```
# -*- coding: utf-8 -*-

import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import multivariate_normal as mv
from scipy.stats import norm

func = mv(mean=[1,9],cov=[[3,0],[0,2]])

x, y = np.mgrid[-2.0:4.0:100j, 6.0:12.0:100j]
xy = np.column_stack([x.flat,y.flat])
z = func.pdf(xy)
z = z.reshape(x.shape)

from mpl_toolkits.mplot3d import Axes3D
fig = plt.figure()
ax = fig.add_subplot(111,projection='3d')
ax.plot_surface(x,y,z,alpha=0.8)
ax.set_title('Base Function')
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('MV Gaussian')
plt.savefig('pdf.pdf')


def mcmc(samp,z,init,cov,prior_mu,samples=1000,proposal_width=0.1):

    posterior = np.empty((0,len(init)))

    prior_std = cov

    var_current = []
    for var in init:
        var_current.append(var)

    for sample in range(samples):

        post = []

        for i in range(len(init)):


            var_proposed = norm(var_current[i],proposal_width).rvs()

            params_prop = [a for a in var_current]
            params_prop[i] = var_proposed


            #Assuming sigma=1 for the simplest case


            likelihood_current = mv(mean=var_current,cov=cov).pdf(samp)
            likelihood_proposed = mv(mean=params_prop,cov=cov).pdf(samp)

            prior_current = mv(mean=prior_mu,cov=prior_std).pdf(var_current)
            prior_proposed = mv(mean=prior_mu,cov=prior_std).pdf(params_prop)
```

```
            p_current = np.sum([np.log(a) for a in likelihood_current])*prior_current
            p_proposed = np.sum([np.log(a) for a in likelihood_proposed])*prior_proposed
            #print(p_proposed/p_current)
            accept = p_proposed / p_current > np.random.rand()

            if accept:
                #print(f"{i}  |  {var_current}  ->  {var_proposed}")
                var_current[i] = var_proposed
            post.append(var_current[i])
        posterior = np.r_[posterior,[post]]
    return posterior

samp = func.rvs(5000)

plt.figure()
plt.scatter(samp[:,0],samp[:,1])
plt.title('Random Draws')
plt.xlabel('x')
plt.ylabel('y')
plt.savefig('draws.pdf')



post = mcmc(samp,z,[0,0],[[3,0],[0,2]],[1,9],samples=10000)

mu = [np.mean(post[:,0]),np.mean(post[:,1])]

fit = mv(mean=mu,cov=[[3,0],[0,2]])

fitz = fit.pdf(xy)
fitz = fitz.reshape(x.shape)

fig = plt.figure()
ax = fig.add_subplot(111,projection='3d')
ax.plot_surface(x,y,fitz,alpha=0.8)
ax.set_title('Fit')
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('MV Gaussian')
plt.savefig('fit.pdf')

color = np.linspace(0,len(post),len(post))
plt.figure()
plt.title('Burn-in')
plt.xlabel('$\mu_x$')
plt.ylabel('$\mu_y$')
plt.scatter(post[:,0],post[:,1],c=color)
plt.set_cmap('brg')
clb = plt.colorbar()
clb.ax.set_title("Step")
plt.savefig('burn.pdf')
```