TME n°1 - Introduction à MATLAB et à l'arithmétique à virgule fottante

Victor Piriou - M1 IMA

Exercice 1

1.

On a : $Highram(x) = (^(2^n))\sqrt{x})^(2^n) = x$

Donc, $Highram(4) = 4$. Le programme doit normalement afficher 4.

2.

En exécutant le programme, on obtient :

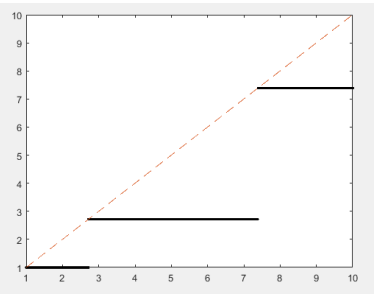
```
ans =

    2.718281808182473e+00
```

La fonction racine carré renvoie souvent des nombres qui ont beaucoup de chiffres après la virgule. Or, on sait que la taille de la mantisse est limitée à la double précision. Par conséquent, il y a souvent des chiffres significatifs qui sont perdus. Etant donné que ces racines carrées sont répétées plusieurs fois (dans la boucles for), ces erreurs d'arrondis sont cumulées. De plus, elles sont aussi amplifiées à cause de la puissance 2.

D'après le cours on sait avec la norme IEEE 754, l'arrondi est effectué après avoir effectué les opérations comme si elles étaient faites en précision infini.

Par défaut on arrondi au plus près.

3.



La ligne en pointillé réprésente la courbes des valeurs qu'on devrait normalement obtenir avec Highram(x). Tandis que la fonction en escalier correspond aux valeurs que l'on obtient dans la pratique.
Highram(x) renvoie les bonnes valeurs seulement pour environ x=1, x=2.72 et x=7.39.

Exercice 2

1.

```
function res = I(n)
    res = 1 - exp(-1);
    for i=1:n
        res = -exp(-1) + i*res;
    end
end
```

2.

on a : $0 \leq x^n \leq 1$ et $e^{-1} \leq e^{-x} \leq 1$

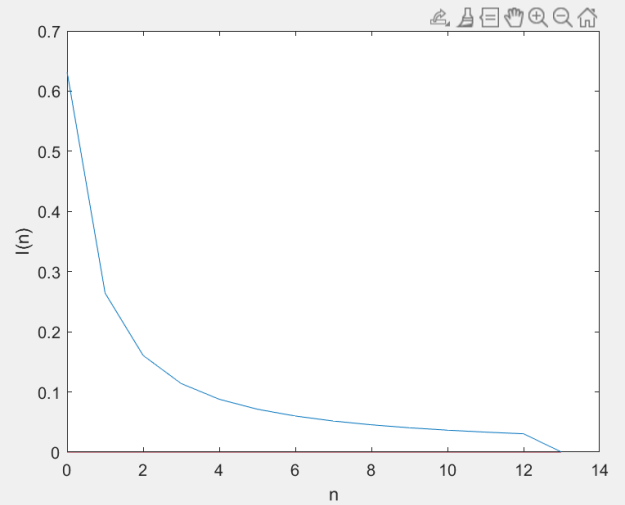
le produit de ces deux inéquations donne : $0 \leq x^n e^{-x} \leq 1$

$\Leftrightarrow \int_0^1 0\,dx \leq \int_0^1 x^n e^{-x}\,dx \leq \int_0^1 1\,dx$

$\Leftrightarrow 0 \leq \int_0^1 x^n e^{-x}\,dx \leq 1$

$\Leftrightarrow 0 \leq I_n \leq 1$

```
format longE;

n = 13;
global y;
y = zeros(n+1);
I(n);
x = linspace(0,n,n+1);
plot(x,y);
xlabel('n');
ylabel('I(n)');

function res = I(n)
    res = 1 - exp(-1);
    global y;
    for i=1:n
        y(i) = res;
        res = -exp(-1) + i*res;
    end
end
```



On constate que $I(n)$ est bien compris en 0 et 1. Inutile de vérifier les valeurs avec $n = 13$ car la fonction ne peut que décroître.

3.

$I_n = -e^{-1} + n.I_{n-1}$

On pose $n' = n - 1$, ce qui donne :

$I_{n'+1} = -e^{-1} + (n' + 1).I'_n$

$\Leftrightarrow I_{n'} = \frac{I_{n'+1}+e^{-1}}{n'+1}$

soit $I_n = \frac{I_{n+1}+e^{-1}}{n+1}$

4.

```
function res = I(n,m)
    if (m == 50)
        res = 12;
    else
        res = (I(n+1,m+1) + exp(-1))/(n+1);
    end
end
```

5.

```
format longE;

m = [10 20 50 100];

y = zeros(length(m));
n = 5;

for i=1:length(m)
    y(i) = I(n,0,m(i));
    fprintf('pour m=%d on a I(%d)=%d\n',m(i),n,y(i))
end

plot(m, y, '-x');
xlabel('m');
ylabel(strcat('I(',string(n),')'));

function res = I(n,x,m)
    if (x == m)
        res = 12;
    else
        res = (I(n+1,x+1,m) + exp(-1))/(n+1);
    end
end
```

on obtient :

```
pour m=10 on a I(5)=7.130218e-02
pour m=20 on a I(5)=7.130218e-02
pour m=50 on a I(5)=7.130218e-02
pour m=100 on a I(5)=7.130218e-02
```

On constate que la valeur $I(5)$ est la même pour ces 4 valeurs.

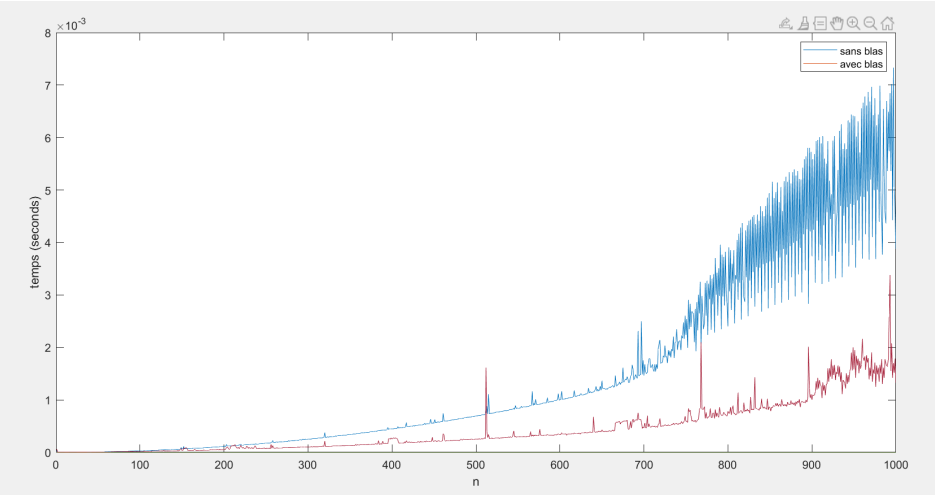
Exercice 3

1.

```
n = 1000;
y_no_blas = zeros(n);
y_blas = zeros(n);

for num_ligne_col=1:n
    M = randi([-10000,10000], [num_ligne_col,num_ligne_col]);
    total_blas = 0;
    total_no_blas = 0;
    for i=1:5 % moyenne pour augmenter la fiabilité
        tic;
        s_with_BLAS(M);
        total_blas = total_blas + toc;
        tic;
        s(M);
        total_no_blas = total_no_blas + toc;
    end
    moyenne_blas = total_blas/5;
    y_blas(num_ligne_col) = moyenne_blas;
    moyenne_no_blas = total_no_blas/5;
    y_no_blas(num_ligne_col) = moyenne_no_blas;
end

x = linspace(1,n,n);
plot(x,y_no_blas,x,y_blas);
xlabel('n');
ylabel('temps (seconds)');
legend("sans blas", "avec blas" )
```

1.

```
function S = s(M)
    [m,n] = size(M);
    S = zeros(n,1);

    for i=1:m
        for j=1:n
            S(i) = S(i) + abs(M(i,j));
        end
    end
end

function S = s_with_BLAS(M)
    [m,n] = size(M);
    S = zeros(n,1);
    for i=1:m
        S(i) = norm(M(i,:),1);
    end
end
```



Pas de surprise, on constate que l'utilisation des BLAS permet de gagner en temps de calcul.

2.

Il reste une erreur à résoudre dans le code mais c'est le même principe que la question d'avant pour mesurer les performances. Les BLAS permettront de gagner du temps.

```
n = 501;
pas = 100;

y_no_blas = zeros(cast(n/pas,'uint8')+1);
y_blas = zeros(cast(n/pas,'uint8')+1);


for num_ligne_col=1:pas:n
    M1 = randi([-10000,10000], [num_ligne_col,num_ligne_col]);
    M2 = randi([-10000,10000], [num_ligne_col,num_ligne_col]);
    C = zeros(num_ligne_col,num_ligne_col);
    total_blas = 0;
    total_no_blas = 0;
    for m=1:5 % moyenne pour augmenter la fiabilité
        tic;
        % produit des deux matrices :
        % pour chaque ligne de A
        for i = 1:num_ligne_col
            % pour chaque colonne de B
            for j = 1:num_ligne_col
                % s contiendra la valeur de C(i,j)
                % donc on réinitialise s avant
                s = 0;
                % pour chaque colonne de A et chaque ligne de B
                for k = 1:num_ligne_col
                    s = s + M1(i,k) * M2(k,j); % calcul de C(i,j)
                end
                C(i,j) = s;


            end
        end
        total_blas = total_blas + toc;
        tic;
        M1*M2;
        total_no_blas = total_no_blas + toc;
    end
    moyenne_blas = total_blas/5;
    y_blas(num_ligne_col) = moyenne_blas;
    moyenne_no_blas = total_no_blas/5;
    y_no_blas(num_ligne_col) = moyenne_no_blas;
end

x = linspace(1,n,n/pas+1);
disp(size(x));
disp(size(y_no_blas));
plot(x,y_no_blas,x,y_blas);

xlabel('n');
ylabel('temps (seconds)');
legend("sans blas", "avec blas" )
```