Méthodes numériques (révisions MPSI)

I. Recherche des zéros d'une application à valeurs réelles

Soit $f : I = [a, b] \longrightarrow \mathbb{R}$ une applications réelle ($a > b$), continue sur I et admettant au moins une valeur $c \in]a, b[$ telle que $f(c) = 0$. Quitte à choisir correctement a et b, nous allons supposer qu'il existe **un et un seul réel** $c \in]a, b[$ tel que $f(c) = 0$: le réel c est un zéro de l'application f.

Deux algorithmes aux programme d'IPT permettent de trouver c : l'algorithme de recherche par *dichotomie* et l'algorithme de *Newton*.

1) Recherche dichotomique

Applications

1. Écrire une fonction Python qui traduit l'application mathématique $f : x \longmapsto x^2 - 2$ de $\mathbb{R}$ dans $\mathbb{R}$. Appliquer la méthode par dichotomie pour en trouver les zéros à 10^{-12} près.
2. Trouver une valeur approchée de π en appliquant cette méthode à l'application $x \longmapsto \sin(x)$. On importera le module `math`.

2) Méthode de Newton

On suppose que l'application f est de classe C^1 sur un intervalle $I = [a, b]$ de $\mathbb{R}$ ($b > a$) et on se place dans le cas où il existe un unique réel $c \in]a, b[$ tel que :

$$f(c) = 0 \quad \text{et} \quad f'(c) \neq 0$$

Quitte à restreindre l'intervalle I, nous allons supposer que la dérivée f' garde un signe constant sur I.

Applications

Tester la fonction sur les deux exemples de la section précédente.

II. Intégration d'une application

Soit $f : I = [a, b] \longrightarrow \mathbb{R}$ une application à valeurs réelles, continue sur l'intervalle fermé borné $I = [a, b]$ ($b > a$). On sait que f est alors intégrable sur I. On expose dans cette section deux méthodes au programme d'IPT qui permettent de calculer :

$$J = \int_a^b f(x)\,\mathrm{d}x$$

de façon approchée : la **méthode d'Euler** et la **méthode des trapèzes**.

1) Méthode d'Euler

2) Méthode des trapèzes

III. Systèmes différentiels et méthode d'Euler

On cherche la solution approchée d'un système différentiel de la forme :

$$\left\{\begin{array}{lcl} y'(t) & = & f(t, y(t)) \\ y(t_0) & = & y_0 \end{array}\right.$$

où y' désigne la dérivée d'une application y. Il s'agit d'un système différentiel du premier ordre puisque seule la dérivée première y' intervient.

Exemple :

1) Méthode d'Euler 1D

Nous cherchons une solution $y : t \longrightarrow y(t)$ définie sur un intervalle $I = [t_0, t_f]$ $(t_f > t_0)$, de sorte que :

$$\forall\, t \in I,\ y'(t) = f(t, y(t)) \quad \text{et} \quad y(t_0) = y_0 \ \text{(condition initiale)}$$

où y_0 est donné.

Tout comme dans la section "Intégration", on considère une subdivision régulière $(t_i)_{0 \leqslant i \leqslant n}$ de l'intervalle I de sorte que :

$$t_i = t_0 + ih$$

où h est le **pas** de cette subdivision.

Il y a deux méthodes d'Euler : la méthode *explicite* et la méthode *implicite*.

a) Méthode d'Euler explicite

Mise en oeuvre :

b) Exercices

1. Tester la fonction `euler` sur le système suivant :

$$y'(t) = -\,0.5\,y(t) \quad \text{avec} \quad y(0) = 1$$

On définira l'application $f : (t,x) \longmapsto f(t,x)$ associée à cette équation différentielle et on l'écrira comme une fonction python dans le fichier.

Vous pourrez utiliser le module **matplotlib.pyplot** pour tracer la courbe associée à $t \longmapsto y(t)$. Tracer ensuite la courbe de la solution exacte $t \longmapsto y_{\text{exacte}}(t)$: une superposition des deux courbes permet de bien étudier leurs différences. Étudier ce qui se passe avec un pas h de plus en plus petit.

2. Application à la chute d'un objet dans l'atmosphère (problème du parachutiste). Un objet de masse m tombe selon la verticale Oz dirigée vers le bas, en étant soumis à son poids et à la force de frottement fluide exercée par l'atmosphère, de la forme $\overrightarrow{F} = -\,\alpha\, v^2\, \overrightarrow{u_z}$ ($\alpha > 0$).

Le principe fondamental de la dynamique conduit à l'équation différentielle :

$$\frac{\mathrm{d}v}{\mathrm{d}t} = -\,\frac{\alpha}{m}\,v^2 + g \quad \text{avec} \quad v(0) = 0$$

On prendra pour valeurs numériques : $\alpha = 0.29$ uSI, $m = 70$ kg, $g = 9.8$ m.s^{-2}.

- Résoudre cette équation différentielle avec votre fonction `euler` et représenter la courbe associée à $t \longmapsto v(t)$ en prenant différentes valeurs de h
- Comparer avec la solution exacte :

$$v(t) = \sqrt{\frac{mg}{\alpha}}\,\tanh\left(\sqrt{\frac{\alpha g}{m}}\,t\right)$$

où tanh est la tangente hyperbolique (le module **math** contient toutes les fonctions nécessaires).

Remarque :

Pour éviter tout conflit d'identifiant avec la variable m qui représente la masse, si vous importez le module math avec un alias il sera judicieux d'écrire `import math as mp` et non `import math as m`

2) Méthode d'Euler 2D

Nous allons développer l'exemple suivant :

$$y''(t) + y'(t)\sin[\,y(t)\,] = t$$

où $y : t \longrightarrow y(t)$ est une application. Nous recherchons donc la solution de cette équation différentielle vérifiant les conditions initiales $y(t_0) = y_0$ et $y'(t_0) = v_0$ données.

Par la suite nous allons poser $v = y'$ (application vitesse). Nous cherchons une solution approchée sur l'intervalle $[t_0, t_f]$ ($t_f > t_0$).

a) Méthode d'Euler explicite 2D

Sur feuille

b) Exercices

1. L'équation différentielle du pendule simple dans le champ de pesanteur, sans approximation des petits mouvements, s'écrit :

$$y''(t) + \omega^2\,\sin(\,y(t)\,) = 0$$

avec $\omega^2 = g/L$. On prendra comme valeurs numériques $g = 10$ m.s^{-2} et $L = 1$ m.

Écrire la fonction Python `F(t,X)` associée à cette équation différentielle.

2. Tester la fonction `euler2D` sur cette équation différentielle et tracer les courbes des applications $t \longmapsto y(t)$ et $t \longmapsto v(t)$.
3. Tracer le portrait de phase $y \longmapsto v$.
4. Le système proies - prédateurs.

   La méthode précédente peut être appliquée à des équations différentielles d'un autre type que les équations différentielles d'ordre 2.

   Considérons un système proies - prédateurs et notons $x(t)$ la population des proies à l'instant t (par exemple des lièvres) et $y(t)$ la population des prédateurs à t (par exemple des lynx) On suppose que l'évolution est décrit par un système de deux équations différentielles couplées, appelé équations de Lotka - Volterra :

$$\left\{\begin{array}{lcl} x'(t) & = & a\,x(t) - b\,x(t)y(t) \\ y'(t) & = & -c\,y(t) + d\,x(t)y(t) \end{array}\right.$$

   où a, b, c et d sont des constantes positives. On pourra prendre par exemple : $a = 2.5$; $b = 0.05$; $c = 0.8$ et $d = 0.05$.

   Transformer le système couplé précédent en une équation différentielle d'ordre 1 de la forme :

$$Y'(t) = F(\,t, Y(t)\,)$$

   en explicitant l'application F de $\mathbb{R}^3$ dans $\mathbb{R}^2$.

   L'écrire sous la forme d'une fonction python `Fpp(t,X)` et tracer les courbes $t \longmapsto x(t)$ et $t \longmapsto y(t)$ ainsi que la courbe $x \longmapsto y$.