gibbs_sampling

February 2, 2018

```
In [228]: import numpy as np
          from matplotlib import pyplot as plt
          import csv
          from scipy.stats import norm
          from scipy.stats import invgamma
          from scipy.stats import gaussian_kde
```

1 Code to Implement Gibbs Method of MCMC

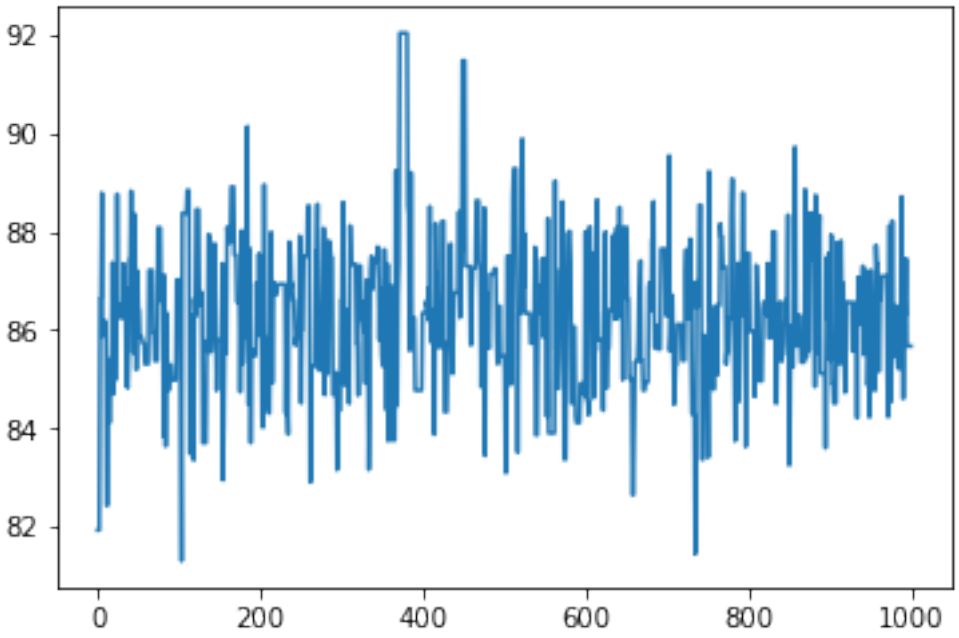
```
In [229]: def gibbs(y, nu, tau2, alpha, beta, n_samples):
              """
              Assuming a likelihood and priors
              y_i ~ N(mu, sigma2),
              mu ~ N(nu, tau2),
              sigma2 ~ IG(alpha, beta),
              sample from the posterior distribution
              P(mu, sigma2 | y, nu, tau2, alpha, beta)
              using a gibbs sampler.
              Parameters
              ----------
              y : ndarray of shape (N,)
              The data
              nu : float
              The prior mean parameter for mu
              tau2 : float > 0
              The prior variance parameter for mu
              alpha : float > 0
              The prior alpha parameter for sigma2
              beta : float > 0
              The prior beta parameter for sigma2
              n_samples : int
              The number of samples to draw
              Returns
              -------
              samples : ndarray of shape (n_samples,2)
              1st col = mu samples, 2nd col = sigma2 samples
```

```
        """
        mu, sig2=norm.rvs(nu,scale=np.sqrt(tau2)), invgamma.rvs(alpha,scale=beta)
        samples=np.zeros((n_samples, 2))
        samples[0]=[mu, sig2]
        for i in range(1,n_samples):
            j=np.random.binomial(1,.5)
            if j==0:
                tau2_star=1/((1/tau2) + (len(y)/sig2))
                nu_star=tau2_star *((nu/tau2)+(1/sig2)*np.sum(y))
                mu=norm.rvs(nu_star, scale=np.sqrt(tau2_star))
            else:
                alpha_star=alpha+len(y)/2.
                beta_star=beta+(1/2)*np.sum((y-mu)**2)
                sig2=invgamma.rvs(alpha_star, scale=beta_star)
            samples[i]=[mu,sig2]
        return samples
```

In [230]:
```
v=80
tau2=16
alpha=3
beta=50
N=1000
scores=[]
with open('examscores.csv', 'r') as filename:
    exams=csv.reader(filename,delimiter='\n')
    for row in exams:
        scores.append(int(row[0]))
scores=np.array(scores)
solution=gibbs(scores,v, tau2, alpha, beta, N)
```

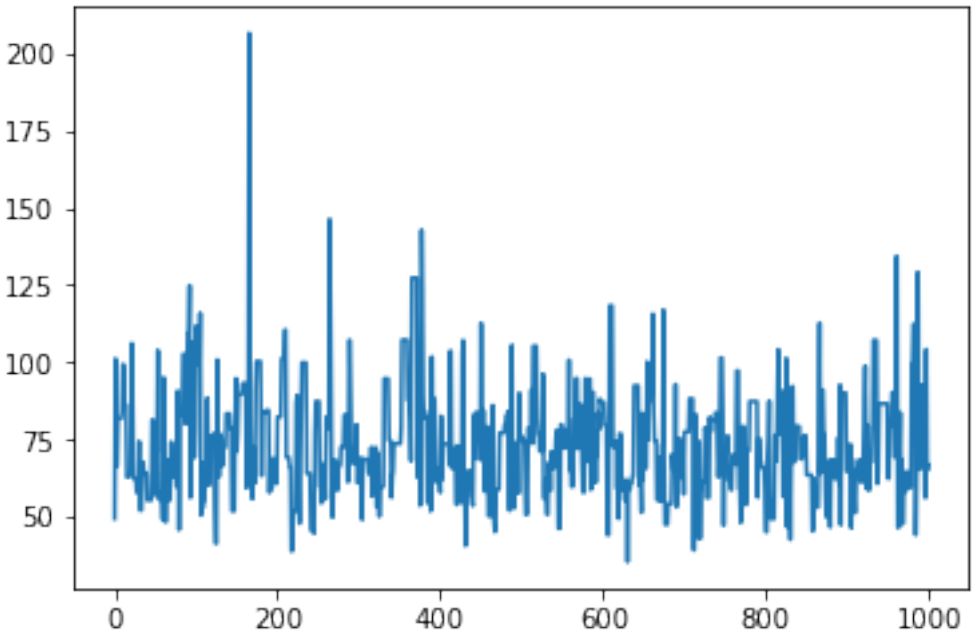
1.0.1 Mu samples

In [231]:
```
plt.plot(range(N), solution[:,0])
plt.show()
```



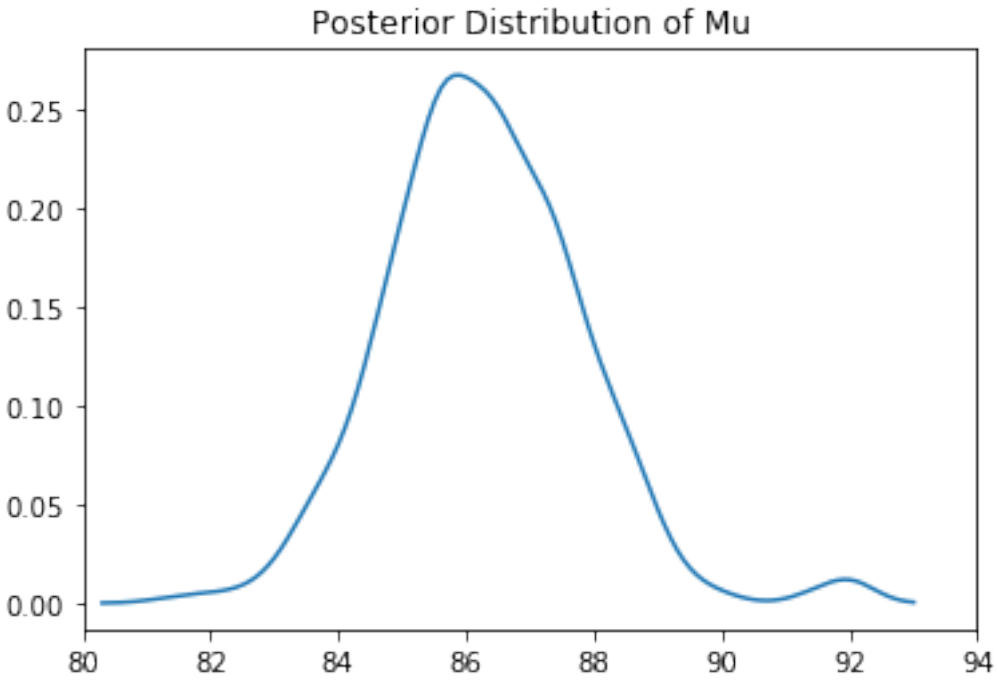
1.0.2 Sigma^2 Samples

```
In [232]: plt.plot(range(N), solution[:,1])
          plt.show()
```

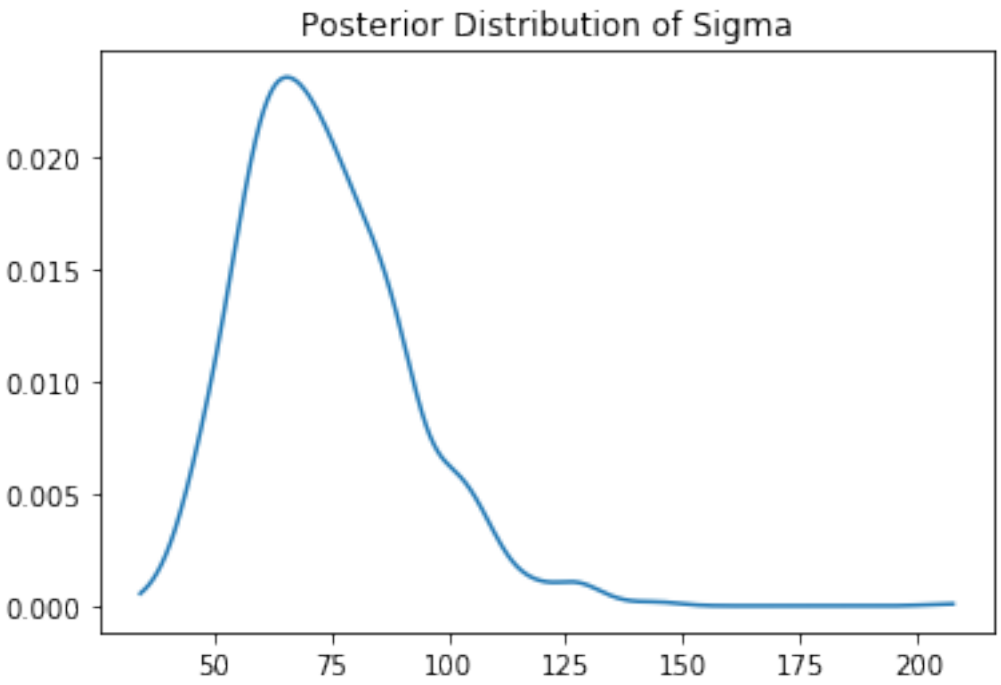


2 Problem 2

```
In [233]: mu_kernel=gaussian_kde(solution[:,0])
          xmin=min(solution[:,0])-1
          xmax=max(solution[:,0])+1
          x=np.arange(xmin,xmax,step=.1)
          plt.plot(x,mu_kernel(x))
          plt.title("Posterior Distribution of Mu")
          plt.xlim(80,94)
          plt.show()
```




```
In [234]: mu_kernel=gaussian_kde(solution[:,1])
          xmin=min(solution[:,1])-1
          xmax=max(solution[:,1])+1
          x=np.arange(xmin,xmax,step=.1)
          plt.plot(x,mu_kernel(x))
          plt.title("Posterior Distribution of Sigma")
          plt.show()
```


Posterior Distribution of Sigma
0.020
0.015
0.010
0.005
0.000
50
75
100
125
150
175
200

In [ ]:

In [ ]: