

Homework 3

Number 2

After applying the Boundary Element Method on the circular cylinder, the following plots are made for different values of n for the ψ_1 and $\frac{\partial \psi_1}{\partial n}$. It can be seen that even when changing the value of n , the trend of the lines does not change and looks sinusoidal in nature.

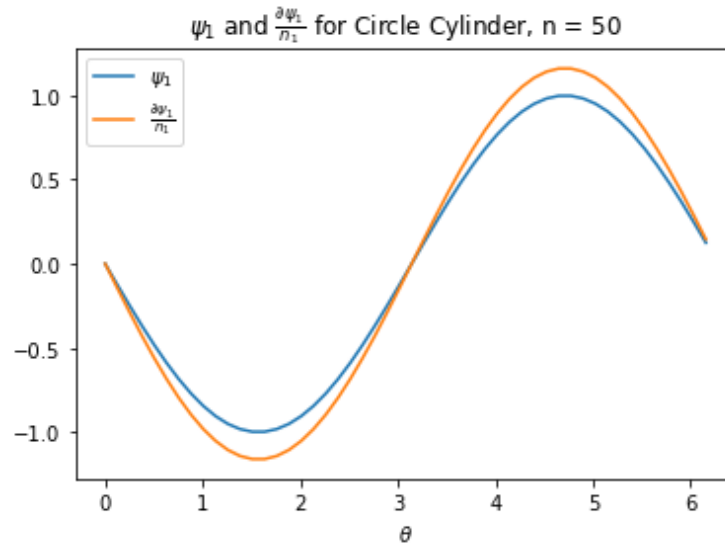


Figure 1: Plots of ψ_1 and $\frac{\partial \psi_1}{\partial n}$ for $n=50$

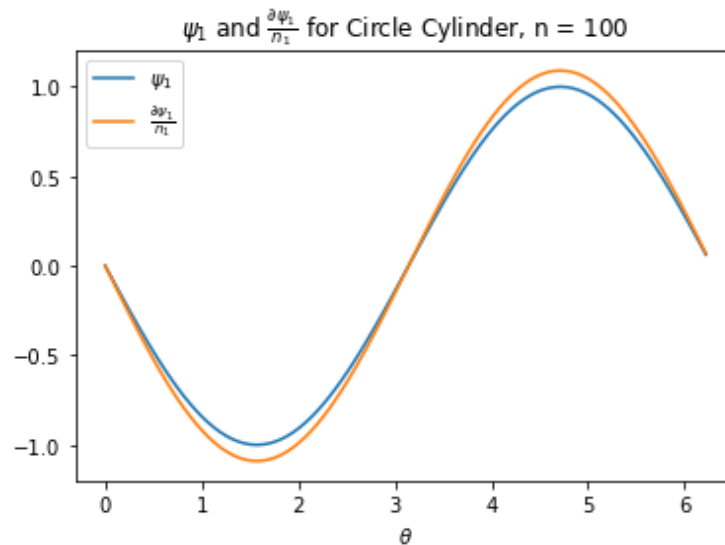


Figure 2: Plots of ψ_1 and $\frac{\partial \psi_1}{\partial n}$ for $n=100$

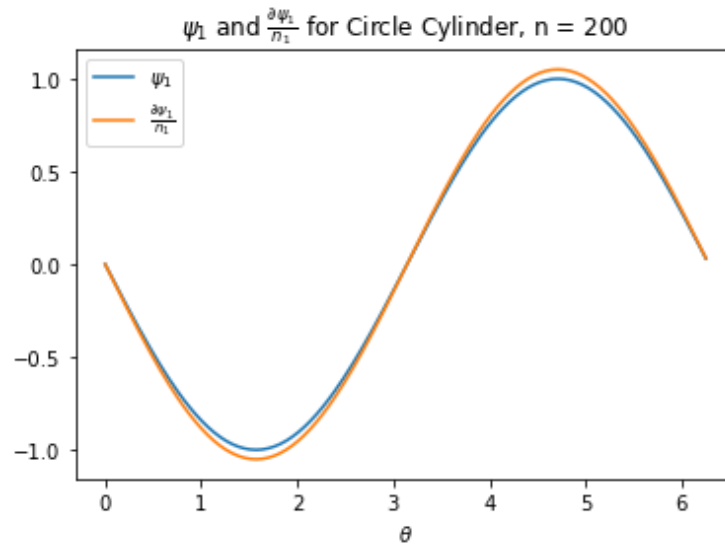


Figure 3: Plots of ψ_1 and $\frac{\partial\psi_1}{\partial n}$ for $n=200$

It can be seen that as n increase, the partial derivative has its peak getting closer to the 1 value.

Number 3

The following figures below shows the contours of the streamfunction. This corresponds to the streamlines of the flow around the cylinder.

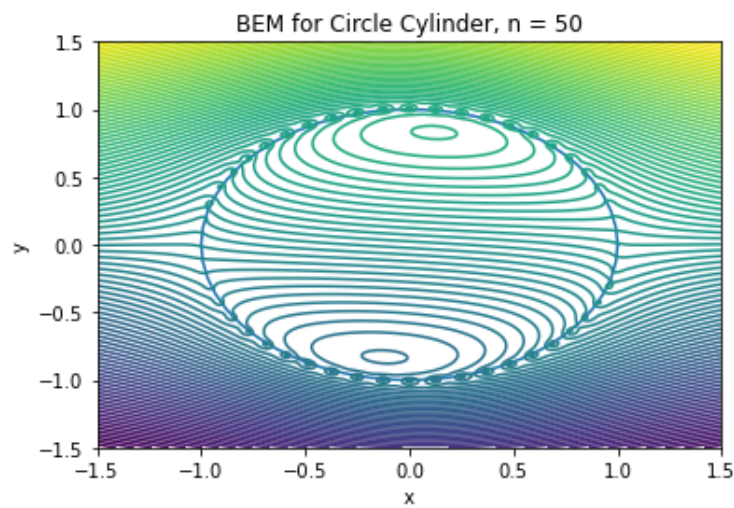


Figure 4: Streamlines for $n=50$

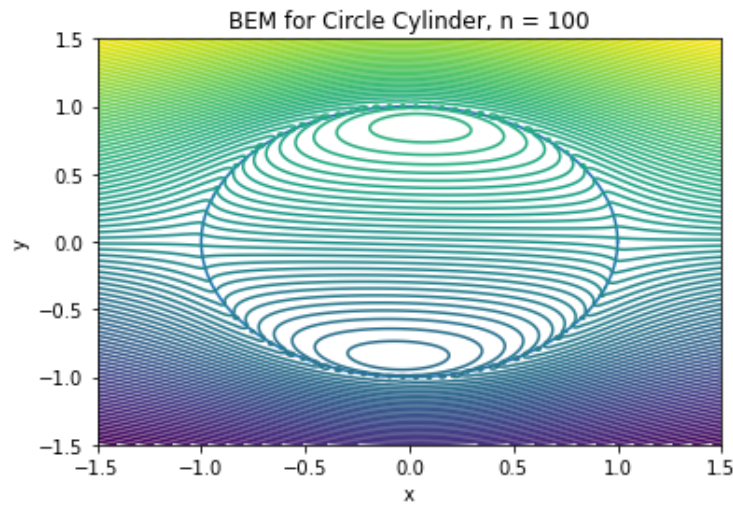


Figure 5: Streamlines for $n=100$

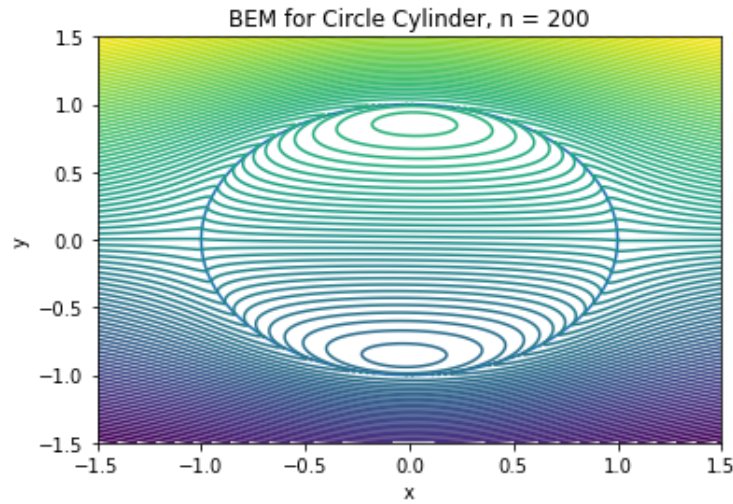


Figure 6: Streamlines for $n=200$

It can be seen that as N increases, the flow near the edge of the cylinder gets more refined. There is not that much of a difference as N increases though quantitatively.

Number 5

After modifying the code to apply to a uniform flow past a 4:1 ellipse parallel and also perpendicular to the flow. The following plots were made below. It can be seen that as n increases, the partial derivative gets smoother. For the horizontal ellipse, the flow doesn't seem to be affected that much because of its streamlines shape. The vertical ellipse changes the flow a lot due to the massive area blocking the flow. There is supposed to be no lines inside so it can be made better next time.

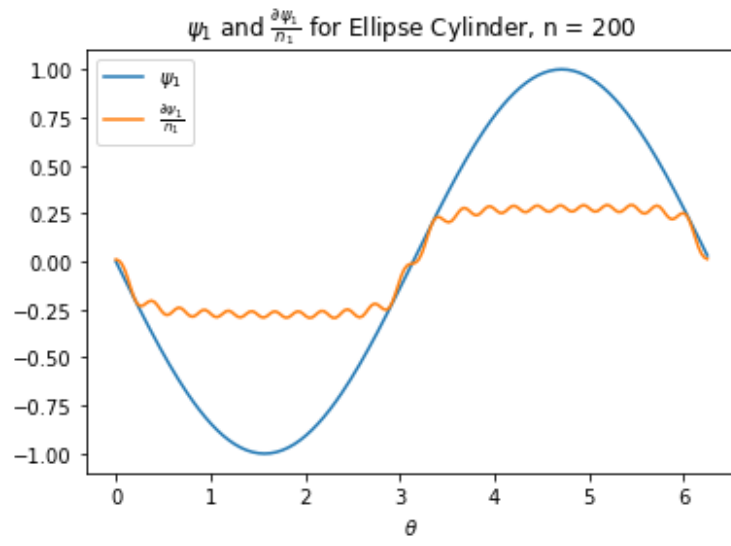


Figure 7: Plots of ψ_1 and $\frac{\partial \psi_1}{\partial n}$ for $n=200$ Horizontal

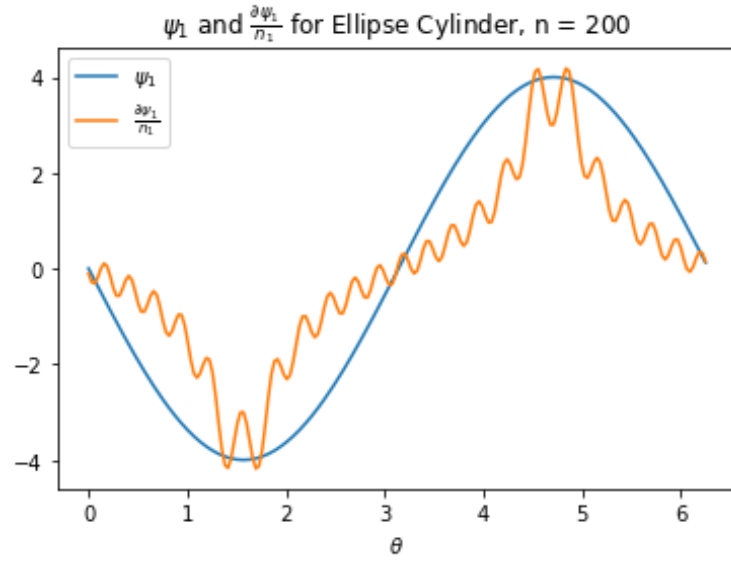


Figure 8: Plots of ψ_1 and $\frac{\partial \psi_1}{\partial n}$ for $n=200$ Vertical

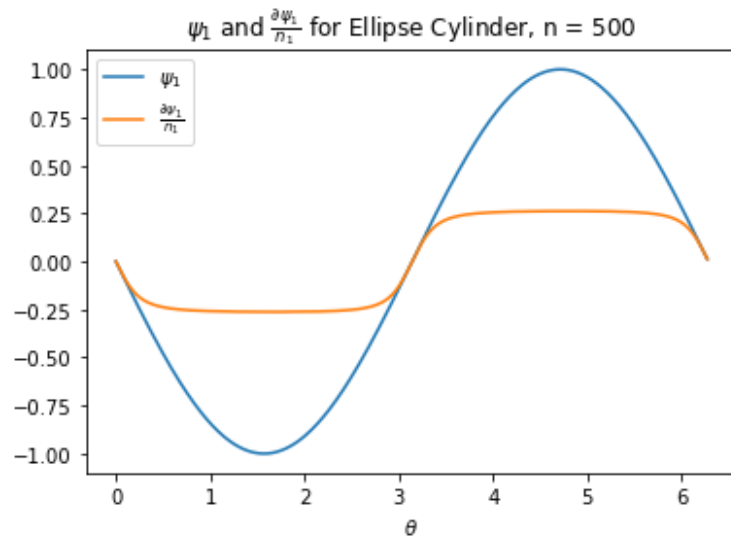


Figure 9: Plots of ψ_1 and $\frac{\partial \psi_1}{\partial n}$ for $n=500$ Horizontal

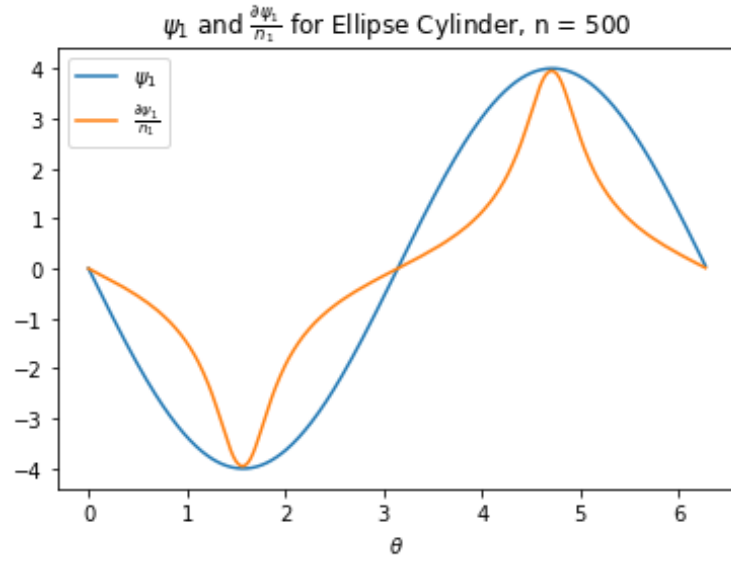


Figure 10: Plots of ψ_1 and $\frac{\partial \psi_1}{\partial n}$ for $n=500$ Vertical

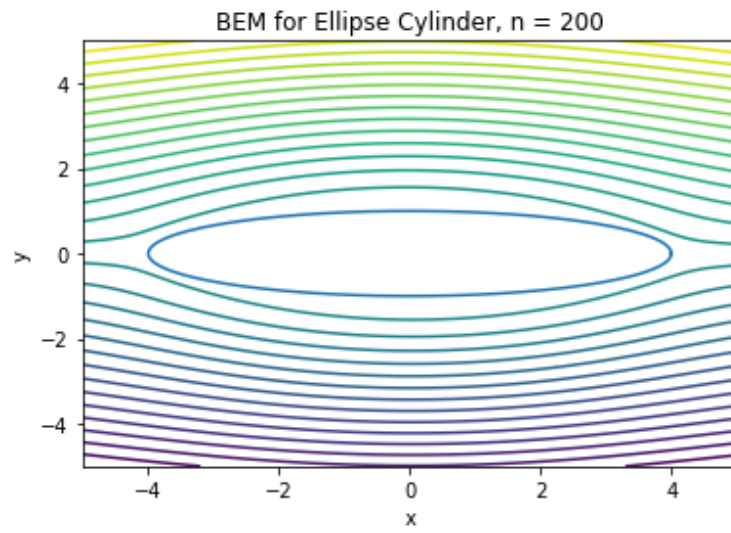


Figure 11: Streamlines for $n=500$ Horizontal

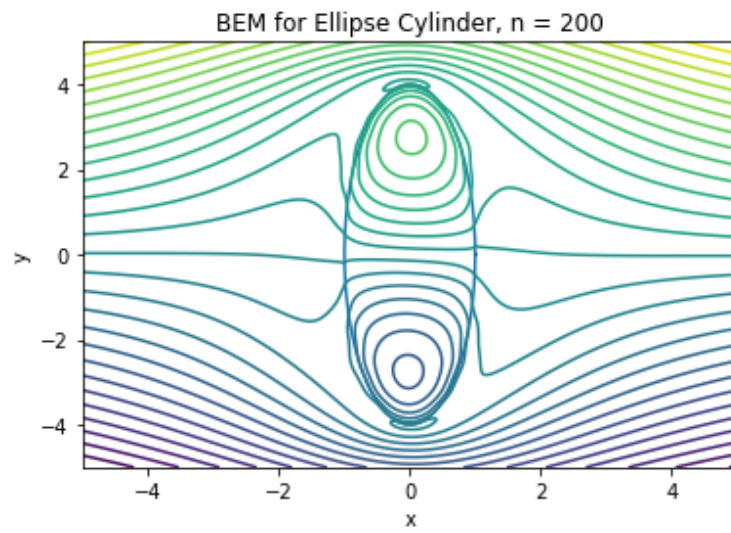


Figure 12: Streamlines for $n=500$ Vertical

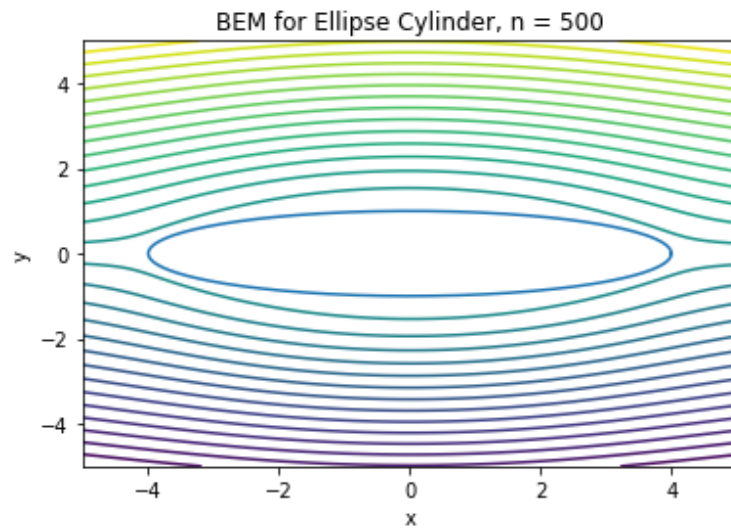


Figure 13: Streamlines for n=500 Horizontal

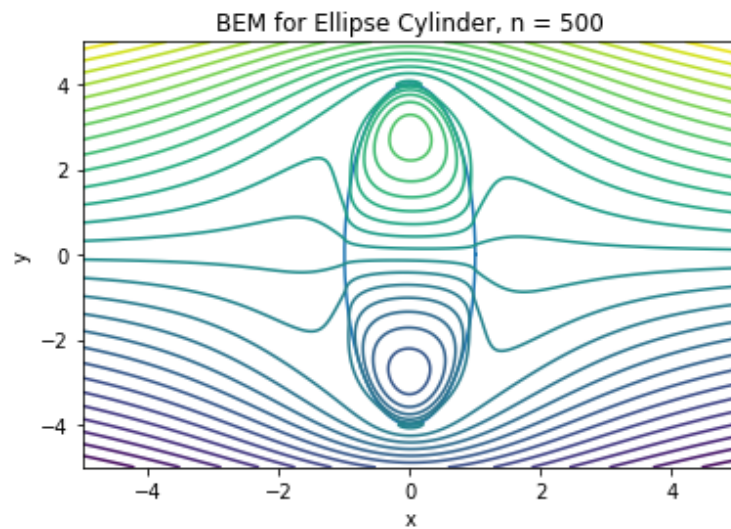


Figure 14: Streamlines for n=500 Vertical

Appendix)

Python Code for Problems 2,3,5

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Fri Feb 10 10:06:00 2023
4
5 @author: jjser
6 """
7 # Imported from CFD Project
8 import numpy as np
9 import matplotlib.pyplot as plt
10 import math
11 from numpy import linalg as la
12 from numpy import asarray
13 import pandas
14 import time

```

```

15
16 #####
17 def G(r):
18     return np.log(r)/(2*np.pi)
19 def dGdn(R,N,r):
20     return R @ N / (2*np.pi*r)
21
22 def prob(a,b,n):
23     if a == b:
24         shape = 'Circle Cylinder'
25     else:
26         shape = 'Ellipse Cylinder'
27     dtheta = 2*np.pi / n
28     x0 = np.zeros(n+1)
29     y0 = np.zeros(n+1)
30     theta = np.zeros(n+1)
31
32     # Creating the actual circle first
33     for i in range(n+1):
34         theta[i] = i * dtheta
35         #r = math.sqrt((a*b)/(a**2*np.sin(theta[i])**2+b**2*np.cos(theta[i])**2))
36         x0[i] = a*np.cos(theta[i])
37         y0[i] = b*np.sin(theta[i])
38     # Applying BEM
39     A = np.zeros((n,n))
40     B = np.zeros((n,n))
41     area = np.zeros(n)
42     cx = np.zeros(n)
43     cy = np.zeros(n)
44     N = np.zeros((n,2))
45
46     #s = 2*max(a,b)
47     s=1.5
48     fig0, ax0 = plt.subplots()
49     title = 'BEM for ' + shape + ', n = ' + str(n)
50     ax0.set_title(title)
51     ax0.set_xlabel('x')
52     ax0.set_ylabel('y')
53     ax0.set_xlim([-s,s])
54     ax0.set_ylim([-s,s])
55
56     for i in range(n):
57         area[i] = math.sqrt((x0[i+1]-x0[i])**2+(y0[i+1]-y0[i])**2)
58         cx[i] = 0.5 * (x0[i+1]+x0[i])
59         cy[i] = 0.5 * (y0[i+1]+y0[i])
60         N[i][0] = -(y0[i+1]-y0[i])
61         N[i][1] = (x0[i+1]-x0[i])
62         N[i] = N[i] / la.norm(N[i])
63     for i in range(n):
64         for j in range(n):
65             if i==j:
66                 B[i,j] = 0
67                 A[i,j] = -0.5
68             else:
69                 R = [cx[j]-cx[i],cy[j]-cy[i]]
70                 r = la.norm(R)
71                 R = R/r
72                 A[i,j] = dGdn(R,N[j],r) * area[j]
73                 B[i,j] = G(r) * area[j]
74                 #ax0.quiver(cx[i],cy[i],R[0],R[1])
75
76     # plotting
77     ax0.plot(x0,y0)
78     #ax0.plot(cx,cy,'r*')
79     #ax0.quiver(cx,cy,N[:,0],N[:,1])
80
81     # psi1 and dpsiidn
82     psi1 = np.zeros(n)
83     for i in range(n):
84         psi1[i] = -y0[i]
85
86     dpsiidn = la.inv(B) @ A @ psi1.T

```



```

87
88
89 # Plot psi1 dpsiidn
90 fig, ax = plt.subplots()
91 title = '$\\psi_1$ and $\\frac{\\partial \\psi_1}{\\partial n}$ for ' + shape + ', n = ' + str(n)
92 )
93 ax.set_title(title)
94 ax.set_xlabel('$\\theta$')
95
96 ax.plot(theta[:-1],psi1,label='$\\psi_1$')
97 ax.plot(theta[:-1],dpsiidn,label='$\\frac{\\partial \\psi_1}{\\partial n}$')
98 ax.legend()
99
100 # Plotting the Cylinder
101 g = 500
102 x = np.linspace(-s,s,g)
103 y = np.linspace(-s,s,g)
104 X,Y = np.meshgrid(x,y)
105 psi = np.zeros((g,g))
106
107 # Streamfunction Contours
108 for k in range(n):
109     rx = cx[k] - X
110     ry = cy[k] - Y
111     rmag = np.sqrt(rx**2+ry**2)
112     psi += (psi1[k]*(rx*N[k][0]+ry*N[k][1])/(2*np.pi*rmag)-G(rmag)*dpsiidn[k])*area[k]
113     psi += Y
114 levels = np.linspace(np.min(psi),np.max(psi),100)
115
116 for i in range(g):
117     for j in range(g):
118         if X[i,j]:
119             psi[i,j] = NaN
120
121 ax0.contour(x, y, psi, levels=levels)
122 ##### BEM for a 4:1 ellipse
123 prob(1,1,50)
124 prob(1,1,100)
125 prob(1,1,200)
126
127 #prob(4,1,200)
128 #prob(1,4,200)
129 #prob(4,1,500)
130 #prob(1,4,500)
131 start_time = time.time()
132 print("--- %10s seconds ---" % np.round((time.time() - start_time),4))

```