Πανεπιστήμιο Θεσσαλίας
Τμήμα Πολιτικών Μηχανικών

Ανάλυση επιφανειακών φορέων

Ενότητα 5:
Προγραμματισμός στο Matlab

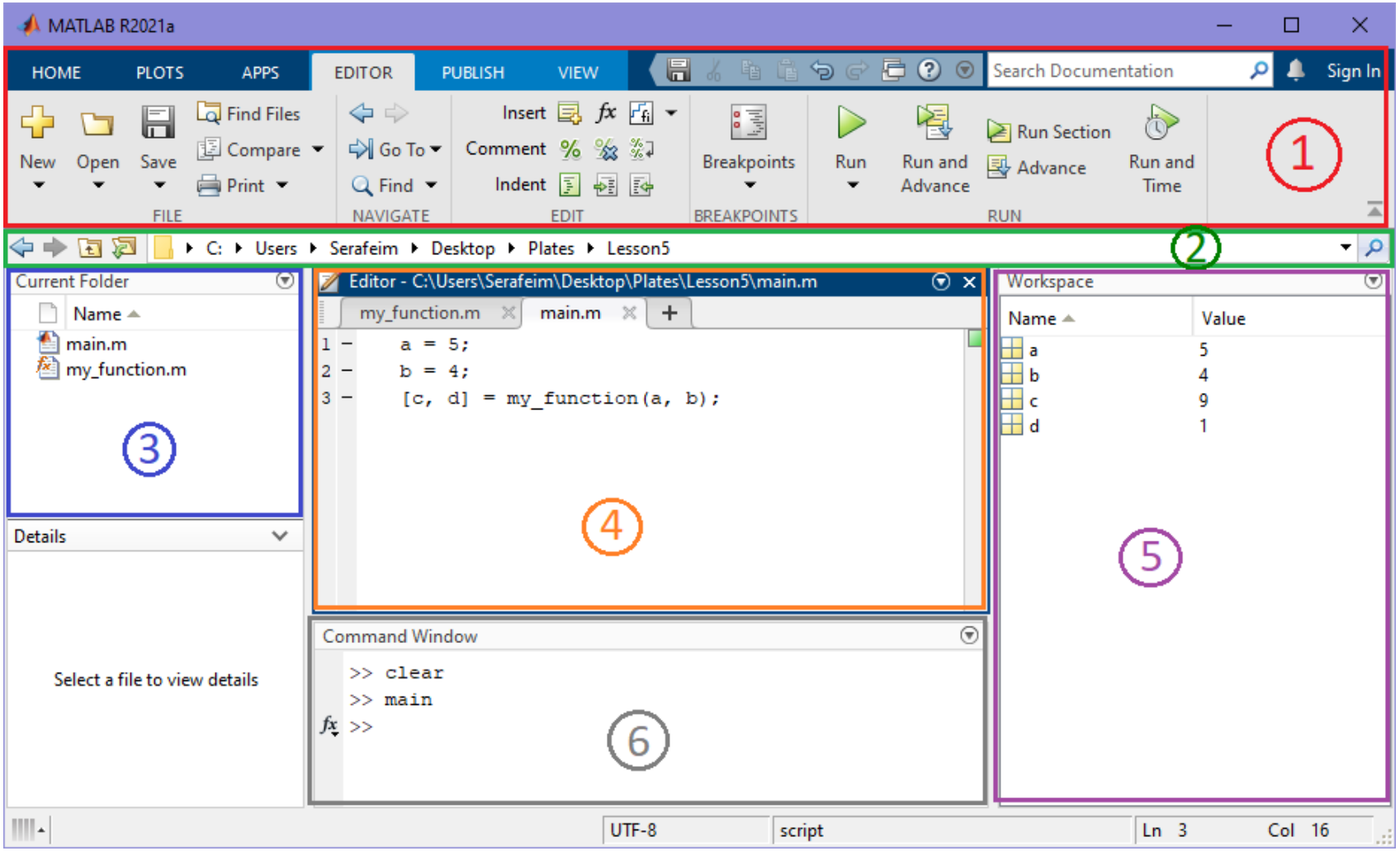
Σεραφείμ Μπακαλάκος
Phd Πολιτικός Μηχανικός
2023-2024

Εισαγωγή

- Το Matlab είναι ένα λογισμικό που μας επιτρέπει να γράφουμε προγράμματα εύκολα και γρήγορα, μέσα από ένα γραφικό περιβάλλον.
- Συμπεριλαμβάνει πάρα πολλές βιβλιοθήκες, τις οποίες μπορούμε να χρησιμοποιούμε καλώντας εύκολες συναρτήσεις.
- Π.χ. γραμμική άλγεβρα, βελτιστοποίηση, συμβολική άλγεβρα (computer algebra system) & ανάλυση, τεχνητή νοημοσύνη, σχεδίαση, κτλ.
- Λόγω της ευκολίας του και των παραπάνω βιβλιοθηκών, είναι εξαιρετικό εργαλείο για να δοκιμάσουμε να προγραμματίσουμε κάτι νέο.
- Σε αυτό το μάθημα θα δούμε κάποιες βασικές δυνατότητες του Matlab και πως θα τις χρησιμοποιήσουμε στην περίπτωση ανάλυσης πλακών.
- Προς το παρόν, θα περιοριστούμε στις αναλυτικές μεθόδους Navier, Levy. Σε επόμενα μαθήματα θα χρησιμοποιήσουμε το Matlab για τις αριθμητικές μεθόδους Πεπερασμένων Διαφορών και Πεπερασμένων Στοιχείων.

Το γραφικό περιβάλλον

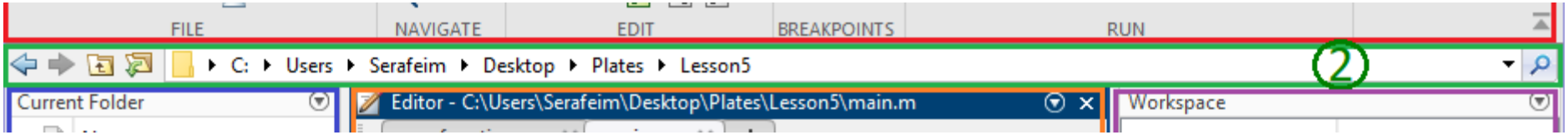
(1) Γραμμή εργαλείων. Από εδώ μπορούμε να δημιουργούμε/αποθηκεύουμε αρχεία, να τρέχουμε προγράμματα, να προσαρμόζουμε τις ρυθμίσεις του Matlab, κτλ.



Το γραφικό περιβάλλον

(2) Path. Είναι η τοποθεσία/φάκελος όπου θα τρέξει το πρόγραμμα που θα φτιάξουμε.

- Τα προγράμματα που θα φτιάχνουμε θα αποτελούνται από πολλά αρχεία. Όταν θα τρέξουμε κάποιο πρόγραμμα, το Matlab θα πρέπει να ψάξει όλα τα σχετικά αρχεία.
- Θα γλιτώσουμε από πολλά λάθη, αν όλα τα αρχεία για ένα πρόγραμμα βρίσκονται στον ίδιο φάκελο.



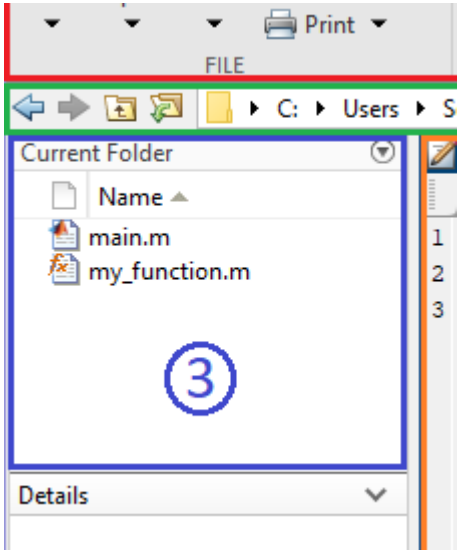
Πρόταση:

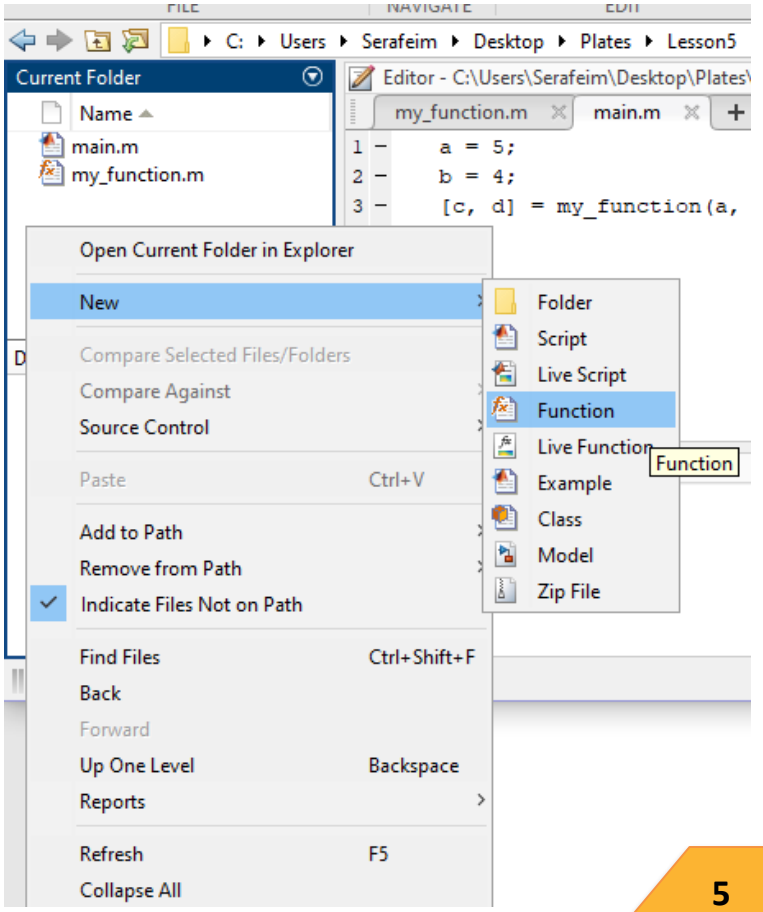
- Στο Desktop θα έχουμε ένα φάκελο Plates, για όλα τα αρχεία του μαθήματος.
- Μέσα σε αυτόν, θα έχουμε έναν υποφάκελος για κάθε άσκηση, ερώτημα θέματος, κτλ. Π.χ. Lesson3_2, Exercise1_1, Project_FEM, Project_FD
- Κάθε τέτοιος φάκελος θα αντιστοιχεί σε 1 πρόγραμμα. Όλα τα αρχεία που θα χρησιμοποιούνται στο προγράμμα (π.χ. κάποια ασκηση) θα περιέχονται εδω μέσα (και ας έχουμε συνολικά πολλά ίδια αρχεία για διαφορετικές ασκήσεις).

Το γραφικό περιβάλλον

(3) Current folder. Περιέχει τα αρχεία μέσα στο φάκελο που έχει οριστεί σαν Path (δες περιοχή 2).

- Τα αρχεία αυτά υπάρχουν μέσα στον υπολογιστή μας με ακριβώς την ίδια μορφή και οργάνωση.
- Από εδώ μπορούμε να ανοίγουμε κάποιο αρχείο για επεξεργασία.
- Μπορούμε επίσης να δημιουργούμε νέα αρχεία με δεξί κλικ -> New.



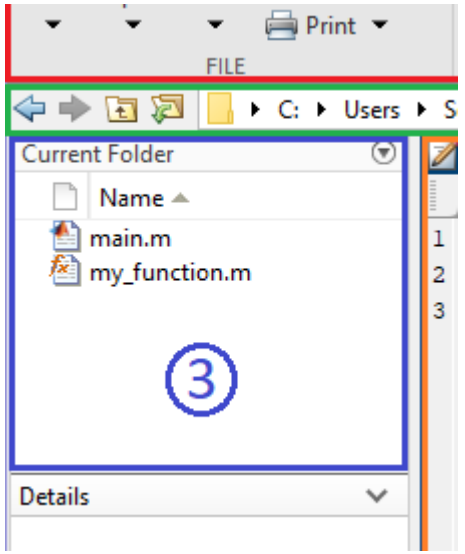


Το γραφικό περιβάλλον

(3) Current folder. Περιέχει τα αρχεία μέσα στο φάκελο που έχει οριστεί σαν Path.

Πρόταση:

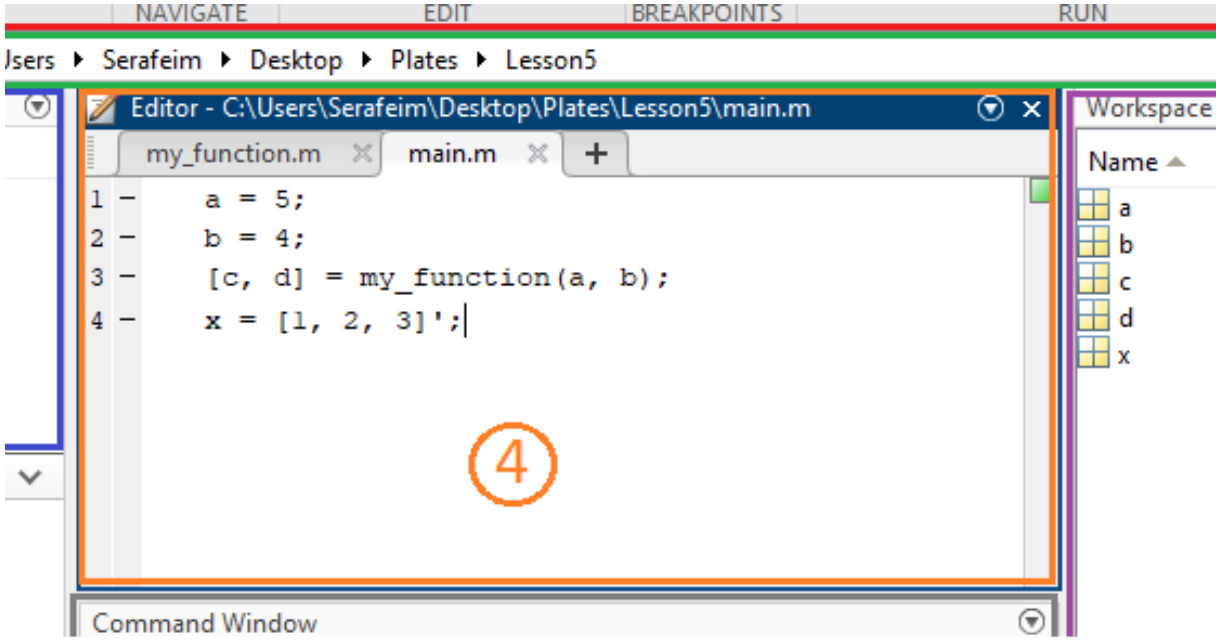
- Στον φάκελο του κάθε προγράμματος να υπάρχει
  - 1 αρχείο *script* με το όνομα main.m
  - Πολλά αρχεία *function* με ονόματα my_function.m (π.χ. calc_Mxy, plot_w)
- Για να τρέξουμε το πρόγραμμα, θα λέμε στο Matlab να τρέξει το *script* main.m
  - Μέσα στο main.m, θα φροντίζουμε να καλούνται όλα τα αρχεία *function* που χρειάζονται.
  - Το μεγαλύτερο μέρος της δουλειάς θα προγραμματίζεται στα αρχεία *function*. Το main.m απλά θα τα συντονίζει.



Το γραφικό περιβάλλον

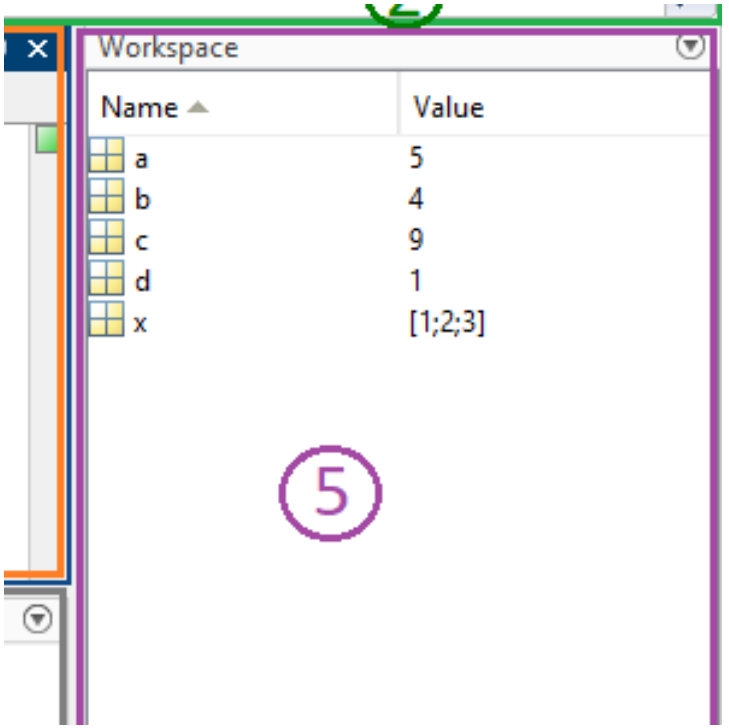
(4) Editor. Εδώ βλέπουμε και τροποποιούμε το περιεχόμενο των αρχείων μας.

- Κάθε αρχείο είναι ανοιγμένο σε ξεχωριστή καρτέλα. Π.χ. my_function.m, main.m.
- Οι καρτέλες ανοίγουν επιλέγοντας τα αντίστοιχα αρχεία από την περιοχή (2) (Current folder).
- Μπορούμε να κλείσουμε μία καρτέλα όταν τελειώσουμε με αυτή.
- Κάθε γραμμή του κώδικα/αρχείου έχει έναν αύξοντα αριθμό που φαίνεται στα αριστερά της.
- Αυτή είναι η βασική περιοχή που δουλεύουμε όταν γράφουμε ένα πρόγραμμα.
- Στα επόμενα θα εστιάσουμε σε αυτή την περιοχή.



Το γραφικό περιβάλλον

(5) Workspace. Εδώ βλέπουμε τις μεταβλητές που χρησιμοποιεί το πρόγραμμα.

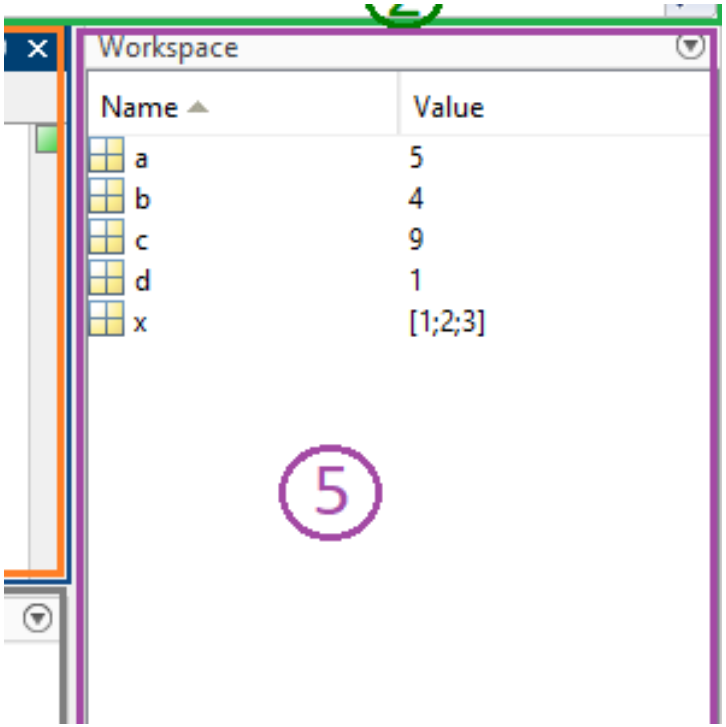


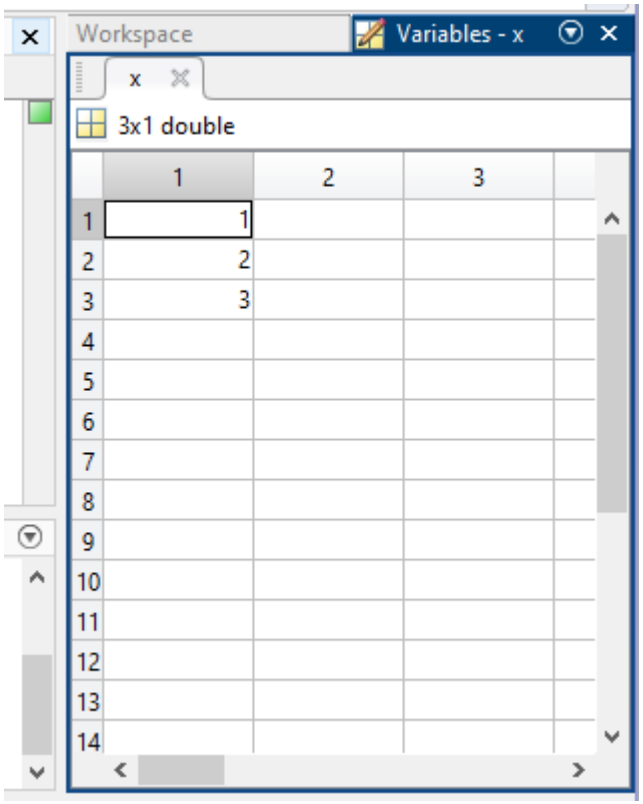
- Οι μεταβλητές είναι συνήθως μονόμετρα μεγέθη, διανύσματα ή πίνακες.
- Κατά την εκτέλεση του προγράμματος οι τιμές τους αλλάζουν.
- Στο Workspace φαίνεται η τελευταία τιμή που έχουν πάρει.
- Πριν ξεκινήσει να εκτελείται το πρόγραμμα, το Workspace είναι κενό.

Το γραφικό περιβάλλον

(5) Workspace. Εδώ βλέπουμε τις μεταβλητές που χρησιμοποιεί το πρόγραμμα.

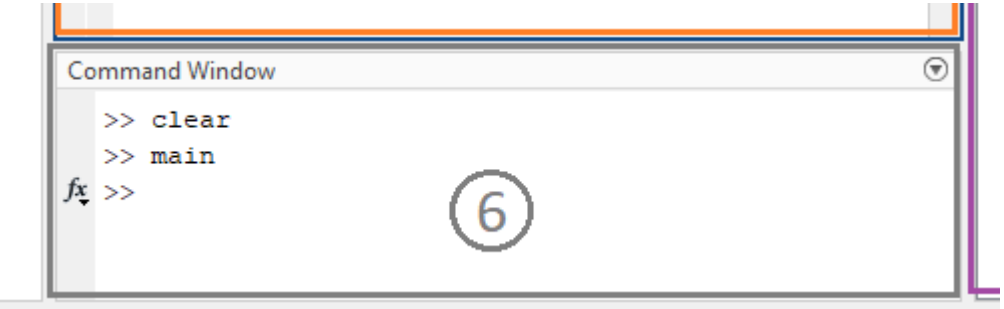
- Οι μεταβλητές είναι συνήθως μονόμετρα μεγέθη, διανύσματα ή πίνακες
- Κάνοντας διπλό κλικ σε κάποια μεταβλητή στο Workspace, αυτή μεταφέρεται σε υπολογιστικό φύλλο. Χρήσιμο για να εξετάσουμε όλες τα στοιχεία μεγάλων διανυσμάτων ή πινάκων.





Το γραφικό περιβάλλον

(6) Command Window. Από εδώ μπορούμε να εκτελούμε απευθείας εντολές ή να καλούμε άλλα αρχεία *script* και *function.*

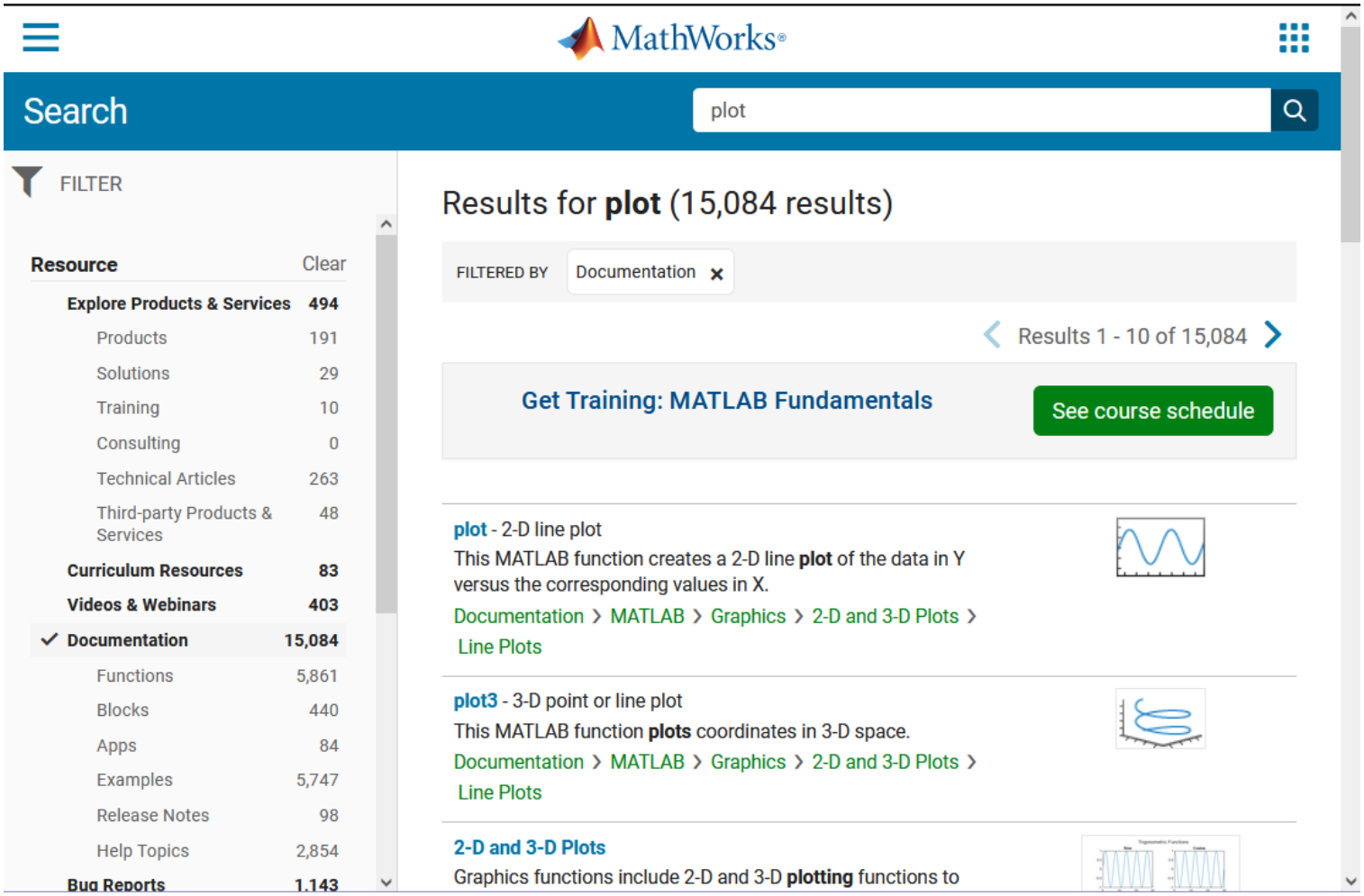


Πρόταση:

- Καλύτερα να μην καλούμε εντολές από εδώ ως αρχάριοι. Είναι προτιμότερο να γράφουμε τις εντολές μέσα στο main.m και κυρίως μέσα στα *function* αρχεία.
- Έτσι οι εντολές που θα γράψουμε μια φορά θα εκτελούνται πάντα.
- Η βασική λειτουργία του Command Window θα είναι να καλεί το main.m script. Το main.m θα καλεί τα υπόλοιπα.
- Πιθανή εξαίρεση: Αφού έχουμε τρέξει ένα πρόγραμμα και έχουμε πάρει αποτελέσματα, μπορεί να θέλουμε να τα εξετάσουμε καλύτερα. Π.χ. μπορεί να μην είμαστε σίγουροι αν ενα αποτέλεσμα είναι σωστό, οπότε να τρέξουμε απευθείας στο Command Window κάποιους ελέγχους μόνο για αυτή την περίπτωση.

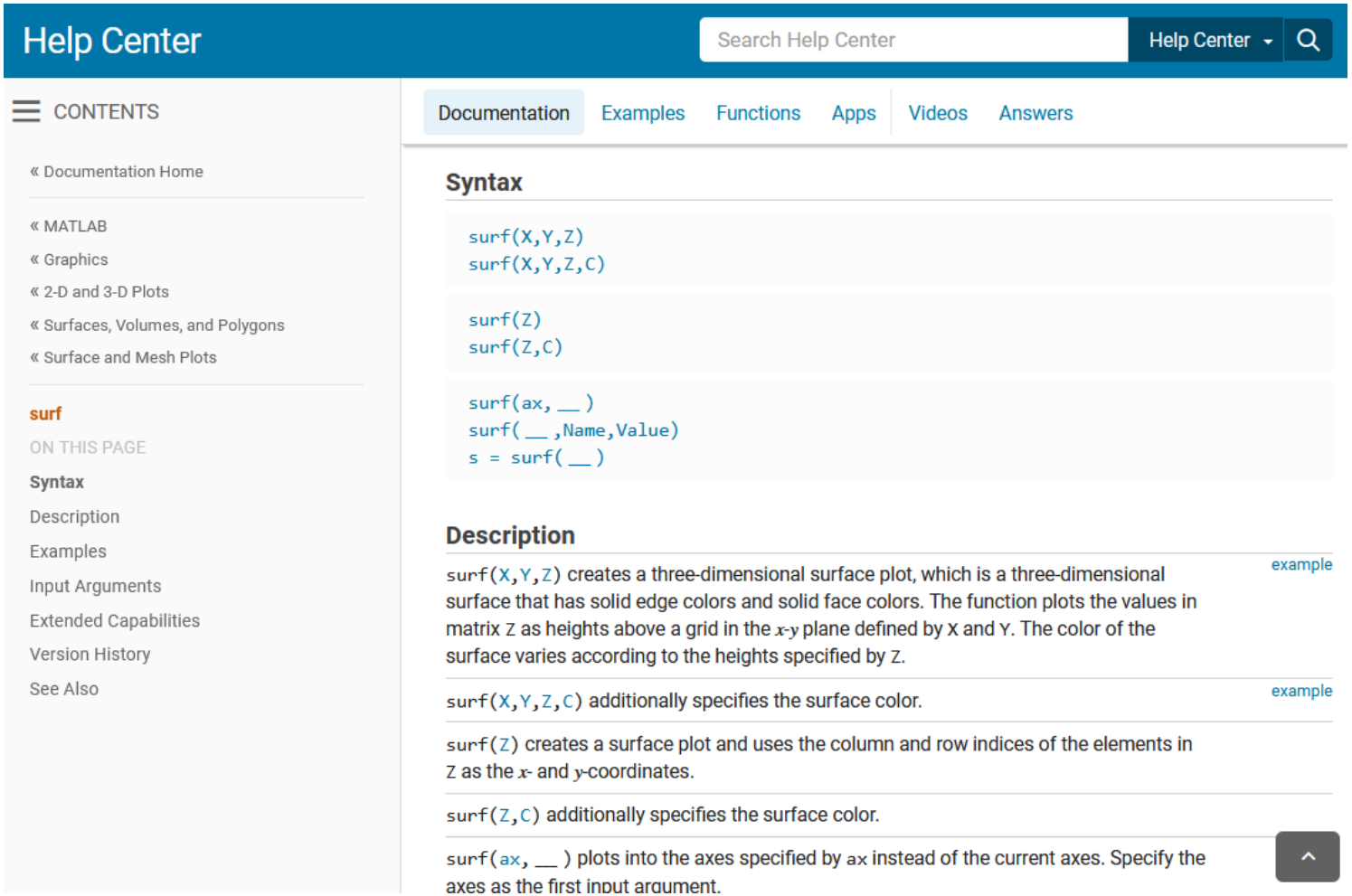
Επιπλέον οδηγίες

- Το Matlab έχει εκτενή σελίδες με οδηγίες στο site https://www.mathworks.com/
- Π.χ. έστω ότι ψάχνουμε πως θα σχεδιάσουμε 3D διαγράμματα.
- Γραφουμε “plot” στο search bar και διαλέγουμε από τα αποτελέσματα.



Επιπλέον οδηγίες

- Έστω ότι διαλέξαμε το αποτέλεσμα “surface plot”. Ανοίγει η αντίστοιχη σελίδα.
- Εδώ εξηγείται αναλυτικά ποια συνάρτηση πρέπει να καλέσουμε και με τι ορίσματα.
- Επίσης υπάρχουν αρκετά παραδείγματα για κάθε υποπερίπτωση.



Παράδειγμα

- Έστω ότι έχουμε λύσει το παράδειγμα 1 της μεθόδου Navier.
- Βύθιση

$$w(x,y) = \frac{16q_0}{\pi^6 D} \sum_{m=1,3,\dots}^{\infty} \sum_{n=1,3,\dots}^{\infty} \frac{sin\frac{m\pi x}{a} sin\frac{n\pi y}{b}}{mn\left(\frac{m^2}{a^2} + \frac{n^2}{b^2}\right)^2}$$

- Ροπή σε παρειά κάθετη στον άξονα x

$$M_x = \frac{16q_0}{\pi^4} \sum_{m=1,3,\dots}^{\infty} \sum_{n=1,3,\dots}^{\infty} \frac{\frac{m^2}{a^2} + \nu\frac{n^2}{b^2}}{mn\left(\frac{m^2}{a^2} + \frac{n^2}{b^2}\right)^2} sin\frac{m\pi x}{a} sin\frac{n\pi y}{b}$$

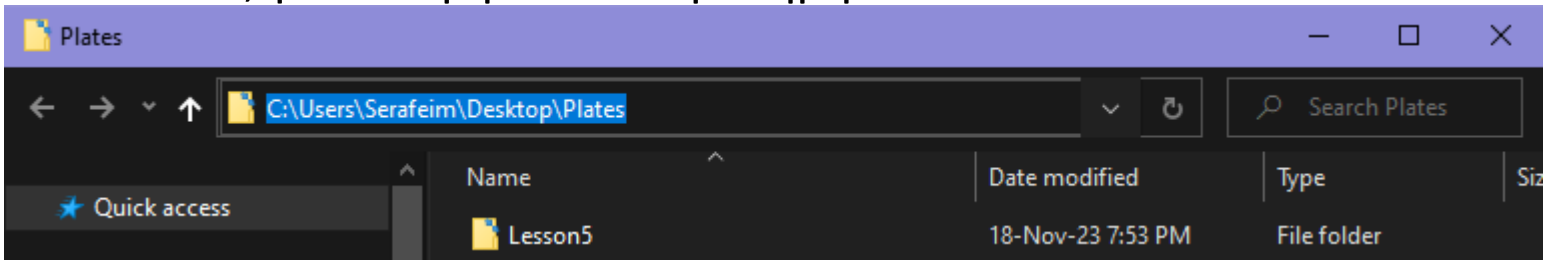
- Θέλουμε να
  - Υπολογίσουμε τη μέγιστη βύθιση $w_{max} = w(\frac{a}{2}, \frac{b}{2})$
  - Σχεδιάσουμε την 2D συνάρτηση της βύθισης w
  - Σχεδιάσουμε διάγραμμα ροπών M_x στην ευθεία $x = a/2$

Παράδειγμα - Στρατηγική

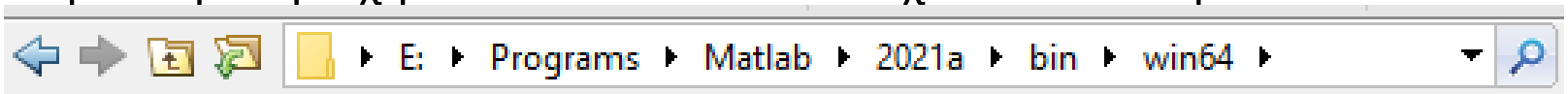
- Θα σπάσουμε το πρόβλημα σε πολλές συναρτήσεις, καθεμία απο τις οποίες θα κάνει μία συγκεκριμένη δουλειά.
- Μια συνάρτηση calc_w(…) θα υπολογίζει τη βύθιση w σε οποιοδήποτε σημείο (x, y).
- Μια συνάρτηση calc_Mx(…) θα υπολογίζει τη ροπή M_x σε οποιοδήποτε σημείο (x,y).
- Μία συνάρτηση plot_w(…) θα σχεδιάζει την παραμορφωμένη επιφάνεια της πλάκας. Θα χρησιμοποιεί την calc_w(…) για να υπολογίζει τη βύθιση σε ένα πλήθος σημείων και θα σχεδιάζει τις τριάδες $(x_i, y_i, w(x_i, y_i))$
- Μία συνάρτηση plot_Mx(…) θα σχεδιάζει το ζητούμενο διάγραμμα ροπής. Θα χρησιμοποιεί την calc_Mx (…) για να υπολογίζει τη M_x σε ένα πλήθος κατά την ευθεία $x = \frac{a}{2}$ και θα σχεδιάζει τα ζεύγη $\left(y_i, M_x\left(\frac{a}{2}, y_i\right)\right)$
- Το script main.m θα δίνει τιμές στις παραμέτρους του προβλήματος (π.χ. α, b, D) και θα καλεί τις plot_w(…), plot_Mx(…). Για το w_{max} θα καλεί την calc_w(...) με όρισμα τις συντεταγμένες $(\frac{a}{2}, \frac{b}{2})$.

Παράδειγμα – Αρχή προγράμματος

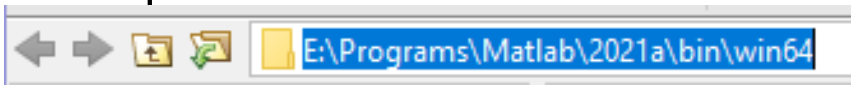
- Ανοίγουμε την εφαρμογή Matlab, π.χ. από την Έναρξη
- Βάζουμε το Path στο οποίο θέλουμε να δουλέψουμε
  - Ανοίγουμε το φάκελο Plates στον υπολογιστή μας. Αν δεν το έχουμε φτιάξει ήδη, τον δημιουργούμε στο Desktop.
  - Αντιγράφουμε την τοποθεσία του φακέλου. Κάνουμε πρώτα αριστερό κλικ στο πεδίο, για να εμφανίσει τη πλήρη τοποθεσία



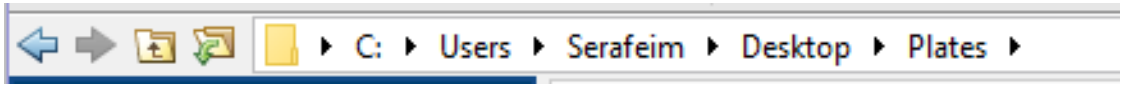
  - Πάμε στην περιοχή του Path. Μάλλον θα έχει το default path του Matlab.



  - Κάνουμε και πάλι κλικ πάνω στο πεδίο (E:>Programs>Matlab>…)

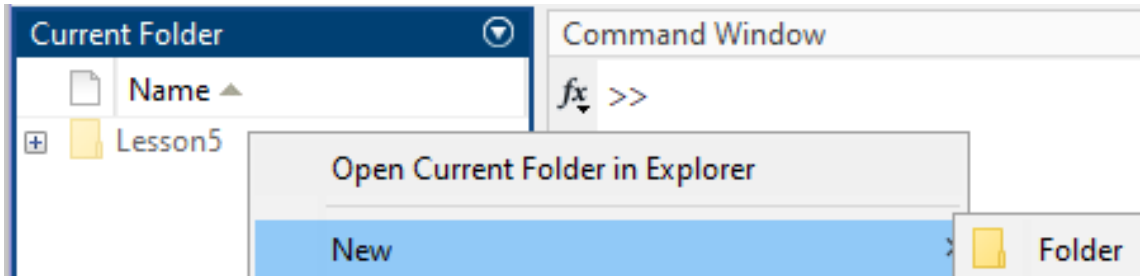


  - Επικολλούμε την τοποθεσία που αντιγράψαμε και πατάμε Enter

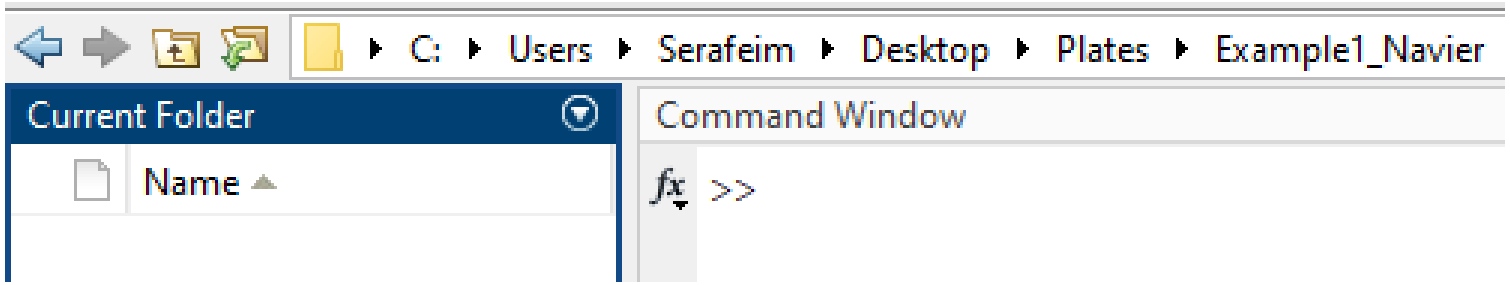


Παράδειγμα – Αρχή προγράμματος

- Με δεξί κλικ στην περιοχή Current Folder, φτιάχνουμε ένα υποφάκελο για τα αρχεία του προγράμματος. Τον ονομάζουμε Example1_Navier.

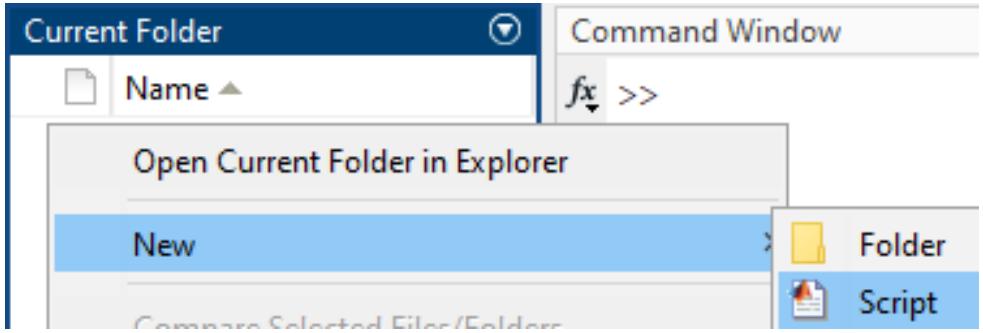


- Κάνουμε διπλό κλικ πάνω του για να τον ανοίξουμε. Παρατηρείστε ότι ανανεώθηκε το Path.

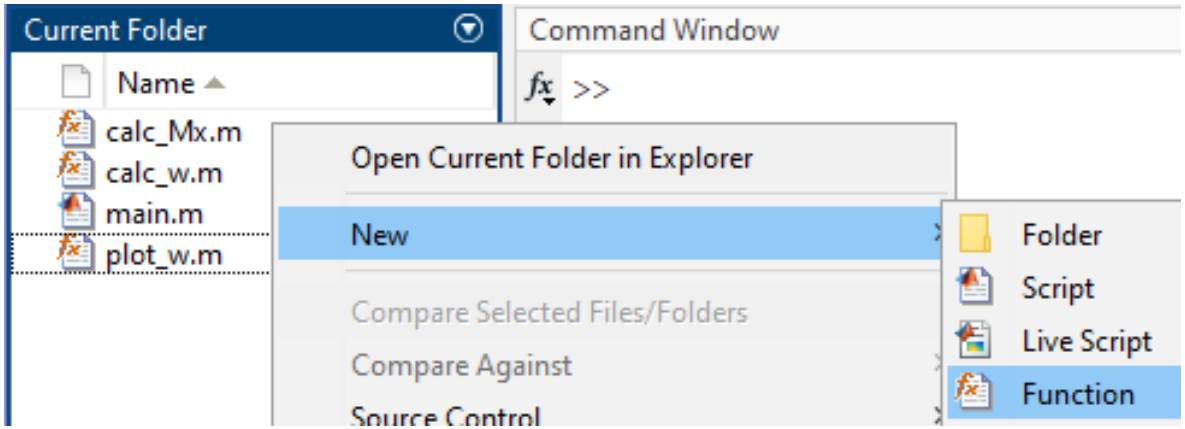


Παράδειγμα – Αρχή προγράμματος

- Δημιουργούμε:
  - Το αρχείο script με όνομα main



  - Τα αρχεία function με ονόματα calc_w, calc_Mx, plot_w, plot_Mx.



Παράδειγμα – calc_w

- Κάνουμε διπλό κλικ στο αρχείο calc_w.m για να το επεξεργαστούμε. Αρχικά το Matlab έχει φτιάξει ένα template

```
1   function [outputArg1,outputArg2] = calc_w(inputArg1,inputArg2)
2   %CALC_W Summary of this function goes here
3   %   Detailed explanation goes here
4   outputArg1 = inputArg1;
5   outputArg2 = inputArg2;
6   end
```

- Το οποίο θα αλλάξουμε σε

```
1   function [w] = calc_w(x, y, a, b, D, q0, max_m, max_n)
```

- Η συνάρτηση θα επιστρέφει ως output (μέσα σε **[ ]**) μία τιμή w
- Η συνάρτηση έχει ορίσματα (μέσα σε **( )**):
  - συντεταγμένες του σημείου (x,y)
  - διαστάσεις πλάκας a, b και
  - καμπτική δυσκαμψία πλάκας D φορτίο q_0
  - τα μέγιστα m και n που θα κρατήσει στις σειρές Fourier (max_m, max_n)

Παράδειγμα – calc_w

- Πριν ξεκινήσουμε να προγραμματίζουμε, γράφουμε βασικά σχόλια για τη συνάρτηση.

```
%calc_w: Calculates the displacement of a plate along
%   axis z for a point (x,y). The plate is loaded with
%   uniform load q0. The solution is derived by
%   Navier's method.
%w: Real number. The displacement along axis z.
%x,y: Real numbers. The coordinates of the point.
%a, b: Real numbers. The lengths of the plate along x, y.
%D: Real number. The stiffness of the plate.
%q0: Real number. The uniform load.
%max_m, max_n: Integer numbers. Maximum m, n to keep
% in the Fourier sums for sin(m*pi*x/a) and sin(n*pi*y/b).

```

- Εξηγούμε τι κάνει η συνάρτηση, ποια η σημασία και ο τύπος του κάθε input και output.
- Δεν βαριόμαστε. Οποιοσδήποτε διαβάσει τον κώδικά μας (και ο μελλοντικός εαυτός μας) θα μας ευχαριστήσει.

Παράδειγμα – calc_w

- Γράφουμε ελέγχους για τα ορίσματα.
- Αν δεν έχουν αναμενόμενες τιμές, φροντίζουμε να σταματήσει η εκτέλεση του προγράμματος, μέσω του μηχανισμού throw(Mexception('Περιγραφή λάθους'))

```
13
14 -    if (a <= 0) || (b <= 0) || (D <= 0)
15 -       throw(MException('a, b, D must be positive'))
16 -    end
17 -    if (max_m < 1) || (max_n < 1)
18 -       throw(MException('max_m, max_n must be at least 1.'))
19 -    end
20 -    if (x < 0) || (x > a) || (y < 0) || (y > b)
21 -       throw(MException('Make sure 0 <= x <= a, 0 <= y <= b'))
22 -    end
```

- Εδώ ελέγχουμε τις διαστάσεις και δυσκαμψία της πλάκας, αλλά όχι το φορτίο.
- Ελέγχουμε επίσης να υπάρχει τουλάχιστον ένας όρος για το άθροισμα Fourier.
- Ελέγχουμε επίσης το σημείο x,y να είναι εντός της πλάκας.
- **||** λογικό ή. **&&** λογικό και

Παράδειγμα – calc_w

- Αν δοθεί λάθος τιμή για κάποιο όρισμα, π.χ. αρνητική τιμή για το a, η εκτέλεση του προγράμματος θα σταματήσει και θα γραφτεί το εξής μήνυμα στο Command Window

```
Command Window
  Error in calc_w (line 15)
     throw(MException('a, b, D, q0 must be positive'))

  Error in main (line 8)
  w_max = calc_w(a/2, b/2, -a, b, D, q0, max_m, max_n);
```

- Καλό είναι να γράφονται πάντα έλεγχοι για τα ορίσματα.
- Αν εξασφαλίζουμε οτι εχουμε σωστές τιμές πριν ξεκινήσουμε, θα γλιτώσουμε πολύ χρόνο, που αλλιώς θα ψάχναμε λάθη σε όλη τη συνάρτηση ή σε όλο το πρόγραμμα.

Παράδειγμα – calc_w

- Προγραμματίζουμε τον υπολογισμό της βύθισης w

$$w(x,y) = \frac{16q_0}{\pi^6 D} \sum_{m=1,3,\ldots}^{\infty} \sum_{n=1,3,\ldots}^{\infty} \frac{\sin\frac{m\pi x}{a} \sin\frac{n\pi y}{b}}{mn\left(\frac{m^2}{a^2} + \frac{n^2}{b^2}\right)^2}$$

```
23
24 -    sum=0;
25 -    for m=1:2:max_m
26 -        for n=1:2:max_n
27 -            v1 = sin(m*pi*x/a) * sin(n*pi*y/b);
28 -            v2 = m*n* (m^2/a^2 + n^2 / a^2)^2;
29 -            sum = sum + v1 / v2;
30 -        end
31 -    end
32 -    w = 16 * q0 * sum / (pi^6 * D);
33
34 -    end
```

Παράδειγμα – calc_w

- Προγραμματίζουμε τον υπολογισμό της βύθισης w

```
23
24 -    sum=0;
25 -    for m=1:2:max_m
26 -        for n=1:2:max_n
27 -            v1 = sin(m*pi*x/a) * sin(n*pi*y/b);
28 -            v2 = m*n* (m^2/a^2 + n^2 / a^2)^2;
29 -            sum = sum + v1 / v2;
30 -        end
31 -    end
32 -    w = 16 * q0 * sum / (pi^6 * D);
33
34 -    end
```

- Για το άθροισμα (μεταβλητή sum), έχουμε ένα διπλό for loop στις γραμμές 25-31.
- Κάθε for αντιστοιχεί σε συγκεκριμένο end. Φροντίζουμε η στοίχιση να δείχνει τις αντιστοιχίες.
- Το τελευταίο end (γραμμή 34) κλείνει τη συνάρτηση, οχι καποιο for loop/if.

Παράδειγμα – calc_w

```
23
24 -    sum=0;
25 -    for m=1:2:max_m
26 -        for n=1:2:max_n
27 -            v1 = sin(m*pi*x/a) * sin(n*pi*y/b);
28 -            v2 = m*n* (m^2/a^2 + n^2 / a^2)^2;
29 -            sum = sum + v1 / v2;
30 -        end
31 -    end
32 -    w = 16 * q0 * sum / (pi^6 * D);
33
34 -    end
```

- Γρ 25: Το m ξεκινά από τιμή **1** και με βήμα **2** φτάνει μέχρι και τιμή **max_m**
- Γρ 27: Υπολογίζω τον αριθμητή και αποθηκεύω στη μεταβλητή v1.
- Το π είναι **pi** και η συνάρτηση του ημιτόνου **sin(…)**
- **+** πολλαπλασιασμός, **/** διαίρεση, **^** ύψωση σε εκθέτη.
- Χρησιμοποιώ παρενθέσεις για τη σειρά των πράξεων.
- Όταν αποθηκεύω σε μεταβλητή, τέλος βάζω **;** στο τέλος της γραμμής.

Παράδειγμα – calc_Mx

- Προγραμματίζουμε τον υπολογισμό της ροπής Mx

$$M_x = \frac{16q_0}{\pi^4} \sum_{m=1,3,\ldots}^{\infty} \sum_{i=1,3,\ldots}^{\infty} \frac{\frac{m^2}{a^2} + \nu \frac{n^2}{b^2}}{mn\left(\frac{m^2}{a^2} + \frac{n^2}{b^2}\right)^2} sin\frac{m\pi x}{a} sin\frac{n\pi y}{b}$$

```
1   function Mx = calc_Mx(x, y, a, b, v, q0, max_m, max_n)
2
3   sum=0;
4   for m=1:2:max_m
5       for n=1:2:max_n
6           v1 = sin(m*pi*x/a) * sin(n*pi*y/b);
7           v2 = m*n* (m^2/a^2 + n^2 / a^2)^2;
8           v3 = m^2/a^2 + v * n^2/b^2;
9           sum = sum + v1 * v3 / v2;
10      end
11  end
12  Mx = 16 * q0 * sum / pi^4;
13
14  end
```

Παράδειγμα – calc_Mx

```
1   function Mx = calc_Mx(x, y, a, b, v, q0, max_m, max_n)
2
3   sum=0;
4   for m=1:2:max_m
5       for n=1:2:max_n
6           v1 = sin(m*pi*x/a) * sin(n*pi*y/b);
7           v2 = m*n* (m^2/a^2 + n^2 / a^2)^2;
8           v3 = m^2/a^2 + v * n^2/b^2;
9           sum = sum + v1 * v3 / v2;
10      end
11  end
12  Mx = 16 * q0 * sum / pi^4;
13
14  end
```

- Σχεδόν ίδια συνάρτηση με την calc_w.
- Σαν όρισμα έχουμε και το συντελεστή Poisson v, αλλα όχι τη δυσκαμψία D.

Παράδειγμα – plot_w

```
1    function [] = plot_w(a, b, D, q0, max_m, max_n, ...
2        num_points_x, num_points_y)
3
```

- Η συνάρτηση θα σχεδιάσει το w σε συγκεκριμένα σημεία σε ένα κάναβο (grid) από num_points_x × num_points_y σημεία (x_i, y_j).
- Στα ορίσματα έχουμε προσθέσει πόσα σημεία θα έχει ο κάναβος σε κάθε άξονα.
- Σκοπός της συνάρτησης είναι να σχεδιάσει, οπότε δεν επιστρέφει output.
- Για να συνεχίσουμε οποιαδήποτε εντολή στην επόμενη σειρά, βάζουμε **...**

Παράδειγμα – plot_w

```
4     % Distances between discrete points
5 -   dist_x = a / (num_points_x - 1);
6 -   dist_y = b / (num_points_y - 1);
7
8     % Matrices with x-y coordinates and w of the discrete points
9 -   X = zeros(num_points_x, num_points_y);
10 -  Y = zeros(num_points_x, num_points_y);
11 -  W = zeros(num_points_x, num_points_y);
12
```

- Γρ 5: Καθορίζουμε την απόσταση μεταξύ σημείων στον άξονα x
- Θα χρησιμοποιήσουμε 3 πίνακες.
  - X θα περιέχει τις x συντεταγμένες των διακριτών σημείων.
  - Y θα περιέχει τις y συντεταγμένες των διακριτών σημείων.
  - W θα περιέχει τις βυθίσεις των διακριτών σημείων.
- Γρ 9: Δημιουργούμε κενό πίνακα (με μηδενικά) που έχει num_points_x γραμμές και num_points_y στήλες.
- Για να μάθουμε πόσες γραμμές ή στήλες έχει ένας πίνακας, χρησιμοποιούμε size(A,1) ή size(A, 2) αντίστοιχα.

Παράδειγμα – plot_w

```
13      % Calculate w at these discrete points
14 -    for i=1:1:num_points_x
15 -        for j=1:1:num_points_y
16 -            x = (i - 1) * dist_x;
17 -            y = (j - 1) * dist_y;
18 -            w = calc_w(x, y, a, b, D, q0, max_m, max_n);
19 -            X(i,j) = x;
20 -            Y(i,j) = y;
21 -            W(i,j) = w;
22 -        end
23 -    end
```

- Γρ 16-17: Βρίσκουμε τις συντεταγμένες των διακριτών σημείων (x_i, y_j)
- Γρ 18: Χρησιμοποιούμε τη συνάρτηση calc_w που φτιάξαμε για να υπολογίσουμε τη βύθιση.
- Γρ 19-21: Αποθηκεύουμε τα παραπάνω στους αντίστοιχους πίνακες.
- Μπορούμε να διαβάσουμε ή να γράψουμε ένα στοιχείο πίνακα ως A(i,j).

Παράδειγμα – plot_w

```
24
25      % Plot the displacement
26 -    figure % create new figure without closing others
27 -    surf(X, Y, W); % surface plot
28 -    title('w(x,y)') % add title to figure
29 -    xlabel('x')
30 -    ylabel('y')
31 -    zlabel('w')
32
33 -    end
```

- Γρ 26: Με την εντολή figure δημιουργούμε νέο σχήμα. Αλλιώς το Matlab θα έκλεινε τα υπάρχοντα
- Γρ 27: 3D σχεδίαση της επιφάνειας $w(x, y)$
- Γρ 28-31: Τίτλοι του σχήματος και των αξόνων.

Παράδειγμα – plot_w

Π.χ.

- a = 3
- b = 2
- num_points_x = 6
- num_points_y = 4

W

6x4 double

	1	2	3	4
1	0	0	0	0
2	0	-0.0017	-0.0017	-2.6150e-19
3	0	-0.0028	-0.0028	-4.0682e-19
4	0	-0.0028	-0.0028	-4.0682e-19
5	0	-0.0017	-0.0017	-2.6150e-19
6	0	-3.7989e-19	-3.7989e-19	-5.8563e-35

X

6x4 double

	1	2	3	4
1	0	0	0	0
2	0.6000	0.6000	0.6000	0.6000
3	1.2000	1.2000	1.2000	1.2000
4	1.8000	1.8000	1.8000	1.8000
5	2.4000	2.4000	2.4000	2.4000
6	3	3	3	3

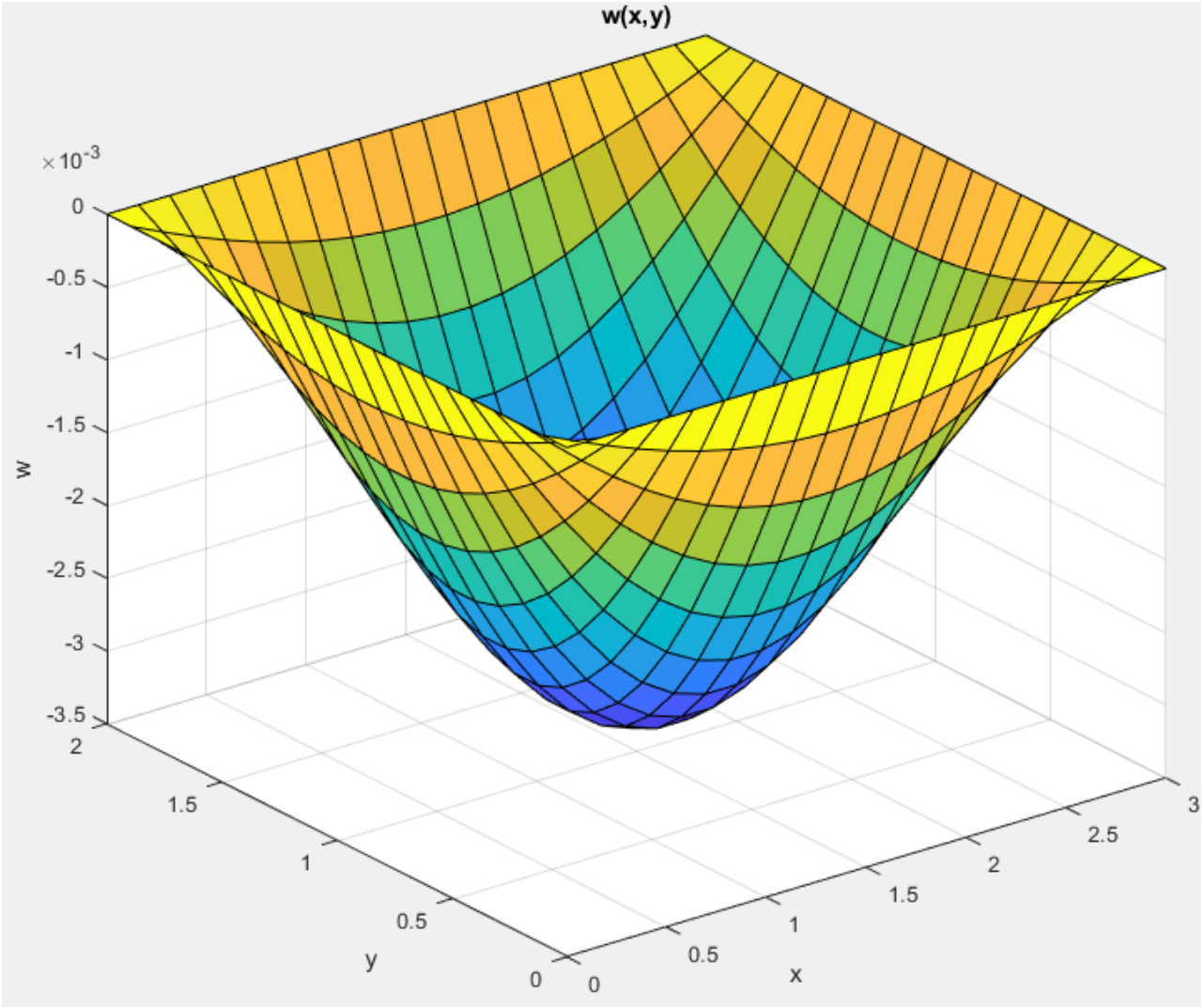
Y

6x4 double

	1	2	3	4
1	0	0.6667	1.3333	2
2	0	0.6667	1.3333	2
3	0	0.6667	1.3333	2
4	0	0.6667	1.3333	2
5	0	0.6667	1.3333	2
6	0	0.6667	1.3333	2

Παράδειγμα – plot_w

- Το γράφημα που φαίνεται είναι για 20×20 σημεία
- Τα θετικά του άξονα w είναι προς τα πάνω και το φορτίο q_0 αρνητικό



Παράδειγμα – plot_Mx

```
1    function [] = plot_Mx(a, b, v, q0, max_m, max_n, ...
2        x0, num_points_y)
```

- Η συνάρτηση θα σχεδιάσει το M_x σε διακριτά σημεία σε μία ευθεία από num_points_y σημεία y_j.
- Η ευθεία είναι η, οπότε τα σημεία τοποθετούνται παράλληλα στον άξονα y.
- Στα ορίσματα έχουμε προσθέσει
  - Πόσα σημεία θα έχει η ευθεία $x = x_0$, που είναι παράλληλη στον άξονα y.
  - Ποιό είναι το x_0
- Σκοπός της συνάρτησης είναι να σχεδιάσει, οπότε δεν επιστρέφει output.

Παράδειγμα – plot_Mx

```
3
4       % Distances between discrete points
5 -     dist_y = b / (num_points_y - 1);
6
7       % Vectors with y coordinates and w of the discrete points
8 -     Y = zeros(num_points_y);
9 -     M = zeros(num_points_y);
```

- Γρ 5: Καθορίζουμε την απόσταση μεταξύ σημείων στον άξονα y
- Θα χρησιμοποιήσουμε 2 διανύσματα.
  - Y θα περιέχει τις y συντεταγμένες των διακριτών σημείων.
  - M θα περιέχει τις ροπές των διακριτών σημείων.
- Γρ 8: Δημιουργούμε κενό διάνυσμα (με μηδενικά) που έχει num_points_y στοιχεία.
- Για να μάθουμε πόσα στοιχεία έχει ένα διάνυσμα x χρησιμοποιούμε length(x)

Παράδειγμα – plot_Mx

```
10
11      % Calculate w at these discrete points
12 -    for j=1:1:num_points_y
13 -        y = (j - 1) * dist_y;
14 -        mx = calc_Mx(x0, y, a, b, v, q0, max_m, max_n);
15 -        Y(j) = y;
16 -        M(j) = mx;
17 -    end
```

- Γρ 13: Βρίσκουμε τις συντεταγμένες των διακριτών σημείων y_j. Ξέρουμε ότι $x = x_0$
- Γρ 14: Χρησιμοποιούμε τη συνάρτηση calc_Mx που φτιάξαμε για να υπολογίσουμε τη ροπή.
- Γρ 15-16: Αποθηκεύουμε τα παραπάνω στους αντίστοιχους πίνακες.
- Μπορούμε να διαβάσουμε ή να γράψουμε ένα στοιχείο διανύσματος x ως x(i).

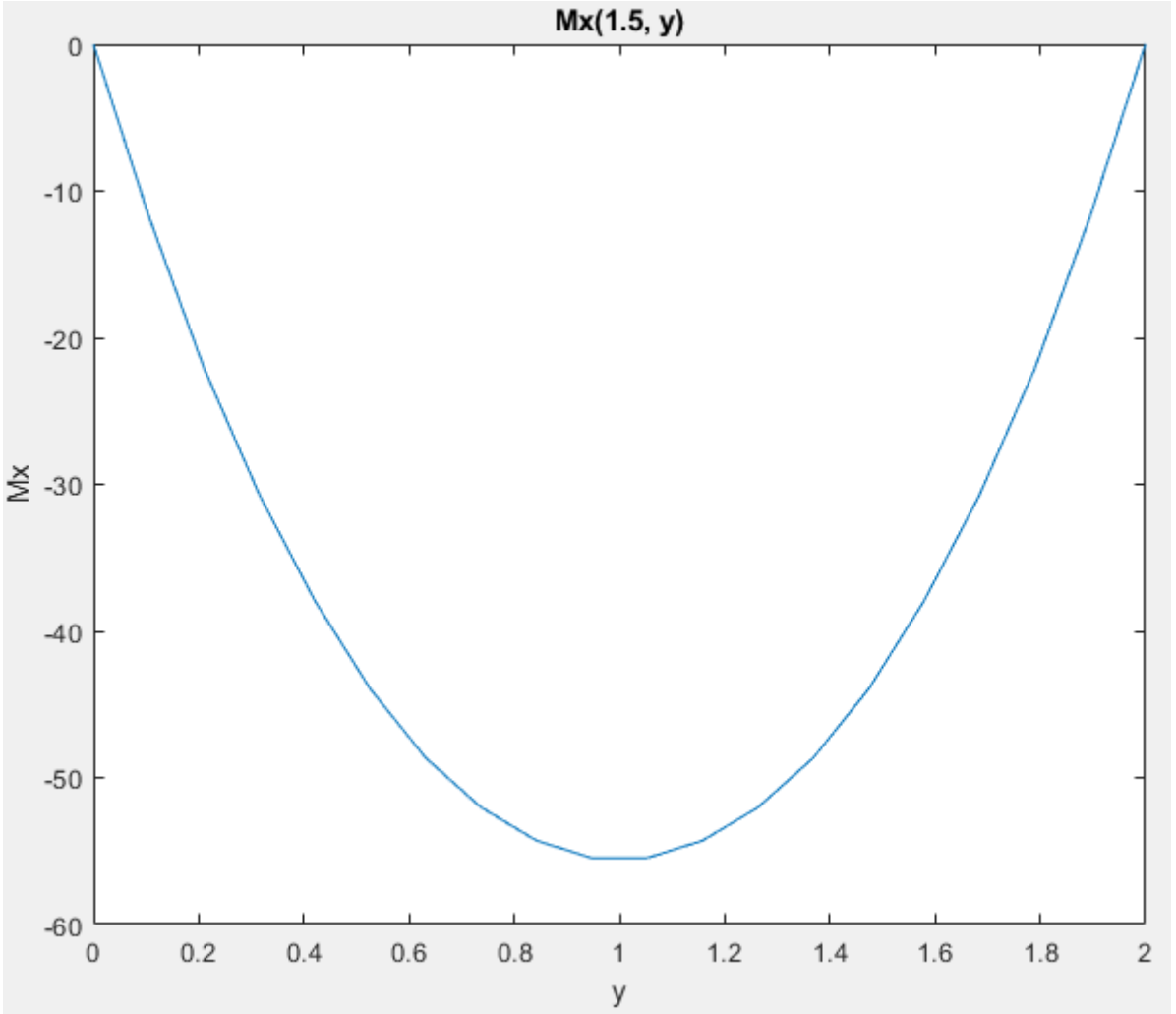
Παράδειγμα – plot_Mx

```
18
19      % Plot the moment
20 -    figure % create new figure without closing others
21 -    plot(Y, M); % 2D line plot
22 -    title(['Mx(', num2str(x0), ', y)']) % add title to figure
23 -    xlabel('y')
24 -    ylabel('Mx')
25
26 -    end
```

- Γρ 20: Με την εντολή figure δημιουργούμε νέο σχήμα. Αλλιώς το Matlab θα έκλεινε τα υπάρχοντα
- Γρ 21: 2D σχεδίαση της καμπύλης $\mathrm{Mx}(y)$
- Γρ 22-24: Τίτλοι του σχήματος και των αξόνων.

Παράδειγμα – plot_Mx

- Το γράφημα που φαίνεται είναι για 20 σημεία πάνω στην ευθεία $x = 3/2$
- Όπως και στην αμφιέρειστη δοκό, το διάγραμμα είναι παραβολή με τα κοίλα άνω.



Παράδειγμα – main

- Είναι script. Απλά τρέχει μια σειρά από εντολές.
- Δεν είναι function για να έχει input-output.
- Γρ 1: Φροντίζει να μην υπάρχουν σχήματα, μεταβλητές και εντολές από προηγούμενη εκτέλεση προγράμματος.
  - close all; Κλείνει όλα τα υπάρχοντα σχήματα.
  - clear; Καθαρίζει τις μεταβλητές από το Workspace
  - clc; Καθαρίζει τις παλιές εντολές από το Command window.
- Γρ 3-9: Παράμετροι του προβλήματος.

main.m

```
1 -    close all; clear; clc;
2
3 -    a = 3;
4 -    b = 2;
5 -    D = 10000;
6 -    v = 0.3;
7 -    q0 = -100;
8 -    max_m = 9;
9 -    max_n = 9;
```

Παράδειγμα – main

```
10
11 -    w_max = calc_w(a/2, b/2, a, b, D, q0, max_m, max_n);
12 -    disp(['Maximum displacement: w_max = ', num2str(w_max)])
```

- Γρ 11: Υπολογίζει τη μέγιστη βύθιση χρησιμοποιώντας τη συνάρτηση calc_w.
- Γρ 12: Εκτυπώνει την παραπάνω τιμή.
- Τύπος κειμένου (string): `'This is text'`
- Γενικώς για να εκτυπώσουμε κείμενο στο Command Window χρησιμοποιούμε: disp(`'This is text'`)
- Για να ενώσουμε πολλά strings: [`'string 1', 'string 2', 'string 3'`]
- Για να μετατρέψουμε μια αριθμιτική μεταβλητή *c* σε string: num2str(c)

Παράδειγμα – main

```
13
14 -    num_points_x = 20;
15 -    num_points_y = 20;
16 -    plot_w(a, b, D, q0, max_m, max_n, num_points_x, num_points_y);
17
18 -    x0 = a/2;
19 -    plot_Mx(a, b, v, q0, max_m, max_n, x0, num_points_y);
```

- Γρ 16: Καλεί την συνάρτηση plot_w για να σχεδιάσει το 3D γράφημα της βύθισης σε όλη τη πλάκα.
- Γρ 19: Καλεί την συνάρτηση plot_Mx για να σχεδιάσει το διάγραμμα ροπής στην ευθεία $x = a/2$