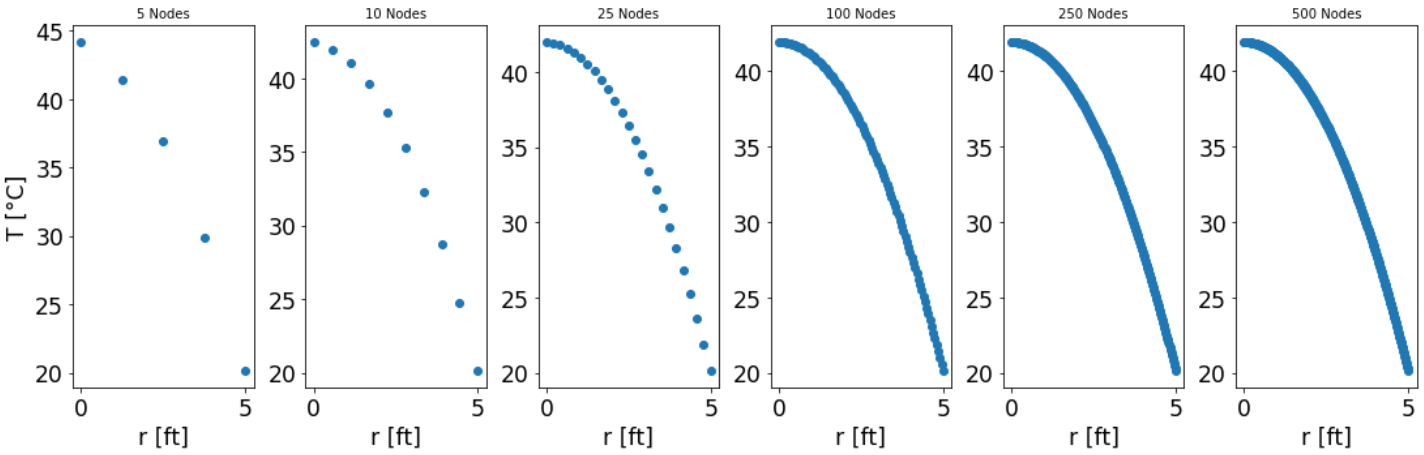
Problem 1

In [16]:

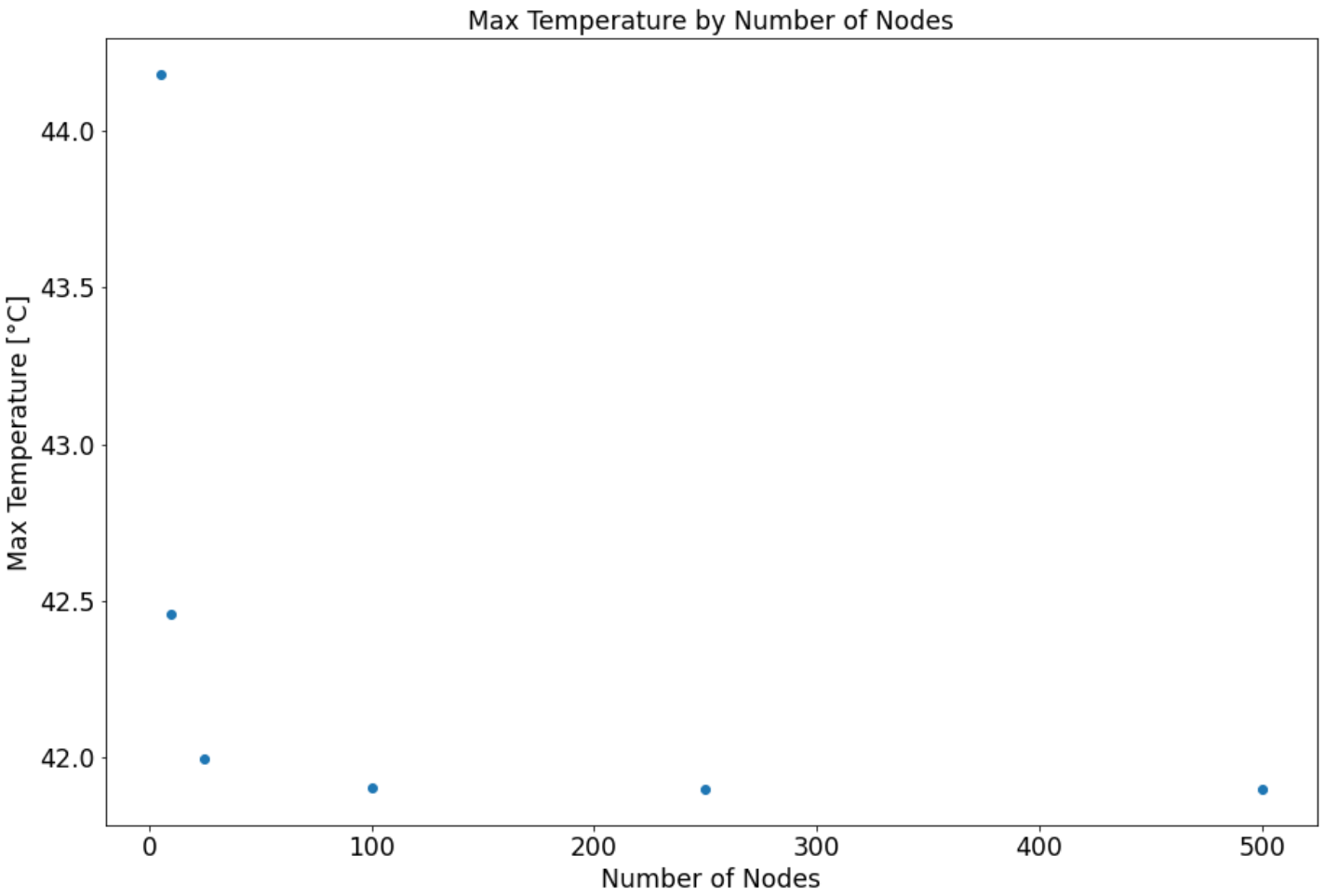
```
#@title
# Set up the python script
# import required modules

#Numerical Math
import math # math functions
import numpy as np # numerical math
from numpy import *
import time

# Scientific Math
import scipy # scientific computing functions
import scipy.special as sp # scientific computing functions

#Symbolic math
import sympy # symbolic math
from sympy import *
from sympy.abc import x

#Import contsants.
from scipy.constants import pi

# Widgets
import ipywidgets as widgets
from IPython.display import display
from ipywidgets import interact

#plotting
params = {'legend.fontsize': 'xx-large',
          'figure.figsize': (15, 10),
         'axes.labelsize': 'xx-large',
         'axes.titlesize':'xx-large',
         'xtick.labelsize':'xx-large',
         'ytick.labelsize':'xx-large'}
import matplotlib.pyplot as plt
plt.rcParams.update(params)

# 1. Set Up

Tinf = 20  #(K) Air temperature
h = 10    #(W/m^2-K) convection coefficient
R_bale = 5*0.3048       #(m) length of plane wall
q_dot = 1.5  #(W/m^5) Coefficient in heat generation rate equation: q_dot = q0*(L^2-x^2)
k =0.04     #(W/m-K) thermal conductivity
k_p=0.15
tp=0.045*0.0254
R1=ln((R_bale+tp)/R_bale)/(2*math.pi*k_p)
R2=1/(2*math.pi*(R_bale+tp)*h)
R_eq=R1+R2
N = 500

q = q_dot;


def tbale(N):
    A=np.zeros((N,N))
    b = np.zeros(N)
    dr = R_bale/(N-1)
    ri=dr
    # 2. Interior Nodes (1...N-2 as python counts from 0)
    for j in range(1,N-1):
        rii=ri-(dr/2)
```

```
        rio=ri+(dr/2)
        A[j,j] = (rii+rio)/dr**2
        A[j,j-1] = -rii/dr**2
        A[j,j+1] = -rio/dr**2
        b[j] = q_dot*ri/k
        ri=ri+dr


    #r=0
    A[0,0] = 1
    A[0,1]=-1
    b[0] = (q_dot*dr**2)/(2*k)

    #r=R_bale
    R_balei=R_bale-(dr/2)
    A[N-1,N-2] = 2*math.pi*R_balei*k/dr
    A[N-1,N-1] = (-k*2*math.pi*R_balei/dr)+(-1/R_eq)
    b[N-1] = -Tinf/R_eq-q_dot*math.pi*R_bale*dr


    # 4. Solve
    t0=time.time()
    T=np.linalg.solve(A,b)
    t1=time.time()
    time_elapsed=t1-t0
    return T, time_elapsed
Ns=np.array([5,10,25,100,250,500])
Tmax=np.zeros(len(Ns))
time_elapsed=np.zeros(len(Ns))
fig, ax=plt.subplots(1,6,figsize=(15,5))
m2=3.28084
for i in range(len(Ns)):
    Tb,time_elapsed[i]=tbale(Ns[i])
    Tmax[i]=np.max(Tb)
    r=np.linspace(0,R_bale, Ns[i])
    ax[i].plot(r*m2,Tb, 'o')
    ax[i].set_xlabel('r [ft]')
    ax[i].set_title(f'{Ns[i]:1.0f} Nodes', size=10)
fig.tight_layout()
ax[0].set_ylabel('T [°C]')
plt.show()
```



In [18]:

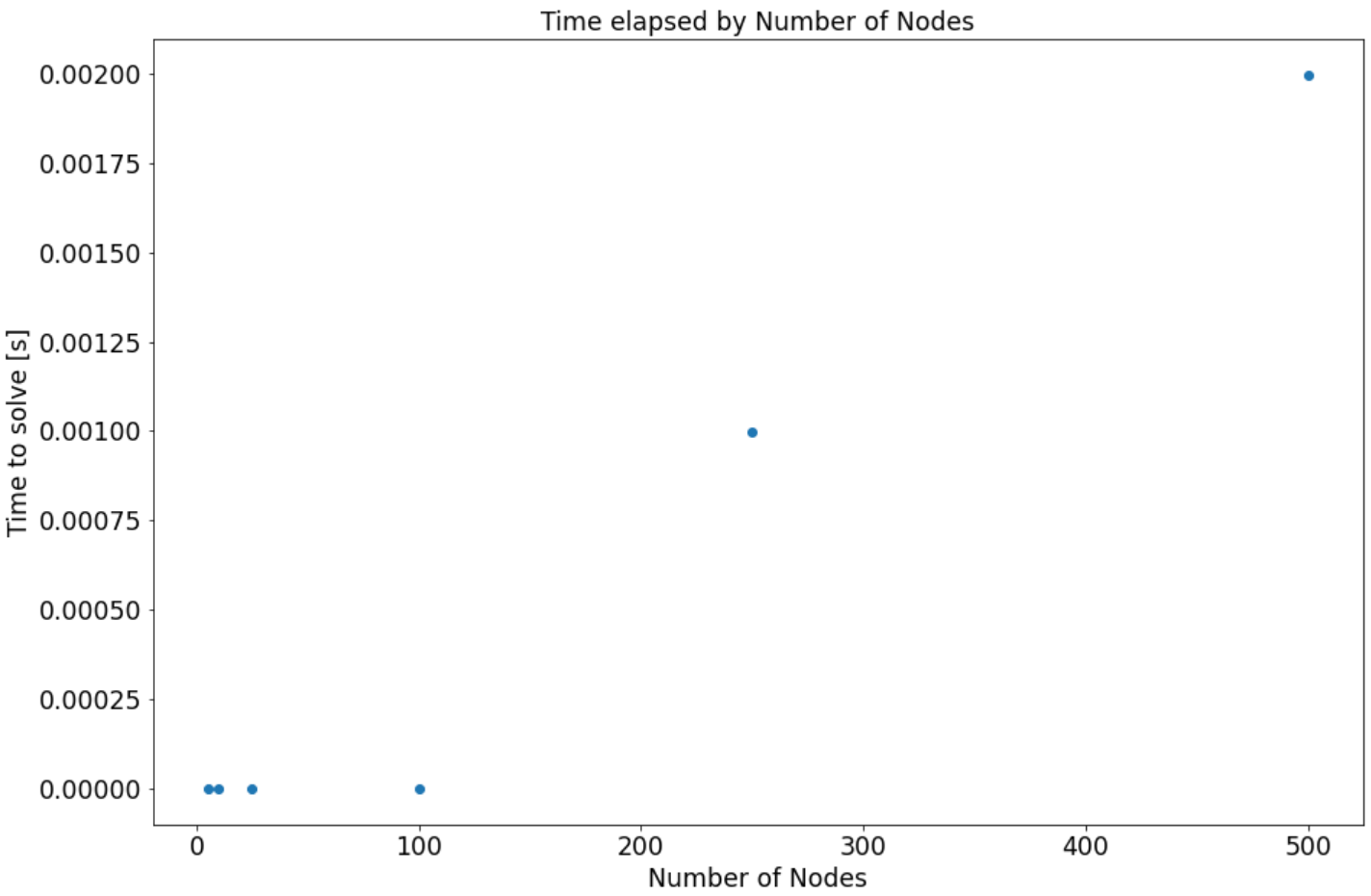
```
fig,ax=plt.subplots()
ax.plot(Ns, Tmax, 'o')
ax.set_title('Max Temperature by Number of Nodes')
ax.set_xlabel('Number of Nodes')
ax.set_ylabel('Max Temperature [°C]')
plt.show()
print('N:\t 5 \t\t 25 \t\t 100 \t\t 250 \t\t 500')
print(f'Tmax: \t {Tmax[0]:1.5f} \t {Tmax[1]:1.5f} \t {Tmax[2]:1.5f} \t {Tmax[3]:1.5f} \t {Tmax[4]:1.5
```




```
N:      5           25          100         250         500
Tmax:   44.18201    42.45714    41.99436    41.90436    41.89839    41.89743
```

In [19]:

```
fig,ax=plt.subplots()
ax.plot(Ns, time_elapsed, 'o')
ax.set_title('Time elapsed by Number of Nodes')
ax.set_xlabel('Number of Nodes')
ax.set_ylabel('Time to solve [s]')
plt.show()
print('N:\t 5 \t\t 25 \t\t 100 \t\t 250 \t\t 500')
print(f'Tmax: \t {time_elapsed[0]:1.5f} \t {time_elapsed[1]:1.5f} \t {time_elapsed[2]:1.5f} \t {time_
```

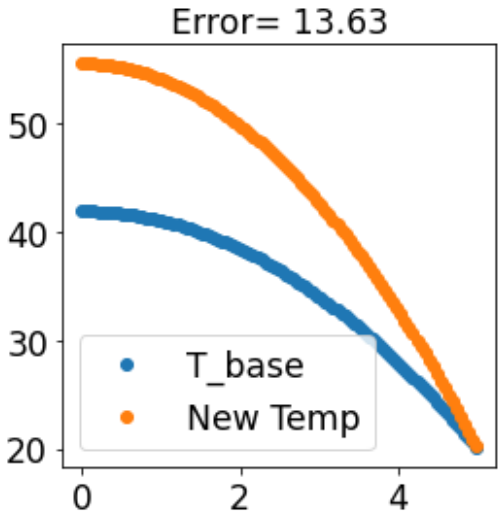


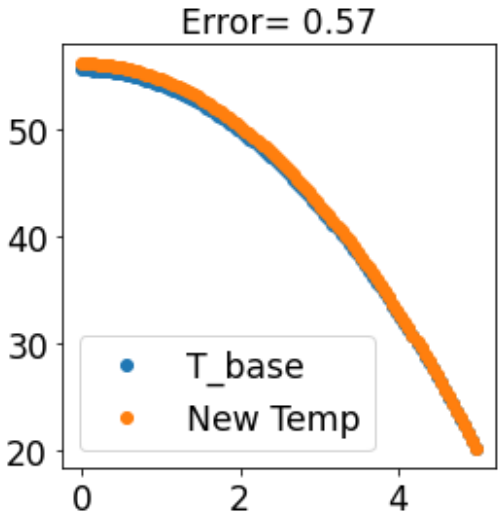
N:	5	25	100	250	500	
Tmax:	0.00000	0.00000	0.00000	0.00000	0.00100	0.00200

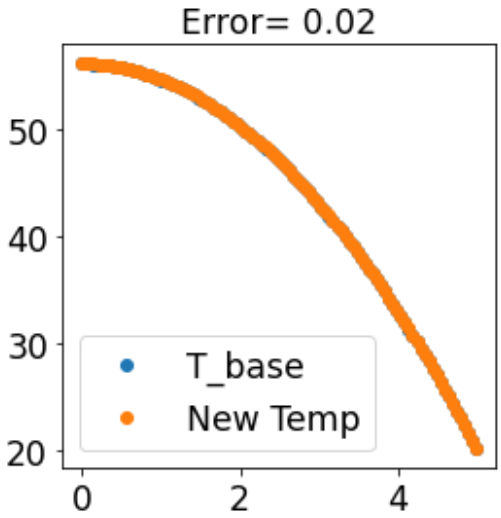
In [21]:

```python
def TbaleQdot(N, q_dot):
    A=np.zeros((N,N))
    b=np.zeros(N)
    dr=R_bale/(N-1)
    ri=dr
    for j in range(1,N-1):
        rii=ri-(dr/2)
        rio=ri+(dr/2)
        A[j,j] = (rii+rio)/dr**2
        A[j,j-1] = -rii/dr**2
        A[j,j+1] = -rio/dr**2
        b[j] = q_dot[j]*ri/k
        ri=ri+dr


    #r=0
    A[0,0] = 1
    A[0,1]=-1
    b[0] = (q_dot[0]*dr**2)/(2*k)

    #r=R_bale
    R_balei=R_bale-(dr/2)
    A[N-1,N-2] = 2*math.pi*R_balei*k/dr
    A[N-1,N-1] = (-k*2*math.pi*R_balei/dr)+(-1/R_eq)
    b[N-1] = -Tinf/R_eq-q_dot[N-1]*math.pi*R_bale*dr

    # 4. Solve
    t0=time.time()
    T=np.linalg.solve(A,b)
    Temp=T+273.15
    t1=time.time()
    time_elapsed=t1-t0
    return Temp
def qdotT(T):
```

```
    qdotTemp=1.5*np.exp((T)/320)**0.5
    return qdotTemp
N=100
r=np.linspace(0, R_bale, N)
Tb=tbale(N)[0]
Tbase=Tb+273.15
Tmax_estimate=np.array([np.max(Tbase)])
w=0
esp=0.1
err=100
while err>esp:
    w=w+1
    if w>1:
        Tbase=Tnew
    Tnew=TbaleQdot(N, qdotT(Tbase))
    Tmax_estimate=np.append(Tmax_estimate, np.max(Tnew))
    err=np.abs(Tmax_estimate[w]-Tmax_estimate[w-1])
    plt.figure(figsize=(4,4))
    plt.plot(r*m2, Tbase-273.15, 'o', label='T_base')
    plt.plot(r*m2, Tnew-273.15, 'o', label='New Temp')
    plt.title(f'Error= {err:1.2f}')
    plt.legend()
    plt.show()
print(f'Converged within {err:1.3f} K after {w:d} iterations')
```



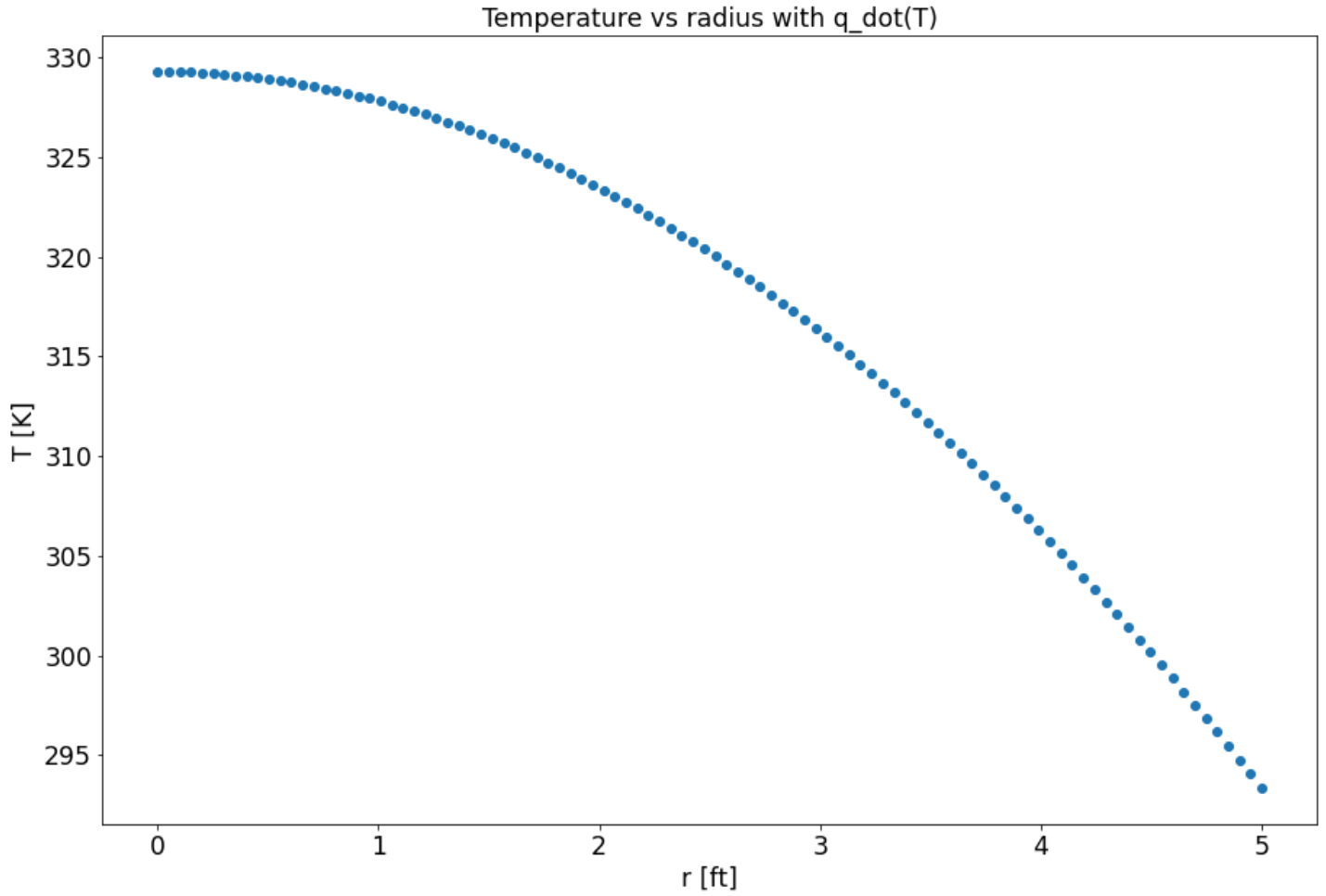





```
Converged within 0.023 K after 3 iterations
```

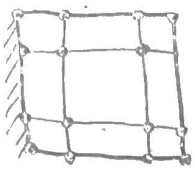
In [22]:

```
plt.figure()
plt.title('Temperature vs radius with q_dot(T)')
plt.plot(r*m2, Tnew, 'o')
plt.xlabel('r [ft]')
plt.ylabel('T [K]')
plt.show()
```

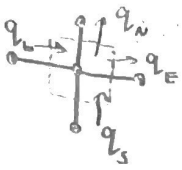


Problem 2

Problem 2



(i) Interior node:



$$\dot{E}_{in} - \dot{E}_{out} + \dot{E}_{gen} = \dot{E}_{st}$$

$$q_W + q_S - q_E - q_N = 0$$

$$-k\Delta z\Delta y\,\frac{T_{i,j}-T_{i-1,j}}{\Delta x} + -k\Delta z\Delta x\,\frac{T_{i,j}-T_{i,j-1}}{\Delta y} - -k\Delta z\Delta y\,\frac{T_{i+1,j}-T_{i,j}}{\Delta x} - -k\Delta z\Delta x\,\frac{T_{i,j+1}-T_{i,j}}{\Delta y} = 0$$

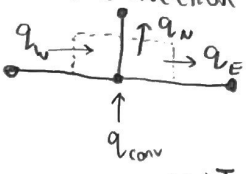
$$-k\Delta y^2(T_{i,j}-T_{i-1,j}) - k\Delta x^2(T_{i,j}-T_{i,j-1}) + k\Delta y^2(T_{i+1,j}-T_{i,j}) + k\Delta x^2(T_{i,j+1}-T_{i,j}) = 0$$

$$k\Delta y^2(-T_{i,j}+T_{i-1,j}+T_{i+1,j}-T_{i,j}) + k\Delta x^2(-T_{i,j}+T_{i,j-1}+T_{i,j+1}-T_{i,j}) = 0$$

$$\frac{T_{i-1,j}-2T_{i,j}+T_{i+1,j}}{(\Delta x)^2} + \frac{T_{i,j+1}-2T_{i,j}+T_{i,j-1}}{(\Delta y)^2} = 0$$

$$\frac{1}{(\Delta x)^2}T_{i-1,j} + \frac{1}{(\Delta y)^2}T_{i,j-1} + \left(\frac{-2}{(\Delta x)^2}+\frac{-2}{(\Delta y)^2}\right)T_{i,j} + \frac{1}{(\Delta x)^2}T_{i+1,j} + \frac{1}{(\Delta y)^2}T_{i,j+1} = 0$$

(ii) Convection node



$$\dot{E}_{in} - \dot{E}_{out} + \dot{E}_{gen} = \dot{E}_{st}$$

$$q_W + q_{conv} - q_N - q_E = 0$$

$$-k\Delta z\left(\frac{\Delta y}{2}\right)\frac{T_{i,j}-T_{i-1,j}}{\Delta x} + h\Delta z\Delta x(T_\infty - T_{i,j}) - -k\Delta z\Delta x\,\frac{T_{i,j+1}-T_{i,j}}{\Delta y} - -k\Delta z\left(\frac{\Delta y}{2}\right)\frac{T_{i+1,j}-T_{i,j}}{\Delta x} = 0$$

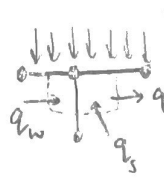
$$-k\,\frac{\Delta y^2}{2}(T_{i,j}-T_{i-1,j}) + h\Delta x^2\Delta y(T_\infty - T_{i,j}) + k\Delta x^2(T_{i,j+1}-T_{i,j}) + k\left(\frac{\Delta y^2}{2}\right)(T_{i+1,j}-T_{i,j}) = 0$$

$$T_{i,j}\left(-k\frac{\Delta y^2}{2} - h\Delta x^2\Delta y - k\Delta x^2 - k\frac{\Delta y^2}{2}\right) + T_{i-1,j}\left(k\frac{\Delta y^2}{2}\right) + T_{i+1,j}\left(k\frac{\Delta y^2}{2}\right) + T_{i,j+1}(k\Delta x^2) + h\Delta x^2\Delta y T_\infty = 0$$

$$-2T_{i,j}\left(\Delta y^2 + \frac{h\Delta x^2\Delta y}{k} + \Delta x^2\right) + T_{i-1,j}\Delta y^2 + T_{i+1,j}\Delta y^2 + 2T_{i,j+1}\Delta x^2 + 2\frac{h}{k}\Delta x^2\Delta y T_\infty = 0$$

$$\frac{1}{2(\Delta x)^2}T_{i-1,j} + \left(\frac{-1}{(\Delta x)^2}+\frac{-1}{(\Delta y)^2}+\frac{-h}{k\Delta y}\right)T_{i,j} + \frac{1}{2(\Delta x)^2}T_{i+1,j} + \frac{1}{(\Delta y)^2}T_{i,j+1} = \frac{-h}{k\Delta y}\theta_\infty$$

(iii) top surface



$\dot{E}_{in} - \dot{E}_{out} + \dot{E}_{gen} = \dot{E}_{st}$

$q_W + q_S + q_m - q_E = 0$

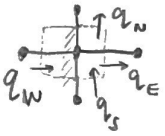
$$-k\Delta z\left(\frac{\Delta y}{2}\right)\frac{T_{i,j}-T_{i-1,j}}{\Delta x} + -k\Delta z\Delta x\,\frac{T_{i,j}-T_{i,j-1}}{\Delta y} + q''_m\Delta z\Delta x - -k\Delta z\left(\frac{\Delta y}{2}\right)\frac{T_{i+1,j}-T_{i,j}}{\Delta x} = 0$$

$$-k\Delta y^2(T_{i,j}-T_{i-1,j}) - 2k\Delta x^2(T_{i,j}-T_{i,j-1}) + 2q''_m\Delta x^2\Delta y + k\Delta y^2(T_{i+1,j}-T_{i,j}) = 0$$

$$\Delta y^2(-T_{i,j}+T_{i-1,j}+T_{i+1,j}-T_{i,j}) + \Delta x^2(-2T_{i,j}+2T_{i,j-1}) + 2q''_m\Delta x^2\Delta y = 0$$

$$\frac{1}{2(\Delta x)^2}T_{i-1,j} + \frac{1}{(\Delta y)^2}T_{i,j-1} + \left(\frac{-1}{(\Delta x)^2} + \frac{-1}{(\Delta y)^2}\right)T_{i,j} + \frac{1}{2(\Delta x)^2}T_{i+1,j} = -\frac{q''_m}{k\Delta y}$$

(iii) A node on the left face



$\dot{E}_{in} - \dot{E}_{out} + \dot{E}_{gen} = \dot{E}_{st}$

$q_S + q_W - q_E - q_N = 0$ $\quad T_{i-1} = T_{i+1}$

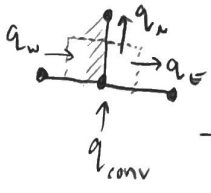
$$-k\Delta z\,\Delta x\,\frac{T_{i,j}-T_{i,j-1}}{\Delta y} + -k\Delta z\Delta y\,\frac{T_{i,j}-T_{i+1,j}}{\Delta x} - -k\Delta z\Delta y\,\frac{T_{i+1,j}-T_{i,j}}{\Delta x} - -k\,\frac{T_{i,j+1}-T_{i,j}}{\Delta y}\Delta x\Delta z = 0$$

$$-k\Delta x^2(T_{i,j}-T_{i,j-1}) - k\Delta y^2(T_{i,j}-T_{i+1,j}) + k\Delta y^2(T_{i+1,j}-T_{i,j}) + k\Delta x^2(T_{i,j+1}-T_{i,j}) = 0$$

$$k\Delta x^2(-T_{i,j}+T_{i,j-1}-T_{i,j}+T_{i,j+1}) + k\Delta y^2(-T_{i,j}+T_{i+1,j}+T_{i+1,j}-T_{i,j}) = 0$$

$$\frac{1}{(\Delta y)^2}T_{i,j-1} + \left(\frac{-2}{(\Delta x)^2} + \frac{-2}{(\Delta y)^2}\right)T_{i,j} + \frac{2}{(\Delta x)^2}T_{i+1,j} + \frac{1}{(\Delta y)^2}T_{i,j+1} = 0$$

(iv) Bottom left



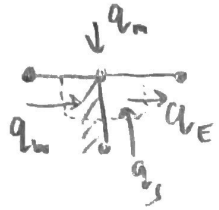
$T_{i-1} = T_{i+1}$ $\quad \dot{E}_{in} - \dot{E}_{out} + \dot{E}_{gen} = \dot{E}_{st}$ (both $= 0$)

$q_W + q_{conv} - q_N - q_E = 0$

$$-k\Delta z\left(\frac{\Delta y}{2}\right)\frac{T_{i,j}-T_{i-1,j}}{\Delta x} + h\Delta x\Delta z(T_\infty - T_{i,j}) - -k\Delta z\Delta x\,\frac{T_{i,j+1}-T_{i,j}}{\Delta y} - -k\Delta z\left(\frac{\Delta y}{2}\right)\frac{T_{i+1,j}-T_{i,j}}{\Delta x} = 0$$

$$\left(\frac{-1}{(\Delta x)^2} + \frac{-1}{(\Delta y)^2} + \frac{-h}{k\Delta y}\right)T_{i,j} + \frac{1}{(\Delta x)^2}T_{i+1,j} + \frac{1}{(\Delta y)^2}T_{i,j+1} = \frac{-hT_\infty}{k\Delta y}$$

(v) top / Left Corner -



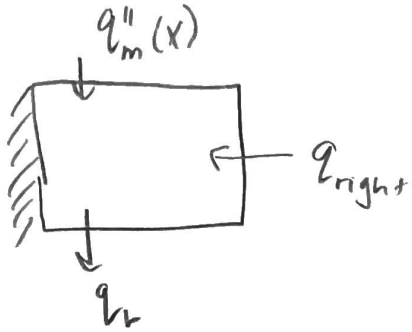
$$\dot{E}_{in} - \dot{E}_{out} + \dot{E}_{gen}^{0} = \dot{E}_{st}^{0} \qquad T_{i-1} = T_{i+1}$$

$$q_w + q_s + q_m - q_E = 0$$

$$-k\,\Delta z\left(\frac{\Delta y}{2}\right)\frac{T_{i,j} - T_{i-1,j}}{(\Delta x)} + -k\,\Delta z\,\Delta x\,\frac{T_{i,j} - T_{i,j-1}}{\Delta y} + q_m$$

$$- -k\,\Delta z\left(\frac{\Delta y}{2}\right)\frac{T_{i+1,j} - T_{i,j}}{\Delta x} = 0$$

$$\frac{1}{(\Delta y)^2}T_{i,j-1} + \left(\frac{-1}{(\Delta x)^2} + \frac{-1}{(\Delta y)^2}\right)T_{i,j} + \frac{1}{(\Delta x)^2}T_{i+1,j} = -\frac{q''_m}{k\,\Delta y}$$

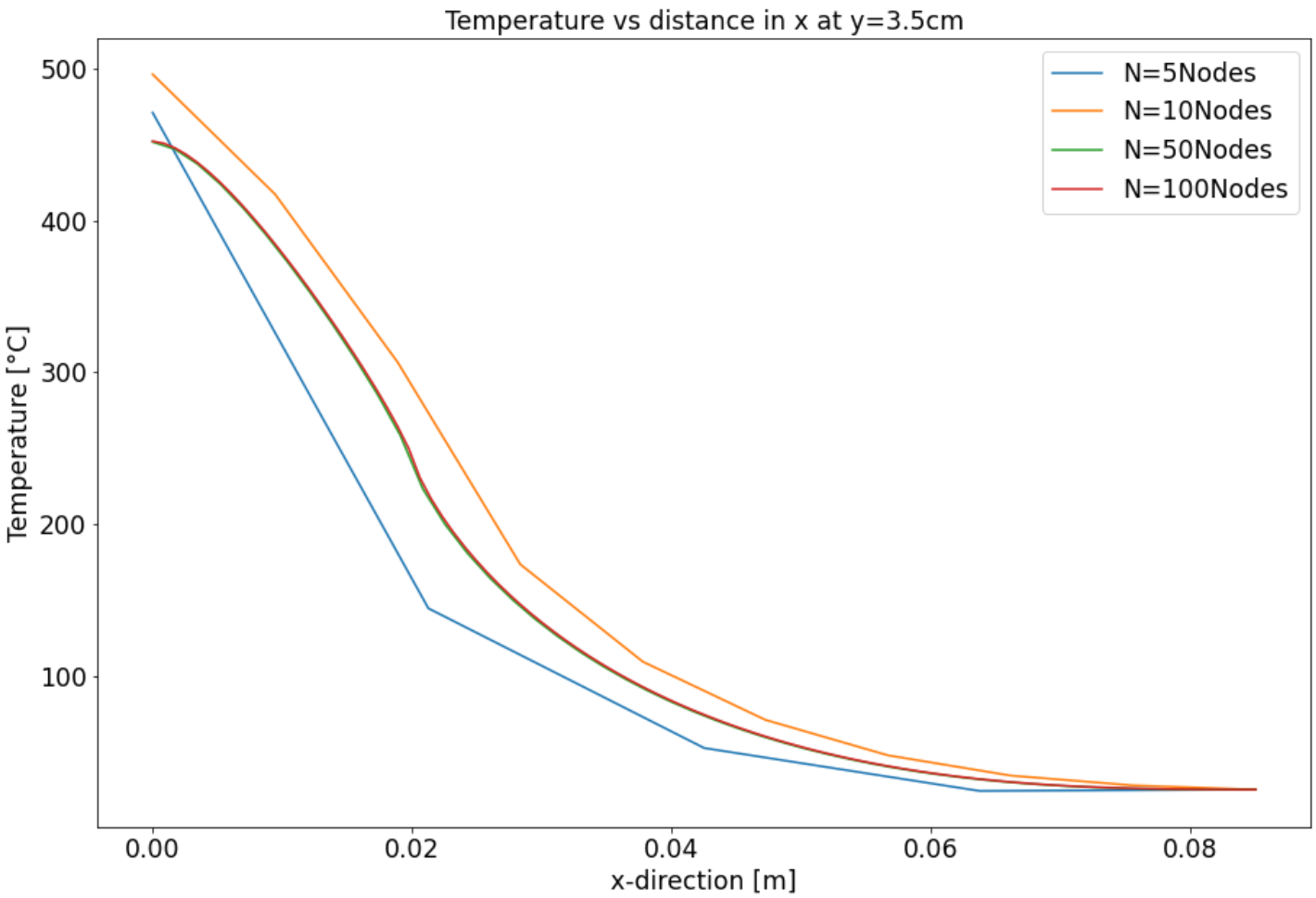


In [23]:

```
import math
# 1. Set Up
N = 50

b = 3.5*10**-2      # [m] Length
Lj=2*10**(-2)
w = 8.5*10**-2    # [m] Width
k = 38      # [W/m K] thermal conductivity
```

```
# h [W/m^2 K] Convection coefficient [south face]
# Tinf [K] Temperature for convection [south face]e
h = 5000
Tf = -35+273.15
Ts=25+273.15
qj=10**6
Lj=2*10**-2


N = [5,10,50,100]
eps=0
M = N
x = [None, None, None, None]
y = [None, None, None, None]
T = [None, None, None, None]
for t in range(4):   # python stops at upper value -1
    n=N[t]
    m=N[t]
    dx=w/(n-1)
    dy=b/(n-1)
    x[t]=np.linspace(0,w,num=n)
    y[t]=np.linspace(0,b,num=n)
    A=np.zeros((n*m, n*m))
    c=np.zeros(n*m)
    d=np.zeros(n*m)

    for i1 in range(1,n-1):
        for j1 in range(1, m-1):
            ind=i1+j1*n
            A[ind, ind]=-2/dx**2-2/dy**2
            A[ind, ind-1]=1/dx**2
            A[ind, ind+1]=1/dx**2
            A[ind, ind-n]=1/dy**2
            A[ind, ind+n]=1/dy**2
            d[ind]=esp
            c[ind]=ind
    for i2 in range(1,n-1):
        j2=0
        ind=j2*n+i2
        A[ind, ind]=-h/dy-k/dx**2-k/dy**2
        A[ind, ind-1]=k/(2*dx**2)
        A[ind, ind+1]=k/(2*dx**2)
        A[ind, ind+n]=k/dy**2
        d[ind]=-h/dy*Tf
        c[ind]=ind
    for i3 in range(1,n-1):
        j3=m-1
        ind=j3*n+i3
        A[ind, ind]=-k/dx**2-k/dy**2
        A[ind, ind-1]=k/(2*dx**2)
        A[ind, ind+1]=k/(2*dx**2)
        A[ind, ind-n]=k/dy**2
        if i3*dx<Lj:
            d[ind]=-(qj*exp(-i3*dx/Lj))/dy
        else:
            d[ind]=esp
        c[ind]=ind
    for j4 in range(m):
        i4=n-1
        ind=j4*n+i4
        A[ind, ind]=1
        d[ind]=Ts
        c[ind]=ind
    i7=0
    for j7 in range(1,m-1):
        ind=j7*n+i7
```

```
            A[ind, ind]=-k/dx**2-k/dy**2
            A[ind, ind-n]=k/(2*dy**2)
            A[ind, ind+n]=k/(2*dy**2)
            A[ind, ind+1]=k/dx**2
            d[ind]=esp
            c[ind]=ind
        i5=0
        j5=0
        ind=j5*n+i5
        A[ind, ind]=-h/(2*dy)-k/(2*dx**2)-k/(2*dy**2)
        A[ind, ind+n]=k/(2*dy**2)
        A[ind, ind+1]=k/(2*dx**2)
        d[ind]=-h/(2*dy)*Tf
        c[ind]=ind

        i6=0
        j6=m-1
        ind=i6+j6*n
        A[ind, ind]=-k/(2*dx**2)-k/(2*dy**2)
        A[ind, ind-n]=k/(2*dy**2)
        A[ind, ind+1]=k/(2*dx**2)
        d[ind]=-qj/(2*dy)
        c[ind]=ind

        Temp=np.linalg.solve(A,d)
        TT=np.reshape(Temp, (m,n))
        T[t]=Temp[n*(m-1):n*m]

for i in range(4):
    label=str("N=" +str(N[i])+'Nodes')
    plt.plot(x[i],T[i]-273.15, label=label)
    plt.xlabel('x-direction [m]')
    plt.ylabel('Temperature [°C]')
    plt.title('Temperature vs distance in x at y=3.5cm')
plt.legend()
plt.show()
```



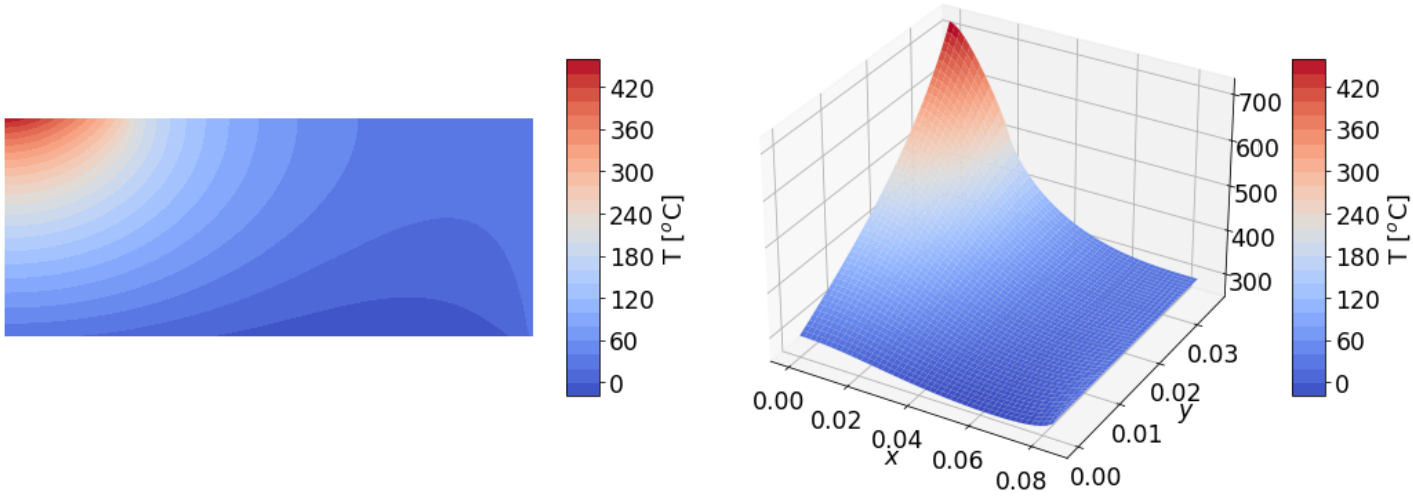
In [24]:

```
import math
# 1. Set Up
N = 50
eps=0
start=time.time()


N = 50
eps=0
M = N
x = [None, None, None, None]
y = [None, None, None, None]
T = [None, None, None, None]

n=N
m=N
dx=w/(n-1)
dy=b/(n-1)
x=np.linspace(0,w,num=n)
y=np.linspace(0,b,num=n)
A=np.zeros((n*m, n*m))
c=np.zeros(n*m)
d=np.zeros(n*m)

for i1 in range(1,n-1):
    for j1 in range(1, m-1):
        ind=i1+j1*n
        A[ind, ind]=-2/dx**2-2/dy**2
        A[ind, ind-1]=1/dx**2
        A[ind, ind+1]=1/dx**2
        A[ind, ind-n]=1/dy**2
        A[ind, ind+n]=1/dy**2
        d[ind]=esp
        c[ind]=ind
```

```
for i2 in range(1,n-1):
    j2=0
    ind=j2*n+i2
    A[ind, ind]=-h/dy-k/dx**2-k/dy**2
    A[ind, ind-1]=k/(2*dx**2)
    A[ind, ind+1]=k/(2*dx**2)
    A[ind, ind+n]=k/dy**2
    d[ind]=-h/dy*Tf
    c[ind]=ind
for i3 in range(1,n-1):
    j3=m-1
    ind=j3*n+i3
    A[ind, ind]=-k/dx**2-k/dy**2
    A[ind, ind-1]=k/(2*dx**2)
    A[ind, ind+1]=k/(2*dx**2)
    A[ind, ind-n]=k/dy**2
    if i3*dx<Lj:
        d[ind]=-(qj*exp(-i3*dx/Lj))/dy
    else:
        d[ind]=esp
    c[ind]=ind
for j4 in range(m):
    i4=n-1
    ind=j4*n+i4
    A[ind, ind]=1
    d[ind]=Ts
    c[ind]=ind
i7=0
for j7 in range(1,m-1):
    ind=j7*n+i7
    A[ind, ind]=-k/dx**2-k/dy**2
    A[ind, ind-n]=k/(2*dy**2)
    A[ind, ind+n]=k/(2*dy**2)
    A[ind, ind+1]=k/dx**2
    d[ind]=esp
    c[ind]=ind
i5=0
j5=0
ind=j5*n+i5
A[ind, ind]=-h/(2*dy)-k/(2*dx**2)-k/(2*dy**2)
A[ind, ind+n]=k/(2*dy**2)
A[ind, ind+1]=k/(2*dx**2)
d[ind]=-h/(2*dy)*Tf
c[ind]=ind

i6=0
j6=m-1
ind=i6+j6*n
A[ind, ind]=-k/(2*dx**2)-k/(2*dy**2)
A[ind, ind-n]=k/(2*dy**2)
A[ind, ind+1]=k/(2*dx**2)
d[ind]=-qj/(2*dy)
c[ind]=ind

Temp=np.linalg.solve(A,d)
TT=np.reshape(Temp, (m,n))
T[t]=Temp[n*(m-1):n*m]

# 5. Plot
xx,yy = meshgrid(x,y)
fig = plt.figure(figsize=plt.figaspect(0.5))
ax = fig.add_subplot(1, 2, 1)
fig.subplots_adjust(wspace=0.1)
fig.set_figheight(9)
fig.set_figwidth(18)

ct = ax.contourf(xx,yy,TT-273.15,25,cmap='coolwarm')
```

```
ax.axis('equal');
cbar = fig.colorbar(ct, shrink=0.5, aspect=10)
cbar.set_label('T [$^o$C]', fontsize = 18)
ax.axis('off')

ax = fig.add_subplot(1, 2, 2, projection='3d')
ax.plot_surface(xx, yy, TT, cmap='coolwarm')
ax.set(xlabel='$x$', ylabel='$y$')
cbar =  fig.colorbar(ct, shrink=0.5, aspect=10)
cbar.set_label('T [$^o$C]', fontsize = 18)
end=time.time()
print('runtime', str(end-start)+'s')
```

```
runtime 0.4368300437927246s
```



In [25]:

```
for i in range(0,n):
    qtop=+(-k*(Temp[n*(m-1)-1-i]-Temp[n*m-1-i])/dy)
print('q_top', qtop*w)
```

```
q_top 81287.04578896609
```

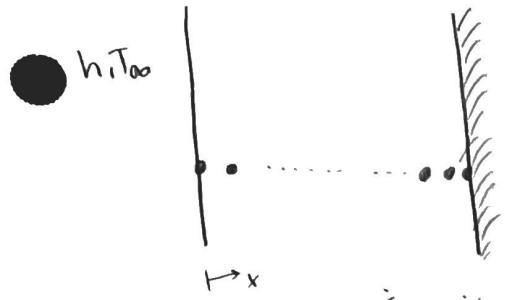
In [26]:

```
for j in range(0,m):
    qright=+(-k*(Temp[(n-1)+n*j-1]-Temp[(n-1)+n*j])/dx)
print('q_right', qright*b)

for i in range(0,n):
    qbottom=+h*(Tf-Temp[i])
print('qbottom', qbottom*w)

for j in range(0,m):
    qleft=+(-k*(Temp[n*j]-Temp[n*j+1])/dx)
print('qleft', qleft*b)

for i in range(0,n):
    qtop=+(-k*(Temp[n*(m-1)-1-i]-Temp[n*m-1-i])/dy)
print('qtop', qtop*w)
qnet=qtop*w+qleft*b+qright*b+qbottom*w
print('qnet', qnet)
```

```
q_right -19.08332046357391
qbottom -25500.00002468088
qleft -3712.9542110337966
qtop 81287.04578896609
qnet 52055.00823278785
```

Problem 3

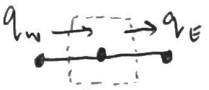
Problem 3



$$\dot{E}_{in} - \dot{E}_{out} + \dot{E}_{gen} = \dot{E}_{st}$$

$$q_w - q_E = \rho C_p \frac{dT}{dt} dV$$

Interior nodes

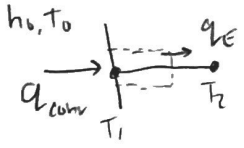


$$-k \Delta z \Delta y \frac{T_i - T_{i-1}}{\Delta x} - -k \Delta z \Delta y \frac{T_{i+1} - T_i}{\Delta x} = \rho C_p \Delta x \Delta y \Delta z \frac{\Delta T}{\Delta t}$$

$$-k \frac{T_i^p - T_{i-1}^p}{\Delta x^2} + k \frac{T_{i+1}^p - T_i^p}{\Delta x^2} = \rho C_p \frac{T_i^{p+1} - T_i^p}{\Delta t}$$

$$\frac{T_{i+1} - 2T_i + T_{i-1}}{\Delta x^2} = \frac{1}{\alpha} \frac{T_i^{p+1} - T_i^p}{\Delta t} \rightarrow (1-2F_0)T_i^p + F_0\left(T_{i-1}^p + T_{i+1}^p\right) = T_i^{p+1}$$

Left Node



$$\dot{E}_{in} - \dot{E}_{out} + \dot{E}_{gen} = \dot{E}_{st}$$

$$q_{conv} - q_E = \rho C_p \frac{dT}{dt} dV$$

$$h(T_0 - T_1) - -k\left(\frac{dT}{dx}\right) = \rho C_p \frac{\partial T}{\partial t}\left(\frac{\Delta x}{2}\right)$$

$$h_0 (T_0 - T_1^p) + k \frac{T_2^p - T_1^p}{dx} = \rho C_p \frac{T_1^{p+1} - T_1^p}{\Delta t}\left(\frac{\Delta x}{2}\right)$$

$$\frac{2h_0}{dx} T_0 - \frac{2h_0}{dx} T_1^p + \frac{2k}{(dx)^2} T_2^p - \frac{2k}{dx^2} T_1^p = \frac{\rho C_p}{dt}\left(T_1^{p+1} - T_1^p\right)$$

$$T_1^{p+1} = \left(1 - \frac{2h_0 dt}{dx \rho C_p} - 2F_0\right)T_1^p + 2F_0 T_2^p + \frac{2h_0 dt}{dx \rho C_p} T_0$$

Right Node

$$\dot{E}_{in} - \dot{E}_{out} + \dot{E}_{gen} = \dot{E}_{st}$$

$$q''_w \Delta y \Delta z = \rho C_p \frac{\partial T}{\partial t} \Delta y \Delta z \left(\frac{\Delta x}{2}\right)$$

$$-k \frac{\partial T}{\partial x} = \rho C_p \frac{\partial T}{\partial t} \left(\frac{\Delta x}{2}\right)$$

$$-k \frac{T_N^p - T_{N-1}^p}{\Delta x} = \rho C_p \frac{T_N^{p+1} - T_N^p}{\Delta t} \left(\frac{\Delta x}{2}\right)$$

$$-\frac{2k}{dx^2} T_N^p + \frac{2k}{dx^2} T_{N-1}^p = \frac{\rho C_p}{dt} T_N^{p+1} - \frac{\rho C_p}{dt} T_N^p$$

$$\boxed{T_N^{p+1} = (1 - 2F_0) T_N^p + 2F_0 T_{N-1}^p}$$

2

Von Neumann stability condition

Interior nodes $1 - 2F_0 > 0 \rightarrow F_0 < 1/2$, $\frac{\alpha dt}{dx^2} < \frac{1}{2}$ $\quad \boxed{dt < \frac{dx^2}{2\alpha}}$

Right Node: $1 - 2F_0 > 0 \rightarrow F_0 < 1/2$ $\quad \boxed{dt < \frac{dx^2}{2\alpha}}$

Left Node: $1 - \frac{2h_0 dt}{dx \rho C_p} - 2F_0 > 0 \rightarrow 1 - \frac{2h_0 dx}{k} \cdot \frac{k\, dt}{(dx)^2 \rho C_p} - 2F_0 > 0$

$1 - \frac{2h_0 dx}{k} F_0 - 2F_0 > 0$ * Wolfram Alpha *

$$\boxed{dt < \frac{k\, dx^2}{\alpha (2h_0 dx + 2k)}}$$

3

Pick N = 25 nodes b/c its fairly smooth & run time isn't long, Neuman Stability Condition is met

In [27]:

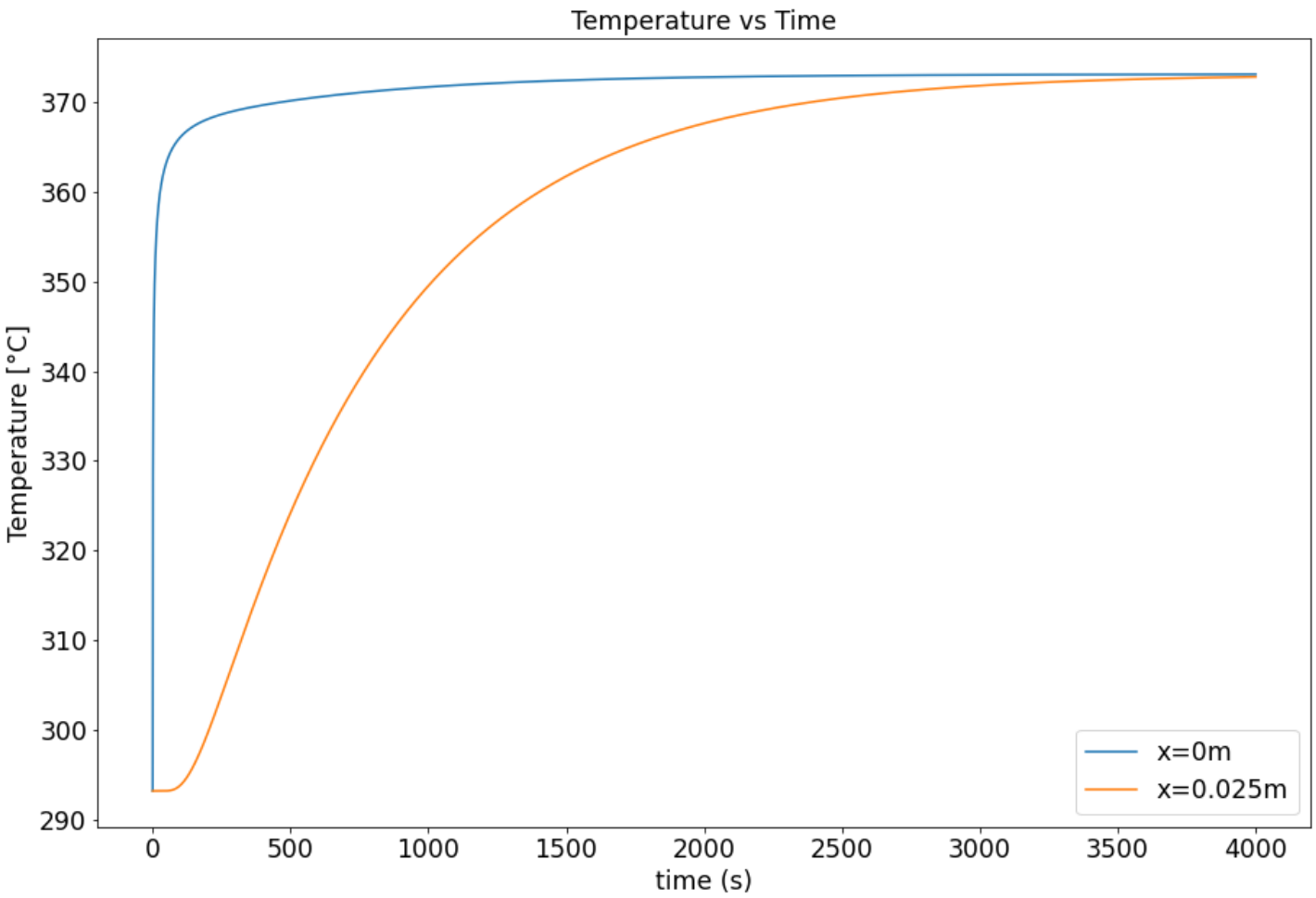
```
L=0.025
Ti=20+273.15
h0=1000
To=100+273.15
rho=2500
cp=1000
k=1
alpha=k/(rho*cp)
```

```
N=25
dx=L/(N-1)
dt_max=k*dx**2/(alpha*(2*h0*dx+2*k))
dt=dt_max/10
print('Neuman Stability', 'dt=' +str(dt)+'s','<',str(dt_max)+'s')
x=np.linspace(0,L,N)
tmax=4000
t=np.linspace(0,tmax, num=int(tmax/dt)+1)
Fo=alpha*dt/(dx**2)
T=np.zeros((size(x), size(t)))

T[:,0]=Ti
start=time.time()
for i1 in range(size(t)-1):
    for i2 in range(1,N-1):
        T[i2, i1+1]=(1-2*Fo)*T[i2,i1]+Fo*(T[i2-1,i1]+T[i2+1,i1])

    T[0,i1+1]=(1-(2*h0*dt)/(dx*rho*cp)-2*Fo)*T[0,i1]+2*Fo*T[1,i1]+(2*h0*dt)/(dx*rho*cp)*To

    T[N-1,i1+1]=(1-2*Fo)*T[N-1,i1]+2*Fo*T[N-2,i1]
end=time.time()
elapsed=end-start
print('Time elapsed', elapsed, 's for',str(N)+' nodes')
```

```
Neuman Stability dt=0.0664328231292517s < 0.664328231292517s
Time elapsed 2.0794436931610107 s for 25 nodes
```

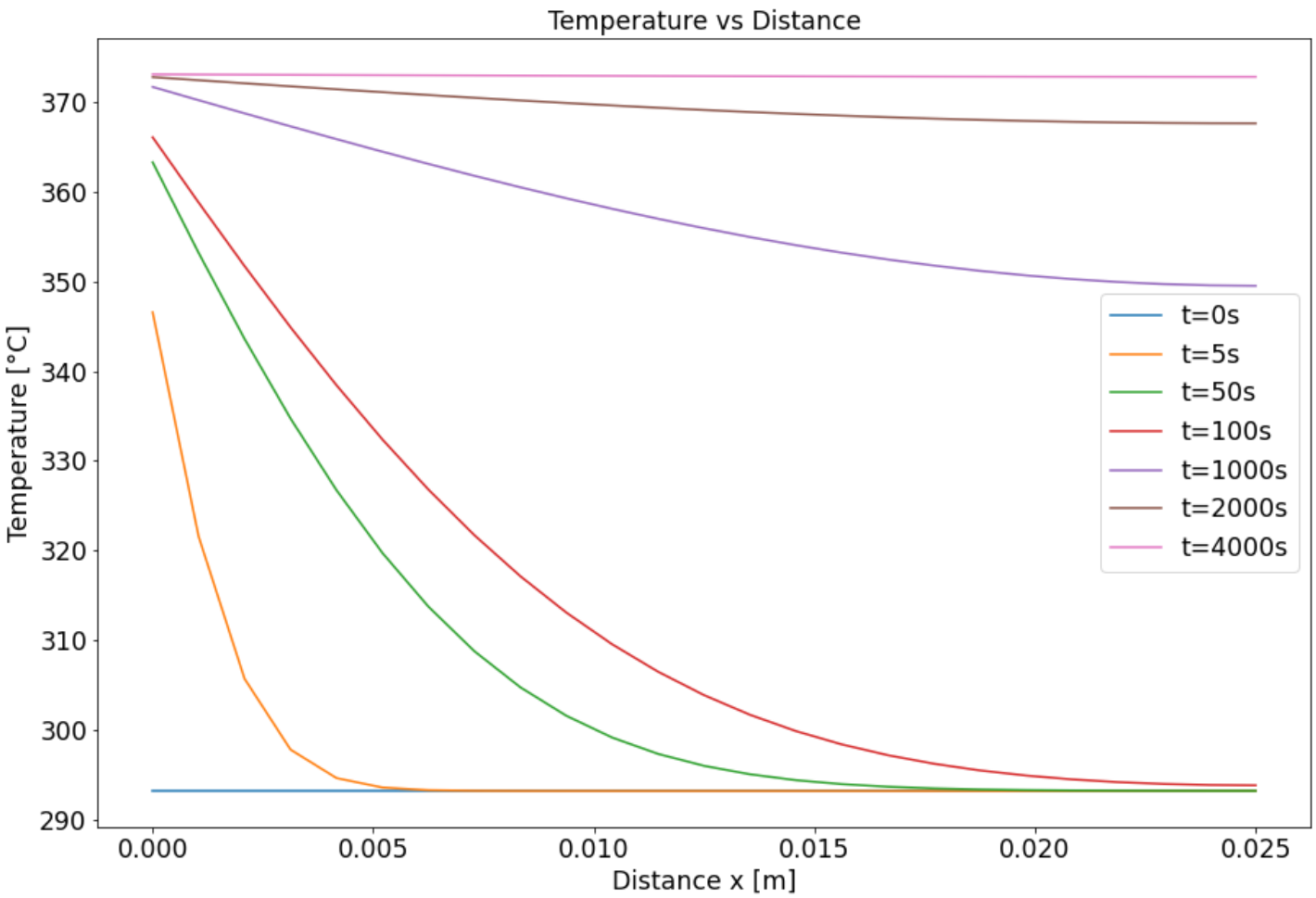
In [36]:

```
xp=[0,L]
label=str("x=" +str(xp[0])+'m')
plt.plot(t, T[0,:], label=label)
plt.xlabel('time (s)')
plt.ylabel('Temperature [°C]')
plt.title('Temperature vs Time')
label=str("x=" +str(xp[1])+'m')
plt.plot(t, T[-1,:], label=label)
plt.legend()
plt.show()
```



In [28]:

```
tp=[0,5,50,100,1000,2000,4000]
for t in tp:
    label=str("t=" +str(t)+'s')
    plt.plot(x, T[:, int(t/tmax*(int(tmax/dt)))], label=label)
    plt.xlabel('Distance x [m]')
    plt.ylabel('Temperature [°C]')
    plt.title('Temperature vs Distance')
plt.legend()
plt.show()
```


Temperature vs Distance
Temperature [°C]
Distance x [m]
370
360
350
340
330
320
310
300
290
0.000
0.005
0.010
0.015
0.020
0.025
t=0s
t=5s
t=50s
t=100s
t=1000s
t=2000s
t=4000s

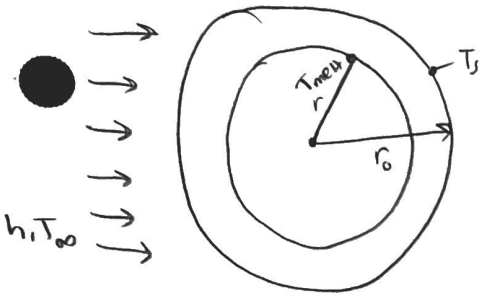
Problem 4

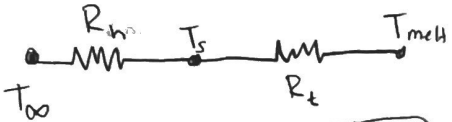
Problem 4

Assume

Stefan Number small

1-D





$$T_\infty - T_s = q_h R_h$$

$$q_h = h(T_\infty - T_s) A_s$$

$$A_s = 2\pi r_0 L$$

$$R'_h = \frac{1}{2\pi r_0 h}$$

$$T_s - T_{melt} = q R_t \rightarrow R_t = \frac{T_s - T_{melt}}{q}$$

$$q = -k_s \left(r \frac{\partial T}{\partial r}\right) A_s(r)\Big|_{r=r} = -k_s \frac{T_{melt} - T_s}{\ln\left(\frac{r_0}{r}\right)} \frac{1}{r} 2\pi r L$$

$$q' = \frac{2\pi k_s}{\ln\left(\frac{r_0}{r}\right)} (T_s - T_{melt})$$

$$R'_t = \frac{\ln(r_0/r)}{2\pi k_s}$$

$$\dot{E}_{in} - \dot{E}_{out} + \dot{E}_{gen} = \dot{E}_{st}$$

$$q'_h - q' = q'_{loss}$$

$$\frac{T_\infty - T_s}{R'_h} - \frac{T_s - T_{melt}}{R'_t} = q'_{loss}$$

→

$$\dot{E}_{in} - \dot{E}_{out} + \dot{E}_{gen} = \dot{E}_{st}$$

$$q_{in} - q_{out} = \left(\rho C_p \dot{\forall} (T_s - T_{melt}) - \dot{\forall} \rho \Delta H\right)\Big|_{r=r} \qquad \forall = 2\pi L (r_0 - r(t)) \qquad \dot{\forall} = 2\pi L\left(-\frac{ds}{dt}\right)$$

$$\left(-k_s \left(r \frac{\partial T}{\partial r}\right)\Big|_{r=r} - -k_\ell r \left(\frac{\partial T}{\partial r}\right)\Big|_{r=r}\right) 2\pi L = \left(\rho C_p (2\pi L)\left(\frac{ds}{dt}\right) = 2\pi L \left(\frac{ds}{dt}\right) \rho \Delta H\Big|_{r=r}\right.$$

$$-k_s \left(r \frac{\partial T}{\partial r}\right)\Big|_{r=r} + k_\ell \, r \left(\frac{\partial T}{\partial r}\right)\Big|_{r=r} = -\frac{ds}{dt} \rho \Delta H \qquad r(t) = r_0 - s(t)$$

Using Neumann solution where solve for solid domain

IC

$t < 0$

$s(0) = 0$

BC

$T(r = r_0, t \geq 0) \rightarrow T_s < T_{melt}$

$T(r = r_0 - s(t), t) = T_{melt}$

$$-k_s \left(r \frac{\partial T}{\partial r}\right)\Big|_{r=r_0 - s(t)} + k_\ell \left(r \frac{\partial T}{\partial r}\right)\Big|_{r=r} = -\frac{ds}{dt} \rho \Delta H$$

$$k_s \left(r \frac{\partial T}{\partial r}\right)\Big|_{r=r_0 - s(t)} = \rho \Delta H \frac{\partial s}{\partial t}$$

$$k_s (r_0 - s(t)) \left(\frac{T_{melt} - T_s}{r_0 - s(t) - r_0}\right) = \rho \Delta H \frac{\partial s}{\partial t}$$

$$k_s (r_0 - s(t)) \left(\frac{T_{melt} - T_s}{-s(t)}\right) = \rho \Delta H \frac{ds}{dt}$$

$$k_s (r_0 - s(t))(T_{melt} - T_s) = -s(t) \rho \Delta H \frac{ds}{dt}$$

$$T_{melt} - T_s = \frac{-s(t) \rho \Delta H \frac{ds}{dt}}{k_s (r_0 - s(t))} \rightarrow T_s = T_{melt} + \frac{s(t) \rho \Delta H}{k_s (r_0 + s(t))} \frac{ds}{dt}$$

Energy balance from surface

$$h(T_\infty - T_s) = -k_s \left(r \frac{dT}{dr}\right)\Big|_{r=r_0 - s(t)} = -k_s (r_0) \left(\frac{T_{melt} - T_s}{r_0 - s(t) - r_0}\right) = k_s r_0 \left(\frac{T_{melt} - T_s}{s(t)}\right)$$

$$h(T_\infty - T_{melt}) = \frac{-k_s r_0}{s(t)} \frac{s(t) \rho \Delta H}{k_s (r_0 - s(t))} \frac{ds}{dt} + \frac{s(t) \rho \Delta H h}{k_s (r_0 - s(t))} \frac{ds}{dt} = \frac{\rho \Delta H}{k_s (r_0 - s(t))} \left(h s(t) - k_s r_0\right) \frac{ds}{dt}$$

$$\int_0^t \frac{k_s h (T_\infty - T_{melt})}{\rho \Delta H} d\tau = \int_0^{s(t)} \frac{h u(t)}{r_0 - u(t)} - \frac{k_s r_0}{r_0 - u(t)} du$$

$$\frac{k_s h (T_\infty - T_{melt})}{\rho \Delta H} t = \int_0^{s(t)} \frac{h u(t)}{r_0 - u(t)} du - \int_0^{s(t)} \frac{k_s r_0}{r_0 - u(t)} du$$

Wolfram Alpha

$$\frac{k_s h (T_\infty - T_{melt})}{\rho \Delta H} t = (h r_0 + k_s r_0) \ln\left(\frac{r_0 - s(t)}{r_0}\right) - h s(t)$$

$$\frac{k_s h (T_\infty - T_{melt})}{\rho \Delta H} = r_0 (h + k_s) \ln\left(\frac{r(t)}{r_0}\right) + h r(t) - h r_0$$

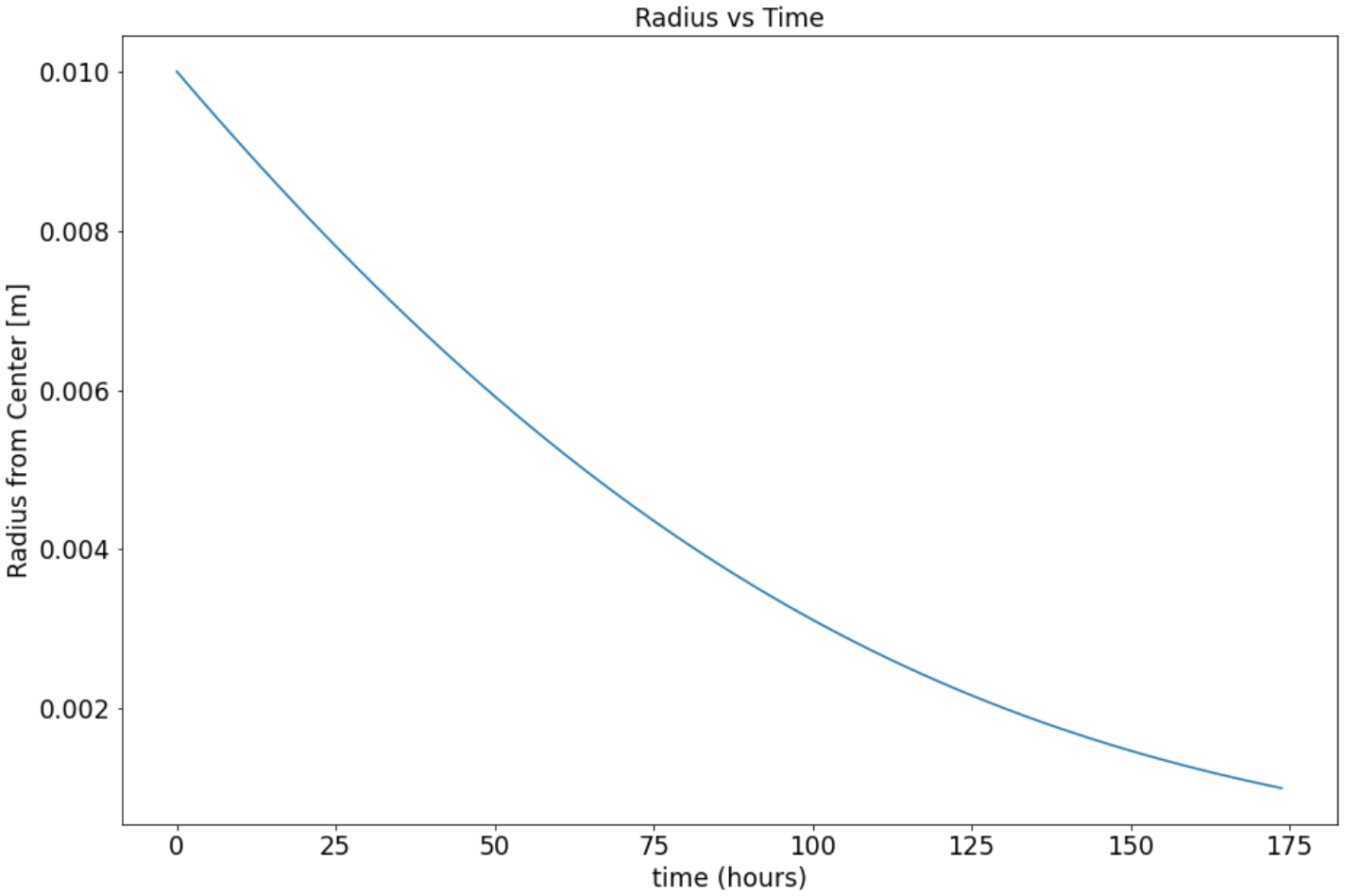
$$\frac{k_s h (T_\infty - T_{melt}) t}{\rho \Delta H} = r_0 (h + k_s) \ln\left(\frac{r_0/2}{r_0}\right) + h\left(\frac{r_0}{2}\right) - h r_0$$

$$t = \left(\frac{\rho \Delta H}{k_s h (T_\infty - T_{melt})}\right)\left(r_0 (h + k_s) \ln(1/2) + h\left(r_0/2\right)\right)$$

In [29]:

```
h=25
Tinf=-5+273.15
r0=0.01
Tmelt=0+273.15
rhos=520
ks=1.88
```

```
cps=2040
hsl=335*10**3
def t(r):
    return((rhos*hsl/(ks*h*(Tinf-Tmelt)))*(r0*(h+ks)*np.log(r/r0)+h*r-h*r0)/3600)
r=np.linspace(r0*0.1,r0,100)
time=t(r)
plt.plot(time, r)
plt.xlabel('time (hours)')
plt.ylabel('Radius from Center [m]')
plt.title('Radius vs Time')
plt.show()
```



In [ ]: