Capacités numériques

Résolution numérique de $f(x) = 0$

Lundi 9 & Vendredi 13 mai 2022

Capacités exigibles : Résolution d'une équation algébrique ou d'une équation transcendante : méthode dichotomique, méthode de Newton.

- Déterminer, en s'appuyant sur une représentation graphique, un intervalle adapté à la recherche numérique d'une racine par la méthode dichotomique ou par la méthode de Newton.
- Mettre en œuvre la méthode dichotomique ou la méthode de Newton afin de résoudre une équation avec une précision donnée.
- Utiliser les fonctions `bisect` ou `newton` de la bibliothèque `scipy.optimize` (leurs spécifications étant fournies).

Travail à faire AVANT la séance du 9 mai :

- Lire toute la partie I.
- Compléter la fonction du §I.2
- PCSI-option PSI : Exercice n°1, tratier la question Q1.
- PCSI-option PC : Exercice n°2, traiter les questions Q1, Q2 et Q3.

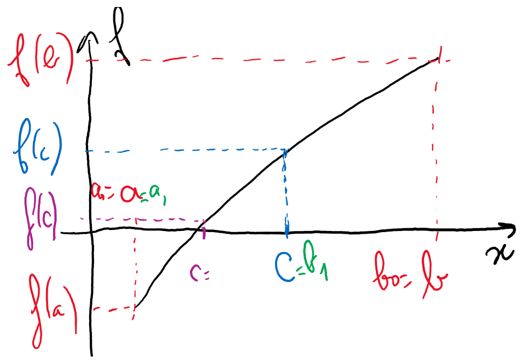
I Méthode de la dichotomie

I.1 Rappel de la méthode

Vous avez déjà étudié la dichotomie en TP d'informatique au premier semestre (cf TP3).

On considère une fonction $f : I = [a, b] \longrightarrow \mathbb{R}$, continue et strictement monotone sur l'intervalle $[a, b]$. De plus, on suppose que f change de signe sur l'intervalle, soit $f(a) \times f(b) < 0$.

On souhaite déterminer une valeur approchée de c tel que $f(c) = 0$.



On cherche la solution approchée à l'équation $f(x) = 0$ sur l'intervalle $[a, b]$, où f est une fonction strictement monotone.

On se place au milieu de l'intervalle $[a, b]$ ($c = (a + b)/2$) et on compare le signe de $f(a)$ avec le signe de $f(c)$ en regardant le signe de $f(a) \times f(c)$.

Si $f(a) \times f(c) < 0$, alors f s'annule sur l'intervalle $[a, c]$. Et on procède de la même façon sur ce nouvel intervalle.

Sinon, f s'annule sur l'intervalle $[c, b]$. Et on procède de la même façon sur ce nouvel intervalle.

On procède ainsi jusqu'à un certain critère imposé, qui peut porter soit sur la largeur de l'intervalle de recherche (on cherche jusqu'à ce que $|b - a|$ devienne inférieur à un certain ε), soit sur la proximité de f à 0 (on cherche jusqu'à ce que $|f(c)|$ devienne inférieur à un certain ε).

Comment choisir l'intervalle $[a, b]$ de recherche ?

- Représenter la fonction f dont on cherche l'annulation ;
- Déterminer un intervalle pertinent, sur lequel f est strictement monotone et s'annule.

I.2 Implémentation en python

On va écrire la fonction **dichotomie** qui renverra la valeur approchée de c tel que $f(c)=0$ à ε près.

```python
def dichotomie(f,a,b,eps):
    '''
    f : fonction dont on cherche le zéro par dichotomie
    a,b : bornes de l'intervalle sur lequel on recherche l'annulation de f, a<b
    eps : précision souhaitée sur c, tel que |f(c)|<eps
    renvoie la valeur de c
    '''
    assert f(a)*f(b)<0 # on s'assure que l'annulation existe bien
    while                                : # (à compléter) : tant que |b-a|>eps
        c=(a+b)/2 # on regarde au milieu de l'intervalle
        if f(a)*f(c)<0: # l'annulation est sur [a,c]
                        # (à compléter)
        else:           # annulation sur [c,b]
                        # (à compléter)
    return              # (à compléter) : renvoie la valeur de c avec la précision souhaitée
```

I.3 Utilisation

Exercice 1 En SII : Direction Assisté Electrique (position angulaire de la roue droite pour une position extrême de la crémaillère)

- $AB'=\lambda$ (λ paramètre la position de la crémaillère)
- θ_{61} paramètre la position angulaire de la roue droite
- BB' =590 mm
- $BC=B'C'$ =356 mm
- DD' =1350 mm
- $CD=C'D'$ =120 mm
- Distance de la droite (D,D') à l'axe de crémaillère : 148 mm
- Rayon primitif du pignon de la crémaillère : 5,3 mm
- $\theta_{51}=\theta_{61}+78°$

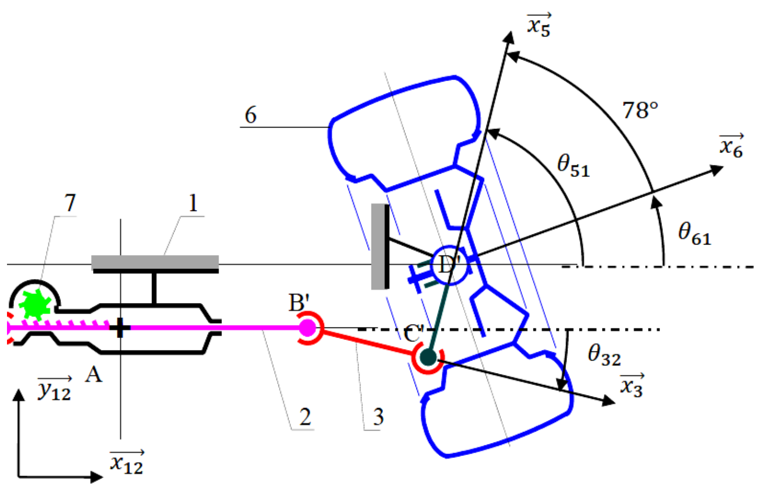


Schéma du système de direction avec uniquement la roue droite représentée

La fermeture géométrique passant par le point $AB'C'D'A$ permet d'obtenir la loi entrée/sortie du mécanisme (relation entre la position de la crémaillère λ et la position angulaire de la roue droite θ_{61}) :

$$\lambda = 675 - 120\times\cos(\theta_{61}+1,36) - \sqrt{356^2-\Big(148-120\times\sin(\theta_{61}+1,36)\Big)^2}$$

(les angles sont en radian et les longueurs en mm)

Objectif : On cherche à déterminer la position angulaire de la roue droite θ_{61} lorsque la crémaillère est dans la position extrême $\lambda=358$ mm

Q1. * Écrire en python la fonction $f(\theta_{61})$ correspondant à la fonction dont on cherche l'annulation afin de connaître la position angulaire de la roue de droite pour $\lambda=358$ mm.

Q2. Représenter la fonction f sur un intervalle pertinent, en adaptant le code ci-dessous :

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

x=np.linspace(x1,x2,n) # (à compléter) : tableau de  valeurs de x
   réparties équitablement entre x1 et x2
plt.plot(abscisse, ordonnée) # (à compléter)
plt.title('titre') # (à compléter)
plt.xlabel('abscisse (unité)') # (à compléter)
plt.ylabel('ordonnée (unité)') # (à compléter)
plt.show() # pour afficher le graphiqe
```

Q3. En déduire un intervalle pertinent pour appliquer l'algorithme de dichotomie.

Q4. Faire appel à la fonction dichotomie afin de déterminer la position angulaire de la roue correspondant à une position de la crémaillère de 358 mm. Coder la conversion pour afficher cette valeur en degré.

Exercice 2 En chimie : Optimisation d'un procédé chimique

Une partie des objectifs de la chimie est la synthèse d'espèces chimiques : il faut en permanence chercher les conditions opératoires optimales. Il existe a priori trois moyens d'action à disposition d'un opérateur pour optimiser la production d'un produit souhaité : la température, la pression totale et la composition du mélange initial.

Cadre de l'exercice

On s'intéresse à la synthèse industrielle de l'ammoniac :

$$N_2(g) + 3H_2(g) = 2NH_3(g)$$

La constante d'équilibre varie avec la température T (en Kelvin) selon :

$$K^\circ(T) = \exp\left(\frac{11095}{T} - 23,9\right)$$

Dans la suite, on notera $n_0 = 1$ mol et $n_1 = 3$ mol les quantités respectives de diazote et de dihydrogène. La quantité initiale d'ammoniac sera nulle.

Q1. * *Question préliminaire*

Montrer que, pour un avancement à l'équilibre $\xi_{éq}$, la constante d'équilibre peut s'écrire sous la forme :

$$K^\circ = \frac{4\xi_{éq}^2 \times \left(n_0 + n_1 - 2\xi_{éq}\right)^2}{\left(n_0 - \xi_{éq}\right) \times \left(n_1 - 3\xi_{éq}\right)^3} \times \left(\frac{P^\circ}{P}\right)^2$$

Partie 1 : Optimisation par modification de la constante d'équilibre

Q2. * Écrire la fonction f (mathématique, pas en python) dont $\xi_{éq}$ est le 0.

Q3. * Écrire la fonction `f_etude(ksi)` en python.

Q4. Quelles sont les valeurs extrêmes de l'avancement à l'équilibre ? Cela donne les bornes d'application de la fonction `dichotomie`.

Q5. Compléter le code ci-dessous permettant de créer deux listes : l'une de température (de 300 K à 800 K, par pas de 1 Kelvin) et l'autre contenant les valeurs $\xi_{éq}$ pour ces différentes températures.

```python
X=[] # liste des températures
Y=[] # liste des valeurs de ksi_éq
for T in range (..... , .....): # (à compléter) : T varie entre 300 et
   800
    K=                              # (à compléter) : valeur de K à la
   température T
    ksi_eq=                                  # (à compléter) : déterminer ksi_éq
    à l'équilibre par dichotomie
                                              # (à compléter) : ajout de T et de
    ksi_éq aux listes X et Y
```

Q6. Représenter graphiquement l'avancement à l'équilibre en fonction de la température (pour des températures telles que 300 K $< T <$ 800 K).

Q7. À quelle température faut-il se placer pour optimiser la transformation ?

Q8. Industriellement, la transformation se fait à 400 K. Commenter et proposer une interprétation.

Partie 2 : Optimisation du quotient réactionnel

Q9. Soit α la fraction molaire d'ammoniac. Montrer que : $\alpha = \dfrac{2\xi_{\text{éq}}}{n_0 + n_1 - 2\xi_{\text{éq}}}$

Q10. Représenter graphiquement la fraction molaire d'ammoniac en fonction que la quantité initiale de dihydrogène. On travaillera sur un intervalle tel que $0,1 < n_1 < 10$ (en mol).

Q11. Interpréter la courbe obtenue.

Q12. Quel est l'intérêt de travailler avec la fraction molaire d'ammoniac plutôt qu'avec l'avancement, ici ?

Q13. Écrire les instructions permettant de déterminer la quantité de matière n_1 correspondant à l'avancement à l'équilibre maximal.

Exercice 3 En SII : Direction Assisté Electrique (position angulaire de la roue droite pour une position extrême de la crémaillère)

Il s'agit du même système que celui décrit à l'exercice 1.

Objectif : Contrairement à l'exercice 1, cette fois-ci on va chercher à tracer l'évolution de la position angulaire de la roue droite θ_{61} correspondant à l'ensemble des positions que peut prendre la crémaillère $\lambda \in [233 \text{ nm}; 358 \text{ mm}]$.

Q1. Écrire en python la fonction $f(\theta_{61})$ correspondant à la fonction dont on cherche l'annulation afin de connaître la position angulaire de la roue de droite pour λ quelconque.

Q2. Pour les deux valeurs extrêmes de la crémaillère ($\lambda = 233$ nm et $\lambda = 358$ mm]) représenter la fonction f sur un intervalle pertinent.

Q3. En déduire un intervalle pertinent pour appliquer l'algorithme de dichotomie, de telle sorte que l'on puisse toujours utiliser le même intervalle quel que soit la position considérée.

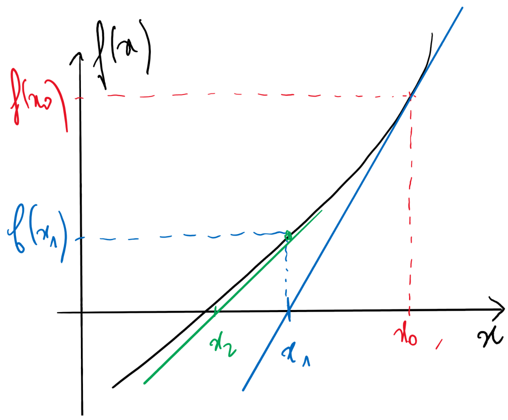
Q4. Tracer l'évolution de la position angulaire de la roue de droite en fonction de λ (pour $\lambda \in [233 \text{ nm}; 358 \text{ mm}]$). Pour cela, vous serez notamment amenés à utiliser la fonction dichotomie.

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

def f(theta61):
    return 675 - Lambda - 120 * np.cos(theta61+1.36) - np.sqrt( 356**2 - ( 148 - 120 * np.sin(theta61+1.36) )**2)

LLambda=np.linspace(233,358,1000) # tableau de 1000 valeurs réparties équitablemant entre 233 et 358
Ltheta61deg=[] # création d'une liste appelée Ltheta61

for i in range(...)): # (à compléter) : à répéter autant de fois que le nombre de valeurs que contient la liste LLambda
    Lambda=... # (à compléter)
    theta61=... # (à compléter) : faire appel à la fonction dichotomie
    theta61deg=... # (à compléter) : conversion en degré
    ... # (à compléter) : ajouter theta61deg à la liste Ltheta61deg

plt.plot(abscisse,ordonnée) # (à compléter) : représentation de theta61deg en fonction de Lambda
plt.title('titre') # (à compléter)
plt.xlabel('abscisse (unité)') # (à compléter)
plt.ylabel('ordonnée (unité)') # (à compléter)
plt.show() # pour afficher le graphique
```

II Méthode de Newton

II.1 Principe de la méthode de Newton

La méthode de Newton est une autre méthode de résolution numérique de $f(x) = 0$.



On part du point d'abscisse x_0, on trace la tangente en ce point, d'équation $y = f(x_0) + f'(x_0)(x - x_0)$, et on cherche son intersection avec l'axe des abscisses. La tangente coupe l'axe des abscisses en $x_1 = x_0 - \dfrac{f(x_0)}{f'(x_0)}$.

Puis, on trace la tangente en ce point, d'équation $y = f(x_1) + f'(x_1)(x - x_1)$, et on cherche son intersection avec l'axe des abscisses. La tangente coupe l'axe des abscisses en $x_2 = x_1 - \dfrac{f(x_1)}{f'(x_1)}$.

Et ainsi de suite...

On détermine les termes de la suite (x_n) des intersections des tangentes successives définie par récurrence :

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

On cherche ces intersections successives jusqu'à la vérification d'un certain critère, qui peut porter :

— sur les deux intersections successives x_n et x_{n+1}, qui doivent être suffisamment proches $|x_{n+1} - x_n| < \varepsilon$;

— ou sur la proximité de f à 0 à ε près.

Comment choisir l'abscisse x_0 de début de recherche ?

— Représenter la fonction f dont on cherche l'annulation ;

— Choisir x_0 proche de l'annulation de f, à distance d'un extremum local de f.

Dichotomie ou Newton ? La méthode de Newton a l'avantage de converger plus rapidement que la méthode de dichotomie, mais nécessite la connaissance de la dérivée de la fonction f, ou alors son calcul. De plus, la dérivée ne doit pas s'annuler sur l'intervalle de recherche.

II.2 Implémentation en python

On écrit la fonction `newton` qui nécessite 4 arguments en entrée :

- la fonction `f` dont on cherche l'annulation ;
- la dérivée `df` de la fonction `f` ;
- l'abscisse de début de la recherche, `x0` ;
- la précision souhaitée, `eps`, telle que $|x_{n+1} - x_n| < \varepsilon$ ou $|f(x_n)| < \varepsilon$.

```python
def newton(f,df,x0,eps):
    """
    f : fonction dont on cherche le zéro
    df : dérivée de la fonction f
    x0 : début de la recherche
    eps : précision souhaitée sur xn, tel que |f(xn)|<eps
    renvoie la valeur de xn
    """
    xn=                         # (à compléter) initialisation de la suite avec x0
                                # (à compléter) tant que |f(xn)|>eps
                                # (à compléter) calcul de x_{n+1} à partir de xn, f(xn) et df(xn)
     return xn
```

Si la dérivée de la fonction f n'est pas connue, ou difficilement calculable, on peut calculer la dérivée numériquement, en utilisant la définition du nombre dérivée en x : $f(x) = \lim\limits_{h \to 0} \dfrac{f(x+h) - f(x)}{h}$.

Au premier ordre, on peut alors écrire $f(x) \approx \dfrac{f(x+h) - f(x)}{h}$, il faudra choisir h suffisamment petit pour rendre compte des variations locales de f (mais pas trop petit...).

```python
def derivee(f,x,h):
    """
    f : fonction
    x : point auquel on calcule le nombre dérivée
    h : pas, à choisir suffisamment petit (mais >2*10*(-16))
    """
    return (f(x+h)-f(x))/h # calcul de la dérivée de f
```

II.3 Utilisation

Exercice 4 Utilisation de la méthode de Newton

Reprendre les exercices précédents en utilisant la méthode de Newton.

III Utilisation des fonctions de python

Plusieurs fonctions sont déjà définies dans Python pour résoudre numériquement des équations différentielles. On va utiliser ici les fonctions `bisect` et `newton` disponibles dans la bibliothèque `scipy.integrate`.

```
import scipy.optimize as sci # on importe scipy.optimize avec l'alias sci
>>> help(sci.bisect)
Help on function bisect in module scipy.optimize.zeros:
bisect(f, a, b) # d'autres arguments optionnels sont possibles
    Find root of a function within an interval using bisection.
    Basic bisection routine to find a zero of the function `f` between the
    arguments `a` and `b`. `f(a)` and `f(b)` cannot have the same signs.
    Slow but sure.
    Parameters
    ----------
    f : function
        Python function returning a number.  `f` must be continuous, and
        f(a) and f(b) must have opposite signs.
    a : scalar
        One end of the bracketing interval [a,b].
    b : scalar
        The other end of the bracketing interval [a,b].
    Returns
    -------
    x0 : float
        Zero of `f` between `a` and `b`.
    r : `RootResults` (present if ``full_output = True``)
        Object containing information about the convergence.  In particular,
    ``r.converged`` is True if the routine converged.

>>> help(sci.newton)
Help on function newton in module scipy.optimize.zeros:
newton(func, x0, fprime)  # d'autres arguments peuvent être placés
    Find a zero of a real or complex function using the Newton-Raphson
    Find a zero of the function `func` given a nearby starting point `x0`.
    The Newton-Raphson method is used if the derivative `fprime` of `func`
   is provided, otherwise the secant method is used.
    Parameters
    ----------
    func : callable
        The function whose zero is wanted. It must be a function of a
        single variable of the form ``f(x,a,b,c...)``, where  ``a,b,c...``
   are extra arguments that can be passed in the `args` parameter.
    x0 : float
        An initial estimate of the zero that should be somewhere near the
   actual zero.
    fprime : callable, optional
        The derivative of the function when available and convenient. If it
   is None (default), then the secant method is used.
    Returns
    -------
    root : float
        Estimated location where function is zero.
```

Exercice 5 Utilisation des fonctions

Q1. Importer la bibliothèque nécessaire, lire et comprendre les spécifications des fonctions utiles.

Q2. Reprendre les exemples précédents en utilisant les fonctions `bisect` et `newton`.

Q3. S'il vous reste du temps, modifier les paramètres optionnels possibles, notamment les tolérances (`xtot` et `rtol` pour `bisect` et `tol` pour `newton`).