

## Correction du TP n°3 - Partie n°1 -

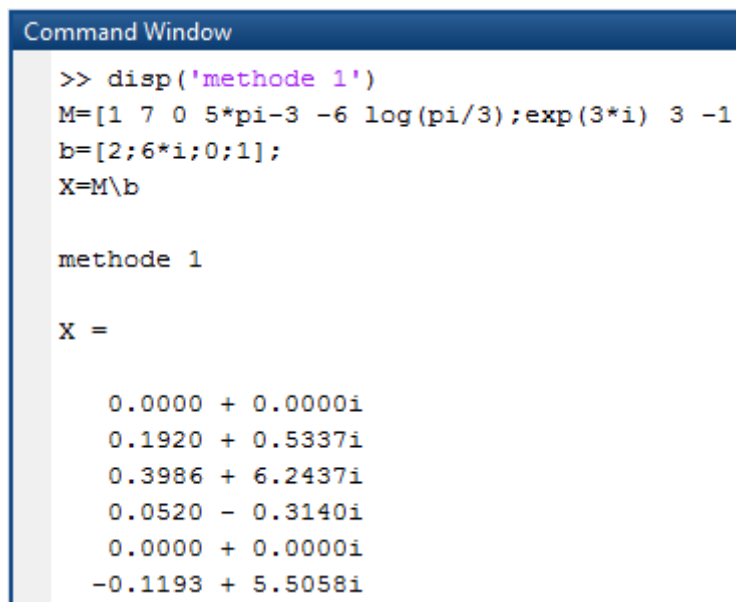
Tout d'abord je voudrais vous informer que cette correction est basée sur les réponses de votre collègue Nabil qui interagit avec moi par ses réponses aux anciennes séries.....Bravo Nabil !!

### Exercice 1:

```
(1.1) disp('methode 1')
M = [1 7 0 5*pi-3 -6 log(pi/3);exp(3*i) 3 -1 tan(sqrt(3)/2) 0 2;1 -2 0 9 -1
     cos(pi/4);1 -1 2 3 -sqrt(2) -2];

b = [2;6*i;0;1];

X=M\b
```



```
Command Window

>> disp('methode 1')
M=[1 7 0 5*pi-3 -6 log(pi/3);exp(3*i) 3 -1
tan(sqrt(3)/2) 0 2;1 -2 0 9 -1
cos(pi/4);1 -1 2 3 -sqrt(2) -2];
b=[2;6*i;0;1];
X=M\b

methode 1

X =

    0.0000 + 0.0000i
    0.1920 + 0.5337i
    0.3986 + 6.2437i
    0.0520 - 0.3140i
    0.0000 + 0.0000i
   -0.1193 + 5.5058i
```

### Méthode 2 :

%%% Vu que la matrice M n'est pas inversible et que le nombre des équations est strictement inférieur au nombre des inconnues ( $m < n$ ) cela vaut dire qu'on a un système « Sous-déterminé » et que le  $\text{rang}(M) = 4 =$  nombre de ligne ça vaut dire que le système admet une infinité des solutions.

**Exercice Supplémentaire :** Pour faciliter cette tache on pourra reformuler le système à 4 inconnus :

$$S' = \begin{cases} x_1 + 7x_2 + (5\pi - 3)x_4 & = 2 \\ e^{3i}x_1 + 3x_2 - x_3 + \tan\left(\frac{\sqrt{3}}{2}\right)x_4 & = 6i \\ x_1 - 2x_2 + 9x_4 & = 0 \\ x_1 - x_2 + 2x_3 + 3x_4 & = 1 \end{cases}$$

et vous devez essayer d'écrire un programme en Matlab qui illustre la méthode de pivot de Gauss

## Exercice 2 :

(2.1) `P = [2 5 sqrt(3) 0 0 6 5 12]`  
`disp('les racines de ce polynôme sont :')`  
`roots(P)`

```
P =  
Columns 1 through 7  
2.0000 5.0000 1.7321 0 0 6.0000 5.0000  
Column 8  
12.0000  
les racines de ce polynôme sont :  
ans =  
-2.2360 + 0.0000i  
0.9811 + 0.6687i  
0.9811 - 0.6687i  
-1.0122 + 0.8320i  
-1.0122 - 0.8320i  
-0.1009 + 1.0482i  
-0.1009 - 1.0482i
```

(2.2) `W = [1 -2 3];`  
`H = conv(W,P)`

```
H =  
Columns 1 through 7  
2.0000 1.0000 -2.2679 11.5359 5.1962 6.0000 -7.0000  
Columns 8 through 10  
20.0000 -9.0000 36.0000
```

(2.3) `[Q,R]=deconv(P,W)`

```
>> [Q,R]=deconv(P,W)  
Q =  
2.0000 9.0000 13.7321 0.4641 -40.2679 -75.9282  
R =  
Columns 1 through 7  
0 0 0 0 0 0 -26.0526  
Column 8  
239.7846
```

```

%verification
P;
T=conv(W,Q)+R
if P~=T
disp('faux')
else
disp('vrai')
end

```

```

>> %verification
P;
T=conv(W,Q)+R
if P~=T
disp('faux')
else
disp('vrai')
end

T =

Columns 1 through 7

    2.0000    5.0000    1.7321         0         0    6.0000    5.0000

Column 8

   12.0000

vrai

```

(2.4) `disp('le polynôme quotient');`  
`Q`  
`disp('et le polynome qui reste de la division =')`  
`R`

```

le polynôme quotient

Q =

    2.0000    9.0000   13.7321    0.4641  -40.2679  -75.9282

et le polynome qui reste de la division =

R =

Columns 1 through 7

         0         0         0         0         0         0    -26.0526

Column 8

   239.7846

```

(2.5) `[r,p,k]=residue(Q,1)`

```

>> [r,p,k]=residue(Q,1)

r =

     []

p =

     []

k =

    2.0000    9.0000   13.7321    0.4641  -40.2679  -75.9282

```

```

%remarque
if k==Q
disp('le polynôme Q n`admet pas une décomposition')
end

```

```
>> %remarque
if k==Q
disp('le polynôme Q n`admet pas une décomposition')
end
le polynôme Q n`admet pas une décomposition
```

(2.6) polyder(P)

```
>> polyder(P)
ans =
14.0000 30.0000 8.6603 0 0 12.0000 5.0000
```

(2.7) polyint(W)

```
>> polyint(W)
ans =
0.3333 -1.0000 3.0000 0
```