وزارة التعليم العالي و البحث العلمي

Université Badji Mokhtar Annaba

Faculté: Sciences de la terre
Département: Géologie

Module: Informatique

Troisième année licence

Spécialités : hydrogéologie, GRM , géologie fondamentale

Mme Boubaker

Sommaire :

Cours 1 : Gestion des paramètres d'une figure

-axis : permet la gestion des échelles sur les axes.

Utilisation : dans les commandes vues les axes sont utilisés comme paramètres des fonctions uniquement. Matlab vous permet de définir aussi les axes sans les fonctions avec la commande axis :

Axis(xmin xmax ymin ymaxzmin zmax)

-title :permet l'ajout d'un texte de titre sur une fenetre

-xlabel : permet l'ajout d'un texte de légende pour l'axe X sur un graphique

-ylabel : permet l'ajout d'un texte légende pour l'axe Y sur un graphique

-zlabel : permet l'ajout d'un texte légende pour l'axe Z sur un graphique

Ces quatre commandes s'utilisent de manière identique :

nom-com('texte à ajouter')

Ou nom-com est une fonction et le texte à insérer est entre cotes.

-text : ajout d'un texte à un endroit quelconque sur un graphique. Le début du texte est positionné à l'endroit précisé par les coordonnées x et y.

text(x,y,'texte')

-gtext : ajout d'un texte à un endroit quelconque dans un graphique. Le texte est placé à l'aide de la souris. Le début du texte est positionné à l'endroit précisé par la croix du curseur.

gtext('texte ')

Cours 2 : Opérations sur les polynômes

-Poly : Création d'un polynôme a partir de ses racines

Utilisation : y=poly([x1 x2 x3]) créée le polynôme y=(x-x1)(x-x2)(x-x3) sous sa forme développée

Y=poly(vec) crée un polynôme à partir d'un vecteur vec. Ce dernier doit comporter la liste des racines de y.

-roots : calcul des racines d'un polynôme

Utilisation : lis-rac=roots(polyno) crée un vecteur lis-rac qui contient les racines du polynôme polyno

C'est l'opération inverse de poly.On fournit un polynôme et la commande retourne les racines. Un polynôme a la forme suivante sous matlab : si on veut saisir y=ax2+bx+c on tape y=[a b c]

-polyval : évaluation d'un polynôme pour une valeur donnée

Utilisation y-val=polyval(y,val) fournit la valeur numérique que prend le polynôme y lorsqu'on lui applique la valeur numérique val

-conv : multiplication de polynômes

Utilisation z=conv(x,y) crée le polynôme z=x*y sous sa forme développé

-deconv : division polynomiale

Utilisation [quot,rest]=deconv[num,den] réalise la division du polynôme num par le polynôme den.

Exercices avec corrections sur les polynômes

Exercices :

Exercice 1 :

Soit le polynôme $y(x)=3x4+5x^2+2x-1$

Donner le programme qui calcule les racines de y(x) ainsi que les évaluations de y(x=1) et y(x=9)

Exercice 2 :

Soient $x(t)=3t^2+2t+1$ et $y(t)=t$

Donner le programme qui calcule le produit des deux polynômes

Exercice 3 :

Soient les fonctions x^2-4x+4 et $x-3$

Donner le programme qui donne la division des deux polynômes

Corrections :

Exercice 1 :

```
>>y=[ 3 0 5 2 -1]

>>roots(y)

>>v1=polyval(y,1)

>>v2=polyval(y,9)
```

Exercice 2 :

```
>>x=[3 2  1]

>>y=[1 0 ]

>>z=conv(x,y)
```

Ans z=[3 2 1 0]

Exercice 3 :

```
>>[quot,rest]=deconv([1 -4 4],[1,-1])
```

Ans quot=[1 -3]   rest =[0 0 1]

Cours 3 : Gestion de cas

Comme dans tous les langages de programmation, on peut sous matlab arbitrer des situations différentes et faire des choix : nous avons deux types d'outils :les instructions conditionnelles et les boucles, ils peuvent s'utiliser séparément ou s'imbriquer.

1. Cas conditionnels :

-If permet l'exécution conditionnelle d'une séquence d'actions

Utilisation : if expression,

action1

action2

.

.

Action n

End

Si expression est vérifiée, les actions de 1à n sont exécutées sinon elles ne le sont pas et on passe directement à la commande située immédiatement après end.

-else permet la combinaison de deux séquences d'actions différentes au sein d'un if

Utilisation : if expression,

action1

action2

else

action3

action4

end

Si expression est vérifiée, les actions 1 et 2 sont exécutées sinon les actions 3 et 4 le sont.

La commande else est toujours utilisée avec if. On peut trouver un if sans else mais pas le contraire.

-elseif : permet l'imbrication de plusieurs cas de types if au sein d'un if.

Utilisation : if expression1,

action1

elseif expression2

action2

end

si expression1 est vérifiée laction1 est exécuté. Si expression1 n'est pas vérifiée mais que expression2 est vérifiée alors l'action2 est exécutée. Si expression1 et expression2 ne sont pas vérifiées, aucune des deux actions action1 et action2 n'est exécutée. D'une manière générale, les actions suivant le elseif ne sont exécutées que si toutes les expressions liées au if et elseif précédents sont fausses tandis que l'expression du elseif considérée est vérifiée.

La commande elseif est toujours utilisée avec if. On peut trouver un if sans elseif mais pas le contraire.

-switch permet d'aiguiller conditionnellement vers une séquence d'actions

Utilisation : switch switch-expression

case case-expression1,

action11,action12,……

case case-expression2,

action21,action22,…..

……

Otherwise,

Actionn1,actionn2,…..

End

Si l'expression case-expression1, correspondant au premier cas est égale à l' switch-expression alors les actions suivant le case considéré sont exécutées (action1i). Sinon, switch-expression est comparée à case-expression2 et ainsi de suite jusqu'à trouver une expression de type case-expression x qui corresponde à switch-expression. Si ça n'est jamais le cas alors ce sont les actions suivant otherwise qui sont exécutées. Si otherwise n'existe pas (on peut ne pas le mettre), on sort de la séquence et la commande située immédiatement après le end est exécutée. Un seul case-expression x est exécuté lors de toute la séquence.

End : termine une séquence (if ou switch) ou une boucle (for ou while)

Error : affiche un message (prévu par l'utilisateur). Par exemple, lors d'une séquence switch, si tous les cas échouent, on peut dans le cas d'otherwise mettre comme action le message ('le cas n'est pas prévu').

2. Boucles

On a deux types de boucles : les boucles for et les boucles while qui sont des outils bien connus utilisés dans tous les langages de programmation :

-for : permet de répéter une séquence d'actions un nombre de fois fixé par l'utilisateur.

Utilisation : for variable=deb : pas :fin,

action1,

action2,

.

.

actionn,

End

La variable est un indice de boucle qui prend la valeur deb lorsqu'on entre dans la boucle, qui s'incrémente de la valeur pas à chaque fois que la séquence d'actions action1,action2,…actionn a été effectuée. Cette séquence est réalisée le nombre de fois qu'il faut jusqu'à ce que variable atteigne la valeur fin.

-while : permet la répétition d'une séquence d'actions un nombre de fois indétermoné mais conditionné à la réalisation d'une condition particulière.

Utilisation while expression,

action1,

action2,

.

.

End

Dans expression, on peut mettre une variable ou une relation incluant un opérateur relationnel ou un opérateur logique. La séquence d'actions action1, action2, ….actionn est exécutée tant que expression est vérifiée.

Exercices avec corrections sur la gestion de cas :

Exercices :

Exercice 1 :

Donner le programme qui écrit 4fois le mot ‘bonjour’ avec la commande while sachant que la commande d’affichage est ‘disp’

Exercice 2 :

Donner le programme qui calcule la somme des 10 premiers nombres pairs

Exercice 3 :

Soit le polynôme $y=20x5+4x4+5x3+2x^2$

Donner le programme qui calcule les valeurs de y pour :

Si x>=1 y est calculé pour x=1

Si x<1 on distingue deux cas :

Si x>=sqrt(2)/2 y est calculé pour x=2

Si x<sqrt(2)/2 y est calculé pour x=3

Corrections :

Exercice 1 :

```
i=1
While i<=4,
Disp('bonjour'),
i=i+1,
end
```

Exercice 2 :

```
for i=0 :2 :10,
s=s+i,
end
```

Exercice 3 :

```
x=3(en exemple)
y=[2 0 4 5 2 0 0]
if x>=1,
y1=polyval(y,1)
elseif x>(sqrt(2)/2)
y2=polyval(y,2)
else
y3=polyval(y,3)
end
```