Exercise 1: matrix madness

1. Perform the matrix multiplication

$$\begin{bmatrix} 3 & -2 & 1 \\ 6 & 0 & 4 \\ 5 & 2 & 3 \end{bmatrix} \begin{bmatrix} -1 & 4 \\ 3 & 2 \\ 7 & -4 \end{bmatrix}$$

2. Find the inverse of the matrix

$$\begin{bmatrix} 3.6 & -4.1 & 1.3 & 2.5 \\ 0.4 & 6.2 & -1.5 & 3.0 \\ 5.9 & 7.0 & -3.1 & 8.2 \\ 4.8 & -5.6 & 2.9 & 7.3 \end{bmatrix}$$

3. Solve the linear system

$$\begin{bmatrix} 4 & 1 & -3 & 2 \\ 0 & 7 & 5 & 1 \\ -9 & 2 & 7 & 3 \\ -4 & 5 & 1 & 8 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 2 \\ 8 \\ 1 \\ 7 \end{bmatrix}$$

and check the result.

In [18]:

```
# Code Here
import numpy as np
import numpy.linalg as la
m1 = np.array([[3,-2,1],[6,0,4],[5,2,3]])
m2 = np.array([[-1,4],[3,2],[7,-4]])
res = m1@m2 #another way of matrix multiplication
print(res)
#Q2 inv
m3 = np.array([[3.6,-4.1,1.3,2.5],[0.4,6.2,-1.5,3.0],[5.9,7,-3.1,8.2],[4.8,-5.6,2.9
res2 = la.inv(m3)
print(res2)
m4 = np.array([[4,1,-3,2],[0,7,5,1],[-9,2,7,3],[-4,5,1,8]])
b = np.array([[2],[8],[1],[7]])
x = la.solve(m4,b)
print(x)
print(np.dot(m4,x) -b) #these here are conditioning errors
```

```
[[-2  4]
 [22  8]
 [22 12]]
[[ 0.97813868  0.52433284 -0.17479228 -0.35411624]
 [ 0.60841088  0.67406156 -0.25603252 -0.19777332]
 [ 1.35007366  1.45103917 -0.70650193 -0.26506655]
 [-0.71276416 -0.40411869  0.19918855  0.32341375]]
[[-0.198]
 [ 1.514]
 [-0.498]
 [-0.108]]
[[4.44089210e-16]
 [0.00000000e+00]
 [0.00000000e+00]
 [1.77635684e-15]]
```

Exercise 2: rotation

Rotating a 2D vector through an angle μ corresponds to multiplying on the left by the matrix

$$R(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

1. Create a list of 2D position vectors corresponding to a 10x10 grid of points in the unit square. Plot the positions using dots.

2. Define a function that returns the rotation matrix for a given angle θ.

3. Use your function to rotate the points through an angle of 25° and plot the new positions using dots of a different color.

4. How would you get the points to rotate about (0.5, 0.5) instead of the origin?

In [54]:

```
# Code here
# R_theta = np.array([[np.cos],])
# np.cos(0)
import matplotlib.pyplot as plt
def rotation_matrix(theta):
    return np.array([[np.cos(theta),-np.sin(theta)],[np.sin(theta),np.cos(theta)]])

# points in example are (0,0),()
x = np.linspace(0, 1, 10)
y = np.linspace(0, 1, 10)
X, Y = np.meshgrid(x, y)
print(X)
# print(X.ravel())
positions = ([X.ravel(), Y.ravel()])
# print(positions)
plt.figure(figsize=(6, 6))
plt.scatter(positions[0], positions[1], c='blue', s=10)
plt.title('Grid containing 100 points using x,y meshgrid')
plt.xlabel('X')
```

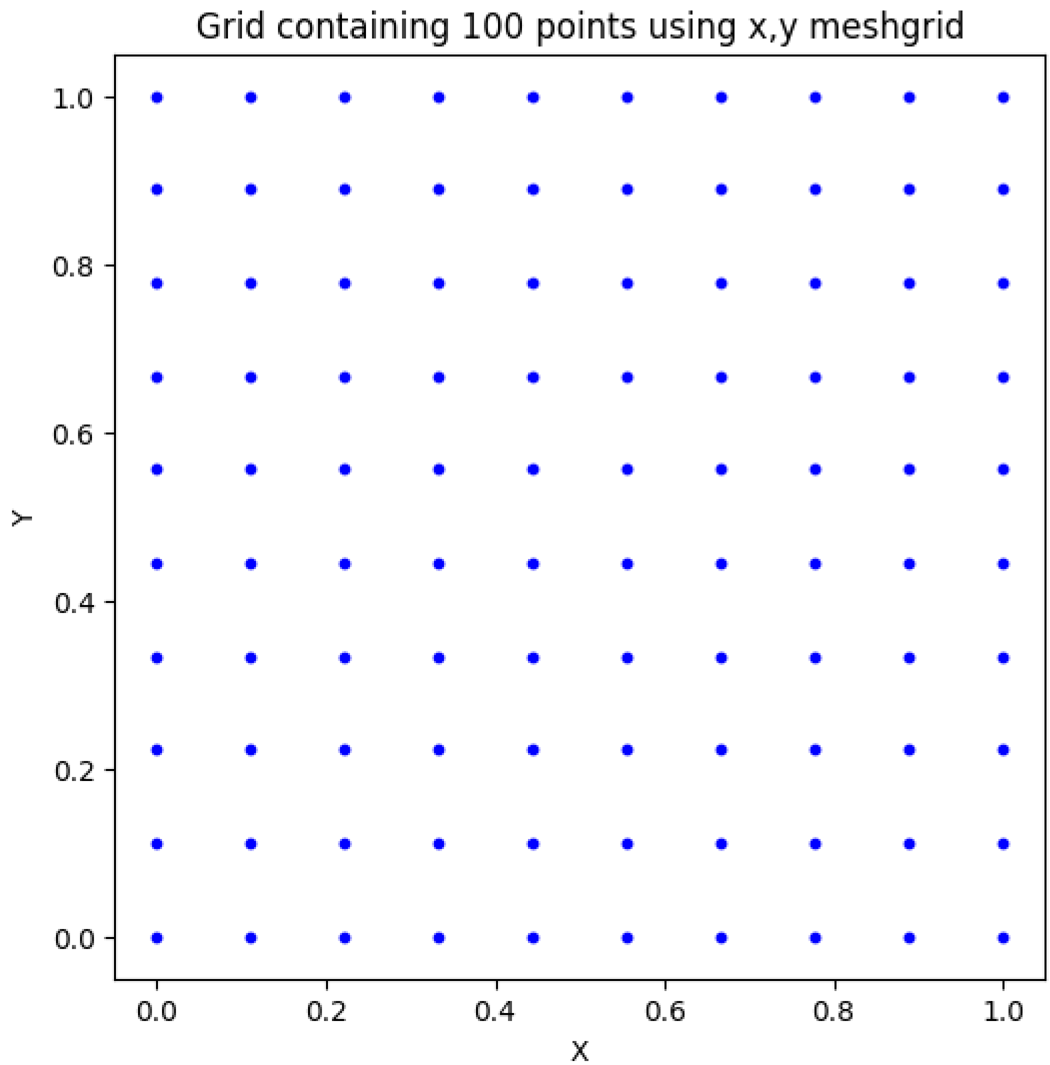
```python
plt.ylabel('Y')
degress = 25
R = rotation_matrix(degress)
new_pos = R @ positions

plt.figure(figsize=(6, 6))
plt.scatter(new_pos[0], new_pos[1], c='red', s=10)
plt.title('Grid containing 100 points rotated using x,y meshgrid')
plt.xlabel('X')
plt.ylabel('Y')
# together HERE
plt.figure(figsize=(6, 6))
plt.scatter(positions[0], positions[1], c='blue', s=10)
plt.scatter(new_pos[0], new_pos[1], c='red', s=10)
plt.title('Grid containing 100 points both original and rotated using x,y meshgrid'
plt.xlabel('X')
plt.ylabel('Y')
```

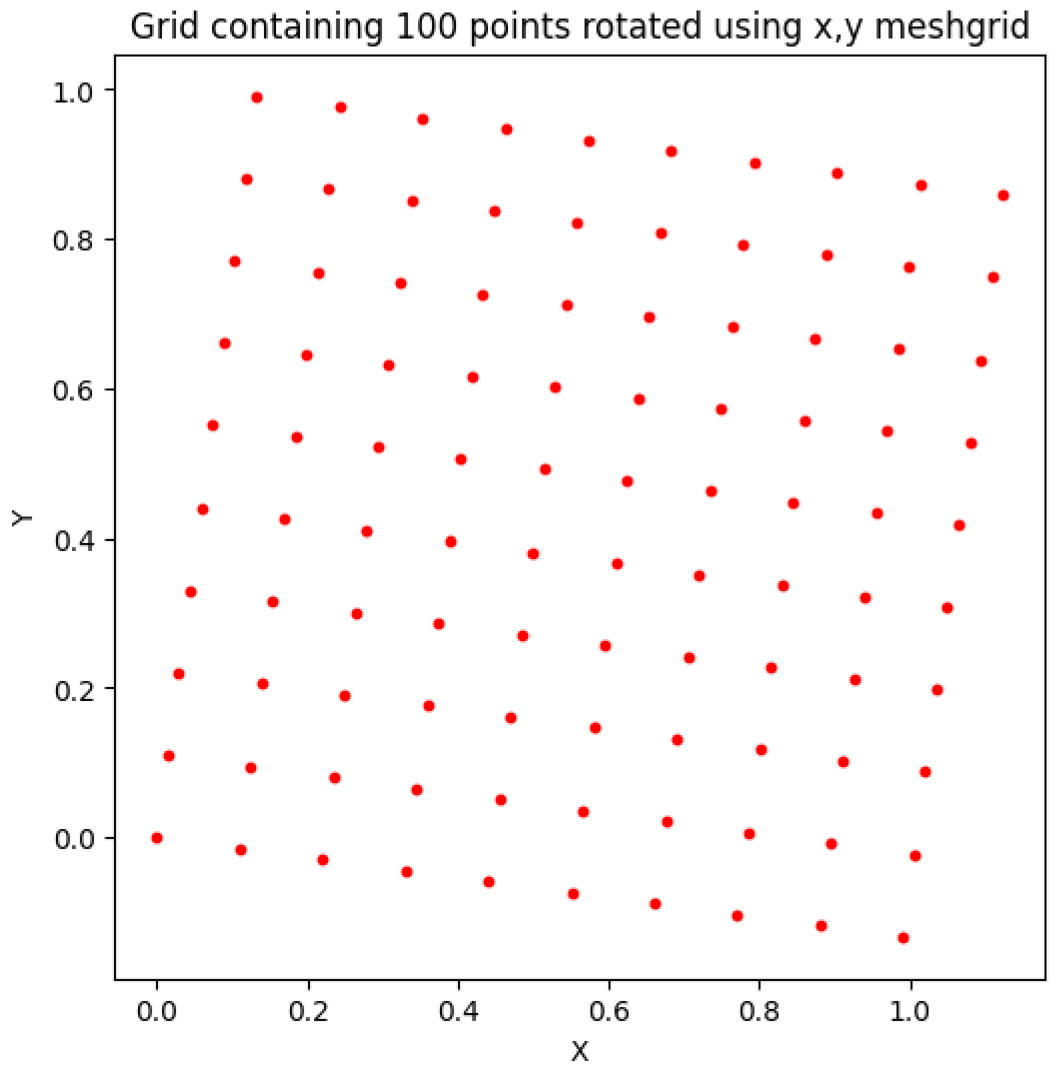
```
[[0.         0.11111111 0.22222222 0.33333333 0.44444444 0.55555556
  0.66666667 0.77777778 0.88888889 1.        ]
 [0.         0.11111111 0.22222222 0.33333333 0.44444444 0.55555556
  0.66666667 0.77777778 0.88888889 1.        ]
 [0.         0.11111111 0.22222222 0.33333333 0.44444444 0.55555556
  0.66666667 0.77777778 0.88888889 1.        ]
 [0.         0.11111111 0.22222222 0.33333333 0.44444444 0.55555556
  0.66666667 0.77777778 0.88888889 1.        ]
 [0.         0.11111111 0.22222222 0.33333333 0.44444444 0.55555556
  0.66666667 0.77777778 0.88888889 1.        ]
 [0.         0.11111111 0.22222222 0.33333333 0.44444444 0.55555556
  0.66666667 0.77777778 0.88888889 1.        ]
 [0.         0.11111111 0.22222222 0.33333333 0.44444444 0.55555556
  0.66666667 0.77777778 0.88888889 1.        ]
 [0.         0.11111111 0.22222222 0.33333333 0.44444444 0.55555556
  0.66666667 0.77777778 0.88888889 1.        ]
 [0.         0.11111111 0.22222222 0.33333333 0.44444444 0.55555556
  0.66666667 0.77777778 0.88888889 1.        ]
 [0.         0.11111111 0.22222222 0.33333333 0.44444444 0.55555556
  0.66666667 0.77777778 0.88888889 1.        ]]
```

Out[54]:

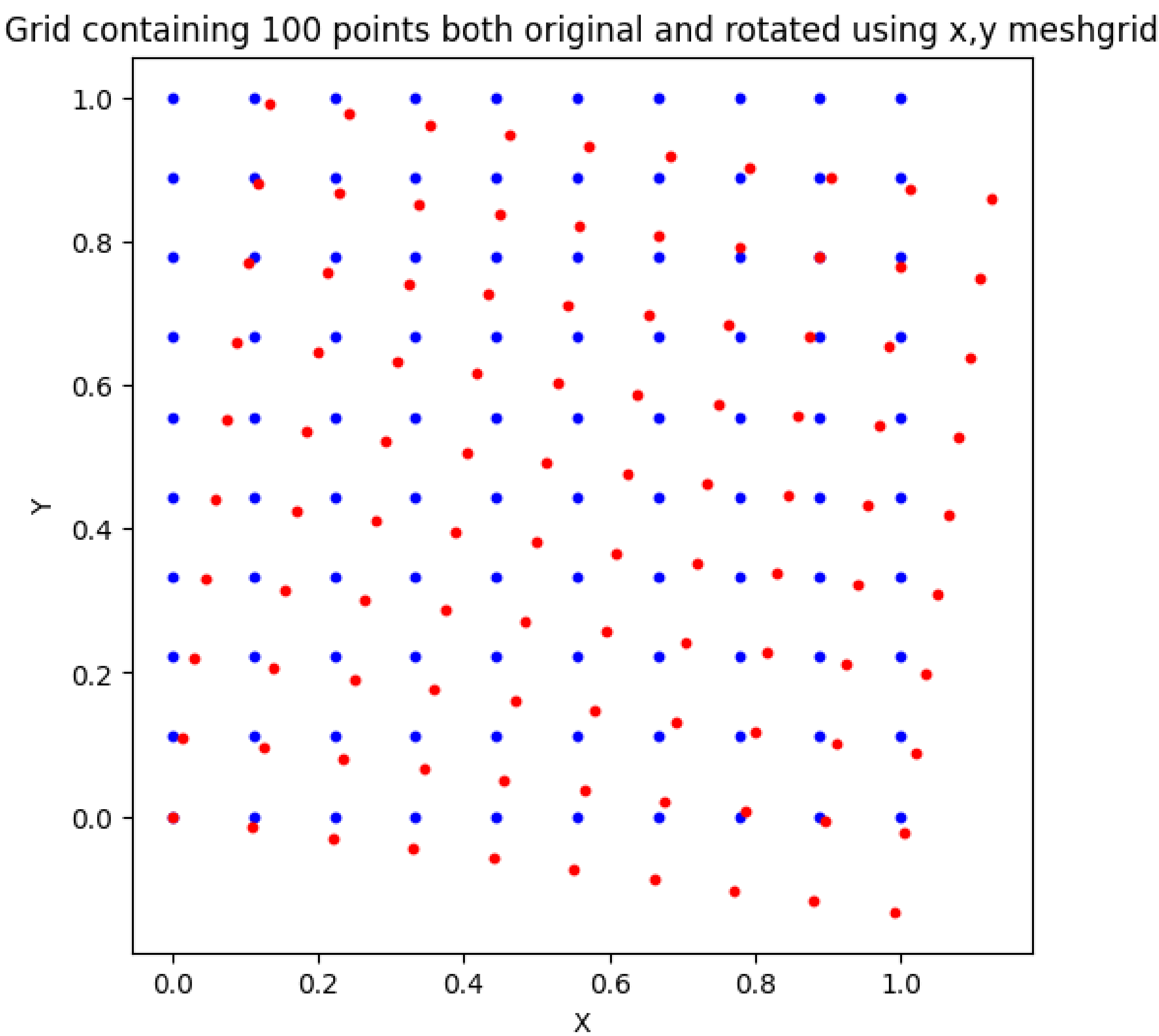
```
Text(0, 0.5, 'Y')
```



Grid containing 100 points using x,y meshgrid
1.0
0.8
0.6
0.4
0.2
0.0
Y
0.0
0.2
0.4
0.6
0.8
1.0
X



Grid containing 100 points rotated using x,y meshgrid
1.0
0.8
0.6
0.4
0.2
0.0
Y
0.0
0.2
0.4
0.6
0.8
1.0
X



Exercise 3: chemical equilibrium

Consider a mixture of three particle species that interact via the reactions

$$A + A \rightleftharpoons B$$
$$A + B \rightleftharpoons C$$

At a particular temperature, the reaction rates are such that, at equilibrium, we have... (please refer pdf)

In [60]:
```
# Code Here
from scipy.linalg import solve
Kaa = 10**-8 #cm**3s**-1
Kab = 10**-5
taub = 100
tauc = 0.1
n_total = 10**6
# n = na+nb+nc = 10**6 # just to help me solve
# nA/n,nB/n, Nc/n
n_A = n_total
n_B = 0
```

```python
for _ in range(25): #iteratively solving with threshold 25
    m7 = np.array([
        [Kaa / tauc, -1 / taub],
        [Kab / tauc + 1 / tauc, 1]
    ])
    c = np.array([0, n_total / tauc])
    n_A, n_B = solve(m7, c)
    n_C = n_total - n_A - n_B
n_A = n_A / n_total
n_B = n_B / n_total
n_C = n_C / n_total
n_A, n_B, n_C
```

Out[60]: (0.9999890001209987, 9.999890001209985e-06, 9.999890000805518e-07)