

## Computer Practical: Gibbs Sampling Model Answers

### Model code in Python

```
1 from scipy import*
2 from pylab import plot, show, figure, axis, step
3 from scipy import stats
4
5 #Task 1
6 # Generating 3000 samples using the Gibbs sampling method.
7 n = 3000
8 #set the mean and covariance parameters
9 mu = [0,0]
10 cov = [[4,1],[1,4]]
11
12 #define functions to return the mean of the full conditional distributions
13 def sam1(x):
14     x1 = mu[0] + cov[0][1]/cov[1][1]*(x-mu[1])
15     return x1
16 def sam2(x):
17     x2 = mu[1] + cov[1][0]/cov[0][0]*(x-mu[0])
18     return x2
19
20 #compute the covariance of the full conditional distributions
21 val1 = sqrt(cov[0][0]-cov[0][1]**2/cov[1][1])
22 val2 = sqrt(cov[1][1]-cov[1][0]**2/cov[0][0])
23 print val1, val2
24
25 X1 = zeros(n)
26 X2 = zeros(n)
27
28 #set the starting value of the Gibbs sampling algorithm
29 X1[0]=0
30 X2[0]=0
31
32 #run the Gibbs sampling algorithm for n steps
33 for i in xrange(1,n):
34     X1[i]=random.normal(sam1(X2[i-1]), val1)
35     X2[i]=random.normal(sam2(X1[i]), val2)
36
37 #Task 2
38 figure(0)
39 plot(X1[2900:],X2[2900:])
40 figure(1)
41 plot(X1, 'b')
42 figure(2)
43 plot(X2, 'r')
44 show()
45
46 #Task 3
47 #estimate the probability
48 prob = zeros(n)
49 prob[0]= (X1[0]>=0 and X2[0]>=0)
50
51 for i in xrange(1,n):
52     prob[i]= prob[i-1]+(X1[i]>=0 and X2[i]>=0)
53 Prob = prob/xrange(1,n+1)
54
55 print(Prob)
56
57 figure(3)
```

```
58 plot(Prob)
59 axis([-200,n+200,0.2,0.6])
60 show()
61
62 #Task 4
63 #set the new covariance parameter, and repeat steps above
64 cov = [[4,2.8],[2.8,4]]
```

## Model code in R

```
1 #Tasks 1 and 3
2 #set the mean parameter
3 mu <- c(0,0)
4 #set the covarince parameter
5 sigma <- matrix(c(4,1,1,4), nrow=2)
6
7 #Gibbs sampling
8 #set the desired sample size
9 n <- 3000
10 x <- matrix(nrow=n, ncol=2)
11
12 #set the starting value
13 cur.x <- c(0,0)
14
15 #run the Gibbs sampler; estimate  $P(X_1 \geq 0, X_2 \geq 0)$ 
16 prop <- rep(0, times=n)
17
18 sd1 <- sqrt(sigma[1,1] - sigma[1,2]^2/sigma[2,2])
19 sd2 <- sqrt(sigma[2,2] - sigma[1,2]^2/sigma[1,1])
20
21 for(i in 1:n){
22   mean1 <- mu[1] + sigma[1,2]/sigma[2,2] * (cur.x[2] - mu[2])
23   cur.x[1] <- rnorm(1, mean = mean1, sd = sd1)
24
25   mean2 <- mu[2] + sigma[1,2]/sigma[1,1] * (cur.x[1] - mu[1])
26   cur.x[2] <- rnorm(1, mean = mean2, sd = sd2)
27
28   if(i == 1){
29     prop[i] <- as.numeric((cur.x[1] >= 0) && (cur.x[2] >= 0))
30   }
31   else{
32     prop[i] <- as.numeric((cur.x[1] >= 0) && (cur.x[2] >= 0)) + prop[i-1]
33   }
34
35   x[i,] <- cur.x
36 }
37
38 prop <- prop/1:n
39
40 #Task 2
41 #plot the last 100 values, with subsequent values linked by a line
42 plot(x[901:1000,], type='b', xlab="X_1", ylab="X_2")
43
44 #look at sample plots of both variables
45 par(mfrow=c(2,1))
46 plot(x[,1], type="l", xlab="t", ylab="X_1")
47 plot(x[,2], type="l", xlab="t", ylab="X_2")
48
49 #look at the estimate of  $P(X_1 \geq 0, X_2 \geq 0)$ 
50 plot(prop, ylim=c(0.1,0.4), type='l', xlab="t", ylab="Estimate_of_P(X_1>=0,X_2>=0)")
51
52 #Task 4
53 #Now set the covarince parameter as follows, and repeat.
54 sigma <- matrix(c(4,2.8,2.8,2), nrow=2)
```