TCM

Utilização de Métodos Finitos para a Resolução da Equação do Calor

Alexandre Abrahami Pinto da Cunha
acunha86@hotmail.com
18/0041169

16 de outubro de 2020

1 Introdução

Trabalhamos aqui com 2 conceitos principais: a equação geral da difusão de calor e o método das diferenças finitas.

A equação geral da difusão de calor, também conhecida como equação do calor, é uma das principais equações da Termodinâmica, importante em muitos campos da ciência e da matemática aplicada, que descreve como um corpo se comportará dada a influência de um temperatura externa à suas extremidades em função do espaço e do tempo. Ela pode ser definida matematicamente como:

$$\rho C_p \frac{\partial T}{\partial t} = k\nabla^2 T$$

onde ρ é a massa específica, C_p é o calor específico à pressão constante, T é o campo de temperatura e k é a condutividade térmica do meio.

O método das diferenças finitas é uma ferramenta numérica de resolução de equações diferenciais que se baseia na aproximação de derivadas por diferenças finitas. Esse método facilita e deixa mais prático a resolução de problemas com dimensões maiores que às vezes podem acabar não tendo uma resolução analítica viável. A fórmula de aproximação obtém-se da série de Taylor da função derivada. Para fins práticos, simulando uma malha e definindo arbitrariamente um número de nós pode-se representar a discretização da temperatura para o método das diferenças finitas por:

$$T_{m,n}^{p+1} = F_o(T_{m+1,n}^{p} + T_{m-1,n}^{p} + T_{m,n+1}^{p} + T_{m,n-1}^{p}) + (1 - 4F_o)T_{m,n}^{p}$$

Com base nessa equação, conseguimos desenvolver um programa computacional que descreve a distribuição da temperatura em diferentes corpos com diferentes geometrias de forma muito mais prática quando comparado ao cálculo analítico.

2 Problemática e Procedimento

Este relatório tem como objetivo a resolução numérica da equação da difusão de calor no domínio do caso 3 repassado pelo professor, representado abaixo:

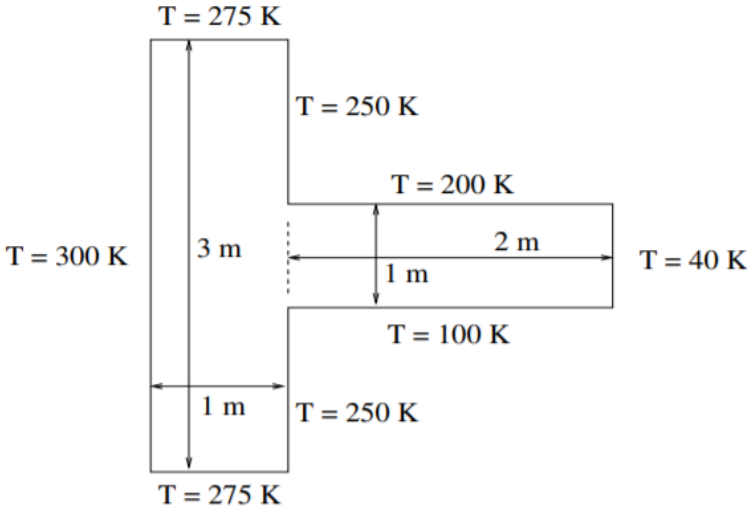

Figura 1 – Caso 3

Foi desenvolvido um código em **Python** para essa resolução devido à facilidade/proximidade com a linguagem e a possibilidade de utilização de diferentes bibliotecas prontas que auxiliaram na criação da malha e na plotagem dos gráficos, sendo elas: o matplotlib que possibilitou a plotagem de imagens, gráficos e animações; a math que foi utilizada para a resolução e aplicação de funções matemáticas; e a numpy que facilitou a criação e a manipulação das matrizes.

Basicamente no programa são construídas duas matrizes uma para o instante "atual"do objeto e outra para o instante "anterior". A matriz atual é atualizada de acordo com a equação do método das diferenças finitas utilizando-se os valores da matriz anterior. É criada então uma função recursiva em que a cada interação cada nó analisa os nós na sua vizinhança para determinar a sua temperatura. Dada as condições de contorno, após um número n de repetições na função conseguimos determinar a temperatura distribuída em cada nó em regime permanente pegando-se um nó aleatório próximo ao centro da figura, realizando diversos testes, observando-se qual é a temperatura máxima que ele atinge e determinando a condição de parada da função.

Por fim, o programa chama duas funções previamente definidas cuja únicas funções são as de plotar e salvar os gráficos da análise da temperatura em diversos pontos do objeto e a imagem da distribuição de temperatura.

Um comentário a ser feito sobre o programa é que a cada interação ele salva uma imagem do instante "atual"do objeto à medida em que ele vai aplicando a equação do método das diferenças finitas até atingir a condição de parada afim de, utilizando-se o auxílio de um programa/site externo, juntar todas essas imagens e gerar um vídeo demonstrando a distribuição da temperatura no corpo em função do tempo

3 Código

Segue abaixo o código utilizado para a criação da malha e descrição da distribuição de temperatura. O código está todo comentado e explica passo a passo o que cada parte está fazendo e significa.

```
from matplotlib import cm
import matplotlib.pyplot as plt
import math
import numpy as np

no = 100 #Número de nós na malha decidido arbitrariamente
t = 0
tempo = 10000 #Tempo decidido arbitrariamente
intervalos = 10000 #Número de divisões dentro do período analisado decidido arbitrariamente
anterior = np.zeros((no,no), float)
atual = np.copy(anterior)
#Criei duas matrizes para determinar a temperatura em um nó em um momento anterior e em um momento
    atual.

dt = (tempo/intervalos) #Definindo o passo de tempo
dx = (3/no) #Definindo o incremento infinitesimal em x

#Para a definição do alfa vou considerar que o material do objeto éde cobre
alfa = 117/(10**6) #Difusividade térmica do cobre = 117
fourier = (alfa*dt)/(dx**2)
```

```
#Região de código utilizado para análises de domínios de cálculo
def gerar_graf(no, matriz):  #Iremos gerar 5 gráficos de uma vez afim de se economizar tempo e
    esforço
    altura_analisada = [round(no/2), round(no/1.2)]
    comprimento_analisado = [round(no/6), round(no/2), round(no/1.2)]
    alturas = (1.5,2.5)   #Pontos em Y definidos arbitrariamente
    comprimentos = (0.5,1.5,3)   #Pontos em X definidos arbitrariamente
    intervalo = np.linspace(0,3, num = no)
    atual = matriz

    for x in range(len(comprimento_analisado)): #Gerando os gráficos analisando os pontos com base
        no eixo x
        temperaturas = np.array(atual[:,comprimento_analisado[x]])
        plt.plot(intervalo,temperaturas)
        plt.title('Perfil de Temperatura em X = ' + (str)(comprimentos[x]))
        plt.xlabel('Altura Y (m)')
        plt.ylabel('Temperatura (K)')
        plt.savefig('X_'+ (str)(comprimentos[x]) + '.png')
        plt.close('all')

    for y in range(len(altura_analisada)):  #Gerando os gráficos analisando os pontos com base no
        eixo y
        temperaturas = np.array(atual[altura_analisada[y],:])
        plt.plot(intervalo,temperaturas)
        plt.title('Perfil de Temperatura em Y = ' + (str)(alturas[y]))
        plt.xlabel('Comprimento X (m)')
        plt.ylabel('Temperatura (K)')
        plt.savefig('Y_'+ (str)(alturas[y]) + '.png')
        plt.close('all')


#Aqui plotamos o gráfico da distribuição de temperatura na figura
def gerar_distr():
    plt.figure()
    plt.title('Distribuição de Temperatura ' + (str)(no) + 'X' + (str)(no))
    plt.imshow(np.flipud(atual), cmap = 'twilight_shifted', interpolation = 'none', extent=[0, 3,
        0, 3])
    colorbar = plt.colorbar()
    plt.xlabel('Distncia horizontal (m)')
    plt.ylabel('Distncia vertical (m)')
    plt.show()

#Aqui iremos determinar as condições de contorno na figura

#Primeiro, determinamos nas paredes verticais da figura
for parede_vert in range(no):
    anterior[parede_vert][0] = 300.00
```

```
    if(parede_vert < (round(no/3) - 1)):
        anterior[parede_vert][(round(no/3) - 1)] = 250.00
    elif(parede_vert > (round(no/3) - 1) and parede_vert <(2*round(no/3) - 1)):
        anterior[parede_vert][(no - 1)] = 40.00
    else:
        anterior[parede_vert][(round(no/3) - 1)] = 250.00


#Em seguida, determinamos nas paredes horizontais da figura
for parede_hz in range(no):
    if(parede_hz < (round(no/3) - 1)):
        anterior[0][parede_hz] = 275.00
        anterior[(no - 1)][parede_hz] = 275.00
    else:
        anterior[(round(no/3) - 1)][parede_hz] = 200.00
        anterior[(2*round(no/3) - 1)][parede_hz] = 100.00


#Determinando agora as temperaturas em cada nó da malha
while t < 100000:  #Loop temporal
    for y in range(no):   #Loop no eixo y
        for x in range(no):   #Loop no eixo x
            if (atual[round(no/2), round(no/3)] > 160):
                t = 100001      #Condição de parada: quando este nó atingir 160 o corpo se encontra
                    próximo do regime estacionário
            elif(x == 0):
                atual[y, x] = 300.00
            elif((y == 0 or y == (no - 1)) and x <= (round(no/3) - 1)):
                atual[y, x] = 275.00
            elif(y == (round(no/3) - 1) and x > (round(no/3) - 1)):
                atual[y, x] = 200.00
            elif(y == (2*round(no/3) - 1) and x > (round(no/3) - 1)):
                atual[y, x] = 100.00
            elif(x == (round(no/3) - 1) and (y <= (round(no/3) - 1) or y >= (2*round(no/3) - 1))):
                atual[y, x] = 250.00
            elif(x == (no - 1) and (round(no/3) - 1)<y<(2*round(no/3) - 1)):
                atual[y, x] = 40.00
            elif((x > (round(no/3) - 1) and y < (round(no/3) - 1)) or (y > (2*round(no/3) - 1) and x
                > (round(no/3) - 1))):
                atual[y, x] = 0.00
            else:
                atual[y, x] = fourier * (anterior[y+1, x] + anterior[y-1, x] + anterior[y, x+1] +
                    anterior[y, x-1]) + (1-(4*fourier))*anterior[y, x]
                anterior[y, x] = atual[y, x]

    t = t+1
    plt.imsave('./imagens/' + str(t) + '.png', np.flipud(atual), cmap='twilight_shifted')
    #Aqui estamos salvando as imagens em uma pasta "Imagens" para cada instante afim de juntá-las
```

```
        posteriormente e gerar um vídeo

#Gerando os perfis de temperatura para 5 pontos diferentes do corpo analisado
gerar_graf(no, atual)

#Gerando a imagem da distribuição de temperatura
gerar_distr()
```

4 Análise de resultados

4.1 Distribuição de Temperatura

Compilando o código acima alterando-se os valores da variável *no* para 50, 100, 150 e 200, obtive a distribuição de temperatura no corpo para quatro malhas distintas com diferentes números de nós: 2500, 10000, 22500 e 40000 nós. Esses resultados podem ser observados nas imagens abaixo:

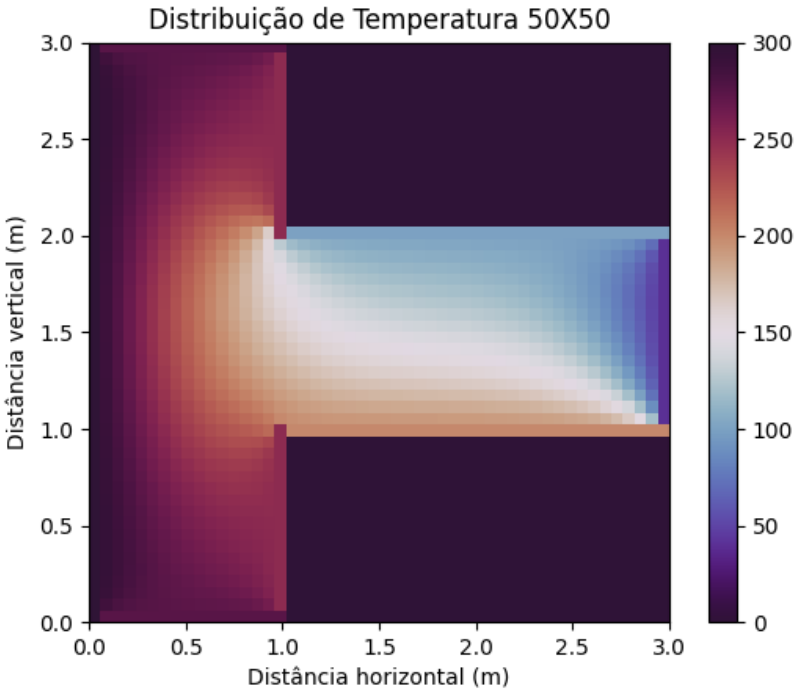

Figura 2 – Distribuição de temperatura em uma malha com 2500 nós

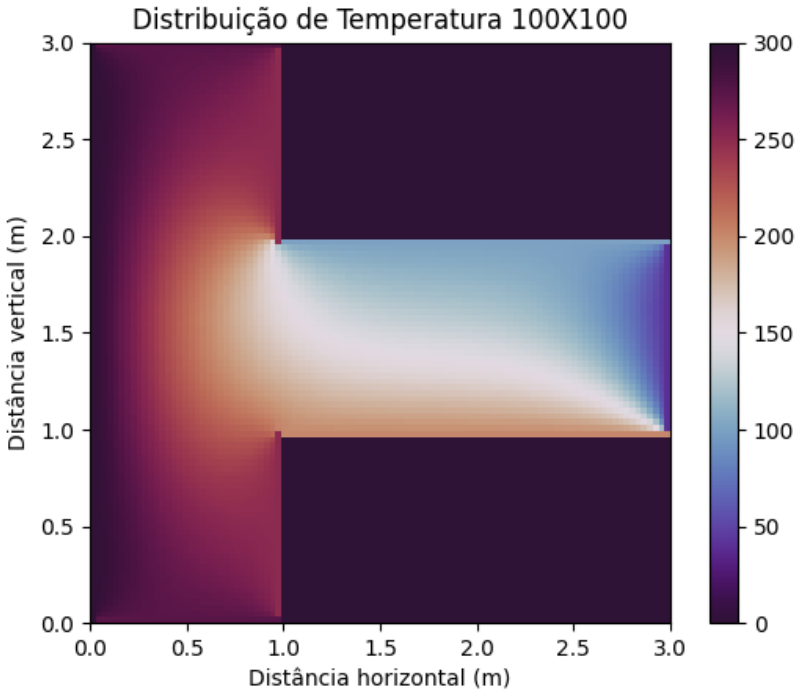

Figura 3 – Distribuição de temperatura em uma malha com 10000 nós

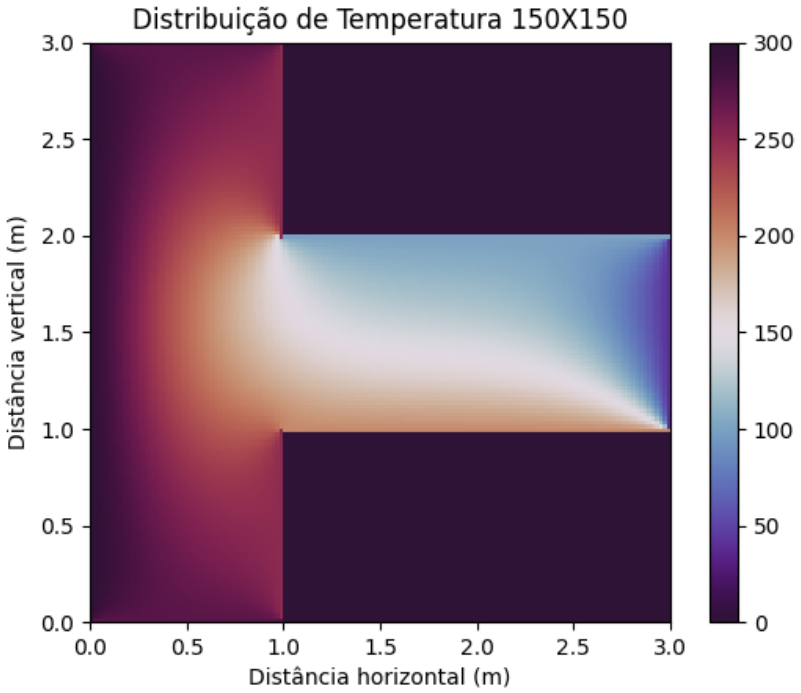

Figura 4 – Distribuição de temperatura em uma malha com 22500 nós

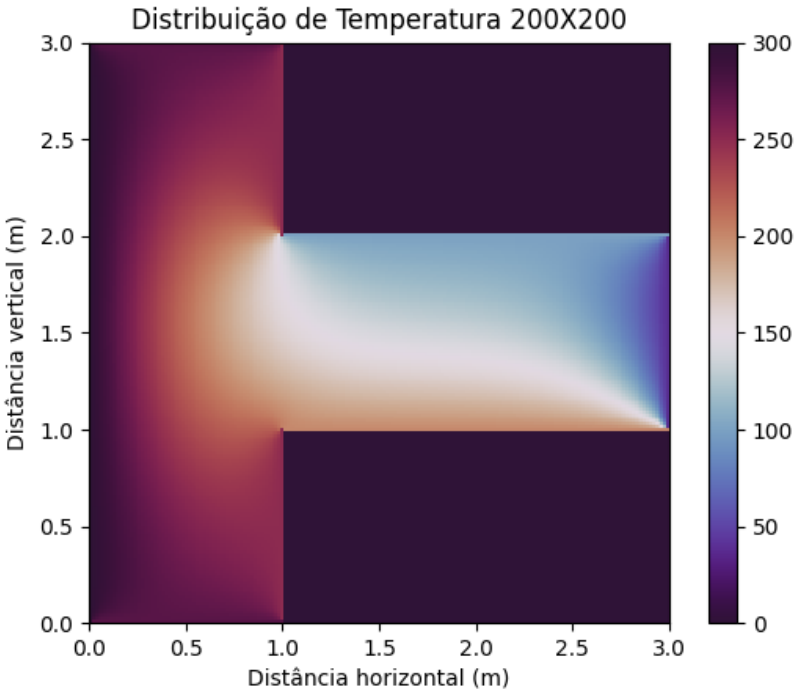

Figura 5 – Distribuição de temperatura em uma malha com 40000 nós

Percebe-se facilmente que à medida em que se aumenta o número de nós na malha, a qualidade da imagem e a precisão do valor da temperatura em cada ponto do corpo também aumenta. É bem perceptível essa mudança ao se comparar a malha de 2500 nós com a 40000 nós. Entretanto, para gerar uma melhor qualidade é necessário um esforço maior da máquina e leva mais tempo para obtermos os resultados, como podemos observar abaixo no gráfico que demonstra o tempo computacional levado:

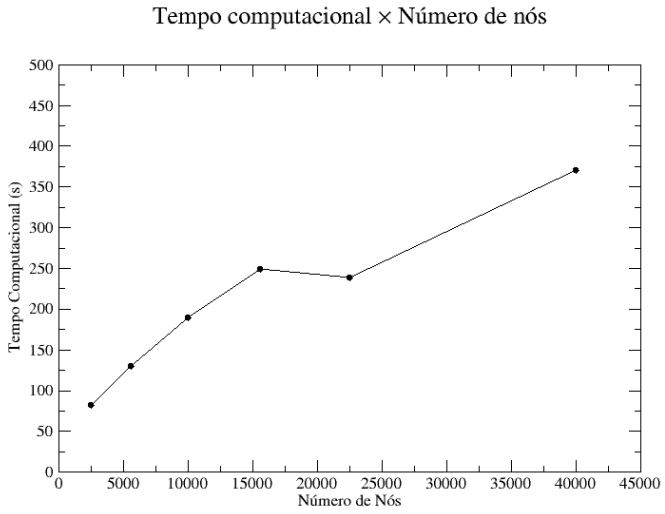

Figura 6 – Gráfico demonstrando o custo operacional por número de nós

Para a produção deste gráfico foi utilizado o software **QtGrace**. Coletado através do programa o tempo levado para rodar os valores de 2500, 5625, 10000, 15625, 22500 e 40000 nós, foi confeccionado manualmente um documento .dat com esses valores e plotado no software. Podemos perceber que majoritariamente obtivemos um resultado linear dado algumas nuances. Isso ocorreu devido ao fato da variável *fourier* depender do tamanho da malha e influenciar no desenvolvimento do programa enquanto o rodamos, podendo acabar influenciando no tempo levado. Além disso, vale ressaltar que para os valores muito grandes da variável *no* de 150 e 200 tive que diminuir o valor da váriável *tempo* para 7000 afim de que a variável *fourier* não ficasse demasiadamente grande e os valores de temperatura extrapolassem o limite e dessem erro, o que acabou influenciando como se pode observar no gráfico.

Analisando, uma percepção que podemos tirar desse gráfico é que um contratempo que pode impedir o aumento da precisão da malha é a quantidade de esforço que se exige da máquina para produzir essa distribuição de temperatura. À medida que foi-se aumentando o número de nós na malha, mais tempo demorava para o código conseguir produzir os resultados e mais da máquina ele cobrava, existindo assim um limite.

4.2 Perfis de Temperatura

Foi plotado os perfis de temperatura para 5 diferentes pontos da malha, sendo estes: $X = 0,5$, $X = 1,5$, $X = 3$, $Y = 1,5$, $Y = 3$. A malha utilizada foi a de 200 por 200 nós devido à grande precisão encontrada nela, dando resultados de qualidade, estes que se encontram abaixo:

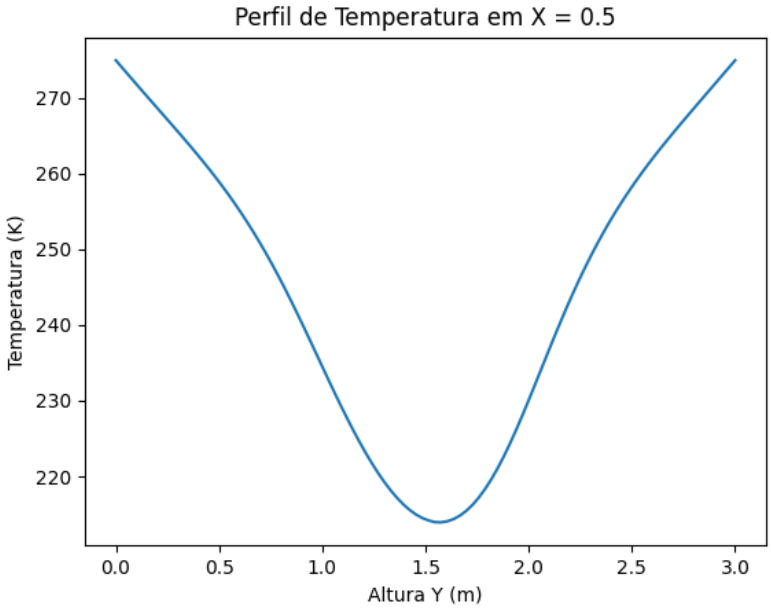

Figura 7 – Variação da temperatura em função da componente Y para $X = 0,5$

Podemos perceber no gráfico acima que perto dos valores 0 e 3 de Y a temperatura se aproxima de 300K, assim como se esperava visto que nesses pontos há a parede com essa condição de contornO. Entre 0 e 3 a temperatura varia atingindo seu valor mínimo próximo de 220K quando $Y \approx 1,5$, como pode-se observar de fato na figura 5.

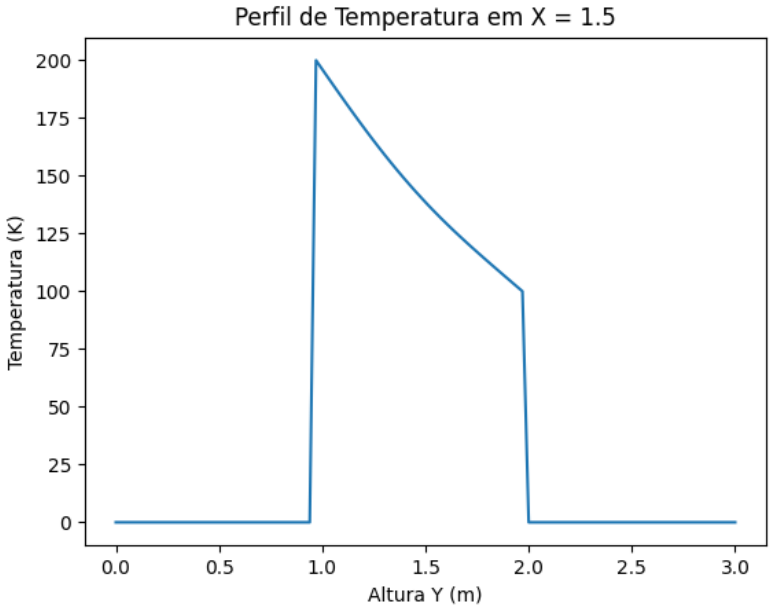

Figura 8 – Variação da temperatura em função da componente Y para $X = 1,5$

Analisando o gráfico acima, percebe-ses que para valores de $Y < 1$ e $Y > 2$ a temperatura é igual a 0K devido ao fato de não fazerem parte do domínio do caso analisado quando $X = 1,5$. Para $Y = 1$ e $Y = 2$ temos respectivamente as temperaturas de 200K e 100K como se esperava devido às condições de contorno. Como para esse valor de X as maiores influências em cada nó vêm das paredes de 100K e 200K, o decaimento da temperatura entre $1 < Y < 2$ é praticamente linear, como pode-se observar de fato na figura 5.

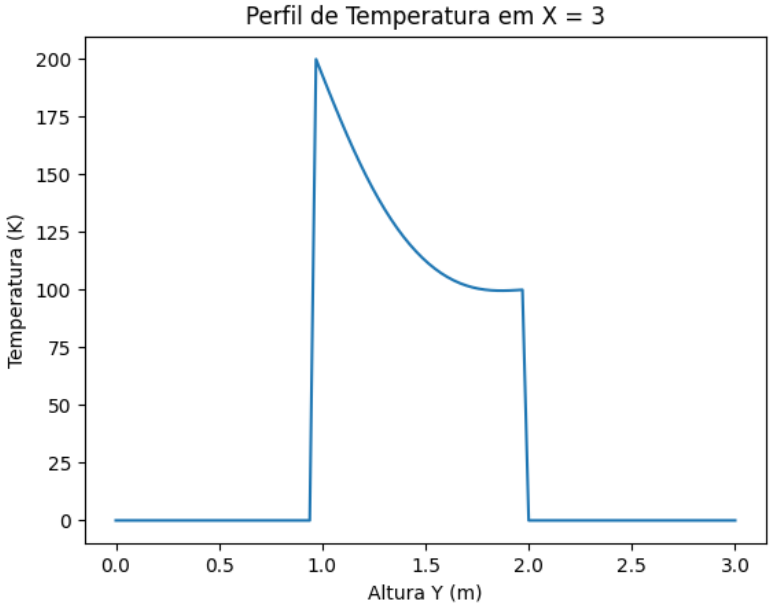

Figura 9 – Variação da temperatura em função da componente Y para $X = 3$

Analogamente ao gráfico anterior, percebe-se que para valores de $Y < 1$ e $Y > 2$ a temperatura é igual a 0K devido ao fato de não fazerem parte do domínio do caso analisado quando $X = 3$ e para $Y = 1$ e $Y = 2$ temos respectivamente as temperaturas de 200K e 100K como se esperava devido às condições de contorno. Entretanto esperava-se observar como o programa interpreta as condições de contorno em bordas de temperaturas muito diferentes. Entre $1 < Y < 2$ esperava-se observar o valor de 40K devido à condição de contorno, o que não ocorre de fato e sim há uma atenuação muito grande da temperatura entre 200K até 100K, devido à uma tentativa do programa em não gerar funções descontínuas.

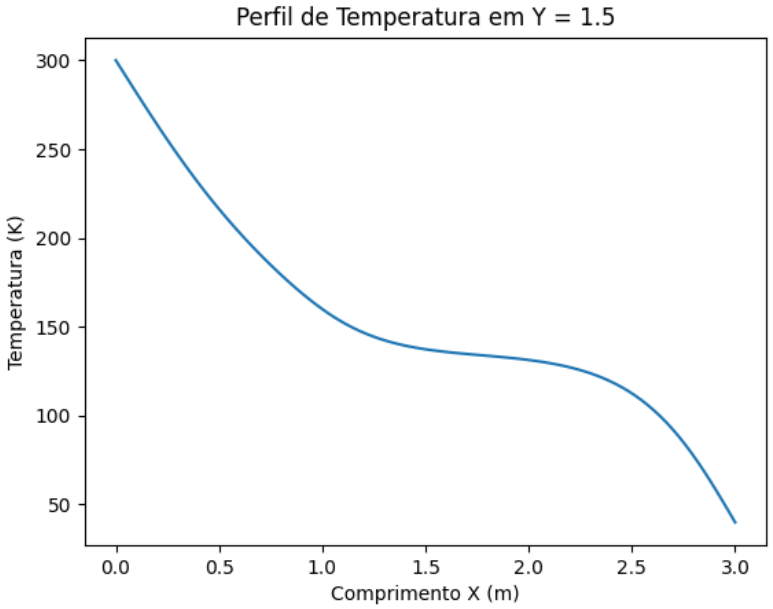

Figura 10 – Variação da temperatura em função da componente X para $Y = 1,5$

Analisando agora em relação ao eixo X, percebe-se que para o valor de $Y = 1,5$ toda a extensão do eixo X faz parte do domínio do caso analisado. Quando $X = 0$ e $X = 3$ temos respectivamente as temperaturas de 300K e 40K assim como se esperava devido às condições de contorno. Entre esse valores a temperatura decai praticamente linearmente até $X = 1$ onde começa a ter influência das paredes encima e embaixo com os valores de 100K e 200K. Observando a figura 5 percebe-se que o resultado obtido está de acordo com o esperado.

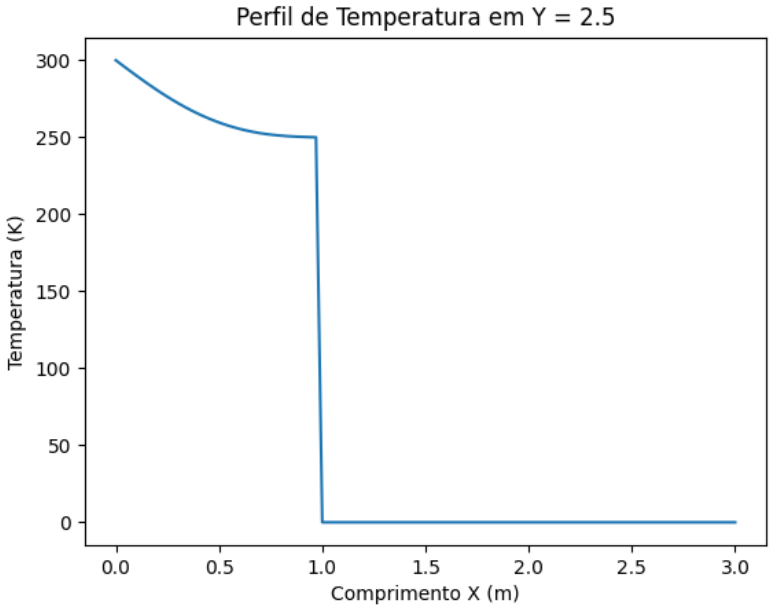

Figura 11 – Variação da temperatura em função da componente X para $Y = 2,5$

Por fim, percebe-se que para o valor de $X > 1$ a temperatura é igual a 0K devido ao fato de não fazerem parte do domínio do caso analisado quando $Y = 2,5$. Para $X = 0$ e $X = 1$ temos respectivamente as temperaturas de 300K e 250K como se esperava devido às condições de contorno e entre as valores a temperatura decai curvalmente, sem ser linear como em $Y = 1,5$, devido à proximidade com a parede de 300K em $Y = 3$.

5 Conclusão

A partir da construção do código para a resolução do problema apresentado na figura 1, pode-se perceber como o método das diferenças finitas é prático e a partir de um auxílio computacional conseguimos descrever a distribuição de temperatura em um corpo com diferentes geometrias, analisando a temperatura interna em cada ponto de seu domínio. Os resultados obtidos foram surpreendentes e extremamente interessantes, onde pode-se comprovar que estavam de acordo com o esperado na maioria dos casos.

Além do citado, o trabalho também me proporcionou uma reaproximação com a programação, principalmente a linguagem **Python**, que estava ficando defasada em meus conhecimentos devido ao desuso dela. Por isso, devo agradecer ao professor. Obrigado.