Homework 2

Davide Berasi

2023-05-27

Exercise 1: Estimating a log-odds with a normal prior

Suppose Y has a binomial distribution with parameters n and p, and we are interesting in the log-odds value $\theta = \text{logit}(p) = \log(p/(1-p))$. Our prior for θ is that $\theta \sim N(\mu, \sigma)$ (parametrized as mean and standard deviation). It follows that the posterior density of θ is given, up to a proportional constant, by

$$g(\theta|y) \propto \frac{\exp(y\theta)}{(1+\exp(\theta))^n} \exp\left[-\frac{(\theta-\mu)^2}{2\sigma^2}\right]$$

More concretely, suppose we are interested in learning about the probability that a special coin lands heads when tossed. A priori we believe that the coin is fair, so we assign θ an $N(0, .25)$ prior. We toss the coin $n = 5$ times and obtain $y = 5$ heads.

> - Using the prior density as a proposal density, design an Accept-Reject algorithm for sampling from the posterior distribution. Using simulated draws from your algorithm, approximate the probability that the coin is biased toward heads.

Let $q \sim N(0, 25)$ be our prior distribution and let $\bar{g}$ be the kernel of our target distribution, that is:

$$g(\theta|y) \propto \frac{\exp(y\theta)}{(1+\exp(\theta))^n} \exp\left[-\frac{(\theta-\mu)^2}{2\sigma^2}\right] =: \bar{g}$$

In order to implement the accept-reject sampling algorithm we need a constant M such that $\bar{g} \leq Mq$. Since $y \leq n$, we have that:

$$\bar{g} = \left(\frac{\exp(\theta)}{1+\exp(\theta)}\right)^y \frac{1}{(1+\exp(\theta))^{n-y}} \exp\left[-\frac{(\theta-\mu)^2}{2\sigma^2}\right] \leq 1 \cdot \exp\left[-\frac{(\theta-\mu)^2}{2\sigma^2}\right] = \sqrt{2\pi}\sigma \cdot q$$

Thus we can take $M = \sqrt{2\pi}\sigma \approx$ 0.6266571. Here is the implementation of the algorithm:

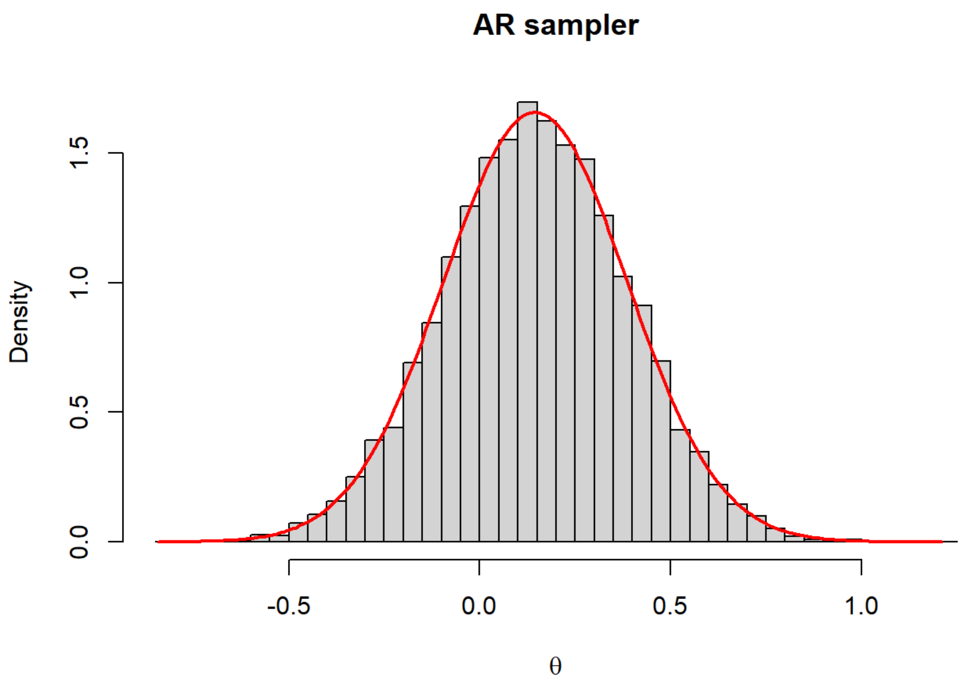
```
g_bar <- function(t, y=5){
  return ( (exp(y*t -((t/0.25)**2) / 2 ) / (1+exp(t))**5) )
}

sampler_AR <- function(N){
  # returns N samples from our posterior distribution g.
  M <- sqrt(2*pi)*0.25
  samples <- rep(0, N)
  num_samples <- 0
  while (num_samples < N) {
    t <- rnorm(1, mean=0, sd=0.25)
    u <- runif(1)
    if (u * M*dnorm(t, mean=0, sd=0.25) < g_bar(t)){
      num_samples <- num_samples + 1
      samples[num_samples] <- t
    }
  }
  return(samples)
}
```

To visualize how good the algorithm is, we can compare the histogram of the samples with the posterior density.

```
N <- 10000
g_AR <-  sampler_AR(N)
hist(g_AR, main = 'AR sampler', xlab=TeX(r"($\theta$)"), breaks = 50, freq = FALSE)

# We approximate the normalizing constant by integrating.
norm_const <- integrate(function(t) g_bar(t,5),-Inf, Inf)$value
tt <- seq(min(g_AR), max(g_AR), 0.01)
g_tt <- g_bar(tt) / norm_const
lines(tt, g_tt, col='red', lwd=2)
```



Given N samples $(\theta_1, \ldots, \theta_N)$ from the posterior distribution, a good approximation of the probability that the coin is biased towards head is:

$$\mathbf{P}(\theta > 0|Y = 5) \approx \frac{1}{n} \sum_{i=1}^{N} \mathbf{1}(\theta_i > 0)$$

Using $N =$ 10^{4} samples from the accept-reject algorithm, our approximation is:

```
sum(g_AR > 0) / N
```

```
## [1] 0.7291
```

We can also approximate a 95% confidence interval for θ with $[g_{0.025}, g_{0.975}]$, where $g_{0.025}, g_{0.975}$ are the 0.025 and 0.975 quantile for our sample.

```
quantile(g_AR, c(0.025, 0.975))
```

```
##       2.5%      97.5%
## -0.3271540  0.6115432
```

> - Using the prior density as a proposal density, simulate from the posterior distribution using a Sampling Importance Resampling (SIR) algorithm. Approximate the probability that the coin is biased toward heads.

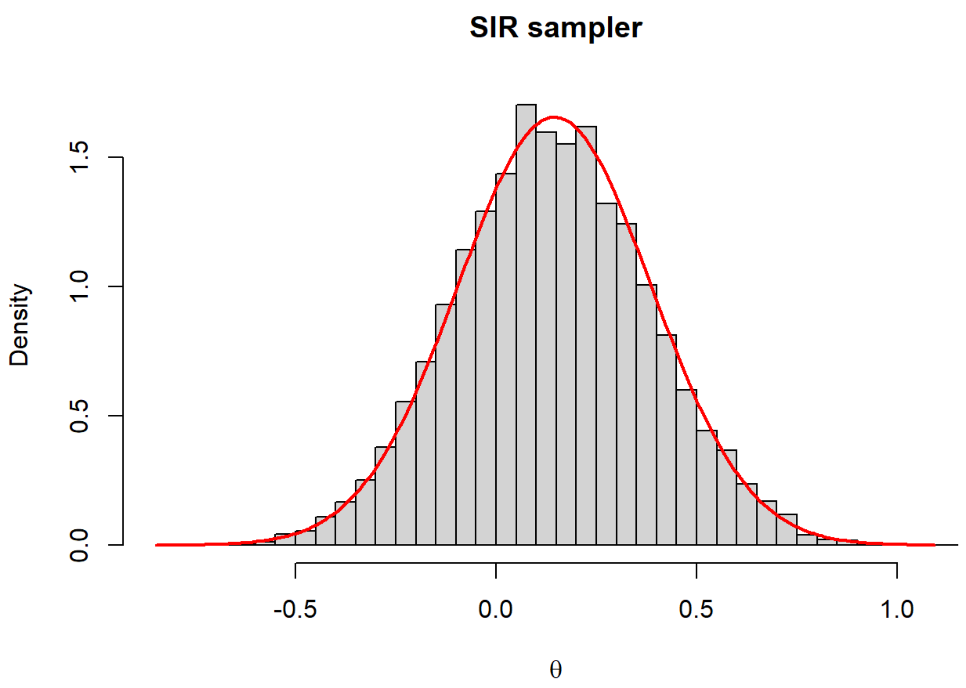
The SIR sampler is very easy to implement:

```
sampler_SIR <- function(N){
  # Sampling
  t <- rnorm(N, mean=0, sd=0.25)
  # Importance
  w <- g_bar(t) / dnorm(t, mean=0, sd=0.25)
  w <- w / sum(w)
  # Resampling
  t <- sample(t, size=N, replace=TRUE, prob=w)
return(t)
}
```

As before we can visualize how good the SIR algorithm is.

```
g_SIR <-  sampler_AR(N)
hist(g_SIR, main = 'SIR sampler', xlab=TeX(r"($\theta$)"), breaks = 50, freq = FALSE)

tt <- seq(min(g_SIR), max(g_SIR), 0.01)
g_tt <- g_bar(tt) / norm_const
lines(tt, g_tt, col='red', lwd=2)
```



Using the SIR sampler, the approximation of the probability that the coin is biased towards head is:

```
g_SIR <- sampler_SIR(N)
sum(g_SIR > 0) / N
```

```
## [1] 0.7216
```

In this case the approximation for a 95% confidence interval for θ is:

```
quantile(g_SIR, c(0.025, 0.975))
```

```
##       2.5%      97.5%
## -0.3323399  0.6168812
```

- Use a Laplace approximation to estimate the probability that the coin is biased toward heads.

In the Laplace approximation we approximate our posterior g with the "best fitting" normal $\mathcal{N}(m, v)$. Its mean is the mode m of g. We can compute it numerically.

```
# To compute the maximum we can use the function optimize.
m <- optimize(g_bar, c(0, 0.5), maximum = TRUE)$maximum
m
```

```
## [1] 0.1449478
```

For the standard deviation v we know that

$$-\frac{1}{v^2} = \frac{d^2}{d\theta^2}\log(g(\theta))\Big|_{\theta=m} = -n\frac{\exp(m)}{(1+\exp(m))^2} - \frac{1}{\sigma^2}$$

Thus we can approximate it:

```
d2_log_g <- -5*exp(m) / (1 + exp(m))^2 - 1/0.25^2
v <- sqrt(-1/d2_log_g)
v
```

```
## [1] 0.2408174
```

With Laplace approximation, the probability that the coin is biased towards head is:

```
pnorm(0, mean=m, sd=v, lower.tail=FALSE)
```

```
## [1] 0.7263794
```

In this case the approximation for a 95% confidence interval for θ is:

```
qnorm(c(0.025, 0.975), mean=m, sd=v)
```

```
## [1] -0.3270456  0.6169412
```

Exercise 2: Genetic linkage model

Suppose 197 animals are distributed into four categories with the following frequencies

Category	Frequency
1	125
2	18
3	20
4	34

Assume that the probabilities of the four categories are given by the vector

$$\left(\frac{1}{2}+\frac{\theta}{4}, \frac{1}{4}(1-\theta), \frac{1}{4}(1-\theta), \frac{\theta}{4}\right) \ ,$$

where θ is an unknown parameter between 0 and 1. If θ is assigned a uniform prior, then the posterior density of θ is given by

$$h(\theta|\text{data}) \propto \left(\frac{1}{2}+\frac{\theta}{4}\right)^{125}\left(\frac{1}{4}(1-\theta)\right)^{18}\left(\frac{1}{4}(1-\theta)\right)^{20}\left(\frac{\theta}{4}\right)^{34} \ ,$$

where $0<\theta<1$.

- If θ is transformed to the real-valued logit $\eta = \log(\theta/(1-\theta))$, then calculate the posterior density of η.

The distribution of $\eta|\text{data}$ is $g(\eta|\text{data}) = h(\theta|\text{data})\left|\frac{d}{d\eta}\theta(\eta)\right|$. Since

$$\theta(\eta) = \frac{\exp(\eta)}{1+\exp(\eta)} \qquad \text{and} \qquad \left|\frac{d}{d\eta}\theta(\eta)\right| = \frac{\exp(\eta)}{(1+\exp(\eta))^2}$$

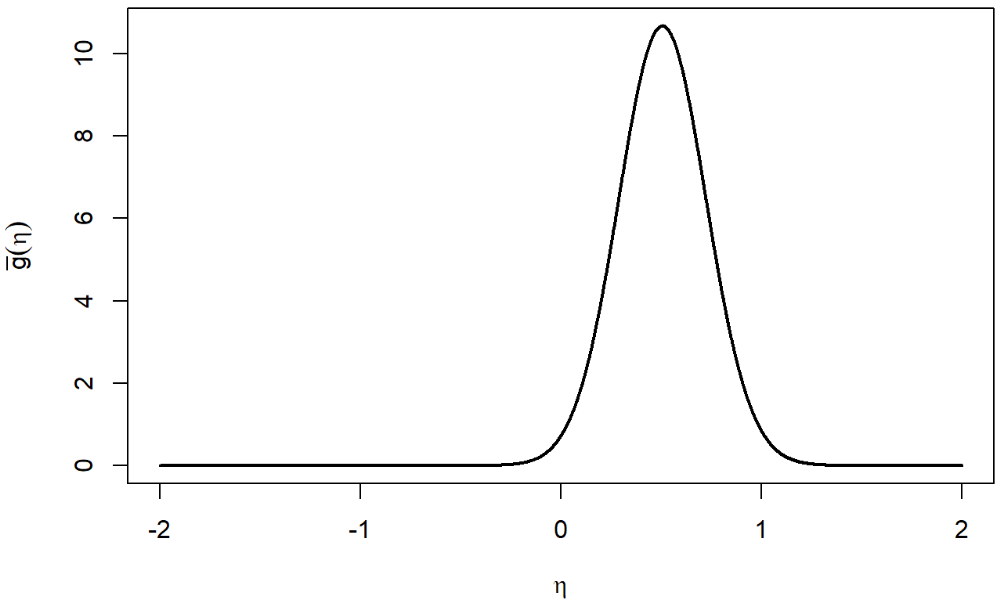
the posterior of η is

$$\begin{aligned}
g(\eta|\text{data}) &\propto \left(\frac{1}{2}+\frac{\exp(\eta)}{4(1+\exp(\eta))}\right)^{125}\left(\frac{1}{4}\left(1-\frac{\exp(\eta)}{1+\exp(\eta)}\right)\right)^{18}\left(\frac{1}{4}\left(1-\frac{\exp(\eta)}{1+\exp(\eta)}\right)\right)^{20}\left(\frac{\exp(\eta)}{4(1+\exp(\eta))}\right)^{34}\frac{\exp(\eta)}{(1+\exp(\eta))^2} \\
&= \left(\frac{2+3\exp(\eta)}{4(1+\exp(\eta))}\right)^{125}\left(\frac{1}{4(1+\exp(\eta))}\right)^{18}\left(\frac{1}{4(1+\exp(\eta))}\right)^{20}\left(\frac{\exp(\eta)}{4(1+\exp(\eta))}\right)^{34}\frac{\exp(\eta)}{(1+\exp(\eta))^2} \\
&\propto \frac{(2+3\exp(\eta))^{125}\exp(35\eta)}{(1+\exp(\eta))^{199}} =: \bar{g}(\eta)
\end{aligned}$$

We can plot it:

```
# The kernel defined as above takes very small values. In order to avoid problems with the float precision, I multiply it by 10^91.
g_bar2 <- function(e){
log_num <- 125*log(2+3*exp(e))+35*e + 91*log(10)
log_den <- 197*log(4)+199*log(1+exp(e))
return(exp(log_num-log_den))
}

ee <- seq(-2, 2, 0.001)
plot(ee, g_bar2(ee), type='l', lwd=2, xlab=TeX(r"($\eta$)"), ylab=TeX(r"($\bar{g}(\eta)$)"))
```



- Use a normal approximation to find a 95% probability interval for η. Transform this interval to obtain a 95% probability interval for the original parameter of interest θ.

As in Exercise 1, a normal approximation is given by the Laplace approximation $\mathcal{N}(m, v)$. In this case the mean is

```
m <- optimize(g_bar2, c(0, 1), maximum = TRUE)$maximum
m
```

```
## [1] 0.5066005
```

while for the standard deviation we compute the second derivative of $\log(g(\eta|\text{data}))$ symbolically and evaluate it in m

```
log_g = expression(125*log(2+3*exp(eta)) + 35*eta - 199*log(1+exp(eta)))
d2_log_g <- D(D(log_g, 'eta'), 'eta')
eta <- m
v <- sqrt(-1 / eval(d2_log_g))  # sd of of our normal approximation
v
```

```
## [1] 0.2175282
```

Thus, the approximation of the 95% confidence interval for η is

```
I_eta <- qnorm(c(0.025, 0.975), mean=m, sd=v)
I_eta
```

```
## [1] 0.08025304 0.93294791
```

Since the transformation $\eta = \eta(\theta)$ is monotonically increasing, if I_η is the CI for η, then the corresponding CI for θ $I_\eta = \eta^{-1}(I)$.

```
eta_inv <- function(eta) exp(eta) / (1 + exp(eta))
I_theta <- eta_inv(I_eta)
I_theta
```

```
## [1] 0.5200525 0.7176730
```

> - Design an Accept-Reject sampling algorithm for simulating from the posterior density η. Use a t proposal density using a small number of degrees of freedom and mean and scale parameters given by the normal approximation.

We can use a student-t with 3 degrees of freedom because it has finite mean and variance.

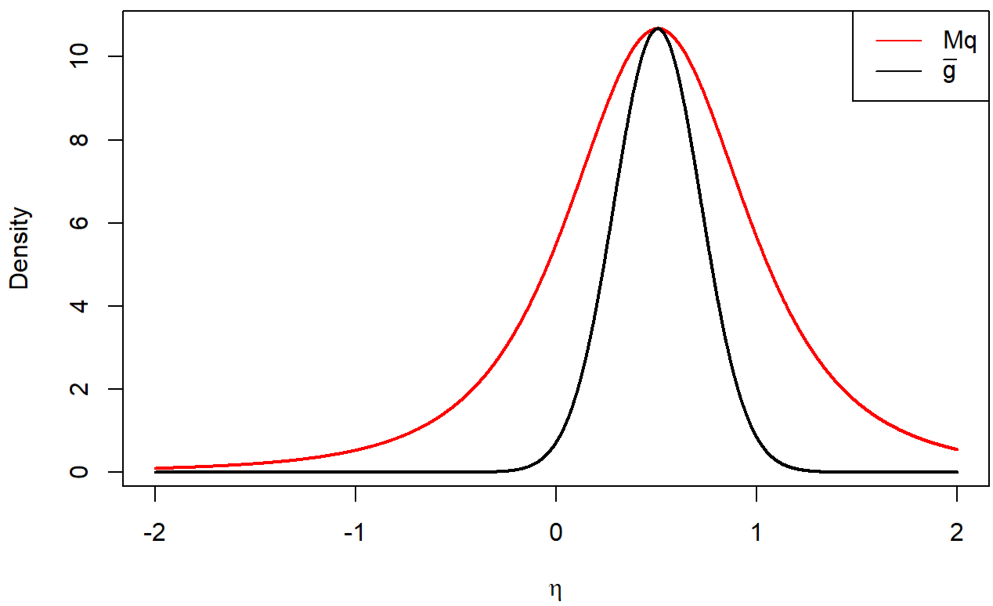
Then, we need M such that $\bar{g} \leq Mq$ or equivalently $M \geq \frac{\bar{g}}{q}$. So we can take $M = \max_\eta \frac{\bar{g}}{q}$.

```
library(mnormt)
M <- optimize(function(e) g_bar2(e) / dmt(e, mean=m, S=v, df=3), interval = c(0, 1), maximum = TRUE)$objective
M
```

```
## [1] 13.54832
```

We can plot the two densities to verify that $\bar{g}$ is below $M \cdot q$.

```
plot(ee, M*dmt(ee, mean=m, S=v, df=3), type='l', col='red', lwd=2, xlab=TeX(r"($\eta$)"), ylab = 'Density')
lines(ee, g_bar2(ee), lwd=2)
legend(x='topright', legend=c(TeX(r"($Mq$)"), TeX(r"($\bar{g}$)")), lty=1, col=c("red", "black"))
```



```
sampler_AR2 <- function(N){
  # returns N samples from our posterior distribution (g).
  samples <- rep(0, N)
  num_samples <- 0
  while (num_samples < N) {
    e <- rmt(1, mean=m, S=v, df=3)
    u <- runif(1)
    if (u * M*dmt(e, mean=m, S=v, df=3) < g_bar2(e)){
      num_samples <- num_samples + 1
      samples[num_samples] <- e
    }
  }
  return(samples)
}
```

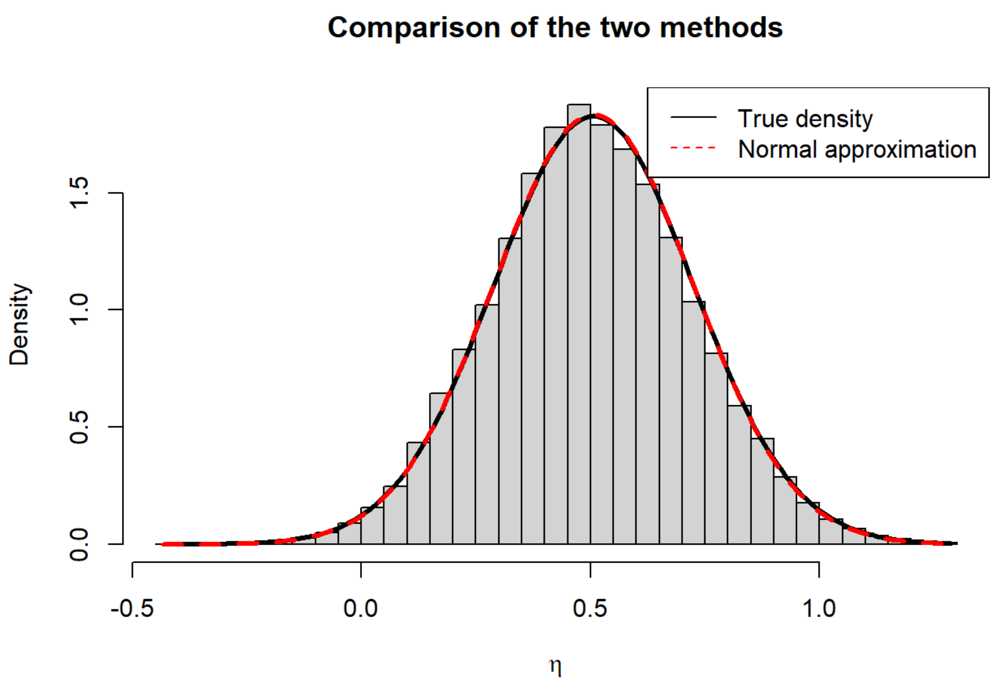
- Compare the results of the two procedures.

We can have a visual comparison of the two procedure by comparing the plots of the density of the normal, the histogram of a lot of samples from the AR algorithm and the density of the true distribution.

```
N <- 10000
g_AR2 <-  sampler_AR2(N)
hist(g_AR2, main = 'Comparison of the two methods', xlab = expression(eta), breaks = 50, freq = FALSE)

ee <- seq(min(g_AR2), max(g_AR2), 0.01)
g_ee <- g_bar2(ee) / integrate(function(e) g_bar2(e),-3, 3)$value
lines(ee, g_ee, col=, lwd=3)
lines(ee, dnorm(ee, mean=m, sd=v), lty='dashed', col='red', lwd=3)

legend(x='topright', legend=c("True density", "Normal approximation"), lty=1:2, col=c("black", "red"))
```



We can see that both methods are very good approximations.

Exercise 3: Poisson Regression

Consider an experiment involving subjects reporting one stressful event. The collected data $y_1, \cdots, y_{18}$ where y_i is the number of events recalled i months before the interview. Suppose y_i is Poisson distributed with mean λ_i, where the λ_is satisfy the loglinear regression model

$$\log \lambda_i = \beta_0 + \beta_1 i \,.$$

The data are shown in the following table

x	y
1	15
2	11
3	14
4	17
5	5
6	11
7	10
8	4
9	8
10	10
11	7
12	9
13	11
14	3
15	6
16	1
17	1
18	4

If (β_0, β_1) is assigned a uniform prior, then the log of the posterior density is given, up to an additive constant, by

$$\log(g(\beta_0, \beta_1|data)) \propto \sum_{i=1}^{18} [y_i(\beta_0 + \beta_1 i) - \exp(\beta_0 + \beta_1 i)]$$

- Write an R function to compute the log of the posterior density of (β_0, β_1).

```
log_post <- function(beta){
  i <- 1:18
return(sum(y*(beta[1] + beta[2]*i) - exp(beta[1] + beta[2]*i)))
}
```

- Suppose we are interested in estimating the posterior mean and standard deviation for the slope β_1. Approximate these moments by a normal approximation about the posterior mode.

Since we are approximating the bivariate posterior with a normal $\mathcal{N}(m = (m_0, m_1), V)$, then the marginal of β_1 is a normal of mean m_1 and variance $V[2, 2]$.

The function `optim` that we use to compute the mode and the hessian of the posterior needs a starting point. In order to get one, we can fit a line to the data, in the log scale.

```
library(stats)  # To fit the line to the data.

fitted_line <- glm(log(y) ~ x)
beta_start <- fitted_line[1]$coefficients
normal_param <- optim(par = beta_start, fn = log_post, hessian=TRUE, control=list(fnscale=-1))
mode_beta <- normal_param$par
V <- - solve(normal_param$hessian) # Covariance matrix of the posterior.

mean_beta1_norm <- mode_beta[2]
sd_beta1_norm <- sqrt(V[2, 2])
```

For the normal approximation of β_1, the mean is -0.0837361 and standard deviation is 0.0167991.

> - Use a multivariate t proposal density and the SIR algorithm to simulate 1000 draws from the posterior density. Use this sample to estimate the posterior mean and standard deviation of the slope β_1.

The SIR algorithm for the bivariate case is very easy to implement.

```
log_post_vec <- function(beta){
  # Same as log_post, but takes a matrix as input.
  sum_1_18 <- 0
  for (i in 1:18) {
    sum_1_18 <- sum_1_18 + y[i]*(beta[,1] + beta[,2]*i) - exp(beta[,1] + beta[,2]*i)
  }
return(sum_1_18)
}
sampler_SIR3 <- function(N){
  # Sampling
  beta <- rmt(N, mean=mode_beta, S=V, df=3)
  # Importance
  # To avoid problems with float precision I subtract 174 from log_post_vec(beta).
  w <- exp(log_post_vec(beta) - 174 - log(dmt(beta, mean=mode_beta, S=V, df=3)))
  w <- w / sum(w)
  # Resampling
  idx <- sample(1:N, size=N, replace=TRUE, prob=w)
return(beta[idx,])
}
```

We can use the sampler to estimate the posterior mean and standard deviation of β_1.

```
g_SIR3 <- sampler_SIR3(1000)
mean_beta1_SIR <- mean(g_SIR3[,2])
sd_beta1_SIR <- sd(g_SIR3[,2])
```

The SIR approximation for the mean is -0.0836072 and for the standard deviation is 0.01679.

> - Compare your estimates with the estimates using the normal approximation.

We can compute the difference between the parameters obtained with the two approximations.

```
mean_beta1_SIR - mean_beta1_norm
```

```
##            x
## 0.0001288212
```

```
sd_beta1_SIR - sd_beta1_norm
```

```
## [1] -9.029799e-06
```

The two approximations are very close.

Exercise 4: Variance components model

Consider the data concerning batch-to-batch variation in yields of dyestuff. The following data arise from a balanced experiment whereby the total product yield was determined for five samples from each of six randomly chosen batches of raw material.

	S1	S2	S3	S4	S5
1	1545	1440	1440	1520	1580
2	1540	1555	1490	1560	1495
3	1595	1550	1605	1510	1560
4	1445	1440	1595	1465	1545
5	1595	1630	1515	1635	1625
6	1520	1455	1450	1480	1445

Let y_{ij} denote the jth observation in batch i. To determine the relative importance of between-batch variation versus sampling variation, the following multilevel model is applied (N denotes the number of batches and n denotes the number of observations per batch).

- $y_{ij} \sim N(\mu + b_i, \sigma_y)$, $i = 1, \dots, N$, $j = 1, \dots, n$.
- $b_i \sim N(0, \sigma_b)$, $i = 1, \dots, N$.
- (σ_y^2, σ_b^2) is assigned a uniform prior.

In this situation, the focus is on the marginal posterior distribution of the variance components. It is possible to analytically integrate out the random effects b_is, resulting in the marginal posterior density of $(\mu, \sigma_y^2, \sigma_b^2)$ given, up to a proportionally constant, by

$$\prod_{i=1}^{N} \left[\phi\left(\bar{y}_i | \mu, \sqrt{\sigma_y^2/n + \sigma_b^2}\right) f_G\left(S_i | (n-1)/2, 1/(2\sigma_y^2)\right)\right],$$

where $\hat{y}_i$ and S_i are respectively the mean yield and the "within sum of squares" of the ith batch, $\phi\left(y|\mu,\sigma\right)$ is the normal density of mean μ and standard deviation σ, and $f_G(y|a,b)$ is the gamma density proportional to $y^{a-1}\exp(-by)$.

> - Write an R function for the log of the posterior density with parametrization $\theta = (\mu, \log\sigma_y, \log\sigma_b)$.

We have to consider the transformation

$$h : (\mu, \sigma_y^2, \sigma_b^2) \to \left(\mu, \frac{1}{2}\log(\sigma_y^2), \frac{1}{2}\log(\sigma_b^2)\right) = (\mu, \log(\sigma_y), \log(\sigma_b)) =: \theta$$

that has inverse

$$h^{-1}(\mu, \log(\sigma_y), \log(\sigma_b)) \to \left(\mu, e^{2\log(\sigma_y)}, e^{2\log(\sigma_b)}\right)$$

The determinant of the Jacobian is

$$|J| = \begin{vmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{2\sigma_y^2} & 0 \\ 0 & 0 & \frac{1}{2\sigma_b^2} \end{vmatrix} = \frac{1}{4\sigma_y^2\sigma_b^2}$$

so $|J^{-1}| = 4\sigma_y^2\sigma_b^2$. The log of the posterior density of theta, up to an additive constant, is

$$\begin{aligned} \log(g(\theta|y)) &\propto \sum_{i=1}^{N} \log\left(\phi\left(\bar{y}_i|\mu, \sqrt{e^{2\log(\sigma_y)}/n + e^{2\log(\sigma_y)}}\right)\right) + \log\left(f_G\left(S_i|(n-1)/2, 1/(2e^{2\log(\sigma_y)})\right)\right) + \log\left(4e^{2\log(\sigma_y)}e^{2\log(\sigma_b)}\right) \\ &\propto \sum_{i=1}^{N} -\log(\sigma_y) - \frac{1}{2}\frac{(\bar{y}_i - \mu)^2}{e^{2\log(\sigma_y)}/n + e^{2\log(\sigma_y)}} + \frac{n-3}{2}\log(S_i) - \frac{S_i}{2e^{2\log(\sigma_y)}} + 2\log(\sigma_y) + 2\log(\sigma_b) \end{aligned}$$

Before writing the R function for the log of the posterior density of θ I compute $\bar{y}_i$ and S_i.

```
y_bar <- rowMeans(y)
S <- rowSums((y - y_bar)^2)
```

Now we can write the function

```
log_post_theta <- function(theta, y_bar_=y_bar, S_=S){
  mu <- theta[1]
  sig2y <- exp(2*theta[2])
  sig2b <- exp(2* theta[3])

  log_phi <- dnorm(y_bar_, mu, sqrt(sig2y/5+sig2b),log=TRUE)
  log_fG <- dgamma(S_, shape = 2, rate = 1/(2*sig2y), log=TRUE)
  return( sum(log_phi + log_fG) + log(4*sig2y*sig2b) )
}
```

> - Using a normal approximation, to this aim, find the posterior mode of θ using a numerical method and try the following alternative starting values:
>   - $\theta = (1500, 3, 3)$
>   - $\theta = (1500, 1, 1)$
>   - $\theta = (1500, 10, 10)$
>
> and assess the sensitivity of the numerical method to the starting value.

As always, we can compute the posterior mode using the function `optim`.

```
laplace3 <- optim(c(1500, 3, 3), log_post_theta, hessian = TRUE, control = list(fnscale=-1))
laplace1 <- optim(c(1500, 1, 1), log_post_theta, hessian = TRUE, control = list(fnscale=-1))
laplace10<- optim(c(1500,10,10), log_post_theta, hessian = TRUE, control = list(fnscale=-1))

mode <- rbind(laplace3$par, laplace1$par, laplace10$par)
rownames(mode) <- c("(1500,  3,  3) ", "(1500,  1,  1) ", "(1500, 10, 10) " )
colnames(mode) <- c("mu  ", " log(sigma_y)", " log(sigma_b)")

print(mode)
```

```
##                      mu    log(sigma_y)  log(sigma_b)
## (1500,  3,  3)  1527.504      3.936728      3.933113
## (1500,  1,  1)  1527.599      3.936642      3.932815
## (1500, 10, 10)  1527.490      3.936826      3.932787
```

An estimate of the sensitivity to the starting value is the following ratio:

```
d_g <- norm(laplace3$par - laplace1$par, type='2')
d_theta <- norm(c(1500, 3, 3)- c(1500, 1, 1), type='2')
sensitivity <- d_g / d_theta
sensitivity
```

```
## [1] 0.03326983
```

- Use the normal approximation to find 90% interval estimates for the log of the standard deviation $\log\sigma_y, \log\sigma_b$.

We can use the estimate of the mode and the hessian obtained with the first starting value. Thus, the parameters of our normal are:

```
m <- laplace3$par
V <- - solve(laplace3$hessian)
```

So an estimate of the 90% confidence interval for $\log\sigma_y$ is

```
I_log_sigy <- qnorm(c(0.05, 0.95), mean=m[2], sd=sqrt(V[2, 2]))
I_log_sigy
```

```
## [1] 3.693275 4.180181
```

while for $\log\sigma_b$ is

```
I_log_sigb <- qnorm(c(0.05, 0.95), mean=m[3], sd=sqrt(V[3, 3]))
I_log_sigb
```

```
## [1] 3.241286 4.624940
```

- Using the results from the previous point find 90% interval estimates for the variance components σ_y^2, σ_b^2.

If $\sigma > 0$, then $a \leq \log(\sigma) \leq b \iff e^{2a} \leq \sigma^2 \leq e^{2b}$. So the confidence interval for σ_y^2 is

```
exp(2 * I_log_sigy)
```

```
## [1] 1614.128 4274.242
```

while for σ_b^2 is

```
exp(2 * I_log_sigb)
```

```
## [1]   653.6504 10403.3071
```