#===================================

# ADE\_Baraket\_Clark

#===================================

import math

import numpy as np

import matplotlib.pyplot as plt

from mpl\_toolkits.mplot3d import Axes3D

m=40 # number of horizontal divisions

n=40 # number of vertical divisions

l=1. # length of the domain

h=1. # height of the domain

dx=l/m

dy=h/n

dt=5 # time step

alpha=23.1\*1e-6 # diffusivity of the iron

cr1=1e-3 # convergence criteria for steady state condition

T1=60 # left hand side temperature

T2=20 # right hand side temperature

T3=10 # bottom side temperature

T4=30 # top side temperature

Fo\_x=alpha\*dt/dx\*\*2 # Fourier number

Fo\_y=alpha\*dt/dy\*\*2 # Fourier number

# initial condition

T=np.zeros((m+1,n+1));T\_old=np.zeros((m+1,n+1));p=np.zeros((m+1,n+1));q=np.zeros((m+1,n+1))

# boundary conditions

for i in range(m+1):

T[i,0]=T3

T[i,n]=T4

for j in range(n+1):

T[0,j]=T1

T[m,j]=T2

#============================== main loop ==============================

t=0

err1=1

while err1>cr1:

t=t+dt

T\_old[:,:]=T[:,:]

p[:,:]=T[:,:]

q[:,:]=T[:,:]

# step 1

for i in range(1,m):

for j in range(1,n):

p[i,j]=( p[i,j]+Fo\_x\*(p[i-1,j]-p[i,j]+p[i+1,j]) +Fo\_y\*(p[i,j-1]-p[i,j]+p[i,j+1]) )/(1+Fo\_x+Fo\_y)

# step 2

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

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

q[i,j]=( q[i,j]+Fo\_x\*(q[i-1,j]-q[i,j]+q[i+1,j]) +Fo\_y\*(q[i,j-1]-q[i,j]+q[i,j+1]) )/(1+Fo\_x+Fo\_y)

err1=0

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

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

T[i,j]=0.5\*(p[i,j]+q[i,j])

err1=err1+(T[i,j]-T\_old[i,j])\*\*2/((m-1)\*(n-1))

err1=math.sqrt(err1)

# printing time and second norm of the error

print('{:10.2f}{:15.2e}'.format(t,err1))

#=============================== results ===============================

print('\n=================================================================\n')

print(' calculation completed\n')

print(' mesh:{:8d}\*{:2d}'.format(m,n))

print(' x Fourier number:{:8.3f}'.format(Fo\_x))

print(' y Fourier number:{:8.3f}'.format(Fo\_y))

print(' time until reaching steady state condition:{:10.2f}s\n'.format(t))

# generating the grid

Y,X=np.meshgrid(np.linspace(0,h,n+1),np.linspace(0,l,m+1))

# temperature contour

fig1=plt.figure(1)

plt.contourf(X,Y,T,50,cmap='jet')

plt.axes().set\_aspect('equal')

plt.xlabel('x(m)');plt.ylabel('y(m)')

# temperature surface

fig2=plt.figure(2)

ax=plt.axes(projection='3d')

ax.plot\_surface(X,Y,T,cmap='jet')

ax.set\_xlabel('x(m)');ax.set\_ylabel('y(m)');ax.set\_zlabel('T($^oC$)')

plt.show()