Last but gnagna...

Vendredi 18 mars 2016

Buts du TP

– Se mettre au point avant les écrits.
– Et puis c'est tout.

Exercice 1. *Créer (au bon endroit) un dossier associé à ce TP. Dans ce dossier, placer les fichiers suivants, récupérés sur le réseau :*

`notes_colle.db` `squelette.py`

Le fichier `squelette.py` contient ce qu'il faut pour que vous puissiez avancer vite dans le TP, sans vous préoccuper des tâches annexes et chronophages.

Exercice 2. *Lancer Spyder (ou Pyzo), sauvegarder immédiatement le fichier édité au bon endroit sous un nom pertinent. Écrire une commande absurde, de type* `print(6*7)` *dans l'éditeur ; sauvegarder.*
Ceux qui travaillent sous Spyder pourront, avec F6, imposer les directives « exécuter dans un nouvel interpréteur dédié » et « interagir avec l'interpréteur Python après exécution ».
Exécuter le fichier de script. Ouvrir également le fichier `squelette.py` *et copier/coller (au fur et à mesure du TP) le contenu.*

1 Des requêtes SQL (20 minutes)

On travaille ici avec la base `notes_colle.db`

Ouvrir cette base. Écrire (papier, crayon...) son schéma relationnel. Comprendre le lien entre les différents attributs des différentes tables.

Exercice 3. C'est parti

1. *Déterminer la liste des noms et prénoms des professeurs... et ceux des élèves.*
2. *Déterminer le nombre de 20 qui ont été attribués, ainsi que le nombre de notes majorées par* 6.

Exercice 4. Avec jointures

1. *Déterminer les notes de Jacques-Louis Lions (triées selon les semaines croissantes).*
2. *Refaire la même chose, avec cette fois le nom des colleurs.*

Exercice 5. Des agrégats

1. *Déterminer la moyenne des notes de colle de Jacques-Louis Lions.*
2. *Déterminer la liste des couples (élève, moyenne).*
3. *Parmi tous les élèves, déterminer le nom de ceux ayant eu au moins* 10 *notes strictement sous la moyenne.*

2 Un tri (15 minutes)

Exercice 6. *Écrire* rapidement *une fonction réalisant le tri en place d'un tableau. Donner sa complexité en termes de nombre de comparaisons d'éléments du tableau.*

Je suggère le tri-bulle : pour j descendant de $n-1$ à 1, on effectue un balayage permettant de remonter une bulle : pour i allant de 0 à $j-1$, on échange les éléments en positions i et $i+1$ si nécessaire.
Le squelette fourni contient un test pour votre fonction de tri.

3 Informatique du collège/des années 80 (20 minutes)

On (re)découvre les fonctions `input` et `print`... qu'aiment tant certains auteurs de sujets de concours !

Exercice 7. *Écrire un programme (passionnant) demandant d'entrer deux entiers/réels au clavier, calculant et affichant le produit (mais ne retournant rien).*

Dans l'exemple suivant, après l'appel du programme (dans l'interpréteur), on a tapé 10 puis 27 au clavier.

```
>>> exo_stupide()
Entrer a : 10
Entrer b : 27
Le produit a.b vaut 10 x 27 = 270
```

Un peu mieux ?

Exercice 8. *Écrire un programme tirant au sort un entier entre* 1 *et* 1000*, puis le faisant chercher à l'utilisateur... comme dans l'exemple qui suit !*

```
>>> moins_stupide()
Proposition numéro 1 ? 500
Plus grand !
Proposition numéro 2 ? 750
Plus petit !
Proposition numéro 3 ? 625
Plus grand !
Proposition numéro 4 ? 680
Plus petit !
Proposition numéro 5 ? 650
Plus grand !
Proposition numéro 6 ? 665
Plus grand !
Proposition numéro 7 ? 672
Plus petit !
Proposition numéro 8 ? 669
Plus petit !
Proposition numéro 9 ? 667
Bravo !
```

4 L'inévitable Euler (25 minutes)

Exercice 9. *Compléter le programme* `Euler` *du squelette pour que l'appel* `Euler(f, y0, a, b, nbpts)` *calcule puis retourne une évaluation en b de l'unique solution au problème de Cauchy*

$$\begin{cases} y'(t) = f(t, y(t)) \\ y(a) = y_0 \end{cases}$$

On mettra évidemment en œuvre la méthode d'Euler avec `nbpts` *pas !*

Testons votre fonction sur un exemple où la solution est explicitement connue :

$$\begin{cases} y'(t) = y(t) + \cos(t) \\ y(0) = 1 \end{cases}$$

La solution est $z : t \mapsto -\frac{1}{2}\cos t + \frac{1}{2}\sin t + \frac{3}{2}\mathrm{e}^t$, qui vérifie $z(\pi) = \frac{1}{2} + \frac{3}{2}\mathrm{e}^{\pi} \simeq 35,211$.

Exercice 10. *Tester la fonction* `Euler` *sur l'exemple précédent.*
Vous n'avez qu'à définir correctement la fonction f_0 : le reste est dans le squelette, avec en particulier les différentes valeurs du nombre de pas.

Pour des équations différentielles d'ordre 2

$$y'' = f\left(t, y(t), y'(t)\right)$$

il y a deux écoles :

– Ceux qui ont compris la vectorialisation, et qui appliquent donc la méthode *générique* au système différentiel d'ordre 1 dans $\mathbb{R}^2$. Ceux-là n'ont pas à réécrire une nouvelle fonction `Euler` : ils transforment simplement l'équation d'ordre 2 en une équation d'ordre 1, et vérifient/modifient leur fonction `Euler` pour qu'elle fonctionne sur des vecteurs (`array`) aussi bien que sur des réels (ce qui ne doit pas poser de problème).
– Ceux qui n'ont pas compris la vectorialisation réinventent la roue, avec une méthode différente pour chaque nouveau problème ! Ici, ils vont retrouver le même résultat que le Euler générique avec une méthode d'Euler consistant à calculer des approximations des $y(t_k)$ et $y'(t_k)$, essentiellement via $y(t_{k+1}) \simeq y(t_k) + hy'(t_k)$ et

$$y'(t_{k+1}) \simeq y'(t_k) + hy''(t_k) = y'(t_k) + hf\left(t_k, y(t_k), y'(t_k)\right)$$

Pour les deux exercices suivant, on s'intéresse au problème de Cauchy/du pendule

$$\begin{cases} y''(t) = -\sin\left(y(t)\right) \\ y(0) = 0 \\ y'(0) = 1 \end{cases}$$

dont l'unique solution vérifie $z(10) \simeq 0,1143$ et $z'(10) = -0,9935$.

Exercice 11. *Vectorialiser ce problème en définissant une fonction F_1 de $\mathbb{R} \times \mathbb{R}^2$ dans $\mathbb{R}^2$ de sorte que cette équation différentielle devienne $Y'(t) = F_1(t, Y(t))$. Exécuter alors la fonction* `Euler` *avec les bonnes conditions initiales.*

Bon, on va faire comme si on ne connaissait/comprenait pas la vectorialisation [1] !

Exercice 12. *Écrire une fonction* `Euler2` *mettant en œuvre la méthode vue plus haut en calculant à la main les approximations successives des $y(t_k)$ et $y'(t_k)$. Tester sur le problème de Cauchy vu plus haut.*

5 Du `numpy` (15 minutes)

Pour créer une matrice d'un format donné, on peut commencer par en créer une nulle : `numpy.zeros( (n,p) )` Ensuite, on peut la remplir via une boucle !

Exercice 13. *Créer le vecteur* $(1, 2, ..., 10)$*, la matrice*

$$H = \begin{pmatrix} 1 & 1/2 & \cdots & 1/5 \\ 1/2 & 1/3 & \cdots & 1/6 \\ \vdots & & & \vdots \\ 1/5 & \cdots & \cdots & 1/9 \end{pmatrix} \in \mathcal{M}_5(\mathbb{R})$$

de terme général $H_{i,j} = \dfrac{1}{i+j-1}$ (avec les notations du matheux ; attention : `numpy` *indexe ses matrices depuis* $(0,0)$ *!)*
Que vaut le déterminant de H ?

On s'intéresse maintenant à l'équation $AX = Y$ d'inconnue $X \in \mathcal{M}_{2,1}(\mathbb{R})$, avec

$$A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \\ \vdots & \\ 19 & 20 \end{pmatrix} \in \mathcal{M}_{10,2}(\mathbb{R}) \qquad \text{et} \qquad Y = \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \in \mathcal{M}_{10,1}(\mathbb{R})$$

Malheureusement, Y n'est pas dans l'image de A, donc il n'existe pas de solution X. À défaut, on va chercher X tel que $\|AX - Y\|$ soit minimale (il s'agit de la norme euclidienne usuelle) ; c'est la méthode dite *des moindres carrés* (pourquoi à votre avis ?).

Exercice 14. *En notant/admettant que ${}^tAA$ est inversible, rappeler (cerveau, papier, crayon) pourquoi ce minimum est atteint en l'unique X_0 tel que ${}^tAAX_0 = {}^tAY$.*
En déduire la valeur du X_0 correspondant, ainsi que AX_0.

On pourra après-coup aller voir du côté de la fonction `lstsq` (least square) de la bibliothèque `numpy.linalg`.

1. Soupirs...

6 Des dérivées (15 minutes)

On définit (lorsque le membre de droite est défini !) :

$$f(x) = \cos\left(\sqrt{1 + \tan\left(\ln\left(1 + x\cos^2 x\right)\right)}\right)$$

ainsi que :

$$g(x, y) = x^2 \cos(x - y\mathrm{e}^x) \ln\left(y^2 + \mathrm{e}^{x-y}\right)$$

Exercice 15. *Évaluer* $f'(945)$ *ainsi que* $f''(2016)$.

Pour f' on comparera deux formules :

$$f'(x_0) \simeq \frac{f(x_0 + h) - f(x_0)}{h} \qquad \text{et} \qquad f'(x_0) \simeq \frac{f(x_0 + h) - f(x_0 - h)}{2h}$$

avec différentes valeurs de h.
Pour f'', on pourra retrouver une formule d'approximation en partant de

$$\begin{cases} f(x_0 + h) = f(x_0) + hf'(x_0) + \dfrac{h^2}{2} f''(x_0) + o(h^2) \\ f(x_0 - h) = f(x_0) - hf'(x_0) + \dfrac{h^2}{2} f''(x_0) + o(h^2) \end{cases}$$

Et ici encore, on pourra tester différentes valeurs de h...

Exercice 16. *Évaluer* $\dfrac{\partial g}{\partial x}(-1, -1)$ *ainsi que le laplacien*

$$\Delta g = \frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2}$$

en $(1, 1)$

Je vous laisse inventer votre formule d'approximation !