MATH5835M Practical Solutions

This text gives an example solution for the MATH5835M practical.

- Each question is worth 20 marks, and there are also 20 marks for presentation, for a total of $4 \times 20 + 20 = 100$ marks.

Overall marking criteria:

- Are all methods used and results explained?
- Do the explanations make sense?
- Is the presentation concise and well-structured?
- Is all relevant R code included in the report? Is all R code in the report relevant?
- Is R used competently?

Marking criteria for individual questions are listed at the end of each question.

Solutions

Task 1

The first task is to import the data and to give a short description of the data. After downloading the file `test-pre3.csv.gz` from from

`https://www.seehuhn.de/maths/letters/`

we can use the following code (from the web page) to import the data:

```
test <- read.csv("test-pre3.csv.gz")
im <- as.matrix(test[,-1])
lab <- as.factor(LETTERS[test[,1]+1])
rm(test)
```

We can see that the data set contains 5835 samples of size 1728 each: according to the web page, each of the samples corresponds to a 54×32 image. The column names `r1c1`, `r1c2` etc. indicate that the first few values corrspond to row 1 of the picture (`r` stands for "row", `c` stands for "column"):

```
str(im)
```

```
##  int [1:5825, 1:1728] 255 255 255 255 255 255 255 255 255 255 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : NULL
##   ..$ : chr [1:1728] "r1c1" "r1c2" "r1c3" "r1c4" ...
```

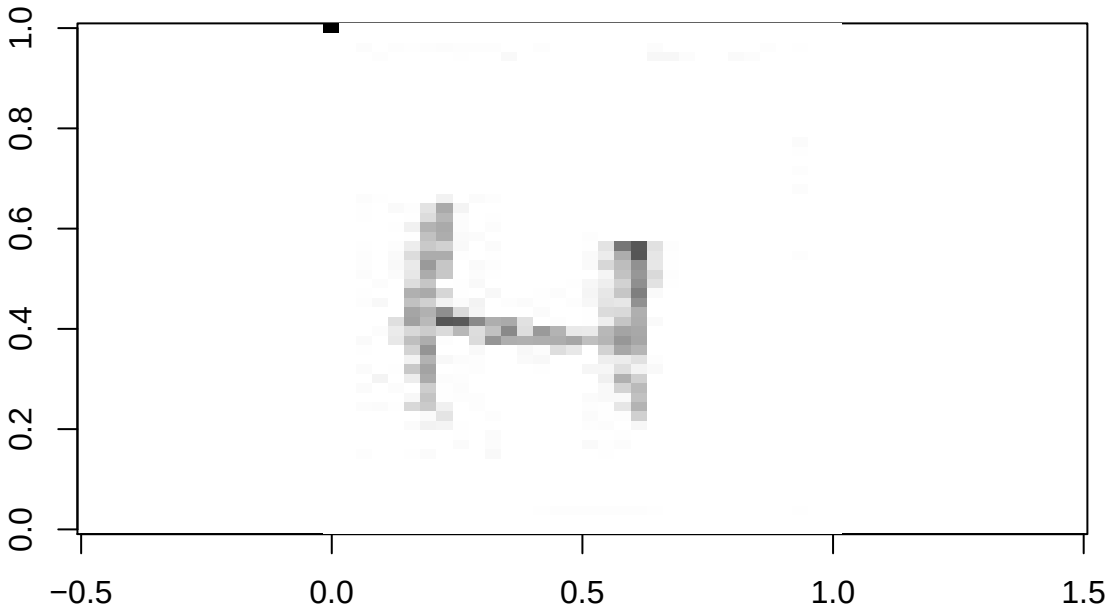
The provided labels indicate which letter each image corresponds to:

```
str(lab)
```

```
##  Factor w/ 26 levels "A","B","C","D",..: 8 25 26 17 25 26 1 17 20 26 ...
```

As a test we consider the first letter. Inspecting `lab[1]`, we see that this letter is mean to be an H. Using the code similar to that from the web page, we can show the corresponding image. We simultaneously verify that the value 0 corresponds to black, and the the first value in each row corresponds to the top-left pixel, but setting the corresponding value in the image data to 0.

```
pic <- matrix(im[1,], 32, 54)
pic[1, 1] <- 0
image(pic[,54:1], asp=1, col = gray((0:255)/255), xlab="", ylab="")
```



This seems indeed to represent an H, so everything is consistent. As expected the top-left pixel is shown black, because we set `im[1, 1]` to 0.

Criteria for marking:

- Does the report convince the reader that the data was imported correctly?
- Is there an explanation of how the numbers in R correspond to images?
- Is there an explanation of the role of the labels provided as part of the data?
- Is the R code shown, understandable and correct?
- Is there any evidence that the R code from the web page has been understood, rather than just copied?

Task 2

In this task we have to compute the standard deviation of the pixels in each image. This gives a measure of how much variation there is within each image:

```
s <- apply(im, 1, sd)
str(s)
```

```
##  num [1:5825] 19.2 49.1 46 61.3 68.7 ...
```

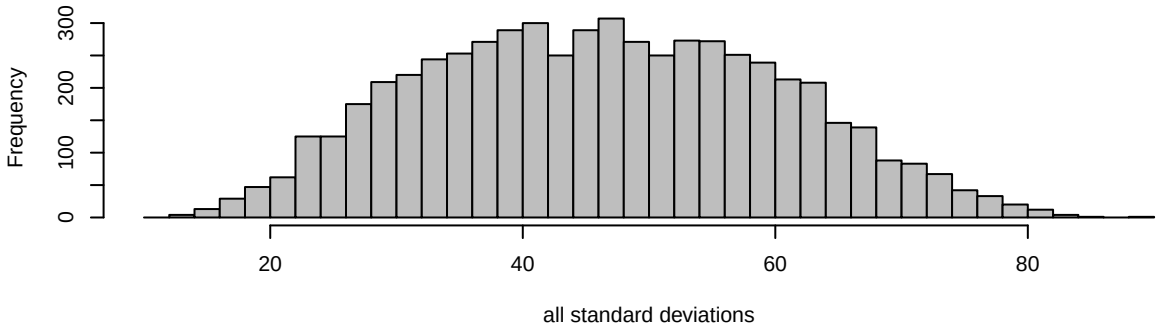
```
s[1]
```

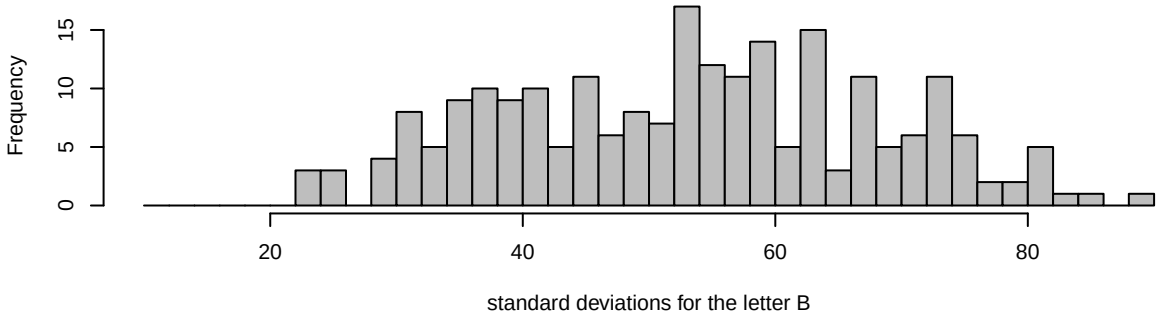
```
## [1] 19.16695
```

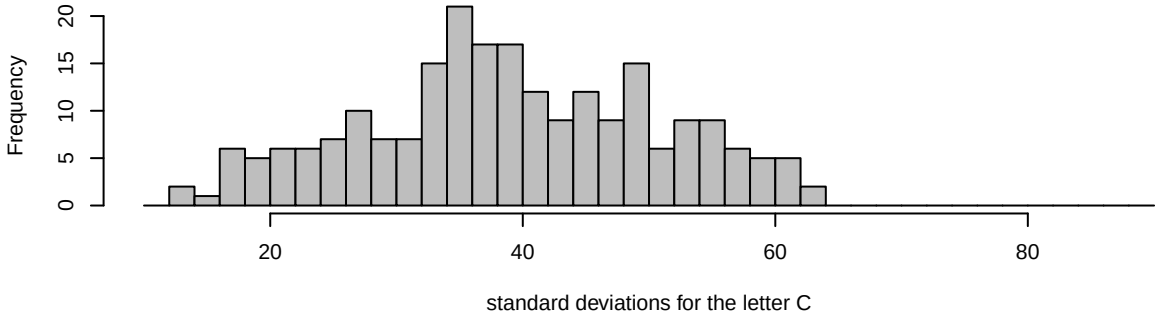
The result shows that the vector of standard deviations has the correct length 5825, and that the first standard deviation equals the value given in the hint.

```
par(mfrow=c(3, 1))
b = seq(10, 90, length.out = 41)
hist(s, breaks=b, main=NULL, col="grey",
```

```
     xlab="all standard deviations")
hist(s[which(lab == "B")], breaks=b, main=NULL, col="grey",
     xlab="standard deviations for the letter B")
hist(s[which(lab == "C")], breaks=b, main=NULL, col="grey",
     xlab="standard deviations for the letter C")
```







We can see that standard deviations for the letter "B" are towards the higher end of standard deviations, whereas the values for "C" are towards the lower end of the range. Maybe this is a consequence of "B" needing more ink to write than "C"?

Marking criteria:

- Does the report convince the reader that the standard deviations have been computed correctly?
- Has a reasonable variable name been chosen for the standard deviations (e.g. s to match the s_i on the question sheet).

- Has care been taken to make the histograms look nice?
- Are the samples for letters "B" and "C" selected correctly?
- Is there some discussion of the fact that "B" has higher standard deviations than "C"?

Task 3

Here we have to find the posterior distribution of the mean μ_L of the standard deviation for instances of the letter L, using some Bayesian model.

The prior distribution for μ_L is the uniform distribution $\mathcal{U}[38, 55]$, with density

$$p(\mu) = \frac{1}{55-38} 1_{[38,55]}(\mu) = \frac{1}{17} 1_{[38,55]}(\mu)$$

If we denote the standard deviations corresponding to the letter L by $s = (s_1, \ldots, s_{n_L})$, the likelihood is

$$\begin{aligned} p\big(s \mid \mu\big) &= \prod_{i=1}^{n_L} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\Big(-\frac{(s_i-\mu)^2}{2\sigma^2}\Big) \\ &= (2\pi\sigma^2)^{-n_L/2} \exp\Big(-\frac{1}{2\sigma^2}\sum_{i=1}^{n_L}(s_i-\mu)^2\Big) \end{aligned}$$

where $\sigma^2 = 13^2$ is the given variance. Using Bayes' formula, we can find the posterior distribution, up to constants, as

$$\begin{aligned} p\big(\mu \mid s\big) &= \frac{p\big(s \mid \mu\big)p(\mu)}{p(s)} \\ &\propto \exp\Big(-\frac{1}{2\sigma^2}\sum_{i=1}^{n_L}(s_i-\mu)^2\Big) 1_{[38,55]}(\mu) \end{aligned}$$

Or aim is to find a proposal density, which can be used to generate samples from the posterior, using rejection sampling. To find an appropriate proposal density, we rewrite the exponential term to be proportional to a normal distribution:

$$\begin{aligned} p\big(\mu \mid s\big) &\propto \exp\Big(-\frac{1}{2\sigma^2}\big(n_L\mu^2 - 2\mu\sum_{i=1}^{n_L} s_i + \sum_{i=1}^{n_L} s_i^2\big)\Big) 1_{[38,55]}(\mu) \\ &\propto \exp\Big(-\frac{1}{2\sigma^2/n_L}\big(\mu^2 - 2\mu\frac{1}{n_L}\sum_{i=1}^{n_L} s_i\big)\Big) 1_{[38,55]}(\mu) \\ &\propto \frac{1}{\sqrt{2\pi\sigma^2/n_L}} \exp\Big(-\frac{\big(\mu - \frac{1}{n_L}\sum_{i=1}^{n_L} s_i\big)^2}{2\sigma^2/n_L}\Big) 1_{[38,55]}(\mu) \\ &=: \varphi(\mu). \end{aligned}$$

By writing the target density φ in this form, we see that a good proposal density will be the density of a $\mathcal{N}(\frac{1}{n_L}\sum_{i=1}^{n_L} s_i, \frac{1}{n_L}\sigma^2)$-distribution, *i.e.*

$$\psi(x) = \frac{1}{\sqrt{2\pi\sigma^2/n_L}} \exp\Big(-\frac{\big(\mu - \frac{1}{n_L}\sum_{i=1}^{n_L} s_i\big)^2}{2\sigma^2/n_L}\Big).$$

For this choice of φ and ψ we have $\varphi(\mu) \leq c\psi(\mu)$ where $c = 1$. Thus, we get the following algorithm:

- Generate $\mu \sim \mathcal{N}(\frac{1}{n_L}\sum_{i=1}^{n_L} s_i, \frac{1}{n_L}\sigma^2)$

- Generate $U \sim \mathcal{U}[0,1]$
- Accept μ, iff $c\psi(\mu)U \leq \varphi(\mu)$, i.e. if $\mu \in [38, 55]$.

Marking criteria:

- Has the target density been found correctly?
- Has a feasible proposal density been found?
- Has the constant c been found, and does the report convince the reader that $\varphi(\mu) \leq c\psi(\mu)$ holds for all μ?
- Does the report explain the resulting rejection sampling algorithm?

Task 4

Here we have to implement the rejection sampling algorithm from task 3. We start, by gathering the required values n_L and $\frac{1}{n_L}\sum_{i=1}^{n_L} s_i$ for all letters:

```
nL <- integer(26)
sL.mean <- numeric(26)
for (i in 1:26) {
    rows <- which(lab == LETTERS[i])
    nL[i] <- length(rows)
    sL.mean[i] <- mean(s[rows])
}
posterior <- data.frame(n=nL, s.mean=sL.mean, row.names = LETTERS)
posterior
```

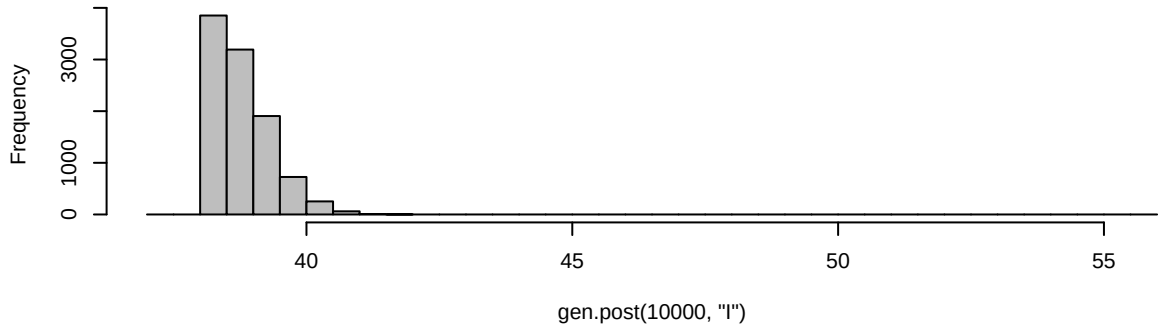
```
##     n   s.mean
## A 226 52.77319
## B 226 53.29567
## C 226 38.98246
## D 226 48.94556
## E 226 47.43154
## F 226 45.17012
## G 225 48.74648
## H 226 47.41852
## I 225 38.18254
## J 226 43.93489
## K 226 47.10197
## L 226 38.87120
## M 226 51.84142
## N 226 48.92815
## O 226 42.63989
## P 222 46.74690
## Q 222 52.48538
## R 222 51.87018
## S 222 41.84402
## T 222 41.06984
## U 222 44.79789
## V 221 43.14124
## W 221 49.17373
## X 221 43.79143
## Y 220 44.22220
## Z 222 49.78000
```

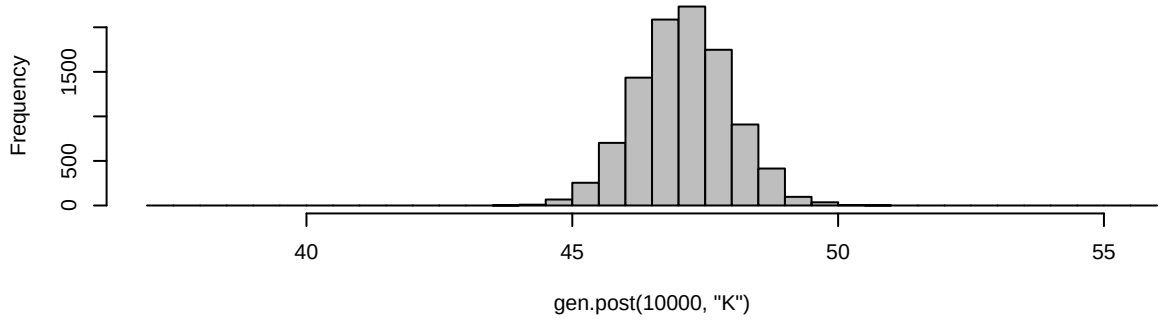
Using these values, we can now generate the samples. To be able to experiment with different letters more easily, we write a function to generate the sample:

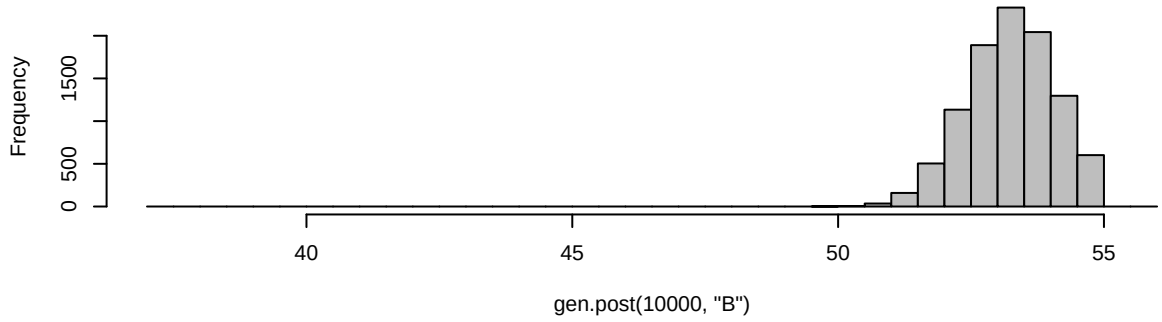
```
gen.post <- function(N, letter) {
    nL <- posterior[letter,]$n
    sL.mean <- posterior[letter,]$s.mean

    res <- numeric(N)
    done <- 0
    while (done < N) {
        m <- rnorm(1, sL.mean, 13 / sqrt(nL))
        if (m >= 38 && m <= 55) {
            done <- done + 1
            res[done] <- m
        }
    }
    res
}
```

To test the function, we produce histograms of posterior samples for “I”, “K” and “B” (top to bottom). These letters are chosen, because “I” and “B” have the lowest and highest mean standard deviation, respectively, while “K” falls into the middle of the range.

```
par(mfrow=c(3, 1))
hist(gen.post(10000, "I"), breaks=seq(37, 56, l=39), main=NULL, col="grey")
hist(gen.post(10000, "K"), breaks=seq(37, 56, l=39), main=NULL, col="grey")
hist(gen.post(10000, "B"), breaks=seq(37, 56, l=39), main=NULL, col="grey")
```







Using `gen.post()`, we can now generate samples for Monte Carlo estimates. In your report, you need to only consider a few letters. Here I make a table of all results, for the help with marking:

```
N <- 1e5
post.mean <- numeric(26)
post.mean.RMSE <- numeric(26)
post.var <- numeric(26)
for (i in 1:26) {
    mu <- gen.post(N, LETTERS[i])
    post.mean[i] <- mean(mu)
    post.mean.RMSE[i] <- sd(mu) / sqrt(N)
    post.var[i] <- var(mu)
}
results <- data.frame(post.mean = post.mean, post.var = post.var,
                      post.mean.RMSE = post.mean.RMSE,
```

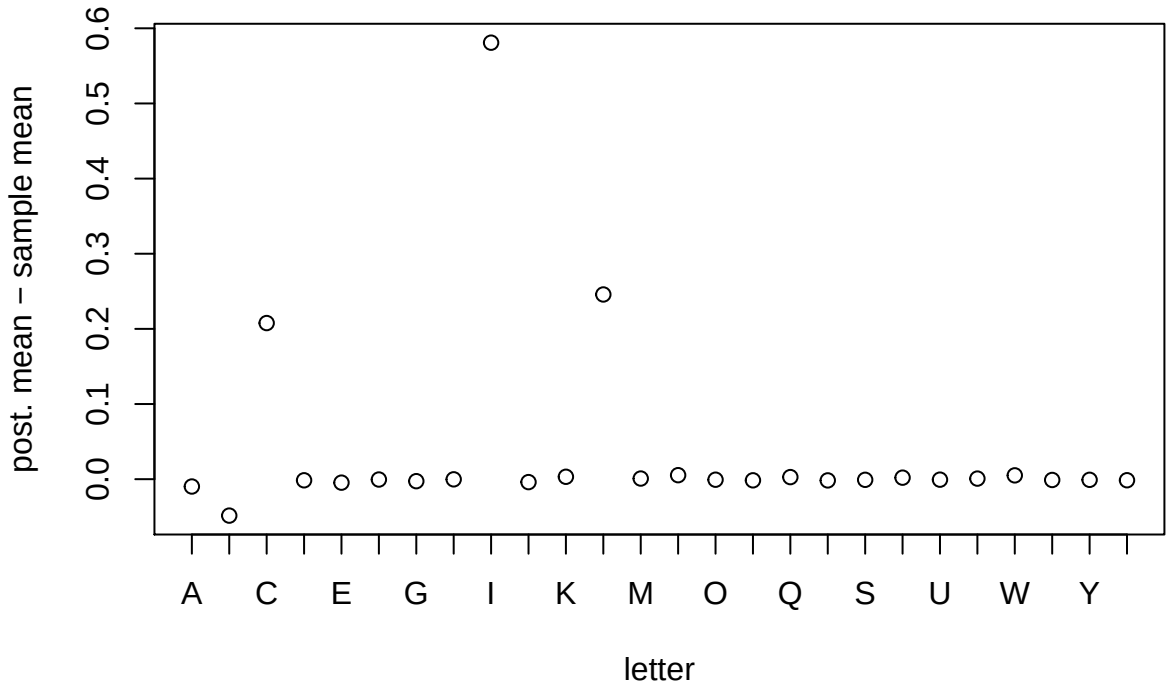
```
                      sample.mean = posterior$s.mean,
                      row.names = LETTERS)
results
```

```
##   post.mean  post.var post.mean.RMSE sample.mean
## A  52.76345 0.7187312    0.002680916    52.77319
## B  53.24720 0.6601801    0.002569397    53.29567
## C  39.19023 0.5004900    0.002237163    38.98246
## D  48.94405 0.7506250    0.002739754    48.94556
## E  47.42694 0.7499811    0.002738578    47.43154
## F  45.16967 0.7459982    0.002731297    45.17012
## G  48.74383 0.7510585    0.002740545    48.74648
## H  47.41832 0.7445465    0.002728638    47.41852
## I  38.76340 0.3088426    0.001757392    38.18254
## J  43.93097 0.7438062    0.002727281    43.93489
## K  47.10516 0.7516856    0.002741689    47.10197
## L  39.11698 0.4786779    0.002187871    38.87120
## M  51.84226 0.7475293    0.002734098    51.84142
## N  48.93346 0.7475965    0.002734221    48.92815
## O  42.63917 0.7461833    0.002731636    42.63989
## P  46.74538 0.7593192    0.002755575    46.74690
## Q  52.48813 0.7462125    0.002731689    52.48538
## R  51.86855 0.7557281    0.002749051    51.87018
## S  41.84317 0.7563940    0.002750262    41.84402
## T  41.07186 0.7581684    0.002753486    41.06984
## U  44.79724 0.7617465    0.002759976    44.79789
## V  43.14202 0.7651858    0.002766199    43.14124
## W  49.17880 0.7715461    0.002777672    49.17373
## X  43.79037 0.7599078    0.002756642    43.79143
## Y  44.22133 0.7638933    0.002763862    44.22220
## Z  49.77858 0.7570437    0.002751443    49.78000
```

The table shows all the results. The column `post.mean.RMSE` shows that the sample size N is large enough, by far, that the posterior mean is estimated accurately. A more thorough analysis could also take accuracy of the posterior variance estimate into account. Producing the table takes a few seconds on my laptop, so there is scope to increase N, if needed.

The table, and the plot below, shows that the Bayesian estimate for the mean differs clearly from the classical estimate (*i.e.* from the sample mean) only for the letters "C", "I" and "L". There is also a small deviation for the letter "B":

```
plot(results$post.mean - results$sample.mean, xaxt="n",
     xlab="letter", ylab="post. mean - sample mean")
axis(1, at=1:26, labels=LETTERS)
```



The table shows the results.

Marking criteria:

- Are the Monte Carlo samples generated correctly?
- Are the posterior mean and variance for at least two different letters estimated?
- Is at least one of the letters considered close enough to the edge of the prior interval, that the Bayesian estimate differs from the classical estimate?
- Is the Monte Carlo sample size chosen wisely?