

**Objectif :** Trouver une valeur approchée de la solution d'une équation (utile en particulier si on ne sait pas trouver une valeur exacte).

## I dichotomie

**Principe :** Le Théorème des Valeurs Intermédiaires (TVI) nous assure que si :

- $f$  une fonction continue sur un intervalle  $[a; b]$
- $f(a)$  et  $f(b)$  sont de signes opposés, c'est à dire  $f(a) \times f(b) < 0$

alors  $f$  s'annule au moins une fois sur  $[a; b]$ .

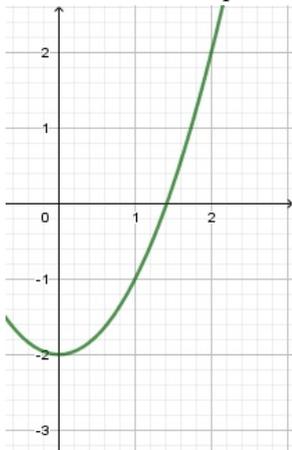
**Algorithme de dichotomie :** Tant que l'intervalle  $[a; b]$  est d'une largeur supérieure à la précision souhaitée.

1. On calcule  $m$  centre de l'intervalle  $[a; b]$
2. On évalue  $f(m)$ , valeur de  $f$  au centre de l'intervalle  $[a; b]$
3. on calcule  $f(a) \times f(m)$  et selon son signe
  - si  $f(a) \times f(m) \leq 0$ ,  $f(a)$  et  $f(m)$  n'ont pas le même signe donc, d'après le TVI,  $f$  s'annule au moins une fois sur  $[a; m]$ .  
On attribue à  $b$  la valeur de  $m$ .
  - sinon,  $f(a)$  et  $f(m)$  ont le même signe mais alors,  $f(b)$  et  $f(m)$  n'ont pas le même signe (car  $f(b)$  et  $f(a)$  sont de signe opposés). Donc, d'après le TVI,  $f$  s'annule au moins une fois sur  $[m; b]$ .  
On attribue à  $a$  la valeur de  $m$ .

*Remarque :* Si  $f$  est strictement monotone, le corollaire du TVI nous assure de l'unicité de la solution. Sinon, il peut y avoir plusieurs solutions sur l'intervalle et la dichotomie ne nous en donnera qu'une.

**Exemple d'approximation de  $\sqrt{2}$  :**  
on cherche le zéro positif de  $f(x) = x^2 - 2$

Écrivons une fonction  $dicho(f, a, b, e)$



- Preuve de la terminaison :

au bout de  $k$  itération, on a un encadrement de largeur  $\frac{b-a}{2^k}$ .

Or,  $\forall (a, b) \in \mathbb{R}^2$ ,  $\lim_{k \rightarrow +\infty} \frac{b-a}{2^k} = 0$  donc  $\forall e > 0$ , il existe un rang  $N$  à partir duquel  $\frac{b-a}{2^k} \leq e$ .

- Calcul de la complexité (estimation du nombre de tours de boucle) :

$$\frac{b-a}{2^k} \leq e \iff 2^k \geq \frac{b-a}{e} \iff k \ln 2 \geq \ln \frac{b-a}{e} \iff k \geq \frac{\ln \frac{b-a}{e}}{\ln 2}$$

$$\text{donc } N = \left\lfloor \frac{\ln \frac{b-a}{e}}{\ln 2} \right\rfloor$$

- Pour en prouver la correction, on justifie qu'on a un invariant de boucle :

$$\text{« A chaque itération, } f(g) \times f(d) \leq 0 \text{ »}$$

A l'entrée de la boucle c'est vrai.

Si  $f(g) \times f(m) \leq 0$ , on remplace  $d$  par  $m$  donc en sortie de boucle,  $f(g) \times f(d) \leq 0$

Sinon,  $f(g) \times f(m) > 0$  donc  $f(g)$  et  $f(m)$  ont le même signe, donc  $f(m) \times f(d) \leq 0$  et comme on remplace  $g$  par  $m$ , on a bien en sortie de boucle,  $f(g) \times f(d) \leq 0$

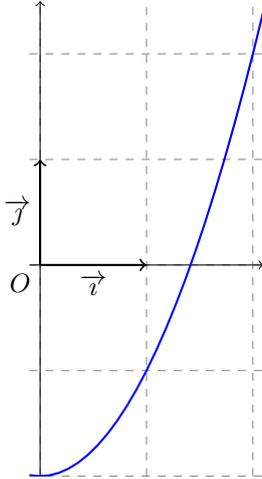
## II La méthode de Newton

### Principe :

On supposera  $f$  de classe  $\mathcal{C}^2$  dont la dérivée ne s'annule pas.

On va construire une suite récurrente  $(x_n)$  partant d'une valeur  $x_0$  donnée. En  $x_n$ , on détermine l'équation de la tangente  $T_n$  à la courbe  $\mathcal{C}_f$ . Cette tangente coupe l'axe des abscisses en  $x_{n+1}$ .

**Exemple d'approximation de  $\sqrt{2}$  :** on (re)cherche le zéro positif de  $f(x) = x^2 - 2$



Équation de  $T_n$  (tangente en  $x_n$ ) :  $y = f'(x_n)(x - x_n) + f(x_n)$

Donc  $T_n$  coupe l'axe des abscisses quand  $x - x_n = -\frac{f(x_n)}{f'(x_n)}$ .

On a donc la formule de récurrence  $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$ .

Écrivons une fonction  $Newton(f, df, a, e)$

### Difficultés :

#### Conditions et vitesse de convergence :

d'après l'inégalité de Taylor Lagrange Pour tout  $x$  et  $x_n$  appartenant à  $[a; b]$  :

$$|f(x) - (f(x_n) + f'(x_n)(x - x_n))| \leq \frac{(x - x_n)^2}{2} \sup |f''|$$

On cherche un zéro de  $f$  de  $f(x) = 0$ .

On a de plus  $-f(x_n) = (x_{n+1} - x_n)f'(x_n)$  par construction de la suite  $(x_n)$ .

$$\text{donc } |-f(x_n) - f'(x_n)(x - x_n)| = |(x_{n+1} - x_n)f'(x_n) - f'(x_n)(x - x_n)| = |f'(x_n)(x_{n+1} - x)|$$

$$\text{donc } |f'(x_n)||x_{n+1} - x| \leq \frac{(x - x_n)^2}{2} \sup |f''|$$

$$\text{donc } |x_{n+1} - x| \leq (x - x_n)^2 \frac{\sup |f''|}{2 \inf |f'|} \text{ (s'il y a convergence, on dit qu'elle est quadratique)}$$

par récurrence :

$$|x_n - x| \leq (x - x_0)^{2^n} \left( \frac{\sup |f''|}{2 \inf |f'|} \right)^{2^n - 1}$$

On est assuré de la convergence si  $|x - x_0| \frac{\sup |f''|}{2 \inf |f'|} < 1$ , autrement dit si  $x_0$  est « assez proche » de la solution recherchée.

**Dilemme robustesse/rapidité** (du cours de M Lusseau)

Nous avons vu différentes méthodes de résolution de l'équation  $f(x) = 0$ . Que choisir ?

La méthode la plus naïve est la dichotomie. A partir d'une fonction continue sur un intervalle  $[a,b]$ , avec  $f(a) \times f(b) \leq 0$ , la recherche de  $x$  tel que  $f(x) = 0$  est simple et robuste mais lente (convergence linéaire). Si l'on souhaite un programme simple et robuste, la dichotomie est donc une solution. Si on cherche une valeur avec précision, une convergence quadratique sera préférée (méthode de Newton). A ce moment-là, il convient d'avoir une idée approximative de la solution pour proposer un premier candidat au voisinage de cette solution de sorte que la fonction y présente de bonnes propriétés. Si on ne souhaite pas calculer la dérivée de la fonction, on peut alors utiliser la méthode de la fausse position. Dans le cas où l'on cherche rapidité et stabilité, on peut utiliser la méthode par dichotomie dans un premier temps pour localiser le zéro de la fonction, puis appliquer un algorithme de Newton. En cas d'instabilité de l'algorithme de Newton, il est toujours possible de réutiliser une méthode par dichotomie.

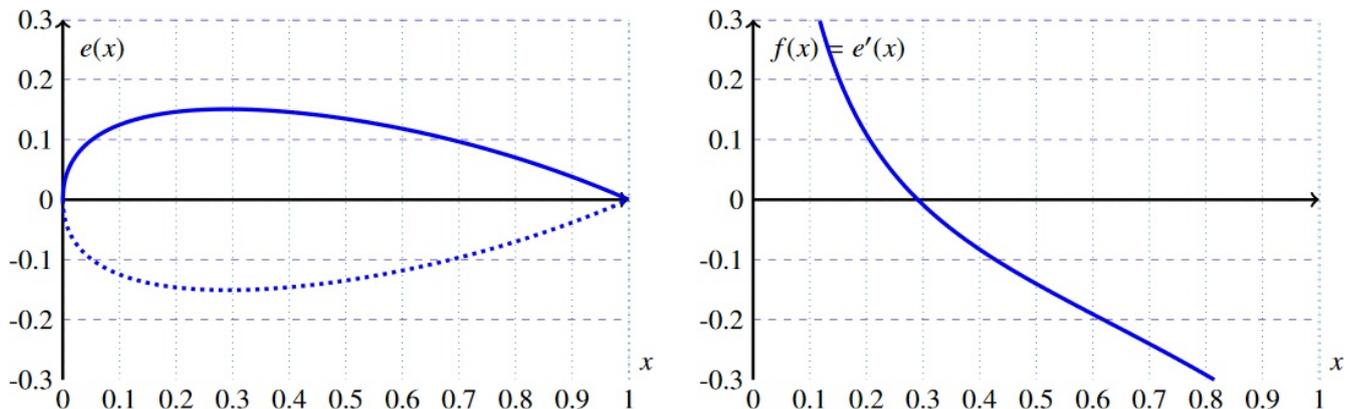
Enfin, mentionnons qu'il existe aussi d'autres types d'algorithmes, comme les algorithmes génétiques pour trouver les différentes solutions du problème.

**III Applications**

**Exo 1 :** La demi épaisseur d'une aile d'avion est approchée par la fonction  $e(x) = 0.15 \times (3.7\sqrt{x} - 3.4x - 0.3x^4)$  pour tout  $x \in [0,1]$ . Rechercher la valeur de  $x$  pour laquelle le profil est le plus épais revient à résoudre l'équation

$$0.15\left(\frac{3.7}{2\sqrt{x}} - 3.4 - 1.2x^3\right) = 0$$

Déterminer par la méthode de votre choix (ou les deux si vous avez le temps), la valeur de  $x$  pour laquelle le profil d'aile d'avion est le plus épais.



**Exo 2 :** Filtrage de l'accéléromètre du drone

**Partie A : Identification et validation du modèle de la MCC**

Un programme informatique est réalisé afin d'identifier rigoureusement la fonction de transfert  $\frac{\Omega_m(p)}{U_n(p)}$ . Il exploite les données expérimentales stockées dans le fichier de mesures « results.csv » issues d'un essai.

- 1) Aller sur le site récupérer les fichiers (en prenant soin de savoir où vous enregistrez vos fichiers de mesures) puis ouvrir le fichier « Identification.py ».

Dans un premier temps l'identification de la courbe obtenue à la réponse indicielle d'un système du premier ordre est testée. On notera  $\tau$  la constante de temps et  $K$  le gain statique de ce système, de sorte que l'expression analytique de la vitesse de rotation soit :

$$\omega_m(t) = KU_0(1 - e^{-t/\tau})$$

avec Avec  $U_0$  la tension consigne (avec  $U_0 = 10V$ ).

Afin d'identifier la courbe, on se donne la fonction  $err(tau)$  permettant d'estimer l'erreur entre la mesure  $L\_omega_i$  et la fonction analytique  $\omega_m(t)$  :

$$err(tau) = \sum_i (\omega_m(t_i) - L\_omega_i)^2$$

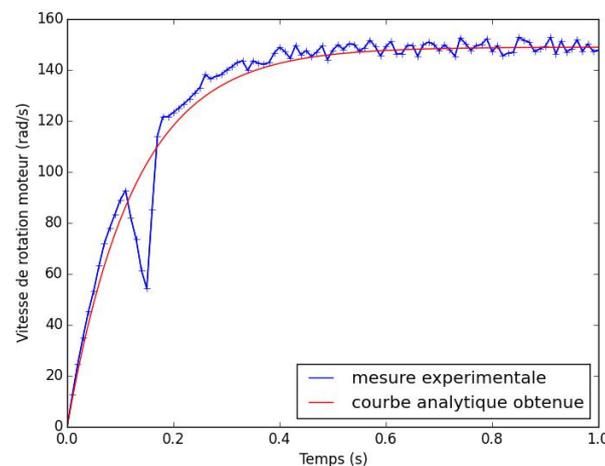
critère d'erreur au sens des moindres carrés.

où  $\omega_m(t_i)$  est la valeur de la fonction analytique de la vitesse de rotation à l'instant  $t_i$  et  $L\_omega_i$  est la  $i$ ème valeur de vitesse de rotation mesurée à l'instant  $t_i$ .

La trame du programme proposé (Identification) contient :

- les instructions permettant d'extraire les données du fichier de mesures, où deux tableaux de mêmes tailles sont créés, l'un contenant les instants de prise de mesure `L_temps` et l'autre les valeurs mesurées de la vitesse de rotation  $\omega_m(t)$  `L_omega` exprimée en deg/s. Le  $i$ ème élément de `L_omega` est la valeur de vitesse de rotation mesurée à l'instant correspondant à la  $i$ ème valeur de `L_temps`.
  - la fonction `omega_anal(t,tau)` retournant la valeur analytique de  $\omega_m(t)$ .
  - la fonction (à compléter ultérieurement) `Erreur(tau, L_temps, L_omega)` prenant en argument la valeur de  $tau$  et les résultats de la mesure, et qui renvoie  $err(tau)$ .
  - la fonction `Erreur_der(tau, L_temps, L_exp)` retournant une estimation de la dérivée de l'erreur  $\frac{derr}{dtau}(tau)$  en  $tau$  (à condition que `Erreur()` soit bien définie).
  - la fonction `Erreur_der2(tau, L_temps, L_exp)` retournant une estimation de la dérivée seconde de l'erreur  $\frac{d^2err}{dtau^2}(tau)$  en  $tau$  (à condition que `Erreur()` soit bien définie).
- 2) Lire puis exécuter le script après avoir mis à jour l'adresse de votre fichier de mesures. Modifier le script afin que les valeurs de convertir en radian par seconde les valeurs de `L_omega`.
  - 3) A l'aide des données extraites du fichier de mesure `L_temps` et `L_omega`, écrire les instructions permettant d'afficher la courbe représentant l'évolution de la mesure de la vitesse de rotation du moteur  $\omega_m(t)$  (en rad/s) en fonction du temps. En déduire une estimation de la valeur numérique de  $K$ , que l'on utilisera pour la suite, et la renseigner dans la fonction `omega_anal(t,tau)`.
  - 4) Compléter la fonction `Erreur(tau, L_temps, L_omega)` prenant en argument la valeur de  $tau$  et les résultats des mesures, et qui renvoie  $err(tau)$ . Tracer l'évolution de  $err(tau)$  en fonction de  $tau$  pour  $tau$  allant de 10ms à 1s, par incréments de 10ms. Commenter le résultat obtenu.
  - 5) En utilisant un algorithme de dichotomie, écrire la fonction `minimum(L_temps, L_omega)` prenant en argument les listes de mesures, et qui renvoie la valeur de  $tau$  pour laquelle l'erreur est minimale. On recherche  $tau$  à 0.1ms près.

Le programme précédent aboutit à une valeur  $tau=126.9ms$ . On peut tracer sur un même graphique, la courbe analytique ainsi que les résultats expérimentaux, voir figure ci-après.



Des procédures similaires réalisées en assimilant le système à un système d'ordres supérieurs aboutissent à des valeurs d'erreur très supérieures à celle obtenue pour le cas précédent.

- 6) Donner alors, avec les valeurs numériques, la fonction de transfert  $\frac{\Omega_m(p)}{U_n(p)}$

## Partie B : Filtrage accéléromètre

La figure ci-contre est issue du diagramme des exigences du drone. Pour une exploitation satisfaisante de l'accéléromètre, les utilisateurs considèrent que les vibrations de fréquence supérieures à 1 Hz environ ne doivent pas être ressenties par l'accéléromètre.

«usabilityRequirement»
<b>filtrer le signal de l'accéléromètre</b>
Id = "10.2.1.1"
Text = "la bande passante à -3dB de l'accéléromètre doit être limitée à 1 Hz"

On se propose de filtrer les mesures fournies par l'accéléromètre par un filtre de fonction de transfert  $H(p) = \frac{K}{1 + \tau p}$

- 1) Justifier la méthode proposée, par un diagramme de Bode par exemple, et proposer des valeurs pour  $K$  et  $\tau$  permettant de satisfaire cette exigence.

Le fichier « results\_angle.csv » contient différentes mesures d'angle :

- Colonne 1 : variable `temps`.
- Colonne 2 : variable `Angle_AccX`, mesure brute d'angle fourni par l'accéléromètre.
- Colonne 3 : variable `Angle_FC`, mesure d'angle issue du filtrage complémentaire
- Colonne 4 : variable `Angle_Gyro_FC`, mesure d'angle issue du gyromètre et filtrée
- Colonne 5 : variable `Angle_Acc_FC`, mesure d'angle issue de l'accéléromètre et filtrée
- Colonne 6 : variable `Angle_Pivot`, mesure d'angle issue du capteur de position au niveau de la liaison pivot.
- Colonne 7 : variable `Angle_Gyro`, mesure d'angle brute fournie par le gyromètre.

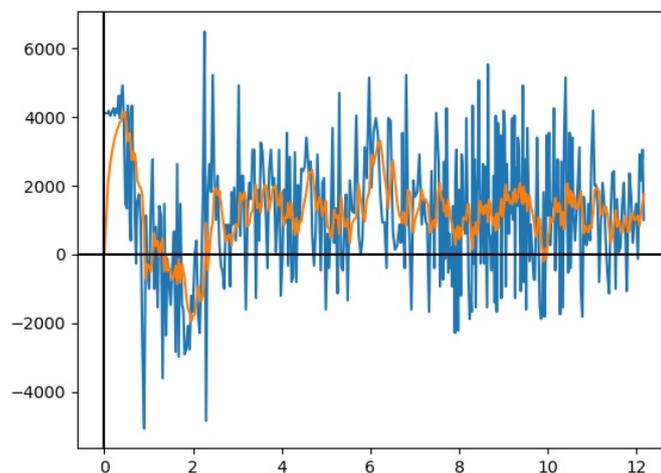
Les procédures correspondantes à la lecture du fichier sont fournies dans le script « Filtrage.py ».

- 2) A l'aide de Python, programmer le filtrage proposé et afficher sur le même graphique l'angle pivot, la mesure brute issue de l'accéléromètre, et le résultat du filtrage. Pour ce faire :

- (a) Discrétiser l'équation de filtrage :

$$s(t) + \tau \frac{ds(t)}{dt} = Ke(t)$$

- (b) Ouvrir le fichier « Filtrage.py ».
- (c) Ce fichier est à compléter : la récupération des données est déjà programmée, il reste à implémenter le filtrage.
- (d) Tester le filtrage pour différentes valeurs de  $\tau$ . Indiquer le compromis à réaliser pour choisir effectivement la valeur de  $\tau$ .



La mesure `Angle_ACC_FC` correspond à la mesure filtrée effectivement réalisée au niveau du logiciel (il s'agit d'un filtrage d'ordre plus élevé que celui testé).

- 3) Tracer cette mesure à l'aide de Python.  
Indiquer en quoi ce traitement seul de la mesure fournie par l'accéléromètre est insuffisant pour réaliser l'asservissement du drone.