Séance 3

12 janvier 2004

Les scripts et les fonctions pour réaliser ces exercices se trouvent à l'URL
`http://www-rocq.inria.fr/~haddar/Cours/ENSMP/`

Exercice 1 : Erreurs d'arrondis

Évaluer la fonction $f : x \mapsto (e^x - 1)/x$ pour $10^{-15} \leq x \leq 10^{-6}$ (une définition possible en Scilab est :
`x=logspace(-6,-15,10)`
`deff('y=f(x)','y=(exp(x)-1)./x')`).

A-t-on, numériquement, $\lim_{x\to 0} f(x) = 1$?

Reprendre la question en utilisant la formule modifiée :

$$f(x) = \begin{cases} 1 & \text{si } y = 1 \\ (y-1)/\ln(y) & \text{sinon} \end{cases} \qquad \text{avec } y = e^x, \tag{1}$$

ce qui peut s'écrire en Scilab

```
deff('y=f2(x)','z=exp(x);if (z==1) then y=1; else y=(z-1)./log(z);end')
```

Vous pouvez aussi exécuter directement le script `arith.sce`.

Exercice 2 : Conditionnement d'un système linéaire

Soit le système linéaire $Ax = b$, avec

$$A = \begin{bmatrix} 10 & 7 & 8 & 7 \\ 7 & 5 & 6 & 5 \\ 8 & 6 & 10 & 9 \\ 7 & 5 & 9 & 10 \end{bmatrix}, \quad b = \begin{pmatrix} 32 \\ 23 \\ 33 \\ 31 \end{pmatrix}$$

1. Résoudre le système (commande `x=A\b`).
2. Calculer la décomposition en valeurs singulières de $A = U\Sigma V^t$, et le conditionnement de A. Les commandes Scilab correspondantes sont : `svd`, `cond`.

3. On perturbe le système en ajoutant à b le vecteur $\delta b = 1/5\, v_4$ (un multiple du vecteur singulier associé à la plus petite valeur singulière de A, de façon que $||\delta b||/||b|| \approx 3/1000$). Il se calcule en Scilab par `db=v(:,4)/5`. Calculer la nouvelle solution $x + \delta x$

4. Calculer $\dfrac{||\delta x||/||x||}{||\delta b||/||b||}$, et expliquer ce résultat (comparer avec cond(A)).

Exercice 3 : Stabilité rétrograde de la factorisation de Cholesky

Prendre une matrice triangulaire aléatoire W (commande Scilab : `W=triu(rand(50,50))`. Calculer la matrice $A = W'W$, puis le facteur de Cholesky de A (la commande Scilab `R=chol(A)` renvoie une matrice triangulaire supérieure). Il est possible que vous obteniez un message d'erreur signalant que A n'est pas définie positive. Dans ce cas, recommencez !

On définit ensuite la matrice R_1 par $R_1 = R + 10^{-6}\delta R$ où δR est une matrice triangulaire supérieure aléatoire à éléments dans $]0, 1[$, (en Scilab, `R1=R+1e-6*triu(rand(50,50))`.

Comparez les quantités $\|W - R\|/\|W\|$ et $\|W - R_1\|/\|W\|$, ainsi que $\|A - R^tR\|/\|A\|$ et $\|A - R_1^tR_1\|/\|A\|$. Qu'observez-vous ?

Exercice 4 : Complexité de la factorisation LU

Le script `linbench.sce` (tapez : `exec 'linbench.sce'`) permet de mesurer la complexité de la factorisation LU. Il effectue une factorisation sur une suite de matrices aléatoires de taille 50 à 500 (par pas de 50). Le graphique représente le temps d'exécution en fonction de la taille (en échelle log–log, il s'agit donc approximativement d'une droite, de pente « théorique » égale à 3).

Exercice 5 : Lissage polynômial

On cherche à approcher une fonction par un polynôme de degré donné, en minimisant l'écart quadratique entre la fonction et le polynôme. Soient $(t_i)_{i=1,\ldots,m}$ les points où la fonction f est connue. Notons n le degré du polynôme, et $(x_j)_{j=1,\ldots,n}$ ses coefficients.

Dans cet exercice, on prend $f(t) = \exp(\sin(4t))$ sur $[0, 1]$, avec $m = 100$ et $n = 15$.

5.1 - Montrer que la matrice et le second membre du problème de moindres carrés correspondant sont donnés par $A_{ij} = t_i^j, \quad z_i = f(t_i)$. Ces formules se traduisent par le code Scilab suivant :

```
t=(0:m-1)'/(m-1); A=[];
```

```
for i=1:n,  A = [A t.^(i-1)]; end
b=exp(sin(4*t)); b=b/2006.7874531048527;
```

La valeur «magique» ci-dessus est la dernière composante de la solution «exacte» (calculée par Maple). Ainsi, on doit trouver $x_{15} = 1$.

5.2 - Comparer les solutions obtenues par les quatre méthodes suivantes :

1. Équations normales (`x1=(A'*A)\(A'*b)`) ;
2. Algorithme QR (`[Q,R]=qr(A); x2=R\(Q'*b);`) ;
3. SVD de A (`[U,S,V]=svd(A); x3=V*(S\(U'*b));`) ;
4. Algorithme par défaut de Scilab (`x4=A\b;`).

Comparer graphiquement les données «experimentales» avec les différentes prédictions (remarquer qu'elles se calculent par `A*x`, où `x` est l'une des solutions calculées ci-dessus.

Exercice 6 : SVD et compression d'image

Une image (en niveau de gris) peut être considérée comme une matrice dont chaque élément est la valeur du pixel correspondant (ramené dans $]0, 1[$). Notons A cette matrice, et soit $A = U\Sigma V^t$ sa décomposition en valeurs singulières. Pour $1 \leq k \leq n$, notons $\Sigma_k = \text{diag}(\sigma_1, \ldots, \sigma_k, 0, \ldots, 0)$, et remplaçons A par $A_k = U\Sigma_k V^t$.

Comment stocker économiquement A_k? Quelle est l'économie réalisée?

Le script `imcomp.sce` (tapez : `exec 'imcomp.sce'`) lit une image, calcule la SVD de la matrice correspondante, et représente les images obtenues en ne gardant que les quelques plus grandes valeurs propres. Il quantifie l'erreur commise, et l'économie réalisée. Il représente également les valeurs propres de la matrice (en échelle semi logarithmique).