ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ Η/Υ ΣΕ ΠΕΡΙΒΑΛΛΟΝ MATLAB

Μαρινάκη Μαγδαληνή
(magda@dssl.tuc.gr)

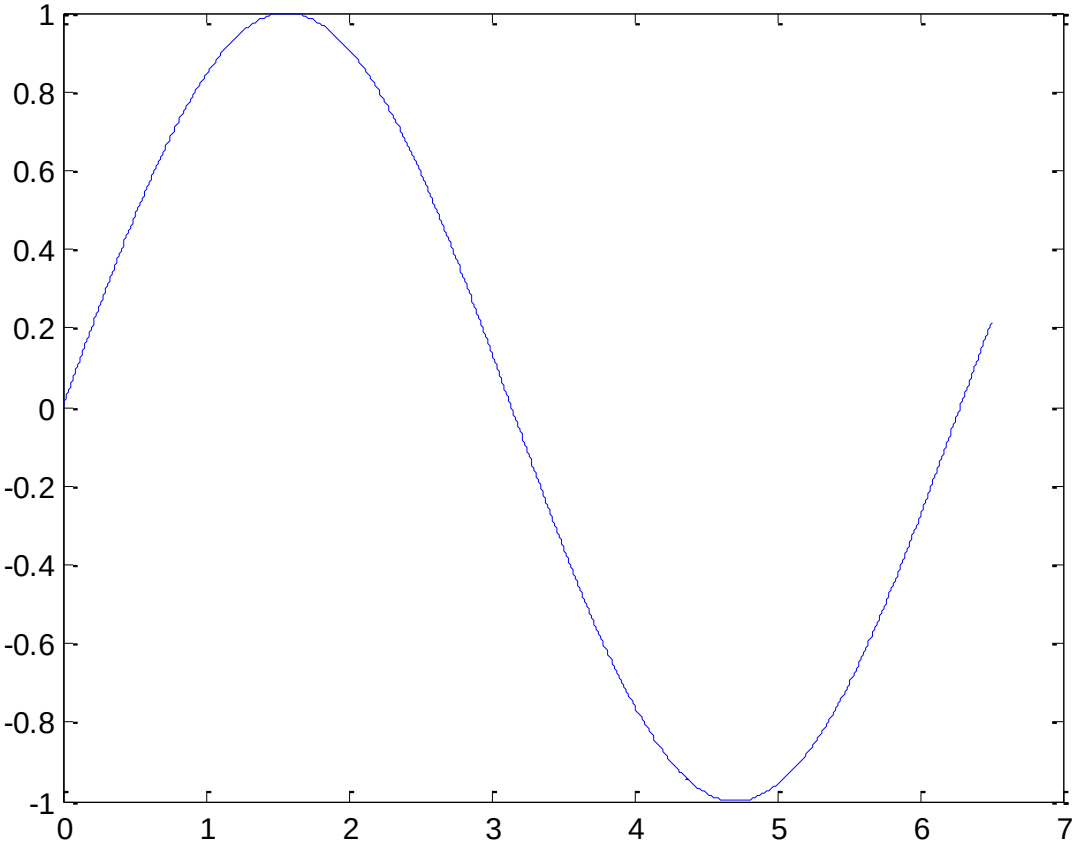
Γραφικές Παραστάσεις

- Για να **σχεδιάσουμε δεδομένα σε 2 διαστάσεις** χρησιμοποιούμε την **εντολή plot.**
- Η εντολή αυτή ανοίγει ένα **παράθυρο γραφικών που ονομάζεται Figure** και θέτει τα δεδομένα σε ένα σύστημα κάθετων αξόνων και ενώνει τα σημεία με μία γραμμή. Για παράδειγμα οι εντολές

```
>> x=0:0.005:6.5;
>> y=sin(x);
>> plot(x,y)
```

έχουν σαν αποτέλεσμα την ακόλουθη γραφική παράσταση:

Γραφικές Παραστάσεις



Γραφικές Παραστάσεις

- Για την **κατάτμηση ενός διαστήματος** μπορεί να χρησιμοποιηθεί και η **εντολή linspace( )**.
- Η συνάρτηση linspace μπορεί να πάρει **δύο ή τρεις παραμέτρους**.
- Αν χρησιμοποιηθεί με 2 παραμέτρους (π.χ. linspace(x1, x2)) δίνει ένα διάνυσμα γραμμής με 100 στοιχεία των οποίων οι τιμές είναι από x1 μέχρι x2. Για παράδειγμα, ο ορίζοντιος άξονας της προηγούμενης γραφικής παράστασης μπορεί να δημιουργηθεί και με την παρακάτω εντολή

```
>> x=linspace(0, 6.5)
```

- Αν η linspace έχει τρεις παραμέτρους, η τρίτη παράμετρος καθορίζει το πλήθος των στοιχείων του διανύσματος.

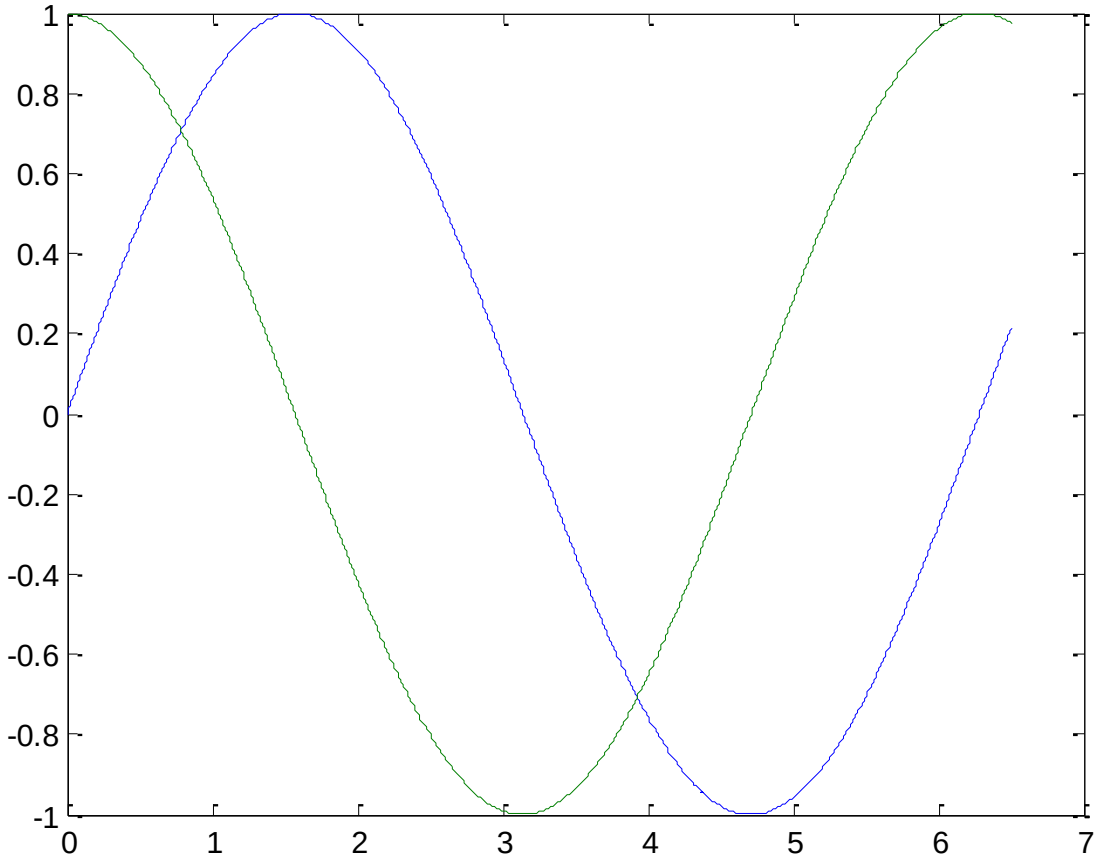
Γραφικές Παραστάσεις

- Στο **ίδιο παράθυρο μπορούμε να σχεδιάσουμε περισσότερες από μία γραφικές παραστάσεις.** Για παράδειγμα, οι εντολές

```
>> x=0:0.005:6.5;
>> y=sin(x);
>> z=cos(x);
>> plot(x,y,x,z)
```

έχουν σαν αποτέλεσμα το παρακάτω γράφημα

Γραφικές Παραστάσεις



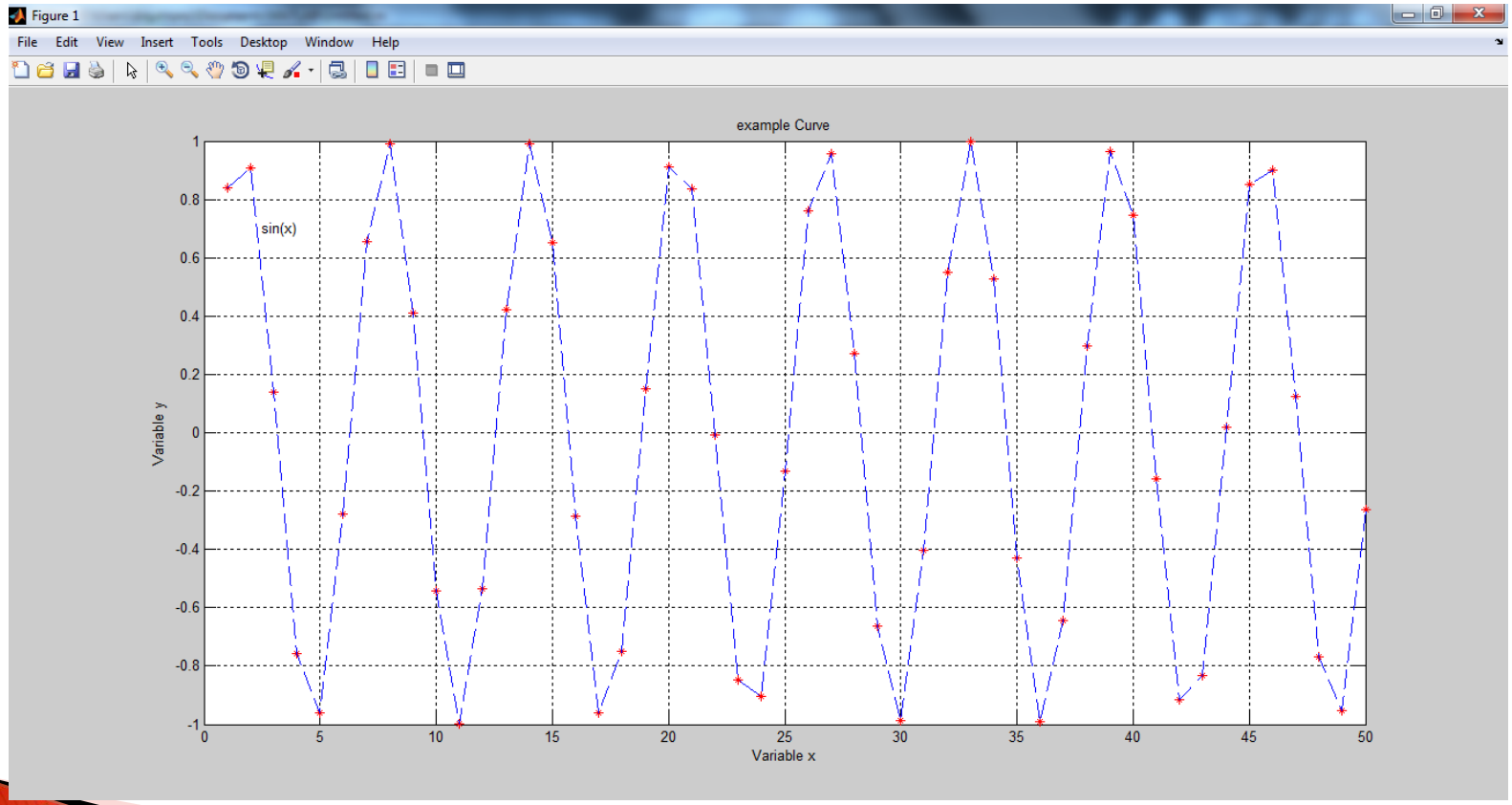
Γραφικές παραστάσεις

```
x=[1:50];
for i=1:50
    y(i)=sin(x(i));
end
for i=1:50
    plot(x(i),y(i),'*','MarkerEdgeColor','r')
    hold on
end
grid on
xlabel ('Variable x')
ylabel ('Variable y')
title ('example Curve')
text(2.5, 0.7, 'sin(x)')
plot(x,y,'--')
```

Κρατάει ότι έχουμε σχεδιάσει ως τώρα και προσθέτει ότι κάνουμε επιπλέον

Γραφικές παραστάσεις

Για να σχεδιάσουμε δεδομένα σε 2 διαστάσεις χρησιμοποιούμε την εντολή plot.



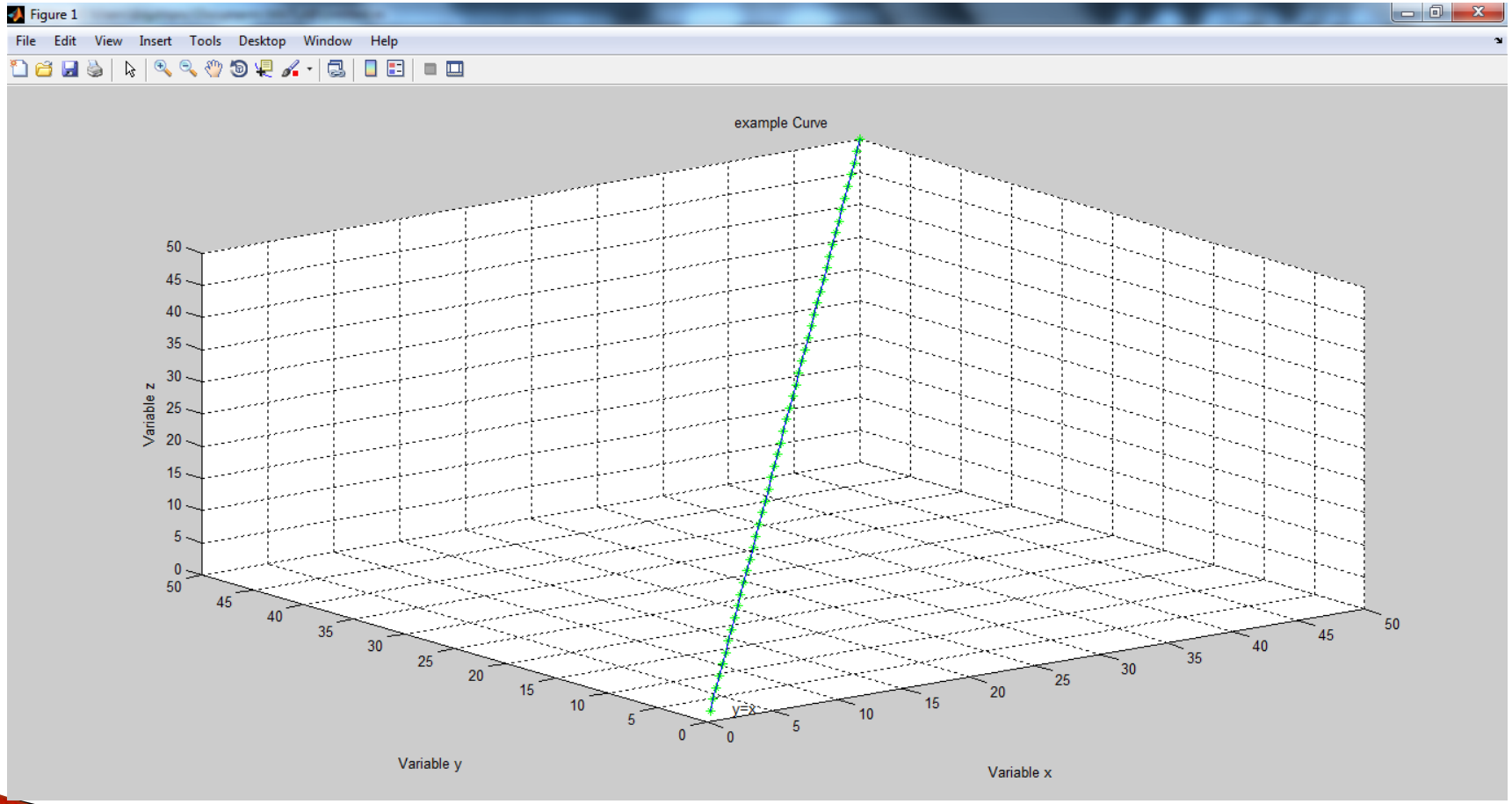
Γραφικές παραστάσεις

Σύμβολο	Χρώμα	Σύμβολο	Σημείο	Σύμβολο	Στυλ Γραμμής
b	Blue	.	Point	-	Solid line
g	Green	°	Circle	:	Doted line
r	Red	x	x-mark	-.	Dash-dot line
m	Magenta	+	Plus	--	Dashed line
y	Yellow	*	Star		
k	Black	s	Square		
w	White	d	Diamond		
c	Cyan	^	Triangle (up)		
		<	Triangle (left)		
		>	Triangle (right)		
		p	Pentagram		
		h	Hexagram		

Γραφικές παραστάσεις

```
z=[1:50];
x=[1:50];
for i=1:50
    y(i)=x(i);
end
for i=1:50
    plot3(x(i),y(i),z(i),'*','MarkerEdgeColor','g')
    hold on
end
grid on
xlabel ('Variable x')
ylabel ('Variable y')
zlabel ('Variable z')
title ('example Curve')
text(2.5, 0.7, 1, 'y=x')
plot3(x,y,z)
```

Γραφικές παραστάσεις

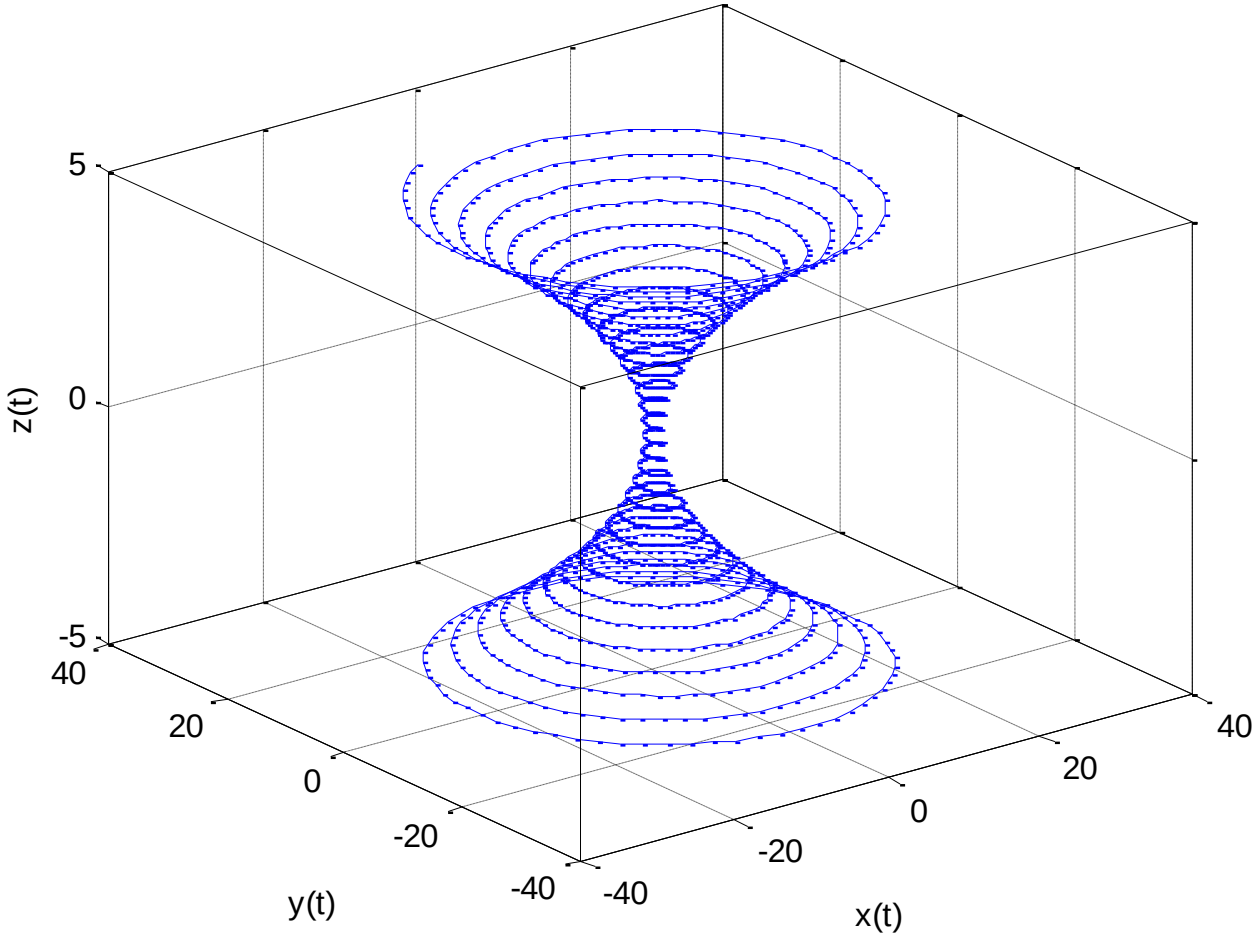


Γραφικές Παραστάσεις

```
>> t=-5:0.005:5;
>> x=(1+t.^2).*sin(20*t);
>> y=(1+t.^2).*cos(20*t);
>> z=t;
>> plot3(x,y,z)
>> xlabel('x(t)'), ylabel('y(t)'), zlabel('z(t)')
>> title('\it{plot3 example}', 'Fontsize', 14)
>> grid on
>> box on
```

Γραφικές Παραστάσεις

plot3 example



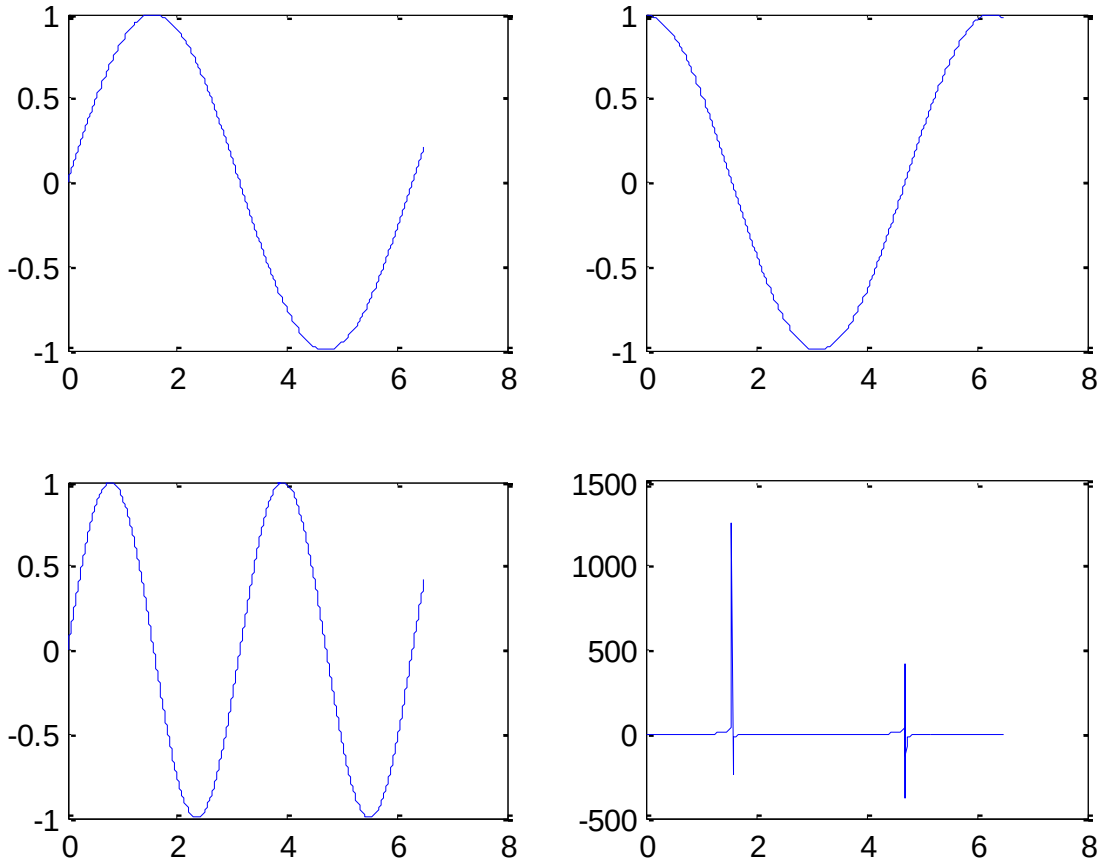
Γραφικές Παραστάσεις

- Αν θέλουμε **μπορούμε στο ίδιο παράθυρο να έχουμε πολλές γραφικές παραστάσεις σε χωριστούς άξονες**, χρησιμοποιώντας την **εντολή subplot**.
- Η **εντολή subplot(m, n, p)** διαιρεί το παράθυρο σε m x n περιοχές και επιλέγει την p-οστή σαν ενεργή.

Γραφικές Παραστάσεις

```
>> x=0:0.005:6.5;
>> y=sin(x);
>> z=cos(x);
>> subplot(2,2,1)
>> plot(x,y)
>> subplot(2,2,2)
>> plot(x,z)
>> subplot(2,2,3)
>> plot(x, 2*y.*z)
>> subplot(2,2,4)
>> plot(x,y./z)
```

Γραφικές Παραστάσεις



Γραφικές Παραστάσεις

- Για την **σχεδίαση μαθηματικών συναρτήσεων** χρησιμοποιείται η **συνάρτηση fplot**, η οποία υπολογίζει μία μαθηματική συνάρτηση σε διάφορα σημεία για να παράγει ένα αντιπροσωπευτικό γράφημα. Η συνάρτηση fplot συντάσσεται ως εξής:

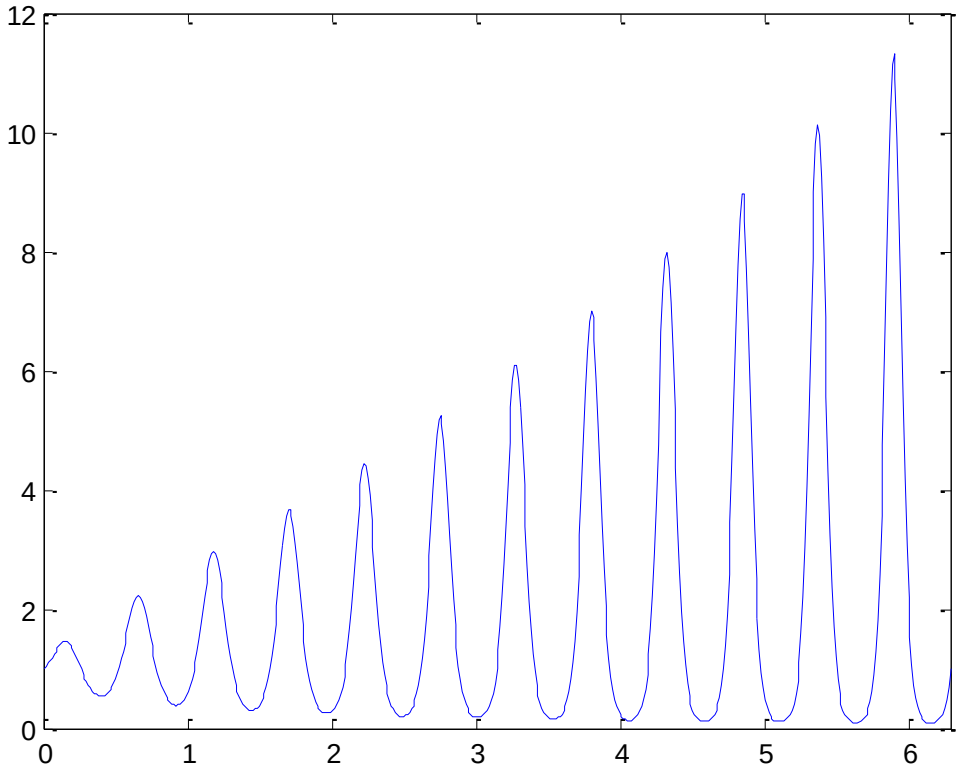
- fplot('fun', lims, tol, N, 'Line Spec', p1, p2, …)

όπου

- fun είναι η συνάρτηση που θα σχεδιαστεί
- Τα όρια του x και/ή του y δίνονται από το lims
- Το 'tol' δίνει την ανοχή του σχετικού σφάλματος (η default value είναι $2 \cdot 10^{-3}$ που αντιστοιχεί σε 0.2% ακρίβεια)
- Θα χρησιμοποιηθούν για το γράφημα τουλάχιστον N+1 σημεία
- Το 'Line Spec' καθορίζει τον τύπο της γραμμής
- p1, p2, … είναι οι παράμετροι που χρησιμοποιούνται για τα δεδομένα εισόδου.

Γραφικές Παραστάσεις

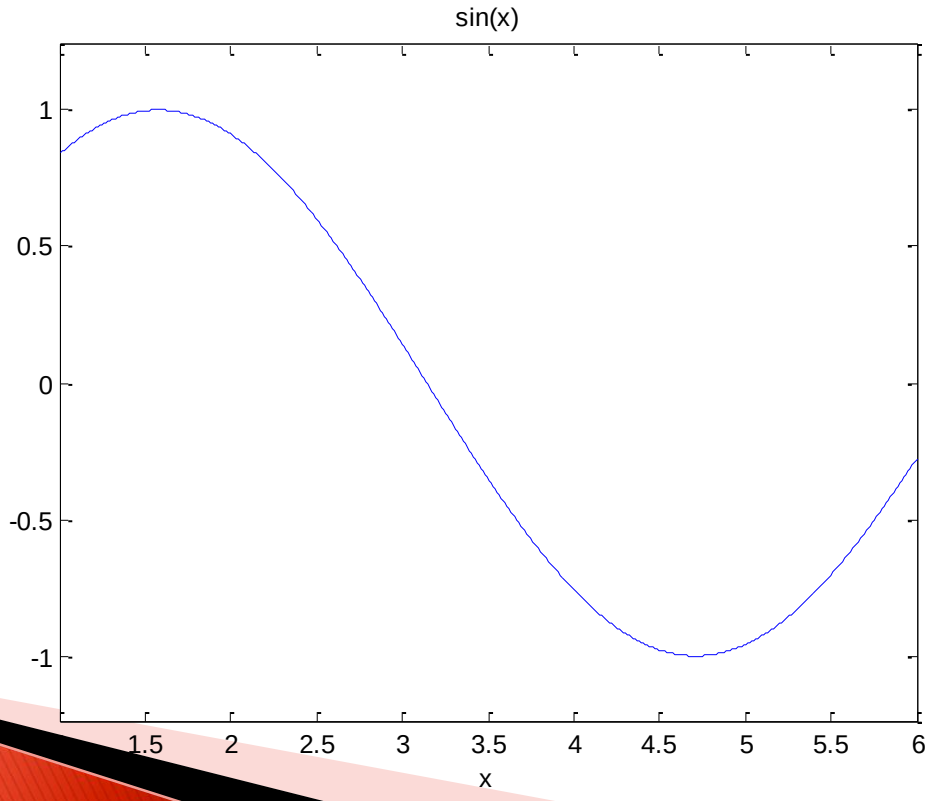
- >> fplot('exp(sqrt(x)*sin(12*x))',[0 2*pi])



Γραφικές Παραστάσεις

- Μία **εύκολη στη χρήση μορφή του fplot είναι η ezplot**, η οποία συντάσσεται ως εξής: **ezplot(f, [xmin, xmax])**.

```
>> ezplot('sin(x)',[1,6])
```



M-Files

- Το Matlab παρέχει τη δυνατότητα κλήσης συναρτήσεων και αρχείων εντολών που κατασκευάζονται από το χρήστη.
- Τα αρχεία αυτά (script M-files και function M-files) είναι αρχεία κειμένου που περιέχουν κώδικα Matlab.

- ***Script M-Files (ή command files)***

- Τα αρχεία αυτά δεν έχουν ορίσματα (μεταβλητές εισόδου και εξόδου), χρησιμοποιούνται για την αυτόματη εκτέλεση εργασιών και λειτουργούν σε μεταβλητές του χώρου εργασίας του Matlab ή φτιάχνουν δικές τους μεταβλητές οι οποίες παραμένουν ενεργές στο τρέχον Workspace και μετά την εκτέλεση τους.

Script M-Files

- Χρησιμοποιούνται για την αυτόματη εκτέλεση εργασιών.
- Για να ξεκινήσουμε ένα M-File από το μενού File επιλέγουμε New->M-file.
- Για παράδειγμα ανοίγουμε ένα M-File και γράφουμε τις ακόλουθες εντολές:

```
a=2;
b=5;
A=(2*a^2-3*a*b+b^2)/(a^2+b^2)
```

Στη συνέχεια επιλέγουμε File->Save as και στο όνομα αρχείου πληκτρολογούμε το όνομα example και κλείνουμε το παράθυρο του M-File.

Script M-Files

- Για να εκτελέσουμε το M-File μας πληκτρολογούμε στην προτροπή του Command Window (>>) τη λέξη example.

- Πατάμε Enter και το αποτέλεσμα που θα εμφανιστεί στο Command Window θα είναι:

```
>>example

A=

   0.1034
```

Script M-Files

Αν θέλουμε να υπολογίσουμε την τιμή του A για διάφορες τιμές του a και b μπορούμε να αντικαταστήσουμε τις εντολές a=2 και b=5 με τις εντολές:

```
a=input('Enter the number a')
b=input('Enter the number b')
```

Αν τώρα καλέσουμε το example στην οθόνη εμφανίζεται το μήνυμα «Enter the number a» και ο κέρσορας αναβοσβήνει δίπλα περιμένοντας να δώσουμε κάποιο αριθμό. Το ίδιο και για το b. Αφού δοθούν τιμές και για τις 2 μεταβλητές υπολογίζετε το A.

Script M-Files

- Τα a και b του αρχείου example μπορούν να διαβαστούν και από ένα αρχείο excel.

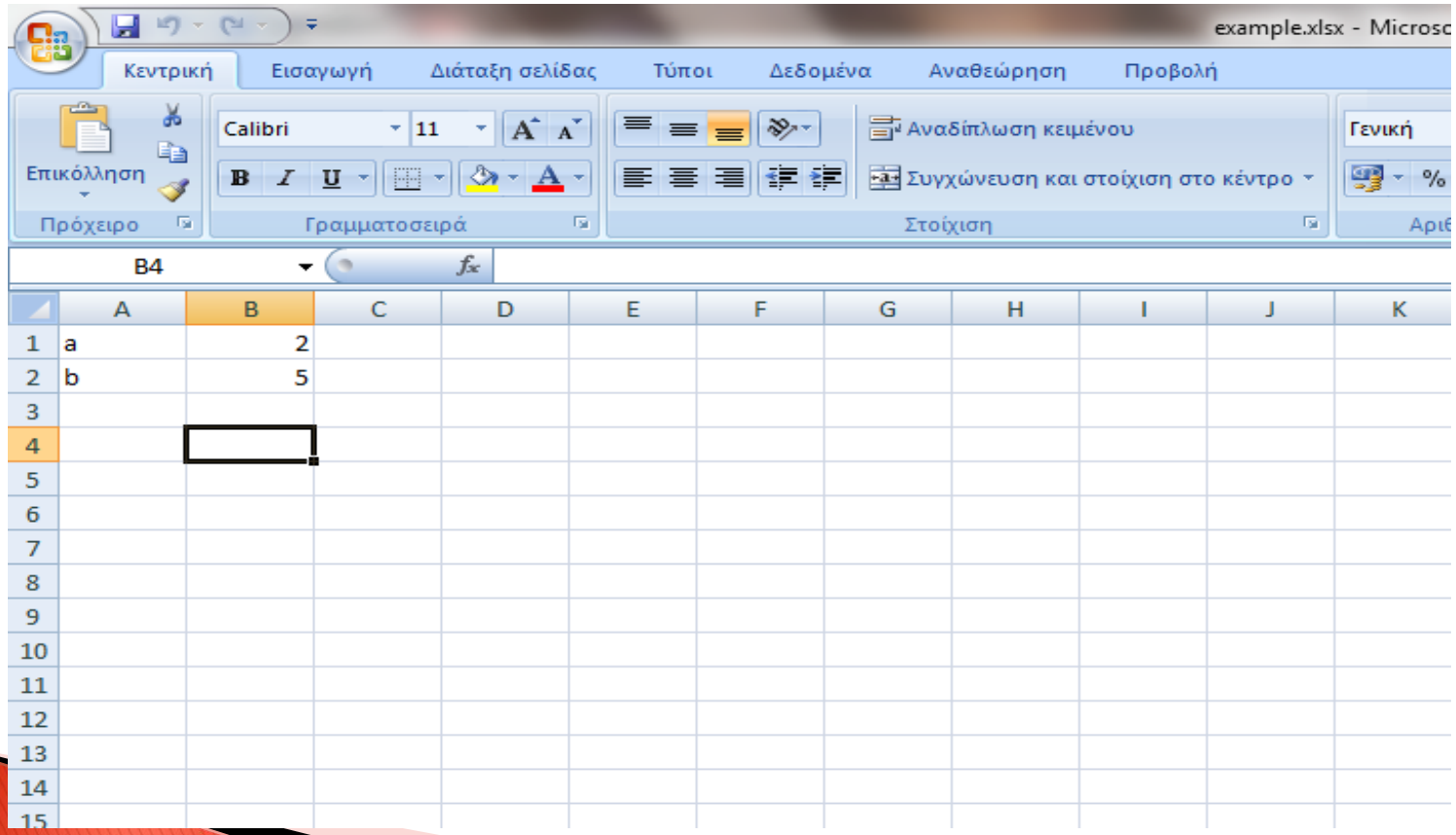
- Για να διαβάσουμε ένα αρχείο excel μέσα από ένα M-File χρησιμοποιούμε την εντολή:

```
>>N = xlsread('filename', sheet, 'range')
```

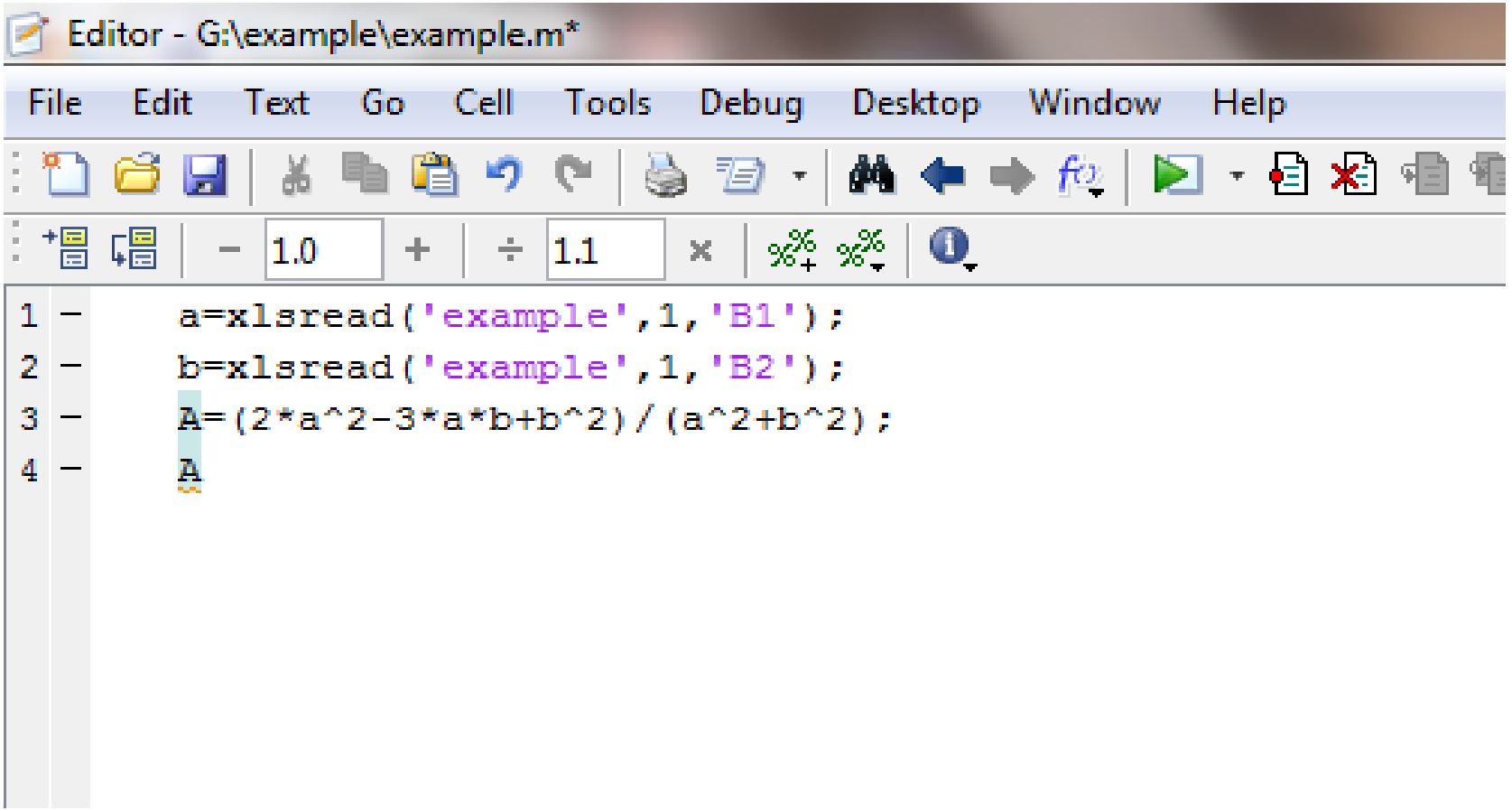
Όπου αποθηκεύεται στην μεταβλητή N το περιεχόμενο του xls αρχείου με όνομα “filename”, που είναι περασμένο στο φύλλο “sheet” και στα κελία με εύρος “range”.

Script M-Files

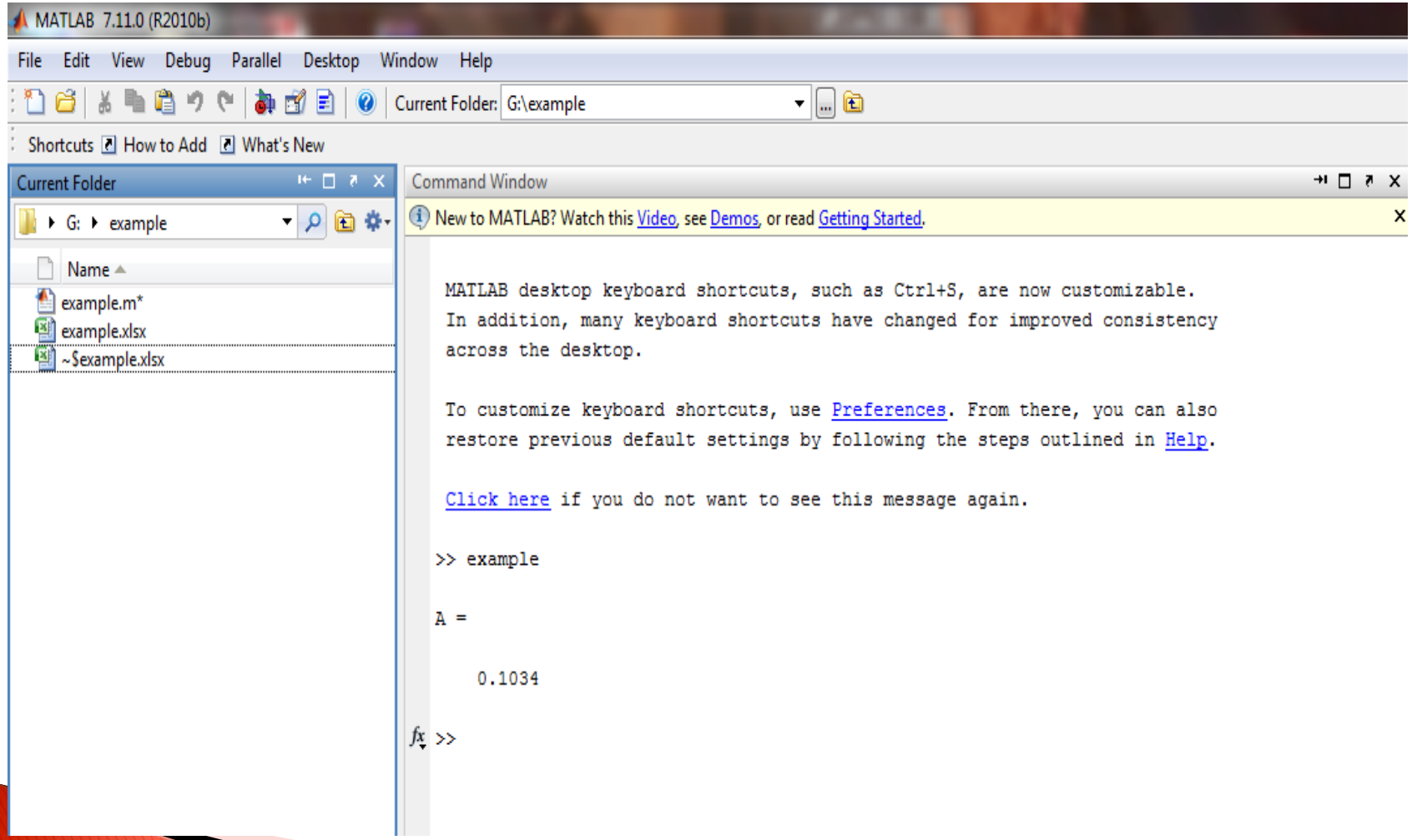
Παράδειγμα για διάβασμα xls αρχείου στο M-File "example.m"



Script M-Files



Script M-Files



Function M-Files

- Περιέχουν μία γραμμή καθορισμού μίας συνάρτησης
- Δέχονται μεταβλητές εισόδου (ορίσματα) και επιστρέφουν μεταβλητές εξόδου.
- Οι εσωτερικές μεταβλητές είναι τοπικές στη συνάρτηση.
- Έχουν τη μορφή:

function μεταβλητές εξόδου=όνομα συνάρτησης (μεταβλητές εισόδου)

- Το Matlab παρέχει έτοιμες συναρτήσεις
- Ας γράψουμε ένα function M-file που υπολογίζει το μέγιστο σε απόλυτη τιμή ενός πίνακα, π.χ. την "maxentry" που υπολογίζει το μέγιστο σε απόλυτη τιμή ενός πίνακα.

```
function y=maxentry(A)
%MAXENTRY   Largest absolute value of matrix entries.
y=max(max(abs(A)));
```

```
>> maxentry(1:10)

ans =

    10
```

Προγραμματισμός

Γενικά

Το 70% του χρόνου που απαιτείται για την δημιουργία μιας εφαρμογής, ξοδεύεται για την διόρθωση των λογικών και άλλων σφαλμάτων που υπάρχουν στη δομή του προγράμματος.

Η δημιουργία μιας εφαρμογής περνάει από τα παρακάτω στάδια:

1. Κατάστρωση προβλήματος.

2. Ανάπτυξη ή επιλογή αλγορίθμου.

3. Μετατροπή του αλγορίθμου σε πρόγραμμα.

4. Αποσφαλμάτωση-επαλήθευση.

Προγραμματισμός

Λογικοί και Σχεσιακοί Τελεστές

Σχεσιακοί Τελεστές

<	Μικρότερο
<=	Μικρότερο ή Ίσο
>	Μεγαλύτερο
>=	Μεγαλύτερο ή Ίσο
==	Ίσο
~=	Διάφορο

True=1

False=0

```
>> A=1:10
A =
    1    2    3    4    5    6    7    8    9   10
>> C=A>4
C =
    0    0    0    0    1    1    1    1    1    1
```

Λογικοί Τελεστές

&	AND
|	OR
~	NOT

Τι τιμές θα έχει το D;

```
>> A=1:10
A =
    1    2    3    4    5    6    7    8    9   10

>> D=(A>2)&(A<8)
D =  0    0    1    1    1    1    1    0    0    0
```

Προγραμματισμός

Εντολές Ελέγχου & Επανάληψης

Βρόχος for

```
for x=array
  Εντολές
end
```

Παράδειγμα

```
for n=1:10
  x(n)=sin(n*pi/10);
end
```

Nested Loops

Χρησιμοποιείται για να προσπελάσουμε έναν πολυδιάστατο πίνακα.

Παράδειγμα

```
n=2;
A=[0 0;0 0];
for i=1:n
   for j=1:n
      A(i,j)=1;
   end
end
```

Προγραμματισμός

Εντολές Ελέγχου & Επανάληψης (2)

Εντολή if

```
if συνθήκη
  εντολές
End
```

Παράδειγμα

```
x=20;
cost=x*100;
if cost>1000
  cost=0.8*cost;
end
```

Εντολή if-else-end

```
if συνθήκη
   εντολές
else
   εντολές
End
```

Παράδειγμα

```
X=20;
if x<20
   cost=x*100;
else
   cost=x*80;
end
```

Προγραμματισμός

Εντολές Ελέγχου & Επανάληψης (3)

Εντολή if-elseif-end

```
if συνθήκη1
   Εντολές1
elseif συνθήκη2
   Εντολές2
elseif συνθήκη3
   Εντολές3
   .
   .
   .
else
   Εντολές
end
```

Βρόχος while

Προσοχή: *Δεν γνωρίζουμε ακριβώς πόσες φορές θα γίνει η επανάληψη.*

```
while έκφραση
   Εντολές
End
```

Παράδειγμα

```
x=0;
y=5;
While y~=0
   x=x+1;
   y=y-1;
end
```

ΣΥΜΒΟΛΙΚΕΣ ΜΕΤΑΒΛΗΤΕΣ

- Το Matlab μας επιτρέπει να χρησιμοποιούμε μεταβλητές σαν σύμβολα, χωρίς να δίνουμε συγκεκριμένες αριθμητικές τιμές. Για να ορίσουμε μία συμβολική μεταβλητή χρησιμοποιούμε την εντολή
- >>x=sym('x')

ή

- >> syms x, y, w

- Η εισαγωγή συμβολικών μεταβλητών μας επιτρέπει να λαμβάνουμε ως αποτέλεσμα μαθηματικές εκφράσεις αντί απλών αριθμών.

ΣΥΜΒΟΛΙΚΕΣ ΜΕΤΑΒΛΗΤΕΣ

- Έτσι για παράδειγμα μπορούμε να βρούμε την λύση της $y=ax^2+bx+c$ χρησιμοποιώντας την εντολή solve( ) ως εξής:

- >> syms a b c x
- >> y=solve(a*x^2+b*x+c)

- Θα πάρουμε

- y =

1/2/a*(-b+(b^2-4*a*c)^(1/2))
1/2/a*(-b-(b^2-4*a*c)^(1/2))

- Καθώς δεν εξισώσαμε την εξίσωση με κάποιο αριθμό, το Matlab θεωρεί ότι είναι ίση με το 0. Οπότε η προηγούμενη εντολή είναι ισοδύναμη με:

- >> y=solve('a*x^2+b*x+c=0')

ΣΥΜΒΟΛΙΚΕΣ ΜΕΤΑΒΛΗΤΕΣ

- Επίσης, καθώς δεν καθορίσαμε κάποιο όρισμα σαν άγνωστο το Matlab εφαρμόζει την συνάρτηση findsym στην έκφραση a*x^2+b*x+c για να καθορίσει την μεταβλητή που είναι πιο κοντά αλφαβητικά στο x και λύνει για αυτή την μεταβλητή σαν άγνωστο. Έτσι, η προηγούμενη εντολή είναι ισοδύναμη με:

- >> y=solve(a*x^2+b*x+c,x)

- Φυσικά μπορούμε να λύσουμε την εξίσωση αυτή ως προς a, ως εξής:

- >> y=solve('a*x^2+b*x+c=0',a)

- οπότε παίρνουμε

- y =

- -(b*x+c)/x^2
- Αν θέλουμε να υπολογίσουμε την τιμή της y για αριθμητικές τιμές των παραμέτρων μπορούμε να χρησιμοποιήσουμε την συνάρτηση subs( ). Έτσι, για το προηγούμενο παράδειγμα έστω:

- >> x=1;b=-1;c=-1;
- >> subs(y)

- ans =

  2

ΕΙΣΟΔΟΣ – ΕΞΟΔΟΣ

- Με την συνάρτηση **input** μπορούμε να αποκτήσουμε τα δεδομένα εισόδου από τον χρήστη. Η εντολή input περιμένει την απάντηση του χρήστη. Για παράδειγμα, αν γράψουμε:
- >> x=input('starting point')

- εμφανίζεται στην κάτω γραμμή:

- starting point

και κέρσορας αναβοσβήνει, περιμένοντας από τον χρήστη να δώσει κάποια τιμή. Μόλις ο χρήστης δώσει την τιμή, το x παίρνει αυτή την τιμή:

x =

2

ΕΙΣΟΔΟΣ – ΕΞΟΔΟΣ

- Το input μεταφράζεται σε string όταν προστίθεται το όρισμα ‘s’ στην εντολή. Π.χ.

```
>> mytitle=input('Title for the work','s')
Title for the work  Example 1
```

- mytitle =

  Example 1

- Η εντολή disp εμφανίζει την τιμή μίας μεταβλητής βάση της τρέχουσας μορφοποίησης. Για παράδειγμα:

- >> disp('Here is a 3-by-3 identity matrix'), eye(3)
- Here is a 3-by-3 identity matrix

```
ans =

     1     0     0
     0     1     0
     0     0     1
```

ΕΙΣΟΔΟΣ – ΕΞΟΔΟΣ

- Πιο περίπλοκη μορφοποίηση μπορεί να δοθεί με την χρήση της συνάρτησης **fprintf**, η οποία συντάσσεται ως εξής: **fprintf(format, list-of-expressions),** όπου format είναι ένα string που καθορίζει την ακριβής μορφοποίηση για κάθε στοιχείο που θα τυπωθεί. Π.χ.:

```
>> fprintf('%6.3f\n',pi)
3.142
>> fprintf('%6.4f\n',pi)
3.1416
>> fprintf('%10.8f\n',pi)
3.14159265
```

όπου το σύμβολο % δηλώνει την έναρξη της μορφοποίησης η οποία δημιουργεί ένα πεδίο με 6 ψηφία τα 3 από τα οποία θα είναι μετά την υποδιαστολή και το '\n' συμβολίζει την νέα γραμμή. Το f χρησιμοποιείται στην μορφοποίηση πραγματικών αριθμών.

ΕΙΣΟΔΟΣ – ΕΞΟΔΟΣ

- Αν το καθορισμένο πεδίο δεν είναι αρκετά μεγάλο το Matlab το επεκτείνει όσο πρέπει, π.χ.

```
>> fprintf('%6.3f\n',pi^20)
8769956796.083
```

- Μπορούμε επίσης αντί του f να χρησιμοποιήσουμε το e στην μορφοποίηση των πραγματικών αριθμών, οπότε παίρνουμε:

```
>> fprintf('%6.3e\n',pi^20)
8.770e+009
```

- **Σημείωση**: Σε ένα αρνητικό αριθμό το σύμβολο – καταλαμβάνει μία θέση. Έτσι:

```
>> fprintf('%5.2f\n%5.2f\n',exp(1),-exp(1))
2.72
-2.72
```

- Αν χρησιμοποιήσουμε το σύμβολο – μετά το % τότε το πεδίο στοιχίζεται αριστερά. Π.χ.

```
>> fprintf('%-5.2f\n%-5.2f\n',exp(1),-exp(1))
2.72
-2.72
```

ΕΙΣΟΔΟΣ – ΕΞΟΔΟΣ

- Επίσης στο πεδίο της εντολής format μπορεί να υπάρχει και κείμενο που θα γραφεί όπως δίνεται. Π.χ.

```
>> iter=10;
>> fprintf('iter = %2.0f\n',iter)
iter = 10
```

- Αντί του f και του e μπορούμε να χρησιμοποιήσουμε το g το οποίο χρησιμοποιεί τη μορφοποίηση που προκύπτει είτε με την χρήση του f είτε του e ανάλογα με το πιο παράγει το μικρότερο (όσο αφορά το μήκος του πεδίου) αποτέλεσμα. Π.χ.

```
>> fprintf('%g %g\n', exp(1), exp(20))
2.71828 4.85165e+008
```

- Η συνάρτηση sprintf είναι ανάλογη της fprintf αλλά η έξοδος της είναι string. Π.χ.

```
>> n=4;
>> err_msg=sprintf('must supply a %d-by-%d matrix',n,n)

err_msg =
must supply a 4-by-4 matrix
```

ΕΙΣΟΔΟΣ – ΕΞΟΔΟΣ

- Αν θέλουμε να γράψουμε και να διαβάσουμε από αρχεία πρέπει πρώτα να τα ανοίξουμε χρησιμοποιώντας την συνάρτηση **fopen** της οποίας τα ορίσματα είναι το όνομα του αρχείου και ένα όρισμα το οποίο δείχνει τις επιτρεπόμενες ενέργειες που μπορούν να γίνουν σε αυτό το αρχείο (μεταξύ αυτών των ορισμάτων είναι τα 'r' για διάβασμα από ένα αρχείο και 'w' για γράψιμο δεδομένων σε ένα αρχείο).

```
>> A=[30 40 60 70];
>> fid=fopen('myoutput','w');
>> fprintf(fid,'%g miles/hour= %g kilometers/hour\n', [A;8*A/5]);
>> fclose(fid);
```

δημιουργεί το αρχείο myoutput που περιέχει

```
30 miles/hour= 48 kilometers/hour
40 miles/hour= 64 kilometers/hour
60 miles/hour= 96 kilometers/hour
70 miles/hour= 112 kilometers/hour
```

ΕΙΣΟΔΟΣ – ΕΞΟΔΟΣ

- Με την εντολή **type** μπορούμε να τυπώσουμε το αρχείο στην οθόνη. Έτσι:

```
>> type myoutput

30 miles/hour= 48 kilometers/hour
40 miles/hour= 64 kilometers/hour
60 miles/hour= 96 kilometers/hour
70 miles/hour= 112 kilometers/hour
```

- Αυτό το αρχείο μπορούμε να το διαβάσουμε ως εξής:

```
>> fid=fopen('myoutput','r');
>> X=fscanf(fid,'%g miles/hour= %g kilometers/hour')

X =
    30
    48
    40
    64
    60
    96
    70
   112

>> fclose(fid);
```

ΕΙΣΟΔΟΣ – ΕΞΟΔΟΣ

- Η συνάρτηση fscanf στην παραπάνω εντολή διαβάζει ένα πραγματικό αριθμό (%g), παραλείπει το string ‘miles/hour=’, διαβάζει ένα άλλο πραγματικό αριθμό (%g) και παραλείπει το string ‘kilometers/hour’.

- Μπορούμε να μορφοποιήσουμε το διάνυσμα στην αρχική του μορφή ως εξής:

- >> X=reshape(X,4,2)

X =

30 60
48 96
40 70
64 112