**Assn5\_Dadlani\_22101079**

Name : Yash Dadlani

Roll No: 22101079

Course : Computational Fluid Dynamics and Heat Transfer

Course Code: ME 630A

Instructor: Prof. Subrata Sarkar

Branch : Aerospace Engineering

Spec: MTech Aerodynamics

Assignment: #5, Convection from Fin Tip

Fin Dimensions: Length(l) = 0.1 m, Width(w) = 0.5 m, Thickness(t) = 0.002 m, Ac = 0.5\*0.002=0.001m2 ; Perimeter(p) = 2\*(w + t) = 1.004m; K = Thermal Conductivity

Aluminum: 237 W/m-K , Stainless Steel: 17 W/m-K , Glass: 0.8 W/m-K

Physical Conditions: Temp Tbase = 250° C (constant) , T surr = 25° C (constant),

Convection Heat transfer Coeff. from fin lateral surface, h = 25W/m2K, Convection Heat transfer Coeff. from fin tip, he = 10W/m2K

Boundary Conditions: T (x=0) = 250° C = Tbase (1)

he\*Ac\*(T(x=l)- T surr) = -K\*Ac\*( (dT/dx)| at x = l ) (2)

**Assumptions**

Steady State

**Governing equation**: T’’ – m2 T = 0 , T = Temp. at any point on the fin , m2 = (h\*p) / (K\*Ac)

Non – Dimensional Form : Dimensionless Temp. = (T – T surr) / ( Tbase - T surr) = Ɵ, We get

Ɵ’’ – m2 Ɵ = 0 (3)

From Boundary Conditions, (1) => Ɵ (x = 0) = 0 (4)

(2) => Ɵ’ (x=l) = -(he/k) \* Ɵ (x=l) (5)

**Computational Domain and Spatial Discretization**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |

x = 0 i-1 i i+1 x =0.1

Consider uniformly spaced grid points , dividing length of fin 0.1m into 10 intervals, dx = 0.01

In (3), We select central difference in space. So for any grid point i , we have

(**Ɵi+1 – 2Ɵi + Ɵi-1**) / (dx2) = m2 **Ɵi** : Stencil for interior grid points from i = 1 to i = n-1 (i=9) (for our case, n =10) (6)

Now For i = 1 to i = 9

For i = 1: Ɵ**1** \*(-2 - m2 \* dx2) + Ɵ**2** = - Ɵ**0**

Similarly for other grid points, say i = 2, we get Ɵ**1** + (-2 - m2 \* dx2)\*Ɵ**2** + Ɵ**3** = 0

Let -2 - m2 \* dx2 = a

From (5) we have for i = 10, applying second order backward difference for i=10, dƟ/dx | x=l = -(he/k) \* Ɵ (x=l)

(3Ɵ10 - 4Ɵ9 + Ɵ8)/(2\*dx) = -(he/k)\*Ɵ10

(3 +(2\*dx\*(he/k)))Ɵ10 - 4Ɵ9 + Ɵ8 = 0

Let 3 +(2\*dx\*(he/k)) = b, so for last grid point we have, Ɵ8 - 4Ɵ9 + bƟ10 = 0. We get the following Finite difference equations :

aƟ1 + Ɵ2 = Ɵ0

Ɵ1 + aƟ2 + Ɵ3 = 0

Ɵ2 + aƟ3 + Ɵ4  = 0

Ɵ3 + aƟ4 + Ɵ5 = 0

Ɵ4 + aƟ5 + Ɵ6 = 0

Ɵ5 + aƟ6 + Ɵ7 = 0

Ɵ6 + aƟ7 + Ɵ8 = 0

Ɵ7 + aƟ8 + Ɵ9 = 0

Ɵ8 + aƟ9 + Ɵ10 = 0

Ɵ8 - 4Ɵ9 + bƟ10 = 0

The set of equations will not form TDM. So we carry out Row Operation R10 -R9, We get

aƟ1 + Ɵ2 = -Ɵ0

Ɵ1 + aƟ2 + Ɵ3 = 0

Ɵ2 + aƟ3 + Ɵ4  = 0

Ɵ3 + aƟ4 + Ɵ5 = 0

Ɵ4 + aƟ5 + Ɵ6 = 0

Ɵ5 + aƟ6 + Ɵ7 = 0

Ɵ6 + aƟ7 + Ɵ8 = 0

Ɵ7 + aƟ8 + Ɵ9 = 0

Ɵ8 + a Ɵ9 + Ɵ10 = 0

(-4-a) Ɵ9 + (b-1) Ɵ10 = 0

Now these set of equations are tridiagonal. We form a matrix .

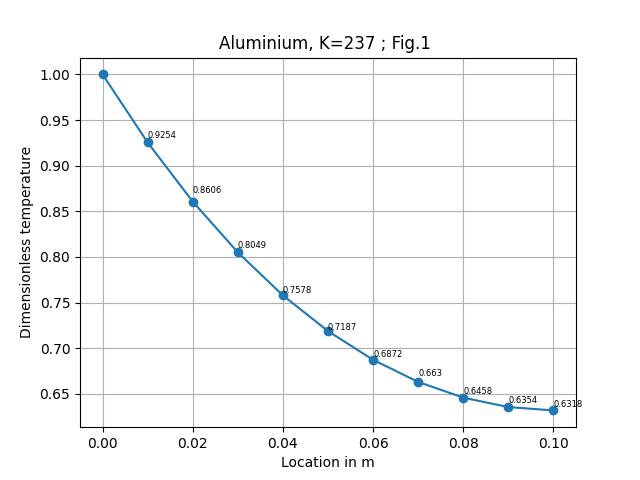
=

Let α be a row vector that constitutes lower diagonal values,

Let β be row vector that constitutes main diagonal values,

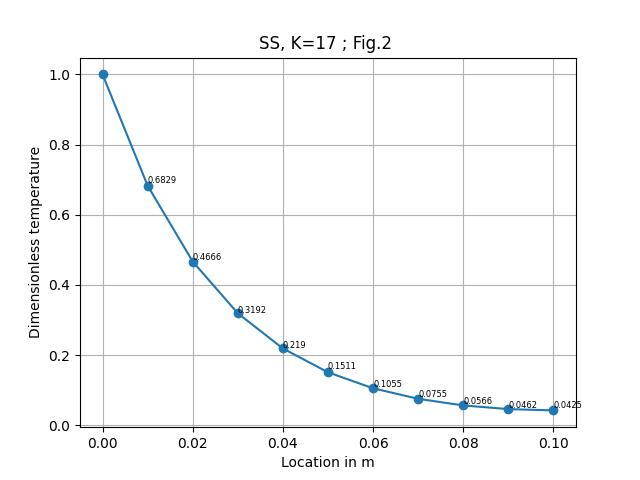
Let δ be row vector that constitutes upper diagonal values,

**Part (i) Results:**

****

|  |  |  |  |
| --- | --- | --- | --- |
| **Position** | **NDT (N)** | **NDT (exact)** | **DT (in °C) (N)** |
| 0.0 | 1.0000000 | 1.00000000 | 250.0 |
| 0.01 | 0.92542045 | 0.92538305 | 233.21960187 |
| 0.02 | 0.86064177 | 0.86057522 | 218.64439869 |
| 0.03 | 0.8049779 | 0.80488955 | 206.12002858 |
| 0.04 | 0.75783933 | 0.75773576 | 195.5138495 |
| 0.05 | 0.71872682 | 0.71861402 | 186.71353439 |
| 0.06 | 0.68722614 | 0.68710964 | 179.62588161 |
| 0.07 | 0.66300368 | 0.66288867 | 174.17582783 |
| 0.08 | 0.6458029 | 0.64569436 | 170.30565306 |
| 0.09 | 0.63544164 | 0.63534446 | 167.97436939 |
| 0.1 | 0.63181016 | 0.63172925 | 167.15728685 |

**Table.1 Temp distribution(for Aluminum), NDT = non dimensional temp, (N) = Numerical**

****

|  |  |  |  |
| --- | --- | --- | --- |
| **Position** | **NDT (N)** | **NDT (exact)** | **DT (in °C) (N)** |
| 0.0 | 1.0 | 1.0 | 250.0 |
| 0.01 | 0.68290128 | 0.681313 | 178.65278904 |
| 0.02 | 0.46663094 | 0.46446368 | 129.99196045 |
| 0.03 | 0.31925727 | 0.31703897 | 96.83288603 |
| 0.04 | 0.219021 | 0.21700292 | 74.27972596 |
| 0.05 | 0.15112254 | 0.14940288 | 59.00257249 |
| 0.06 | 0.10553688 | 0.10413448 | 48.74579884 |
| 0.07 | 0.07553343 | 0.07443133 | 41.99502254 |
| 0.08 | 0.05668227 | 0.05585364 | 37.75351134 |
| 0.09 | 0.04620008 | 0.04562454 | 35.39501858 |
| 0.1 | 0.0425392 | 0.04221506 | 34.57131974 |

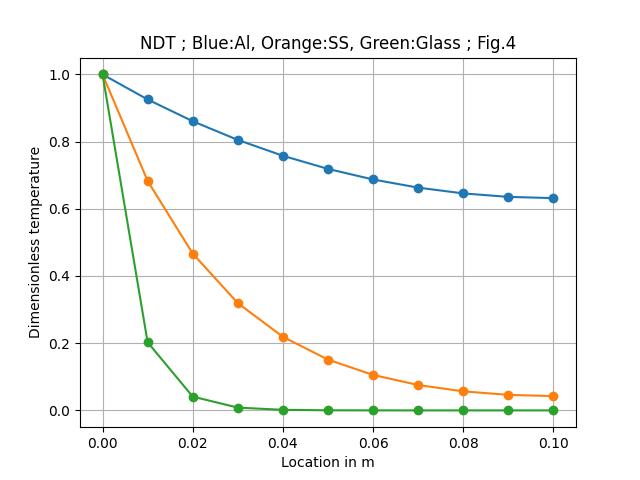
**Table.2 Temp distribution(for SS), NDT = non dimensional temp, (N) = Numerical**

**Chart, line chart

Description automatically generated**

|  |  |  |  |
| --- | --- | --- | --- |
| **Position** | **NDT (N)** | **NDT (exact)** | **DT (in °C) (N)** |
| 0.0 | 1.0 | 1.0 | 250.0 |
| 0.01 | 2.02639994e-01 | 1.70111877e-01 | 70.59399855 |
| 0.02 | 4.10629670e-02 | 2.89380507e-02 | 34.23916757 |
| 0.03 | 8.32099936e-03 | 4.92270612e-03 | 26.87222486 |
| 0.04 | 1.68616723e-03 | 8.37410779e-04 | 25.37938763 |
| 0.05 | 3.41684796e-04 | 1.42453522e-04 | 25.07687908 |
| 0.06 | 1.40275171e-05 | 2.42330503e-05 | 25.01557864 |
| 0.07 | 6.92384064e-05 | 4.12241389e-06 | 25.00315619 |
| 0.08 | 2.82796250e-06 | 7.01766632e-07 | 25.00063629 |
| 0.09 | 5.01140291e-07 | 1.22289086e-07 | 25.00011276 |
| 0.1 | -2.53354258e-07 | 3.79106667e-08 | 24.999943 |

**Table.3 Temp distribution(for Glass), NDT = non dimensional temp, (N) = Numerical**



**Part (ii) Results:**

**Ɵexact = ((cosh (m(l-x)) + (he/mK)\*sinh m(l-x))/ (cosh ml + (he/mK)\*sinh ml) = NDT (exact)**

**Chart, line chart

Description automatically generated**

**Chart, line chart

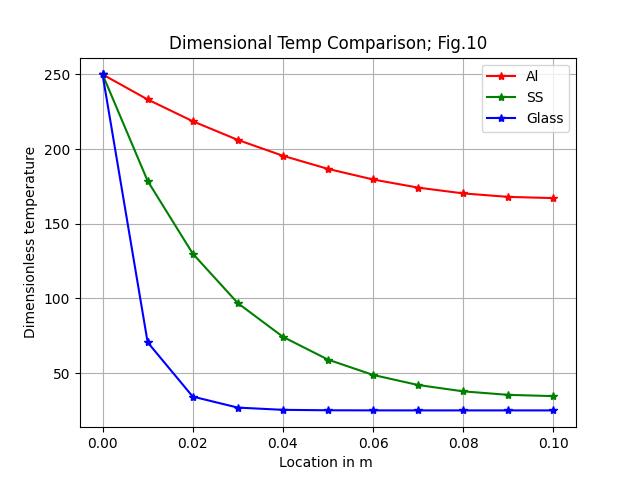
Description automatically generated**

**Chart, line chart

Description automatically generated**The zoomed up view for Stainless steel shows that there is very small error between numerical exact solution.

Chart, line chart

Description automatically generated



**Heat Loss Computation:**

Q = -K\*Ac\*(dT/dx | x=0) = mK\*Ac\*(Tb - Tsurr)((he/mK + tanh ml)/(1+(he/mK)\*tanh ml)

Q\_exact = mK\*Ac\*(Tb - Tsurr)((he/mK + tanh ml)/(1+(he/mK)\*tanh ml)

Q\_numerical1 = -K\*Ac\*((T1- T0)/(dx)) , applying first order forward

Q\_numerical2 = -K\*Ac\*((-T2+4 T1 -3 T0) / (2\*dx)) , applying first order forward

|  |  |  |  |
| --- | --- | --- | --- |
| Material | Q\_exact | Q\_numerical1 | Q\_numerical2 |
| Aluminum | 425.40602701091353 | 397.69543560314344 | 423.8269956382408 |
| Stainless Steel | 146.84413473725814 | 121.29025863745197 | 140.57368366156942 |
| Glass | 31.88338125105301 | 14.352480115636514 | 20.074526934231734 |

**Table. 4 Heat Loss Comparison**

**T vs X output code for Temp distribution and plotting (in python)**

import numpy as np

import matplotlib.pyplot as plt

from math import \*

K = float(input("Enter a value of thermal conductivity : "))

Tb = 250 ; Ts = 25 ; h = 25 ; ht = 10

l = 0.1 ; w = 0.5 ; t = 0.002 ; Ac = w\*t ; p = 2\*(w+t);

m = sqrt(h\*p/(K\*Ac))

hnd = ht/(m\*K)

dx = float(input("Enter a value of step size : "))

n = int(l/dx)

x = np.linspace(0,l,n+1)

theta = np.empty(n+1,float)

theta[0] = 1

a = -2 - ((m\*dx)\*\*2)

b = 3 + (ht/K)\*(2\*dx)

alpha = [0]\*n

alpha[0] = 0

alpha[n-1] = -4-(1\*a)

for i in range(1,n-1):

     alpha[i]= 1

beta = [0]\*n

beta[n-1] = b-1

for i in range(0,n-1):

     beta[i]= a

delta = [1]\*n

delta[n-1]=0

sol = [0]\*n

sol[0] = -1

e = [0]\*n

f = [0]\*n

th = [0]\*n

e[0] = delta[0]/beta[0]

f[0] = sol[0]/beta[0]

for i in range(1, n-1):

     d = (beta[i] - (alpha[i]\*e[i-1]))

     e[i] = delta[i]/d

     f[i] = (sol[i] - alpha[i]\*f[i-1])/d

th[n-1] = (sol[n-1] - (alpha[n-1]\*f[n-2])) / (beta[n-1] - (alpha[n-1]\*e[n-2]))

for j in range(n-2,-1,-1):

     th[j] = f[j] - e[j]\*th[j+1]

 # print('th = ' , [round(i, 5) for i in th])

for i in range(1,n+1):

     theta[i] = th[i-1]

T = np.empty(n+1,float)

print("x\t\t T(numerical temperature)")

for i in range(0,n+1):

    T[i]= theta[i]\*(Tb-Ts) + Ts

    print("%f \t %f" %(x[i],T[i]))

q\_fof = -1\*K\*Ac\*((T[1]-T[0])/(dx))

q\_sof = -1\*K\*Ac\*((4\*T[1]-3\*T[0]-T[2])/(2\*dx))

if K==237:

 print("Numerical Heat Loss for Al from fof scheme : " + str(q\_fof))

 print("Numerical Heat Loss for Al from sof scheme : " + str(q\_sof))

 plt.plot(x,T,marker='o')

 plt.xlabel('Location in m')

 plt.ylabel('Dimensional temperature')

 plt.grid()

 plt.title("Temp. Distribution for Aluminum")

 plt.show()

if K==17:

 print("Numerical Heat Loss for SS from fof scheme : " + str(q\_fof))

 print("Numerical Heat Loss for SS from sof scheme : " + str(q\_sof))

 plt.plot(x,T,marker='o')

 plt.xlabel('Location in m')

 plt.ylabel('Dimensional temperature')

 plt.grid()

 plt.title("Temp. Distribution for SS")

 plt.show()

if K==0.8:

 print("Numerical Heat Loss for Glass from fof scheme : " + str(q\_fof))

 print("Numerical Heat Loss for Glass from sof scheme : " + str(q\_sof))

 plt.plot(x,T,marker='o')

 plt.xlabel('Location in m')

 plt.ylabel('Dimensional temperature')

 plt.grid()

 plt.title("Temp. Distribution for Glass")

 plt.show()

**fof: First Order Forward**

**sof: Second Order Forward**

**Output: for dx = 0.01**

**For eg: K = 237**

**x T(numerical temperature)**

**0.000000 250.000000**

**0.010000 233.219602**

**0.020000 218.644399**

**0.030000 206.120029**

**0.040000 195.513849**

**0.050000 186.713534**

**0.060000 179.625882**

**0.070000 174.175828**

**0.080000 170.305653**

**0.090000 167.974369**

**0.100000 167.157287**

**Numerical Heat Loss for Al from fof scheme : 397.69543560314344**

**Numerical Heat Loss for Al from sof scheme : 423.8269956382408**

**Grid Independence Test :**

**For Glass : K= 0.8**

dx = 0.02, grid points = 6

Chart, line chart

Description automatically generated

dx = 0.01, grid points = 11

Chart, line chart

Description automatically generated

for dx = 0.005, grid points = 21

Chart, line chart

Description automatically generated

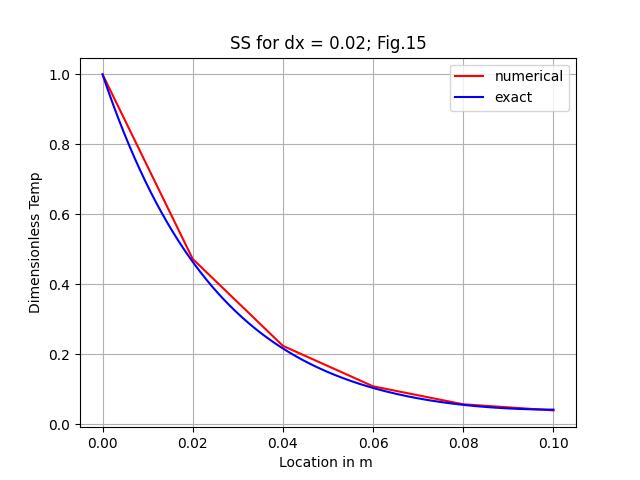
for dx = 0.0025, grid points = 41

Chart, line chart

Description automatically generated

**We can observe that for dx =0.005 and for dx = 0.0025, if a mesh size is chosen between these two values, the grid will be independent of mesh size and further mesh refinement less than dx = 0.0025 is not needed.**

**For Stainless Steel : K= 17**

** grid points = 6**

**Chart, line chart

Description automatically generated grid points = 11**

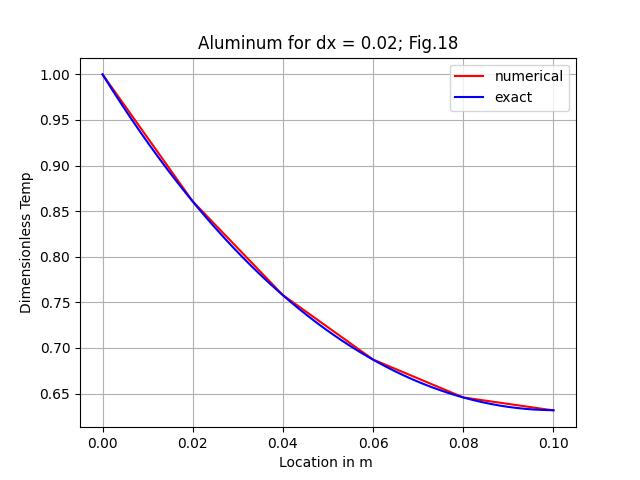
**Chart, line chart

Description automatically generated grid points = 21**

**We can observe that for dx =0.005 , if a mesh size is chosen lesser, the grid will be independent of mesh size and further mesh refinement less than dx = 0.005 is not needed.**

**For Aluminum , K = 237**

For dx = 0.02 , grid points = 6



For dx = 0.01 , grid points = 11Chart, line chart

Description automatically generated

**We can observe that for dx =0.01 , if a mesh size is chosen lesser, the grid will be independent of mesh size and further mesh refinement less than dx = 0.01 is not needed.**