



---

# FINITE ELEMENT METHOD (FEM)

---

ΜΕΡΟΣ Α: ΗΛΕΚΤΡΟΣΤΑΤΙΚΟ ΠΕΔΙΟ



APRIL 23, 2019

**ΠΑΛΑΣΚΟΣ ΜΑΡΙΟΣ (8492)**

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ & ΜΗΧΑΝΙΚΩΝ Η/Υ – ΤΟΜΕΑΣ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

## A.0 Υπολογισμός της χωρητικότητας μέσω των πεπερασμένων στοιχείων

Οι βαθμοί ελευθερίας (άγνωστοι) είναι οι τιμές  $\varphi_i$  της άγνωστης συνάρτησης στους κόμβους του τριγώνου και το πλήθος τους θα συμβολίζονται με  $N_f$ . Το δυναμικό αναλύεται στις συναρτήσεις βάσης ως εξής:

$$\varphi = \sum_{i=1}^{N_n} \varphi_i N_i$$

όπου  $N_i$  οι ολικές συναρτήσεις βάσης και  $N_n$  το συνολικό πλήθος των κόμβων

Υπολογισμός της συνολικής ανά μονάδα μήκους ενέργεια του ηλεκτροστατικού πεδίου:

$$W_e = \frac{1}{2} \iint_S \varepsilon |E|^2 ds = \frac{1}{2} \iint_S \varepsilon |\nabla \varphi|^2 ds = \frac{1}{2} \iint_S \nabla \varphi \cdot \varepsilon \nabla \varphi ds = \frac{1}{2} \iint_S \left( \sum_{i=1}^{N_n} \varphi_i \nabla N_i \right) \cdot \varepsilon \left( \sum_{j=1}^{N_n} \varphi_j \nabla N_j \right) ds = \\ \frac{1}{2} \sum_{i=1}^{N_n} \sum_{j=1}^{N_n} \varphi_i \left( \iint_S \nabla N_i \cdot \varepsilon \nabla N_j ds \right) \varphi_j = \frac{1}{2} \sum_{i=1}^{N_n} \sum_{j=1}^{N_n} \varphi_i S_{ij} \varphi_j = \frac{1}{2} \mathbf{F}^T \mathbf{S} \mathbf{F} \quad (1)$$

όπου το διάνυσμα-στήλη των δυναμικών σε όλους τους κόμβους έχει βρεθεί μετά τη συμπλήρωση των τιμών του διανύσματος  $X$  (από την επίλυση του αραιού συστήματος  $AX=B$ ) στις σωστές θέσεις του διανύσματος  $X0$ , που περιέχει τις τιμές των γνωστών κόμβων, ενώ έχει μηδενικά στους άγνωστους κόμβους. Οι τιμές του  $S_{ij}$  υπολογίζονται εύκολα στοιχείο προς στοιχείο (τμήματικά), καθώς προστίθονται κάθε φορά, από το υπό εξέταση στοιχείο, οι επιμέρους τιμές των τοπικών πινάκων  $S^{\{ne\}}$  (ne: ne-th triangle) στον ολικό πίνακα  $S$ . Όμως:

$$W_e = \frac{1}{2} CV^2 \xrightarrow{(1)} C = \frac{2W_e}{V^2} \Rightarrow C = \frac{\mathbf{F}^T \mathbf{S} \mathbf{F}}{V^2} \quad (2)$$

## A.1 Ομοαξονικό καλώδιο με διηλεκτρικό τον αέρα

Η χαρακτηριστική αντίσταση του ομοαξονικού καλωδίου δίνεται από τη σχέση:

$$Z_0 = \frac{60}{\sqrt{\varepsilon_r}} \ln \left( \frac{b}{a} \right) \quad (3)$$

Επομένως για  $b = 1.75 \text{ mm}$  και λύνοντας ως προς  $\alpha$  ( $\varepsilon_r = 1, Z_0 = 50 \Omega$ ), τελικά παίρνουμε  $a = 0.7605 \text{ mm}$

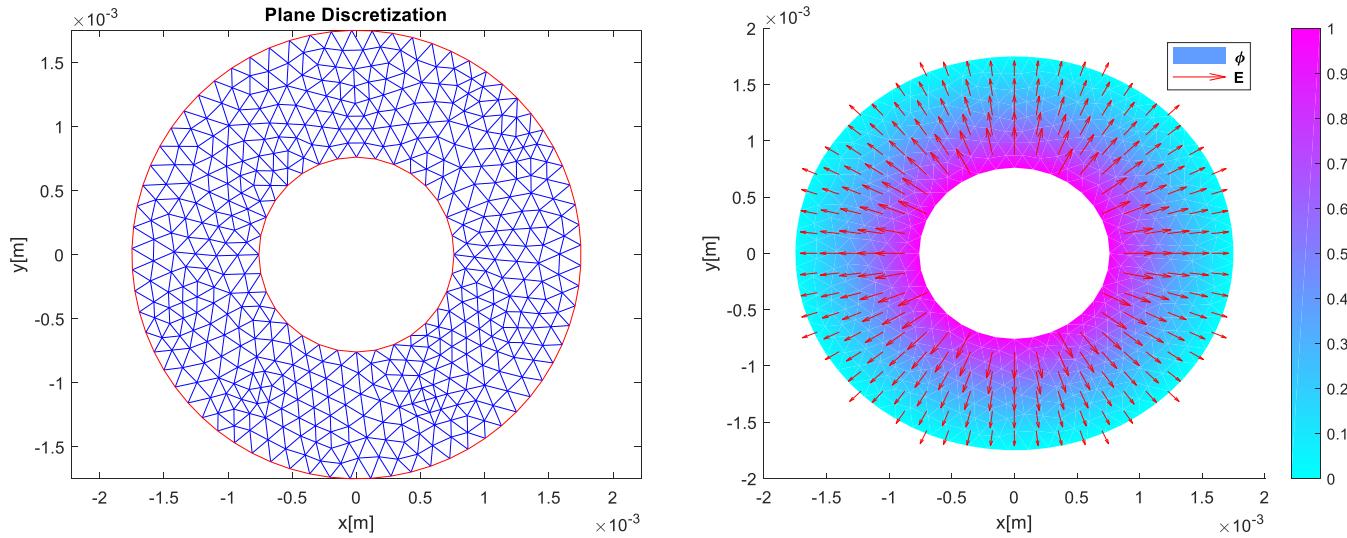
Ανατρέχοντας στο βιβλίο του κ. Τσιμπούκη, η χωρητικότητα του κυλινδρικού πυκνωτή δίνεται από τη σχέση:

$$C = \frac{2\pi\varepsilon}{\ln \left( \frac{b}{a} \right)} \xrightarrow{(3)} C = \frac{2\pi\varepsilon_r \varepsilon_0}{\frac{Z_0 \sqrt{\varepsilon_r}}{60}} \Rightarrow C_{th} = \frac{120\pi\varepsilon_0 \sqrt{\varepsilon_r}}{Z_0} \quad (4) \quad (\text{theoretical})$$

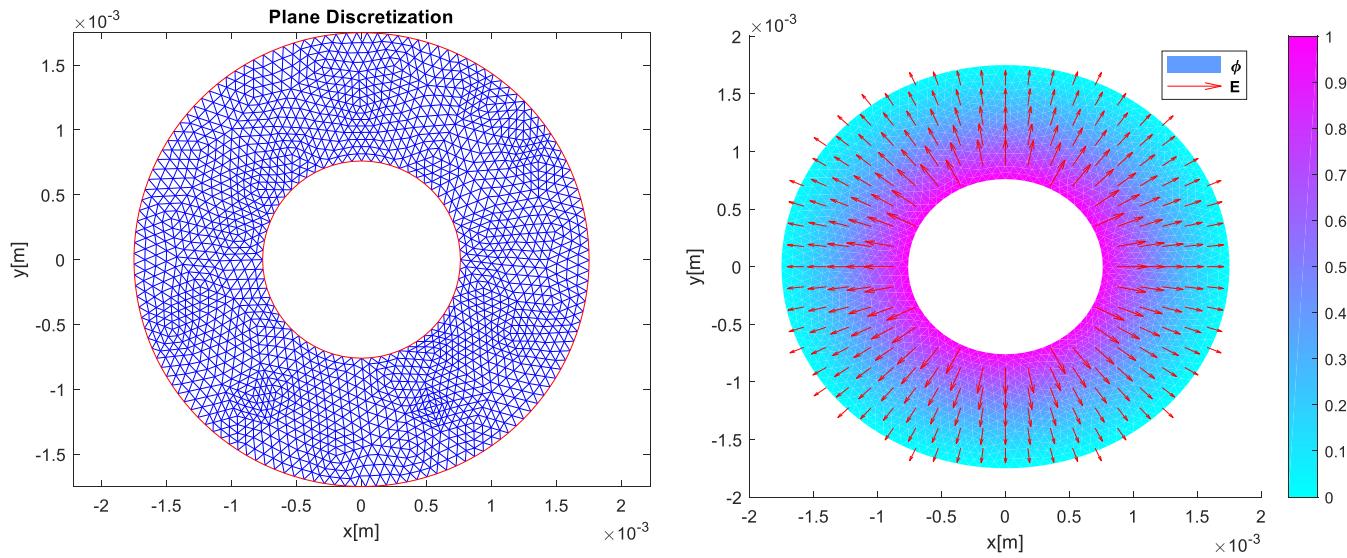
Επομένως ελέγχουμε την ορθότητα των αποτελεσμάτων συγκρίνοντας τα αποτελέσματα των (2) και (4). Αντικαθιστώντας τις δοσμένες τιμές προκύπτει ότι η θεωρητική τιμή της χωρητικότητας είναι:

$$C_{th} = 66.759 \text{ pF}$$

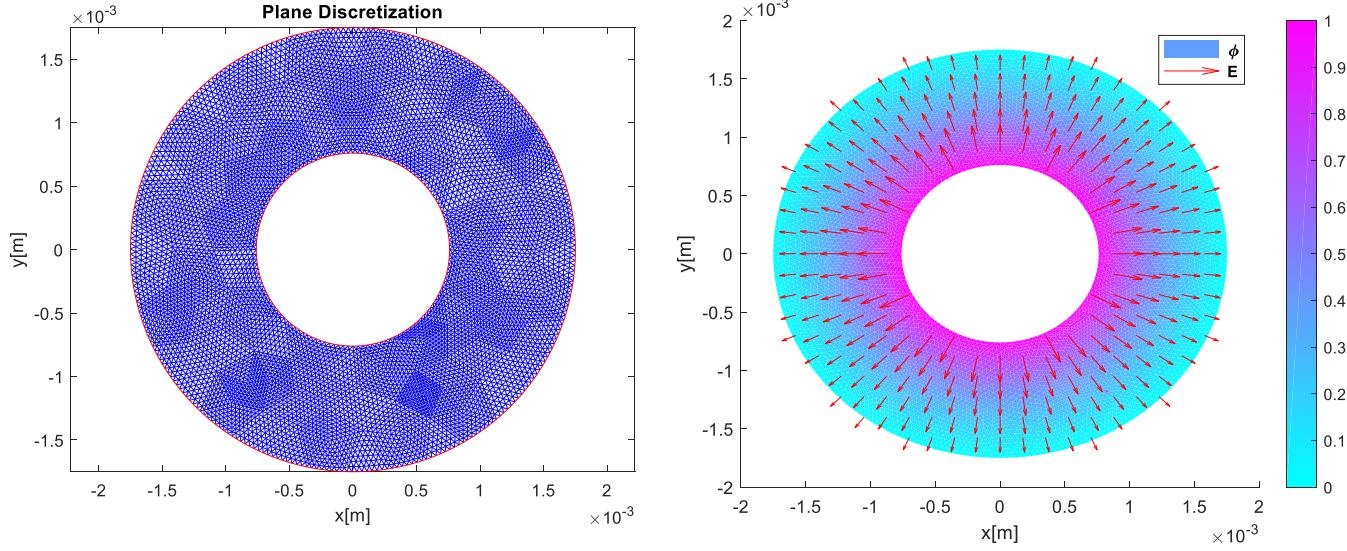
- 1 refinement ( $N_{ref} = 1$ )



- 2 refinements ( $N_{ref} = 2$ )



- 3 refinements ( $N_{ref} = 3$ )



Στον παρακάτω πίνακα παραθέτουμε τα αποτελέσματα με την εξής σειρά:

- $N_{ref}$ : αριθμός των «refinements»
- $N_f$ : πλήθος βαθμών ελευθερίας
- $C$ : χωρητικότητα που υπολογίζεται με τη μέθοδο των πεπερασμένων στοιχείων
- $(C_{th} - C)/C_{th}$ : το σχετικό σφάσμα στην χωρητικότητα
- $\tau DS$ : ο χρόνος εκτέλεσης του direct solver
- $\tau IS$ : ο χρόνος εκτέλεσης του iterative solver (αυξάνοντας το  $N_{ref}$  ορίζουμε κάθε φορά τον αριθμό Ν των επαναλήψεων, έως ότου το αποτέλεσμα εξόδου flag της συνάρτησης bicg() του matlab να δίνει αποτέλεσμα=0, το οποίο σημαίνει ότι ο αλγόριθμος έφτασε το ζητούμενο tolerance μέσα στο πλήθος Ν των επαναλήψεων -> περισσότερες λεπτομέρειες μέσα στον κώδικα στο αντίστοιχο σημείο)

| $N_{ref}$ | $N_f$ | $C$ (pF) | $\frac{C_{th} - C}{C_{th}}$ | $\tau DS$    | $\tau IS$ (N: iterations)<br>(tol = $10^{-6}$ ) | $\tau IS/\tau DS$ |
|-----------|-------|----------|-----------------------------|--------------|---|-------------------|
| 1         | 390   | 66.7570  | 3e-5                        | 0.000831 sec | 0.004703 sec (40)                               | 5.66              |
| 2         | 1660  | 66.7548  | 6.29e-5                     | 0.003258 sec | 0.011339 sec (80)                               | 3.48              |
| 3         | 6840  | 66.7542  | 7.19e-5                     | 0.018053 sec | 0.065712sec (170)                               | 3.63              |

### ΣΥΜΠΕΡΑΣΜΑΤΑ:

- Το πεδίο έχει διεύθυνση κάθετη στους δύο αγωγούς (ακτινικό), με φορά από τον εσωτερικό αγωγό (υψηλό δυναμικό) προς τον εξωτερικό αγωγό (γειωμένος). Το αποτέλεσμα αυτό είναι σύμφωνο με τον τύπο (2.75) σελ.145 του βιβλίου του κ. Τσιμπούκη. Η απόδειξη του τύπου αυτού έγινε στο μάθημα της Διάδοσης II, λύνοντας της εξίσωση Laplace στο χωρίο S (εγκάρσιο επίπεδο του κυλινδρικού πυκνωτή) μεταξύ των δύο ομόκεντρων κύκλων. Αναλύοντας τη λαπλασιανή στον αντίστοιχο τύπο από τις κυλινδρικές συν/νες και λαμβάνοντας υπόψιν:
- a) Την κυλινδρική συμμετρία:  

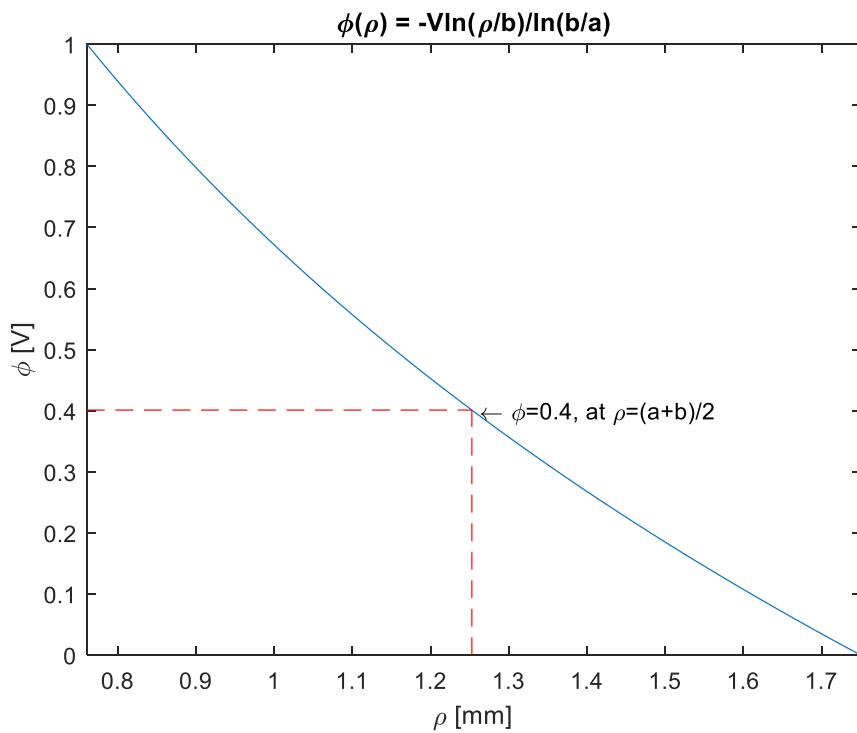
$$\varphi = \varphi(\rho), \quad \frac{\partial \varphi}{\partial \rho} = \frac{\partial \varphi}{\partial z} = 0$$
- b) Τις οριακές συνθήκες:  

$$\varphi(a) = V, \quad \varphi(b) = 0$$

καταλήξαμε στον τύπο του δυναμικού και του πεδίου για  $a \leq \varphi \leq b$ :

$$\varphi(\rho) = -\frac{V}{\ln\left(\frac{b}{a}\right)} \ln\left(\frac{\rho}{b}\right) \quad \vec{E} = -\nabla \varphi = \frac{V}{\ln\left(\frac{b}{a}\right)} \frac{1}{\rho} \hat{\rho}$$

Η αναπαράσταση την πρώτης σχέσης στο matlab φαίνεται παρακάτω: Παρατηρούμε ότι η τάση «πέφτει» απότομα λόγω των μικρών τιμών της ακτίνας που μπαίνουν ως ορίσματα στον λογάριθμο. Επιστρέφοντας στα προηγούμενα σχήματα, μπορούμε να επιβεβαιώσουμε ότι η τιμή  $\varphi = 0.4$  αντιστοιχίζεται σε μπλε ανοιχτό χρώμα (όχι γαλάζιο) και αυτό το χρώμα βρίσκεται σε κύκλο ακτίνας ίση με το ημιάθροισμα των ακτίνων του εσωτερικού και του εξωτερικού κύκλου.



- Όσο αφορά τους αλγορίθμους για την επίλυση του συστήματος, παρατηρούμε ότι η άμεση μέθοδος (DS) είναι μέχρι και 5.5 περίπου φορές πιο γρήγορη συγκριτικά με την επαναληπτική μέθοδο (IS). Για περισσότερους βαθμούς ελευθερίας, η DS παραμένει γρηγορότερη, αλλά η διαφορά μειώνεται. Ωτόσο παραμένει σημαντική, διότι φαίνεται ότι αυξάνει σημαντικά (σχεδόν διπλασιάζεται σε κάθε Nf) ο αριθμός των επαναλήψεων του αλγορίθμου, προκειμένου να φτάσει την default ανοχή (tol: tolerance).
- Άξιο παρατήρησης είναι επίσης το γεγονός ότι, με την αύξηση των βαθμών ελευθερίας, αυξάνεται και το σχετικό σφάλμα της χωρητικότητας, το οποίο βέβαια εξακολουθεί να είναι πολύ μικρό.

## ΚΩΔΙΚΑΣ

```

%% 0.General
clear;clc;close all;

V = 1; % V[Volt]
eo = 8.854187817e-12; % absolute dielectric permittivity of a vacuum
er = 1; % relative dielectric permittivity of the air
dc = er*eo; % 'd'ielectric 'c'onstant
Z0 = 50;

%% 1.Geometry
ra = 0.7605e-3;
rb = 1.75e-3;

gd = [1 1; 0 0; 0 0; ra rb]; % geometry description matrix
ns = [82 82; 49 50];
sf = 'R2-R1';

%% 2.Mesh
d1 = decsg(gd,sf,ns);
[p, e, t] = initmesh(d1);

```

```

Nref = 3; % number of "refinements"
for i=1:Nref
    [p, e, t] = refinemesh(d1,p,e,t);
end

figure, pdeplot(p,e,t);
xlabel('x[m]', ylabel('y[m]')
title('Plane Discretization')

%% 3."Re-number" the unknown nodes

Nn = size(p,2); % number of 'n'odes
Nd = size(e,2); % number of e'd'ges
Ne = size(t,2); % number of 'e'lements

% "1" for the unknown nodes, "0" for the known nodes
node_id = ones(Nn,1);
% initialize the final solution
X0 = zeros(Nn,1);

for id = 1:Nd
    % check if the id-th edge belongs to an interface
    if ( (e(6,id)==0) || (e(7,id) == 0) )
        % mark both two nodes of the edge as known
        node_id(e(1,id)) = 0;
        node_id(e(2,id)) = 0;
        % check which edges belong to the ra-circle -> +V(volt)
        % sqrt(x^2 + y^2) < (a+b)/2
        if (p(1,e(1,id))^2 + p(2,e(1,id))^2 < (ra+rb)^2/4)
            X0(e(1,id)) = V;
            X0(e(2,id)) = V;
        end
    end
end

% "re-number" the unknown nodes
index = zeros(Nn,1);
sum = 0;
for in = 1:Nn % in-th node
    if(node_id(in)>0)
        sum = sum + 1;
        index(in) = sum;
    end
end

%% 4.Main

Nf = sum; % total number of the unknown nodes
S = spalloc(Nn,Nn,7*Nn);
Sff = spalloc(Nf,Nf,7*Nf);
B = zeros(Nf,1);

for ie = 1:Ne % ie-th element
    n(1:3) = t(1:3,ie); % 3-nodes
    rg = t(4,ie); % region of the ie-th triangle
    x(1:3) = p(1,n(1:3)); y(1:3) = p(2,n(1:3)); % nodes' coordinates (x,y)
    De = det([ones(3,1) x' y']);
    Ae = abs(De/2); % area of the triangle

    % z(i) = a(i) + b(i)*x + c(i)*y, i={1,2,3}
    b = zeros(3,1);

```

```

c = zeros(3,1);
for k = 0:2
    xk = circshift(x,-k);
    yk = circshift(y,-k);
    b(k+1) = (yk(2)-yk(3))/De;
    c(k+1) = (xk(3)-xk(2))/De;
end

% S_ff & B
Se = zeros(3);
for i=1:3
    for j=1:3
        Se(i,j) = dc*(b(i)*b(j)+c(i)*c(j))*Ae;
        S(n(i),n(j)) = S(n(i),n(j)) + Se(i,j);
        if (node_id(n(i))~=0)
            if (node_id(n(j))~=0) % i,j:unknown
                Sff(index(n(i)),index(n(j))) = Sff(index(n(i)),index(n(j))) +
Se(i,j);
            else % i:unknown j:known
                B(index(n(i))) = B(index(n(i))) - Se(i,j)*X0(n(j));
            end
        end
    end
end
end

% direct solver
tic
X = Sff\B;
toc

% iterative solver: "biconjugate gradient" -> Ax = b
% [x,flag] = bicg(A,b,tol,maxit)
% If tol is [], then bicg uses the default, 1e-6
% maxit -> maximum number of iterations
% flag=0 -> bicg converged to the desired tolerance tol within maxit iterations
% flag=1 -> bicg iterated maxit times but did not converge.
tic
[X1, flag] = bicg(Sff,B,[],170); % 170 iterations for Nref=3 in order to converge
(tol=1e-6)
toc

% complete the X0 matrix with the computed values of the unknown nodes
for i = 1:Nn
    if(index(i)>0)
        X0(i) = X(index(i));
    end
end

figure, pdeplot(p,e,t,'XYdata',X0)
hold on;
[ux,uy] = pdegrad(p,t,-X0);
pdeplot(p,t,'FlowData',[ux; uy])
xlabel('x[m]'), ylabel('y[m]')
legend('\bf \phi','\bf E')

En = X0'*S*X0/2; % Energy
C = X0'*S*X0/V^2; % FEM capacity
Cth = 120*pi*eo*sqrt(er)/Z0; % theoretical capacity

%% phi -> theoretical approach
K = -V/log(rb/ra);

```

```

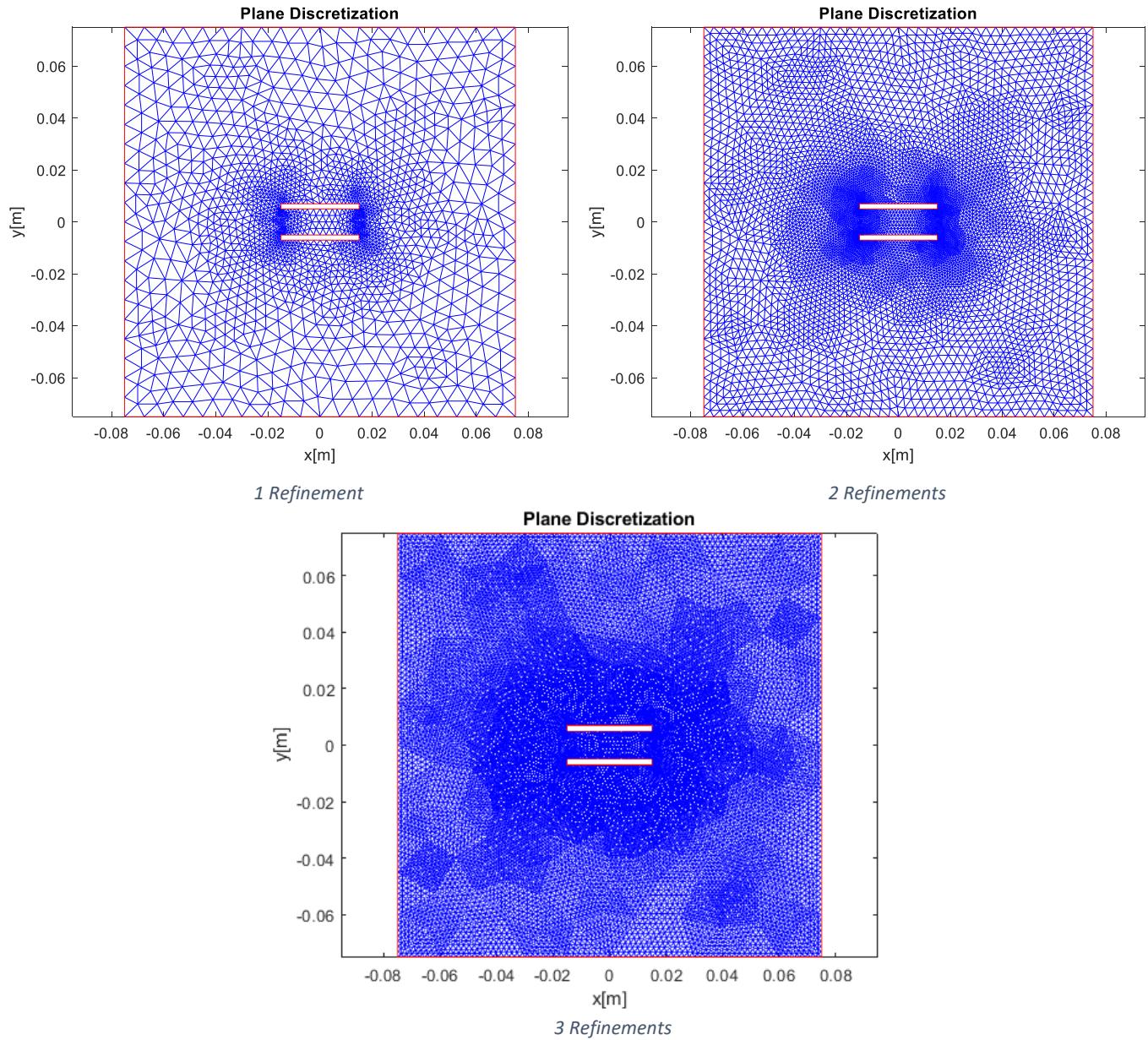
p = linspace(ra, rb, 200);
v = K*log(p./rb);

figure, plot(1000*p,v)
xlim(1000*[ra,rb])
xlabel('\rho [mm]'), ylabel('\phi [V]')
title('\phi(\rho) = -Vln(\rho/b)/ln(b/a)')

line(1000*[p(100),p(100)], [0,v(100)], 'Color','red','LineStyle', '--')
line(1000*[0,p(100)], [v(100),v(100)], 'Color','red','LineStyle', '--')
text(1000*p(100),v(100), ' \leftarrow \phi=0.4, at \rho=(a+b)/2')

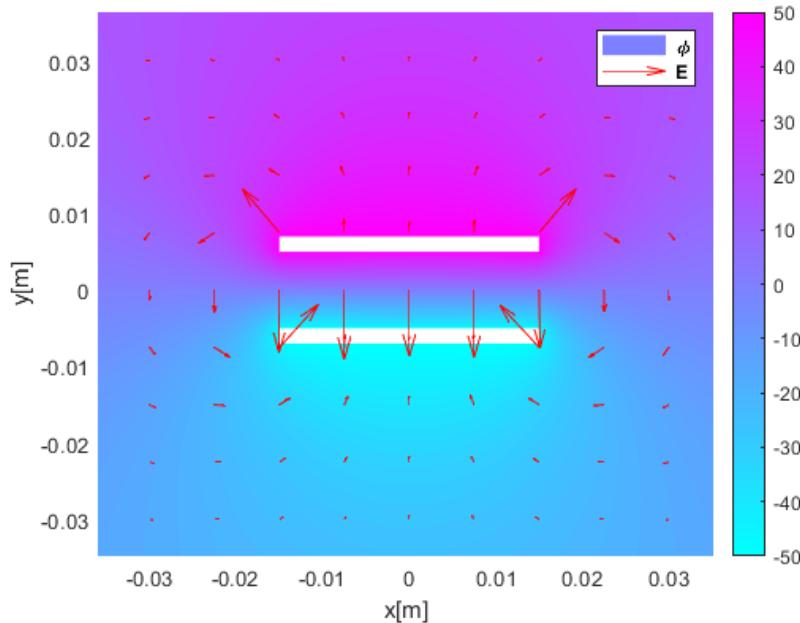
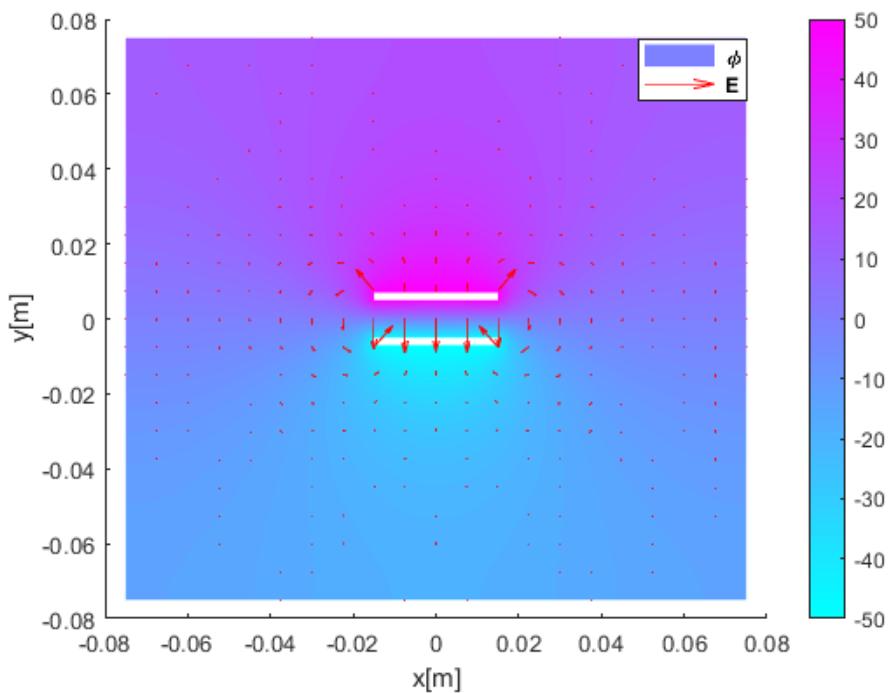
```

## Α.2 Πυκνωτής παράλληλων πλακών πεπερασμένου πλάτους



Παρακάτω παρουσιάζεται ο αντίστοιχος πίνακας με τους βαθμους ελευθερίας και τις χωρητικότητες στις τρεις περιπτώσεις:

| $N_{ref}$ | $N_f$ | $C (pF)$ |
|-----------|-------|----------|
| 1         | 1896  | 54.8462  |
| 2         | 7040  | 61.3931  |
| 3         | 27701 | 65.9795  |



### ΣΥΜΠΕΡΑΣΜΑΤΑ:

- Από τη θεωρία γνωρίζουμε ότι το πεδίο στο εσωτερικό ιδανικού πυκνωτή δίνεται από τη σχέση:

$$\vec{E} = \epsilon_r \epsilon_0 \frac{Q}{S} \hat{y}$$

όπου  $S$  είναι οι επιφάνειες (εμβαδά) των δύο πλακών και  $Q$  το ομοιόμορφα κατανεμημένο στην πανω πλάκα ( $+V/2$ ) φορτίο.

Από το παραπάνω σχήμα μπορεί πράγματι να επιβεβαιωθεί ότι το πεδίο είναι κάθετο στις δύο πλάκες με φορά από τη θετικά προς της αρνητικά φορτισμένη πλάκα του πυκνωτή, ενώ συγχρόνως είναι σταθερό μέσα στον πυκνωτή, διότι τα βέλη έχουν το ίδιο μήκος. Επιπλέον, μπορούμε να παρατηρήσουμε και τα «φαινόμενα των άκρων», τα οποία εξασθενούν όσο απομακρυνόμαστε από τον

πυκνωτή. Τέλος, επειδή το πεδίο είναι σταθερό, συνεπάγεται ότι το δυναμικό θα εξαρτάται γραμμικά από την απόσταση από την πάνω πλάκα του πυκνωτή. Αυτό επίσης φαίνεται από το σχήμα, διότι στη μέση της απόστασης των οπλισμών το δυναμικό είναι μηδεν, ενώ εκατέρωθεν της έχει ίση και αντίθετη τιμή.

- Η χωρητικότητα  $C$  του επίπεδου πυκνωτή δίνεται από τη σχέση:

$$C = \varepsilon_r \varepsilon_0 \frac{S}{d}$$

όπου  $d$  η απόσταση των οπλισμών των πυκνωτή

Για να ελέγξουμε την ορθότητα του αλγορίθμου σκεφτόμαστε ώς εξης. Μπορούμε μέσω του κώδικα στο matlab να υπολογίσουμε τη χωρητικότητα του πυκνωτη για δύο περιπτώσεις: μία με διηλεκτρικό τον αέρα ( $\widetilde{C}_1$ , για  $\varepsilon_r = 1$ ) και μία με το δοσμένο διηλεκτρικό ( $\widetilde{C}_2$ , για  $\varepsilon_r = 2.2$ ). Συνεπώς, θεωρητικά, θα πρέπει ο λόγος  $\widetilde{C}_2/\widetilde{C}_1 \xrightarrow{N_f \rightarrow \infty} C_2/C_1 = \varepsilon_r = 2.2$

Χρησιμοποιώντας τις ακριβείς τιμές του matlab, για  $N_f = 1896$  (1 refinement) προκύπτει  $\widetilde{C}_2/\widetilde{C}_1 = 1.365$ , ενώ για  $N_f = 27701$  (3 refinements)  $\widetilde{C}_2/\widetilde{C}_1 = 1.65$ . Συνεπώς υπάρχει μια βελτίωση στο λόγο των χωρητικοτήτων, η οποία αναμένουμε να προσεγγίσει το 2.2 για 5 refinements. Ωστόσο, για  $N_{ref} \geq 4$  ο αλγόριθμος έκανε αρκετή ώρα να τρέξει.

Για το λόγο αυτό έγινε η σκέψη να «πυκνωσουμε» το πλέγμα μόνο σε δύο ορθογώνιες περιοχές μήκους  $w/10$  και ύψους  $d$  και κέντρα (σημείο τομής των διαγωνίων) τα  $(w/2, 0)$  και  $(-w/2, 0)$ . Και αυτό, διότι στον κώδικα του matlab δίνων τιμή  $\varepsilon_r = 2.2$  μόνο στα τρίγωνα που βρίσκονται εξ'ολοκλήρου μέσα στο ορθογώνιο που σχηματίζεται από τους δύο πυκνωτές. Ωστόσο, υπάρχουν και τρίγωνα που έχουν μεγάλο μέρος μέσα στο ορθογώνιο αυτό, τα οποία δε λαμβάνονται υπόψιν, διότι κάποιος από τους κόμβους τους δεν ανήκει σε αυτό. Είναι φανερό, προφανώς, ότι όσο πυκνώνει το πλέγμα το σφάλμα να μειώνεται και άρα ο παραπάνω λόγος να αυξάνεται. Παρόλα αυτά, η πύκνωση στα δύο συγκεκριμένα «ορθογωνιάκια» δεν βελτιώσει σημαντικά τον παραπάνω λόγο.

Η παραπάνω διαδικασία έγινε μετά τα πρώτα refinements, χρησιμοποιώντας στην refinemesh και ένα τέταρτο όρισμα (το “re” στον δικό μου κώδικα), που είναι ένα διάνυσμα στήλη και περιέχει μία λίστα από τα τρίγωνα τα οποία θέλω να κάνω refine. Η εύρεση τους έγινε με αναζήτηση εκείνων των τριγώνων που οι κομβοί τους ανήκουν μέσα στα δύο «ορθογωνιάκια». Η υπολογιστική πολυπλοκότητα αυξάνει, σαρώνοντας όλους τους κόμβους όλων των τριγώνων και η μέθοδος δε δίνει ικανοποιητικά αποτελέσματα. Συνεπώς, θεωρώ ότι θα πρέπει να πυκνώσει συνολικά το πλέγμα και μάλιστα με προσαρμοστικό (adaptive) τρόπο, ώστε  $\frac{\widetilde{C}_2}{\widetilde{C}_1} \rightarrow 2.2$

## ΚΩΔΙΚΑΣ

```
%% 0.General
clear;clc;
V = 100; % V[Volt]
eo = 8.85418e-12; % absolute dielectric permittivity of a vacuum
er = 2.2; % relative permittivity

%% 1.Geometry
w = 0.03;
h = 2e-3;
d = 0.01;
```

```

gd1 = [w/2 w/2 -w/2 -w/2 d/2 d/2+h d/2+h d/2]';
gd2 = -gd1;
gd3 = 2.5*w*[1 1 -1 -1 -1 1 1 -1]';

gd = [3 3 3; 4 4 4; gd1 gd2 gd3]; % geometry description matrix
ns = [82 82 82; 49 50 51];
sf = 'R3-R1-R2';

%% 2.Mesh
dl = decsg(gd,sf,ns);

[p, e, t] = initmesh(dl);
Ne = size(t,2); % number of 'e'lements

Nref = 1; % number of "refinements"
for i=1:Nref
    [p, e, t] = refinemesh(dl,p,e,t);
end

% refine the triangles that belong to a (w/10)x(d)-rectangle
% center: (w/2,0) and (-w/2,0)
j = 1;
% re = zeros(1,400);
for ie = 1:Ne
    for en = 1:3
        if ( abs(p(1,t(en,ie)))>=w/2-w/20 && abs(p(1,t(en,ie)))<=w/2+w/20 &&
abs(p(2,t(en,ie)))<=d/2 )
            re(j) = ie;
            j = j + 1;
        end
    end
end

[p, e, t] = refinemesh(dl,p,e,t,re');

figure, pdeplot(p,e,t);
xlabel('x[m]', ylabel('y[m]')
title('Plane Discretization')

%% 3."Re-number" the unknown nodes

Nn = size(p,2); % number of 'n'odes
Nd = size(e,2); % number of e'd'ges
Ne = size(t,2); % number of 'e'lements

% "1" for the unknown nodes, "0" for the known nodes
node_id = ones(Nn,1);
% initialize the final solution
X0 = zeros(Nn,1);

for id = 1:Nd
    % check if the id-th edge belongs to an interface
    if ( (e(6,id)==0) || (e(7,id) == 0) )

        % The nodes of the edge are known only if they belong to the
        % "internal" interface, (i.e. capacitor). Thus, only the nodes
        % that belong to an interface, inside a (WxD)-rectangle (W>D), are KNOWN.
        % *****SOS*****
        % We assume that the "external" nodes are UNKNOWN, because of
    end
end

```

```

% the Neumann boundary conditions applied to that interface.

if ( abs(p(1,e(1,id)))<w && abs(p(2,e(1,id)))<d )
    % mark both two nodes of the edge as known
    node_id(e(1,id)) = 0;
    node_id(e(2,id)) = 0;
    % check which conductor we refer to
    if ( p(2,e(1,id))>0 ) % conductor with y>0
        X0(e(1,id)) = V/2;
        X0(e(2,id)) = V/2;
    else
        X0(e(1,id)) = -V/2;
        X0(e(2,id)) = -V/2;
    end
end
end

% "re-number" the unknown nodes
index = zeros(Nn,1);
sum = 0;
for in = 1:Nn % in-th node
    if(node_id(in)>0)
        sum = sum + 1;
        index(in) = sum;
    end
end

%% 4.Main
Nf = sum; % number of unknown nodes
S = spalloc(Nn,Nn,7*Nn);
Sff = spalloc(Nf,Nf,7*Nf);
B = zeros(Nf,1);

for ie = 1:Ne % ie-th element

    n(1:3) = t(1:3,ie); % 3-nodes
    rg = t(4,ie); % region of the ie-th triangle
    x(1:3) = p(1,n(1:3)); y(1:3) = p(2,n(1:3)); % nodes' coordinates (x,y)
    if ( max(abs(x))<=w/2 && max(abs(y))<=d/2 )
        dec = er*eo; % dielectric of the capacitor
    else
        dec = eo; % dialectric outside the capacitor
    end
%
    dec = eo;
    De = det([ones(3,1) x' y']);
    Ae = abs(De/2); % area of the triangle

    % z(i) = a(i) + b(i)*x + c(i)*y, i={1,2,3}
    b = zeros(3,1);
    c = zeros(3,1);
    for k = 0:2
        xk = circshift(x,-k);
        yk = circshift(y,-k);
        b(k+1) = (yk(2)-yk(3))/De;
        c(k+1) = (xk(3)-xk(2))/De;
    end

    % S_ff & B
    Se = zeros(3);
    for i=1:3
        for j=1:3

```

```

Se(i,j) = dec*(b(i)*b(j)+c(i)*c(j))*Ae;
S(n(i),n(j)) = S(n(i),n(j)) + Se(i,j);
if (node_id(n(i))~=0)
    if (node_id(n(j))~=0) % i,j:unknown
        Sff(index(n(i)),index(n(j))) = Sff(index(n(i)),index(n(j)))+ Se(i,j);
    else % i:unknown j:known
        B(index(n(i))) = B(index(n(i))) - Se(i,j)*X0(n(j));
    end
end
end
end

% direct solver
X = Sff\B;

% complete the X0 matrix with the computed values of the unknown nodes
for i = 1:Nn
    if(index(i)>0)
        X0(i) = X(index(i));
    end
end

figure, pdeplot(p,e,t,'XYdata',X0)
hold on;
[ux,uy] = pdegrad(p,t,-X0);
pdeplot(p,t,'FlowData',[ux; uy])
xlabel('x[m]'), ylabel('y[m]')
legend('\bf \phi','\bf E')

En = X0'*S*X0/2; % Energy
C = X0'*S*X0/V^2; % Capacity

```