BVP

April 9, 2019

1 LAB 9

2 PDE

2.1 gauss siedel iterative scheme

Parabolic PDE of 2 variables,

$$u_{xx} + u_{yy} = 0 \quad for \quad 0 \leq x, y \leq 4$$

Boundary Condition:

$$u = x^2 y^2 ontheboundary$$

$$u_{ij}^0 = 0$$

Experiment with different δx.

```
In [4]: import numpy as np
        import pandas as pd
```

```
In [71]: x1 = 0
         x2 = 4
```

```
In [110]: def gauss_siedel(dx,x_last,x_next,y_last,y_next):
              return (x_last+x_next+y_last+y_next)/4


          def main_(dx=0.5):
              err = 0.000000001

              n = int((x2-x1)/dx)

              xy = np.zeros((n+1,n+1,2))
              u_xy_prev = np.zeros((n+1,n+1))
              u_xy_recent = np.zeros((n+1,n+1))
```

```
    for i in range(n+1):
        for j in range(n+1):
            xy[i][j][0] = x1+i*dx
            xy[i][j][1] = x1+j*dx
            if i==n or j==n:
                u_xy_prev[i][j]=xy[i][j][0]*xy[i][j][1]*xy[i][j][0]*xy[i][j][1]
                u_xy_recent[i][j]=xy[i][j][0]*xy[i][j][1]*xy[i][j][0]*xy[i][j][1]

    flag=10
    count = 0
    while(flag!=0):
        count = count+1
        for i in range(n-1):
            for j in range(n-1):
                u_xy_recent[i+1][j+1] = gauss_siedel(dx,u_xy_prev[i+2][j+1], u_xy_rece
                                         u_xy_prev[i+1][j+2], u_xy_recent[i+1][j])

        if np.max(np.absolute(u_xy_prev-u_xy_recent))<err and count > n+2:
            flag = 0
        u_xy_prev = u_xy_recent
        #flag = flag-1

    return [u_xy_recent, xy, count]
```

```
In [111]: a_1, x_1, c1 = main_(1)
          a_2, x_2, c2 = main_(0.5)
          a_3, x_3, c3 = main_(0.2)
          a_4, x_4, c4 = main_(0.1)
```

```
In [112]: print(c1,c2,c3,c4)
```

```
2 2 2 2
```

```
In [101]: np.set_printoptions(formatter={'float': lambda x: "{0:0.2f}".format(x)})
          print(a_2)
```

```
[[0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00]
 [0.00 0.11 0.40 1.08 2.36 4.23 6.15 6.85 4.00]
 [0.00 0.40 1.27 3.08 6.20 10.54 15.13 17.94 16.00]
 [0.00 1.08 3.08 6.70 12.45 20.10 28.34 34.74 36.00]
 [0.00 2.36 6.20 12.45 21.65 33.45 46.36 57.70 64.00]
 [0.00 4.23 10.54 20.10 33.45 50.21 68.84 86.66 100.00]
 [0.00 6.15 15.13 28.34 46.36 68.84 94.47 120.80 144.00]
 [0.00 6.85 17.94 34.74 57.70 86.66 120.80 158.40 196.00]
 [0.00 4.00 16.00 36.00 64.00 100.00 144.00 196.00 256.00]]
```

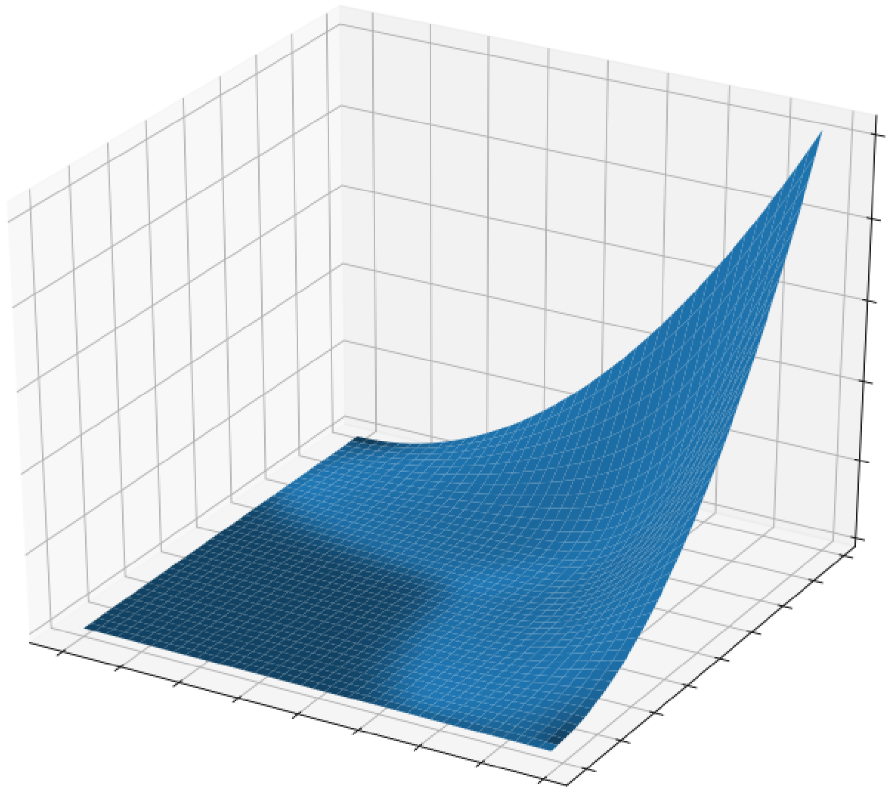
```
In [105]: from mpl_toolkits.mplot3d import Axes3D
          import matplotlib.pyplot as plt
```

```
fig = plt.figure(num=None, figsize=(12, 10), dpi=80, facecolor='w', edgecolor='k')
ax = fig.add_subplot(111, projection='3d')

x = y = np.arange(-1.0, 1.0, 2/a_4.shape[1])
X, Y = np.meshgrid(x, y)
Z = a_4
ax.plot_surface(X, Y, Z)

ax.tick_params(
    bottom=False,       # ticks along the bottom edge are off
    top=False,          # ticks along the top edge are off
    labelbottom=False, # labels along the bottom edge are off
    labelleft=False)

plt.show()
```



In [ ]:

In [ ]:

```
In [106]: x1 = 0
          x2 = 1
```

```
In [114]: def gauss_siedel(dx, x, y, x_last,x_next,y_last,y_next):
              return -1*(dx*dx*(x*x+y*y)-x_last-x_next-y_last-y_next)/4


          def main_2(dx=0.25):
              err = 0.000000001

              n = int((x2-x1)/dx)

              xy = np.zeros((n+1,n+1,2))
              u_xy_prev = np.zeros((n+1,n+1))
              u_xy_recent = np.zeros((n+1,n+1))

              for i in range(n+1):
                  for j in range(n+1):
                      xy[i][j][0] = x1+i*dx
                      xy[i][j][1] = x1+j*dx

              flag=10
              count = 0
              while(flag!=0):
                  count = count+1
                  for i in range(n-1):
                      for j in range(n-1):
                          u_xy_recent[i+1][j+1] = gauss_siedel(dx,xy[i][j][0], xy[i][j][1], u_xy
                                                   u_xy_prev[i][j+1],
                                                   u_xy_prev[i+1][j+2],
                                                   u_xy_prev[i+1][j])

                  if np.max(np.absolute(u_xy_prev-u_xy_recent))<err and count > n+2:
                      flag = 0
                  u_xy_prev = u_xy_recent
                  #flag = flag-1

              return [u_xy_recent, xy, count]
```

```
In [116]: a_1, x_1, c1 = main_2(1)
          a_2, x_2, c2 = main_2(0.5)
          a_3, x_3, c3 = main_2(0.2)
          a_4, x_4, c4 = main_2(0.1)
```

```
In [117]: print(c1,c2,c3,c4)
```

```
7 11 23 43
```

```
In [118]: np.set_printoptions(formatter={'float': lambda x: "{0:0.2f}".format(x)})
          print(a_2)
```

```
[[0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00]
 [0.00 -0.24 -0.55 -0.96 -1.45 -1.92 -2.17 -1.78 0.00]
 [0.00 -0.55 -1.15 -1.86 -2.66 -3.35 -3.59 -2.77 0.00]
 [0.00 -0.96 -1.86 -2.82 -3.80 -4.57 -4.69 -3.48 0.00]
 [0.00 -1.45 -2.66 -3.80 -4.86 -5.63 -5.61 -4.07 0.00]
 [0.00 -1.92 -3.35 -4.57 -5.63 -6.33 -6.21 -4.47 0.00]
 [0.00 -2.17 -3.59 -4.69 -5.61 -6.21 -6.06 -4.41 0.00]
 [0.00 -1.78 -2.77 -3.48 -4.07 -4.47 -4.41 -3.33 0.00]
 [0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00]]
```

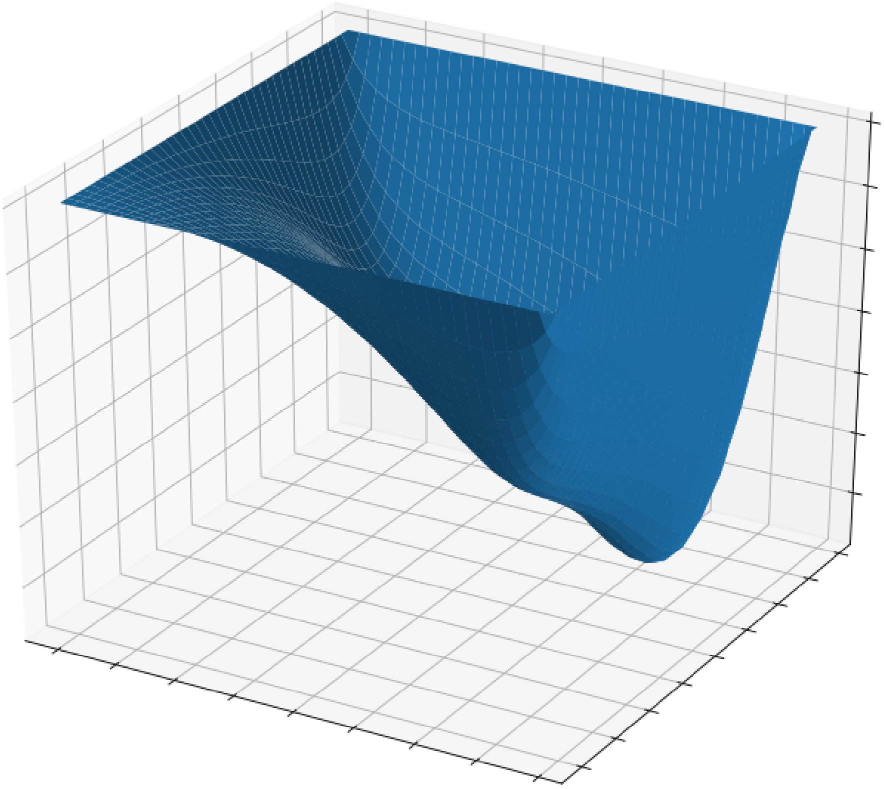
```
In [121]: from mpl_toolkits.mplot3d import Axes3D
          import matplotlib.pyplot as plt

          fig = plt.figure(num=None, figsize=(12, 10), dpi=80, facecolor='w', edgecolor='k')
          ax = fig.add_subplot(111, projection='3d')

          x = y = np.arange(-1.0, 1.0, 2/a_4.shape[1])
          X, Y = np.meshgrid(x, y)
          Z = a_4
          ax.plot_surface(X, Y, Z)

          ax.tick_params(
              bottom=False,      # ticks along the bottom edge are off
              top=False,         # ticks along the top edge are off
              labelbottom=False, # labels along the bottom edge are off
              labelleft=False)

          plt.show()
```



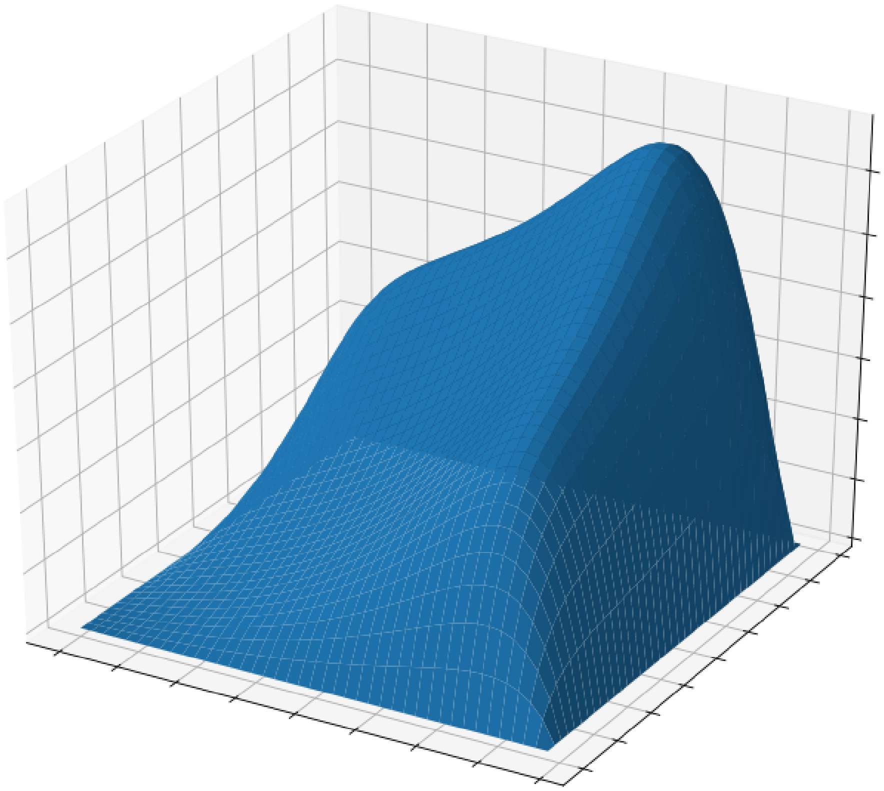
```
In [122]: from mpl_toolkits.mplot3d import Axes3D
          import matplotlib.pyplot as plt

          fig = plt.figure(num=None, figsize=(12, 10), dpi=80, facecolor='w', edgecolor='k')
          ax = fig.add_subplot(111, projection='3d')

          x = y = np.arange(-1.0, 1.0, 2/a_4.shape[1])
          X, Y = np.meshgrid(x, y)
          Z = a_4
          ax.plot_surface(X, Y, np.absolute(Z))

          ax.tick_params(
              bottom=False,      # ticks along the bottom edge are off
              top=False,         # ticks along the top edge are off
              labelbottom=False, # labels along the bottom edge are off
              labelleft=False)

          plt.show()
```



```
In [ ]:
```