

## Computer Practical: Importance Sampling Model Answers

### Model code in Python

```
1 from scipy import *
2 from scipy import stats
3 import pylab
4 import sys
5 sys.path.append("/home/ludger/lib/python2.6/site-packages/")
6 import statistics
7
8 def gammaPDF(x, alpha, beta):
9     return stats.gamma(alpha).pdf(x*beta)*beta
10
11 def gammaSample(n, alpha, beta):
12     return stats.gamma(alpha).rvs(n)/beta
13
14 def kde(x, t, weights):
15     weights = weights / sum(weights) * len(weights)
16     return statistics.pdf(x, t, weights)
17
18 # Task 1
19
20 y = array([3, 6, 3, 5, 9, 14, 12, 11, 19, 18,
21           15, 4, 1, 6, 11, 21, 11, 3, 7, 18])
22
23 # Task 2
24
25 alpha = 0.1
26 beta = 0.1
27
28 alpha_post_1 = alpha + sum(y[0:5])
29 alpha_post_2 = alpha + sum(y[5:10])
30 beta_post = beta + 5
31 print(alpha_post_1)
32 print(alpha_post_2)
33 print(beta_post)
34
35 # Task 3
36
37 post_mean_1_labelled = alpha_post_1 / beta_post
38 post_mean_2_labelled = alpha_post_2 / beta_post
39 print(post_mean_1_labelled)
40 print(post_mean_2_labelled)
41
42 # Task 4
43
44 n = 10000
45 lambda_1 = gammaSample(n, alpha_post_1, beta_post)
46 lambda_2 = gammaSample(n, alpha_post_2, beta_post)
47 weights = zeros(n)
48 for i in xrange(n):
49     weights[i] = prod(0.5 * stats.poisson(lambda_1[i]).pmf(y[10:20]) +
50                    0.5 * stats.poisson(lambda_2[i]).pmf(y[10:20]))
51
52 # Task 5
53
54 post_mean_1_all = sum(lambda_1*weights) / sum(weights)
55 post_mean_2_all = sum(lambda_2*weights) / sum(weights)
56 print(post_mean_1_all)
57 print(post_mean_2_all)
```

```

58
59 # Task 6
60
61 t = linspace(0,20,1000)
62 lambda_1_density = kde(lambda_1, t, weights)
63 pylab.figure(1)
64 pylab.plot(t, lambda_1_density, 'r-')
65 pylab.plot(t, gammaPDF(t, alpha_post_1, beta_post), 'g:')
66 pylab.show()
67
68 lambda_2_density = kde(lambda_2, t, weights)
69 pylab.figure(2)
70 pylab.plot(t, lambda_2_density, 'r-')
71 pylab.plot(t, gammaPDF(t, alpha_post_2, beta_post), 'g:')
72 pylab.show()
73
74 # Task 7
75
76 allocations = zeros((n,10))
77 for i in xrange(n):
78     allocations[i,:] = stats.poisson(lambda_1[i]).pmf(y[10:20]) /
79         (stats.poisson(lambda_1[i]).pmf(y[10:20]) + stats.poisson(lambda_2[i]).pmf(y[10:20]))
80
81 prob_of_group_1 = zeros(10)
82 for i in range(10):
83     prob_of_group_1[i] = sum(allocations[:,i]*weights) / sum(weights)
84
85 print prob_of_group_1

```

## Model code in R

```
1 #Task 1
2
3 y <- c(3,6,3,5,9,14,12,11,19,18,15,4,1,6,11,21,11,3,7,18)
4
5 #Task 2
6
7 alpha <- 0.1
8 beta <- 0.1
9
10 alpha_post_1 = alpha + sum(y[1:5])
11 alpha_post_2 = alpha + sum(y[6:10])
12 beta_post = beta + 5
13 alpha_post_1
14 alpha_post_2
15 beta_post
16
17 #Task 3
18 #Notice that there is a difference between the parameterisation of the Gamma
19 #density in the practical, and that used in R; type help(rgamma) to see
20 #the details. Hence, for the R functions, we need to set the scale parameter
21 #to 1/beta.
22
23 n <- 10000
24 lambda_1 <- rgamma(n, shape=alpha_post_1, scale=1/beta_post)
25 lambda_2 <- rgamma(n, shape=alpha_post_2, scale=1/beta_post)
26
27 weights <- rep(NA, times=n)
28 for(i in 1:n){
29   weights[i] <- prod(0.5 * dpois(y[11:20], lambda_1[i]) + 0.5 * dpois(y[11:20], lambda_2[i]))
30 }
31
32 #Task 4
33
34 post_mean_1_labelled = alpha_post_1 / beta_post
35 post_mean_2_labelled = alpha_post_2 / beta_post
36 post_mean_1_labelled
37 post_mean_2_labelled
38
39 post_mean_1_all = sum(lambda_1 * weights) / sum(weights)
40 post_mean_2_all = sum(lambda_2 * weights) / sum(weights)
41 post_mean_1_all
42 post_mean_2_all
43
44 #Task 5
45 #Note that the function density must take as argument a vector of weights that sum to 1.
46
47 lambda_1_density = density(lambda_1, weights=weights/sum(weights))
48 lambda_2_density = density(lambda_2, weights=weights/sum(weights))
49
50 t <- seq(0, 20, length.out = 1000)
51
52 par(mfrow=c(1,2))
53 plot(lambda_1_density, main="Posterior density of lambda1")
54 lines(t, dgamma(t, shape=alpha_post_1, scale=1/beta_post), lty=2)
55 legend("topright",lty=c(1,2), legend=c("All data","Labelled data"))
56 plot(lambda_2_density, main="Posterior density of lambda2")
57 lines(t, dgamma(t, shape=alpha_post_2, scale=1/beta_post), lty=2)
58 legend("topright",lty=c(1,2), legend=c("All data","Labelled data"))
59
60 #Task 6
61
62 allocations <- matrix(NA, nrow=n, ncol=10)
63 for(i in 1:n){
64   allocations[i,] <- dpois(y[11:20], lambda_1[i]) /
65     (dpois(y[11:20], lambda_1[i]) + dpois(y[11:20], lambda_2[i]))
66 }
```

```
67 |  
68 | prob_of_group_1 = rep(NA, times=10)  
69 | for(i in 1:10){  
70 |   prob_of_group_1[i] = sum(allocations[,i] * weights) / sum(weights)  
71 | }  
72 | prob_of_group_1
```