Solutions: Modelling Winning Times of Scottish Hill Races

PART 1: A BAYESIAN MODEL FOR WINNING TIMES

1.1.

$$\begin{aligned}\pi(\boldsymbol{\beta},\omega|y_1,\ldots,y_n) &\propto \omega^{n/2}\exp\left(-\frac{\omega}{2}\sum_{i=1}^{n}(y_i-\mathbf{x}_i^T\boldsymbol{\beta})^2\right)\\ &\times \exp\left(-\frac{1}{2}(\boldsymbol{\beta}-\boldsymbol{\mu}_0)^T\mathbf{C}_0^{-1}(\boldsymbol{\beta}-\boldsymbol{\mu}_0)\right)\omega^{\alpha_0-1}e^{-\lambda_0\omega}I[\omega>0].\end{aligned}$$

1.2 and 1.3.

$$\begin{aligned}\pi(\boldsymbol{\beta}|\omega,\mathbf{y}) &\propto \exp\left\{-\frac{\omega}{2}\sum_{i=1}^{n}(y_i-\mathbf{x}_i^T\boldsymbol{\beta})^2\right\}\exp\left\{-\frac{1}{2}(\boldsymbol{\beta}-\boldsymbol{\mu}_0)^T\mathbf{C}_0^{-1}(\boldsymbol{\beta}-\boldsymbol{\mu}_0)\right\}\\ &= \exp\left\{-\frac{\omega}{2}(\mathbf{y}-\mathbf{X}\boldsymbol{\beta})^T(\mathbf{y}-\mathbf{X}\boldsymbol{\beta})-\frac{1}{2}(\boldsymbol{\beta}-\boldsymbol{\mu}_0)^T\mathbf{C}_0^{-1}(\boldsymbol{\beta}-\boldsymbol{\mu}_0)\right\}\\ &= \exp\left\{-\frac{1}{2}\Big(\omega\mathbf{y}^T\mathbf{y}-2\boldsymbol{\beta}^T\omega\mathbf{X}^T\mathbf{y}+\omega\boldsymbol{\beta}^T\mathbf{X}^T\mathbf{X}\boldsymbol{\beta}+\boldsymbol{\beta}^T\mathbf{C}_0^{-1}\boldsymbol{\beta}-2\boldsymbol{\beta}^T\mathbf{C}_0^{-1}\boldsymbol{\mu}_0+\boldsymbol{\mu}_0^T\mathbf{C}_0^{-1}\boldsymbol{\mu}_0\Big)\right\}\\ &\propto \exp\left\{-\frac{1}{2}\Big(-2\boldsymbol{\beta}^T\omega\mathbf{X}^T\mathbf{y}+\omega\boldsymbol{\beta}^T\mathbf{X}^T\mathbf{X}\boldsymbol{\beta}+\boldsymbol{\beta}^T\mathbf{C}_0^{-1}\boldsymbol{\beta}-2\boldsymbol{\beta}^T\mathbf{C}_0^{-1}\boldsymbol{\mu}_0\Big)\right\}\\ &= \exp\left\{-\frac{1}{2}\Big[\boldsymbol{\beta}^T(\omega\mathbf{X}^T\mathbf{X}+\mathbf{C}_0^{-1})\boldsymbol{\beta}-2\boldsymbol{\beta}^T(\omega\mathbf{X}^T\mathbf{y}+\mathbf{C}_0^{-1}\boldsymbol{\mu}_0)\Big]\right\}\\ &= \exp\left\{-\frac{1}{2}\Big[\boldsymbol{\beta}^T\mathbf{C}_1^{-1}\boldsymbol{\beta}-2\boldsymbol{\beta}^T\mathbf{C}_1^{-1}\boldsymbol{\mu}_1\Big]\right\}\\ &\propto \exp\left\{-\frac{1}{2}\Big[\boldsymbol{\beta}^T\mathbf{C}_1^{-1}\boldsymbol{\beta}-2\boldsymbol{\beta}^T\mathbf{C}_1^{-1}\boldsymbol{\mu}_1+\boldsymbol{\mu}_1^T\mathbf{C}_1^{-1}\boldsymbol{\mu}_1\Big]\right\}\\ &= \exp\left\{-\frac{1}{2}(\boldsymbol{\beta}-\boldsymbol{\mu}_1)^T\mathbf{C}_1^{-1}(\boldsymbol{\beta}-\boldsymbol{\mu}_1)\right\}\end{aligned}$$

Therefore,

$$\boldsymbol{\beta}|\omega,\mathbf{y}\sim N_2(\boldsymbol{\mu}_1,\mathbf{C}_1)$$

Finally,

$$\begin{aligned}\pi(\omega|\mathbf{y},\boldsymbol{\beta}) &\propto \exp\left\{-\frac{\omega}{2}\sum_{i=1}^{n}(y_i-\mathbf{x}_i^T\boldsymbol{\beta})^2\right\}\omega^{\frac{n}{2}+\alpha_0-1}e^{-\lambda_0\omega}I[\omega>0]\\ &= \exp\left\{-\frac{\omega}{2}(\mathbf{y}-\mathbf{X}\boldsymbol{\beta})^T(\mathbf{y}-\mathbf{X}\boldsymbol{\beta})\right\}\omega^{\frac{n}{2}+\alpha_0-1}e^{-\lambda_0\omega}I[\omega>0],\end{aligned}$$

from where it is immediate that

$$\omega|\mathbf{y},\boldsymbol{\beta}\sim\text{Gamma}\left(\alpha_0+\frac{n}{2},\lambda_0+\frac{(\mathbf{y}-\mathbf{X}\boldsymbol{\beta})^T(\mathbf{y}-\mathbf{X}\boldsymbol{\beta})}{2}\right).$$

1.4. Gibbs sampling algorithm:
1. Start the chain at some values $\boldsymbol{\beta}^{(0)}$, $\omega^{(0)}$.
2. Simulate $\boldsymbol{\beta}^{(1)}$ from the distribution

$$N_2(\boldsymbol{\mu}_1, \mathbf{C}_1) \quad \text{with} \quad \mathbf{C}_1 = (\mathbf{C}_0^{-1} + \omega^{(0)}\mathbf{X}^T\mathbf{X})^{-1} \text{ and } \boldsymbol{\mu}_1 = \mathbf{C}_1(\mathbf{C}_0^{-1}\boldsymbol{\mu}_0 + \omega^{(0)}\mathbf{X}^T\mathbf{y})$$

3. Simulate $\omega^{(1)}$ from the distribution

$$\text{Gamma}\left(\alpha_0 + \frac{n}{2}, \lambda_0 + \frac{(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}^{(1)})^T(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}^{(1)})}{2}\right)$$

4. Iterate the simulation procedure a large number of times. Discard an initial number of draws (burn-in period) and keep the remaining ones.

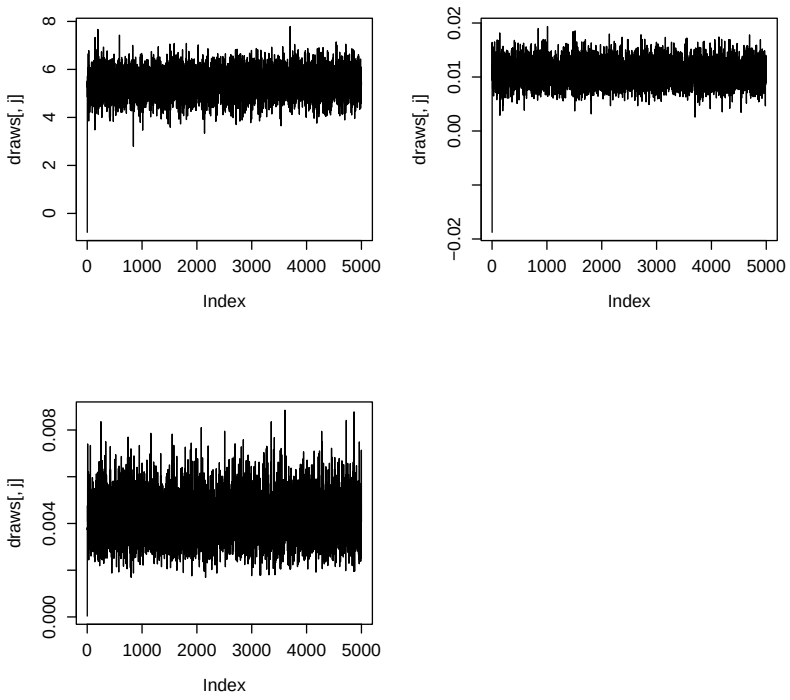
1.5.

```
> hillsgibbs
function (y=hills$time,x=cbind(hills$dist,hills$climb),
mu0=c(0,0),kappa0=c(0.1,0.1),alpha0=0.1,lambda0=0.1,
nburn=0,ndraw=5000)
{

#GIBBS sampler for hills practical, PART 1:

n <- length(y)

xt <- t(x)
xtx <- xt %*% x

c0minus1 <- diag(kappa0)

alpha1 <- alpha0 + (n/2)

# INITIAL VALUES DRAWN FROM PRIOR:

beta <- rmnorm(2,mu0,diag(1/kappa0))

omega <- rgamma(1,alpha0,1)/lambda0

# matrix for recorded draws:

draws <- matrix(ncol=3,nrow=ndraw)

# MCMC LOOP FOLLOWS:

it <- -nburn
```

```
while(it < ndraw){ it <- it+1;

# draw beta:
c1 <- solve(c0minus1 + omega*xtx)
mu1 <- as.vector(c1 %*% (c0minus1 %*% mu0 + omega* xt %*% y))
beta <- rmnorm(2,mu1,c1)

# draw omega:
ymxbeta <- as.vector(y - x %*% beta)
lambda1 <- as.vector( lambda0 + (t(ymxbeta) %*% ymxbeta)/2 )
omega <- rgamma(1,alpha1,1)/lambda1


# after burn-in, record:
if(it>0){
draws[it,c(1:2)] <- beta
draws[it,3] <- omega
}


}

# END MCMC

return(draws)

}
```

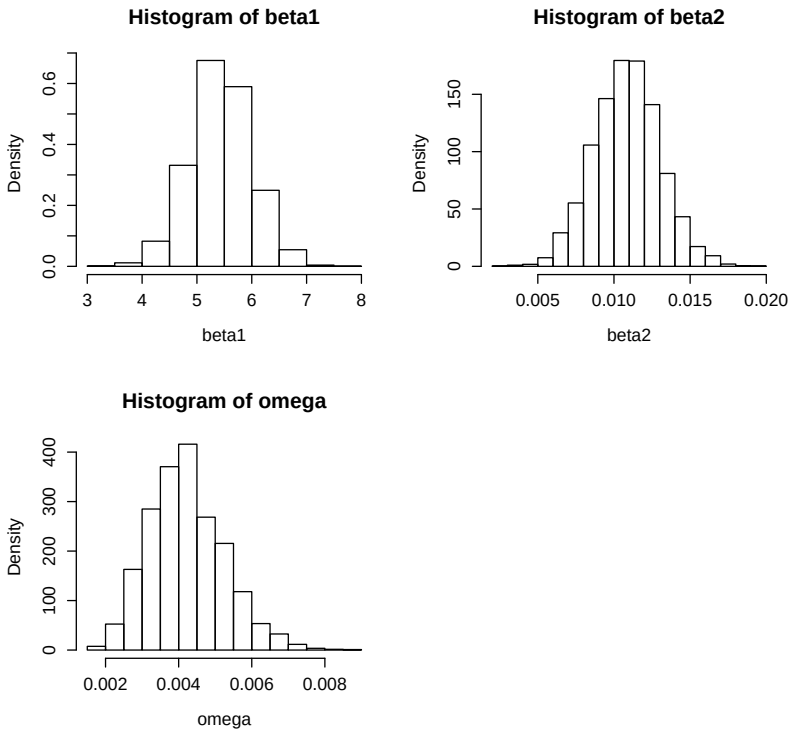
To run the function using the arguments we have set as default:

```
> draws <- hillsgibbs()

> par(mfrow=c(2,2))
> for(j in 1:3) plot(draws[,j],type="l")
```

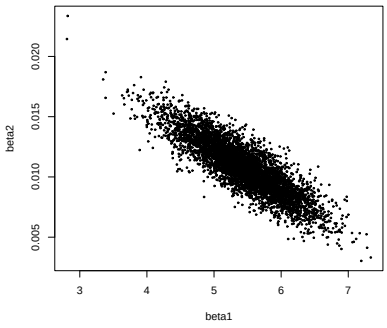


A burn-in period of 1,000 iterations seems ample for convergence:

```
> beta1 <- draws[-c(1:1000),1]
> beta2 <- draws[-c(1:1000),2]
> omega <- draws[-c(1:1000),3]

> par(mfrow=c(2,2))
> hist(beta1,probab=T)
> hist(beta2,probab=T)
> hist(omega,probab=T)

> summary(beta1)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  3.341   5.070   5.436   5.436   5.822   7.789
> summary(beta2)
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
0.002574 0.009365 0.010870 0.010840 0.012300 0.019360
> summary(omega)
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
0.001691 0.003492 0.004135 0.004215 0.004854 0.008848
```




```
> par(mfrow=c(1,1))
> plot(beta1,beta2,pch=20,cex=.5)
```



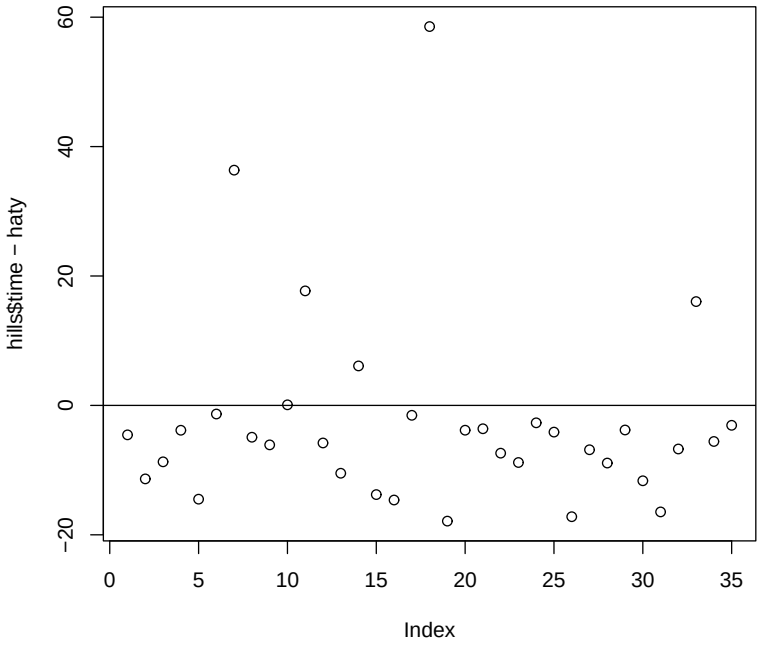
Conclusion is that both distance and climb have a significant effect on winning times of races. Looking at the marginal distributions of β_1 and β_2, each additional mile increases the expected winning time by approximately 5.5 minutes, whereas each additional 100 feet of ascend increases expected winning time by about 1.1 minutes (1 minute and 6 seconds). However, there is strong negative posterior correlation between β_1 and β_2, so it's difficult to draw separate inference about each of these parameters.

1.6. For this, we could set up a counter, let us call it `meanbeta1`. At the beginning of the program, initiliase the counter to 0, that is `meanbeta1 <- 0`. Add the simulated value of β_1 to the counter, that is `meanbeta1 <- meanbeta1 + beta[1]`, at each

MCMC iteration after the burn-in period. Finally, after the MCMC loop, divide the value of the counter by the number of MCMC iterations after the burn-in period, that is `meanbeta1 <- meanbeta1/ndraw`.

1.7.

```
> haty <- mean(beta1)*hills$dist + mean(beta2)*hills$climb
> par(mfrow=c(1,1))
> plot(hills$time - haty)
> abline(h=0)
```



Two things are noticeable. First, there are a couple of observations whose winning times are hugely underpredicted (that is, $\hat{y}_i << y_i$). At the same time, for the vast majority of the observations we have $\hat{y}_i > y_i$. This suggests that we have at least a couple of outliers in the data (for our model choice) and that they are having a big influence on the fitted regression line.

PART 2: A SAMPLING MODEL RESISTANT TO OUTLIERS

2.1.

$$\begin{aligned}\pi(\boldsymbol{\beta},\omega|\mathbf{y}) &\propto \omega^{n/2}\left\{\prod_{i=1}^{n}\frac{1}{1+\omega(y_i-\mathbf{x}_i^T\boldsymbol{\beta})^2}\right\}\\ &\times \exp\left(-\frac{1}{2}(\boldsymbol{\beta}-\boldsymbol{\mu}_0)^T\mathbf{C}_0^{-1}(\boldsymbol{\beta}-\boldsymbol{\mu}_0)\right)\omega^{\alpha_0-1}e^{-\lambda_0\omega}I[\omega>0]\end{aligned}$$

2.2. Conditional posterior density of $\boldsymbol{\beta}$:

$$\pi(\boldsymbol{\beta}|\omega,\mathbf{y}) \propto \left\{\prod_{i=1}^{n}\frac{1}{1+\omega(y_i-\mathbf{x}_i^T\boldsymbol{\beta})^2}\right\}\exp\left(-\frac{1}{2}(\boldsymbol{\beta}-\boldsymbol{\mu}_0)^T\mathbf{C}_0^{-1}(\boldsymbol{\beta}-\boldsymbol{\mu}_0)\right)$$

Conditional posterior density of ω:

$$\pi(\omega|\boldsymbol{\beta},\mathbf{y}) \propto \omega^{n/2}\left\{\prod_{i=1}^{n}\frac{1}{1+\omega(y_i-\mathbf{x}_i^T\boldsymbol{\beta})^2}\right\}\omega^{\alpha_0-1}e^{-\lambda_0\omega}I[\omega>0]$$

None of the two densities corresponds to any well-known distribution from which we can draw directly, so Gibbs sampling is not possible.

2.3 Conditional posterior density of $\boldsymbol{\beta}$:

$$\begin{aligned}\pi(\boldsymbol{\beta}|\omega,\mathbf{y},\mathbf{z}) &\propto \exp\left\{-\frac{\omega}{2}\sum_{i=1}^{n}z_i(y_i-\mathbf{x}_i^T\boldsymbol{\beta})^2\right\}\exp\left\{-\frac{1}{2}(\boldsymbol{\beta}-\boldsymbol{\mu}_0)^T\mathbf{C}_0^{-1}(\boldsymbol{\beta}-\boldsymbol{\mu}_0)\right\}\\ &= \exp\left\{-\frac{\omega}{2}(\mathbf{y}-\mathbf{X}\boldsymbol{\beta})^T\mathrm{Diag}(z_1,\ldots,z_n)(\mathbf{y}-\mathbf{X}\boldsymbol{\beta})-\frac{1}{2}(\boldsymbol{\beta}-\boldsymbol{\mu}_0)^T\mathbf{C}_0^{-1}(\boldsymbol{\beta}-\boldsymbol{\mu}_0)\right\},\end{aligned}$$

where $\mathrm{Diag}(z_1,\ldots,z_n)$ denotes a diagonal matrix whose diagonal entries are $z_1,\ldots,z_n$. Doing algebraic manipulations very similar to those in questions 1.2 and 1.3, it is easy to see that

$$\boldsymbol{\beta}|\omega,\mathbf{y},\mathbf{z} \sim N_2(\boldsymbol{\mu}_1,\mathbf{C}_1)$$

where now

$$\mathbf{C}_1=\left(\mathbf{C}_0^{-1}+\omega\mathbf{X}^T\mathrm{Diag}(z_1,\ldots,z_n)\mathbf{X}\right)^{-1} \text{ and } \boldsymbol{\mu}_1=\mathbf{C}_1\left(\mathbf{C}_0^{-1}\boldsymbol{\mu}_0+\omega\mathbf{X}^T\mathrm{Diag}(z_1,\ldots,z_n)\mathbf{y}\right).$$

Conditional posterior density of ω:

$$\begin{aligned}\pi(\omega|\mathbf{y},\boldsymbol{\beta},\mathbf{z}) &\propto \exp\left\{-\frac{\omega}{2}\sum_{i=1}^{n}z_i(y_i-\mathbf{x}_i^T\boldsymbol{\beta})^2\right\}\omega^{\frac{n}{2}+\alpha_0-1}e^{-\lambda_0\omega}I[\omega>0]\\ &= \exp\left\{-\frac{\omega}{2}(\mathbf{y}-\mathbf{X}\boldsymbol{\beta})^T\mathrm{Diag}(z_1,\ldots,z_n)(\mathbf{y}-\mathbf{X}\boldsymbol{\beta})\right\}\omega^{\frac{n}{2}+\alpha_0-1}e^{-\lambda_0\omega}I[\omega>0],\end{aligned}$$

from where it is immediate that

$$\omega|\mathbf{y},\boldsymbol{\beta},\mathbf{z} \sim \mathrm{Gamma}\left(\alpha_0+\frac{n}{2},\lambda_0+\frac{(\mathbf{y}-\mathbf{X}\boldsymbol{\beta})^T\mathrm{Diag}(z_1,\ldots,z_n)(\mathbf{y}-\mathbf{X}\boldsymbol{\beta})}{2}\right).$$

Conditional posterior density of z_i, $i = 1, \ldots, n$:

$$\begin{aligned}\pi(z_i|\boldsymbol{\beta}, \omega, \mathbf{y}, \mathbf{z}_{-i}) &\propto z_i^{1/2} \exp\left(-\frac{\omega z_i}{2}(y_i - \mathbf{x}_i^T\boldsymbol{\beta})^2\right) z_i^{-1/2} e^{-\frac{1}{2}z_i} I[z_i > 0] \\ &= \exp\left\{-\frac{z_i}{2}\left[1 + \omega(y_i - \mathbf{x}_i^T\boldsymbol{\beta})^2\right]\right\} I[z_i > 0],\end{aligned}$$

where $\mathbf{z}_{-i}$ denotes the vector $(z_1, \ldots, z_n)$ excluding z_i. It now follows immediately that

$$z_i|\boldsymbol{\beta}, \omega, \mathbf{y}, \mathbf{z}_{-i} \sim \text{Gamma}\left(1, \frac{1 + \omega(y_i - \mathbf{x}_i^T\boldsymbol{\beta})^2}{2}\right).$$

Note that $z_1, \ldots, z_n$ are conditionally independent, which helps the sampler.

•Gibbs sampling algorithm:
1. Start the chain at some values $(\boldsymbol{\beta}^{(0)}, \omega^{(0)}, z_1^{(0)}, \ldots, z_n^{(0)})$
2. Simulate $\boldsymbol{\beta}^{(1)}$ from the distribution $N_2(\boldsymbol{\mu}_1, \mathbf{C}_1)$, where

$$\mathbf{C}_1 = \left(\mathbf{C}_0^{-1} + \omega^{(0)}\mathbf{X}^T\text{Diag}(z_1^{(0)}, \ldots, z_n^{(0)})\mathbf{X}\right)^{-1} \text{ and } \boldsymbol{\mu}_1 = \mathbf{C}_1\left(\mathbf{C}_0^{-1}\boldsymbol{\mu}_0 + \omega^{(0)}\mathbf{X}^T\text{Diag}(z_1^{(0)}, \ldots, z_n^{(0)})\mathbf{y}\right).$$

3. Simulate $\omega^{(1)}$ from the distribution

$$\text{Gamma}\left(\alpha_0 + \frac{n}{2}, \lambda_0 + \frac{(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}^{(1)})^T\text{Diag}(z_1^{(0)}, \ldots, z_n^{(0)})(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}^{(1)})}{2}\right).$$

3. For $i = 1, \ldots, n$, simulate z_i from the distribution

$$\text{Gamma}\left(1, \frac{1 + \omega^{(1)}(y_i - \mathbf{x}_i^T\boldsymbol{\beta}^{(1)})^2}{2}\right).$$

4. Iterate this simulation procedure a large number of times. Discard an initial number of draws (the burn-in period) and keep the remaining ones.

2.4 R code:

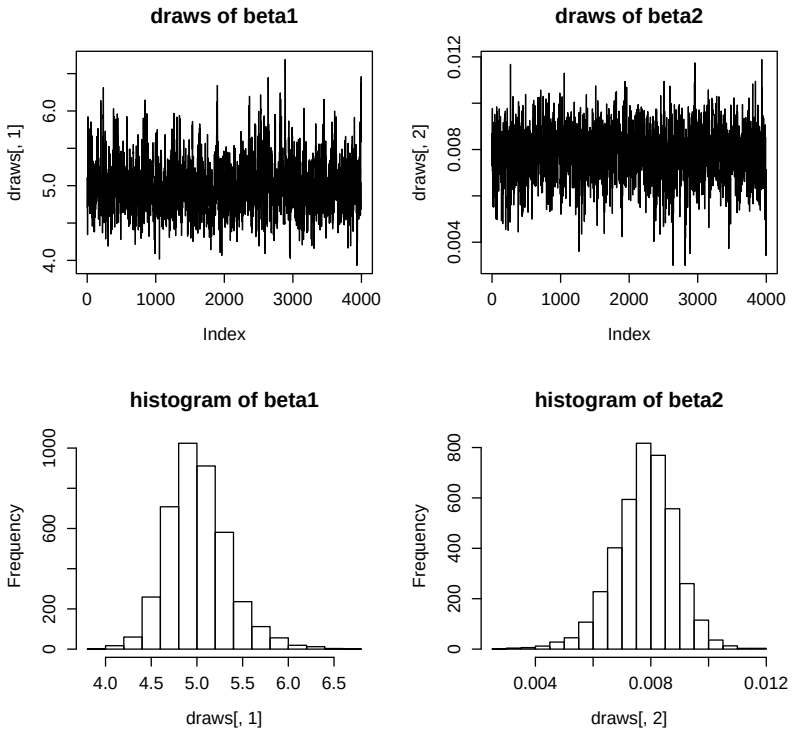
```
> hillsgibbsaug
function (data=hills$time, x=cbind(hills$dist,hills$climb),
mu0=c(0,0),kappa0=c(0.1,0.1),alpha0=0.1,lambda0=0.1,nburn=0,ndraw=5000)
{

#Gibbs sampling with data augmentation
#Part 2 of hill races practical

n <- length(data)

xt <- t(x)
```

```
c0minus1 <- diag(kappa0)

alpha1 <- alpha0 + (n/2)

# INITIAL VALUES DRAWN FROM PRIOR:

beta <- rmnorm(2,mu0,diag(1/kappa0))
omega <- rgamma(1,alpha0,1)/lambda0
z <- rgamma(n,1/2,1)/(1/2)

# matrix to store draws:

draws <- matrix(ncol=3+n,nrow=ndraw)

# MCMC LOOP FOLLOWS:

it <- -nburn
while(it < ndraw){ it <- it+1;

# draw beta:
c1 <- solve(c0minus1 + omega*xt %*% diag(z) %*% x)
mu1 <- as.vector( c1 %*% (c0minus1 %*% mu0 + omega* xt %*% diag(z) %*%data) )
beta <- rmnorm(2,mu1,c1)

# draw omega:
ymxbeta <- as.vector( data - x %*% beta )
lambda1 <- as.vector( lambda0 + (t(ymxbeta) %*% diag(z) %*% ymxbeta)/2 )
omega <- rgamma(1,alpha1,1)/lambda1

# draw z:
div <- (1+omega*(ymxbeta**2))/2
z <- rgamma(n,1,1)/div

# after burn in, record mu, omega and z:
if(it>0){
draws[it,c(1:2)] <- beta
draws[it,3] <- omega
draws[it,c(4:(3+n))] <- z
}

}

# END MCMC
```

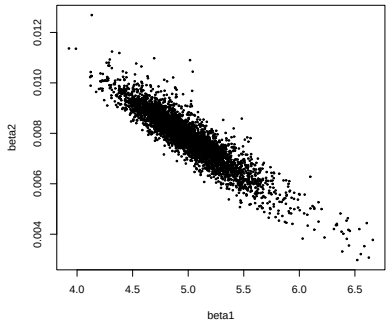
```
return(draws)

}
```

2.5.

```
> draws <- hillsgibbsaug(nburn=1000,ndraw=4000)
> par(mfrow=c(2,2))
> plot(draws[,1],type="l",main="draws of beta1")
> plot(draws[,2],type="l",main="draws of beta2")
> hist(draws[,1],main="histogram of beta1")
> hist(draws[,2],main="histogram of beta2")
> summary(draws[,1])
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  3.933   4.788   4.989   5.013   5.204   6.689
> summary(draws[,2])
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
0.002999 0.007143 0.007868 0.007790 0.008493 0.011890
```



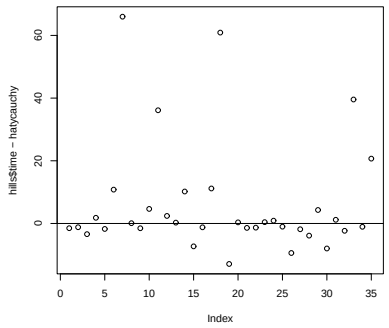

```
> beta1 <- draws[,1]; beta2 <- draws[,2]
> par(mfrow=c(1,1))
> plot(beta1,beta2,pch=20,cex=.5)
```



Posterior inference on β_1 and β_2 is quite similar to the inference obtained under the Normal sampling model in Part 1, although the posterior distributions are now centred at somewhat lower values. The strong negative posterior correlation between β_1 and β_2 is also present here.

2.6.

```
> meanbeta1 <- mean(draws[,1])
> meanbeta2 <- mean(draws[,2])
> hatycauchy <- meanbeta1*hills$dist + meanbeta2*hills$climb
> par(mfrow=c(1,1))
> plot(hills$time-hatycauchy)
> abline(h=0)
```



Using the Cauchy model means that the model fit is better for the vast majority of the observations. In particular, it is no longer the case that for most of the observations the expected winning time is larger than the observed time. In short, it has downweighted the extreme observations allowing the fit to be better for most of the observations.

PART 3: METROPOLIS–HASTINGS FOR CAUCHY MODEL

3.1. The joint posterior density for $(\beta_1, \beta_2, \omega)$ was written down, up to a proportionality constant, in question 2.1. Picking up only the factors involving β_1, we get:

$$\pi(\beta_1|\beta_2, \omega, \mathbf{y}) \propto \left\{\prod_{i=1}^{n} \frac{1}{1+\omega(y_i - x_{i1}\beta_1 - x_{i2}\beta_2)^2}\right\} \exp\left(-\frac{\kappa_{01}}{2}(\beta_1 - \mu_{01})^2\right).$$

Similarly, considering only the factors that involve β_2 we get:

$$\pi(\beta_2|\beta_1, \omega, \mathbf{y}) \propto \left\{\prod_{i=1}^{n} \frac{1}{1+\omega(y_i - x_{i1}\beta_1 - x_{i2}\beta_2)^2}\right\} \exp\left(-\frac{\kappa_{02}}{2}(\beta_2 - \mu_{02})^2\right).$$

Finally, considering only the factors involving ω:

$$\pi(\omega|\beta_1, \beta_2, \mathbf{y}) \propto \omega^{n/2} \left\{\prod_{i=1}^{n} \frac{1}{1+\omega(y_i - x_{i1}\beta_1 - x_{i2}\beta_2)^2}\right\} \omega^{\alpha_0 - 1} e^{-\lambda_0 \omega} I[\omega > 0]$$

So an MCMC algorithm for sampling the posterior distribution of $(\beta_1, \beta_2, \omega)$ using random walk Metropolis steps with Normal increments would be:

1. Start the chain at some values $\beta_1^{(0)}$, $\beta_2^{(0)}$, $\omega^{(0)}$.

2. Draw $\beta_1^{can} \sim N(\beta_1^{(0)}, v_{\beta_1})$, where the variance v_{β_1} is some suitably chosen value. Take as the new value of the chain

$$\beta_1^{(1)} = \begin{cases} \beta_1^{can} & \text{with probability } p \\ \beta_1^{(0)} & \text{with probability } 1-p \end{cases}$$

where

$$p = \min\left\{1,\ \exp\left(\frac{\kappa_{01}}{2}\left[(\beta_1^{(0)} - \mu_{01})^2 - (\beta_1^{can} - \mu_{01})^2\right]\right) \prod_{i=1}^{n} \frac{1+\omega^{(0)}(y_i - x_{i1}\beta_1^{(0)} - x_{i2}\beta_2^{(0)})^2}{1+\omega^{(0)}(y_i - x_{i1}\beta_1^{can} - x_{i2}\beta_2^{(0)})^2}\right\}.$$

(the latter is carried out by sampling $u \sim$Uniform$(0,1)$, and taking $\beta_1^{(1)} = \beta_1^{can}$ if and only if $u < p$).

3. Draw $\beta_2^{can} \sim N(\beta_2^{(0)}, v_{\beta_2})$, where the variance v_{β_2} is some suitably chosen value. Take as the new value of the chain

$$\beta_2^{(1)} = \begin{cases} \beta_2^{can} & \text{with probability } p \\ \beta_2^{(0)} & \text{with probability } 1-p \end{cases}$$

where

$$p = \min\left\{1,\ \exp\left(\frac{\kappa_{02}}{2}\left[(\beta_2^{(0)} - \mu_{02})^2 - (\beta_2^{can} - \mu_{02})^2\right]\right) \prod_{i=1}^{n} \frac{1+\omega^{(0)}(y_i - x_{i1}\beta_1^{(1)} - x_{i2}\beta_2^{(0)})^2}{1+\omega^{(0)}(y_i - x_{i1}\beta_1^{(1)} - x_{i2}\beta_2^{can})^2}\right\}.$$

(the latter is carried out by sampling $u \sim$Uniform$(0,1)$, and taking $\beta_2^{(1)} = \beta_2^{can}$ if and only if $u < p$).

4. Draw $\omega^{can} \sim N(\omega^{(0)}, v_\omega$, where the variance v_ω is some suitably chosen value. Take as the new value of the chain

$$\omega^{(1)} = \begin{cases} \omega^{can} & \text{with probability } p \\ \omega^{(0)} & \text{with probability } 1-p \end{cases}$$

where

$$p = \min\Bigg\{1, \quad \left(\frac{\omega^{can}}{\omega^{(j)}}\right)^{\alpha_0+\frac{n}{2}-1} \exp\Big[\lambda_0(\omega^{(j)} - \omega^{can})\Big] \\ \prod_{i=1}^{n} \frac{1+\omega^{(0)}(y_i - x_{i1}\beta_1^{(1)} - x_{i2}\beta_2^{(1)})^2}{1+\omega^{can}(y_i - x_{i1}\beta_1^{(1)} - x_{i2}\beta_2^{(1)})^2} I[\omega^{can} > 0]\Bigg\}.$$

(the latter is carried out by sampling $u \sim$Uniform$(0,1)$, and taking $\omega^{(1)} = \omega^{can}$ if and only if $u < p$).
Note that if $\omega^{can} \leq 0$, the acceptance probability $p = 0$, so there is no need to compute p and we know that, in that case, $\omega^{(1)} = \omega^{(0)}$.

3.2.

```
> hillsrwmetrop
function (data=hills$time,x1=hills$dist,x2=hills$climb,
mu0=c(0,0),kappa0=c(0.1,0.1),
alpha0=0.1,lambda0=0.1,nburn=1000,ndraw=5000,
stdevbeta1=1,stdevbeta2=1,stdevomega=1)
{

#Random walk Metropolis for Part 3 of hills practical:

n <- length(data)

alpha1 <- (n/2) + alpha0 - 1

#initial values:drawn from prior

beta1 <- rnorm(1,mu0[1],sqrt(1/kappa0[1]))
beta2 <- rnorm(1,mu0[2],sqrt(1/kappa0[2]))
omega <- rgamma(1,alpha0,1)/lambda0

# create matrix for recorded draws:

draws <- matrix(ncol=3,nrow=ndraw)

# counters for acceptance probabilities:
```

```
acceptbeta1 <- 0
acceptbeta2 <- 0
acceptomega <- 0

# MCMC LOOP FOLLOWS:

it <- -nburn
while(it < ndraw){ it <- it+1;

# draw beta1can from N(beta1,stdevbeta1^2):
beta1can <- rnorm(1,beta1,stdevbeta1)

# calculate log(acceptance probability):
logp <- (kappa0[1]/2)*((beta1-mu0[1])^2-(beta1can-mu0[1])^2) +
sum( log( 1+omega*((data-x1*beta1-x2*beta2)^2)) -
     log( 1+omega*((data-x1*beta1can-x2*beta2)^2)) )

# draw u ~ Uniform(0,1):
u <- runif(1,0,1)

# if u<p, accept beta1can:
if (log(u) < logp){acceptbeta1 <- acceptbeta1 + 1; beta1 <- beta1can}

# draw beta2can from N(beta2,stdevbeta2^2):
beta2can <- rnorm(1,beta2,stdevbeta2)

# calculate log(acceptance probability):
logp <- (kappa0[2]/2)*((beta2-mu0[2])^2-(beta2can-mu0[2])^2) +
sum( log( 1+omega*((data-x1*beta1-x2*beta2)^2)) -
     log( 1+omega*((data-x1*beta1-x2*beta2can)^2)) )

# draw u ~ Uniform(0,1):
u <- runif(1,0,1)

# if u<p, accept beta2can:
if (log(u) < logp){acceptbeta2 <- acceptbeta2 + 1; beta2 <- beta2can}

# draw omegacan from N(omega,stdevomega^2):
omegacan <- rnorm(1,omega,stdevomega)

# only compute acceptance probability if omegacan>0:
if(omegacan > 0){

# log(acceptance probability):
```

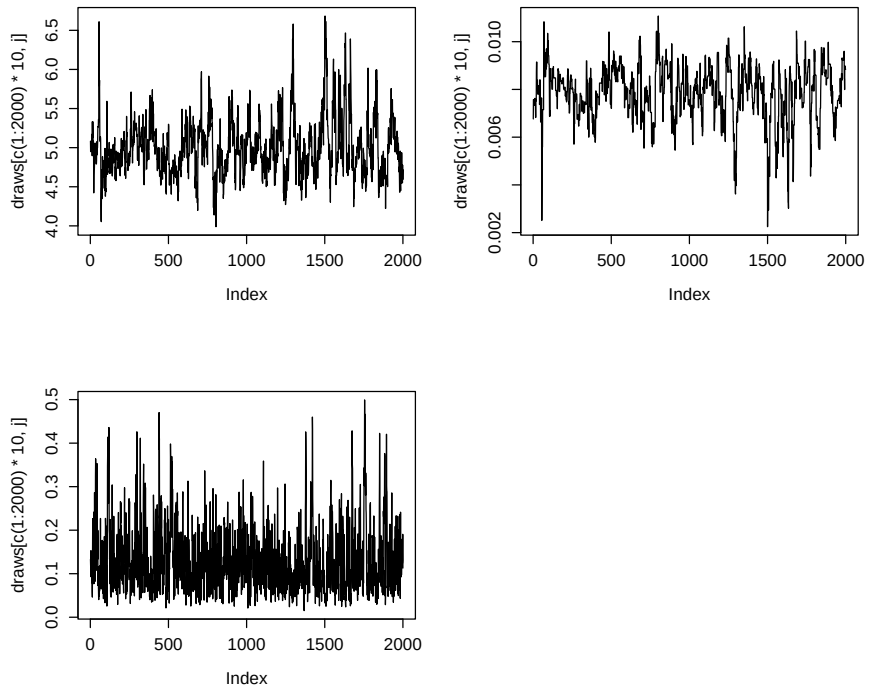
```
logp <- alpha1*(log(omegacan)-log(omega)) + lambda0*(omega-omegacan) +
sum(log(1+omega*((data-x1*beta1-x2*beta2)^2))-
    log(1+omegacan*((data-x1*beta1-x2*beta2)^2)) )

# draw u ~ Uniform(0,1):
u <- runif(1,0,1)

# if u<p, accept omegacan:

if (log(u) < logp){acceptomega <- acceptomega + 1; omega <- omegacan}

}

# after bun-in record (mu,omega):
if(it>0){
draws[it,1] <- beta1
draws[it,2] <- beta2
draws[it,3] <- omega
}

}

# END MCMC

print(c("percentage accepted draws for beta1:",100*acceptbeta1/(nburn+ndraw)))
print(c("percentage accepted draws for beta2:",100*acceptbeta2/(nburn+ndraw)))
print(c("percentage accepted draws for omega:",100*acceptomega/(nburn+ndraw)))

return(draws)

}
```

3.3. From running the function `hillsrwmetrop` and examining the output, it is clear than mixing is worse than for the data augmentation algorithm developed in Part 2 of the practical. Thus, we use fairly long burn-in and recording periods.

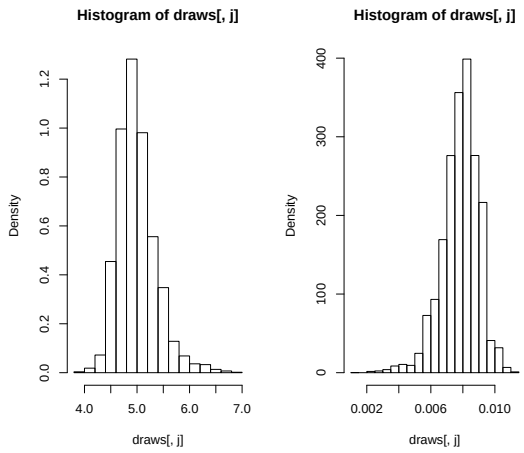
```
> draws <- hillsrwmetrop(stdevbeta1=1,stdevbeta2=0.008,stdevomega=0.035,
+ nburn=10000,ndraw=20000)
[1] "percentage accepted draws for beta1:"
[2] "29.2466666666667"
[1] "percentage accepted draws for beta2:"
[2] "22.1533333333333"
[1] "percentage accepted draws for omega:"
[2] "63.5733333333333"
```

We plot 2,000 equi-spaced draws:

```
> par(mfrow=c(2,2))
> for(j in 1:3) plot(draws[c(1:2000)*10,j],type="l")
```




```
> par(mfrow=c(1,2))
> for(j in 1:2) hist(draws[,j],probab=T)
```




```
> summary(draws[,1])
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
```

```
  3.934   4.757   4.954   4.998   5.189   6.830
> summary(draws[,2])
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
0.001299 0.007217 0.007951 0.007851 0.008617 0.011080
```

Results are very similar to those in Part 2 but, clearly, the chain now mixes more slowly than the chain in Part 2.