

Computer Practical: Importance Sampling

In this computer practical, you can use either Python or R. There is a list of useful Python and R functions at the end of this handout.

In this practical you will program the importance sampling algorithm used in example 3.6.

Data

The data of example 3.6 is given in the table below:

Group	Count Y_i	Group	Count Y_i	Group	Count Y_i	Group	Count Y_i
1	3	2	14	*	15	*	21
1	6	2	12	*	4	*	11
1	3	2	11	*	1	*	3
1	5	2	19	*	6	*	7
1	9	2	18	*	11	*	18

Task 1. Create a vector (array) \mathbf{y} which contains the observations (y_1, \dots, y_{20}) .

Model

We assume that in each group Y_i is from a Poisson distribution with common mean λ_j , i.e.

$$\begin{aligned} Y_i &\sim \text{Poi}(\lambda_1) && \text{if the } i\text{-th observation is from group 1} \\ Y_i &\sim \text{Poi}(\lambda_2) && \text{if the } i\text{-th observation is from group 2} \end{aligned}$$

Our aim is inference about the two parameters λ_1 and λ_2 . We use a $\text{Gamma}(\alpha, \beta)$ distribution as prior distribution for both parameters, i.e.

$$f(\lambda_j) = \frac{1}{\Gamma(\alpha)} \lambda_j^{\alpha-1} \beta^\alpha \exp(-\beta \lambda_j).$$

We will use the prior parameters $\alpha = \beta = 0.1$.

Analysing the labelled data

We start with analysing the labelled data only. In the lectures we have seen that the posterior distributions of λ_1 and λ_2 are given by

$$\begin{aligned} f(\lambda_1 | y_1, \dots, y_5) &\propto \lambda_1^{\alpha + \sum_{i=1}^5 y_i} \exp(-(\beta + 5)\lambda_1) \\ f(\lambda_2 | y_6, \dots, y_{10}) &\propto \lambda_2^{\alpha + \sum_{i=6}^{10} y_i} \exp(-(\beta + 5)\lambda_2), \end{aligned}$$

thus

$$\lambda_1 | Y_1, \dots, Y_5 \sim \text{Gamma}(\alpha_1^{\text{post}}, \beta^{\text{post}}), \quad \lambda_2 | Y_6, \dots, Y_{10} \sim \text{Gamma}(\alpha_2^{\text{post}}, \beta^{\text{post}}),$$

with $\alpha_1^{\text{post}} = \alpha + \sum_{i=1}^5 y_i$, $\alpha_2^{\text{post}} = \alpha + \sum_{i=6}^{10} y_i$, and $\beta^{\text{post}} = \beta + 5$.

Task 2. Compute the parameters α_1^{post} , α_2^{post} , and β^{post} of the posterior distributions given the labelled data.

Analysing the entire data set

The likelihood of the entire data is given by:

$$f(y_1, \dots, y_{20} | \lambda_1, \lambda_2) = f(y_1, \dots, y_5 | \lambda_1) \cdot f(y_6, \dots, y_{10} | \lambda_2) \cdot f(y_{11}, \dots, y_{20} | \lambda_1, \lambda_2)$$

$$= \left(\prod_{i=1}^5 \frac{\exp(-\lambda_1) \lambda_1^{y_i}}{y_i!} \right) \cdot \left(\prod_{i=6}^{10} \frac{\exp(-\lambda_2) \lambda_2^{y_i}}{y_i!} \right) \cdot \left(\prod_{i=11}^{20} \left(\frac{1}{2} \cdot \frac{\exp(-\lambda_1) \lambda_1^{y_i}}{y_i!} + \frac{1}{2} \cdot \frac{\exp(-\lambda_2) \lambda_2^{y_i}}{y_i!} \right) \right)$$

In the lectures we have derived that the joint posterior distribution of λ_1 and λ_2 given all data is

$$f(\lambda_1, \lambda_2 | y_1, \dots, y_{20}) \propto f(\lambda_1 | y_1, \dots, y_5) f(\lambda_2 | y_6, \dots, y_{10}) \cdot \left(\prod_{i=11}^{20} \left(\frac{1}{2} \cdot \frac{\exp(-\lambda_1) \lambda_1^{y_i}}{y_i!} + \frac{1}{2} \cdot \frac{\exp(-\lambda_2) \lambda_2^{y_i}}{y_i!} \right) \right).$$

Unfortunately, I do not know how to sample from $f(\lambda_1, \lambda_2 | y_1, \dots, y_{20})$ directly, so we need to use importance sampling. We will use the posterior distributions obtained from the labelled data as instrumental distribution, i.e.

$$g(\lambda_1, \lambda_2) = f(\lambda_1 | y_1, \dots, y_5) f(\lambda_2 | y_6, \dots, y_{10}).$$

The corresponding weights are given by

$$w(\lambda_1, \lambda_2) = \frac{f(\lambda_1, \lambda_2 | y_1, \dots, y_{20})}{g(\lambda_1, \lambda_2)} = \prod_{i=11}^{20} \left(\frac{1}{2} \cdot \frac{\exp(-\lambda_1) \lambda_1^{y_i}}{y_i!} + \frac{1}{2} \cdot \frac{\exp(-\lambda_2) \lambda_2^{y_i}}{y_i!} \right).$$

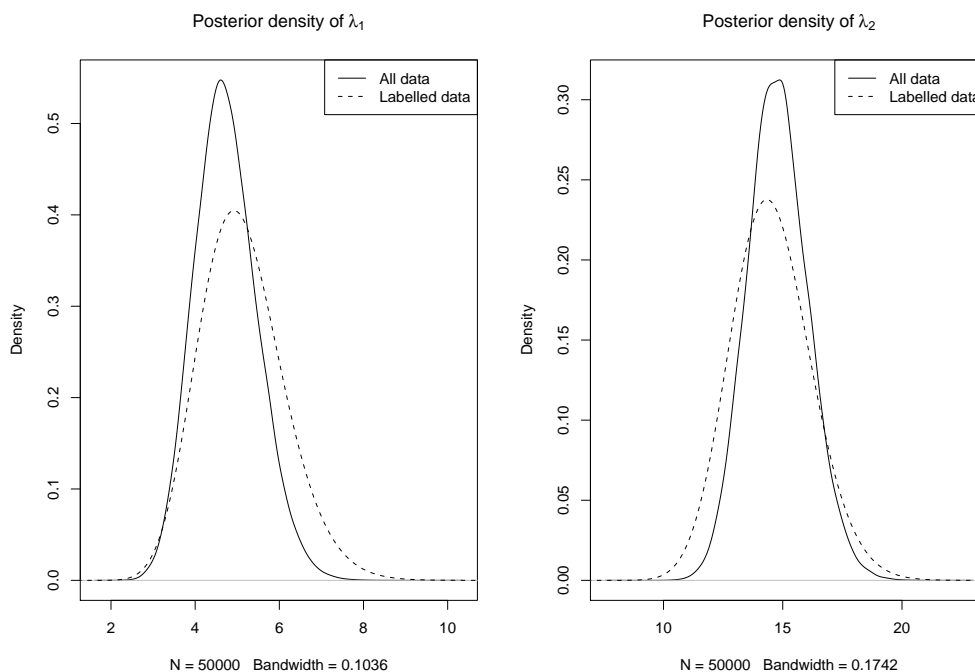
Task 3. Draw a weighted sample of size $n = 10000$ from $f(\lambda_1, \lambda_2 | y_1, \dots, y_{20})$ using $g(\lambda_1, \lambda_2)$ as instrumental distribution, i.e. draw

$$\lambda_1 \sim \text{Gamma}(\alpha_1^{\text{post}}, \beta^{\text{post}}), \quad \lambda_2 \sim \text{Gamma}(\alpha_2^{\text{post}}, \beta^{\text{post}}),$$

and compute $w(\lambda_1, \lambda_2)$.

Task 4. Compute the self-normalised estimate of the posterior means of λ_1 and λ_2 and compare them to the posterior means obtained in task 2.

Task 5. Compute a weighted kernel density estimate of the weighted sample of λ_1 and λ_2 and compare it to the posterior distribution obtained in task 2. Your plots should look similar to the ones shown below.



Estimating the allocations

One might be interested in the question from which group the unlabelled observations are from.

Given λ_1, λ_2 we have for $i > 10$ that

$$\begin{aligned} \mathbb{P}(\text{Observation } i \text{ from group 1} | y_i, \lambda_1, \lambda_2) &= \frac{\mathbb{P}(\text{Observation } i \text{ from group 1}, y_i | \lambda_1, \lambda_2)}{\mathbb{P}(y_i | \lambda_1, \lambda_2)} \\ &= \frac{\mathbb{P}(\text{Observation } i \text{ from group 1}, y_i | \lambda_1, \lambda_2)}{\mathbb{P}(\text{Observation } i \text{ from group 1}, y_i | \lambda_1, \lambda_2) + \mathbb{P}(\text{Observation } i \text{ from group 2}, y_i | \lambda_1, \lambda_2)} \\ &= \frac{\frac{1}{2} \cdot \frac{\exp(-\lambda_1)\lambda_1^{y_i}}{y_i!}}{\frac{1}{2} \cdot \frac{\exp(-\lambda_1)\lambda_1^{y_i}}{y_i!} + \frac{1}{2} \cdot \frac{\exp(-\lambda_2)\lambda_2^{y_i}}{y_i!}} = \frac{\exp(-\lambda_1)\lambda_1^{y_i}}{\exp(-\lambda_1)\lambda_1^{y_i} + \exp(-\lambda_2)\lambda_2^{y_i}} \end{aligned}$$

We shall call this quantity $h(\lambda_1, \lambda_2)$. Then

$$\mathbb{P}(\text{Observation } i \text{ from group 1} | y_i) = \mathbb{E}(h(\lambda_1, \lambda_2)),$$

which is an expectation, which we can estimate using importance sampling.

Task 6. Compute the self-normalised importance sampling estimate of the posterior probability that the i -th observation ($i = 11, \dots, 20$) is from group 1.

Useful Python functions

The functions listed below are available if you have imported the following libraries:

```
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
```

The following Python functions might be of use:

Poisson density You can evaluate the probability mass function (density) $f_{(\lambda)}(y) = \frac{\exp(-\lambda)\lambda^y}{y!}$ of the Poisson distribution with parameter λ using the function `stats.poisson(lambda).pmf(y)`.

Gamma density Use the function below to evaluate the density $f_{(\alpha,\beta)}$ of the Gamma(α, β) distribution.

```
7 def gammaPDF(x, alpha, beta):
8     return stats.gamma(alpha).pdf(x*beta)*beta
```

Sampling from the gamma distribution Use the function below to sample from the Gamma(α, β) distribution.

```
9 def gammaSample(n, alpha, beta):
10    return stats.gamma(alpha).rvs(n)/beta
```

Kernel density estimation Use the function below to compute a weighted kernel density estimate of the data in `x` evaluated at `t`.

```
11 def kde(x, t, weights):
12     weights = weights / sum(weights) * len(weights)
13     return statistics.pdf(x, t, weights)
```

Example:

```
14 n=100
15 normal_sample = stats.norm().rvs(n)           # Generate sample from N(0,1)
16 weights = ones(n)
17 t = linspace(-3, 3, 100)                     # Grid at which density will be evaluated
18 density_estimate=kde(normal_sample, t, weights) # Compute density estimate
19 pylab.figure()
20 pylab.plot(t, density_estimate)               # Plot density estimate
21 pylab.show()
```

Useful R functions

The following Python functions might be of use:

Poisson density The function `dpois(x, lambda)` evaluates the probability mass function (density) $f_{(\lambda)}(x)$ of the $\text{Poi}(\lambda)$ distribution.

Gamma density The function `dgamma(x, alpha, beta)` evaluates the density $f_{(\alpha,\beta)}(x)$ of the $\text{Gamma}(\alpha,\beta)$ distribution.

Sampling from the gamma distribution The function `rgamma(n, alpha, beta)` draws a sample of size n from the $\text{Gamma}(\alpha,\beta)$ distribution.

Kernel density estimation The function `density(x, weights=weights)` compute a weighted kernel density estimate of the data in x .

```
1 n <- 100
2 normal.sample <- rnorm(n)           # Generate sample from N(0,1)
3 weights <- rep(1,n)
4 density.estimate <- density(normal.sample, weights=weights)
5                                     # Compute density estimate
6 plot(density.estimate)             # Plot density estimate
```