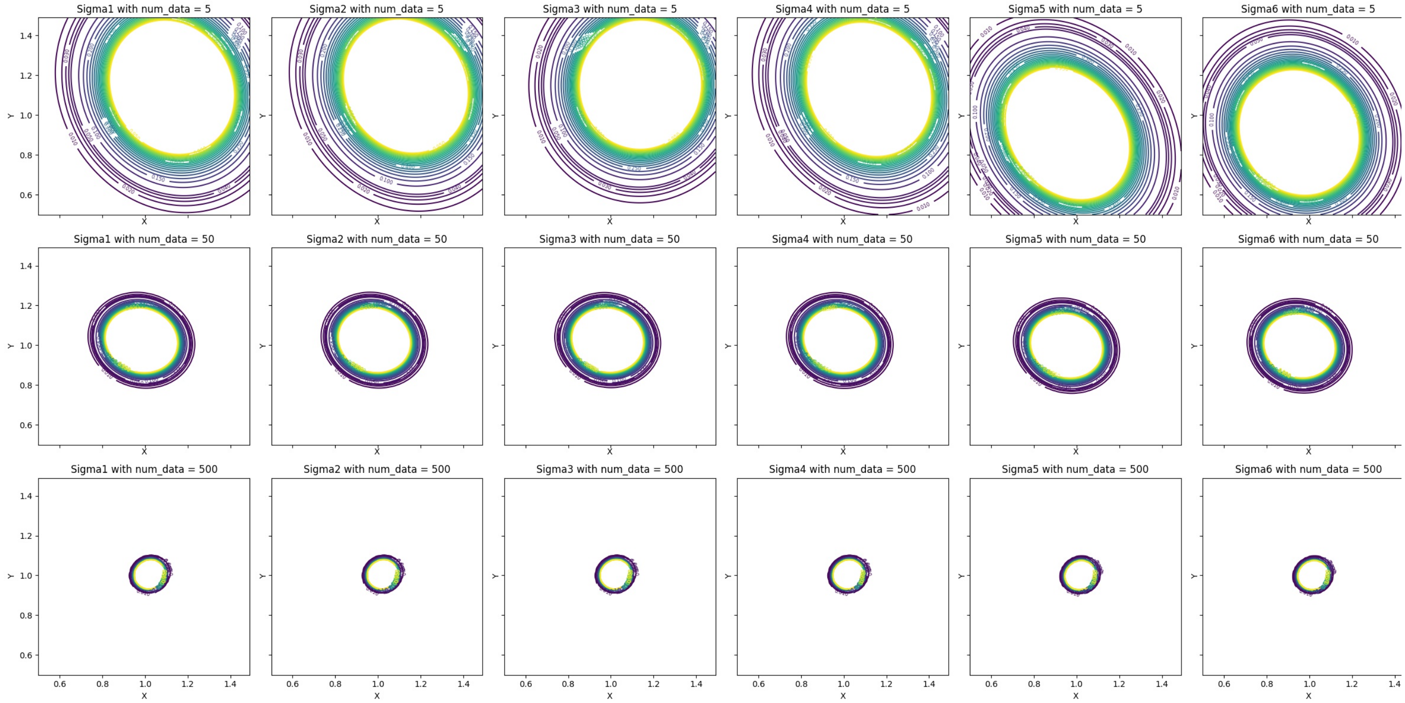
3d

As the number of date point increases, the posterior mean becomes closer to true w, and posterior variance becomes smaller.



```
import matplotlib
matplotlib.use('Qt5Agg')
import numpy as np
from numpy.linalg import inv
import matplotlib.pyplot as plt
from matplotlib.widgets import Slider
np.random.seed(0)
from math import sqrt
```

```
def generate_data(n):
    """
    This function generates data of size n.
    """
    X = np.random.multivariate_normal([0, 0], [[5, 0], [0, 5]], size = n)
    z = np.random.normal(0, 1, size = n).reshape((n, 1))
    y = X @ np.array([[1], [1]]) + z
    print(X.shape)
    print(z.shape)
    print(y.shape)

    return (X,y)

def tikhonov_regression(X,y,Sigma):
    """
    This function computes w based on the formula of tikhonov_regression.
    """
    mux, muy, dummy, dummy, dummy = compute_mean_var(X,y,Sigma)
    return [mux, muy]

def compute_mean_var(X,y,Sigma):
    """
    This function computes the mean and variance of the posterior
    """
    pos_var = inv(X.T @ X + inv(Sigma))
    pos_mu = pos_var @ X.T @ y
    mux = pos_mu[0, 0]
    muy = pos_mu[1, 0]
    sigmax = sqrt(pos_var[0, 0])
    sigmay = sqrt(pos_var[1, 1])
    sigmaxy = pos_var[0, 1]
    return mux,muy,sigmax,sigmay,sigmaxy

Sigmas = [np.array([[1,0],[0,1]]), np.array([[1,0.25],[0.25,1]]),
          np.array([[1,0.9],[0.9,1]]), np.array([[1,-0.25],[-0.25,1]]),
          np.array([[1,-0.9],[-0.9,1]]), np.array([[0.1,0],[0,0.1]])]
names = [str(i) for i in range(1,6+1)]


fig, ax = plt.subplots(3, 6, figsize = (24, 12), sharex = True, sharey = True)

for j, num_data in enumerate([5,50,500]):
    X,Y = generate_data(num_data)
    for i,Sigma in enumerate(Sigmas):
```

```
    mux,muy,sigmax,sigmay,sigmaxy = compute_mean_var(X,Y,Sigma)

    x = np.arange(0.5, 1.5, 0.01)
    y = np.arange(0.5, 1.5, 0.01)
    X_grid, Y_grid = np.meshgrid(x, y)

    Z = matplotlib.mlab.bivariate_normal(X_grid,Y_grid, sigmax, sigmay, mux, muy, sigmaxy)



    # plot
    #plt.figure(figsize=(10,10))
    #CS = plt.contour(X_grid, Y_grid, Z,
    #            levels = np.concatenate([np.arange(0,0.05,0.01),np.arange(0.05,1,0.05)]))
    #plt.clabel(CS, inline=1, fontsize=10)
    #plt.xlabel('X')
    #plt.ylabel('Y')
    #plt.title('Sigma'+ names[i] + ' with num_data = {}'.format(num_data))
    #plt.savefig('Sigma'+ names[i] + '_num_data_{}.png'.format(num_data))

    CS = ax[j, i].contour(X_grid, Y_grid, Z,
            levels = np.concatenate([np.arange(0,0.05,0.01),np.arange(0.05,1,0.05)]))
    ax[j, i].clabel(CS, inline=1, fontsize=6)
    ax[j, i].set_xlabel('X')
    ax[j, i].set_ylabel('Y')
    #ax[j, i].axis('equal')
    ax[j, i].set_title('Sigma'+ names[i] + ' with num_data = {}'.format(num_data))
plt.tight_layout()
plt.savefig('3d.jpg')
```