Université Paris-Sud **Introduction au calcul scientifique**
Licence de Mathématiques **S4 UE Math209 - Année 2016-2017**

Travaux Pratiques en Python 4

Exercice 1 : Modèle de croissance logistique
Modélisation : Nous cherchons à modéliser l'évolution de la taille d'une population de bactéries au cours du temps. A première vue, la variation de la population dépend du taux de reproduction de cette population (c'est-à-dire de la différence entre le taux de naissance et le taux de mortalité). Pour un modèle plus réaliste, nous considérons aussi que le milieu ne peut accueillir qu'un nombre maximal d'individus. Mathématiquement, cela se traduit par l'équation suivante appelée «modèle de croissance logistique» :

$$\begin{cases} \frac{\mathrm{d}}{\mathrm{dt}}X(t) = rX(t)\left(1 - \frac{X(t)}{K}\right), \\ X(0) = 3, \end{cases} \tag{1}$$

pour laquelle

- $X(t)$ représente la taille de la population à l'instant t,
- $\frac{\mathrm{d}}{\mathrm{dt}}X(t)$ représente la variation de la population au cours du temps,
- r est le *taux de reproduction* de la population,
- K représente la *capacité biotique du milieu*, c'est-à-dire la taille limite de la population que le milieu peut supporter.

Remarque 1 : Si $r > 0$, le taux de naissance est plus important que le taux de mort, la population va croître jusqu'à atteindre sa taille limite K. Si $r < 0$, le taux de mortalité est cette fois-ci plus important et la population va disparaître.

Résolution en python : Pour résoudre à l'aide de python le modèle (1), on choisit de calculer la solution en certains temps t_n seulement et on remplace la dérivée $\frac{\mathrm{d}}{\mathrm{dt}}X(t)$ par un taux de variation. En notant $X_n = X(t_n)$, on a deux choix possibles :

$$\frac{X_{n+1} - X_n}{t_{n+1} - t_n} = rX_n\left(1 - \frac{X_n}{K}\right) \tag{2}$$

ou

$$\frac{X_{n+1} - X_n}{t_{n+1} - t_n} = rX_{n+1}\left(1 - \frac{X_{n+1}}{K}\right). \tag{3}$$

Pour simplifier, on prendra par la suite $t_{n+1} - t_n = 1$, les équations (2) et (3) se réécrivent alors

$$X_{n+1} - X_n = rX_n\left(1 - \frac{X_n}{K}\right) \tag{4}$$

et

$$X_{n+1} - X_n = rX_{n+1}\left(1 - \frac{X_{n+1}}{K}\right). \tag{5}$$

1. **Méthode du point fixe sur l'équation** (4)
   (a) Écrire l'équation (4) sous la forme $F(X_n) = X_{n+1}$.
   On suppose $K = 10$ **et** $r = 0.7$
   (b) Définir cette fonction en python et la tracer sur $[0, 20]$. N'oublier pas un titre et une légende à votre graphique. Quel semble être le point fixe de F ? On pourra tracer au besoin la droite $x \mapsto x$.

(c) Créer une suite (X_n) définie par récurrence par $F(X_n) = X_{n+1}$ et représenter les 20 premières itérations de la suite. (Nous rappelons que $X_0 = 3$).

(d) Assiste-t-on à la disparition de l'espèce ou à sa croissance ? Selon la remarque 1, vers quelle valeur la population semble-t-elle converger ?

(e) Nous voulons savoir ensuite la dynamique de la suite (X_n), c'est-à-dire si elle est monotone ou alternée. Sur une figure, représenter, avec des couleurs différentes, la courbe d'équation $x \mapsto x$, la courbe de F ainsi que la ligne brisée reliant les points $(X_0, X_0), (X_0, X_1), (X_1, X_1), (X_1, X_2)$...

(f) La suite est-elle alternée ou monotone ?

(g) Tracer en échelle logarithmique la droite reliant les points $(X_n - K)$ en abscisse et $(X_{n+1} - K)$ en ordonnée. En déduire l'ordre de convergence de la suite. Pour cela, on pourra tracer en échelle logarithmique les droites d'équation $x \mapsto x$ et $x \mapsto x^2$.

On suppose $K = 10$ et $r = 2$

(h) Reprendre les questions avec cette fois $K = 10$ et $r = 2$.

2. **Méthode de Newton sur l'équation** (5)

(a) Réécrire l'équation (5) comme $G(X_{n+1}) = 0$. La fonction G pourra dépendre de X_n.

(b) Définir cette fonction en python. Définir aussi sa dérivée que l'on appellera `dG`.

(c) À l'aide de l'algorithme de Newton, on veut calculer X_1 en partant de X_0. On va donc créer une suite de points (x_k) définie par

$$\begin{cases} x_{k+1} = x_k - \dfrac{G(x_k)}{G'(x_k)} \\ x_0 = X_0. \end{cases}$$

Implémenter cette suite (x_k) dans python.

(d) En combien d'itérations la suite semble-t-elle converger ? Que vaut X_1 dans ce cas ?

Exercice 2 : Rappels sur l'interpolation de Lagrange

Nous rappelons ici les définitions des polynômes d'interpolation de Lagrange $P(X)$. Pour interpoler une fonction $f(x)$, on se donne n points $\{x_i\}_{i=1,\dots,n}$. On définit $P(X)$ comme:

$$P(X) = \sum_{i=1}^{n} f(x_i) L_i(X)$$

avec

$$L_j(X) = \prod_{i=1, i\neq j}^{n} \frac{X - x_i}{x_j - x_i}.$$

1. Pour interpoler $f(x) = x^3$ par des polynômes de Lagrange dans l'intervalle $[0.5, 3.5]$, on choisit trois points $\{x_i\}_{i=1,2,3}$: 1, 2 et 3. Calculer d'abord les polynômes élémentaires de Lagrange $L_1(x)$, $L_2(x)$, et $L_3(x)$.

2. Puis donner et tracer le polynôme d'interpolation de Lagrange $P(X)$. Vérifier que le graphe du polynôme passe par ces trois points $(x_i, f(x_i))$.

Exercice 3 : Calcul du polynôme d'interpolation de Lagrange

Dans cet exercice, nous allons calculer le polynôme d'interpolation de Lagrange (P.I.L) par deux méthodes vues en classe : par une matrice de Vandermonde et par la méthode des différences divisées. Nous prendrons dans toute la suite,

```
xi=np.array([1., 4., -2., 9.5, 13.]),
yi=np.array([[3.], [6.], [9.], [-12.], [-1.32]]).
```

1. Supposons que le P.I.L s'écrive $P(X) = a_0 + a_1 X + a_2 X^2 + ... + a_{n-1} X^{n-1}$ et que son graphe doive passer par les points (x_i, y_i), alors les coefficients a_i sont solutions du système matriciel

$$\underbrace{\begin{pmatrix} 1 & x_1 & x_1^2 & ... & x_1^{n-1} \\ 1 & x_2 & x_2^2 & ... & x_2^{n-1} \\ \vdots & \vdots & \vdots & ... & \vdots \\ 1 & x_n & x_n^2 & ... & x_n^{n-1} \end{pmatrix}}_{A} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_{n-1} \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}.$$

   A l'aide de la commande Python `np.linalg.solve(A,yi)`, où A est la matrice de Vandermonde, déterminer les coefficients a_i du polynôme d'interpolation de Lagrange.

2. Tracer ce polynôme sur $[-15, 15]$ et comparer avec le résultat fourni par la fonction `inter.barycentric_interpolate(xi,yi,x)` de `scipy.interpolate`.

3. Une deuxième manière d'évaluer le polynôme d'interpolation de Lagrange est d'utiliser la méthode des différences divisées. On définit le P.I.L par

   $$P_n(x) = \delta[x_1] + \delta[x_1, x_2](x - x_1) + \delta[x_1, x_2, x_3](x - x_1)(x - x_2) + ... + \delta[x_1, ...x_n](x - x_1)...(x - x_{n-1}),$$

   avec

   $$\delta[x_m, ...x_{k+1}] = \frac{\delta[x_{m+1}, ...x_{k+1}] - \delta[x_m, ...x_k]}{x_{k+1} - x_m}.$$

   Il ne reste plus qu'à évaluer $P(x)$ avec la méthode de Horner. Programmer deux fonctions Python `differences_divisees(xi,yi)` et `horner(xi,yi,x)` qui fourniront une matrice contenant les différents δ pour la première fonction et qui évaluera le polynôme en les différents points du vecteur x pour la deuxième.

4. Tester ces deux fonctions en affichant sur une même figure le polynôme obtenu par différences divisées et le polynôme obtenu par la fonction `inter.barycentric_interpolate(xi,yi,x)` de `scipy.interpolate`.

Exercice 4 : Phénomène de Runge

Nous allons mettre en évidence le phénomène de Runge sur la fonction

$$f(x) : x \in [a, b] = [-1, 1] \mapsto \frac{1}{4x^2 + 1}.$$

C'est-à-dire le fait que la suite de polynômes d'interpolation de Lagrange $(P_n)_n$ ne converge pas uniformément vers f sur $[a, b]$ lorsque les points $x_1^{(n)}, ..., x_n^{(n)}$ sont uniformément répartis dans $[a, b]$ (voir le paragraphe sur le comportement de P lorsque n tend vers l'infini, page 2 de la fiche numéro 6 du cours).

1. Tracer dans une fenêtre Python le graphe de f sur $[-1, 1]$.

2. Le polynôme d'interpolation de Lagrange peut se calculer de la manière suivante

```
import scipy.interpolate as inter
y=inter.barycentric_interpolate(xi,yi,x)
```

   où `xi` et `yi` représentent les points d'interpolation (x_i, y_i) par lesquels le graphe du polynôme doit passer et `x` les points en lesquels on veut connaître la valeur du polynôme d'interpolation. En prenant pour `xi`, 11 points régulièrement espacés dans $[-1, 1]$ et pour `yi` les images $f(x_i)$, tracer dans une fenêtre Python le polynôme d'interpolation de Lagrange en bleu.

3. Vérifier par des marqueurs aux points (x_i, y_i) que le graphe de ce polynôme passe bien par ces points d'interpolation.

4. Sans rien changer d'autre à votre code, augmenter le nombre de points `xi` à 101 points régulièrement espacés dans $[-1, 1]$ (et calculer les `yi` correspondant).

5. Qu'observez-vous ? Le graphe du polynôme d'interpolation passe-t-il toujours par les couples de points (x_i, y_i) ? Le polynôme approche-t-il uniformément f sur $[-1, 1]$?

Exercice 5 : Points d'interpolation de Tchebychev

Nous allons dans cet exercice montrer que le phénomène de Runge ne se produit plus si les points d'interpolation sont pris de manière particulière (et non uniformément répartis dans $[a, b]$). En effet, si $x_j^{(n)} = \cos(\frac{(2j-1)\pi}{2n})$ et si f est lipschitzienne alors le polynôme d'interpolation de Lagrange approche uniformément f sur $[a, b]$.

1. En reprenant votre code de l'exercice 4, modifier les points d'interpolations `xi` en `xi`$= x_j^{(n)} = \cos(\frac{(2j-1)\pi}{2n})$ pour $j \in \{1, .., n\}$ et pour $n = 101$.

2. Tracer le polynôme d'interpolation de Lagrange avec ces nouveaux points d'interpolation.

3. Qu'observez-vous ? Y a-t-il encore le phénomène de Runge ?