Institut Fourier
Université Grenoble I

L3 Méthodes Numériques
$2^{\text{ème}}$ semestre 2007/2008

Projets Scilab

Chaque élève doit choisir un des projets suivants et rendre un dossier sur ce projet le **mardi 29 avril** au plus tard. La note comptera pour moitié du contrôle continu. Plusieurs élèves pourront prendre le même projet mais devront rendre des dossiers séparés et *différents*.

Consignes pour la phase recherche : le texte du projet n'est qu'un guide, le projet reste relativement ouvert. On pourra suivre les pistes proposées, mais aussi en imaginer d'autres. Bien sûr, le gros du travail doit s'appuyer sur des programmes Scilab. Il ne faut absolument pas hésiter à venir discuter du projet avec les enseignants. Les élèves pourront aussi discuter entre eux d'un projet, mais il est ensuite demander de travailler séparemment, par exemple sur des problématiques différentes ou sur des simulations différentes.

Consignes pour le dossier : le dossier doit être rédigé individuellement et doit contenir :
- une introduction présentant la problématique,
- un résumé de la démarche suivie, des problèmes rencontrés et comment ils ont été éventuellement résolus,
- les explications des programmes,
- les résultats des simulations et les conclusions.

Concernant la note : la note représentera la moitié du contrôle continu. Ce qui comptera n'est pas forcément le nombre de questions résolues, mais plutôt la qualité et l'originalité du travail fourni, la capacité à choisir le programme adapté au problème (donc aussi à justifier ce choix) et à programmer correctement un code scilab. Les programmes seront à fournir aux enseignants en vue d'un test : pensez bien à les sauvegarder.

Projet n°1 : le mouvement des planètes.
On se place dans $\mathbb{R}^2$. En mécanique classique, une planète P_1 de masse m_1 et de position $x_1 \in \mathbb{R}^2$ subit, de la part d'une autre planète de masse m_2 et de position x_2, une force de gravité donnée par

$$F = -\mathcal{G}\frac{m_1 m_2 (x_1 - x_2)}{\|x_1 - x_2\|^3} \ ,$$

où $\mathcal{G}$ est la constante universelle de gravitation. Le mouvement de la planète P_1 est donc décrit par sa position $x_1(t) \in \mathbb{R}^2$ et sa vitesse $v_1(t) \in \mathbb{R}^2$ dont l'évolution est donnée par

$$\begin{cases} \partial_t x_1(t) = v_1(t) \\ \partial_t v_1(t) = -\mathcal{G}\frac{m_2(x_1 - x_2)}{\|x_1 - x_2\|^3} \\ (x_1, v_1)(t=0) = (x_1(0), v_1(0)) \in \mathbb{R}^2 \times \mathbb{R}^2 \end{cases}$$

1) Ecrire un programme Scilab simulant la trajectoire d'une planète P gravitant autour d'un soleil fixe placé au point $(0,0)$. Observer la forme des trajectoires. Comparer le résultat obtenu par la méthode d'Euler et par RK4 (on regardera en particulier la périodicité des orbites).
2) Vérifier numériquement les lois de Kepler :
- les trajectoires fermées sont des ellipses dont le soleil est un foyer,
- l'aire balayée par le vecteur $\overrightarrow{Ox}(t)$ entre un temps t et $t+\tau$ ne dépend que de τ et pas de t,
- si T est la période de l'orbite et a le demi grand axe de l'ellipse décrite, le rapport T^2/a^3 est une constante et ne dépend pas de la trajectoire de la planète.
3) Simuler le comportement de deux objets célestes de même masse orbitant l'un autour de l'autre.

Projet n°2 : compression et débruitage d'un signal sonore.
On considère des signaux sonores c'est-à-dire la donnée d'une fonction dérivable $t \in [0,1] \mapsto f(t)$ représentant la pression de l'air au cours du temps. D'un point de vue numérique, le signal est enregistré dans un vecteur v de longueur n donné par $v_i = f(i.dt)$ avec $dt = 1/n$ (n sera grand, par exemple de l'ordre de 1000).
1) Démontrer qu'il existe des coefficients (a_n) tels que

$$\forall t \in]0,1[, \ f(t) = \sum_{n=1}^{\infty} a_n \sin(n\pi t) \ .$$

Donner l'expression des coefficients a_n en fonction de f.
2) Faire un programme Scilab calculant les coefficients a_n et affichant le graphique de f et celui de $\sum a_n \sin(n\pi t)$. NB : on prendra garde de travailler sur un signal choisi de façon à bien tester le programme. Par exemple pas déjà de la forme $\sum a_n \sin(n\pi t)$, mais

plutôt comprenant différents aspects (une zone de plateau, une forte variation, une zone d'oscillation etc.).
3) Supposons maintenant que le signal est bruité, c'est-à-dire qu'il est perturbé par une partie aléatoire $W(t)$. L'expérience montre que ce bruit modifie surtout les hautes fréquences du signal, c'est-à-dire les coefficients a_n pour n grand. Proposer une façon de débruiter le signal et la programmer. Est-elle efficace ?
4) On souhaite réduire la taille du stockage du signal qui est la taille n du vecteur v. Proposer une façon de compresser le signal et la programmer. Etudier numériquement l'erreur commise lors de cette compression en fonction du taux de compression.

Projet n°3 : calcul du volume de la boule unité de $\mathbb{R}^d$.
On souhaite calculer une intégrale multiple $\int_{\mathbb{R}^d} f(x)dx$. Si la dimension d est grande, les méthodes de type rectangles ou trapèzes sont très couteuses. Une solution est d'utiliser une méthode appelée méthode de Monte-Carlo. Afin de se fixer les idées, on prend $f(x) = 1$ si $\|x\| < 1$ et 0 sinon. Autrement dit, on souhaite calculer le volume V_d de la boule unité de $\mathbb{R}^d$.
1) Trouver une relation de récurrence entre V_{d+1} et V_d. En déduire une expression explicite de V_d. Faire une fonction scilab renvoyant cette valeur.
2) Faire un programme scilab calculant V_d à l'aide de la méthode des rectangles.
La méthode de Monte-Carlo est la suivante. On tire au hasard de nombreux points x dans $[-1, 1]^d$. D'après la loi des grands nombres, le rapport $V_d/2^d$ sera approché par la proportion de points x dans la boule unité par rapport au nombre total de points tirés au hasard.
3) Faire un programme calculant V_d à l'aide de la méthode de Monte-Carlo. Comparer les performances avec la méthode des rectangles pour d petit et pour d grand.
4) On peut prouver que l'on peut aussi effectuer le même type d'algorithme en remplaçant la suite de points aléatoires par une suite de points du type $x_{n+1} = x_n + \tau$, où $\tau = (\tau_1, ..., \tau_d)$ est un vecteur tel que les nombres $(1, \tau_1, ..., \tau_d)$ sont irrationnels entre eux (i.e. si $\sum r_i\tau_i \in \mathbb{Q}$ avec $r_i \in \mathbb{Q}$, alors $r_i = 0$ pour tout i). Programmer cette méthode.

Projet n°4 : modèle logistique et chaos.
On définit la fonction suivante.

$$f_r : \begin{pmatrix} [0,1] & \longrightarrow & [0,1] \\ x & \longmapsto & f_r(x) = rx(1-x) \end{pmatrix}$$

On considère le système dynamique $S_r(n)$ qui consiste à itérer cette fonction : pour $u_0 \in [0, 1]$, on pose $S_r(n)(u_0) = f_r^n(u_0)$. Une trajectoire de ce système est donc une suite $S_r(n)(u_0)$ définie par récurrence par $S_r(n+1)(u_0) = f_r\left(S_r(n)(u_0)\right)$.
Il s'agit d'un modèle intéressant pour la biologie mathématique. En effet, c'est le modèle le plus simple pour l'évolution d'une population animale dont $S_r(n)(u_0)$ représente la densité

après n années (ou autres unités de temps). La fonction f_r est choisie car elle répond aux principes suivants : si $S_r(n)(u_0)$ est trop petit, $S_r(n+1)(u_0)$ sera petit (petite densité de reproducteurs), si $S_r(n)(u_0)$ est trop grand (proche de 1), alors $S_r(n+1)(u_0)$ sera aussi petit (trop grande concurrence, surexploitation du milieu etc.). Le but est de comprendre la dynamique de ce modèle. Par exemple, on peut chercher à savoir si, pour un certain r donné, $S_r(n)(u_0)$ tend toujours vers 0, c'est-à-dire que l'espèce animale finit toujours par s'éteindre.
1) Montrer que pour $r \in [0,4]$, $S_r(n)$ est toujours bien défini de $[0,1]$ dans $[0,1]$.
2) On suppose que $r \in [0,1]$. A l'aide d'un programme Scilab, étudier le comportement de $S_r(n)$ sur différentes données initiales. L'espèce animale survit-elle ? Prouver mathématiquement le comportement observé.
3) On suppose que $r \in [1,3]$. Etudier le comportement de $S_r(n)$ numériquement. Expliquer mathématiquement du mieux possible le comportement observé.
4) Ecrire un programme Scilab qui trace en fonction de $r \in [0,4]$ les points de $[0,1]$ qui sont dans l'adhérence de trajectoires du système $S_r(n)$ (indication : à r fixé, on peut regarder la valeur de $S_r(n)(u_0)$ pour n grand et pour un grand nombre de données initiales u_0 tirées au hasard et tracer l'ensemble des points obtenus). Commenter le graphique obtenu.
5) Mettre en évidence un r pour lequel toute trajectoire tend vers une orbite périodique de période 2.
6) Mettre en évidence un r pour lequel $S_r(n)$ a un comportement proche du chaos, c'est-à-dire que la trajectoire $S_r(n)(u_0)$ dépend très fortement de la donnée u_0 (une petite erreur sur u_0 change complètement le comportement de la trajectoire).

Projet n°5 : une équation avec retard.
On considère le modèle biologique suivant. Soit $x(t)$ le nombre d'adultes d'une espèce animale au temps t. On note $g(x)$ le nombre de naissances pour x adultes. Les jeunes mettant un certain temps à quitter l'âge juvénile, ils ne sont pris en compte dans la population $x(t)$ qu'après un temps $\tau \geq 0$. On trouve donc une équation du type

$$x'(t) = -x(t) + g(x(t-\tau)) \ , \tag{1}$$

où $g : \mathbb{R}_+ \longrightarrow \mathbb{R}_+$ est une fonction continue que l'on supposera positive et bornée. On prendra typiquement $g(x) = 50\frac{x}{1+x^{10}}$.
1) Rappeler le comportement de l'équation différentielle ordinaire obtenue quand $\tau = 0$.
2) En utilisant la méthode de la variation de la constante, montrer que pour tout $T \geq 0$ et $t \geq 0$,

$$x(T+t) = e^{-t}x(T) + \int_0^t e^{-(t-s)} g\left(x(T+s-\tau)\right) \, ds \ . \tag{2}$$

En déduire que si $x(t)$ est connu et positif pour $t \in [-\tau, 0]$, alors $x(t)$ existe et est positif pour tout $t \geq 0$.

3) Caractériser toutes les solutions $x(t)$ constantes en temps.
4) En s'inspirant du schéma d'Euler classique, proposer et programmer une simulation numérique de (2).
5) Observer que :
- si $\tau > 0$ est petit, les solutions convergent vers un équilibre,
- si τ est plus grand, les solutions peuvent converger vers des trajectoires périodiques,
- si τ est encore plus grand, des comportements chaotiques peuvent apparaître.

Projet n°6 : écoulement de l'air autour d'une montagne.
On se place dans $\mathbb{R}^3$ et on note $X = (x, y, z)$ les points de $\mathbb{R}^3$. On considère un fluide de masse volumique $\rho(X, t) \in \mathbb{R}$ et de vitesse $\vec{v}(X, t) \in \mathbb{R}^3$. Le bilan de masse lors de l'écoulement du fluide aboutit à l'équation

$$\partial_t \rho + \mathrm{div}(\rho \vec{v}) = 0 \ .$$

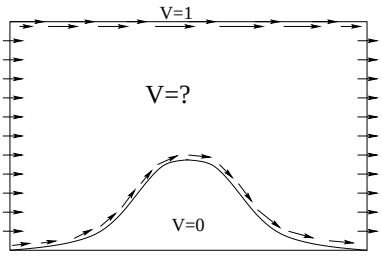
On suppose que le fluide est incompressible, c'est-à-dire que ρ est constant. On suppose en outre que l'écoulement est bidimensionnel et stationnaire et donc ne dépend que de x et z. On peut donc se restreindre à $X = (x, z) \in \mathbb{R}^2$. On peut alors montrer qu'il existe une fonction $v : \ (x, z) \mapsto v(x, z) \in \mathbb{R}$ telle que

$$\vec{v} = (\partial_z v) \ \vec{x} - (\partial_x v) \ \vec{z} \ .$$

Si on suppose de plus que $\mathrm{rot}(\vec{v}) = 0$ i.e. que le fluide est irrotationnel, c'est-à-dire que l'écoulement se fait sans tourbillons, alors on obtient comme équation

$$\Delta v(x, z) := \frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial z^2} = 0 \ . \tag{3}$$

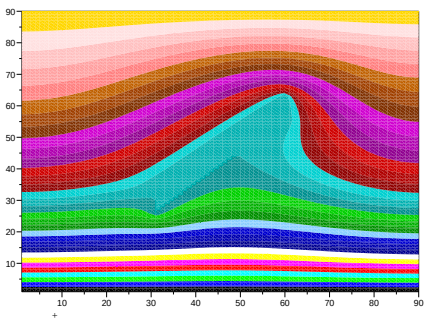
Pour obtenir l'écoulement d'un fluide, il suffit de trouver v vérifiant (3). Les lignes de niveau de la fonction v obtenue donnent les trajectoires des particules du fluide.
On souhaite simuler l'écoulement de l'air autour d'une montagne. Pour cela, on va résoudre numériquement (3) dans un domaine rectangulaire en imposant les conditions suivantes.
- Sur le bord du haut du rectangle, la circulation d'air est horizontale. La fonction V est donc constante (par exemple égale à 1). Autrement dit, on suppose que ce bord est assez loin de la montagne qu'elle ne modifie pas la circulation de l'air sur ce bord.
- Sur le sol, la circulation suit le profil de la montagne et donc v doit être constant sur toute la montagne (par exemple $v = 0$ sur le sol).
- Sur les bords gauche et droit du rectangle, on suppose pour simplifier que la fonction V varie linéairement de 1 en haut à 0 en bas. Cela revient à supposer que la montagne n'influe pas sur la fonction v sur ces bords et que cette dernière est donc égale à la solution sans la montagne.
En résumé, les conditions ressemblent à ceci :



Soit $n \in \mathbb{N}^*$. On discrétise le problème en posant remplaçant v par la matrice $n \times n$ schématiquement donnée par $V(i,j) = v(\frac{i-1}{n-1}, \frac{j-1}{n-1})$ $(i = 1, ..., n,\ j = 1, ..., n)$. On note $\mathcal{E}$ l'ensemble des matrice $n \times n$ vérifiant une version discrétisée des conditions aux bords précédentes.
1) Préciser en quoi consiste l'ensemble $\mathcal{E}$ et donner la discrétisation de l'équation (3) sous la forme $D(V) = 0$ et$V \in \mathcal{E}$, où D est un opérateur linéaire agissant sur les matrices $n \times n$.
2) Montrer que le problème revient à minimiser sur $\mathcal{E}$ la fonctionnelle

$$J(U) = \sum_{(i,j) \text{ voisin de } (i',j')} |V(i,j) - V(i',j')|^2 \ .$$

3) Pour résoudre (3), on propose l'algorithme suivant. On parcourt un grand nombre de fois l'ensemble des coefficients de la matrice V. Sur un coefficient donné, on regarde si on est autorisé à le modifier ou s'il est fixé par les conditions aux bords. Si on peut le changer, on lui donne la valeur pour laquelle le Laplacien discret $D(V)$ est nul à cet endroit, sinon on n'y touche pas. Ecrire le code Scilab correspondant et observer le résultat.
4) Montrer que l'algorithme précédent est une méthode d'optimisation de type relaxation à pas optimal et qu'elle converge.
5) Pour être plus réaliste, on peut envisager de changer la condition au bord sur les bords gauche et droit. On suppose simplement que le flot est horizontal, c'est-à-dire que $\partial_x V = 0$, mais on n'impose plus la valeur de V sur ces bords. Comment faire pour introduire cette condition au bord dans le programme ? A l'aide de cette nouvelle condition au bord, observer l'écoulement autour d'une surface inclinée type aile d'avion. On cherchera à obtenir un résultat similaire à celui-ci :



Projet n°7 : attracteur de Lorenz.
On considère le système d'équations différentielles suivant:

$$\begin{cases} \frac{dx}{dt} = 10(y-x) \\ \frac{dy}{dt} = -xz + \rho x - y \\ \frac{dz}{dt} = xy - \frac{8}{3}z \end{cases}$$

où $\rho > 1$ est un paramètre réel.
1) Trouver les solutions d'équilibre, c'est-à-dire les solutions où x, y, z sont indépendants du temps.
2) Fixer $\rho = 28$, et approcher la solution du système avec comme condition initiale $x(0) = -3$, $y(0) = -6$, $z(0) = 12$, sur l'intervalle de temps $[0, 20]$. Dans un premier temps, on pourra utiliser la méthode rk4, et étudier la variation des solutions en fonction du pas choisi (on pourra aussi comparer avec les résultats obtenus avec d'autres méthodes).
3) Etudier l'effet des erreurs, en perturbant les conditions initiales, par exemple en prenant $z(0) = 12.001$. Etudier en fonction de l'erreur initiale le premier temps pour lequel la solution perturbée est à distance ≥ 1 de la solution d'origine (distance mesurée par rapport à la norme euclidienne dans $\mathbb{R}^3$).
4) Etudier les maxima locaux de la troisième composante $z(t)$ de la solution (pour les conditions initiales non-perturbées). On notera z_n la valeur de $z(t)$ au n-ième maximum local. Faire un graphe de z_{n+1} en fonction de z_n.
5) Etudier la stabilité des points d'équilibre trouvés au 1) en fonction du paramètre ρ.

Projet n°8 : un algorithme de sélection naturelle.
Soit P_n un polygone à n côtés.
1) On rappelle que l'aire d'un triangle ABC est donnée par $\frac{1}{2}\left|\det\left(\overrightarrow{AB}, \overrightarrow{AC}\right)\right|$. Faire une fonction scilab calculant le périmètre et l'aire d'un polygone P_n.
On souhaite trouver P_n minimisant son périmètre pour une aire fixée, ce qui revient à trouver P_n minimisant $J(P_n) =$périmètre$(P_n)/\sqrt{\text{aire}(P_n)}$. Pour cela on propose l'idée suivante. On part d'un polygone P_n et on tire au hasard un nouveau polygone P'_n. Si $J(P'_n) < J(P_n)$ on remplace P_n par P'_n et on recommence ainsi.
2) Justifier la convergence de l'algorithme.
La convergence de cet algorithme est très lente. Pour l'accélérer, on choisit de changer de stratégie après un certain nombre d'itérations. On ne considère alors que les polygones proches de P_n c'est-à-dire que l'on tire au hasard une petite mutation de P_n et on regarde si cette mutation améliore l'efficacité de P_n. Si oui, elle est bien sûr gardée et on recommence le processus.
3) Programmer l'algorithme de mutation et observer son efficacité pour des petits et grands nombres n.
4) Programmer une autre méthode d'optimisation pour J (par exemple la méthode du

gradient à pas constant).

Astuces et commandes Scilab :

• Attention à bien tester dans un premier temps les programmes sur des paramètres petits. Mieux vaut voir les erreurs après quelques secondes de calculs que plusieurs minutes.

• L'affichage d'un graphique ralentit énormément une boucle. Pour observer l'évolution d'un phénomène sans trop le ralentir, on pourra afficher des graphiques que toutes les n itérations.

• Pour tracer un graphique composé de points en Scilab, en utilise `plot2d` avec l'option `style=n` où `n` est un entier négatif. Par exemple la ligne de commandes `plot2d([1,2],[3,4],style=-5,rect=[-1,-1,2,2])` dessinera deux petits ronds aux points $(1, 3)$ et $(2, 4)$ sans aucun trait les reliant.

• La commande `Matplot(A)` dessine une grille de la taille de la matrice A et dont chaque case a la couleur du coefficient correspondant dans A (qui doit donc être un entier positif).

• Pour dessiner une courbe paramétrée $(x(t), y(t), z(t))$ dans $\mathbb{R}^3$, il faut prendre un vecteur $t = a : h : b$, calculer $x = x(t)$, $y = y(t)$, $z = z(t)$, puis exécuter la commande `param3d1(x,y,z)`.

• Pour dessiner les lignes de niveaux d'une fonction $f(x, y)$, on utilise `Sgrayplot(x,y,A)`, où `A` est la matrice des valeurs de f sur les vecteurs `x` et `y`.

• L'aide Scilab en ligne est à l'adresse :
`http://www.scilab.org/product/man-eng/index.html`