Quelques corrections de TP sur le chapitre 1 “Erreurs de calculs...” (en SCILAB)

Solution des exercices 1 et 2

```
//Programme "pluspetitreel.sce"
//calcul du plus petit reel et plus grand reel >0
clear
x=1;
while x>0
      y=x;
      x=x/2;
end
'le plus petit reel strictement positif est '
y  //y est le plus petit reel >0
N1=log2(y)

x=1;
while x<%inf
      y=x;
      x=2*x;
end
'le plus grand reel est'
y
N1=log2(y)
```

Remarquer que si N est le nombre de chiffres dans la mantisse, et P le nombre de chiffre dans l'exposant, le plus petit réel $x > 0$ représentable est

$$x = 2^{-(N-1)} * 2^{-2^P-1},$$

et le plus grand réel représentable est

$$X = 2 * 2^{2^P-1},$$

de sorte que

$$\begin{cases} log_2(x) = -(N-1) - (2^P - 1), \\ log_2(X) = 2^P, \end{cases}$$

donc la connaissance de x et X permet de calculer N et P. Dans notre cas, on trouve $log2(x) = -1074 = -51 - 1023$, et $log_2(X) = 1023$. Comme $2^{10} = 1024$, on vérifie que cela correspond bien à $N = 52$ et $P = 10$ (à une unité près, qui dépend des règles d'arrondis du logiciel et du test d'arrêt dans notre algorithme).

Solution des exercices 3 et 4

```
//Programme condition.sce
//Quelques essais de calcul
clear
A=[1 10;0 1] //remarquer que A n'est pas symetrique
//deux manieres de calculer la condition en norme 2
'condition 2 = '
cond(A)
//pour comparer
Lambda=spec(A'*A);
sqrt(max(Lambda)/min(Lambda))
//condition en norme 1 (deux manieres)
'condition 1 = '
1/rcond(A)
norm(A,1)*norm(inv(A),1)
//condition en norme infinie (trois manieres)
'condition inf = "
1/rcond(A')
norm(A,'inf')*norm(inv(A),'inf')
norm(A',1)*norm(inv(A'),1)

//condition de la matrice du laplacien
clear
x=[];
y2=[];//tableau du conditionnement en norme 2
y1=[];//tableau du conditionnement en norme 1
yinf=[];//tableau du conditionnement en norme infinie
kmax=6;
for k=2:kmax
    N=2^k
    A=2*diag(ones(N,1),0)-diag(ones(N-1,1),-1)-diag(ones(N-1,1),1);
    x=[x N];
    y2=[y2 cond(A)];
    y1=[y1 1/rcond(A)];
    yinf=[yinf 1/rcond(A')];
end
clf
plot2d(x',[y2',y1',yinf'],logflag="ll",leg="K2@K1@Kinf")
```

correction de l'exercice 5 (matrice de Van der Monde) SCILAB

```
//condition de la matrice de Van der Monde
TabN=[];//tableau des abscisses
y2=[];//tableau des ordonnees (condition en norme 2)
y1=[];//idem en norme 1
yinf=[];//idem en norme infinie
Nmax=40;
for N=1:Nmax
    x=[0:1/N:1];//construction des x_i, points equirepartis
    V=zeros(N+1,N+1);
```

```
    //construction de la matrice de Van der Monde
    for i=0:N
        for j=0:N
             V(i+1,j+1)=x(i+1)^j;
        end
    end //NB : la numerotation des tableaux commence toujours a 1 en Scilab
    TabN=[TabN N];
    y2=[y2 cond(V)];
    y1=[y1 1/rcond(V)];
    yinf=[yinf 1/rcond(V')];
end
clf
plot2d(TabN',[y2',y1',yinf'],logflag="nl",leg="K2@K1@Kinf")
xtitle('condition de la matrice de Van der Monde','N','Kappa')
```

NB : on n'a traité que le cas des points équidistants sur $[0, 1]$.

Le dessin affiché par ce programme est donne dans la Figure 1. On constate que le conditionnement croit de maniere exponentielle en fonction de N, puisque le logarithme de K_N, où $K_N = cond(V_N)$, est une fonction affine de N : $log_{10}(K_N) \approx aN + b$, ou encore, $K_N \approx 10^{aN+b} = C\delta^N$, avec $C = 10^b$ et $\delta = 10^a > 1$.

On remarque également que lorsque le conditionnement est de l'ordre de 10^{16}, les calculs ne sont plus fiables (les résultats sont de l'ordre de 10^{16}, de manière aléatoire). Cela est directement lié à la précision des calculs dans SCILAB ($\epsilon \approx 10^{-16}$). La matrice de Van der Monde est connue pour avoir un très mauvais conditionnement : pour $N \geq 25$ (ce qui n'est pas très grand pour une taille de matrice), on dépasse la précision de l'ordinateur.

Si on compare avec l'exemple précédent où le conditionnement est en $O(N^2)$, il faudrait une taille de matrice $N \approx 10^8$ pour arriver à la limite de précision.

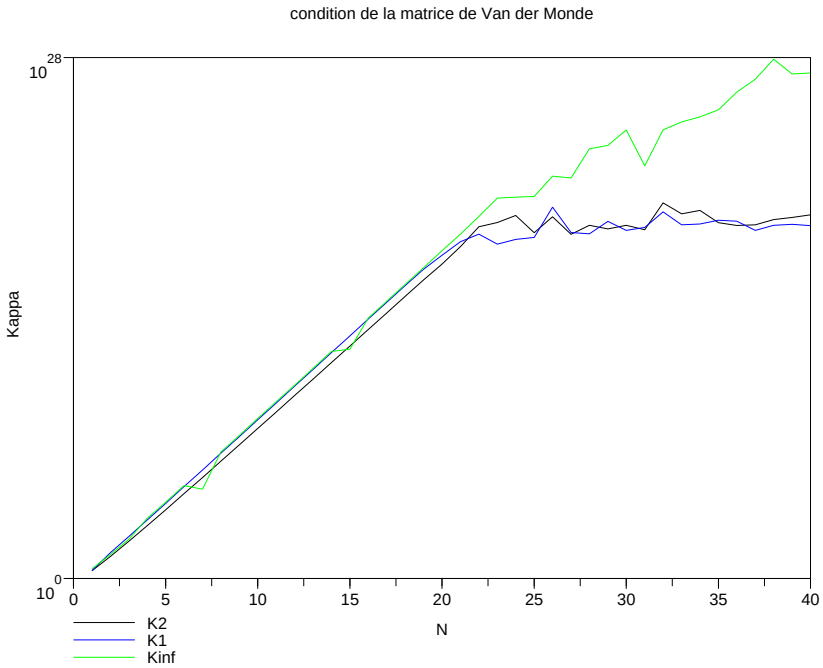

Figure 1: Condition de la matrice de Van der Monde

Programmation de la suite recurrente (exercice 7)

$$\begin{cases} x_0 = 1,\ x_1 = (1-\sqrt{5})/2, \\ x_{n+2} = x_{n+1} + x_n \qquad\qquad n \geq 0. \end{cases}$$

On constate que la suite semble d'abord converger vers 0 avant d'exploser vers $\pm\infty$.

```
//Programme "suiterecurrente.sce"
n=100;
x=1  //x_0
y=(1-sqrt(5))/2  //x_1
for k=2:n
    z=y+x  //x_{k}=x_{k-1}+x_{k-2}
    x=y;
    y=z;
end
```