Case Study: Prediction of Daily Steps

Daniel Kwapien

Introduction

This project focuses on a case study involving Bayesian data analysis. The aim is to thoroughly understand and apply Bayesian principles to a real-world dataset, providing valuable insights through a step-by-step analysis.

Objective

The goal of this project is to model the distribution of daily steps given the observed examples along two years.

$$P(\text{steps} \mid \text{data})$$

Data

This data was collected by exporting the apple health (https://www.apple.com/ios/health/) from my personal device. It contains the daily steps recorded in a period of time of two years. I will provide the script I used to extract and preprocess the data along with this project.

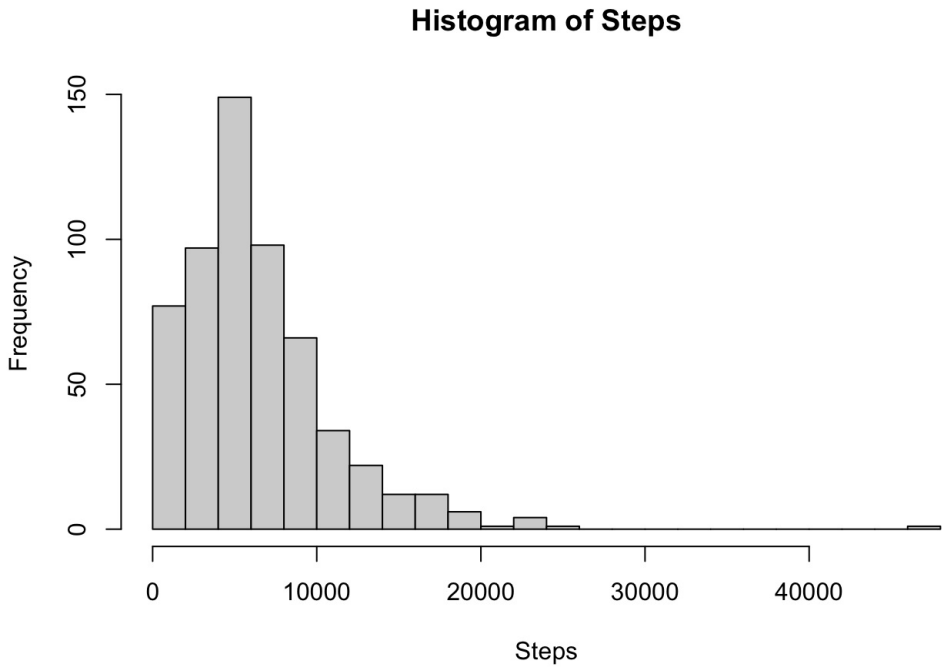
Let's take a look at the data.

```
rm(list=ls())
data = read.csv('./data/cleaned_data.csv')
attach(data)
head(data)
```

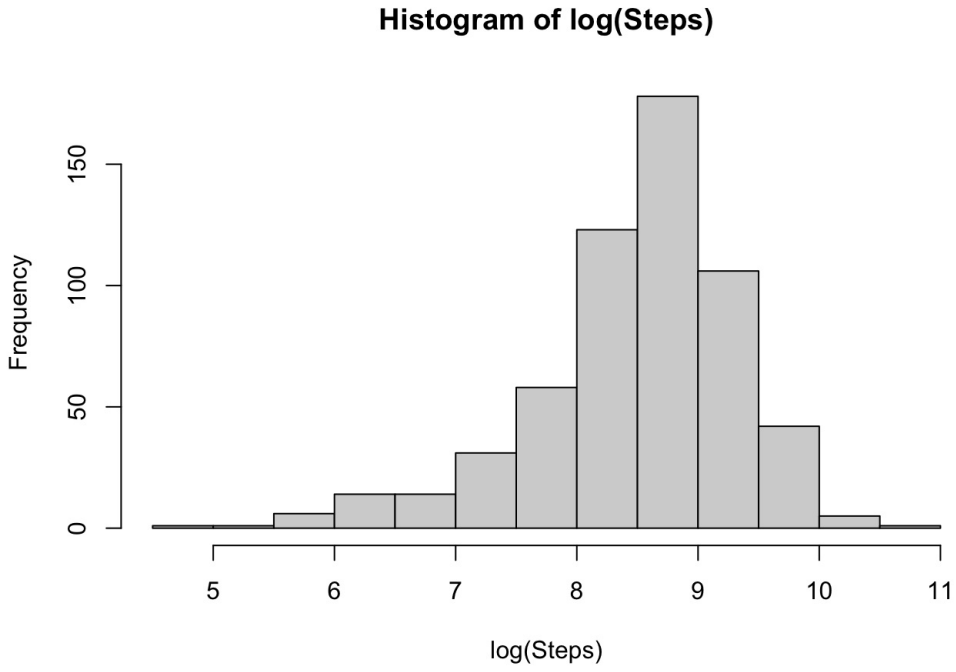
```
##         Date Steps    Speed     Study       Day
## 1 2022-10-18  3988 4.958400 152.33377   Tuesday
## 2 2022-10-19  5689 4.891846  31.25563 Wednesday
## 3 2022-10-20 13570 4.540131   0.00000  Thursday
## 4 2022-10-21  7809 4.532400   0.00000    Friday
## 5 2022-10-22  3902 5.166000   0.00000  Saturday
## 6 2022-10-23  5195 3.932640   0.00000    Sunday
```

This data set was originally thought to make a bayesian linear regression, but finally modeling the steps seems far more interesting

Let's se what we are dealing with



The data seems it could be modeled using a gamma distribution or log-normal. But the scale is very high, which is not ideal. Maybe we could deal with that using a lognormal, but instead we will apply a logarithmic transformation.



As we can see this distribution has a slight negative skew.

Bayesian Weibull model

We will assume that steps, X, follows a weibull distribution

$$X \mid k, \lambda \sim \text{Weibull}(k, \lambda)$$

where the PDF is

$$f(x \mid k, \lambda) = \frac{k}{\lambda}\left(\frac{x}{\lambda}\right)^{k-1} e^{(x/\lambda)^k}$$

We begin by setting up the data and transforming it using the natural logarithm

```
n=length(Steps)
x=log(Steps)
```

This function calculates the log-likelihood of the Weibull distribution given the parameters :

$$f(\text{data}\,|\,k,\lambda) = \prod_{i=1}^{n} \frac{k}{\lambda}\left(\frac{x_i}{\lambda}\right)^{k-1} e^{(x_i/\lambda)^k} \Rightarrow \log L(\text{data}\,|\,k,\lambda) = n\log k - n\log\lambda + (k-1)\sum_{i=1}^{n}\log x_i - \sum_{i=1}^{n}\left(\frac{x_i}{\lambda}\right)^k$$

```
log_likelihood <- function(x, k, lam) {
  return (sum(log(dweibull(x,shape=k,scale=lam))))
}
```

Although we could take as a conjugate prior the inverse gamma function for λ with known k, instead we will assume that:

$$k \sim \text{Log-Normal}(\mu_k, \sigma_k^2)$$

$$\lambda \sim \text{Log-Normal}(\mu_\lambda, \sigma_\lambda^2)$$

Since these priors offer more robustness, gives us only positive results, offers us the log-space flexibility since, for example, the λ parameter has a multiplicative effect, and it offers us skewness handling.

This way, we can get the joint distribution:

$$f(k,\lambda) = f(k)f(\lambda) \Rightarrow \log f(k,\lambda) = \log f(k) + \log f(\lambda)$$

```
log_prior <- function(k, lam) {
  mu_k <- 0.5
  sigma_k <- 1
  mu_lambda <- 0.5
  sigma_lambda <- 1
  log_prior_k <- -0.5 * ((log(k) - mu_k)^2 / sigma_k^2) - log(k * sigma_k * sqrt(2 * pi))
  log_prior_lambda <- -0.5 * ((log(lam) - mu_lambda)^2 / sigma_lambda^2) - log(lam * sigma_lambda * sqrt(2 * pi))
  return(log_prior_k + log_prior_lambda)
}
```

Now, we initialize the parameters for our Markov Chain. First we set the number of iterations:

```
burnin = 2000
iters = 40000
toiter=burnin+iters
```

Now, we intialize the parameters, along with the accpted values

```
k=rep(0,toiter)
lam=rep(0,toiter)

k[1] <- 7
lam[1] <- 70

pac.k = 0
pac.lam = 0
```

Finally, we simulate the Markov Chain

```
for (i in 2:toiter) {
  # Update k
  kc <- rlnorm(1, meanlog = log(k[i-1]), sdlog = 0.1)
  log_alpha_k <- log_likelihood(x, kc, lam[i-1]) + log_prior(kc, lam[i-1]) -
                 log_likelihood(x, k[i-1], lam[i-1]) - log_prior(k[i-1], lam[i-1])

  if (log(runif(1)) < log_alpha_k) {
    k[i] <- kc; if (i>burnin){pac.k=pac.k+1}
  }
  else{
    k[i] <- k[i-1]
  }

  # Update lambda
  lamc <- rlnorm(1, meanlog = log(lam[i-1]), sdlog = 0.02)
  log_alpha_lam <- log_likelihood(x, k[i], lamc) + log_prior(k[i], lamc) -
                   log_likelihood(x, k[i], lam[i-1]) - log_prior(k[i], lam[i-1])

  if (log(runif(1)) < log_alpha_lam) {
    lam[i] <- lamc; if (i>burnin){pac.lam=pac.lam+1}
  }
  else{
    lam[i] <- lam[i-1]
  }
}
```

Let's check the accepted values

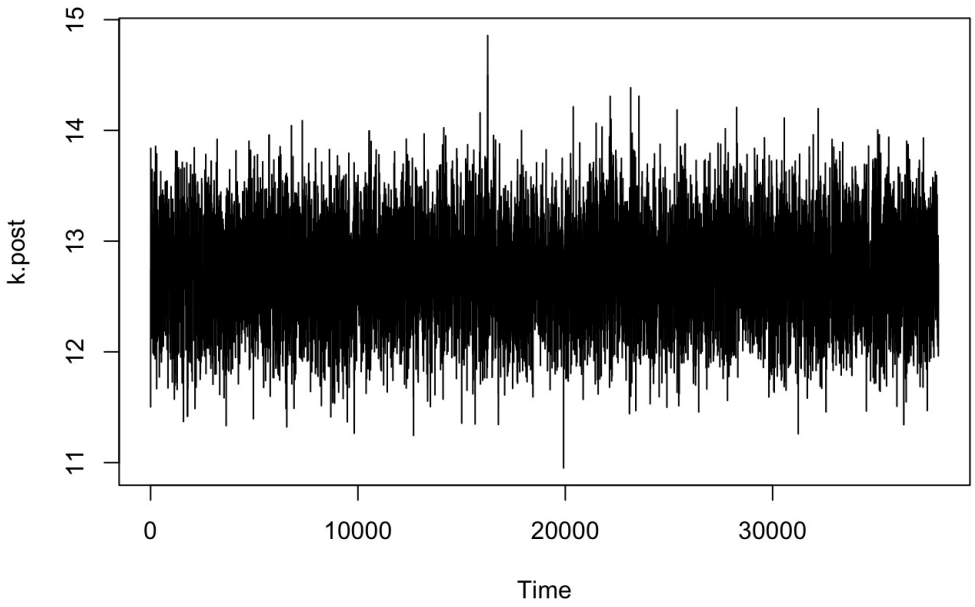
```
pac.k = pac.k/iters
pac.lam = pac.lam/iters
c(pac.k, pac.lam)
```

```
## [1] 0.351400 0.199725
```

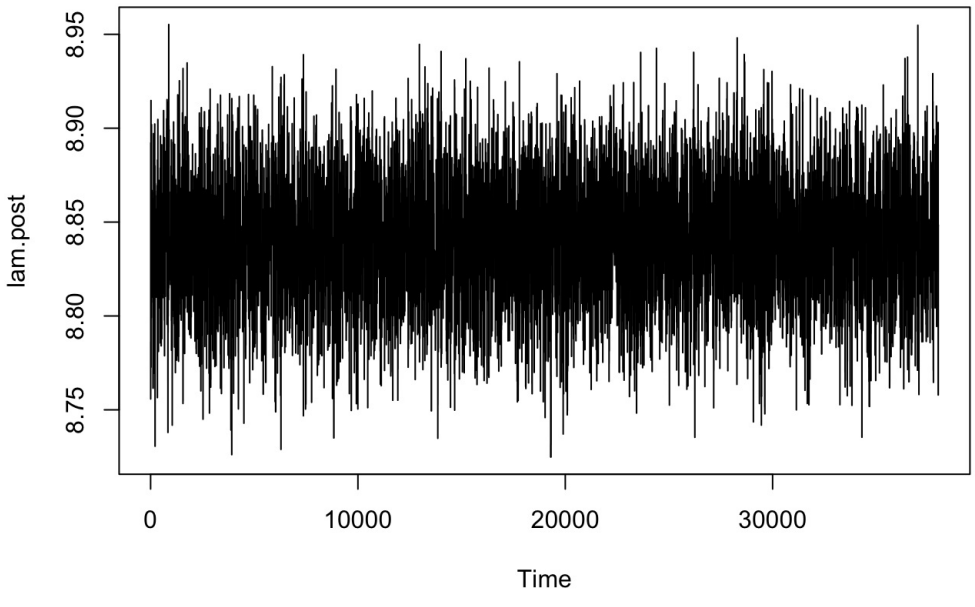
Seems nice, now we will check the convergence of the algorithm

```
thin <- 1
k.post=k[seq(burnin+1,iters,by=thin)]
lam.post=lam[seq(burnin+1,iters,by=thin)]

plot.ts(k.post)
```

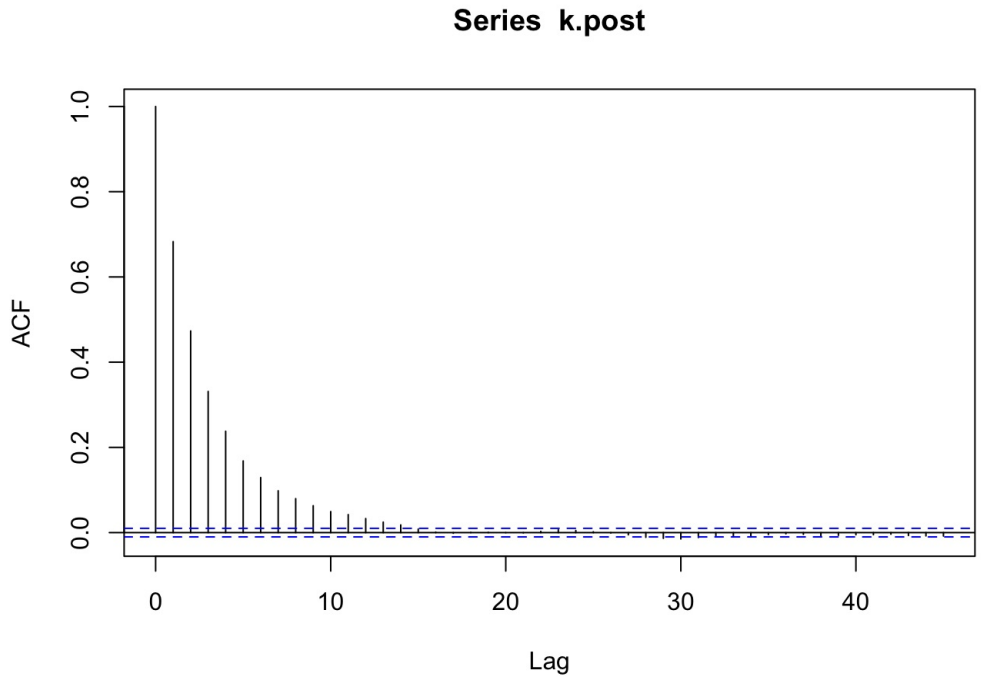



```
plot.ts(lam.post)
```

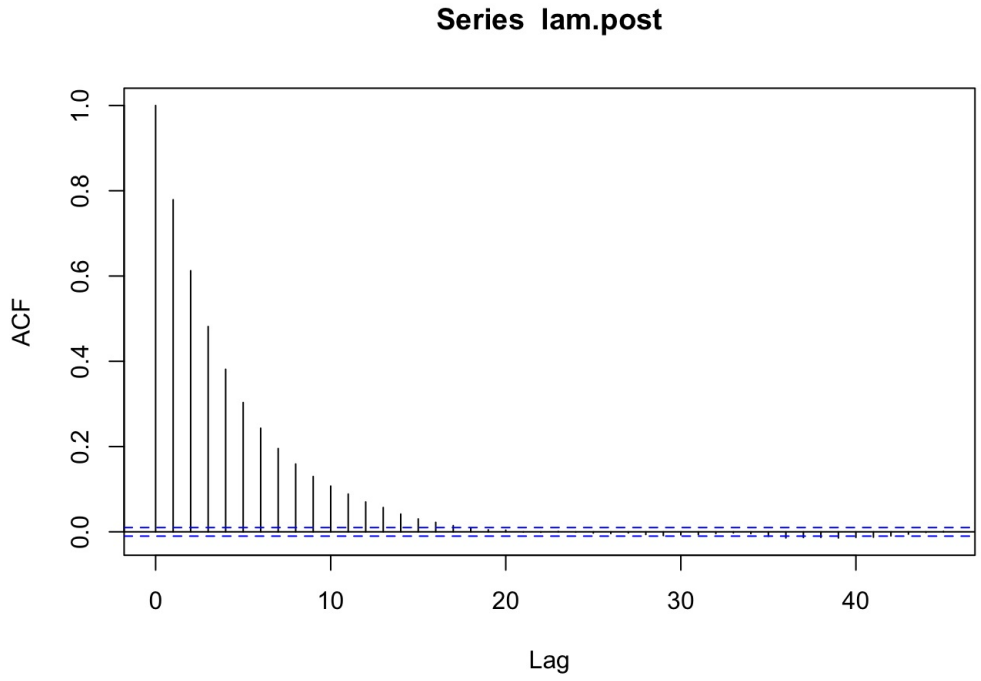


The trace plot seems well enough, the chain is exploring correctly the values and has converged. Now let's check the autocorrelation

```
acf(k.post)
```



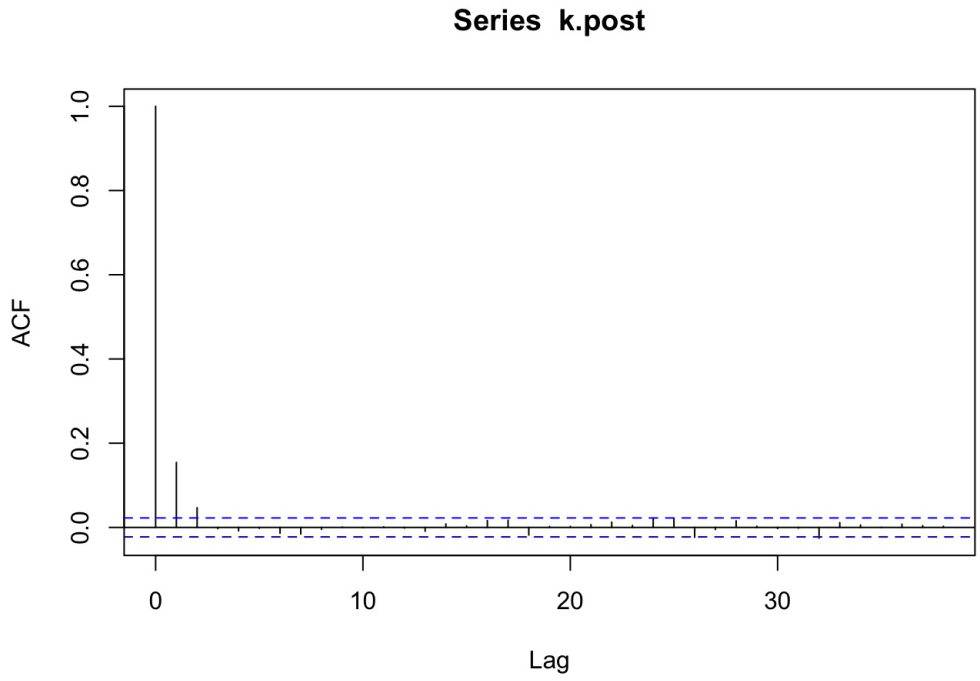

```
acf(lam.post)
```



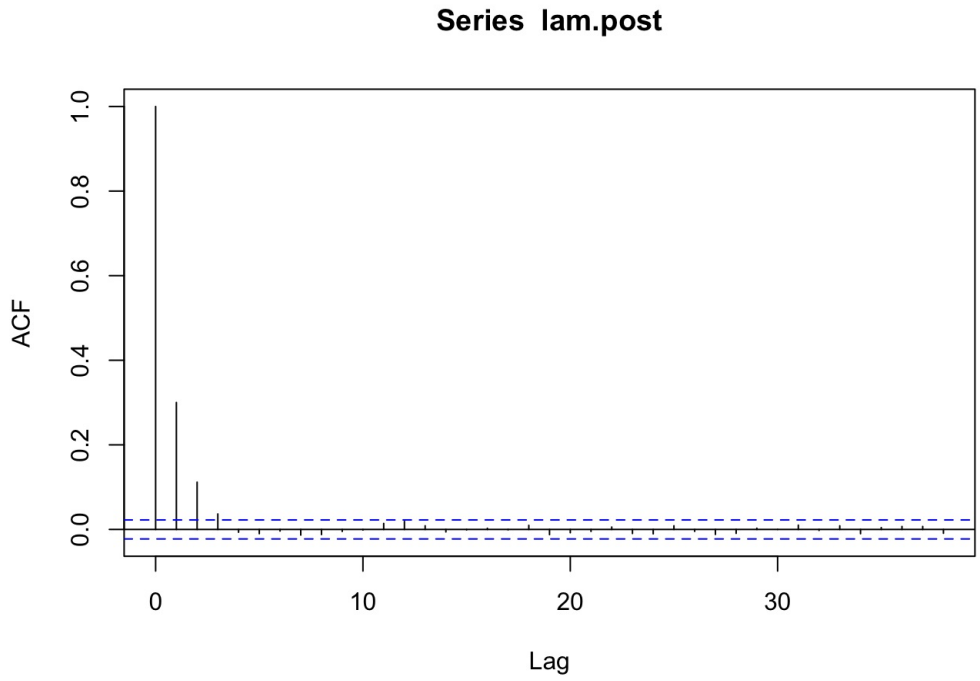
It seems that there is some correlation, we should apply some thinning

```
thin <- 5
k.post=k[seq(burnin+1,iters,by=thin)]
lam.post=lam[seq(burnin+1,iters,by=thin)]


acf(k.post)
```



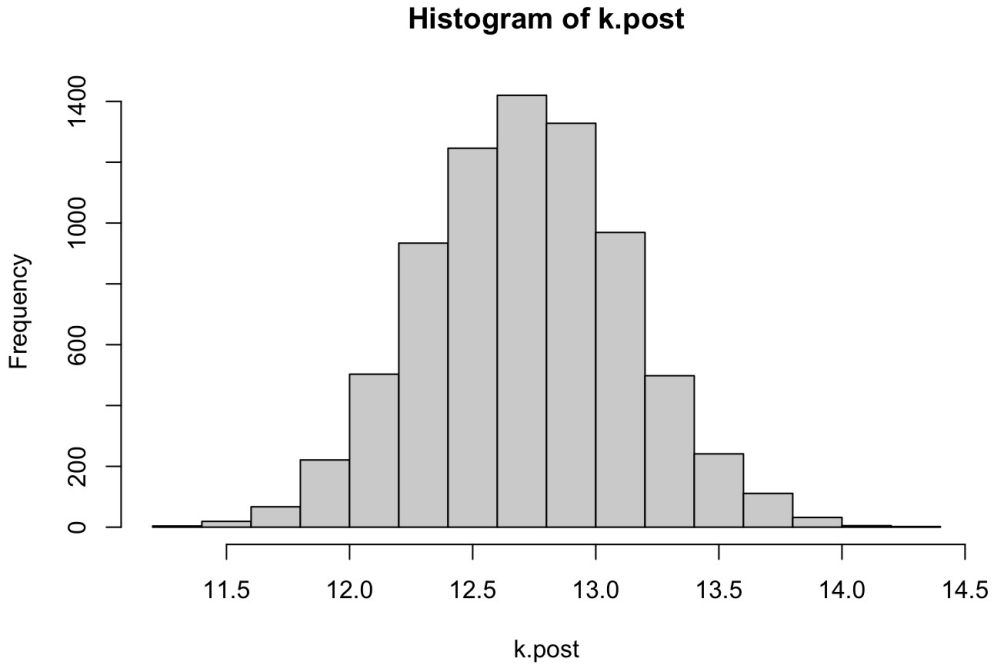

```
acf(lam.post)
```



Much better, now there almost no correlation.

Once we have ensured that the model has converged correctly, we can take a look at the parameters

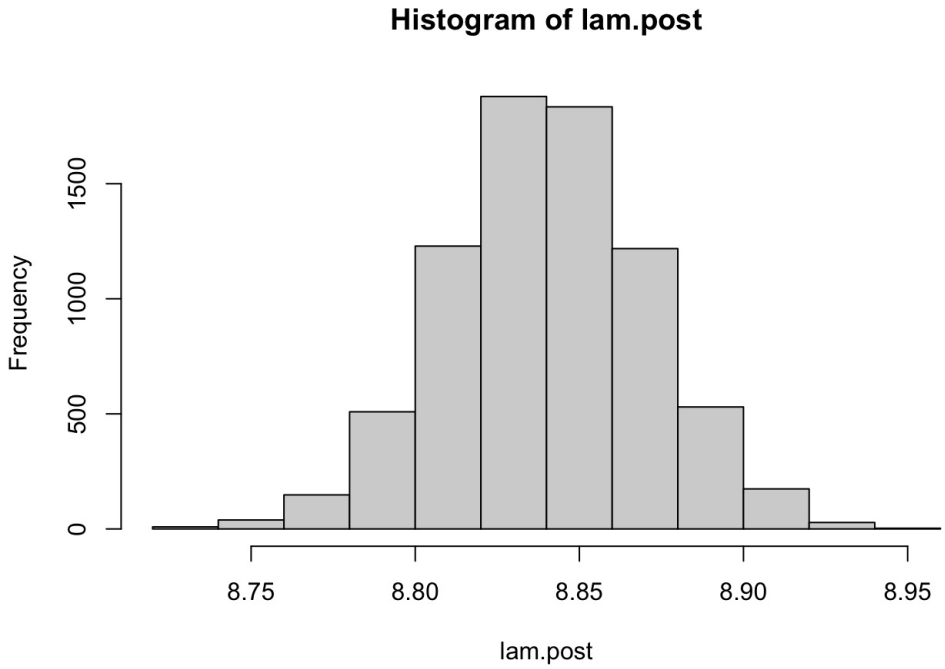
```
hist(k.post)
```




```
mean(k.post)
```

```
## [1] 12.71394
```

```
hist(lam.post)
```




```
mean(lam.post)
```

```
## [1] 8.84004
```

And also obtain the confidence intervals

```
quantile(k.post,c(0.025,0.975))
```

```
##     2.5%    97.5%
## 11.91062 13.54465
```

```
quantile(lam.post,c(0.025,0.975))
```

```
##     2.5%    97.5%
## 8.779800 8.900896
```

Predictive probabilities

Suppose we want to know if I am being healthy enough, it is estimated that around 7000 steps is considered healthy. We want to now X_{n+1} is larger that $\log(7000)$ (remember the transformation), so:

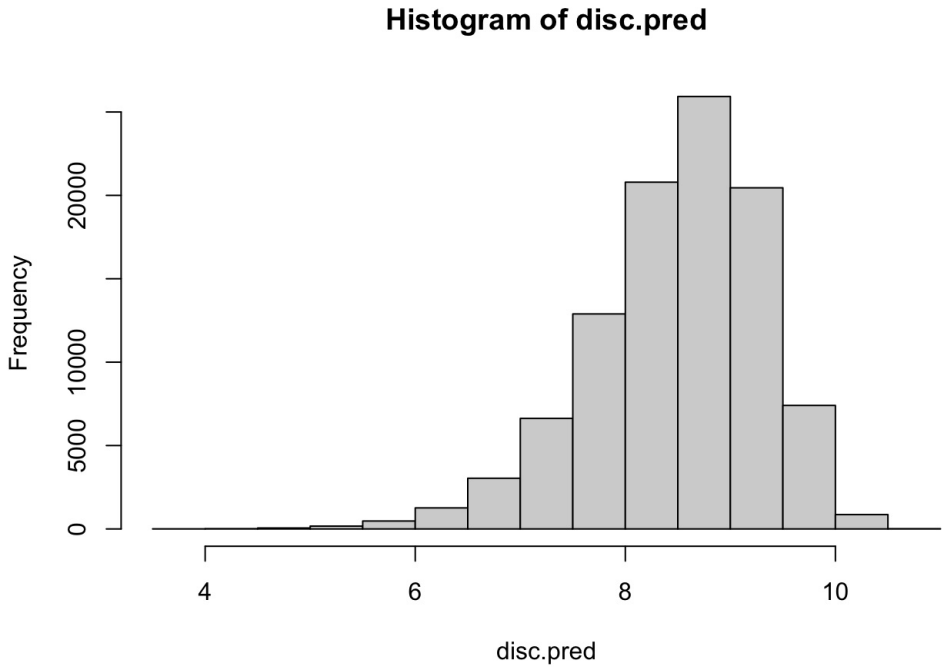
$$\mathrm{P}(X_{n+1} > \log(7000) \,|\, \text{data})$$

The predictive distribution can be obtained by:

$$f(x_{n+1} \,|\, \text{data}) = \int f(x \,|\, k, \lambda) f(k, \lambda \,|\, \text{data}) dk d\lambda$$

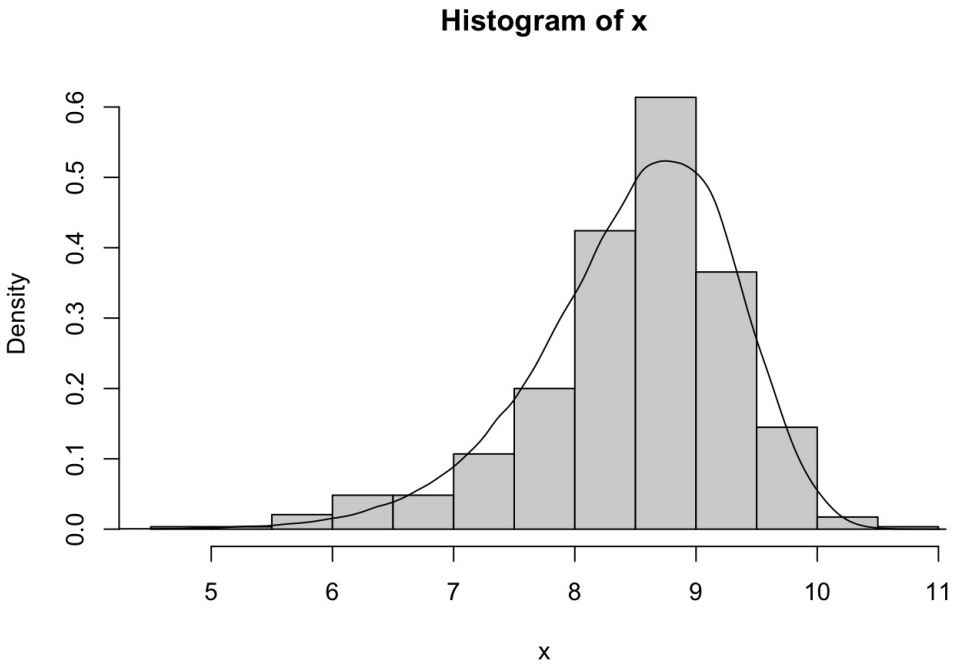
Since this density has no closed-form, we will obtain our posterior MCMC sample to obtain a sample of the predictive distribution:

```
M=100000
disc.pred=rweibull(M,shape=k.post, scale=lam.post)
hist(disc.pred)
```



Now, we can compared with our data

```
hist(x,freq=F, breaks=15)
lines(density(disc.pred))
```



And this is our prediction of the probability of making more than 7000 steps

```
mean(disc.pred>log(7000))
```

```
## [1] 0.3628
```

This probability is just 36.153%! Maybe I should improve my habits

Conclusion

During this project, I embarked on a comprehensive exploration of Bayesian modeling techniques applied to a real-world dataset of daily step counts, aiming to predict future levels.

Using two years of personal data extracted from Apple Health, I examined the step counts' distribution and opted for a Weibull distribution after evaluating various transformation techniques.

Throughout our analysis, I employed a Bayesian Weibull model, assuming log-normal priors for the shape and scale parameters of the distribution, which provided robustness and accommodated the data's skewness. Implementing a Markov Chain Monte Carlo (MCMC) using the Metropolis–Hastings algorithm, I iteratively refined our model parameters, ensuring convergence and adequacy through diagnostic checks, including trace plots and autocorrelation analyses.

The results highlighted an average daily step count, with corresponding confidence intervals for the shape and scale parameters. The predictive analysis, aimed at evaluating the likelihood of surpassing the health benchmark of 7,000 steps.

This case study not only showed the practical application of Bayesian statistics to real-world data but also demonstrated the findings that such an analytical approach can yield.

Future work could extend this model to include external variables like weather or day of the week distinctions, potentially providing a more understanding of step count variability.