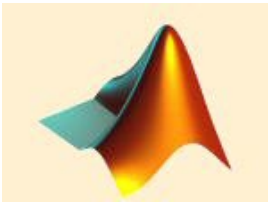
Εισαγωγή στην Επιστήμη των Η/Υ ΙΙ

Μάθημα 4

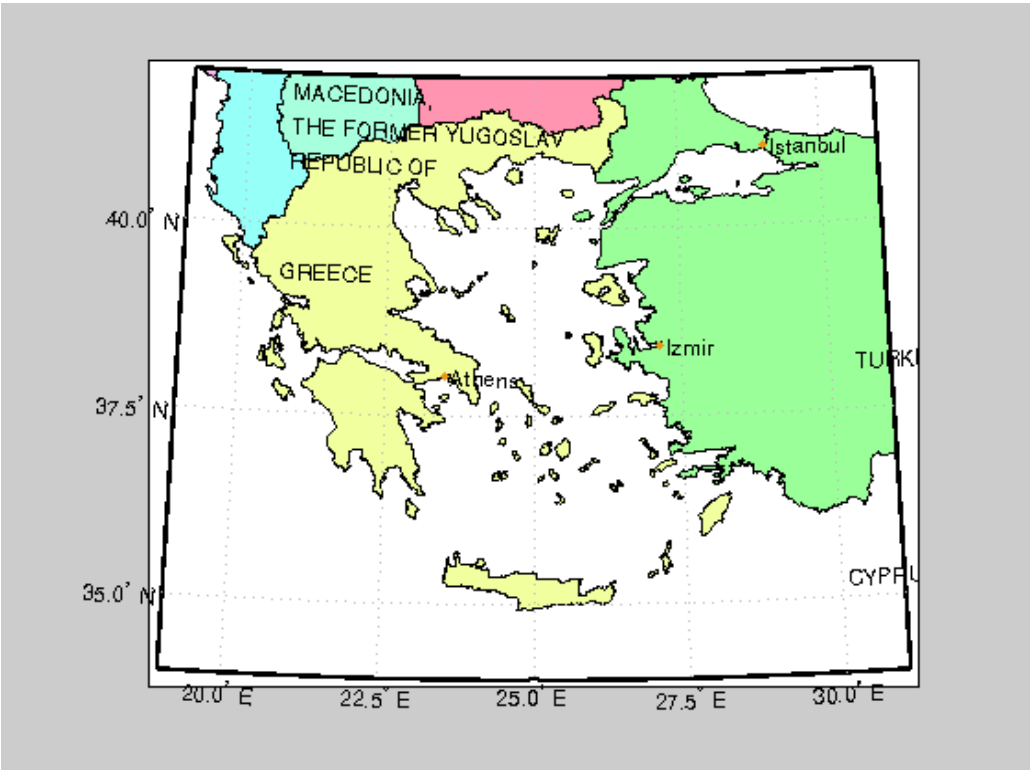


Σημειώσεις

1. Για να δημιουργήσουμε το χάρτη χρησιμοποιούμε την εντολή

>> worldmap('greece')

ή την εντολή >> **worldmap([34 42],[19 31],'patch')** όπου τα νούμερα αντιστοιχούν στις συντεταγμένες του χάρτη (Γεωγρ.πλάτος, μήκος). Με τον ίδιο τρόπο δημιουργούμε χάρτες για διάφορες χώρες ή ευρύτερες περιοχές. Για να δούμε τα ονόματα των χωρών-περιοχών που μπορούν να χρησιμοποιηθούν με την εντολή worldmap δίνουμε απλά >> **worldmap.** Τέλος επιλέγουμε Copy Figure από το μενού Edit της εικόνας και μετά Paste στο Word.



2. Για να προσθέσουμε την θέση κάποιας πόλης πρέπει να γνωρίζουμε τις συντεταγμένες της (είτε να της έχουμε από κάποιο άλλο πρόγραμμα είτε να τις βρούμε με τη Matlab). Για να βρούμε τις συντεταγμένες ενός σημείου στο χάρτη δίνουμε την εντολή inputm, **([lat,long] = inputm(npts), όπου npts ο αριθμός των σημείων που θέλουμε να ψηφιοποιήσουμε)**. Για παράδειγμα με την εντολή ([lat,long] = inputm(2) ψηφιοποιούμε τις θέσεις δύο πόλεων (ο χάρτη πρέπει να έχει ήδη δημιουργηθεί και να είναι ανοιχτός). Δίνοντας για παράδειγμα την εντολή [lat,long]=inputm(4) η Matlab αλλάζει τον κέρσορα σε σταυρόνημα και εμφανίζει τον χάρτη για ψηφιοποίηση, όταν κάνουμε κλικ στα σημεία που θέλουμε οι μεταβλητές lat, long παίρνουν τις τιμές των συντεταγμένων των σημείων (Γεωγραφικό πλάτος, μήκος). Παράδειγμα: για τις πόλεις Πάτρα, Θεσσαλονίκη, Βόλος, Ηράκλειο:

   lat =

   38.1814

   40.5910

   39.2806

   35.3401

   long =

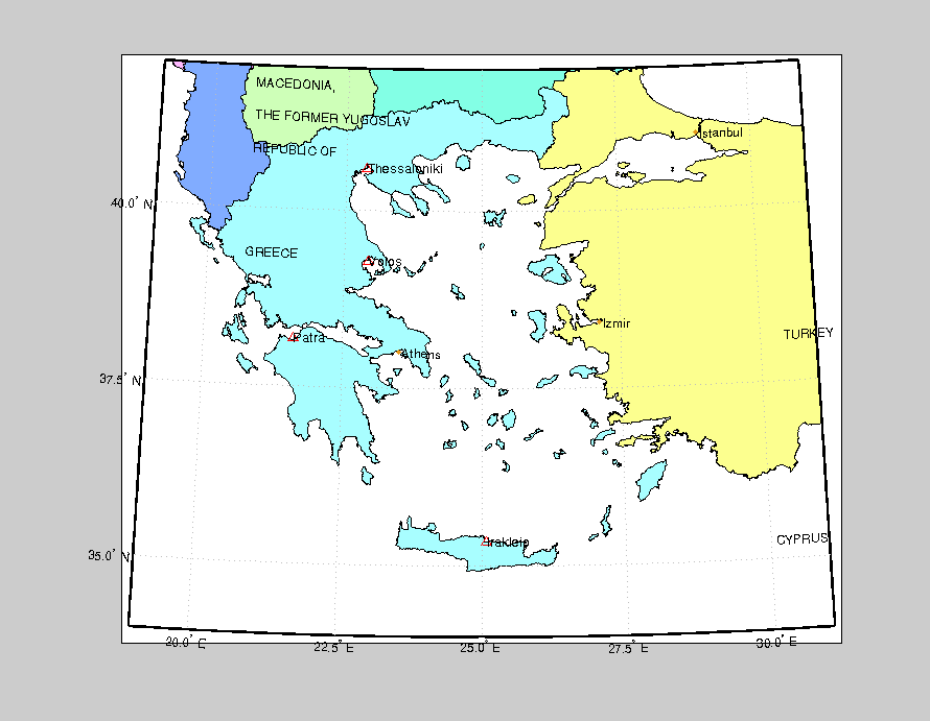
   21.6130

   22.8732

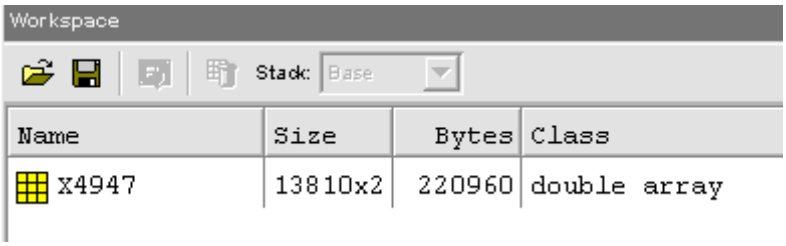
   22.9433

   25.0744

   Τέλος με την εντολή **plotm** μπορούμε να βάλουμε στο χάρτη τη θέση των πόλεων. *(Η plotm δουλεύει όπως η plot αλλά σε χάρτες)*. Παράδειγμα: plotm(lat,long,'^r') θα βάλει πάνω στο χάρτη τις θέσεις των πόλεων χρησιμοποιώντας τρίγωνα με κόκκινο χρώμα. Αν θέλουμε να βάλουμε και το όνομα της πόλης αρχικά δημιουργούμε ένα πίνακα κειμένου με τα ονόματα,
   towns=['Patra '; 'Thessaloniki' ;'Volos ';'Irakleio '];
   και με την εντολή >> textm(lat,long,towns) προσθέτουμε τα ονόματα στο χάρτη.

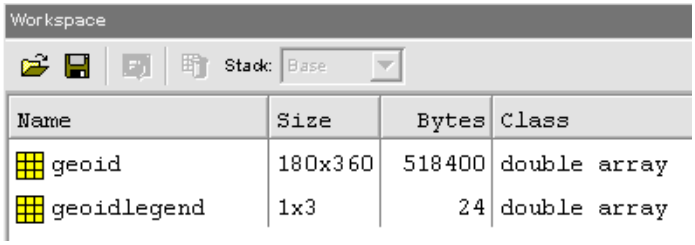


3. Αφού κατεβάσουμε τα δεδομένα (π.χ. αρχείο **4947.dat**) φορτώνουμε το αρχείο στη Matlab με την εντολή load π.χ. **load 4947.dat** και δημιουργείται η μεταβλητή X4947 με τα δεδομένα (2 στήλες).



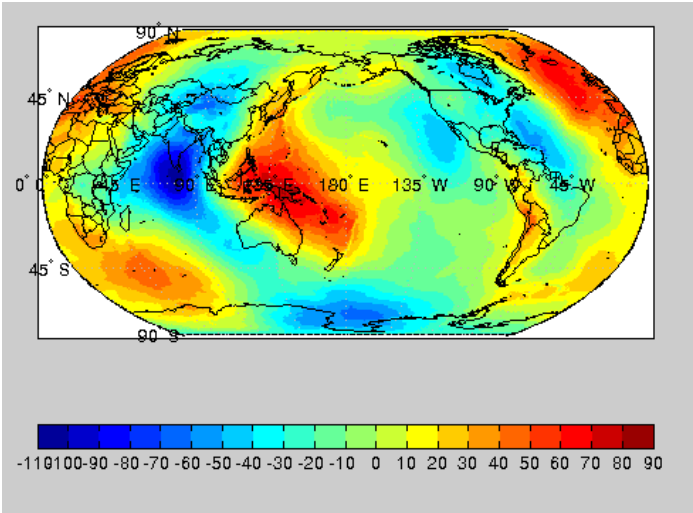
Στη συνέχεια α) επιλέγουμε την προβολή που θα χρησιμοποιήσουμε για να δημιουργήσουμε το χάρτη (για να βρούμε τις διαθέσιμες προβολές δίνουμε >> help mapproj) β) με την εντολή axesm δημιουργούμε την εικόνα με την προβολή που θέλουμε **>> axesm miller** και γ) δημιουργούμε το χάρτη με την εντολή plotm **>> plotm(X4947(:,2),X4947(:,1))**

4. Φορτώνουμε τα δεδομένα geoid με την εντολή load geoid. Το αρχείο geoid περιέχει τη μορφή του γεωειδούς (μια επιφάνεια, στην οποία η βαρυτική έλξη της γης είναι παντού η ίδια). Καλύπτει όλη τη γη, απομακρύνεται δε από την πραγματική επιφάνεια όταν είναι πάνω από περιοχές με μεγαλύτερη πυκνότητα και επομένως ισχυρότερη βαρύτητα. Πάνω από τις λιγότερο πυκνές περιοχές, το γεωειδές κινείται πιο κοντά προς την πραγματική επιφάνεια. Όταν φορτωθεί το αρχείο στη Matlab δημιουργείται ένας πίνακας με διαστάσεις 180x360 δηλαδή το πλεγματικό αρχείο geoid περιέχει δεδομένα ανά 1 μοίρα για όλη τη Γη. Για μεγαλύτερη ακρίβεια είναι δυνατόν να χρησιμοποιήσουμε και την εντολή egm96geoid η οποία θα φορτώσει τα ίδια δεδομένα αλλά από ένα κάναβο 15 λεπτών της μοίρας (για περισσότερες πληροφορίες : http://cddisa.gsfc.nasa.gov/926/egm96/egm96.html)

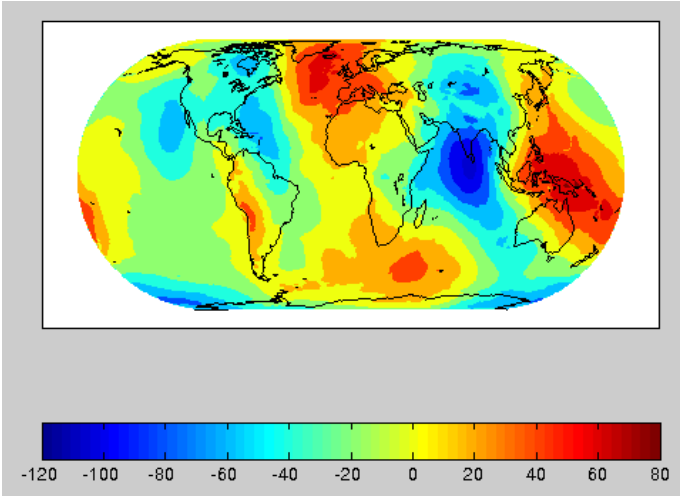


Αφού φορτώσουμε τα δεδομένα στη Matlab, έχουμε διάφορες επιλογές για να δημιουργήσουμε το χάρτη ισοκαμπυλών:

- Με την εντολή worldmap (map, maplegend, 'TYPE') **worldmap(geoid,geoidlegend)**, η μεταβλητή 'TYPE' επιλέγει τον τύπο χάρτη που θα δημιουργηθεί. Οι διαθέσιμες επιλογές είναι patch, line, patchonly, lineonly, none, mesh, meshonly, mesh3d, dem, demonly, dem3d, dem3donly, ldem3d, ldem3donly, lmesh3d, lmesh3donly. Για να προσθέσουμε μια χρωματική κλίμακα χρησιμοποιούμε την εντολή contourcmap π.χ. contourcmap(10,'jet','colorbar','on','location','horizontal').



- Με την εντολή **contourfm(map,maplegend)**. Αρχικά επιλέγουμε την προβολή που θα χρησιμοποιήσουμε **figure; axesm eckert4**. Στη συνέχεια προσθέτουμε τις ακτογραμμές στο χάρτη **plotm(coast,'k')** και τέλος δημιουργούμε το χάρτη ισοκαμπυλών με την εντολή **contourfm(geoid,geoidlegend,-120:20:100)** και την χρωματική κλίμακα με την εντολή **colorbar('horiz')**. Η επιλογή της χρωματικής κλίμακας γίνεται με την εντολή colormap π.χ. colormap(jet).



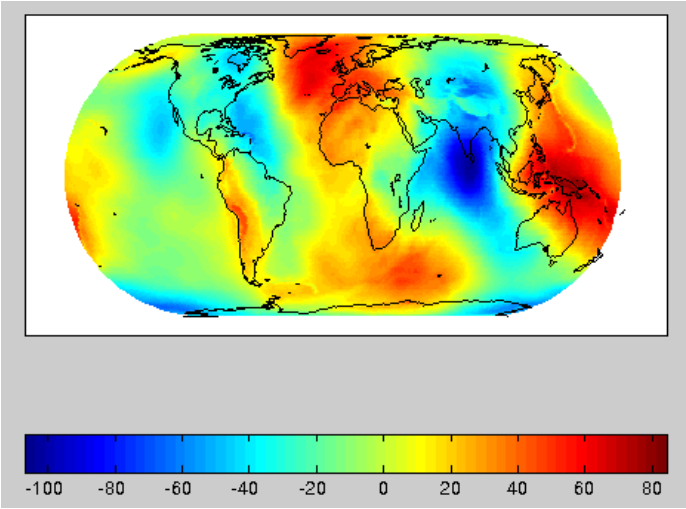
- Με τις εντολές meshm, surfm

  >> axesm eckert4

  >> meshm(geoid,geoidlegend)

  >> plotm(coast,'k')

  >> colorbar('horiz')



  >> figure

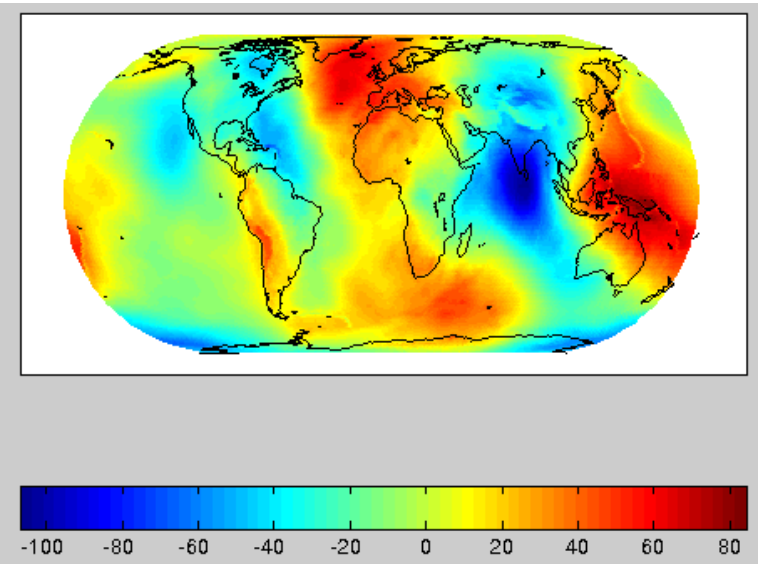
  >> axesm eckert4

  >> plotm(coast,'k')

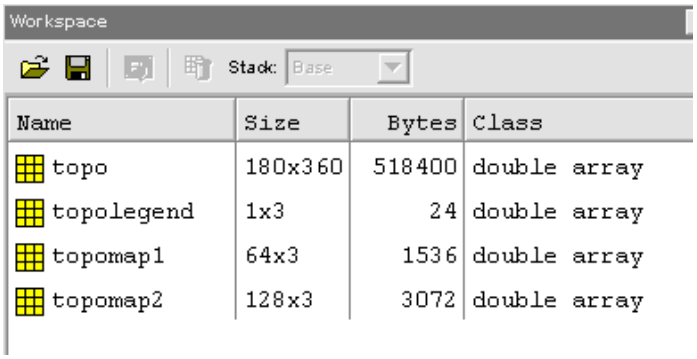
  >> [meshlat,meshlon] = meshgrat(geoid,geoidlegend,[90 180]);

  >> surfm(meshlat,meshlon,geoid)

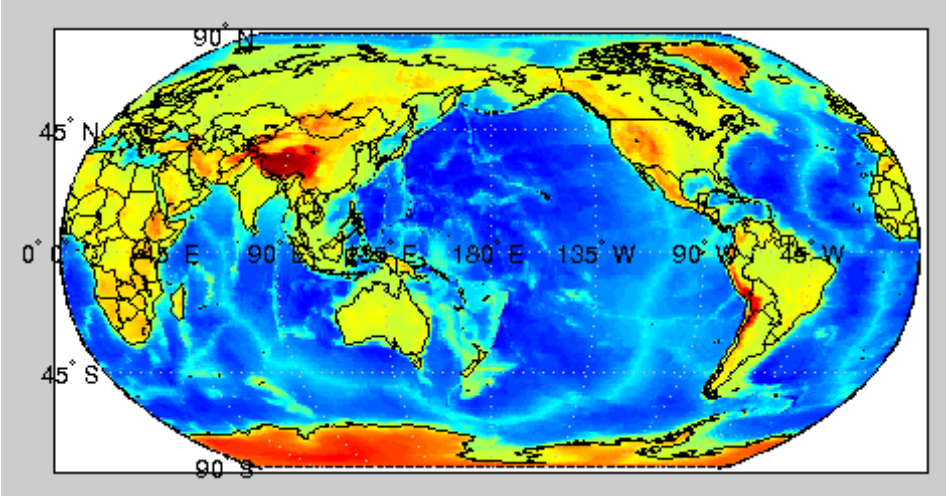
  >> colorbar('horiz')



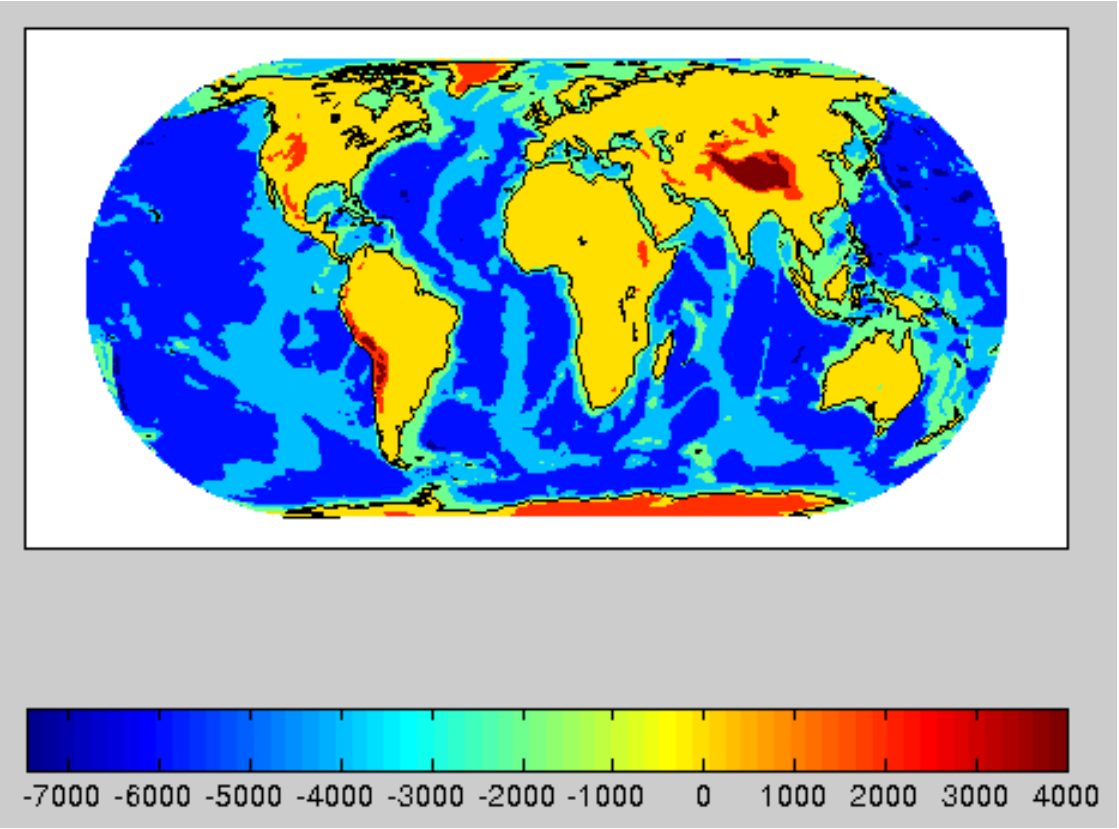
5. Τα δεδομένα του πίνακα topo φορτώνονται στη Matlab με την εντολή **load topo**. Όπως πολύ εύκολα μπορούμε να δούμε από το περιβάλλον εργασίας της Matlab δημιουργείται ένας χάρτης με διαστάσεις 180x360, όπως και στην περίπτωση του πίνακα geoid. Στη συγκεκριμένη περίπτωση όμως τα δεδομένα είναι τοπογραφία σε ένα κάναβο 1x1 μοίρα. Για να δημιουργήσουμε ένα χάρτη ακολουθούμε τα προηγούμενα παραδείγματα.



>> worldmap(topo,topolegend)




```
>> figure
>> axesm eckert4
>> plotm(coast,'k')
>> contourfm(topo,topolegend);
>> colorbar('horiz')
```



6. Για την ψηφιοποίηση θα χρησιμοποιηθεί το αρχείο **minput.m** το οποίο είναι ένα πρόγραμμα σε Matlab (ή αλλιώς μια συνάρτηση της Matlab). Το **minput.m** έχει δημιουργηθεί για το σκοπό αυτό και δεν αποτελεί μέρος του πακέτου της Matlab. Αφού το κατεβάσουμε και το τοποθετήσουμε στο φάκελο εργασίας, μαζί με την εικόνα που θα ψηφιοποιήσουμε (**test.jpg)** δίνουμε την εντολή **data=minput('test.jpg')** η εντολή αυτή θα εκτελέσει το αρχείο **minput.m** με είσοδο το **test.jpg** και θα αποθηκεύσει τα αποτελέσματα στη μεταβλητή **data**.

   Αρχικά από το παράθυρο εντολών δίνουμε τα φυσικά όρια του διαγράμματος (X 1 έως 11, Y 1 έως 11) για το σκοπό αυτό χρησιμοποιείται η εντολή input η οποία ζητά από τον χρήστη την εισαγωγή μιας τιμής και την επιστρέφει στην μεταβλητή (π.χ. η εντολή xmin = input('Specify xmin! '); **1** θα δώσει στην μεταβλητή xmin την τιμή **1**). Στη συνέχεια πρέπει να ορίσουμε τα όρια της εικόνας σε pixel και αυτό γίνεται κάνοντας κλικ στο κάτω αριστερό και πάνω δεξιό άκρο της και πατώντας το return, π.χ.

```
Click on lower left and upper right corner, then <return>
```

```
xcr =
   56.0000
  433.0000
ycr =
  427.0000
   49.0000
```

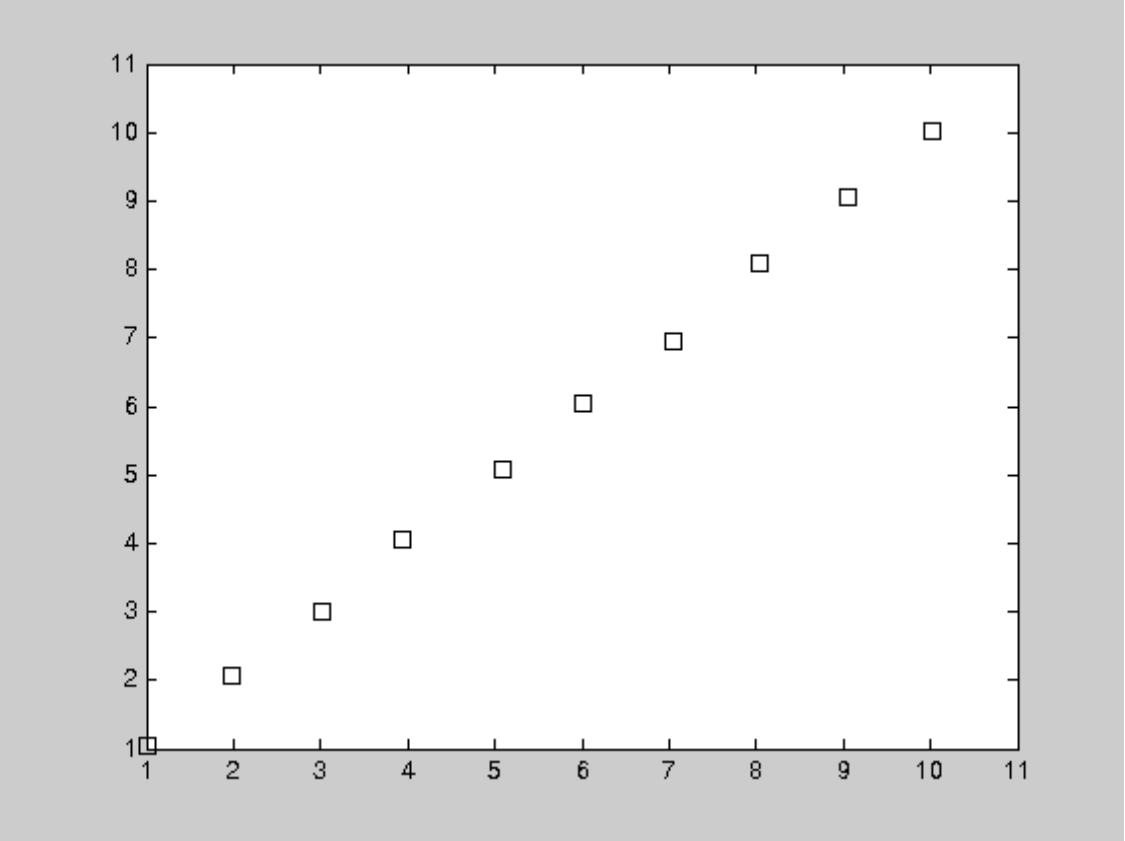
Οι τιμές δίνονται από την εντολή **ginput** η οποία επιστρέφει τις συντεταγμένες του κέρσορα. Οι πιο πάνω τιμές είναι οι τιμές σε pixel των ορίων (X, Y) της εικόνας, τις οποίες το πρόγραμμα θα χρησιμοποιήσει μαζί με τις πραγματικές τιμές που του έχουμε δώσει (X:1-11,Y:1-11) για να υπολογίσει τις πραγματικές τιμές κάθε σημείου.

Τέλος, η Matlab διαβάζει το αρχείο **test.jpg** και το προβάλει σε μια εικόνα, ταυτόχρονα ο κέρσορας έχει αλλάξει σε σταυρόνημα και μπορούμε να ψηφιοποιήσουμε τα σημεία του διαγράμματος κάνοντας διαδοχικά κλικ σε κάθε ένα από αυτά και πατώντας το return στο τέλος, π.χ.

```
Click on data points to digitize, then <return>

data =
    1.0000    1.0529
    1.9814    2.0582
    3.0159    3.0106
    3.9443    4.0423
    5.0849    5.0741
    6.0133    6.0529
    7.0477    6.9524
    8.0292    8.0899
    9.0637    9.0423
   10.0186   10.0212
```

Στο τέλος της διαδικασίας η μεταβλητή data έχει τις πραγματικές τιμές των σημείων και μπορούμε να κάνουμε το διάγραμμα με την εντολή plot, π.χ. **plot(data(:,1),data(:,2),'sk')**.



Το αρχείο **minput.m.**

```
function data = minput(imagefile)
% Εισαγωγή δεδομένων (φυσικά όρια του διαγράμματος)
xmin = input('Specify xmin! ');
xmax = input('Specify xmax! ');
ymin = input('Specify ymin! ');
ymax = input('Specify ymax! ');

% Άνοιγμα και προβολή της εικόνας
B = imread(imagefile);
a = size(B,2); b = size(B,1);
```

```
imshow(B);

% Ορισμός των ορίων της εικόνας σε pixel
disp('Click on lower left and upper right cr, then <return>')
[xcr,ycr]= ginput;
XMIN=xmin-((xmax-xmin)*xcr(1,1)/(xcr(2,1)-xcr(1,1)));
XMAX=xmax+((xmax-xmin)*(a-xcr(2,1))/(xcr(2,1)-xcr(1,1)));
YMIN=ymin-((ymax-ymin)*ycr(1,1)/(ycr(2,1)-ycr(1,1)));
YMAX=ymax+((ymax-ymin)*(b-ycr(2,1))/(ycr(2,1)-ycr(1,1)));

% Ψηφιοποίηση
disp('Click on data points to digitize, then <return>')
[xdata,ydata]= ginput;
XDATA = XMIN + ((XMAX-XMIN)*xdata / size(B,2));
YDATA = YMIN + ((YMAX-YMIN)*ydata / size(B,1));
data(:,1) = XDATA; data(:,2) = YDATA;
```

Η εικόνα test.jpg