



Modélisation et contrôle numérique de systèmes dynamiques en agronomie

Partie 3: Identification par moindres carrés

Module de Formation Continue Supagro Montpellier -
Cursus Data Science -
16-19 Janvier 2018, Montpellier, France

Céline Casenave¹

¹INRA UMR INRA-SupAgro MISTEA, Montpellier, France

Identification de modèles

Objectif

Question: quelles sont les valeurs $\hat{\theta}$ des paramètres θ d'un modèle donné qui permettent d'obtenir des sorties simulées proches des mesures?

Identification de modèles

Objectif

Question: quelles sont les valeurs $\hat{\theta}$ des paramètres θ d'un modèle donné qui permettent d'obtenir des sorties simulées proches des mesures?

Formulation mathématique du problème:

Problème de **minimisation** d'une **fonction objectif** $J(\theta)$, de la forme:

$$\hat{\theta} = \operatorname{argmin}_{\theta \in \Omega} J(\theta)$$

où Ω est l'ensemble des valeurs des paramètres θ admissibles.

Identification de modèles

Objectif

Question: quelles sont les valeurs $\hat{\theta}$ des paramètres θ d'un modèle donné qui permettent d'obtenir des sorties simulées proches des mesures?

Formulation mathématique du problème:

Problème de **minimisation** d'une **fonction objectif** $J(\theta)$, de la forme:

$$\hat{\theta} = \operatorname{argmin}_{\theta \in \Omega} J(\theta)$$

où Ω est l'ensemble des valeurs des paramètres θ admissibles.

⇒ **notion de proximité à clarifier**: “proches” en quel sens? selon quelle distance?

Identification de modèles

Notion de distance en mathématiques

Objectif: pour quantifier l'éloignement entre deux objets de même nature.

Identification de modèles

Notion de distance en mathématiques

Objectif: pour quantifier l'éloignement entre deux objets de même nature.

Distance "intuitive" = distance euclidienne

définie comme la racine de la somme des différences au carré

$$d(x, y) = \sqrt{\sum_{i=1}^N (x_i - y_i)^2}$$

où

- $x = (x_1, \dots, x_N)^T \in \mathbb{R}^N$: vecteur de **données simulées** (avec le modèle et donc dépendantes des **paramètres** θ)
- $y = (y_1, \dots, y_N)^T \in \mathbb{R}^N$: vecteur des **données mesurées** expérimentalement

Identification de modèles

Notion de distance en mathématiques

Objectif: pour quantifier l'éloignement entre deux objets de même nature.

Distance "intuitive" = distance euclidienne pondérée
définie comme la racine de la somme des différences au carré

$$d(x, y) = \sqrt{\sum_{i=1}^N \beta_i (x_i - y_i)^2}$$

où • β_i sont des poids positifs

- $x = (x_1, \dots, x_N)^T \in \mathbb{R}^N$: vecteur de **données simulées** (avec le modèle et donc dépendantes des **paramètres** θ)
- $y = (y_1, \dots, y_N)^T \in \mathbb{R}^N$: vecteur des **données mesurées** expérimentalement

Problème de moindres carrés

Formulation

Problème de minimisation:

$$\hat{\theta} = \operatorname{argmin}_{\theta \in \Omega} J(\theta) \text{ avec } J(\theta) = \sum_{i=1}^N \beta_i (y_i(\theta) - y_i^m)^2$$

- où
- θ est le **vecteur de paramètres** du modèle
 - N est le **nombre d'instants** t_i d'observation
 - y_i^m est le **vecteur des observations (mesures)** à l'instant t_i
 - $y_i(\theta)$ est le **vecteur des estimations données** par le modèle à l'instant t_i en fonction de θ
 - β_i est un **coefficient de pondération** pouvant être différent selon l'instant d'observation

Problème de moindres carrés

Formulation

Problème de minimisation:

$$\hat{\theta} = \operatorname{argmin}_{\theta \in \Omega} J(\theta) \text{ avec } J(\theta) = \sum_{i=1}^N \beta_i (y_i(\theta) - y_i^m)^2$$

- où
- θ est le **vecteur de paramètres** du modèle
 - N est le **nombre d'instants** t_i d'observation
 - y_i^m est le **vecteur des observations (mesures)** à l'instant t_i
 - $y_i(\theta)$ est le **vecteur des estimations données** par le modèle à l'instant t_i en fonction de θ
 - β_i est un **coefficient de pondération** pouvant être différent selon l'instant d'observation

⇒ **solution et méthode à utiliser dépendent du modèle utilisé**

Modèles de regression linéaire

Solution du problème de moindres carrés

Forme des modèles: modèles linéaires par rapport à θ :

$$y_i(\theta) = \phi_i^T \theta$$

où ϕ_i est un vecteur de même taille que θ appelé **régresseur**.

Modèles de regression linéaire

Solution du problème de moindres carrés

Forme des modèles: modèles linéaires par rapport à θ :

$$y_i(\theta) = \phi_i^T \theta$$

où ϕ_i est un vecteur de même taille que θ appelé **régresseur**.

Problème de moindres carrés:

$$\min_{\theta \in \Omega} J(\theta) = \min_{\theta \in \Omega} \sum_{i=1}^N \beta_i (\phi_i^T \theta - y_i^m)^2$$

Modèles de regression linéaire

Solution du problème de moindres carrés

Forme des modèles: modèles linéaires par rapport à θ :

$$y_i(\theta) = \phi_i^T \theta$$

où ϕ_i est un vecteur de même taille que θ appelé **régresseur**.

Problème de moindres carrés:

$$\min_{\theta \in \Omega} J(\theta) = \min_{\theta \in \Omega} \sum_{i=1}^N \beta_i (\phi_i^T \theta - y_i^m)^2$$

Solution analytique appelée **estimateur des moindres carrés**:

$$\hat{\theta} = \left[\sum_{i=1}^N \beta_i \phi_i \phi_i^T \right]^{-1} \sum_{i=1}^N \beta_i \phi_i y_i^m$$

sous condition que la matrice $\sum_{i=1}^N \beta_i \phi_i \phi_i^T$ est inversible.

Modèles de régression linéaire

Exemple: taux de croissance

Fonction de Monod: $\mu(S) = k \frac{S}{a + S}$

Problème: trouver les paramètres a et k de la fonction à partir de N mesures bruitées S_i et μ_i , $i = 1 : N$ de $S(t_i)$ et $\mu(S(t_i))$

Modèles de régression linéaire

Exemple: taux de croissance

Fonction de Monod: $\mu(S) = k \frac{S}{a + S}$

Problème: trouver les paramètres a et k de la fonction à partir de N mesures bruitées S_i et μ_i , $i = 1 : N$ de $S(t_i)$ et $\mu(S(t_i))$

Méthode 1: résoudre le problème de moindres carrés:

$$(\hat{a}, \hat{k}) = \operatorname{argmin}_{(a,k) \in \Omega} \sum_{i=1}^N \beta_i \left(\mu_i - k \frac{S_i}{a + S_i} \right)^2$$

Modèles de régression linéaire

Exemple: taux de croissance

$$\text{Fonction de Monod: } \mu(S) = k \frac{S}{a + S}$$

Problème: trouver les paramètres a et k de la fonction à partir de N mesures bruitées S_i et μ_i , $i = 1 : N$ de $S(t_i)$ et $\mu(S(t_i))$

Méthode 1: résoudre le problème de moindres carrés:

$$(\hat{a}, \hat{k}) = \operatorname{argmin}_{(a,k) \in \Omega} \sum_{i=1}^N \beta_i \left(\mu_i - k \frac{S_i}{a + S_i} \right)^2$$

⇒ modèle non linéaire par rapport à a

⇒ utilisation de méthodes non linéaires

Modèles de régression linéaire

Exemple: taux de croissance

Fonction de Monod: $\mu(S) = k \frac{S}{a + S}$

Problème: trouver les paramètres a et k de la fonction à partir de N mesures bruitées S_i et μ_i , $i = 1 : N$ de $S(t_i)$ et $\mu(S(t_i))$

Méthode 2: transformer le problème pour le rendre linéaire

Modèles de régression linéaire

Exemple: taux de croissance

$$\text{Fonction de Monod: } \mu(S) = k \frac{S}{a + S}$$

Problème: trouver les paramètres a et k de la fonction à partir de N mesures bruitées S_i et μ_i , $i = 1 : N$ de $S(t_i)$ et $\mu(S(t_i))$

Méthode 2: transformer le problème pour le rendre linéaire
Par exemple:

$$\begin{aligned} \mu_i = k \frac{S_i}{a + S_i} &\iff (a + S_i)\mu_i = kS_i \iff S_i\mu_i = kS_i - a\mu_i \\ &\iff S_i\mu_i = [S_i \mid -\mu_i] \begin{bmatrix} k \\ a \end{bmatrix} \iff y_i^m = y_i(\theta) \end{aligned}$$

$$\text{avec: } y_i(\theta) = \phi_i^T \theta, y_i^m = S_i\mu_i, \theta = \begin{bmatrix} k \\ a \end{bmatrix} \text{ et } \phi_i = \begin{bmatrix} S_i \\ -\mu_i \end{bmatrix}$$

Modèles de régression linéaire

Exemple: taux de croissance

Fonction de Monod: $\mu(S) = k \frac{S}{a + S}$

Problème: trouver les paramètres a et k de la fonction à partir de N mesures bruitées S_i et μ_i , $i = 1 : N$ de $S(t_i)$ et $\mu(S(t_i))$

Méthode 2: transformer le problème pour le rendre linéaire
Puis résoudre le problème de moindres carrés:

$$(\hat{a}, \hat{k}) = \operatorname{argmin}_{(a,k) \in \Omega} \sum_{i=1}^N \beta_i \left([S_i \mid -\mu_i] \begin{bmatrix} k \\ a \end{bmatrix} - S_i \mu_i \right)^2$$

Modèles de régression linéaire

Exemple: taux de croissance

$$\text{Fonction de Monod: } \mu(S) = k \frac{S}{a + S}$$

Problème: trouver les paramètres a et k de la fonction à partir de N mesures bruitées S_i et μ_i , $i = 1 : N$ de $S(t_i)$ et $\mu(S(t_i))$

Méthode 2: transformer le problème pour le rendre linéaire
Puis résoudre le problème de moindres carrés:

$$(\hat{a}, \hat{k}) = \operatorname{argmin}_{(a,k) \in \Omega} \sum_{i=1}^N \beta_i \left([S_i \mid -\mu_i] \begin{bmatrix} k \\ a \end{bmatrix} - S_i \mu_i \right)^2$$

Solution:

$$\hat{\theta} = \left[\sum_{i=1}^N \beta_i \begin{bmatrix} S_i \\ -\mu_i \end{bmatrix} [S_i \mid -\mu_i] \right]^{-1} \sum_{i=1}^N \beta_i \begin{bmatrix} S_i \\ -\mu_i \end{bmatrix} S_i \mu_i$$

Modèles de régression linéaire

Exemple: taux de croissance

Fonction de Monod: $\mu(S) = k \frac{S}{a + S}$

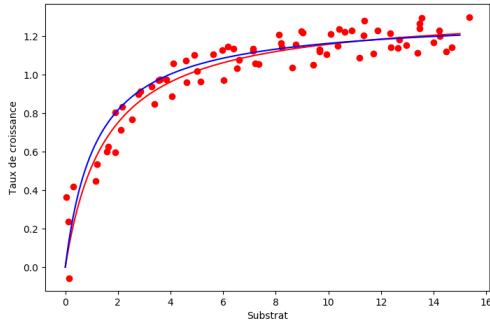
Problème: trouver les paramètres a et k de la fonction à partir de N mesures bruitées S_i et μ_i , $i = 1 : N$ de $S(t_i)$ et $\mu(S(t_i))$

Modèles de régression linéaire

Exemple: taux de croissance

$$\text{Fonction de Monod: } \mu(S) = k \frac{S}{a + S}$$

Problème: trouver les paramètres a et k de la fonction à partir de N mesures bruitées S_i et μ_i , $i = 1 : N$ de $S(t_i)$ et $\mu(S(t_i))$



— modèle exact
• mesures bruitées
— modèle identifié

coefficient k exact = 1.34; estimé = 1.30061095508

coefficient a exact = 1.57; estimé = 1.18748984841

$$k = 1.34, a = 1.57$$

$N = 74$ mesures

$\sigma_1 = 0.2$ écart type du bruit
de mesure sur μ_i

$\sigma_2 = 0.8$ écart type du bruit
de mesure sur S_i

Cas non linéaire

Algorithmes de minimisation

$$\hat{\theta} = \operatorname{argmin}_{\theta \in \Omega} J(\theta)$$

La plupart du temps, J fonction **non linéaire** de θ

- ⇒ impossible de calculer analytiquement la solution
- ⇒ utilisation d'**algorithmes d'optimisation**

Cas non linéaire

Algorithmes de minimisation

$$\hat{\theta} = \operatorname{argmin}_{\theta \in \Omega} J(\theta)$$

La plupart du temps, J fonction **non linéaire** de θ

- ⇒ impossible de calculer analytiquement la solution
- ⇒ utilisation d'**algorithmes d'optimisation**

Objectif des algorithmes: trouver le minimum (ou maximum) d'une fonction et la valeur en laquelle elle atteint cet extremum sur un domaine Ω donné

Cas non linéaire

Algorithmes de minimisation

$$\hat{\theta} = \operatorname{argmin}_{\theta \in \Omega} J(\theta)$$

La plupart du temps, J fonction **non linéaire** de θ

- ⇒ impossible de calculer analytiquement la solution
- ⇒ utilisation d'**algorithmes d'optimisation**

Objectif des algorithmes: trouver le minimum (ou maximum) d'une fonction et la valeur en laquelle elle atteint cet extremum sur un domaine Ω donné

- algorithmes **itératifs**
- démarrent d'une **valeur initiale** donnée

Cas non linéaire

Algorithmes de minimisation

$$\hat{\theta} = \operatorname{argmin}_{\theta \in \Omega} J(\theta)$$

La plupart du temps, J fonction **non linéaire** de θ

- ⇒ impossible de calculer analytiquement la solution
- ⇒ utilisation d'**algorithmes d'optimisation**

Objectif des algorithmes: trouver le minimum (ou maximum) d'une fonction et la valeur en laquelle elle atteint cet extremum sur un domaine Ω donné

- algorithmes **itératifs**
- démarrent d'une **valeur initiale** donnée

Valeur initiale = première estimation "grossière" des paramètres.

Minimum global ou local

Importance de la condition initiale

Existence de deux types de minimum

Pour tout $a \in \Omega$, $f(a)$ est un

- **minimum global de f** si $f(a)$ est la plus petite valeur atteinte par f sur tout le domaine Ω
- **minimum local de f** si il existe un voisinage V de a tel que $f(a)$ est la plus petite valeur atteinte par f sur V

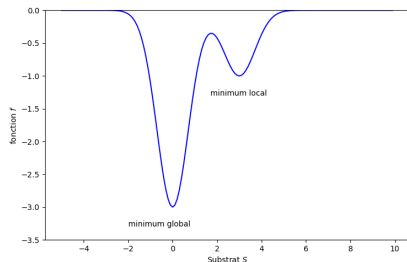
Minimum global ou local

Importance de la condition initiale

Existence de deux types de minimum

Pour tout $a \in \Omega$, $f(a)$ est un

- **minimum global de f** si $f(a)$ est la plus petite valeur atteinte par f sur tout le domaine Ω
- **minimum local de f** si il existe un voisinage V de a tel que $f(a)$ est la plus petite valeur atteinte par f sur V



exemple:

la fonction $f \mapsto -3e^{-x^2} - e^{-(x-3)^2}$
admet deux minimums locaux
sur $[-5, 10]$ dont un est global

Minimum global ou local

Importance de la condition initiale

Existence de deux types de minimum

Pour tout $a \in \Omega$, $f(a)$ est un

- **minimum global de f** si $f(a)$ est la plus petite valeur atteinte par f sur tout le domaine Ω
- **minimum local de f** si il existe un voisinage V de a tel que $f(a)$ est la plus petite valeur atteinte par f sur V

Importance de **bien choisir la condition initiale** car les **algorithmes convergent généralement vers un minimum local**, souvent le plus proche de la condition initiale.

Méthode du gradient

Problème considéré

Algorithme du gradient aussi appelé **algorithme de plus forte pente** ou **de plus profonde descente** = un algorithme d'optimisation (minimisation ou maximisation) de fonction.

Méthode du gradient

Problème considéré

Algorithme du gradient aussi appelé **algorithme de plus forte pente** ou **de plus profonde descente** = un algorithme d'optimisation (minimisation ou maximisation) de fonction.

Problème considéré: minimisation d'une fonction

$f : x \in \Omega \subset \mathbb{R}^n \mapsto f(x) \in \mathbb{R}$ sur un domaine Ω :

$$\hat{x} = \operatorname{argmin}_{x \in \Omega} f(x)$$

Notation: $\nabla f(x)$ = gradient de f en x :

$$\nabla f(x) = \begin{bmatrix} \partial_{x_1} f(x) \\ \vdots \\ \partial_{x_n} f(x) \end{bmatrix}$$

Méthode du gradient

Direction de descente

direction de descente = direction, donc vecteur $d \in \Omega \setminus \{0\}$ selon laquelle, au voisinage de x , la fonction f décroît.

Méthode du gradient

Direction de descente

direction de descente = direction, donc vecteur $d \in \Omega \setminus \{0\}$ selon laquelle, au voisinage de x , la fonction f décroît.

- ⇒ si on suit cette direction, on se rapproche d'un minimum
- ⇒ définie localement autour d'un point $x \in \Omega$

Méthode du gradient

Direction de descente

direction de descente = direction, donc vecteur $d \in \Omega \setminus \{0\}$ selon laquelle, au voisinage de x , la fonction f décroît.

⇒ si on suit cette direction, on se rapproche d'un minimum

⇒ définie localement autour d'un point $x \in \Omega$

Définition mathématique:

$d \in \Omega \setminus \{0\}$ est une **direction de descente** en x pour f si il existe un intervalle $[0, \alpha_0]$ tel que:

$$f(x + \alpha d) \leq f(x), \quad \forall \alpha \in [0, \alpha_0]$$

d est une **direction de descente stricte** si l'inégalité est stricte ($<$ au lieu de \leq).

Méthode du gradient

Direction de descente

direction de descente = direction, donc vecteur $d \in \Omega \setminus \{0\}$ selon laquelle, au voisinage de x , la fonction f décroît.

⇒ si on suit cette direction, on se rapproche d'un minimum

⇒ définie localement autour d'un point $x \in \Omega$

Résultat mathématique

Si $\nabla f(x) \neq 0$, alors:

$d = -\nabla f(x)$ est une direction de descente stricte en x pour f .

avec une fonction f est la direction du gradient et d'autre direction de descente.

Méthode du gradient

Algorithme du gradient

Méthode du gradient

Algorithme du gradient

1. Choix des paramètres: nombre d'itérations maximal N , seuil de précision ϵ et valeur initiale x_0 pour x

Méthode du gradient

Algorithme du gradient

1. Choix des paramètres: nombre d'itérations maximal N , seuil de précision ϵ et valeur initiale x_0 pour x
2. Initialisation: $k = 0$ et calcul de $\nabla f(x_0)$

Méthode du gradient

Algorithme du gradient

1. Choix des paramètres: nombre d'itérations maximal N , seuil de précision ϵ et valeur initiale x_0 pour x
2. Initialisation: $k = 0$ et calcul de $\nabla f(x_0)$
3. Itérations: Tant que $k + 1 \leq N$ et $\|\nabla f(x_k)\| > \epsilon$ alors

$$\begin{aligned}x_{k+1} &= x_k - \alpha_k \nabla f(x_k) \\ k &= k + 1\end{aligned}$$

où α_k peut être choisi selon différentes méthodes.

Méthode du gradient

Algorithme du gradient

1. Choix des paramètres: nombre d'itérations maximal N , seuil de précision ϵ et valeur initiale x_0 pour x
2. Initialisation: $k = 0$ et calcul de $\nabla f(x_0)$
3. Itérations: Tant que $k + 1 \leq N$ et $\|\nabla f(x_k)\| > \epsilon$ alors

$$\begin{aligned}x_{k+1} &= x_k - \alpha_k \nabla f(x_k) \\ k &= k + 1\end{aligned}$$

où α_k peut être choisi selon différentes méthodes.

4. Solution approchée $\hat{x} \simeq x_k$ et $\min_{x \in \Omega} f(x) \simeq f(x_k)$

Méthode du gradient

Algorithme du gradient

1. Choix des paramètres: nombre d'itérations maximal N , seuil de précision ϵ et valeur initiale x_0 pour x
2. Initialisation: $k = 0$ et calcul de $\nabla f(x_0)$
3. Itérations: Tant que $k + 1 \leq N$ et $\|\nabla f(x_k)\| > \epsilon$ alors

$$\begin{aligned}x_{k+1} &= x_k - \alpha_k \nabla f(x_k) \\ k &= k + 1\end{aligned}$$

où α_k peut être choisi selon différentes méthodes.

4. Solution approchée $\hat{x} \simeq x_k$ et $\min_{x \in \Omega} f(x) \simeq f(x_k)$

Deux choix de α_k classiques:

- α_k constante indépendante de $k \Rightarrow$ **gradient à pas fixe**
- α_k choisi pour minimiser $f(x_k - \alpha_k \nabla f(x_k)) \Rightarrow$ **gradient à pas optimal**

Méthode du gradient

Exemple: taux de croissance

Fonction de Monod: $\mu(S) = k \frac{S}{a + S}$

Problème: trouver les paramètres a et k de la fonction à partir de N mesures bruitées S_i et μ_i , $i = 1 : N$ de $S(t_i)$ et $\mu(S(t_i))$

Méthode 1: résoudre le problème de moindres carrés:

$$(\hat{a}, \hat{k}) = \operatorname{argmin}_{(a,k) \in \Omega} \sum_{i=1}^N \left(k \frac{S_i}{a + S_i} - \mu_i \right)^2 = \operatorname{argmin}_{(a,k) \in \Omega} f(a, k)$$

Méthode du gradient

Exemple: taux de croissance

$$\text{Fonction de Monod: } \mu(S) = k \frac{S}{a + S}$$

Problème: trouver les paramètres a et k de la fonction à partir de N mesures bruitées S_i et μ_i , $i = 1 : N$ de $S(t_i)$ et $\mu(S(t_i))$

Méthode 1: résoudre le problème de moindres carrés:

$$(\hat{a}, \hat{k}) = \operatorname{argmin}_{(a,k) \in \Omega} \sum_{i=1}^N \left(k \frac{S_i}{a + S_i} - \mu_i \right)^2 = \operatorname{argmin}_{(a,k) \in \Omega} f(a, k)$$

Le gradient de f :

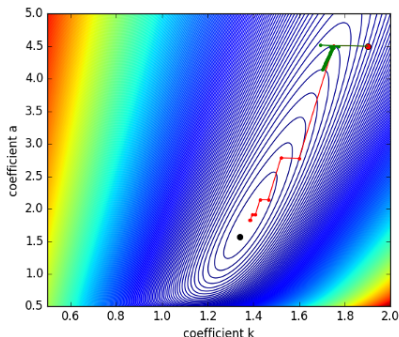
$$\nabla f(k, a) = \begin{bmatrix} \partial_k f(k, a) \\ \partial_a f(k, a) \end{bmatrix} = \begin{bmatrix} \frac{2}{N} \sum_{i=1}^N \frac{S_i}{a + S_i} \left(k \frac{S_i}{a + S_i} - \mu_i \right) \\ -\frac{2}{N} \sum_{i=1}^N k \frac{S_i}{(a + S_i)^2} \left(k \frac{S_i}{a + S_i} - \mu_i \right) \end{bmatrix}$$

Méthode du gradient

Exemple: taux de croissance

$$\text{Fonction de Monod: } \mu(S) = k \frac{S}{a + S}$$

Problème: trouver les paramètres a et k de la fonction à partir de N mesures bruitées S_i et μ_i , $i = 1 : N$ de $S(t_i)$ et $\mu(S(t_i))$



- algorithme du gradient à pas optimal
- algorithme du gradient à pas constant
- solution exacte

**** Gradient à pas optimal ****

nombre d'itération =9

norme du gradient =0.000404466802641

coefficient k exact =1.34; estimé =1.38280601099

coefficient a exact =1.57; estimé =1.83032580136

****Gradient à pas constant****

nombre d'itération =35

norme du gradient =0.00550627319589

coefficient k exact =1.34; estimé =1.70307257574

coefficient a exact =1.57; estimé =4.13896118471

Méthode de Newton-Raphson

Problème considéré

Méthode de Newton-Raphson = algorithme destiné à trouver une approximation numérique d'un zéro (ou racine) d'une fonction f , c'est à dire la valeur de x telle que $f(x) = 0$.

Utilisation pour l'optimisation: pour un ensemble fermé Ω :

$$\hat{x} = \operatorname{argmin}_{x \in \Omega} f(x) \Leftrightarrow f'(\hat{x}) = 0 \text{ OU } \hat{x} \in \partial\Omega$$

⇒ application de la méthode de Newton-Raphson à f'

Méthode de Newton-Raphton

Problème considéré

Méthode de Newton-Raphton = algorithme destiné à trouver une approximation numérique d'un zéro (ou racine) d'une fonction f , c'est à dire la valeur de x telle que $f(x) = 0$.

Utilisation pour l'optimisation: pour un ensemble fermé Ω :

$$\hat{x} = \operatorname{argmin}_{x \in \Omega} f(x) \Leftrightarrow f'(\hat{x}) = 0 \text{ OU } \hat{x} \in \partial\Omega$$

⇒ application de la méthode de Newton-Raphton à f'

Problème considéré:

Trouver $\hat{x} \in \Omega$ tel que:

$$g(\hat{x}) = 0$$

Méthode de Newton-Raphson

Principe

Formule de Taylor à l'ordre 1 de g au voisinage de x_k :

$$g(x) = g(x_k) + g'(x_k)(x - x_k) + R_1(x)$$

où $R_1(x)$ est négligeable devant les autres termes.

Méthode de Newton-Raphson

Principe

Formule de Taylor à l'ordre 1 de g au voisinage de x_k :

$$g(x) = g(x_k) + g'(x_k)(x - x_k) + R_1(x)$$

où $R_1(x)$ est négligeable devant les autres termes.

Autrement dit: si x proche de x_k alors on néglige $R_1(x)$ et on a:

$$g(x) \simeq g(x_k) + g'(x_k)(x - x_k)$$

Méthode de Newton-Raphson

Principe

Formule de Taylor à l'ordre 1 de g au voisinage de x_k :

$$g(x) = g(x_k) + g'(x_k)(x - x_k) + R_1(x)$$

où $R_1(x)$ est négligeable devant les autres termes.

Autrement dit: si x proche de x_k alors on néglige $R_1(x)$ et on a:

$$g(x) \simeq g(x_k) + g'(x_k)(x - x_k)$$

Rappel:

$y = g(x_k) + g'(x_k)(x - x_k)$: équation de la tangente au graphe de g en x_k

Méthode de Newton-Raphson

Principe

Formule de Taylor à l'ordre 1 de g au voisinage de x_k :

$$g(x) = g(x_k) + g'(x_k)(x - x_k) + R_1(x)$$

où $R_1(x)$ est négligeable devant les autres termes.

Autrement dit: si x proche de x_k alors on néglige $R_1(x)$ et on a:

$$g(x) \simeq g(x_k) + g'(x_k)(x - x_k)$$



“autour de x_k la courbe de g est à peu près égale à sa tangente”

Rappel:

$y = g(x_k) + g'(x_k)(x - x_k)$: équation de la tangente au graphe de g en x_k

Méthode de Newton-Raphson

Principe

Idée: Au lieu de chercher $x \in \Omega$ tel que:

$$g(x) = 0$$

on cherche $x \in \Omega$ tel que:

$$g(x_k) + g'(x_k)(x - x_k) = 0$$

Méthode de Newton-Raphson

Principe

Idée: Au lieu de chercher $x \in \Omega$ tel que:

$$g(x) = 0$$

on cherche $x \in \Omega$ tel que:

$$\begin{aligned} g(x_k) + g'(x_k)(x - x_k) &= 0 \\ \Leftrightarrow x &= x_k - \frac{g(x_k)}{g'(x_k)} \end{aligned}$$

Méthode de Newton-Raphson

Principe

Idée: Au lieu de chercher $x \in \Omega$ tel que:

$$g(x) = 0$$

on cherche $x \in \Omega$ tel que:

$$\begin{aligned} g(x_k) + g'(x_k)(x - x_k) &= 0 \\ \iff x &= x_k - \frac{g(x_k)}{g'(x_k)} \end{aligned}$$

\Rightarrow nouvelle valeur de x supposée plus proche du zéro de f que x_k

Méthode de Newton-Raphson

Principe

Idée: Au lieu de chercher $x \in \Omega$ tel que:

$$g(x) = 0$$

on cherche $x \in \Omega$ tel que:

$$\begin{aligned} g(x_k) + g'(x_k)(x - x_k) &= 0 \\ \iff x &= x_k - \frac{g(x_k)}{g'(x_k)} \end{aligned}$$

\Rightarrow nouvelle valeur de x supposée plus proche du zéro de f que x_k

\Rightarrow itération $x_{k+1} = x$

Méthode de Newton-Raphson

Principe

Idée: Au lieu de chercher $x \in \Omega$ tel que:

$$g(x) = 0$$

on cherche $x \in \Omega$ tel que:

$$\begin{aligned} g(x_k) + g'(x_k)(x - x_k) &= 0 \\ \iff x &= x_k - \frac{g(x_k)}{g'(x_k)} \end{aligned}$$

\Rightarrow nouvelle valeur de x supposée plus proche du zéro de f que x_k

\Rightarrow itération $x_{k+1} = x$

\Rightarrow convergence vers le zéro le plus proche

Méthode de Newton-Raphson

Algorithme

Méthode de Newton-Raphson

Algorithme

1. Choix des paramètres: nombre d'itérations maximal N , seuil de précision ϵ et valeur initiale x_0 pour x

Méthode de Newton-Raphson

Algorithme

1. Choix des paramètres: nombre d'itérations maximal N , seuil de précision ϵ et valeur initiale x_0 pour x
2. Initialisation: $k = 0$ et calcul de $g(x_0)$

Méthode de Newton-Raphson

Algorithme

1. Choix des paramètres: nombre d'itérations maximal N , seuil de précision ϵ et valeur initiale x_0 pour x
2. Initialisation: $k = 0$ et calcul de $g(x_0)$
3. Itérations: Tant que $k + 1 \leq N$ et $g(x_k) > \epsilon$ alors

$$\begin{aligned}x_{k+1} &= x_k - \frac{g(x_k)}{g'(x_k)} \\ k &= k + 1\end{aligned}$$

Méthode de Newton-Raphson

Algorithme

1. Choix des paramètres: nombre d'itérations maximal N , seuil de précision ϵ et valeur initiale x_0 pour x
2. Initialisation: $k = 0$ et calcul de $g(x_0)$
3. Itérations: Tant que $k + 1 \leq N$ et $g(x_k) > \epsilon$ alors

$$\begin{aligned}x_{k+1} &= x_k - \frac{g(x_k)}{g'(x_k)} \\ k &= k + 1\end{aligned}$$

4. Solution approchée $\hat{x} \simeq x_k$

Méthode de Newton-Raphson

Exemple: points d'équilibre

Modèle de croissance d'une biomasse B sur un substrat S dans un réacteur batch:

$$\begin{cases} \frac{dB}{dt} = (\mu(S) - \frac{Q}{V})B \\ \frac{dS}{dt} = -k\mu(S)B + \frac{Q}{V}(S_0 - S) \end{cases}$$

Méthode de Newton-Raphson

Exemple: points d'équilibre

Modèle de croissance d'une biomasse B sur un substrat S dans un réacteur batch:

$$\begin{cases} \frac{dB}{dt} = (\mu(S) - \frac{Q}{V})B \\ \frac{dS}{dt} = -k\mu(S)B + \frac{Q}{V}(S_0 - S) \end{cases}$$

Point d'équilibre non nul = solution de l'équation $\mu(S) = \frac{Q}{V}$
= zéro de la fonction $g : S \mapsto \mu(S) - \frac{Q}{V}$

Méthode de Newton-Raphson

Exemple: points d'équilibre

Modèle de croissance d'une biomasse B sur un substrat S dans un réacteur batch:

$$\begin{cases} \frac{dB}{dt} = (\mu(S) - \frac{Q}{V})B \\ \frac{dS}{dt} = -k\mu(S)B + \frac{Q}{V}(S_0 - S) \end{cases}$$

Point d'équilibre non nul = solution de l'équation $\mu(S) = \frac{Q}{V}$
= zéro de la fonction $g : S \mapsto \mu(S) - \frac{Q}{V}$

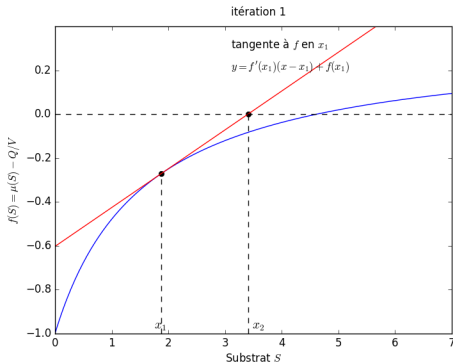
⇒ application de l'algorithme de Newton-Raphson à g

Méthode de Newton-Raphson

Exemple: points d'équilibre

Modèle de croissance d'une biomasse B sur un substrat S dans un réacteur batch:

$$\begin{cases} \frac{dB}{dt} = (\mu(S) - \frac{Q}{V})B \\ \frac{dS}{dt} = -k\mu(S)B + \frac{Q}{V}(S_0 - S) \end{cases}$$

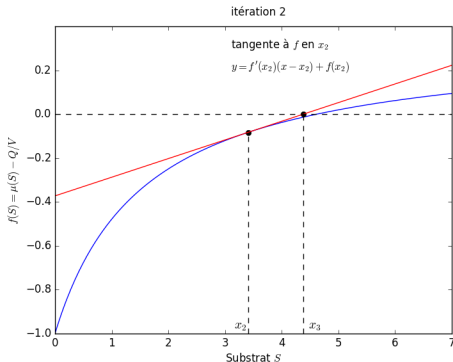


Méthode de Newton-Raphson

Exemple: points d'équilibre

Modèle de croissance d'une biomasse B sur un substrat S dans un réacteur batch:

$$\begin{cases} \frac{dB}{dt} = (\mu(S) - \frac{Q}{V})B \\ \frac{dS}{dt} = -k\mu(S)B + \frac{Q}{V}(S_0 - S) \end{cases}$$

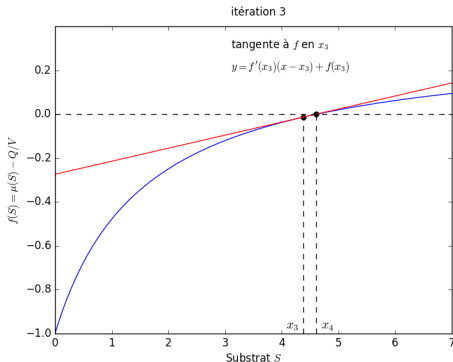


Méthode de Newton-Raphson

Exemple: points d'équilibre

Modèle de croissance d'une biomasse B sur un substrat S dans un réacteur batch:

$$\begin{cases} \frac{dB}{dt} = (\mu(S) - \frac{Q}{V})B \\ \frac{dS}{dt} = -k\mu(S)B + \frac{Q}{V}(S_0 - S) \end{cases}$$

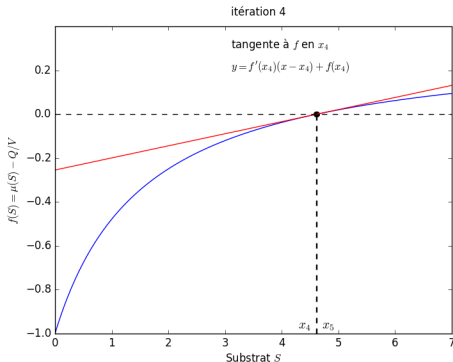


Méthode de Newton-Raphson

Exemple: points d'équilibre

Modèle de croissance d'une biomasse B sur un substrat S dans un réacteur batch:

$$\begin{cases} \frac{dB}{dt} = (\mu(S) - \frac{Q}{V})B \\ \frac{dS}{dt} = -k\mu(S)B + \frac{Q}{V}(S_0 - S) \end{cases}$$

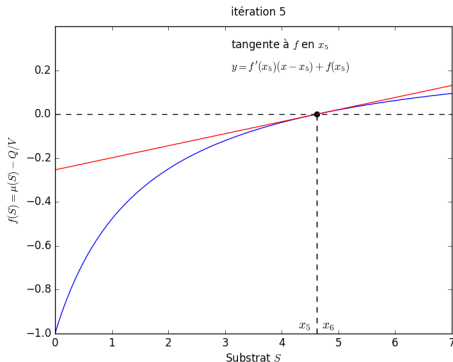


Méthode de Newton-Raphson

Exemple: points d'équilibre

Modèle de croissance d'une biomasse B sur un substrat S dans un réacteur batch:

$$\begin{cases} \frac{dB}{dt} = (\mu(S) - \frac{Q}{V})B \\ \frac{dS}{dt} = -k\mu(S)B + \frac{Q}{V}(S_0 - S) \end{cases}$$

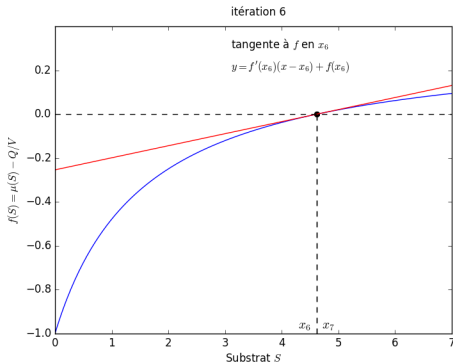


Méthode de Newton-Raphson

Exemple: points d'équilibre

Modèle de croissance d'une biomasse B sur un substrat S dans un réacteur batch:

$$\begin{cases} \frac{dB}{dt} = (\mu(S) - \frac{Q}{V})B \\ \frac{dS}{dt} = -k\mu(S)B + \frac{Q}{V}(S_0 - S) \end{cases}$$

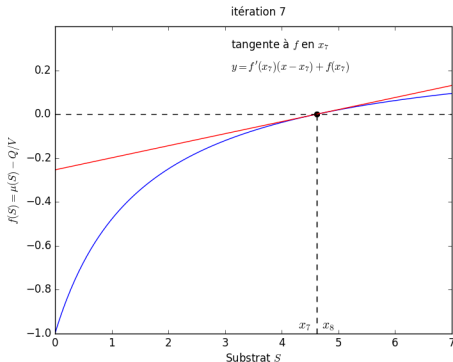


Méthode de Newton-Raphson

Exemple: points d'équilibre

Modèle de croissance d'une biomasse B sur un substrat S dans un réacteur batch:

$$\begin{cases} \frac{dB}{dt} = (\mu(S) - \frac{Q}{V})B \\ \frac{dS}{dt} = -k\mu(S)B + \frac{Q}{V}(S_0 - S) \end{cases}$$

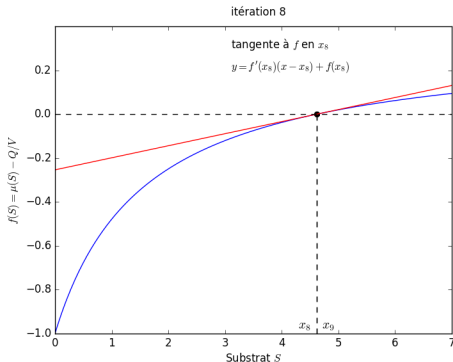


Méthode de Newton-Raphson

Exemple: points d'équilibre

Modèle de croissance d'une biomasse B sur un substrat S dans un réacteur batch:

$$\begin{cases} \frac{dB}{dt} = (\mu(S) - \frac{Q}{V})B \\ \frac{dS}{dt} = -k\mu(S)B + \frac{Q}{V}(S_0 - S) \end{cases}$$



Quelques remarques pour finir

- **Quels paramètres doit on identifier?:**

Analyse de sensibilité pour quantifier l'impact de la variation des paramètres sur les sorties du modèle.

⇒ réduction du nombre de paramètres à identifier: on identifie que les plus importants

- **Fonctions pré-codées:** en python/scilab/R/etc.