ESCOLA POLITÉCNICA DA USP
DEPARTAMENTO DE ENGENHARIA
MECATRÔNICA E DE SISTEMAS MECÂNICOS

Mecânica Computacional

PMR3401

Exercício Programa 2: Método de Diferenças Finitas
(MDF)
06/2020

Alessandro Brugnera Silva – 10334040
Vitor Luiz Lima Carazzi – 9834010

Introdução

Com o objetivo de analisar o esforço gerado pelo vento no telhado de uma construção, foi utilizado um modelo bidimensional da construção de um silo para simular o escoamento de ar em volta da estrutura. Foi considerado que o escoamento é irrotacional, invíscido, em regime permanente, de ar como fluido compressível e sem efeitos gravitacionais.

A análise foi feita com o Método das Diferenças Finitas para se obter as linhas de corrente de escoamento e, com isso, a velocidade e o campo de pressão. A partir do campo de pressão foi obtida a força vertical causada pelo escoamento. Além disso, também foram estudadas as trocas térmicas entre o galpão e o ar com o uso do MDF.

Modelagem do sistema

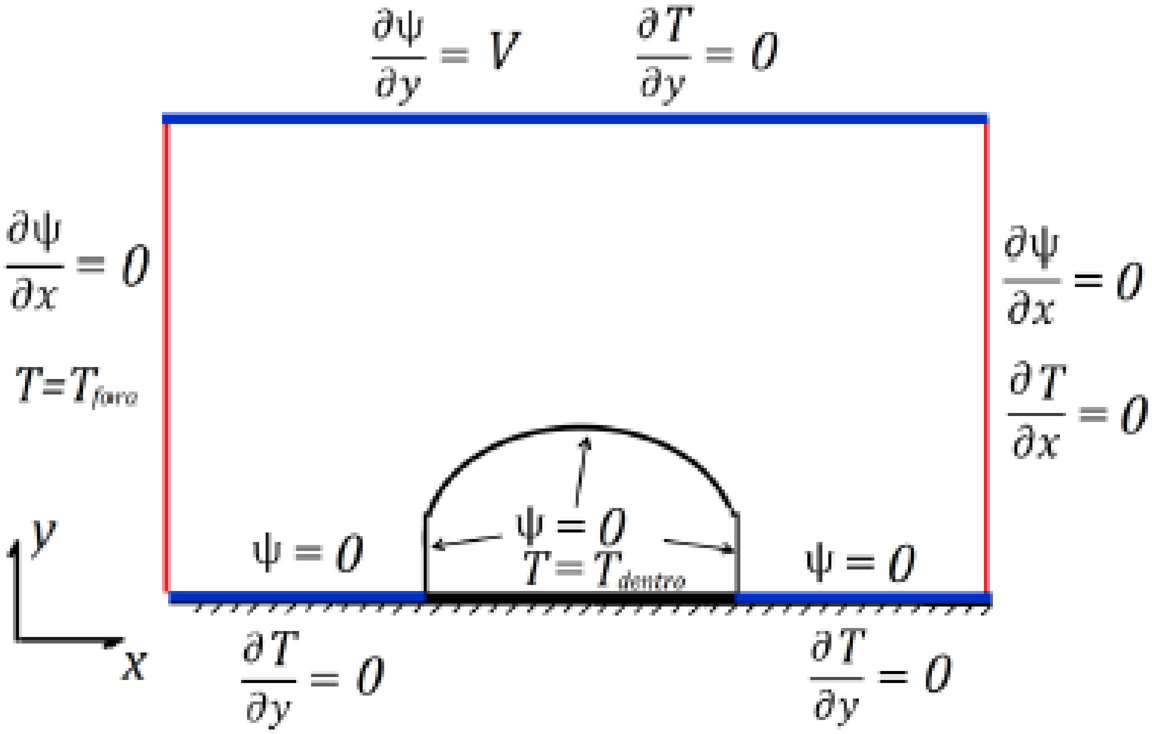


Figura 1: Malha e condições de contorno

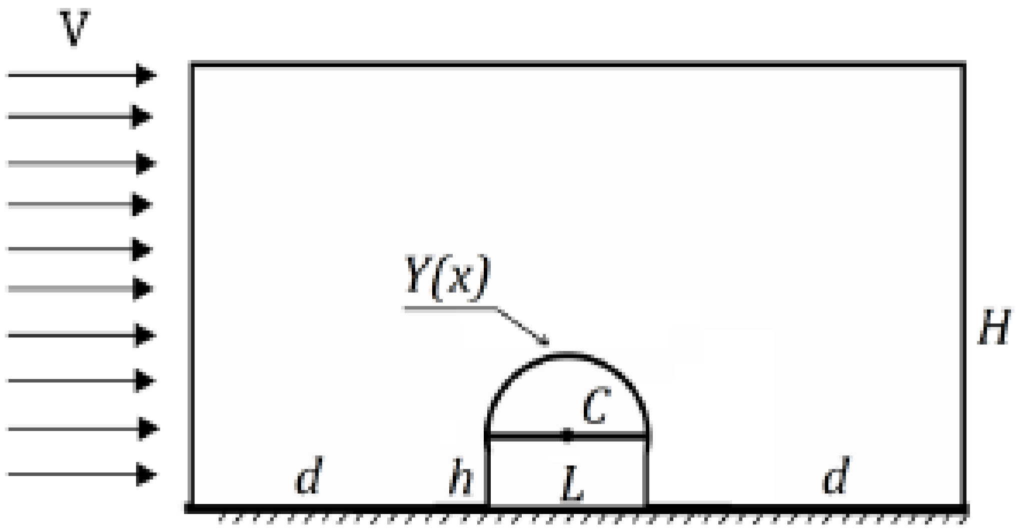


Figura 2: Domínio do problema

A Figura 1 mostra as condições de contorno dadas pelo enunciado do EP2 e a Figura 2 mostra como o problema foi modelado bidimensionalmente, como um galpão sendo atingido lateralmente pelo vento.

Equações

1. Equação de escoamento potencial compressível:

$$\frac{\partial^2 \psi}{\partial^2 x} + \frac{\partial^2 \psi}{\partial^2 y} = 0$$

2. Utilizando o MDF com as condições de contorno dadas, a partir do campo de velocidade **u**, com a equação de Bernoulli, obtemos a variação de pressão no domínio:

$$p(x,y) - p_{atm} = \rho \frac{\gamma_{ar} - 1}{\gamma_{ar}} \left(\frac{(0)^2}{2} - \frac{(\sqrt{u(x,y)^2 + v(x,y)^2})^2}{2} \right)$$

3. Cálculo da deistribuição de temperaturas a partir do campo de velocidade:

$$k\nabla^2 T - \rho c_p \boldsymbol{u}.\nabla T = 0$$

4. Fluxo de calor, em W/m^2, através das superfícies do prédio:

$$\vec{Q}\Big|_{prédio} = -k\frac{\partial T}{\partial n}\vec{n} = -(k\nabla T.\vec{n})\vec{n};$$

5. Quantidade de calor total, em W, trocada pela área A do prédio com o ar:

$$q = \int_A \vec{Q}.\vec{n}dA = -\int_A k\frac{\partial T}{\partial n}dA = -\int_A k\nabla T.\vec{n}dA$$

Dados

- Velocidade do vento: V = 100km/h;
- Dimensões: h=3m, d=5h, L=2h e H=8m;
- Comprimento da estrutura: 60m;
- Temperaturas: T_{dentro}=40°C e T_{fora}=20°C;
- Propriedades do ar: ρ=1,25kg/m^3, γ_{ar}=0,026W/(m.K) e c_{Par}=1002J/(kg.K);

Domínio

O domínio foi discretizado em diversos pontos com tamanho quadrado Δx=Δy. Com atenção especial às bordas e cantos da área externa; além do galpão e do telhado. O telhado foi discretizado com o primeiro ponto externo a equação:

$$Y(x) = \sqrt{\left(\frac{L}{2}\right)^2 - \left(x - d - \frac{L}{2}\right)^2} + h \;\; \text{para } d \leq x \leq d + L.$$

Assim o telhado fica melhor modelado quanto menor Δx.

Bordas e Cantos

Como no método de Liebmann, os pontos internos são as estimados pela média dos 4 pontos vizinhos ao ponto calculado; nas bordas e cantos a equação de Taylor tem que ser aplicada para calcular em bordas e cantos, aplicando as devidas aproximações regressivas, centrais ou progressivas - tanto para calcular a função corrente quanto a temperatura.

Função corrente

- Cantos
    - Superiores
        - Esquerdo

$$\Psi_{1,m} = \frac{\Psi_{1,m-1} + \Psi_{2,m} + \Delta y * V}{2}$$

        - Direito

$$\Psi_{n,m} = \frac{\Psi_{n,m-1} + \Psi_{n-1,m} + \Delta y * V}{2}$$

- Bordas
    - Superior (exceto bordas)

$$\Psi_{i,m} = \Psi_{i,m-1} + \Delta y * V + \frac{\Psi_{i+1,m} + \Psi_{i-1,m}}{2}$$

    - Esquerda

$$\Psi_{1,j} = \Psi_{2,j} + \frac{\Psi_{1,j+1} + \Psi_{1,j-1}}{2}$$

    - Direita

$$\Psi_{n,j} = \Psi_{n-1,j} + \frac{\Psi_{n,j+1} + \Psi_{n,j-1}}{2}$$

    - Inferior (condição de contorno)

$$\Psi_{i,1} = 0$$

Temperatura

No caso da temperatura o sinal de u e v também era importante ser observado para utilizar derivadas primeiras progressivas ou regressivas.

- Cantos
  - Direitos
    - Superior

$$T_{n,m} = \frac{T_{n-1,m} + T_{n,m-1}}{2}$$

    - Inferior

$$T_{n,1} = \frac{T_{n-1,1} + T_{n,2}}{2}$$

- Bordas
  - Superior (exceto cantos)

$$T_{i,m} = \frac{\frac{k}{\Delta y^2} * (2 * T_{i,m-1} + T_{i+1,m} + T_{i-1,m}) + \frac{\rho * C_p * u}{\Delta y} * (T_{i-1,m})}{\frac{4k}{\Delta y^2} + \frac{\rho * C_p}{\Delta y}}$$

  - Esquerda (condição de contorno)

$$T_{1,j} = T_{fora} = 20^{o}C$$

  - Direita

$$T_{n,j} = \frac{\frac{k}{\Delta y^2} * (2 * T_{n-1,j} + T_{n,j+1} + T_{n,j-1}) + \frac{\rho * C_p * u}{\Delta y} * (T_{n,j-1})}{\frac{4k}{\Delta y^2} + \frac{\rho * C_p}{\Delta y}}$$

  - Inferior (exceto cantos)

$$T_{i,1} = \frac{\frac{k}{\Delta y^2} * (2 * T_{i,2} + T_{i+1,1} + T_{i-1,1}) + \frac{\rho * C_p * u}{\Delta y} * (T_{i-1,1})}{\frac{4k}{\Delta y^2} + \frac{\rho * C_p}{\Delta y}}$$

Propriedades extensivas

Como o telhado era simulado com pontos discretos, um valor de $\Delta x = \Delta y = 0.0001$ foi utilizado assim o telhado era muito próximo à uma semi-circunferência real.

Força no galpão

Como a função corrente é simétrica no eixo x em relação ao galpão, somente as parcelas em y da diferença de pressão no telhado são relevantes. Com isso utilizando os pontos discretizados, a força foi calculada mutiplicando a diferença de pressão em cada ponto pela área discretizada:

$$F = \Delta P * Area = \Delta P * (\Delta x * Comprimento)$$

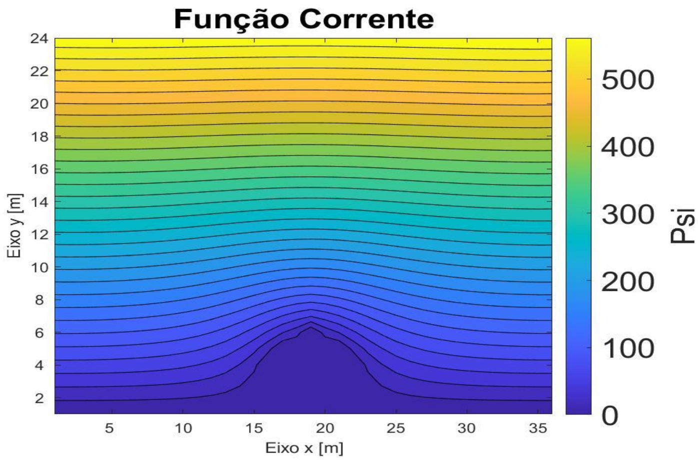
Calor trocado

Para o calor trocado, a área foi calculada de maneira semelhante à da força utilizando a equação 5.

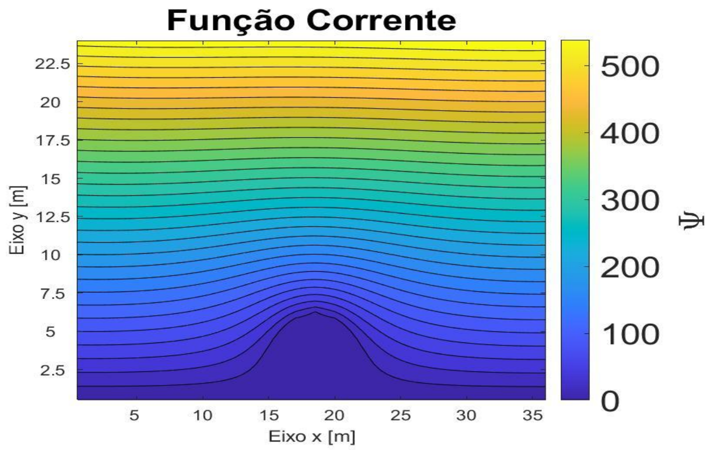
Resultados

A partir de uma malha quadrada, onde Δx=Δy, foi utilizado o método de "sobrerrelaxação" para solução do sistema linear de equações resultante da aplicação do MDF (com λ=1,85 e tolerância de convergência de 0,001).

1. Considerando o telhado como um semi círculo de raio L/2:
    a. Função de corrente ψ do escoamento:
        i. Δx = 1:



        ii. Δx= 0,5:



Discussão:

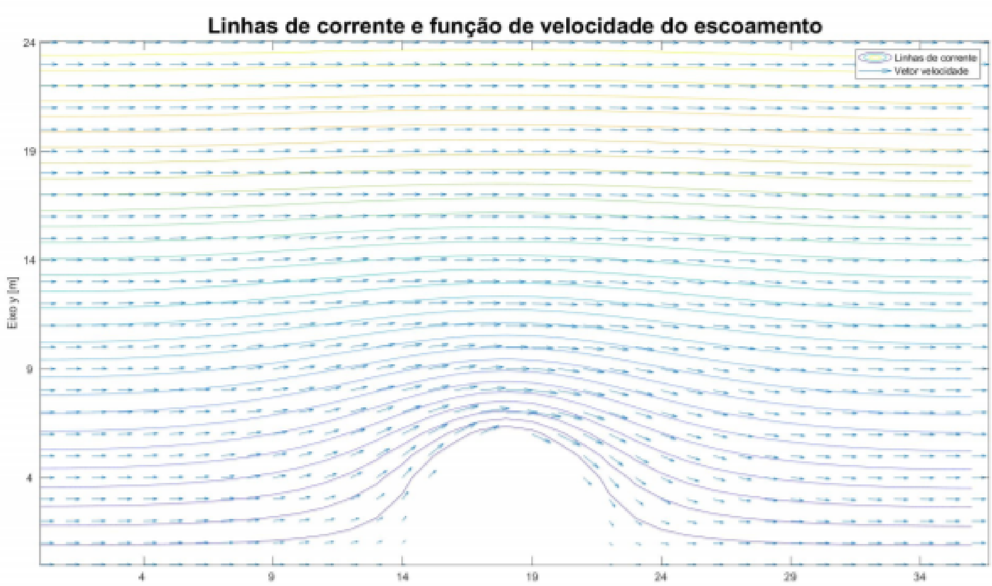
Pode-se observar que quanto menor o valor de Δx mais próximo da realidade será o modelo, de forma que para análises quantitativas mais aprofundadas é necessário o uso de passos menores. Porém, isso implica em maior custo computacional da simulação.

Além disso, foi utilizado um valor de tolerância de convergência de 0,001 com o intuito de diminuir os erros para os cantos do domínio.

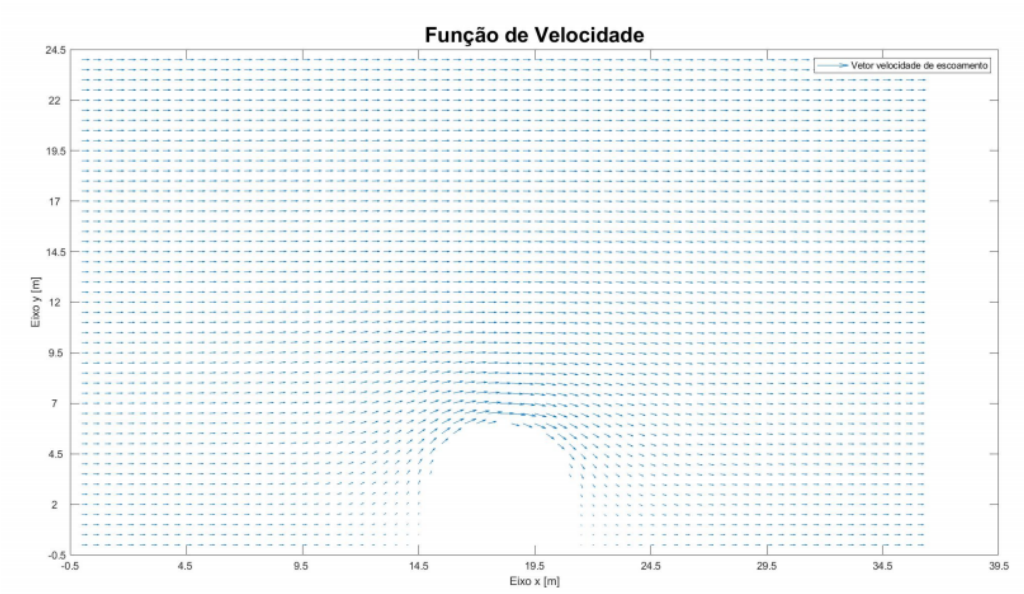
Por fim, é possível ver que a presença do galpão influencia as linhas de corrente até uma altura de 17m, mesmo ele tendo apenas 6m

b. Vetores de velocidade absoluta do escoamento:

i. Δx = 1:



ii. Δx = 0,5:



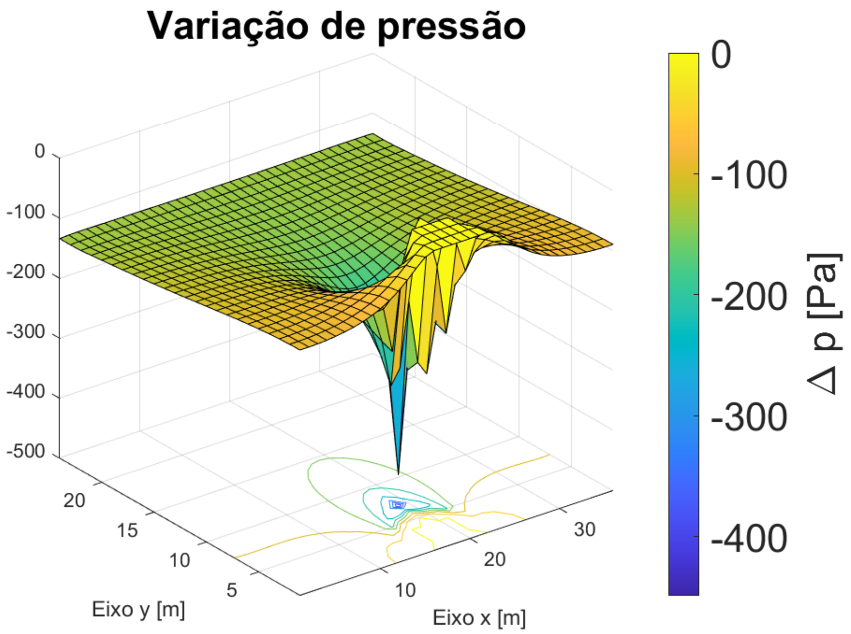
Discussão:

O fluido possui maior velocidade no ponto mais alto do telhado, o que era esperado pela equação da continuidade. Uma vez que com a presença do galpão no domínio do sistema o fluido precisa aumentar sua velocidade para manter a constância do movimento em toda a seção.

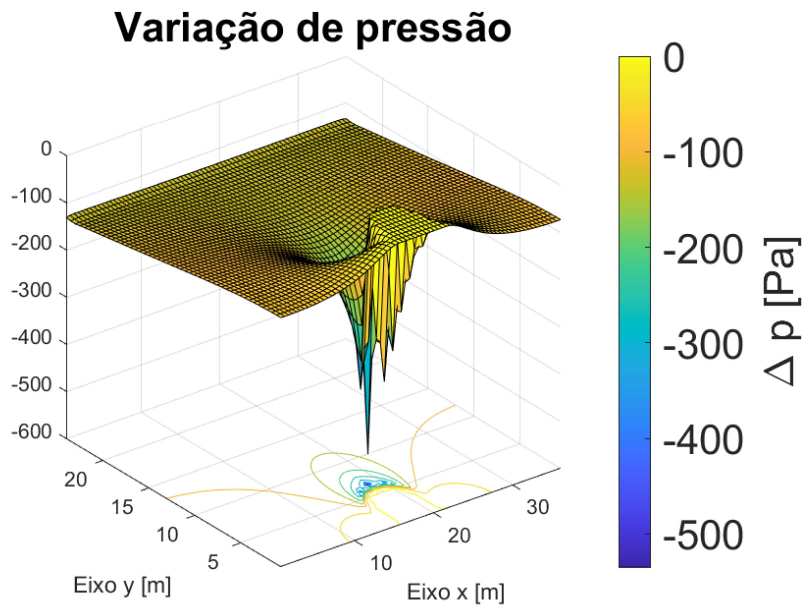
A velocidade dos ventos no telhado do galpão se aproximam de 180km/h, o que seria considerado como um furacão pela Escala Beaufort.

c. Variação de pressão ($p(x,y) - p_{atm}$) no domínio:

i. $\Delta x = 1$:



ii. $\Delta x = 0,5$:

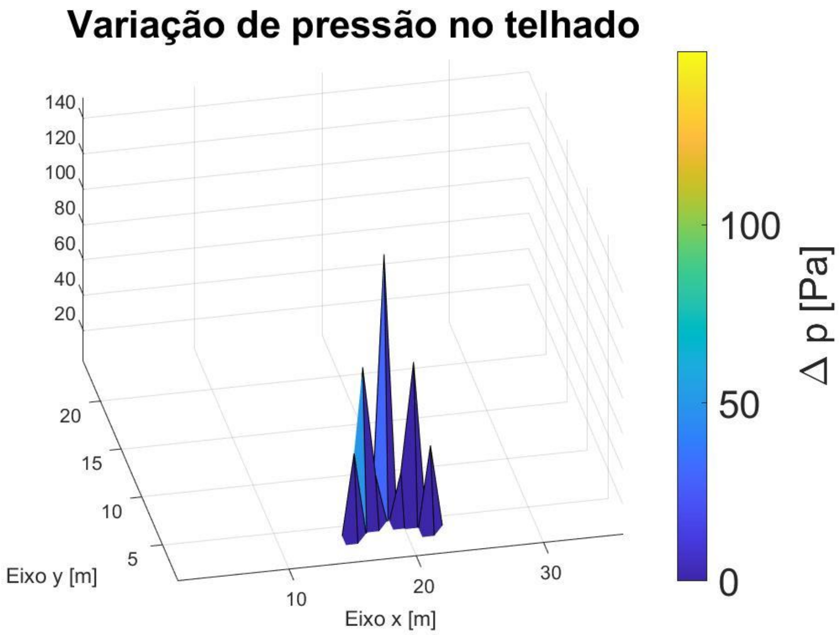


Discussão:

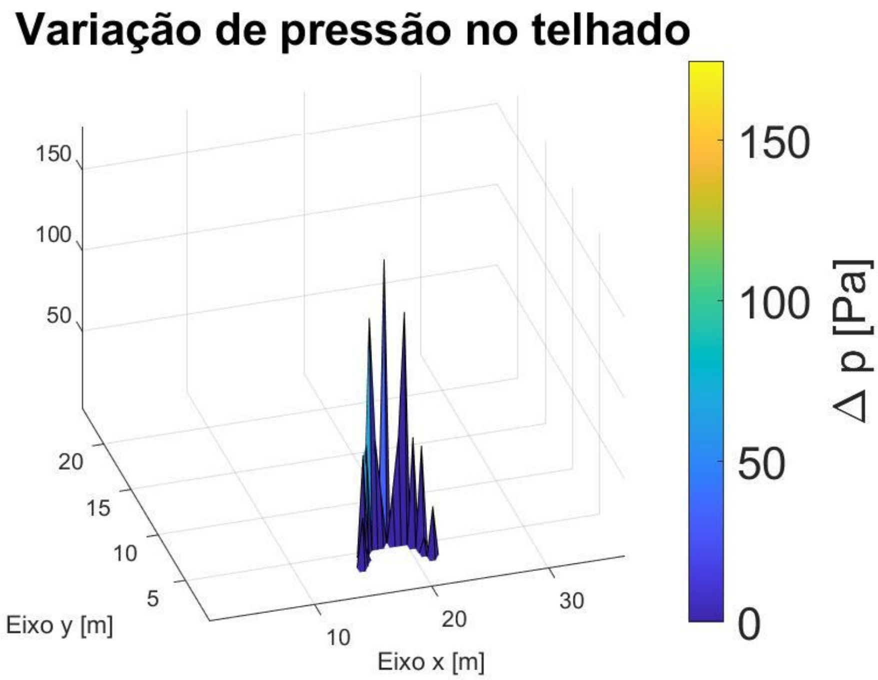
O valor mínimo de pressão obtido (com $\Delta x = 0,5$) é de 470Pa, pressão essa que seria suficiente para erguer uma telha de qualquer material. Dessa forma a fixação dessa telha seria resposável por segura-la.

d. Variação de pressão ($p(x,y) - p_{atm}$) ao longo do telhado:

i. $\Delta x = 1$:



ii. $\Delta x = 0{,}5$:

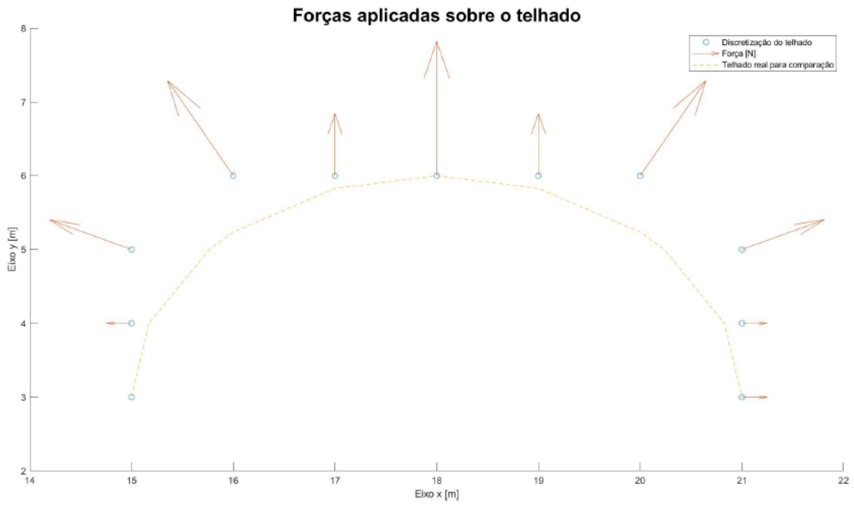


Discussão:

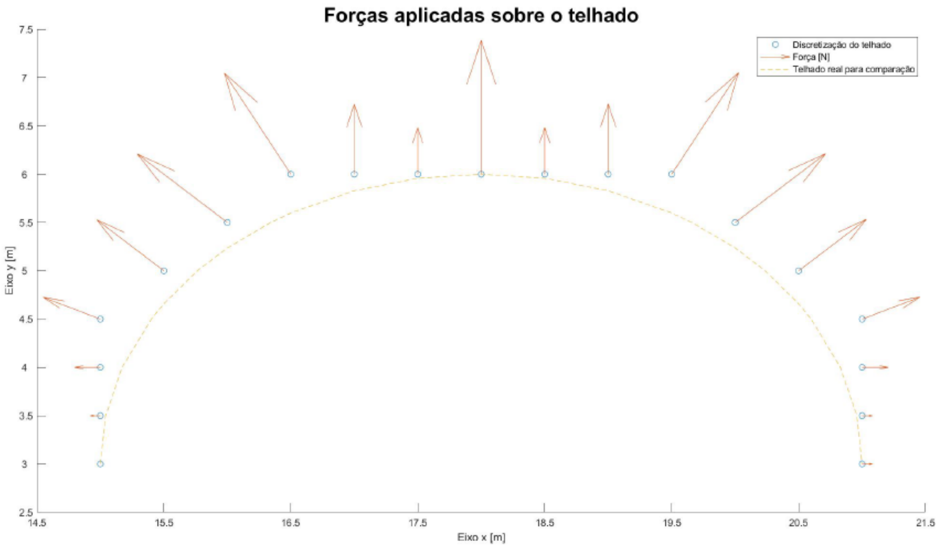
Observa-se que a pressão ascendente que existe em todo o comprimento do telhado é completamente capaz de erguê-lo caso não seja feita uma fixação correta.

e. Cálculo da força vertical resultante que atua no telhado:

i. Δx = 1: **182165N**



ii. Δx = 0,5: **95895N**



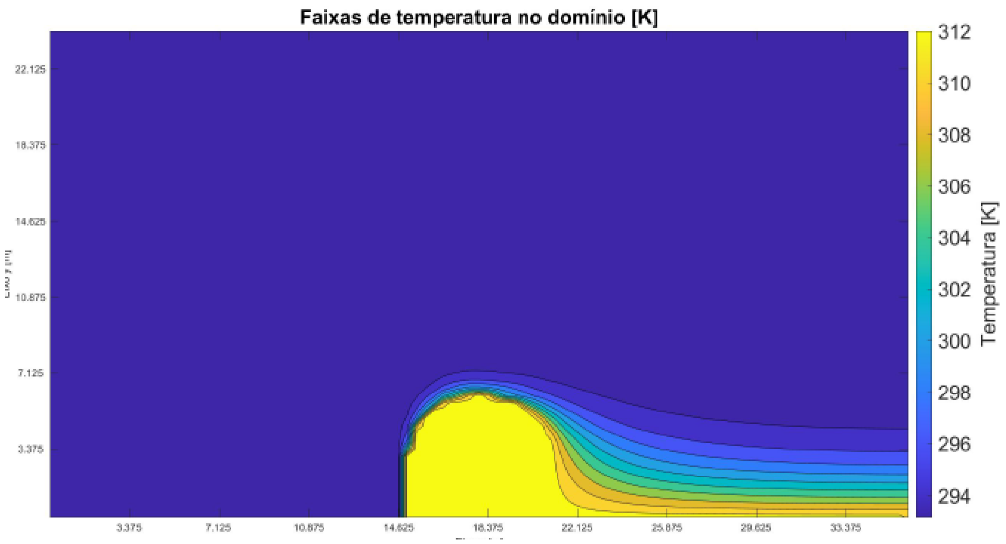
Discussão:

A diferença dos valores encontrados para cada Δx existe por conta da propagação de erro atrelada a cada processo.

É possível notar a simetria do vetores de força que atuam sobre a estrutura, que ocorre por conta do formato simétrico do galpão e dos valores de velocidade sobre o telhado.

A partir das forças resultantes, fica confirmado que o telhado do silo seria facilmente erguido caso a fixação não fosse feita da forma adequada

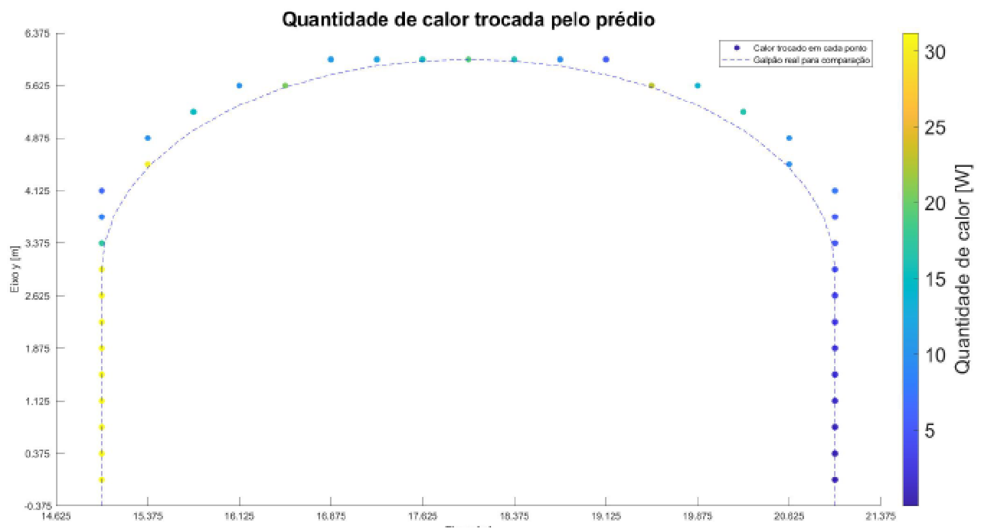
2. A partir dos resultados obtidos, com λ=1,15 e Δx=Δy=h/8:
    a. Distribuição de temperatura no ar, em °C:
        i. Gráfico:



Discussão:

É possível perceber a transferência de calor do galpão para o ambiente, após atingir o galpão o vento apresenta diversas camadas de temperaturas devido a fenômenos de condução e convecção.

    b. Taxa de calor retirada do prédio:
        i. Gráfico:



O calor total retirado do galpão foi de **q = 595W**

Discussão:

O gráfico apresentado mostra que a troca de calor entre o prédio e o vento ocorre, majoritariamente, na superfície lateral que é atingida pelas correntes de vento, por conta da maior diferença de temperatura entre o ambiente e a construção. É possível perceber, também, que no topo do telhado a troca de calor é menor, devido à maior velocidade do vento.

Conclusão Final

Podemos notar, portanto, que as simulações computacionais têm uma importância fundamental para o projeto de prédios e outras construções, como o galpão do exercício. Um projeto desse tipo que fosse feito sem o uso de simulações poderia colocar a vida de muitas pessoas em risco em caso de fortes ventanias, visto que a fixação dos telhados deve ser feita com um planejamento preciso.

Além disso, também ficou evidente a influência da discretização do domínio durante toda a análise. Uma discretização com passo menor faz com que o modelo fique mais próximo da realidade e, logo, mais preciso. Por outro lado essa discretização com menores Δx e Δy implica em um custo computacional muito maior.

Códigos

main

```
close all;clc; clear;

%Constantes
V=100/3.6;
h=3;
d=5*h;
L=2*h;
H=8*h;
ro=1.25;
gama=1.4;
k=0.026;
cp=1002;
patm=10000;
comprimento=60; %fundura

Tdentro=40;
Tfora=20;


%variaveis computacionais
dx=h/8
dy=dx
lambda=1.85
eps=0.001



xi=2;
xf=(2*d+L)/dx-1;
yi=2;
yf=H/dy-1;


corr=zeros(yf+1,xf+1);
pressao=zeros(yf+1,xf+1);
deltap=zeros(yf+1,xf+1);


%popula com 1 o que tem dentro
for i = yi:yf
      for j = xi:xf
            corr(i,j) = 0;
      end
```

```
end

gxi=d/dx+1; %x esquerda
gxf=gxi+(L)/dx; %x direita
gyi=yi; %
gyf=gyi+h/dy-1; %

%popula com 2 o que tem dentro do galpao
for i = gyi:gyf
      for j = gxi:gxf
            corr(i,j) = 0;
      end
end

%popula com 3 o que tem no topo do galpao
for i = gyf:(gyf+(L/2)/dy-1)
      for j = gxi:gxf
            if ((i)*dy) <= (sqrt((L/2)^2 - ((j-1)*dx-d-L/2)^2)+h)
                  corr(i,j) = 0;
            end
      end
end


contador=0;
convergiu=false;

while ~convergiu
      contador = contador+1;
      convergindo=true;

      %aplicando condicoes nas bordas utilizando equacionamento de
taylor
      %para considerar condicoes de Neumann



      %primeiro nas pontas superiores
      corr(yf+1,1)=(corr(yf+1,2)+corr(yf,1)+V*dy)/2;
      corr(yf+1,xf+1)=(corr(yf+1,xf-1)+corr(yf,xf+1)+V*dy)/2;

      %segundo no topo
      for j = xi:xf
            corr(yf+1,j)=(dx*V + corr(yf,j) +
(corr(yf+1,j-1)+corr(yf+1,j+1))/2)/2;
      end

      %segundo nos lados
```

```
      for i = yi:yf
            corr(i,1)=(corr(i,2) + (corr(i-1,1)+corr(i+1,1))/2)/2;
%esquerdo
            corr(i,xf+1)=(corr(i,xf) +
(corr(i-1,xf+1)+corr(i+1,xf+1))/2)/2; %direito
      end


      %Aplicando o metrodo nos nos internos
      %primeiro na altura inferior ao topo do telhado
      for i = gyi:gyf
            for j = xi:gxi-1 %antes do galpao
                  noAntigo =corr(i,j);
                  noAtual =
(corr(i+1,j)+corr(i-1,j)+corr(i,j+1)+corr(i,j-1))/4;
                  corr(i,j) = lambda*noAtual + (1-lambda)*noAntigo;
%sobrerrelaxacao

                  if convergindo %para melhorar desempenho e nao
fazer contas desnecessarias
                        if abs((corr(i,j)-noAntigo))/corr(i,j) > eps
                              convergindo = false; %fazer mais
interacoes
                        end
                  end
            end

            for j = gxf+1:xf %depois do galpao
                  noAntigo =corr(i,j);
                  noAtual =
(corr(i+1,j)+corr(i-1,j)+corr(i,j+1)+corr(i,j-1))/4;
                  corr(i,j) = lambda*noAtual + (1-lambda)*noAntigo;
%sobrerrelaxacao

                  if convergindo %para melhorar desempenho e nao
fazer contas desnecessarias
                        if abs((corr(i,j)-noAntigo))/corr(i,j) > eps
                              convergindo = false; %fazer mais
interacoes
                        end
                  end
            end
      end


      %segundo na altura superior ao topo do telhado
      for i = gyf+1:yf
```

```
            for j = xi:xf
                  if  (i*dy) > (sqrt((L/2)^2 -
((j-1)*dx-d-L/2)^2)+h) %checa se esta nos limites externos do
telhado
                        noAntigo =corr(i,j);
                        noAtual =
(corr(i+1,j)+corr(i-1,j)+corr(i,j+1)+corr(i,j-1))/4;
                        corr(i,j) = lambda*noAtual +
(1-lambda)*noAntigo; %sobrerrelaxacao

                        if convergindo %para melhorar desempenho e
nao fazer contas desnecessarias
                              if abs((corr(i,j)-noAntigo))/corr(i,j)
> eps
                                    convergindo = false; %fazer mais
interacoes
                              end
                        end
                  end
            end
      end
      if convergindo %checa se pode parar de iterar
            convergiu=true;
      end
end

contador








%calculando pressao utilizando primeira diferenca central
for i = yi:yf
      for j = xi:xf %antes do galpao
            u=(corr(i+1,j)-corr(i-1,j))/(2*dy);
            v=(corr(i,j+1)-corr(i,j-1))/(2*dx);
            pressao(i,j) = -ro*(gama-1)/gama*(u^2+v^2)/2 + patm;
      end
end

%calculando nas bordas com diferenças progressivas e regressivas
for j = xi:xf
      %em cima
```

```
        u=(corr(yf+1,j)-corr(yf,j))/(dy);
        if u<0
              1
        end
        v=(corr(yf+1,j+1)-corr(yf+1,j-1))/(2*dx);
        pressao(yf+1,j)=-ro*(gama-1)/gama*(u^2+v^2)/2 + patm;

        %embaixo
        u=(corr(2,j)-corr(1,j))/(dy);
        v=(corr(1,j+1)-corr(1,j-1))/(2*dx);
        pressao(1,j)=-ro*(gama-1)/gama*(u^2+v^2)/2 + patm;
end

for i = yi:yf
        %lado esquerdo
        u=(corr(i+1,1)-corr(i-1,1))/(2*dy);
        v=(corr(i,2)-corr(i,1))/(dx);
        pressao(i,1)=-ro*(gama-1)/gama*(u^2+v^2)/2 + patm;

        %lado direito
        u=(corr(i+1,xf+1)-corr(i-1,xf+1))/(2*dy);
        v=(corr(i,xf+1)-corr(i,xf))/(dx);
        pressao(i,xf+1)=-ro*(gama-1)/gama*(u^2+v^2)/2 + patm;
end

%calculando nas pontas
%embaixo esquerda
u=(corr(2,1)-corr(1,1))/(dy);
v=(corr(1,2)-corr(1,1))/(dx);
pressao(1,1)=-ro*(gama-1)/gama*(u^2+v^2)/2 + patm;

%embaixo direita
u=(corr(2,xf+1)-corr(1,xf+1))/(dy);
v=(corr(1,xf+1)-corr(1,xf))/(dx);
pressao(1,xf+1)=-ro*(gama-1)/gama*(u^2+v^2)/2 + patm;

%em cima esquerda
u=(corr(yf+1,1)-corr(yf,1))/(dy);
v=(corr(yf+1,2)-corr(yf+1,1))/(dx);
pressao(yf+1,1)=-ro*(gama-1)/gama*(u^2+v^2)/2 + patm;

%em cima direita
u=(corr(yf+1,xf+1)-corr(yf,xf+1))/(dy);
v=(corr(yf+1,xf+1)-corr(yf+1,xf))/(dx);
pressao(yf+1,xf+1)=-ro*(gama-1)/gama*(u^2+v^2)/2 + patm;
```

```
%variacao de pressao
for i = 1:yf+1
    for j = 1:xf+1
        deltap(i,j) = pressao(i,j) - patm;
    end
end



maiory=0;
maiorx=0;
menorPressaoTelhado=Inf;
telhadoP=zeros(yf+1,xf+1);
for i = gyf:yf+1
    for j = 1:xf+1
        if deltap(i,j)==0 %checa se é um ponto interno do
telhado
            %testa se alguma do vizinhanca é o telhado vendo
se é diferente de 0 e atribui a matriz do telhado
            %algum valores serao subscritos neles mesmos
            if deltap(i-1,j)==0
                telhadoP(i-1,j) = deltap(i-1,j);
            end
            if deltap(i,j-1)
                telhadoP(i,j-1) = deltap(i,j-1);
            end
            if deltap(i+1,j)
                telhadoP(i+1,j) = deltap(i+1,j);
            end
            if deltap(i,j+1)
                telhadoP(i,j+1) = deltap(i,j+1);
            end
        end
    end
end
```

```
%como o telhado eh simetrico só há uma parcela em y de forca
forca=0;

%percorre em x de cima para baixo ate encontrar o telhado
for j = 1:xf+1
      for i = fliplr(1:yf+1)
            if telhadoP(i,j)~=0
                  forca = forca + (telhadoP(i,j))*(dx*comprimento);
%pressao*area

                  %seno=(i-gyi)/sqrt(((i-gyi))^2+(j-xf/2)^2);
                  %forca = forca +
(telhadoP(i,j)*seno)*(dx*comprimento*seno); %pressao*area
                  break % pula para proximo x
            end
      end
end

forca
```

```
%2a aprte
dx=h/8;
dy=h/8;
lambda=1.15;
eps=0.01;

xi=2;
xf=(2*d+L)/dx-1;
yi=2;
yf=H/dy-1;


T=zeros(yf+1,xf+1);

%condicao de dischlet a esquerda
for i=1:yf+1
      T(i,1) = Tfora;
end

gxi=d/dx+1; %x esquerda
gxf=gxi+(L)/dx; %x direita
gyi=yi; %
gyf=gyi+h/dy-1; %

%popula com Tdentro o que tem dentro do galpao
for i = gyi:gyf
      for j = gxi:gxf
            T(i,j) = Tdentro;
      end
end
```

```
%popula com 3 o que tem no topo do galpao
for i = gyf:(gyf+(L/2)/dy-1)
      for j = gxi:gxf
            if (i*dy) <= (sqrt((L/2)^2 - ((j-1)*dx-d-L/2)^2)+h)
                  T(i,j) = Tdentro;
            end
      end
end




contador=0;
convergiu=false;

while ~convergiu
      contador = contador+1;
      convergindo=true;

      %aplicando condicoes nas bordas utilizando equacionamento de
taylor
      %para considerar condicoes de Neumann


      %primeiro nas pontas
      %cima
      noAtual=(T(yf+1,xf)+T(yf,xf+1))/2;
      noAntigo =T(i,j);
      T(yf+1,xf+1) = lambda*noAtual + (1-lambda)*noAntigo;
%sobrerrelaxacao
      %baixo
      noAtual=(T(1,xf+1)+T(2,xf))/2;
      noAntigo =T(i,j);
      T(1,xf+1) = lambda*noAtual + (1-lambda)*noAntigo;
%sobrerrelaxacao
      %segundo no topo
      u=V;
      for j = xi:xf
            noAtual=(k/dx^2*(2*T(yf,j) +  T(yf+1,j+1)+T(yf+1,j-1)) +
ro*cp*u/dx*(T(yf+1,j-1)))/(4*k/dx^2 + ro*cp*u/dx);
            noAntigo =T(i,j);
            T(yf+1,j) = lambda*noAtual + (1-lambda)*noAntigo;
%sobrerrelaxacao
      end

      %segundo emabvixo
            %antes do galpao
```

```
      for j = xi:gxi-1
            u=(corr(i+1,j)-corr(i-1,j))/(2*dy);
            noAtual=(k/dx^2*(2*T(2,j) +  T(1,j+1)+T(1,j-1)) +
ro*cp*u/dx*(T(1,j-1)))/(4*k/dx^2 + ro*cp*u/dx);
            noAntigo =T(i,j);
            T(1,j) = lambda*noAtual + (1-lambda)*noAntigo;
%sobrerrelaxacao
      end
            %depois do galpao
      for j = gxf+1:xf
            u=(corr(i+1,j)-corr(i-1,j))/(2*dy);
            noAtual=(k/dx^2*(2*T(2,j) +  T(1,j+1)+T(1,j-1)) +
ro*cp*u/dx*(T(1,j-1)))/(4*k/dx^2 + ro*cp*u/dx);
            noAntigo =T(i,j);
            T(1,j) = lambda*noAtual + (1-lambda)*noAntigo;
%sobrerrelaxacao
      end

      %terceito a direita
      for i = yi:yf
            u=(corr(i+1,xf+1)-corr(i-1,xf+1))/(2*dy);
            v=0;
            noAtual=(k/dx^2*(2*T(i,xf) +  T(i+1,xf+1)+T(i-1,xf+1)) +
ro*cp*u/dx*(T(i-1,xf+1)))/(4*k/dx^2 + ro*cp*u/dx);
            noAntigo =T(i,j);
            T(i,xf+1) = lambda*noAtual + (1-lambda)*noAntigo;
%sobrerrelaxacao
      end


      %Aplicando o metrodo nos nos internos
      %primeiro na altura inferior ao topo do telhado
      for i = gyi:gyf
            for j = xi:gxi-1 %antes do galpao
                  u=(corr(i+1,j)-corr(i-1,j))/(2*dy);
                  v=(corr(i,j+1)-corr(i,j-1))/(2*dx);
                  if u<0 & v<0
                        noAtual =
(k/dx^2*(T(i+1,j)+T(i-1,j)+T(i,j+1)+T(i,j-1)) -
ro*cp*u/dx*(2*T(i,j+1)))/(4*k/dx^2 + 2*ro*cp*u/dx);
                  elseif u>0 & v>0
                        noAtual =
(k/dx^2*(T(i+1,j)+T(i-1,j)+T(i,j+1)+T(i,j-1)) +
ro*cp*u/dx*(2*T(i,j-1)))/(4*k/dx^2 - 2*ro*cp*u/dx);
                  else
                        noAtual =
(k/dx^2*(T(i+1,j)+T(i-1,j)+T(i,j+1)+T(i,j-1)) + ro*cp*u/dx*(T(i,j+1)
```

```
- T(i,j-1)))/(4*k/dx^2);
                  end

                  noAntigo =T(i,j);
                  T(i,j) = lambda*noAtual + (1-lambda)*noAntigo;
%sobrerrelaxacao

                  if convergindo %para melhorar desempenho e nao
fazer contas desnecessarias
                        if abs((T(i,j)-noAntigo))/T(i,j) > eps
                              convergindo = false; %fazer mais
interacoes
                        end
                  end
            end

            for j = gxf+1:xf %depois do galpao
                  u=(corr(i+1,j)-corr(i-1,j))/(2*dy);
                  v=(corr(i,j+1)-corr(i,j-1))/(2*dx);
                  if u<0 & v<0
                        noAtual =
(k/dx^2*(T(i+1,j)+T(i-1,j)+T(i,j+1)+T(i,j-1)) -
ro*cp*u/dx*(2*T(i,j+1)))/(4*k/dx^2 + 2*ro*cp*u/dx);
                  elseif u>0 & v>0
                        noAtual =
(k/dx^2*(T(i+1,j)+T(i-1,j)+T(i,j+1)+T(i,j-1)) +
ro*cp*u/dx*(2*T(i,j-1)))/(4*k/dx^2 - 2*ro*cp*u/dx);
                  else
                        noAtual =
(k/dx^2*(T(i+1,j)+T(i-1,j)+T(i,j+1)+T(i,j-1)) + ro*cp*u/dx*(T(i,j+1)
- T(i,j-1)))/(4*k/dx^2);
                  end

                  noAntigo =T(i,j);
                  T(i,j) = lambda*noAtual + (1-lambda)*noAntigo;
%sobrerrelaxacao

                  if convergindo %para melhorar desempenho e nao
fazer contas desnecessarias
                        if abs((T(i,j)-noAntigo))/T(i,j) > eps
                              convergindo = false; %fazer mais
interacoes
                        end
                  end
            end
      end
```

```
        %segundo na altura superior ao topo do telhado
        for i = gyf+1:yf
                for j = xi:xf
                        if  (i*dy) > (sqrt((L/2)^2 -
((j-1)*dx-d-L/2)^2)+h) %checa se esta nos limites externos do
telhado
                                u=(corr(i+1,j)-corr(i-1,j))/(2*dy);
                                v=(corr(i,j+1)-corr(i,j-1))/(2*dx);
                                if u<0 & v<0
                                        noAtual =
(k/dx^2*(T(i+1,j)+T(i-1,j)+T(i,j+1)+T(i,j-1)) -
ro*cp*u/dx*(2*T(i,j+1)))/(4*k/dx^2 + 2*ro*cp*u/dx);
                                elseif u>0 & v>0
                                        noAtual =
(k/dx^2*(T(i+1,j)+T(i-1,j)+T(i,j+1)+T(i,j-1)) +
ro*cp*u/dx*(2*T(i,j-1)))/(4*k/dx^2 - 2*ro*cp*u/dx);
                                else
                                        noAtual =
(k/dx^2*(T(i+1,j)+T(i-1,j)+T(i,j+1)+T(i,j-1)) + ro*cp*u/dx*(T(i,j+1)
- T(i,j-1)))/(4*k/dx^2);
                                end

                                noAntigo =T(i,j);
                                T(i,j) = lambda*noAtual +
(1-lambda)*noAntigo; %sobrerrelaxacao

                                if convergindo %para melhorar desempenho e
nao fazer contas desnecessarias
                                        if abs((T(i,j)-noAntigo))/T(i,j) > eps
                                                convergindo = false; %fazer mais
interacoes
                                        end
                                end
                        end
                end
        end
        if convergindo %checa se pode parar de iterar
                convergiu=true;
        end
end

contador
```

```
%calculando Q
Q=0;

%percorrendo da cima para baixo
for j = 1:xf+1
      for i = fliplr(1:yf+1)
            if T(i,j)==Tdentro
                  %prmeira diferente progressiva
                  Q = Q -k*(T(i+1,j)-T(i,j))/dy *(dx*comprimento);
%pressao*area
                  break % pula para proximo x
            end
      end
end


%percorrendo da esquerda para direita
for i = 1:yf+1
      for j = 1:xf+1
            if T(i,j)==Tdentro
                  %prmeira diferente progressiva
                  Q = Q -k*(T(i,j-1)-T(i,j))/dx *(dy*comprimento);
%pressao*area
                  break % pula para proximo y
            end
      end
end


%percorrendo da direita para esquerda
for i = 1:yf+1
      for j = fliplr(1:xf+1)
            if T(i,j)==Tdentro
                  %prmeira diferente progressiva
                  Q = Q -k*(T(i,j+1)-T(i,j))/dx *(dy*comprimento);
%pressao*area
                  break % pula para proximo y
            end
      end
end
```