



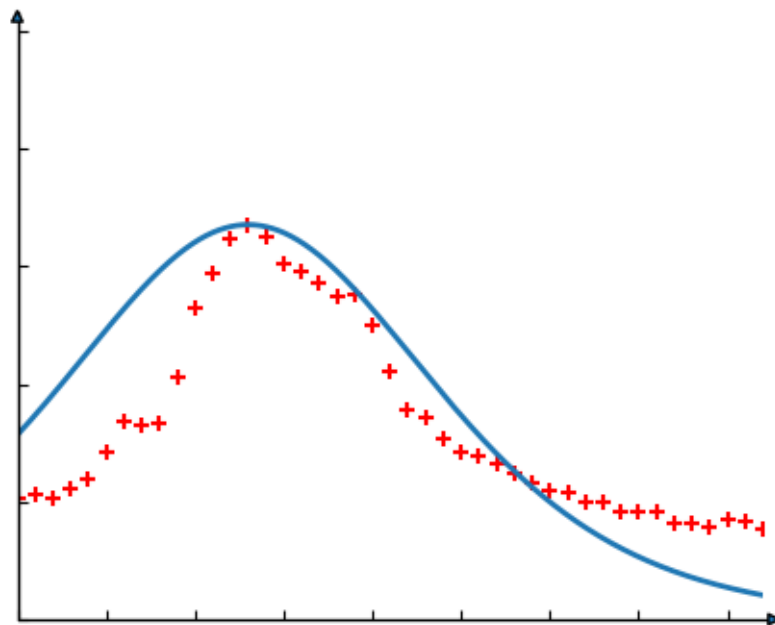
Projet : modélisation de pics de production de ressources

Yoan Thomas

Aya Ismahene Kroussa

Melvin Cerba

April 26, 2022



Contents

1	Élément historique	3
1.1	Modélisation des pics de productions	3
1.2	Courbe de Hubbert et sigmoïde	3
2	Algorithme d'approximation	5
2.1	Caractérisation	5
2.1.1	Critère	5
2.1.2	Gradient	6
2.1.3	Direction de descente	6
2.1.4	Figure des isocourbes avec les direction de descentes	6
2.1.5	Matrice de mise à l'échelle	6
2.1.6	Pseudo code	6
2.2	Test de contrôle de l'algorithme et des fonctions associées	6
2.2.1	Vérification du gradient par différences finies	6
2.2.2	Test de convergence avec données bruitées et non bruitées, en commençant plus ou moins loin de la solution	8
2.3	Performance <i>-optionnel</i>	8
2.3.1	Temps de convergence selon différent paramètres	9
3	Données réelles	9
3.1	Modélisation de la production pétrolière de la France	9
3.1.1	Application de l'algorithme aux données des pays de l'OCDE	10
3.1.2	Modélisation de la production mondiale	10
3.1.3	Explication des lacunes du modèle	10
3.2	Prévisions à partir du modèle	10
3.2.1	Optimisation sur données incomplètes	11
3.2.2	Performance de l'algorithme pour trouver les pics de production sur les données qui peuvent être modélisées par une sigmoïde	11
3.3	Amélioration de modèle	11
3.3.1	Deux sigmoïdes	11
3.3.2	n sigmoïdes	11
4	Conclusion	11

1 Élément historique

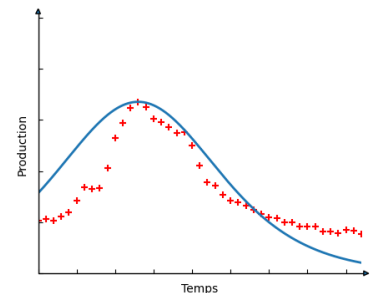
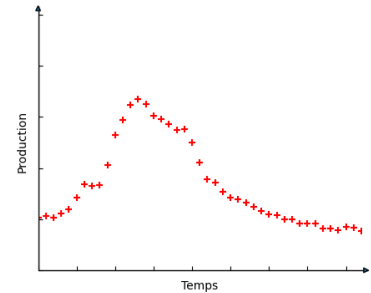
Peut-on prédire l'évolution de la production des ressources non renouvelables ? Pour tenter de répondre à cette question, nous proposons d'étudier à travers ce projet certains outils mathématiques qui permettent de modéliser la production des ressources non renouvelables. Plus spécifiquement, nous nous intéresserons à la production de pétrole.

1.1 Modélisation des pics de productions

Être capable de prédire l'évolution de la production de diverses ressources a toujours été un enjeu majeur, notamment en économie. Il est donc naturel que de nombreux modèles mathématiques promettant de modéliser cette évolution soient apparus. A travers ce projet, nous vous proposons d'étudier en détail l'un d'eux : la courbe de Hubbert.

En 1956, Marion King Hubbert présentait sa "courbe de Hubbert" à l'American Petroleum Institute. Son modèle, qui postule que la production croît, atteint un unique pic, puis décroît au même rythme qu'elle a augmenté en premier lieu, ne fit pas beaucoup parler de lui à l'époque. Mais lorsqu'en 1971, conformément à ses prédictions, la production pétrolière américaine atteignit son maximum et commença à décliner, ses travaux furent réexaminés avec beaucoup plus d'intérêt. Les chocs pétroliers de 1973 et 1979 semblèrent cependant définitivement invalider son modèle, qui perdit rapidement l'attention de l'industrie pétrolière.

Malgré tout, l'avènement du calcul informatique et la grande disponibilité de données poussèrent des auteurs modernes à exhumer le modèle de Hubbert et à l'étendre, notamment en donnant une formule mathématique à sa courbe, permettant ainsi de calculer son intégrale. C'est sur la base de ces nouveaux travaux que nous avons construit notre projet.



1.2 Courbe de Hubbert et sigmoïde

Les auteurs qui se sont approprié la courbe de Hubbert ont notamment travaillé à lui donner une expression mathématique. Ceci leur a permis de définir son intégrale, que nous appellerons "fonction sigmoïde" tout au long de ce rapport. Pour des raisons que nous expliciterons plus bas, cette dernière s'avère plus facile à manier que la courbe de Hubbert.

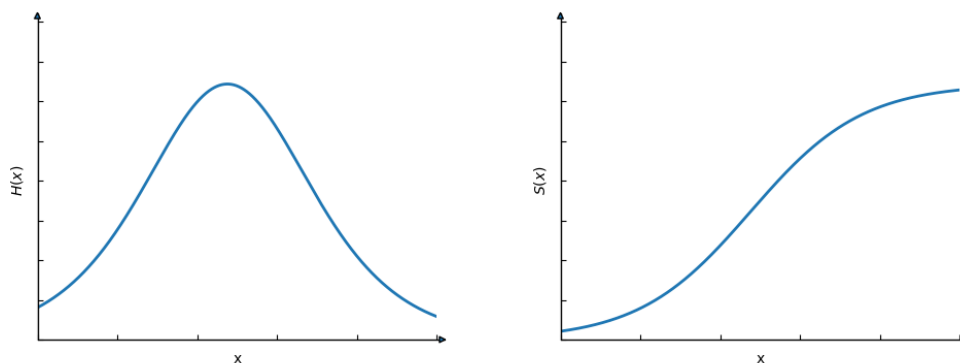
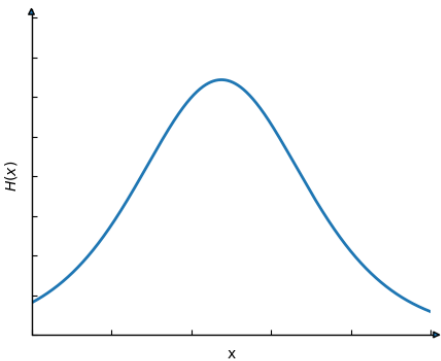


Figure 1: Une courbe de Hubbert et sa sigmoïde (son intégrale)

:

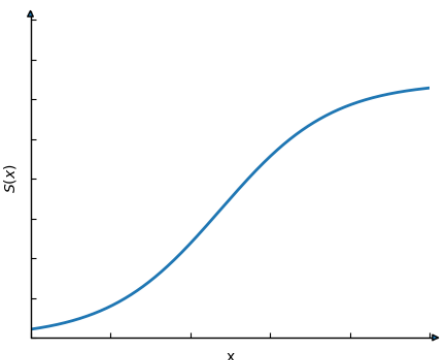
Une *courbe de Hubbert*¹ est une courbe qui décrit de manière explicite l'évolution au cours du temps de la production d'une ressource donnée (par exemple, du pétrole). Cette fonction s'écrit

$$H(t; \theta) = \frac{S_{\max}}{\tau} \frac{e^{-\frac{t-t_{\star}}{\tau}}}{\left(1 + e^{-\frac{t-t_{\star}}{\tau}}\right)^2} \quad (1)$$


Dans l'expression ci-dessus, t représente la variable de temps et le triplet de paramètres $\theta := (S_{\max}, t_{\star}, \tau)$ permet de définir sans ambiguïté la courbe dans la famille paramétrée. Comme le montre l'illustration ci-dessus, cette courbe a la forme d'une “cloche” et les trois paramètres définissent son allure générale. Plus spécifiquement, on a

- S_{\max} contrôle l'amplitude maximale de la courbe ;
- t_{\star} repère la position de ce maximum sur l'axe des temps ;
- τ contrôle le caractère plus ou moins “piqué” de la cloche.

L'intégrale de cette fonction de Hubbert possède la forme explicite suivante :

$$S(t; \theta) = \frac{S_{\max}}{1 + e^{\frac{-(t-t_{\star})}{\tau}}} \quad (2)$$


Cette fonction est largement connue sous le nom de *fonction sigmoïde*. Dans le cadre d'une interprétation de la consommation de ressources naturelles, la sigmoïde modélise l'évolution temporelle *cumulée* de ressource extrait du sol.

Rappelons notre objectif et nos hypothèses : nous voulons modéliser la production de pétrole au cours du temps, et nous supposons que celle-ci atteint un unique pic. Il nous faudra donc déterminer le t pour lequel ce pic est atteint ainsi que la hauteur de ce pic (quantité de pétrole produit à l'instant t).

¹Page wikipédia • Nuclear energy and the fossil fuels [archive] M. King Hubbert, 1956
• The Hubbert parabola [archive], Roberto Canogar

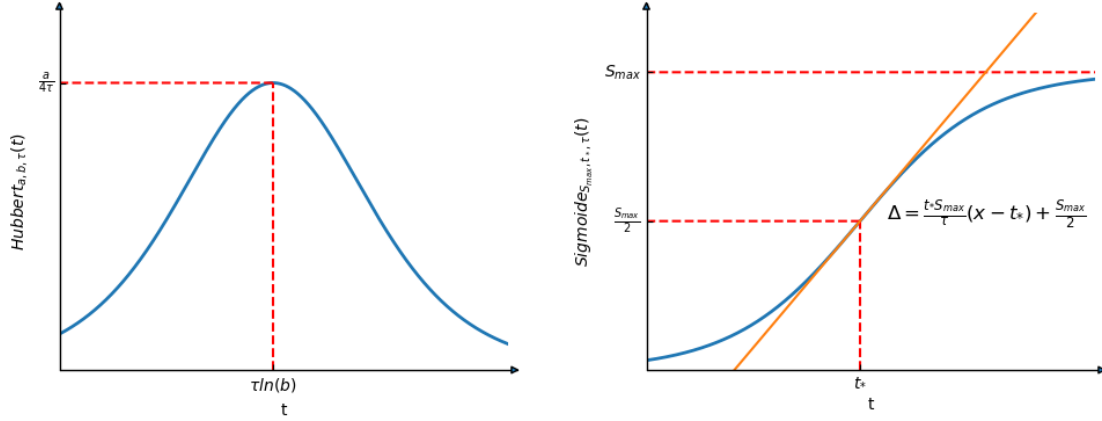


Figure 2: Une courbe de Hubbert et sa sigmoïde (Δ : pente de la sigmoïde au point d'inflexion t_*)

2 Algorithme d'approximation

Afin d'approximer les paramètres de la sigmoïde la plus en adéquation avec nos données, nous allons utiliser une méthode de minimisation de fonction réputée : la *descente de gradient*.

Conditions à vérifier

- *Initialisation* : pour que l'algorithme "descende" vers le minimum absolu de la fonction, il faut l'initialiser dans un "creux" qui le contienne, c'est à dire un intervalle sur lequel la fonction est convexe et dont l'image contient le minimum absolu de la fonction.
- *Direction de descente* : par ailleurs, il faut être certain que l'algorithme sache en tout instant dans quelle direction aller (ie. quelles modifications apporter aux paramètres) pour "descendre" vers le minimum. Pour cela, nous utiliserons le gradient.

Algorithme

L'algorithme fonctionne de la manière suivante :

1. Il calcule $\nabla \text{Crit}_D(S(t; \theta))$ le gradient de la fonction critère.
2. Il définit une direction de descente d à partir de ∇C .
3. Il modifie les paramètres initiaux selon d et un pas δt .
4. Puis il recommence, jusqu'à satisfaire des conditions prédéfinies qui assure une certaine proximité au minimum de la fonction.

2.1 Caractérisation

Dans cette partie, nous

2.1.1 Critère

Avant toute chose, il nous faut une fonction à minimiser. Nous allons donc définir un *critère* qui évalue la distance qui sépare les données de la courbe du modèle. Plus précisément, nous additionnerons pour chaque année le carré de la différence entre l'estimation du modèle et les données réelles.

$$\text{Criterion}_{Data}(Sig(t; \Theta)) = \sum_{k=1}^N \delta t |Data(t_k) - Sig(t_k; \Theta)|^2 \quad (3)$$

avec :

$Data(t_k)$ la somme des données de production de l'instant t_1 à l'instant t_k
 δt

Par la suite, on notera indifféremment $\text{Criterion}_{Data}(Sig(t; \Theta))$, $\text{Crit}_D(Sig(t; \Theta))$ ou $\text{Crit}(Sig(t; \Theta))$.

2.1.2 Gradient

Afin de converger vers les paramètres optimaux, l'algorithme de la descente de gradient a besoin d'une direction de descente. Cela passe avant tout par le calcul du gradient de la fonction à minimiser, afin d'avoir une idée précise de l'évolution de cette dernière en fonction de chaque paramètre.

A partir de (3), on obtient :

$$\nabla \text{Crit}(S(t; \theta)) = \sum_{k=1}^N 2 * \nabla S(t_k; \theta) * (S(t_k; \theta) - D(t_k)) \quad (4)$$

Il nous faut donc définir $\vec{\nabla} \text{Sig}(\Theta)$:

$$\vec{\nabla} \text{Sig}(t; \Theta) = \frac{\partial^2 \text{Sig}}{\partial t \partial \Theta}(t, \Theta) = \begin{bmatrix} \frac{\partial \text{Sig}}{\partial t \partial S_{max}}(t, \Theta) \\ \frac{\partial \text{Sig}}{\partial t \partial t_*}(t, \Theta) \\ \frac{\partial \text{Sig}}{\partial t \partial \tau}(t, \Theta) \end{bmatrix}$$

Par le calcul, on obtient :

$$\vec{\nabla} \text{Sig}(t; \Theta) = \begin{bmatrix} \frac{1}{1 + e^{-\frac{(t-t_*)}{\tau}}} \\ \frac{-S_{max}}{\tau} * \frac{e^{-\frac{(t-t_*)}{\tau}}}{(1 + e^{-\frac{(t-t_*)}{\tau}})^2} \\ \frac{-S_{max}}{\tau^2} * \frac{e^{-\frac{(t-t_*)}{\tau}}}{(1 + e^{-\frac{(t-t_*)}{\tau}})^2} \end{bmatrix} \quad (5)$$

Nous sommes donc capables de calculer le gradient de la sigmoïde pour des paramètres S_{max} , t_* et τ donnés. Mais comment en tirer une direction de descente ?

2.1.3 Direction de descente

Le gradient nous d

2.1.4 Figure des isocourbes avec les direction de descentes

Affichage de figure, et justification de la nécessité de l'utilisation d'une matrice de mise à l'échelle

2.1.5 Matrice de mise à l'échelle

Définition de la matrice de mise à l'échelle

2.1.6 Pseudo code

Pseudo code

2.2 Test de contrôle de l'algorithme et des fonctions associées

Avant de lancer la l'algorithme sur des données réelles on vérifie que nos fonctions soient justes et que l'algorithme converge pour des données simulées

2.2.1 Vérification du gradient par différences finies

Test rapide du gradient par la méthode des différences finies Rappel de la méthode des différences finies² :

En analyse numérique, la *méthode des différences finies* est une technique courante de recherche de solutions approchées d'équations aux dérivées partielles qui consiste à résoudre un système de relations (schéma numérique)

²

G. Allaire. Analyse numérique et optimisation (French Edition). ECOLE POLYTECHNIQUE, 2012.

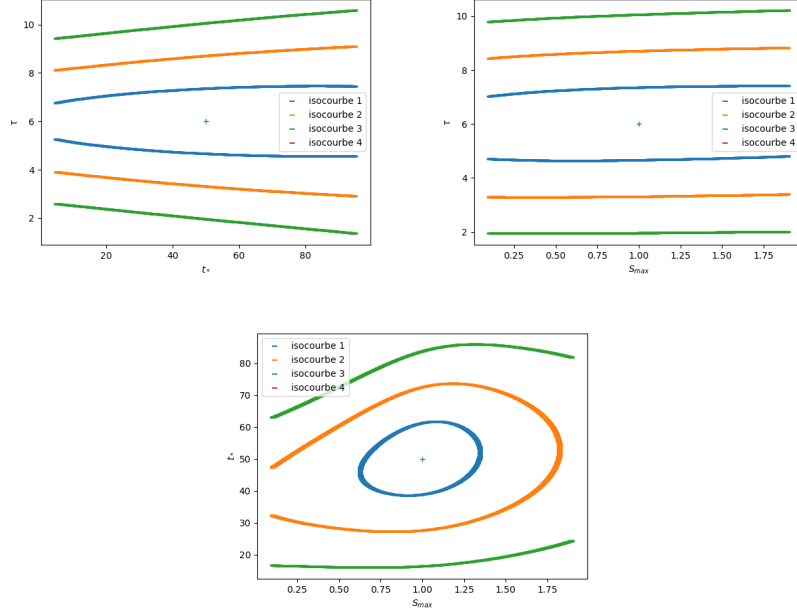


Figure 3: Une courbe de Hubbert et sa sigmoïde (Δ : pente de la sigmoïde au point d'inflexion t_*)

liant les valeurs des fonctions inconnues en certains points suffisamment proches les uns des autres.

Cette méthode apparaît comme étant la plus simple à mettre en œuvre car elle procède en deux étapes : d'une part la discrétisation par *différences finies* des opérateurs de dérivation/différentiation, d'autre part la convergence du schéma numérique ainsi obtenu lorsque la distance entre les points diminue.

Définition : Soit $N \in \mathbb{N}^*$, on appelle *discrétisation régulière* de $[a, b]$ à N pas ou $N + 1$ l'ensemble des points $a + nh$, $n \in [0, N]$ où le pas h est donné par $h = (b - a)/N$.

Soient $\{t^n\}_{n \in [0, N]}$ une discrétisation régulière de $[a, b]$ et $(Dy)_n$ une approximation de $y'(t^n)$. On appelle :

• *Différence finie progressive* l'approximation :

$$(Dy)_n^P = \frac{y(t^{n+1}) - y(t^n)}{h}, \quad \forall n \in [0, N - 1]$$

• *Différence finie rétrograde* l'approximation :

$$(Dy)_n^R = \frac{y(t^n) - y(t^{n-1})}{h}, \quad \forall n \in [1, N]$$

• *Différence finie centrée* l'approximation :

$$(Dy)_n^C = \frac{y(t^{n+1}) - y(t^{n-1})}{2h}, \quad \forall n \in [1, N - 1]$$

Et dans notre cas on va travailler avec l'approximation *progressive* c'est à dire $y'(t) = \lim_{h \rightarrow 0} \frac{y(t+h) - y(t)}{h}$. On appliquant cette formule à notre fonction sigmoïde on aura :

$$\begin{aligned} \frac{\partial S(t, \theta)}{\partial S_{\max}} &= \frac{S(t; S_{\max} + \text{delta}, t_s, \tau) - S(t; S_{\max}, t_s, \tau)}{\text{delta}} \\ \frac{\partial S(t, \theta)}{\partial t_s} &= \frac{S(t; S_{\max}, t_s + \text{delta}, \tau) - S(t; S_{\max}, t_s, \tau)}{\text{delta}} \\ \frac{\partial S(t, \theta)}{\partial S_{\max}} &= \frac{S(t; S_{\max}, t_s, \tau + \text{delta}) - S(t; S_{\max}, t_s, \tau)}{\text{delta}} \end{aligned}$$

Avec $\text{delta} = (t_f - t_i)/N$ qui représente le pas de notre approximation.

Et pour vérifier si notre calcul du gradient est bon, on fait un simple calcul qui est la différence en valeur absolue du gradient calculer dans la partie précédente et le gradient trouver par la méthode de différence finie.

C'est à dire :

$$diff = |\nabla S(t, \theta) - (\frac{\partial S(t, \theta)}{\partial S_{\max}}, \frac{\partial S(t, \theta)}{\partial t_s}, \frac{\partial S(t, \theta)}{\partial S_{\max}})|$$

Et donc on trouve bien que *diff* est un petit nombre qui converge vers le 0.

Alors, on a bien fait la vérification de notre gradient par la méthode de différences finies de la fonction *Sigmoïde*. Et de la même manière on vérifie aussi le gradient de la fonction *Critère*.

2.2.2 Test de convergence avec données bruitées et non bruitées, en commençant plus ou moins loin de la solution

Tentons désormais de définir le cadre au sein duquel notre algorithme fonctionne. Pour ce faire, nous allons générer des données parfaites (c'est à dire qui correspondent exactement à une courbe sigmoïde dont nous aurons choisi les paramètres) et jouer sur deux paramètres :

- La *distance* à la solution des paramètres d'initialisation. Comme nous générons les données à partir d'une courbe sigmoïde que nous définissons, nous avons connaissance des paramètres optimaux Θ_{opti} . Nous pouvons donc définir les paramètres d'initialisation Θ_{init} de l'algorithme en fonction de ceux-ci, et donc décider de la "distance" qui sépare l'état initial de l'état optimal.

$$\Theta_{\text{init}} = (1 + a) * \Theta_{\text{opti}} \quad (6)$$

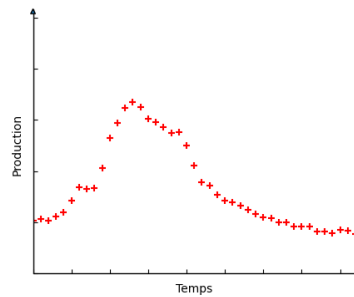
Nous jouerons ici sur le paramètre *a*.

- Le *bruit*. Nous allons "salir" nos données en leur ajoutant un bruit gaussien, c'est à dire une perturbation égale à une variable aléatoire qui suit la loi de probabilités normale centrée en 0 :

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}} \quad (7)$$

Nous jouerons donc sur le paramètre σ .

Cherchons maintenant la distance limite : celle à partir de laquelle l'algorithme ne converge plus, même avec des données non-bruitées.



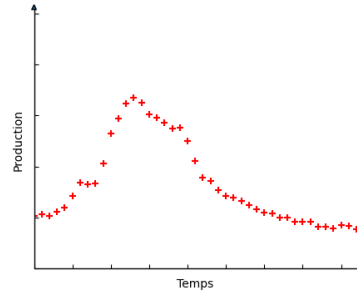
On remarque donc qu'avec des données non bruitées, l'algorithme ne bloque pas, mais ne converge plus à partir de $a = 1$.

Cherchons donc le niveau de bruit limite : la valeur de σ au delà de laquelle notre algorithme ne converge plus, même avec des paramètres initiaux très proches de la solution ($\Theta_{\text{init}} = 1, 10 * \Theta_{\text{opti}}$.)

On constate qu'avec des paramètres initiaux très proches de la solution, l'algorithme bloque à partir d'un bruit de 200. (la matrice B est singulière)

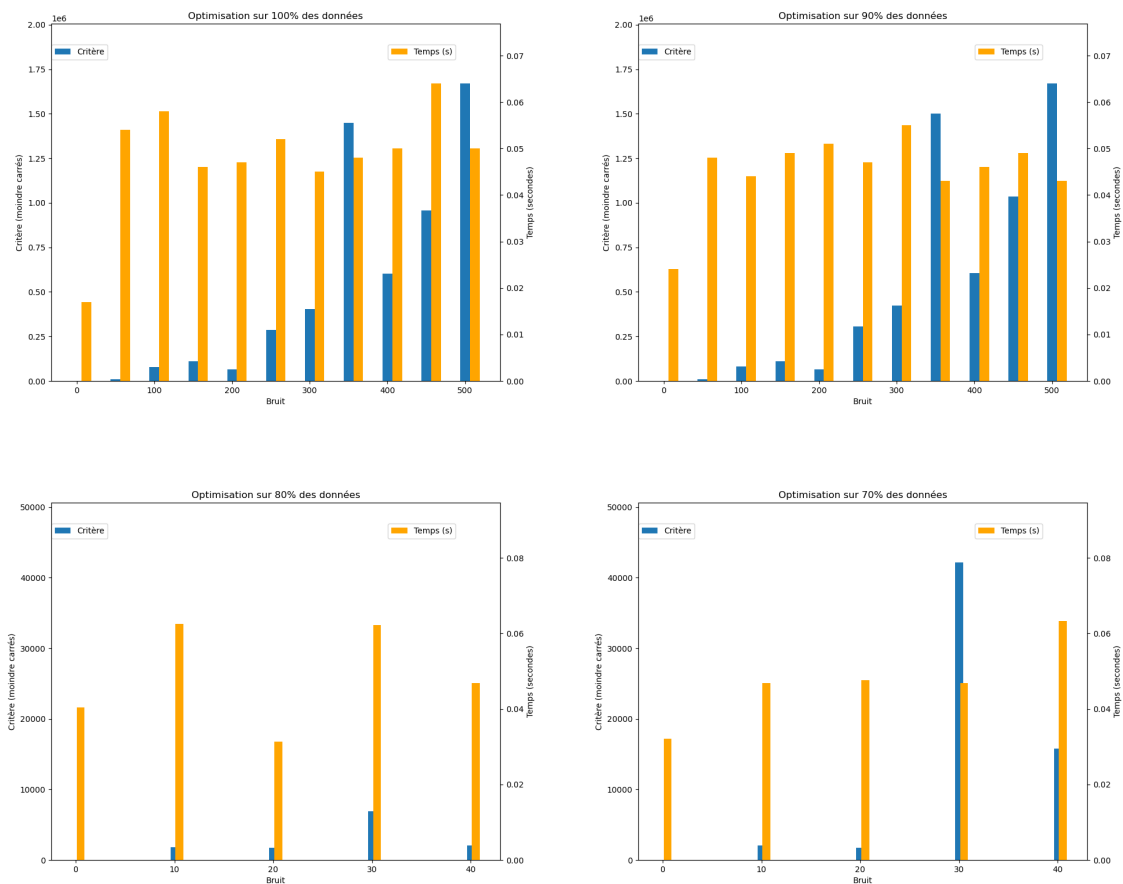
2.3 Performance -optionnel

Testons maintenant les performances de notre algorithme de manière plus rigoureuse.



2.3.1 Temps de convergence selon différent paramètres

Pour chaque valeurs de σ et de a , nous allons lancer l'algorithme d'optimisation 15 fois et calculer les moyennes du temps d'exécution et de la valeur du critère des courbes sigmoïdes obtenues.

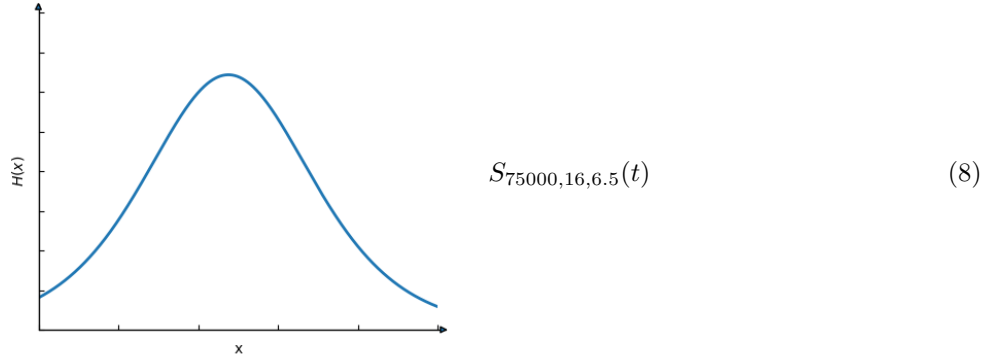


3 Données réelles

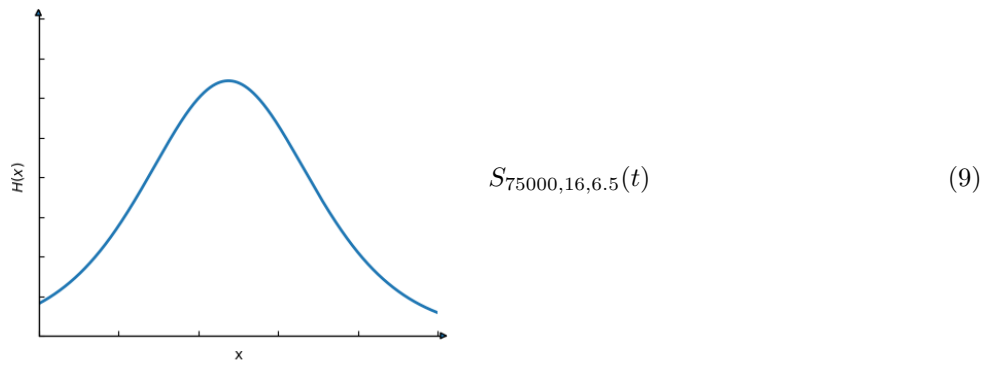
Maintenant que nous savons que notre algorithme fonctionne, voyons si le modèle qu'il génère est pertinent lorsqu'il s'agit de représenter des données réelles. Pour cela, nous utiliserons les données de production pétrolière que l'OCDE met à disposition de chacun sur son site web.

3.1 Modélisation de la production pétrolière de la France

Essayons par exemple d'appliquer notre algorithme aux données de production pétrolière de la France. On jouant manuellement sur les paramètres, on trouve une première approximation de la sigmoïde optimale :



Nous allons donc lancer l'algorithme avec ces paramètres initiaux. Voici le modèle que l'on obtient :



Le résultat est pertinent : il colle bien aux données et reconnaît avec précision le pic de production ainsi que la quantité totale de pétrole produite. A priori, l'algorithme est assez efficace et ses résultats sont pertinents. Mais d'une part nous avons dû l'initialiser manuellement, et d'autre part, la production pétrolière de certains pays ne suit pas aussi clairement une courbe en cloche que celle de la France.

3.1.1 Application de l'algorithme aux données des pays de l'OCDE

Essayons maintenant d'automatiser notre démarche pour la généraliser à l'ensemble des pays dont les données de production pétrolière sont présentes sur le site de l'OCDE.

Dans le cas de la France, nous avons approximé manuellement les des paramètres initiaux. Ici, afin d'automatiser le processus, nous allons définir Θ_{init} comme suit :

$$\Theta_{\text{init}} = \begin{bmatrix} S_{\text{max}} = \max \{D_n\} \\ t_* = \frac{N}{2} \\ \tau = 6,5 \end{bmatrix} \quad (10)$$

3.1.2 Modélisation de la production mondiale

Test de la pertinence du modèle sur une vue globale

3.1.3 Explication des lacunes du modèle

Après avoir mit en exergue les lacunes sur quelques jeux de données, tentatives de caractérisations des hypothèses de validité du modèle

3.2 Prévisions à partir du modèle

Nous avons constaté que, lorsque certaines hypothèses sont vérifiées, le modèle de Hubbert colle relativement bien au données réelles. Cependant, modéliser des données à posteriori n'a que peu d'intérêt. Notre algorithme

est-il capable de plus ? Le modèle obtenu permet-il de prédire l'évolution de la production dans une fenêtre qui dépasse celle des données sur lesquelles il a été optimisé ?

3.2.1 Optimisation sur données incomplètes

La détermination du pics étant simple a posteriori, on peut calculer de la performance de l'algorithme à trouver le pic de production a posteriori

3.2.2 Performance de l'algorithme pour trouver les pics de production sur les données qui peuvent être modélisées par une sigmoïde

En prenant des sets de données qui peuvent être convenablement modélisé par une sigmoïde, on peut chercher à quel point le modèle est prédictif. Pour cela on calcule la précision avec laquelle l'algorithme prédit le pic de production en se limitant aux premières données jusqu'à un temps t

3.3 Amélioration de modèle

Le modèle de Hubbert nous donne des résultats plutôt satisfaisants, mais ses hypothèses sont assez contraignantes. Nous avons donc réfléchi à une façon de l'étendre, de sorte à ce qu'il soit applicable à des cas plus variés.

3.3.1 Deux sigmoïdes

Si il y a deux pics, est-ce que deux sigmoïdes additionnées est une bonne modélisation?

3.3.2 n sigmoïdes

Si il y a n pics, est-ce que n sigmoïdes additionnées est une bonne modélisation?

4 Conclusion