Πανεπιστήμιο Θεσσαλίας
Τμήμα Πολιτικών Μηχανικών

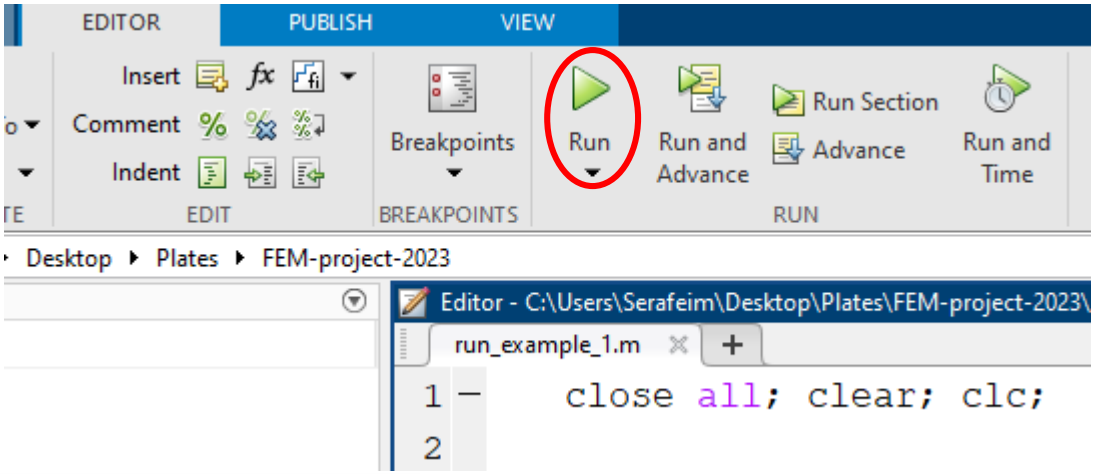
Ανάλυση επιφανειακών φορέων

Ενότητα 11:
Μέθοδος Πεπερασμένων Στοιχείων (μέρος Γ)
Προγραμματισμός της ΜΠΣ στο Matlab

Σεραφείμ Μπακαλάκος
Phd Πολιτικός Μηχανικός
2023-2024

Περιγραφή φορέα

- Script: **run_example_1.m**
- Εδώ θα:
  i. Καλέσουμε functions που θα μας βοηθήσουν να περιγράψουμε το φορέα
  ii. Καλέσουμε τη function που εκτελεί ανάλυση Πεπερασμένων Στοιχείων
  iii. Καλέσουμε συναρτήσεις που σχεδιάζουν τη βύθιση και τις ροπές
- Για τρέξουμε το script αυτό, ανοίγουμε το αρχείο και πατάμε “Run” από την ομάδα εντολών “Editor”



Περιγραφή φορέα

- Script: **run_example_1.m**
- Δίνουμε τις ιδιότητες της πλάκας και του πλέγματος
- Δεν βάζουμε μονάδες. Αντίθετα όλα τα δεδομένα που δίνουμε πρέπει να εχουν τις ίδιες μονάδες μήκους και δύναμης. Έτσι και τα αποτελέσματα θα έχουν αυτές τις μονάδες.
- **nx** και **ny** είναι ο αριθμός των κόμβων του πλέγματος κατά τους άξονες x, y αντίστοιχα.
- Προς το παρόν, θα δουλέψουμε με ομοιόμορφο κατανεμημένο φορτίο q

```
 3      % Describe the problem
 4 -    D = 5000;
 5 -    v = 0.3;
 6 -    q = 10;
 7 -    Lx = 3;
 8 -    Ly = 2;
 9 -    nx = 25;     % number of real nodes along axis x. E.g. 4, 7, 10, 16, 25
10 -    ny = 17;     % number of real nodes along axis y. E.g, 3, 5,  7, 11, 17
```

Περιγραφή φορέα

- Script: **run_example_1.m**
- Η function **create_domain(…)** παίρνει σαν input τις παραπάνω ιδιότητες και δημιουργεί 2 μεταβλητές (**domain** και **dof_supports**).

```
12      % Initialize the domain and mesh
13 -    [domain, dof_supports] = create_domain(D, v, q, Lx, Ly, nx, ny);
```

- Η μεταβλητή **domain** είναι struct, δηλαδή μια δομή δεδομένων που αποθηκεύει διάφορα άλλα δεδομένα.
- Στο **domain** αποθηκεύονται επίσης
  - Οι αποστάσεις dx, dy μεταξύ των κόμβων
  - Τα πλήθη $nelx, nely$ πεπερασμένων στοιχείων κατά τους άξονες x, y
  - Το συνολικό πλήθος $ndofs$ βαθμών ελευθερίας (degrees of freedom – dofs)

Περιγραφή φορέα

- Script: **run_example_1.m**
- Η function **create_domain(…)** παίρνει σαν input τις παραπάνω ιδιότητες και δημιουργεί 2 μεταβλητές (**domain** και **dof_supports**).

```
12      % Initialize the domain and mesh
13 -    [domain, dof_supports] = create_domain(D, v, q, Lx, Ly, nx, ny);
```

- Η μεταβλητή **dof_supports** είναι πίνακας $(ndofs \times 1)$ και περιέχει 0 ή 1 για κάθε βαθμό ελευθερίας.
  - Αν ένας β.ε. (μετακίνηση ή στροφή) δεσμευμένος, τότε υπάρχει 1 στην αντίστοιχη θέση του **dof_supports.**
  - Αν ένας β.ε. είναι ελεύθερος (όχι δεσμευμένος), τότε υπάρχει 0 στην αντίστοιχη θέση του **dof_supports**.
  - Αρχικά όλοι οι β.ε. είναι ελεύθεροι. Θα δεσμευτούν κάποιοι, όταν ορίσουμε συνοριακές συνθήκες.

Περιγραφή φορέα

- Script: **run_example_1.m**
- Η function **create_nodes_elements(…)** παίρνει σαν input το **domain** και δημιουργεί 2 μεταβλητές (**node_coordinates** και **elements**), που περιγράφουν το πλέγμα πεπερασμένων στοιχείων.

```
14 -    [node_coordinates, elements] = create_nodes_elements(domain);
```

- Η μεταβλητή **node_coordinates** είναι πίνακας $(nx \cdot ny \times 2)$ και περιέχει τις συντεταγμένες των κόμβων στο καθολικό σύστημα.
- Η μεταβλητή **elements** είναι πίνακας $(nelx \cdot nely \times 4)$ και περιέχει τους κόμβους των πεπερασμένων στοιχείων.
  - Δουλεύουμε με στοιχεία 4 κόμβων – 12 β.ε., οπότε έχουμε 4 στήλες.
  - Κάθε γραμμή αντιστοιχεί σε 1 πεπερασμένο στοιχείο.
  - Όπως και στη μέθοδο Πεπερασμένων Διαφορών, κάθε κόμβος (i, j) έχει ένα μοναδικό αναγνωριστικό αριθμό s:

$$s = (nx - 1) \cdot j + i$$

  - Αυτά τα s τοποθετούνται στον πίνακα **elements**

Περιγραφή φορέα

- Script: **run_example_1.m**
- Έπειτα περιγράφουμε τις συνοριακές συνθήκες.
- Η function **find_nodes_with_x(x0, y_lower, y_upper, domain)** βρίσκει όλους τους κόμβους με $x = x0$, y_lower ≤ y ≤ y_upper.

```
26 -    nodes_index2D = find_nodes_with_y(0, 0, Lx, domain);
```

- Τους επιστρέφει στην μεταβλητή **nodes_index2D**. Αυτή είναι ένας πίνακας με μία γραμμή για κάθε κόμβο και 2 στήλες. Η 1η στήλη είναι το i του κόμβου. Η 2η στήλη είναι το j του κόμβου
  - Π.χ.

nodes_index2D

3x2 double

	1	2
1	3	3
2	3	4
3	3	5

- Οι functions **find_nodes_with_y(…)** και **find_node_with_xy(…)** λειτουργούν αντίστοιχα με την **find_nodes_with_x(…)**

Περιγραφή φορέα

- Script: **run_example_1.m**
- Η function **apply_supports(support_text, nodes_index2D, dof_supports, domain)** παίρνει σαν input
  - Κείμενο **support_text**: 'fixed' για πάκτωση, 'simple' για απλή έδραση.
  - Τον πίνακα **nodes_index2D** με τα i, j των κόμβων που μας ενδιαφέρουν
  - Τον πίνακα **dof_supports** με τις υπαρχουσες στηρίξεις στους β.ε.
  - Τις ιδιότητες του φορέα στο **domain**
- Επιστρέφει τον πίνακα **dof_supports** ανανεωμένο, έχοντας εφαρμόσει τις στηρίξεις που ζητάμε, στους κόμβους που ζητάμε.

```
17      % Supports at edge x=0
18 -    nodes_index2D = find_nodes_with_x(0, 0, Ly, domain);
19 -    dof_supports = apply_supports('simple', nodes_index2D, dof_supports, domain);
```

```
29      % Supports at edge y=Ly
30 -    nodes_index2D = find_nodes_with_y(Ly, 0, Lx, domain);
31 -    dof_supports = apply_supports('fixed', nodes_index2D, dof_supports, domain);
```

Ανάλυση και αποτελέσματα

- Script: **run_example_1.m**
- Η function **perform_fem_analysis(node_coordinates, elements, dof_supports, domain)** εκτελεί ανάλυση με τη Μέθοδο Πεπερασμένων Στοιχείων.
- Επιστρέφει πίνακα (διάνυσμα εδώ) **Ug** με διαστάσεις (**ndofs** × **1**), που περιέχει τις μετακινήσεις και στροφές σε όλους τους κόμβους του φορέα.

```
34 —    Ug = perform_fem_analysis(node_coordinates, elements, dof_supports, domain);
```

- Το **Ug** περιέχει τις μετακινήσεις και στροφές σε όλους τους β.ε., ελεύθερους και δεσμευμένους

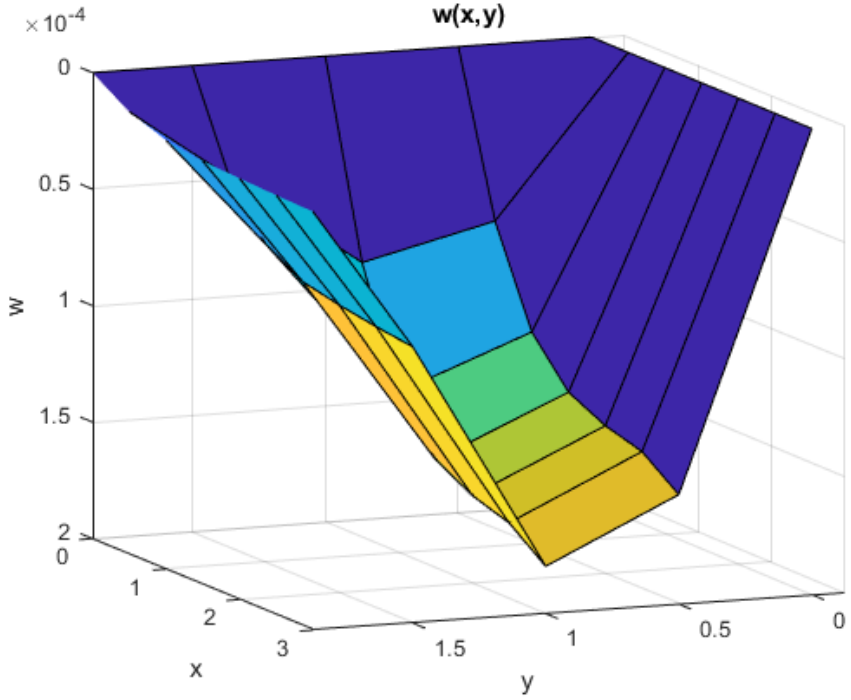
Ανάλυση και αποτελέσματα

- Script: **run_example_1.m**
- Η function **plot_w_at_nodes(Ug, domain)** σχεδιάζει το 3D διάγραμμα βυθίσεων χρησιμοποιώντας μόνο τις επικόμβιες βυθίσεις.
- Η function **plot_w(Ug, domain, n_points)** σχεδιάζει το 3D διάγραμμα βυθίσεων, αφού τις υπολογίσει σε $n\_points \times n\_points$ σημεία, τα οποία συνήθως βρίσκονται στο εσωτερικό των στοιχείων

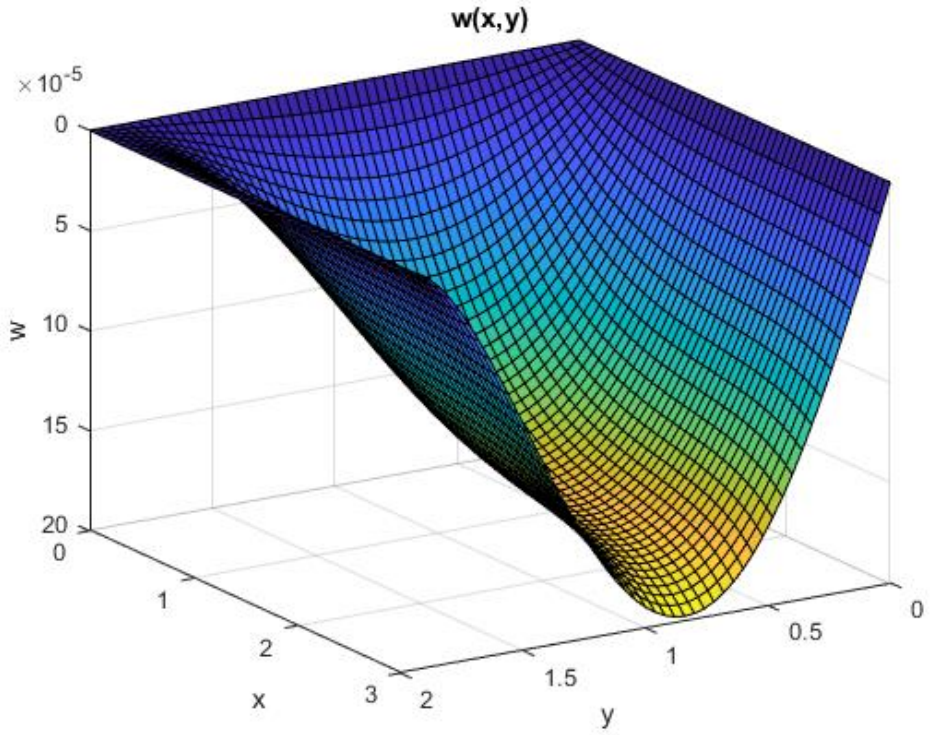
```
36      % Calculate and plot results
37 -    n_points = 51;
38 -    plot_w_at_nodes(Ug, domain);
39 -    plot_w(Ug, elements, domain, n_points);
```

Ανάλυση και αποτελέσματα

plot_w_at_nodes(Ug, domain)

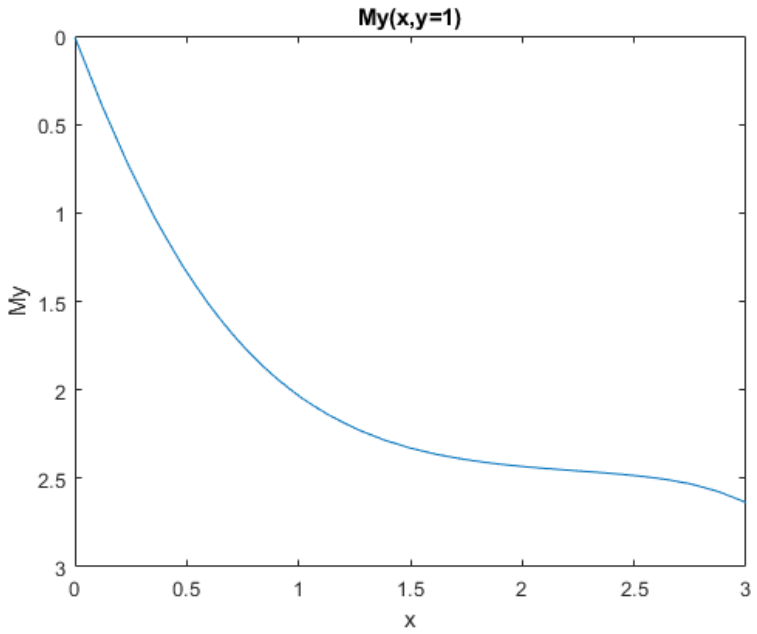


plot_w(Ug, domain)



Ανάλυση και αποτελέσματα

- Script: **run_example_1.m**
- Οι functions **plot_Mx_parallel_to_x(...)** , **plot_Mx_parallel_to_y(...)**, κτλ. σχεδιάζουν διαγράμματα ροπών πάνω σε συγκεκριμένες ευθείες.
- Για “parallel_to_x” θα σχεδιαστεί διάγραμμα πάνω στην ευθεία y=y0
- Για “parallel_to_y” θα σχεδιαστεί διάγραμμα πάνω στην ευθεία x=x0




```
40 -    plot_Mx_parallel_to_x(Ly/2, Ug, elements, domain, n_points);
41 -    plot_My_parallel_to_x(Ly/2, Ug, elements, domain, n_points);
42 -    plot_Mxy_parallel_to_x(Ly/2, Ug, elements, domain, n_points);
43 -    plot_Mx_parallel_to_y(Lx/2, Ug, elements, domain, n_points);
44 -    plot_My_parallel_to_y(Lx/2, Ug, elements, domain, n_points);
45 -    plot_Mxy_parallel_to_y(Lx/2, Ug, elements, domain, n_points);
```

Οργάνωση του κώδικα

- Μέχρι τώρα περιγράψαμε το φορέα, τον αναλύσαμε και είδαμε αποτελέσματα στο script **run_example_1.m**
- Αυτό είναι top-level κώδικας, που καλεί και συντονίζει τις υπόλοιπες functions.
- Σε ένα εμπορικό λογισμικό, θα υπάρχει GUI, αντί για αυτό το αρχείο. Συχνά όμως υπάρχει και η δυνατότητα εργασίας με script, καθώς είναι πολύ βολικά για αυτοματοποίηση.
- Το **run_example_1.m** λύνει το παράδειγμα 1. Για άλλο πρόβλημα χρειαζόμαστε αντίστοιχο script, το οποίο όμως είναι παρόμοιο. Π.χ. **run_example_2.m**
- Οι σπουδαστές καλούνται να συμπληρώσουν το script **run_project.m** για το θέμα εξαμήνου.
- Στη συνέχεια θα εξετάσουμε πιο αναλυτικά κάποια από τα functions που χρησιμοποιούνται

Υπολογισμός βυθίσεων

- Αρχείο **calc_w.m** που περιέχει την ομόνυμη function:

```
1   function [w] = calc_w(x0, y0, Ug, elements, domain)
```

- Αυτή η συνάρτηση υπολογίζει τη βύθιση ένα οποιοδήποτε σημείο (x0,y0)
- Input:
  - Οι συντεταγμένες $(x0, y0)$ του σημείου στο καθολικό σύστημα συντεταγμένων.
  - Το καθολικό διάνυσμα μετατοπίσεων **Ug.**
  - Οι κόμβοι (αναγνωριστικά s) όλων των πεπερασμένων στοιχείων.
  - οι ιδίοτητες του φορέα στο struct **domain**
- Output: η βύθιση w σε αυτό το σημείο

Υπολογισμός βυθίσεων

- Αρχείο **calc_w.m** που περιέχει την ομόνυμη function:
- Για το σημείο ξέρουμε τις καθολικές συντεταγμένες (x_0, y_0).
- Πρέπει να βρούμε:
  - Μέσα σε ποιο πεπερασμένο στοιχείο e βρίσκεται το σημείο
  - Ποιες είναι οι συντεταγμένες (x_{loc}, y_{loc}) του σημείου, στο τοπικό σύστημα συντεταγμένων του στοιχείου e.
- Αυτά πετυχαίνονται με τη συνάρτηση
  global_to_local_coordinates(x0, y0, domain)

```
15    % Find the element 'e' that contains (x0,y0). Also calculate the local
16    % coordinates (xloc, yloc) of the point (x0,y0), inside element 'e'
17 -  [e, xloc, yloc] = global_to_local_coordinates(x0, y0, domain);
```

Υπολογισμός βυθίσεων

- Αρχείο **calc_w.m** που περιέχει την ομόνυμη function:
- Η function **gather_element_displacements(Ug, e, elements, domain)** απομονώνει τις επικόμβιες βυθίσεις και στροφές του στοιχείου $\{d\}$ από το καθολικό διάνυσμα μετατοπίσεων $\{U_g\}$

```
19      % Find the vector 'd' that contains the displacements at dofs of element 'e'
20 -    d = gather_element_displacements(Ug, e, elements, domain);
```

- Η function **matrix_inverseA_element_rectangle(domain)** υπολογίζει τον πίνακα $[A]^{-1}$ για ορθογωνικό στοιχείο πλάκας με 4 κόμβους – 12 β.ε.

```
22      % Inverse of matrix 'A' needed to calculate the shape functions of the
23      % element
24 -    invA = matrix_inverseA_element_rectangle(domain);
```

$$[A]^{-1} = \frac{1}{8a^3b^3}\begin{bmatrix}
2a^3b^3 & a^3b^4 & a^4b^3 & 2a^3b^3 & a^3b^4 & -a^4b^3 & 2a^3b^3 & -a^3b^4 & -a^4b^3 & 2a^3b^3 & -a^3b^4 & a^4b^3 \\
-3a^2b^3 & -a^2b^4 & -a^3b^3 & 3a^2b^3 & a^2b^4 & -a^3b^3 & 3a^2b^3 & -a^2b^4 & -a^3b^3 & -3a^2b^3 & a^2b^4 & -a^3b^3 \\
-3a^3b^2 & -a^3b^3 & -a^4b^2 & -3a^3b^2 & -a^3b^3 & a^4b^2 & 3a^3b^2 & -a^3b^3 & -a^4b^2 & 3a^3b^2 & -a^3b^3 & a^4b^2 \\
0 & 0 & -a^2b^3 & 0 & 0 & a^2b^3 & 0 & 0 & a^2b^3 & 0 & 0 & -a^2b^3 \\
4a^2b^2 & a^2b^3 & a^3b^2 & -4a^2b^2 & -a^2b^3 & a^3b^2 & 4a^2b^2 & -a^2b^3 & -a^3b^2 & -4a^2b^2 & a^2b^3 & -a^3b^2 \\
0 & -a^3b^2 & 0 & 0 & -a^3b^2 & 0 & 0 & a^3b^2 & 0 & 0 & a^3b^2 & 0 \\
b^3 & 0 & ab^3 & -b^3 & 0 & ab^3 & -b^3 & 0 & ab^3 & b^3 & 0 & ab^3 \\
0 & 0 & a^2b^2 & 0 & 0 & -a^2b^2 & 0 & 0 & a^2b^2 & 0 & 0 & -a^2b^2 \\
0 & a^2b^2 & 0 & 0 & -a^2b^2 & 0 & 0 & a^2b^2 & 0 & 0 & -a^2b^2 & 0 \\
a^3 & a^3b & 0 & a^3 & a^3b & 0 & -a^3 & a^3b & 0 & -a^3 & a^3b & 0 \\
-b^2 & 0 & -ab^2 & b^2 & 0 & -ab^2 & -b^2 & 0 & ab^2 & b^2 & 0 & ab^2 \\
-a^2 & -a^2b & 0 & a^2 & a^2b & 0 & -a^2 & a^2b & 0 & a^2 & -a^2b & 0
\end{bmatrix}$$

Υπολογισμός βυθίσεων

- Αρχείο **calc_w.m** που περιέχει την ομόνυμη function:
- Υπολογίζουμε τον πίνακα-γραμμή

$$[x] = [1 \quad x \quad y \quad x^2 \quad xy \quad y^2 \quad x^3 \quad x^2y \quad xy^2 \quad y^3 \quad x^3y \quad xy^3]$$

  - Προσοχή: τα x,y στην παραπάνω σχέση είναι οι τοπικές συντεταγμένες

```
26      % Matrix 'matrix_x', with dimensions (1 x ndofs_of_element), needed to
27      % calculate the shape functions
28 -    matrix_x = [1, xloc, yloc, xloc^2, xloc*yloc, yloc^2, ...
29          xloc^3, xloc^2*yloc, xloc*yloc^2, yloc^3, xloc^3*yloc, xloc*yloc^3];
```

- Τέλος υπολογίζουμε τη βύθιση απο τη σχέση

$$w = [x] \cdot [A]^{-1} \cdot \{d\}$$

```
31      % Calculate the displacement w
32 -    w = matrix_x * invA * d;
```

Μητρώο δυσκαμψίας στοιχείου

- Function: **stiffness_element_rectangle(domain)**

```
1   function [ke] = stiffness_element_rectangle(domain)
```

- Υπολογίζει το μητρώο δυσκαμψίας για ένα ορθογωνικό πεπερασμένο στοιχείο πλάκας 4 κόμβων – 12 βαθμών ελευθερίας
- Input: οι ιδίοτητες του φορέα στο struct **domain**
- Output: το μητρώο δυσκαμψίας του στοιχείου **ke.**
- Οι σπουδαστές καλούνται να το υλοποιήσουν, σύμφωνα με τη θεωρία.

Ισοδύναμες δράσεις στοιχείου

- Function: **force_vector_element_rectangle(domain)**

```
1   function [re] = force_vector_element_rectangle(domain)
```

- Υπολογίζει το διάνυσμα ισοδύναμων επικόμβιων δράσεων για ομοιόμορφο κατανεμημένο φορτίο που ασκείται σε ένα ορθογωνικό πεπερασμένο στοιχείο πλάκας 4 κόμβων – 12 βαθμών ελευθερίας
- Input: οι ιδίοτητες του φορέα στο struct **domain**
- Output**:** το διάνυσμα ισοδύναμων δράσεων του στοιχείου **re.**
- Οι σπουδαστές καλούνται να το υλοποιήσουν, σύμφωνα με τη θεωρία.

Υπολογισμός ροπών

- Functions:

```
function [Mx] = calc_Mx(x0, y0, Ug, elements, domain)
function [My] = calc_My(x0, y0, Ug, elements, domain)
function [Mxy] = calc_Mxy(x0, y0, Ug, elements, domain)
```

- Υπολογίζουν τις ροπές M_x, M_y, M_{xy} σε ένα οποιοδήποτε σημείο $(x0, y0)$
- Input:
  - Οι συντεταγμένες $(x0, y0)$ του σημείου στο καθολικό σύστημα συντεταγμένων.
  - Το καθολικό διάνυσμα μετατοπίσεων **Ug.**
  - Οι κόμβοι (αναγνωριστικά s) όλων των πεπερασμένων στοιχείων.
  - οι ιδίοτητες του φορέα στο struct **domain**
- Output: η αντίστοιχη ροπή**.**
- Οι σπουδαστές καλούνται να το υλοποιήσουν, σύμφωνα με τη θεωρία.
- Υπενθύμιση: $\{M\} = [E_\kappa][\beta][A]^{-1}\{d\}$