ESCOLA POLITÉCNICA DA UNIVERSIDADE DE SÃO PAULO

PTC3213 - 2024

13 de outubro de 2024

PRIMEIRO EXERCÍCIO COMPUTACIONAL
MÉTODO DAS DIFERENÇAS FINITAS

Professor: **Trintinália** Turma: **1** Grupo: **xN**

Henrique Penna Ceravolo Soares	9853360
Hugo Dos Reis	12544308
Narayan Shimanoe Lisboa	14600141

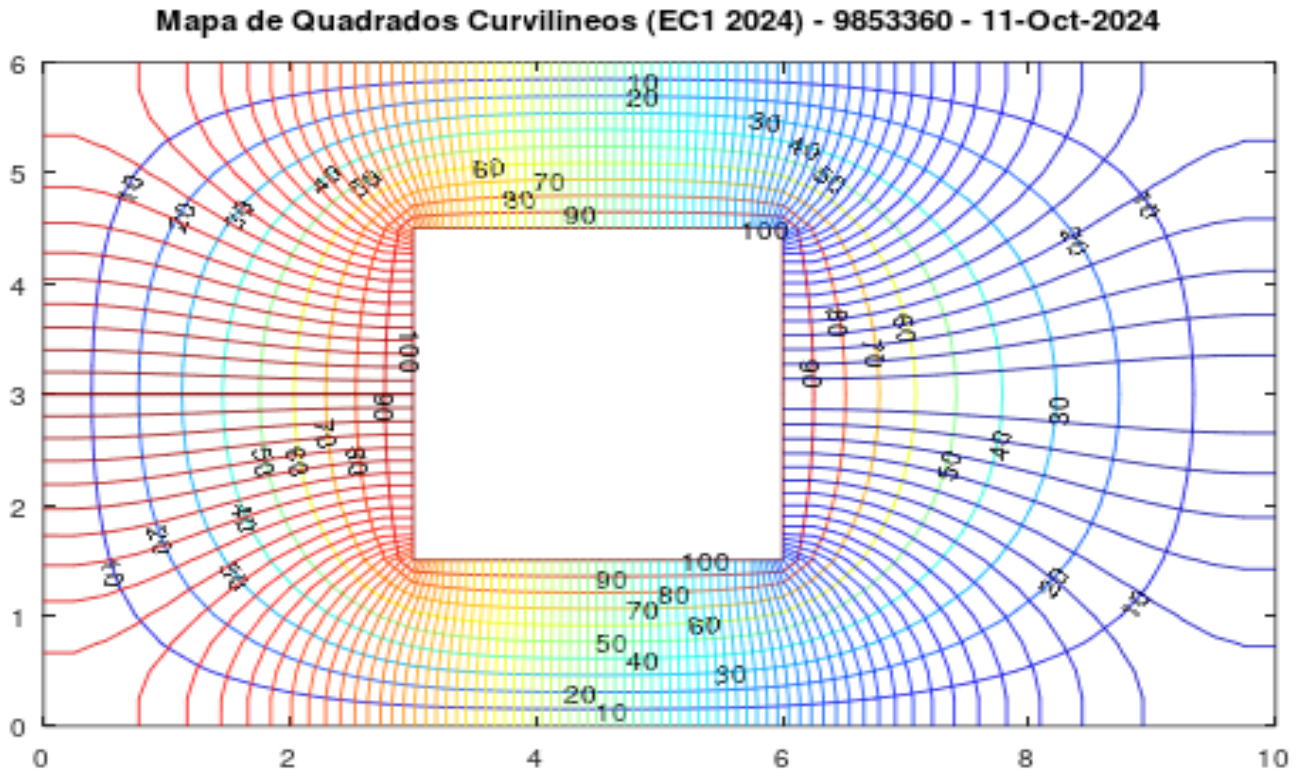
1. Dados do problema

0	6	0	6	3		0	6	3
a (cm)	***b*** (cm)	***c*** (cm)	***d*** (cm)	***g*** (cm)	***h*** (cm)	ε_r	σ (mS/m)	σ_{dual} (mS/m)
10	6	3	3	3	1,5	2	3,0	3,5

2. Resultados Numéricos

R (Ω)	***C*** (pF)	ρ_{Smin} (nC/m^2)	Tubos de corrente	R_{dual} (Ω)
39.099	226.46	-172.96	85.254	1217.9

3. Mapa dos Quadrados Curvilíneos



4. Script Octave

```
%%%%
////////////////////////////////////////////////////////////////////////////

%%%%        PTC3213 - EC1 - 2024 - DIFERENÇAS FINITAS

%%%%    Turma 1

%%%%     9853360 Henrique Penna Ceravolo Soares

%%%%     12544308 Hugo dos Reis

%%%%     14600141 Narayan Shimanoe Lisboa

%%%%

%%%%

%%%%
////////////////////////////////////////////////////////////////////////////

clear;

clf;

%%%

%%%  A linha 15 (pkg install...) deve ser executada uma unica vez no

%%% GNU Octave e necessita de conexao aa internet. Pode tambem ser executada
fora do programa.

%%%

warning ("off");

%pkg install -local -forge   matgeom; %% descomentar para rodar a 1a vez,
somente

pkg load matgeom; % executar este comando apenas 1 vez, após abrir o Octave

warning ("on");

clc;

%%%  ======================================================================

%%%   Dados de entrada
```

```
%%%

NUSP = 9853360  ; % NUSP do 1o aluno (ordem alfab.)

%

a= 10   ; # dimensões em cm

b=  6   ;

c=   3   ;

d=   3   ;

g=   3   ;

h=(b-d)/2;

epsr=   2   ;

sigma=    3.0e-3   ;  # S/m

sigma_dual=   3.5e-3   ; # S/m

eps0=   1.32812817264306e-11   ;  % F/m

Vmin=  0 ;     % Volts

Vmax=   100 ;   % Volts

%%%

%%%  ========================================================================

%%%

%%%            Definicao do dominio

%%%%

%%%% A variavel dx abaixo e a discretizacao utilizada. Valores diferentes

%%%% daqueles sugeridos abaixo nao funcionarao. Diminua o dx para gerar a

%%%% versao final a ser entregue.

%dx=0.05; % Tempo de execucao MUITO longo!!

%dx=0.1;   % Tempo de execucao longo!!

%dx=0.25;  % recomendado para a versao final
```

```
dx=0.25;      %% Mude para dx=0.25 somente quando for gerar os resultados finais!!!

erro=0.0;

start=start_Dual= 50;

iter=0;

dy=dx;

lx=a;

ly=b;

Nx=round(lx/dx)+1;

Ny=round(ly/dx)+1;

ring1= [0 0; lx 0; lx ly; 0 ly; 0 0];

ring2=[g h; g h+d; g+c h+d; g+c h; g h];

polyg={ring1,ring2};

verts = polygonVertices(polyg);

xgv=((1:Nx)-1)*dx;

ygv=((1:Ny)-1)*dx;

[x,y]=meshgrid(xgv,ygv);

verts1 = polygonVertices(ring1);

verts2 = polygonVertices(ring2);

xv1=verts1(:,1);

yv1=verts1(:,2);

xv2=verts2(:,1);

yv2=verts2(:,2);

[in1,on1] = inpolygon(x,y,xv1,yv1);

[in2,on2] = inpolygon(x,y,xv2,yv2);

%%%
```

```
%%%    Atribui Condicoes de contorno
%%%
r=find(in1&~in2|on2); % tudo
p=find(in1&~on1&~in2); %so  nos internos
q=find(on1|on2); %so fronteira
iVmax=find(on2);
iFuro=find(in2&!on2);
Phi_prev=zeros(size(x));
Phi_new=zeros(size(x));
Phi_new(iVmax)= Vmax;
Phi_new(iFuro)= NaN;
Phi_new(p)= start;
%%% ========================================================================
%%%
%%% Contador de iteracoes
iter=0;
%%% Erro maximo entre Phi_new e Phi_prev
erro=max(max(abs(Phi_new-Phi_prev)));
%%%
%%%             Laco iterativo - Metodo das Diferencas Finitas
%%%
while(erro > 1e-4 && iter < 1e4)% Executa ate convergir ou atingir o maximo de
iteracoes
    iter=iter+1; % Incrementa iteracao
%%%    Atualiza o potencial dos nos internos pela media dos 4 vizinhos - Eq.
Laplace - M.D.F.
    for k=1:size(p,1);
```

```
        [i,j]=ind2sub(size(x),p(k));

Phi_new(i,j)=(Phi_new(i-1,j)+Phi_new(i+1,j)+Phi_new(i,j-1)+Phi_new(i,j+1))/4;
    end
%%%% Calcula maximo erro entre Phi_atual e Phi_prev de todo o dominio
    erro=max(max(abs(Phi_new-Phi_prev)));
    eps(iter)=erro;
%%%%    Atualiza a matriz de potenciais
    Phi_prev=Phi_new;


end
niter1=iter;
if (niter1 == 1e4 && erro > 1e-4)
       disp([' Numero maximo de iteracoes atingido sem convergencia :', num2stg(niter1), '   iteracoes \? Erro: \n', num2str(erro), 'Os resultados podem nao ter significado!\n']);
end
%%%
%%%
%%% Problema Dual (Somente para tracado dos Quadrados Curvilineos!)
%%%
%%% Atribui Condicoes de Contorno
iyDual=find( (y(:,:) < ly/1.999) & (y(:,:) > ly/2.001) );
iVmaxdual=find( (x(iyDual) > (-0.01)) & (x(iyDual) < (1.0001*g)));
i0=find( (x(iyDual)> (0.9999*(g+c))) & (x(iyDual)< (1.0001*lx)) );
xfe=find(      x(iVmax)<   1.0001*min(x(iVmax))   );   xfd=find(      x(iVmax)> 0.9999*max(x(iVmax)) );
```

```
yfa=find(      y(iVmax)>   0.9999*max(y(iVmax))   );   yfb=find(      y(iVmax)<
1.0001*min(y(iVmax)) );

tol=1e-4;

for k=1:size(iVmax,1);

            if   (   abs(   x(iVmax(k))-min(x(iVmax))   )<   tol   &&   abs(
y(iVmax(k))-min(y(iVmax)) )< tol)

             [ieb,jeb]=ind2sub(size(x), iVmax(k));

              elseif   (abs(   x(iVmax(k))-min(x(iVmax))   )<   tol   &&   abs(
y(iVmax(k))-max(y(iVmax)) )< tol)

            [iea,jea]=ind2sub(size(x), iVmax(k));

             elseif   (   abs(   x(iVmax(k))-max(x(iVmax))   )<   tol   &&   abs(
y(iVmax(k))-min(y(iVmax)) )< tol)

             [idb,jdb]=ind2sub(size(x), iVmax(k));

              elseif   (abs(   x(iVmax(k))-max(x(iVmax))   )<   tol   &&   abs(
y(iVmax(k))-max(y(iVmax)) )< tol)

            [ida,jda]=ind2sub(size(x), iVmax(k));

    end

 end


Dual_prev=zeros(size(x));

Dual_new=Dual_prev;

Dual_new(r)= -1;

Dual_new(iFuro)= NaN;

Dual_new(iyDual(iVmaxdual))=Vmax;

Dual_new(iyDual(i0))=Vmin;

p2=find(Dual_new(p) < 0);

Dual_new(r)= start_Dual;

Dual_new(iFuro)= NaN;
```

```
Dual_new(iyDual(iVmaxdual))=Vmax;

Dual_new(iyDual(i0))=Vmin;

%%% Contador de iteracoes - dual

iter2=0;

%%% Erro maximo entre Phi_new e Phi_prev (Dual)

erro2=max(max(abs(Dual_new-Dual_prev)));

%

%%%       Laco iterativo (Problema Dual) - MDF

%

while(erro2 > 1e-3 && iter2 < 1e4)% Executa ate convergir ou atingir o maximo
de iteracoes

    iter2=iter2+1; % Incrementa iteracao

%%%   Atualiza o potencial das fronteiras

    Dual_new(1,:)=Dual_prev(2,:);

    Dual_new(Ny,:)=Dual_prev(Ny-1,:);

    Dual_new(:,1)=Dual_prev(:,2);

    Dual_new(2:Ny-1,Nx)=Dual_prev(2:Ny-1,Nx-1);

    for k=2:size(xfe,1)-1

        [ie,je]=ind2sub(size(Dual_new), iVmax(xfe(k)));

        Dual_new(ie,je)=Dual_new(ie,je-1);

    end

    for k=2:size(xfd,1)-1

        [id,jd]=ind2sub(size(Dual_new), iVmax(xfd(k)));

        Dual_new(id,jd)=Dual_new(id,jd+1);

    end

    for k=2:size(yfb,1)-1
```

```
        [ib,jb]=ind2sub(size(Dual_new), iVmax(yfb(k)));
        Dual_new(ib,jb)=Dual_new(ib-1,jb);
    end
    for k=2:size(yfa,1)-1
        [ia,ja]=ind2sub(size(Dual_new), iVmax(yfa(k)));
        Dual_new(ia,ja)=Dual_new(ia+1,ja);
    end
    Dual_new(iyDual(iVmaxdual))=Vmax;
    Dual_new(iyDual(i0))=Vmin;
%%%%
%%%% Atualiza o potencial dos nos internos pela media dos 4 vizinhos - Eq. Laplace - M.D.F.
    for k=1:size(p2,1);
        [i,j]=ind2sub(size(x),p(p2(k)));

Dual_new(i,j)=(Dual_new(i-1,j)+Dual_new(i+1,j)+Dual_new(i,j-1)+Dual_new(i,j+1))/4;
    end
%%% Cantos

Dual_new(ieb,jeb)=(Dual_new(ieb-1,jeb)+Dual_new(ieb+1,jeb)+Dual_new(ieb,jeb-1)+Dual_new(ieb,jeb+1))/4;

Dual_new(iea,jea)=(Dual_new(iea-1,jea)+Dual_new(iea+1,jea)+Dual_new(iea,jea-1)+Dual_new(iea,jea+1))/4;

Dual_new(idb,jdb)=(Dual_new(idb-1,jdb)+Dual_new(idb+1,jdb)+Dual_new(idb,jdb-1)+Dual_new(idb,jdb+1))/4;
```

```
Dual_new(ida,jda)=(Dual_new(ida-1,jda)+Dual_new(ida+1,jda)+Dual_new(ida,jda-1)
+Dual_new(ida,jda+1))/4;

%%% Calcula maximo erro entre Phi_atual e Phi_prev de todo o dominio

    erro2=max(max(abs(Dual_new-Dual_prev)));

    eps2(iter2)=erro2;

%%% Atualiza a matriz de potenciais

    Dual_prev=Dual_new;

%%%

end

niter2=iter2;

if (niter2 == 1e4 && erro2 > 1e-3)

       disp([' Numero maximo de iteracoes atingido sem convergencia :',
num2stg(niter2), '  iteracoes \? Erro: \n', num2str(erro2), 'Interprete este
resultado com ressalvas!\n']);

end

%%%=========================================================================

%%%

%%%               DADOS DE SAIDA

%%%

%%%=========================================================================

%%%

%%%      CORRENTE TOTAL (A)

%%

Somat=sum(Phi_new(2,:))+sum(Phi_new(Ny-1,:))+sum(Phi_new(:,2))+sum(Phi_new(:,N
x-1));

I=  sigma * 1 * Somat  ;

%%%
```

```
%%%       RESISTENCIA em ohms

%%%

R=  (Vmax - Vmin) / I     ;

%%%

%%%        CAPACITANCIA em pF

%%%

Cap=  ((epsr * eps0) * 1 * Somat * 1e12) / (Vmax - Vmin) ;

%%%

%%%     RESISTENCIA DUAL em ohms

%%%

Rdual=  (1 / (R * 1 * sigma * sigma_dual * 1)) / 2  ;

%%%

%%%    VETOR DESLOCAMENTO

%%%

Dn=[Phi_new(2,1:Nx-1),Phi_new(1:Ny-1,Nx-1)',Phi_new(Ny-1,1:Nx-1),Phi_new(1:Ny-
1,2)']*epsr*eps0/dx*100;

%%

%%   Densidade de carga mínima em nC/m^2

%%

Rho_s_min=  -max(Dn) * 1e9  ;

%%

%%  Numero de tubos de corrente

%%

nsnp=    R * sigma * 1   ;

ntubos=10/nsnp;  %% CORRIGIDO

%%%%=============================================================================
```

```
%%%%                IMPRESSAO DE RESULTADOS NO TERMINAL

%%%%                    ATENCAO para as unidades:

%%%%           R e Rdual em ohms      Cap em pF     Rho_s   em nC/m^2

%%%%

%%%%

fprintf('\n\n nUSP: %d\n R = %g ohms\n C = %g pF\n Rho_s_min = %g nC/m^2\n
Rdual = %g ohms\n Tubos: %g\n', NUSP, R, Cap, Rho_s_min,Rdual,floor(ntubos) );

%%%

%%%

FIG=figure (1);

%%

%%%            TRACADO DE EQUIPOTENCIAIS

%%%

V=0:10:Vmax;

colormap cool;

[C,H]=contour(x,y, Phi_new  ,  V  );

clabel(C,V);

axis('equal');

hold on

%%%

%%%   EQUIPOTENCIAIS PROBLEMA DUAL (para tracado dos quadrados curvilineos)

%%%

%%%

%%%

deltaV=  (Vmax - Vmin) / floor(ntubos) ;

V=0:deltaV:Vmax;
```

```
colormap jet;

contour(x,y, Dual_new   ,  V );

axis('equal');

strusp=sprintf('%d',NUSP);

titulo=['Mapa de Quadrados Curvilineos (EC1 2024) - ', strusp, ' - ', date()];

title(titulo);

hold off

%%%

%%%      ARQUIVO DE SAIDA COM O MAPA DOS QUADRADOS CURVILINEOS

%%%(Grava na pasta exibida no Navegador de Arq. da interface gráfica do
Octave)

%%%

arq=['EC1_2021_QC_',strusp,'.png'];

print(FIG,arq);

%%%%
================================================================================

%%%%  FIM

%%%%
```