1500011370_Poisson

December 31, 2018

In [12]:
```
import numpy as np
import matplotlib.pyplot as plt
from math import *
```

In [28]:
```
# Set maximum iteration
maxIter = 100
TOL = 1e-5

# Set Dimension and delta
lenX = 6
lenY = 5 #we set it rectangular
delta = 1

# Boundary condition
Ttop = 100
Tright = 0
def Tbottom(x):
    if x<=6 and x>=0:
        return x(6-x)
    else:
        return 0

def Tleft(y):
    if y<=5 and y>=0:
        return y(5-y)
    else:
        return 0

# heat
q = 1.5
K = 1.04
h = 0.4
k = 1/3
n = int(lenX/h)  # n = 15
m = int(lenY/k) # m = 15
```

In [29]:
```
x = np.arange(n)
y = np.arange(m)
```

```
        w = np.zeros((n, m))
        for i in range(1, n):
            x[i] = i * h
            y[i] = i * k
```

In [30]:
```
        _lambda = h* h /k* k
        mu = 2*(1+k)
        l = 1
```

In [27]:
```
        while(l<=maxIter):

            z = (-pow(h, 2)*(-q/K)+y[m-1]*(5-y[m-1])+
                 _lambda * w[1][m-2]+w[2][m-1])/mu
            NORM = abs(z-w[1][m-1])
            w[1][m-1] = z

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

                z = (-pow(h, 2)*(-q/K)+ 0 +w[i-1][m-1]
                     +w[i+1][m-1] +_lambda*w[i][m-2])/mu
                if abs(w[i][m-1]-z)>NORM:
                    NORM = abs(w[i][m-1]-z)
                w[i][m-1] = z

            z = (-pow(h, 2)*(-q/K)+ 0 + 0
                 +w[n-2][m-1]+w[n-1][m-2] )/mu

            if abs(w[n-1][m-1] - z) >NORM:
                NORM = abs(w[n-1][m-1] - z)
            w[n-1][m-1] = z

            for j in range(m-2, 1, -1): # from m-2, .... , 2

                z = (-pow(h, 2)*(-q/K)+ y[j]*(5-y[j]) +
                     _lambda*w[1][j+1] + _lambda*w[1][j-1]+w[2][j] )/mu
                if abs(w[1][j] - z) > NORM:
                    NORM = abs(w[1][j] - z)
                w[1][j] = z

                for i in range(2, n-1):
                    z = (-pow(h, 2)*(-q/K)+ w[i-1][j]
                         +_lambda*w[i][j+1] + w[i+1][j]+ _lambda*w[1][j-1] )/mu
                    if abs(w[i][j] - z) > NORM :
                        NORM = abs(w[i][j] - z)
                    w[i][j] = z

                z = (-pow(h, 2)*(-q/K)+ 0 +w[n-2][j+1]
```

```
                    + _lambda*w[n-1][j+1]+ _lambda*w[n-1][j-1] )/mu
                if abs(w[n-1][j] - z) > NORM:
                    NORM =  abs(w[n-1][j] - z)
                w[n-1][j] = z

            z = (-pow(h, 2)*(-q/K)+ y[1]*(5-y[1])
                 + _lambda*x[1]*(5-x[1]) + _lambda*w[1][2] + w[2][1])/mu
            if abs(w[1][1] - z) > NORM:
                NORM = abs(w[1][1] - z)
            w[1][1] = z

            for i in range(2, n-1):
                z = (-pow(h, 2)*(-q/K)+ _lambda*x[i]*(5-x[i])
                     + w[i-1][1] + _lambda*w[i][2] + w[i+1][1])/mu
                if abs(w[i][1] - z) > NORM:
                    NORM = abs(w[i][1] - z)
                w[i][1] = z

            z = (-pow(h, 2)*(-q/K)+ 0 + _lambda*x[n-1]*(6-x[n-1])
                 + w[n-2][1] + _lambda*w[n-1][2])/mu
            if abs(w[n-1][1] - z) > NORM:
                NORM = abs(w[i-1][1] - z)
            w[n-1][1] = z

            if NORM <= TOL:

                for i in range(1, n):
                    for j in range(1, m):
                        print(x[i], y[j], w[i][j])

                break

            l = l+1
```

```
0 0 0.3811346339371776
0 0 0.4873991662462554
0 1 2.3501207696656237
0 1 2.6115893277227316
0 1 2.734460684984241
0 2 3.674381451057718
0 2 3.8008478087250492
0 2 3.8175590707180502
0 3 3.8181205397425515
0 3 3.807094994812158
0 3 3.716044431758118
0 4 2.7712797840628323
0 4 2.596137058836237
0 4 2.2821936577936843
```

```
0 0 0.7077551861547948
0 0 0.6322228180320693
0 1 1.5407313314897262
0 1 1.9203731525508625
0 1 2.0558877822165775
0 2 2.5225143657035254
0 2 2.706745518698706
0 2 2.7310125057107695
0 3 2.731605242342983
0 3 2.7166637595366048
0 3 2.6267438650160666
0 4 2.149889761889877
0 4 1.884145471313638
0 4 1.4400460964534967
1 0 1.1744711193143835
1 0 0.6605769537808189
1 1 1.142903236718828
1 1 1.5741978276742863
1 1 1.6962978066949792
1 2 1.951654455880046
1 2 2.1622575100564783
1 2 2.189960750135348
1 3 2.1907014143565724
1 3 2.176157034751235
1 3 2.1054273888842916
1 4 1.8355817910642558
1 4 1.5241863079311018
1 4 1.0261397057720798
1 0 1.447905854030328
1 0 0.6550438193406494
1 1 0.94683961872772
1 1 1.399854882049135
1 1 1.5073490617849692
1 2 1.668398831501582
1 2 1.8910156439769694
1 2 1.920304472520736
1 3 1.9213442206539328
1 3 1.9087015405863093
1 3 1.854889226896415
1 4 1.6764540066460383
1 4 1.3425492585196313
1 4 0.8221793030078859
2 0 1.711237220416363
2 0 0.6433122161982506
2 1 0.8497348961469537
2 1 1.3113572307134753
2 1 1.4084080749027912
2 2 1.5273216099637033
```

2 2 1.7553500568387426
2 2 1.7853833609125114
2 3 1.7867681294472795
2 3 1.7760919017981658
2 3 1.7335756631691308
2 4 1.5954937872852588
2 4 1.250786503530057
2 4 0.7212688057195532
2 0 1.821889298855372
2 0 0.6330909207247506
2 1 0.8010028537242849
2 1 1.2655248331114064
2 1 1.3560726311926494
2 2 1.4560623331750462
2 2 1.6863876907782829
2 2 1.7167650184549568
2 3 1.7184561175384834
2 3 1.709315304857858
2 3 1.673549871239177
2 4 1.5533978790041445
2 4 1.203859018271084
2 4 0.6708078122519631
2 0 1.8552510847597572
2 0 0.625345355040458
2 1 0.7754662027539213
2 1 1.2401484107550267
2 1 1.3268215873757352
2 2 1.4181130449379449
2 2 1.6490943291280826
2 2 1.6795945320452372
2 3 1.6815121021664854
2 3 1.6734208831710504
2 3 1.641496463030132
2 4 1.5295825667124237
2 4 1.1785316987707726
2 4 0.6446372363781754
3 0 1.8347834826900964
3 0 0.6189722491714172
3 1 0.7601286320759343
3 1 1.2229508136661318
3 1 1.307180078884681
3 2 1.3940902295019384
3 2 1.624500603033163
3 2 1.6549466579990946
3 3 1.65699431935129
3 3 1.649554551487705
3 3 1.6198089545230197
3 4 1.5121823985232012

```
3 4 1.1620719874555863
3 4 0.6293222152790174
3 0 1.7478381828555625
3 0 0.6121455621348955
3 1 0.7474773602887302
3 1 1.2055611095904404
3 1 1.2878408164852004
3 2 1.3718509249952682
3 2 1.6000872307027934
3 2 1.6302430755008086
3 3 1.632307355946509
3 3 1.6252095475373822
3 3 1.5967529894486414
3 4 1.492179034210105
3 4 1.1459130980163927
3 4 0.6172364190269497
4 0 1.5375190915932695
4 0 0.6023027271827252
4 1 0.7318156208242206
4 1 1.1794250343584998
4 1 1.259404151195835
4 2 1.3404025733680176
4 2 1.5634533972440745
4 2 1.5928840276844956
4 3 1.5948079998133113
4 3 1.5877674970683628
4 3 1.5600874050918057
4 4 1.458760456142005
4 4 1.121164094890083
4 4 0.602851585381341
4 0 1.3851628669465679
4 0 0.5853489651638107
4 1 0.7068345482244157
4 1 1.1316072122586296
4 1 1.2078715997575158
4 2 1.2845598313105782
4 2 1.4960664294366242
4 2 1.5238721594136788
4 3 1.5253959050549706
4 3 1.5180450178518197
4 3 1.4906333715237612
4 4 1.3935353181197916
4 4 1.073567280621303
4 4 0.5804773234632014
4 0 1.191886276743342
4 0 0.5539377908361237
4 1 0.6634774874237643
4 1 1.0384078044388407
```

```
4 1 1.1077435912971494
4 2 1.1777864641141933
4 2 1.3639462871186152
4 2 1.3882035042193437
4 3 1.3888489713480903
4 3 1.3805727869914532
4 3 1.3524303042556856
4 4 1.260552317960623
4 4 0.9749027133392175
4 4 0.5427515183254769
5 0 0.8338404566903422
5 0 0.4940231819845685
5 1 0.5876943958671704
5 1 0.8536425246849618
5 1 0.9092653961369347
5 2 0.9699409249439533
5 2 1.100612185753549
5 2 1.1171661732568563
5 3 1.1160159353378436
5 3 1.1057017318067512
5 3 1.0745129392354074
5 4 0.9868462873883771
5 4 0.7653508910229487
5 4 0.48025062640925437
5 0 0.7218932882319861
5 0 0.3777442572441023
5 1 0.45846461971338837
5 1 0.48581351770226566
5 1 0.513279412850431
5 2 0.5644847539225153
5 2 0.5737548664941671
5 2 0.573595289260867
5 3 0.5688578114928393
5 3 0.5545230316569106
5 3 0.5152737889936676
5 4 0.42336390879476976
5 4 0.31511742373037216
5 4 0.38478260589717866
```