éloigné du minimum, plus il faut d’étapes pour finir l’algorithme. C’est d’autant plus critique que le taux d’apprentissage est trop faible. Pour la fonction de la figure 10, démarrer par un point dans les zones jaunes va compliquer la descente du gradient comme il s’agit de zones « quasi plates ». Il vaudra mieux démarrer par un point dans le creux. Connaître un bon point de démarrage n’est pas toujours facile, il faudra parfois réaliser de nombreux tests.



*Figure 10 – Graphe de*

**Question 3.3.** A partir de la fonction *GradientDescent* précédente, créer une fonction *GradientDescent2* réalisant cette fois-ci la descente de gradient sans connaître au préalable le gradient de .

Il s’agira de déterminer numériquement le gradient dans la fonction en utilisant la définition de la dérivée partielle en d’une fonction dérivable s’exprimant ainsi (si elle existe) :

La correction utilisera une définition équivalente à celle-ci (et qui découle de cette dernière par ailleurs) centrée autour de , plus cohérente vis-à-vis des approximations numériques que nous souhaitons réaliser :

(cf code *partie3B-descente\_gradient\_2\_var.py*)

## Descente de gradient à variables

On considère désormais des fonctions à variables.

**Question 3.4.** Créer la fonction *GradientDescentMulti* à partir de l’algorithme ci-dessous en adaptant la fonction *GradientDescent* pour pouvoir réaliser une descente de gradient à variables en connaissant le gradient de . On peut faire de même pour la fonction *GradientDescentMulti2* sans connaitre le gradient de (si vous avez compris le principe du premier, prenez le code solution pour cette seconde version). Indication : Utiliser np.array comme type de et simplifiera votre code pour qu’il ressemble pour plus à ce qui suit.

|  |
| --- |
| **GradientDescentMulti(, , , )** |
| **do**      **if**  **then**  **return**  **end if**    **while**  **return** |

(cf code *partie3C-descente\_gradient\_n\_var.py*)

# Regressions linéaires

## Régression linéaire simple

### Fonction des moindres carrées pour deux variables

La descente de gradient peut notamment être utilisée pour déterminer l’équation de la droite des moindres carrés c’est-à-dire la droite telle que les écarts entre les points et sont minimaux.

Cette droite des moindres carrés a bien entendu du sens s’il y a une corrélation suffisamment élevée pour déterminer que donc l’écart doit être le plus proche de 0 possible. On cherche à minimiser l’erreur quadratique moyenne :

On pose donc

On admet que cette fonction est convexe, on peut donc lui appliquer la descente de gradient à 2 variables que sont et (et non et ).

**Question 4.1.** Calculer le gradient de .

|  |
| --- |
| **Calcul du gradient** |
|  |

**Question 4.2.** Implémenter la régression linéaire en Python à partir des éléments précédents (vous pouvez choisir les fonctions précédemment codées de votre choix).

*(voir notebook)*

### Application : concentration en ozone

Nous disposons des données quotidiennes de la concentration en ozone en fonction de divers paramètres dont les températures à 9h, 12h et 15h notées respectivement T9, T12 et T15.

**Question 4.3.** Ouvrir le notebook jupyter *regression\_lineaire.ipynb* et rechercher les coefficients de la droite des moindres carrés notée RMSE en complétant le code proposé. Estimer la concentration en ozone qu’il pourrait y avoir à 40°C à l’heure du midi à l’aide de la RMSE.

*(voir notebook)*

## Régression linéaire multiple

### Transposée d’une matrice

Nous aurons besoin de la transposée d’une matrice qui intervient dans cette partie.

|  |
| --- |
| **Définition.** Soit une matrice de dimensions . On appelle transposée de la matrice notée obtenue en échangeant les lignes et les colonnes de . est de dimension |

**Exemples :**

**Remarques :**

* Un vecteur (colonne) peut être vu comme une matrice de dimension  ;
* Un vecteur ligne peut être vu comme une matrice de dimension  ;
* Certaines ressources considèrent les vecteurs indifféremment sous forme de ligne ou colonne, ce n’est qu’un choix de représentation dépendant de l’usage fait du vecteur ;
* Le produit scalaire entre deux vecteurs et de s’écrit .

### Fonction des moindres carrées pour variables

La descente de gradient peut notamment être utilisée pour déterminer l’équation de la droite des moindres carrés dans l’espace vectoriel c’est-à-dire la droite telle que les écarts entre les points (représentant la -ième donnée) et sont minimaux.

On pose les vecteurs suivants de **dimension**  :

Cette droite des moindres carrés a bien entendu du sens s’il y a une corrélation suffisamment élevée pour déterminer que .

Le 1 en plus dans l’expression de sert à transformer l’équation de la droite en un produit scalaire entre deux vecteurs et de mêmes dimensions. Le 1 s’associe donc avec dans le produit scalaire. Il faut penser à ne pas oublier ce 1 dans l’implémentation, il est en gros présent pour permettre de faire du calcul matriciel.

On cherche donc ici à minimiser l’écart qui doit être le plus proche de 0 possible. On cherche à minimiser :

En posant

et sont constants (ils sont construits à partir de nos données). On pose

**Question 4.4.** Développer en faisant apparaitre des sommes de produits matriciels (cela comprend aussi les vecteurs car tout vecteur est une matrice de ).

|  |
| --- |
| **Développement de** |
| Or et sont des scalaires par produit matriciel (de dimension ), donc est identique à sa transposée : donc se simplifie en  On peut aussi l’écrire  Cette dernière forme sera plus utile pour dériver d’après ce qui suit. |

On admet que la fonction est convexe, on peut donc lui appliquer la descente de gradient à variables.

|  |
| --- |
| **Propriétés.** Soit une matrice carrée à coefficients constants. Soient . Alors : |

On peut trouver la notation de dérivée partielle selon un vecteur qui coïncide avec le gradient, on n’entrera pas dans le détail de ces propriétés et on les admettra. Elles se déduisent des calculs de dérivées successives que l’on pourrait réaliser. On retiendra seulement qu’elle vont nous permettre de faire la même chose qu’avec le gradient à variables mais avec une forme vectorielle (cela facilitera en l’occurrence son implémentation).

**Question 4.5.** Calculer le gradient de selon (équivalent dérivée de selon ).

|  |
| --- |
| **Calcul de** |
| En utilisant les propriétés précédentes, on a :  On conclut donc que : |

On supposera pour la suite que , c’est-à-dire que les lignes de sont indépendantes les unes des autres. Si ce n’était pas le cas, cela signifierait que certaines données sont linéairement liées (ou en doublon par exemple) et peuvent être retirées jusqu’à avoir des données non liées entre elles. Avec les calculs, aboutit à :

**Question 4.6.** Calculer le gradient de . Implémenter la régression linéaire en Python à partir des éléments précédents (vous pouvez choisir les fonctions précédemment codées de votre choix).

(voir code)

### Application : attaques cardiaques

Nous disposons de données concernant le taux de décès par attaque cardiaque chez les hommes de 55 à 59 ans dans différents pays. Les données récoltées dans le fichier attaques\_cardiaques.csv sont représentées par les variables :

**Question 5.** Déterminer la régression linéaire multiple de la forme en utilisant la formule , après avoir construit et à partir des données du fichier attaques\_cardiaques.csv. Discutez de la pertinence du résultat obtenu.

1. *Cauchy A*. [Méthode générale pour la résolution des systèmes d'équations simultanées](https://ci.nii.ac.jp/naid/10026863174). *C.R. Acad. Sci. Paris 25, 536-538, 1847* [↑](#footnote-ref-1)