ANALYSE NUMÉRIQUE - TPs SCILAB

Proposition d'exercices pour la session 1 de mai 2012

Exercice 1 *Correction*

```
->function y=f(x) // La fonction que l'on cherche à interpoler
-> y=1./(1+x.*x)
->endfunction

->n=6 ;
->x=linspace(-5,5,n)' ; // Les points d'interpolation régulièrement espacés
->F=f(x) ; // Les points d'interpolation (x,F), f(x)≝ 1/(1 + x^2)
->xnev= 1 :3 ; // Les points pour lesquels on veut la valeur de P_n
->xc=ones(n,1)*xnev - x*ones(1,size(xnev,'*')) ; // Matrice (xnev(j)-x(i))
->xd=x*ones(1,size(xnev,'*')) ;
->// Chaque colonne de la matrice col correspond aux
->// itérations de l'algorithme de Neville pour une valeur fixée de x (xnev(j)).
->A la fin des itérations col est un vecteur ligne
->col=F*ones(1,size(xnev,'*')) ; // Démarrage de la récursion
->for k=1 :n-1
-> col= xc(1 :-k, :).*col(2 :, :)-xc((1+k) :, :).*col(1 :-1, :) ;
-> col= col0./(xd((1+k) :, :)-xd(1 :-k, :)) ; // Mise à jour (le nombre de lignes de
-> // col diminue de 1)
->end

->clear P ;
->for i=1 :n
-> y=x ;
-> y(i)=[] ; P(i)= poly(y,"x") ;
-> P(i)= P(i)/horner(P(i),x(i)) ;
->end
->Pn= F'*P ;
->y=horner(Pn,xnev) ; // Comparons avec l'interpolation de Lagrange

->if norm(y-col) > 10*%eps then pause ;end
```

Exercice 2 *Correction* :
Ref : http ://u.math.biu.ac.il/ schiff/Teaching/376/376.html

Simple Scilab routine for Newton's method
Please note : the functions below ignore a number of subtleties at the aim of being simple and (hopefully) understandable.

Newton's method is used to solve a system of equations f(x)=0 (here x is a d-dimensional vector and f is a d-dimensional vector of functions). There are 3 functions below :

- The function f(x), which should be written by the user (a simple function of 2 variables is used in the code below).
- The function newtstep(x,h). This takes a single Newton step starting from x. Derivatives of f are computed using the simplest difference formula with "stepsize" h (in the 1 dimensional case this means approximating f'(x) by (f(x+h)-f(x))/h).
- The function newt(x,h,epsilon). This runs Newton's method, starting from x, using h as stepsize in

the derivatives and stopping when the norm of f is less than epsilon. DANGER : Newton's method might not converge, and no stopping criterion has been put in !

```
function a=f(x)
a(1)=x(1)*(x(1)*x(1)-3*x(2)*x(2))-1 ;
a(2)=x(2)*(3*x(1)*x(1)-x(2)*x(2)) ;
endfunction

function a=newtstep(x,h)
n=size(x,1) ; // get the dimension of the COLUMN vector submitted
F=f(x) ; // compute f(x)
M=zeros(n,n) ; // prepare to make the matrix of derivatives
for i=1 :n
v=zeros(n,1) ;
v(i)=h ;
M( :,i)=(f(x+v)-F)/h ;
end
a=x-M\F ; // do the Newton step
endfunction

function y=newt(x,h,epsilon)
y=x ;
z=norm(f(y)) ; // stop when this is below epsilon
while z>epsilon
y=newtstep(y,h) ;
z=norm(f(y)) ;
end
endfunction
```

Example of use
I put the three functions above into a file called "newt". Here is the transcript of a scilab session in which I execute the file "newt" and then use the commands to find the three roots of f=0 (for the f above) :

```
->exec newt ;

->newt([1.5 ;0.5],1e-6,1e-6)  ans =

! 1. !
! - 1.203E-10 !

->newt([-1.0 ;0.8],1e-6,1e-6)
ans =

! - 0.5 !
! 0.8660254 !

->newt([-1.0 ;-0.8],1e-6,1e-6)
ans =

! - 0.5 !
! - 0.8660254 !

->
```