Bayesian Monte-Carlo Risk Modeling of Asset Returns

Edric Paolo Franco

Masters in Statistics
edric.franco@mail.utoronto.ca

Abstract

This paper explores a Bayesian approach in order to estimate the Value-at-Risk (VaR), Conditional Value-at-Risk (CVaR) and Expected Returns of investing in the S&P 500 using Parametric Modeling and Monte-Carlo Methods. A common characteristic of Asset Returns is its heavy tails and one popular distribution to model this is the student's t-distribution. In this paper we will assume that S&P 500 returns follow a location-scale t-distribution and use a Bayesian approach to estimate the degrees of freedom ν, location μ and scale σ parameters of the model. Using the monte-carlo estimated parameters, we then calculate the VaR, CVaR and Expected Returns of investing in the S&P 500 in our model.

1 Introduction

One of the most popular ways to estimate the risk of an investment is by computing its Value-at-Risk (VaR) and Conditional Value-at-Risk (CVaR). The VaR at a level α is defined as the worst-case loss of our investment with probability atleast α. The CVaR or *expected shortfall* is the expected losses given that our losses are greater than the VaR. In other words, these two-metrics tells us the *worst-case* returns of our investment with high probability. This is important because it allows us to manage risk and estimate how much our investment could decline during stressed financial environments. There are multiple approaches in estimating these metrics such as using historical data to estimate the quantile of our net returns, parametric modeling and classical monte-carlo estimation. In this paper, we will use a Bayesian approach in order to create a model for the net-returns of S&P 500. We will assume that the net-returns of S&P 500 follows a *location-scale t-distribution*. Moreover, we will assign a prior distribution on the degrees of freedom ν, location μ and scale σ parameters using current literature available. Finally, we apply various monte-carlo methods in order to estimate the true parameters of our posterior distribution and use these estimated parameters in order to determine the various risk-metrics and expected returns of investing in the S&P 500. We will compare the results of our estimated parameters using *Simulated Annealing* wherein we find the parameters that maximize our posterior distribution and a *Metropolis Algorithm* wherein we estimate the expected value of our true parameters under the posterior.

In section 2, we illustrate the different characteristics of the net-returns of S&P 500 and illustrate the different assumptions of our model. We show how we selected the prior distribution of our parameters using current literature and used this in order to calculate the *unnormalized target posterior density* of our parameters given the S&P 500 net returns. We then introduce the monte-carlo algorithms that we will be using in order to find the parameters that maximize the posterior and the expected value of these parameters. In section 3, we illustrate our main findings and compare the calculations using our methods as compared to a simple historical data estimation of VaR and CVaR. We then show the various strengths and weaknesses of our approach and explore how the monte-carlo methods we used performed in estimating the parameters of our posterior distribution.

2 Methodology

We pull the Adjusted Closing Prices of the S&P 500 from Jan 2000 to Dec 2020 and we calculate the daily net returns of investing in the index. We can see from figure 1 that the net returns tend to exhibit characteristics such as *volatility clustering* and *heavy tails* wherein the extreme values are very large and the volatility of the returns tends to be changing throughout the series. In particular, we can see that during the 2008 financial crisis and 2020 pandemic that the daily net returns seems to be more volatile and have large extreme values. We can also see from the QQ-plot in figure 2 that the Normal Distribution does not have tails that are heavy enough to model the net returns. Hence, it seems that a heavy tailed distribution such as a student's t-distribution is a reasonable model for the net returns. Note that one draw-back when using a heavy tailed distribution as our model is that it is not *geometrically ergodic* and could lead to unstable estimates. However, as stated by this paper [**ergodic**], we can see that the student's t-distribution is still *polynomial ergodic* of order $\nu/2$ for the Metropolis random walk and we still have some guarantees such as bounds on our Monte Carlo error under certain conditions.

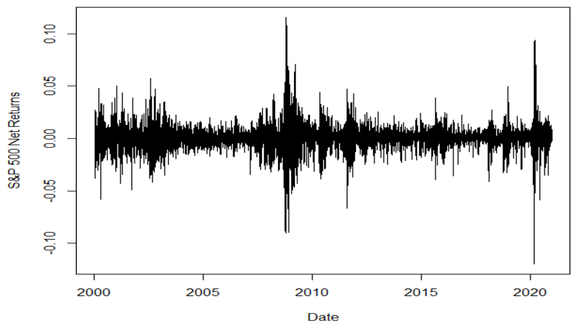

Figure 1: This figure shows the Daily net returns of S&P 500 from Jan 2000 - Dec 2020

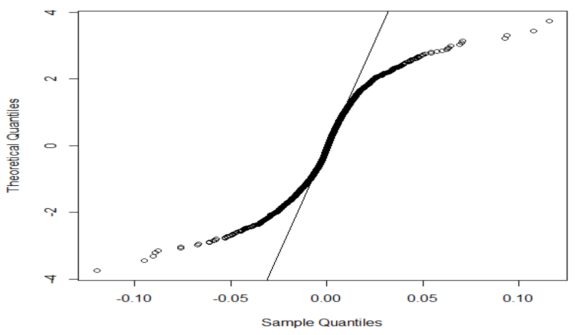

Figure 2: This figure shows the sample quantiles of of the S&P 500 net returns as compared to the true quantiles of a Normal distribution

2.1 Model Assumptions

We will assume that the S&P 500 net returns, y_i, follows a location-scale t-distribution, $y_i \sim t(\nu, \mu, \sigma)$. Assuming that the daily net returns are independent, we can then compute our likelihood function, $\mathrm{P}(Y_1, ..., Y_n|\nu, \mu, \sigma)$, as follows:

$$P(Y_1, ..., Y_n|\nu, \mu, \sigma) = (\frac{\Gamma(\frac{\nu+1}{2})}{\sigma\sqrt{\nu\pi}\Gamma(\frac{\nu}{2})})^n \cdot \prod_{i=1}^{n} (\frac{\nu + (\frac{y_i-\mu}{\sigma})^2}{\nu})^{-\frac{\nu+1}{2}}$$

2.1.1 Selecting our Prior Distributions

We select prior distributions for our parameters that does not dominate our likelihood so that we could successfuly learn from our data. According to this paper [**nu˙prior**], it is suggested that a good prior for the degrees of freedom of a t-distribution is the Gamma(2, 1/10) distribution (i.e $\nu \sim$ Gamma(2, $\frac{1}{10}$)) because it covers a large range of relevant values for the degrees of freedom and it does not explode to large values which is good because for really large values of ν, the t-distribution would approximately be a normal distribution which we have shown is not a good fit for our data. Moreover, we also assume that the location parameter of our model comes from the Normal(0, 100) distribution (i.e $\mu \sim$ N(0,100)) which was shown to have decent results under the skew-t model from this paper [**mu˙prior**]. Finally, we assume a prior distribution of a half-t distribution (i.e absolute value of a Student-t distribution centered at 0) with scale 1 as it was shown to be a good prior for variances in Bayesian Hierarcichal modeling with a small number of groups from this paper [**sig˙prior**]. We summarize our selection of prior distributions for our different parameters below:

$$P(\mu) \propto e^{\frac{-\mu^2}{200}}$$

$$P(\nu) \propto \nu \cdot e^{\frac{-\nu}{10}}$$

$$P(\sigma|\nu) \propto (1 + \frac{\sigma^2}{\nu})^{\frac{-(\nu+1)}{2}}$$

2.1.2 Deriving our Target Posterior Distribution

We can then use the likelihood function and prior probabilities to calculate our *unnormalized posterior distribution* that we will try to simulate from using MCMC algorithms.

$$P(\nu, \mu, \sigma|Y_1, ..., Y_n) \propto P(Y_1, ..., Y_n|\nu, \mu, \sigma)P(\sigma|\nu, \mu)P(\nu|\mu)P(\mu)$$

Note that when running our Monte Carlo algorithms, it is useful to work on the log-scale as it allows for better numerical stability and does not change the maximum of our posterior density. We derive this below:

$$log(P(\nu,\mu,\sigma|Y_1,...,Y_n)) \propto n\cdot log(\frac{\Gamma(\frac{\nu+1}{2})}{\sigma\sqrt{\nu\pi}\Gamma(\frac{\nu}{2})}) - \frac{\nu+1}{2}(\sum_{i=1}^{n}(\frac{\nu+(\frac{y_i-\mu}{\sigma})^2}{\nu})) - \frac{\nu+1}{2}log(1+\frac{\sigma^2}{\nu}) - \frac{\mu^2}{200} + log(\nu) - \frac{\nu}{10}$$

2.2 Monte-Carlo Methods for Estimating our Parameters

We will now use MCMC in order to estimate the parameters of our posterior distribution. We will use a Componentwise Random Scan Metropolis algorithm (symmetric normal proposal distribution) in order to estimate the expected value of each of our 3 parameters and use Simulated Annealing in order to estimate these 3 parameters by maximizing the posterior distribution. After obtaining the final estimates for the parameters of our model ($\nu*,\mu*,\sigma*$), we can then estimate VaR, CVaR and Expected Returns by computing the quantiles of the t($\nu*,\mu*,\sigma*$) distribution.

2.2.1 Random Scan Componentwise Metropolis Algorithm

Given our unnormalized posterior density, it seems that a normal proposal distribution with an optimal spread is sufficient for our Metropolis algorithm. We tweak the spread so that we have an acceptance rate close to 0.234 [**opt**]. Moreover, it is important to consider that the range of values of our 3 parameters are different wherein the degrees of freedom, ν, tends to range from 2 to 3 while the location, μ, tends to range from 0.0002 to 0.001 and the scale, σ, tends to range from 0.0065 to 0.0075. Hence, it is reasonable to use a smaller spread for the proposal distribution of μ and σ. For this exercise, we will work on a log-scale and accept the proposal to move from the current state, X_{n-1}, to the proposed state, $Y_n \sim N(X_{n-1},\sigma^2)$, if log(g($Y_n$)) - log(g($X_{n-1}$)) is greater than the log of a generated uniform(0, 1) random variable (where g is our unnormalized target density). Moreover, we will use a spread of $\sigma = 1/3$ for the degrees of freedom, $\sigma = 1/2400$

for our location parameter and $\sigma = 1/1500$ for the scale parameter for the spreads of our Normal proposal distribution which we will show later gives us a decent acceptance rate of 0.27 and good mixing of our chains.

2.2.2 Simulated Annealing

Simulated annealing is a method that we can use in order to find the parameters that maximizes our posterior distribution. The idea is, we start with a *flatter* version of our target density by applying a cooling scheme, $\pi(x)^{1/\tau}$, where π is the target density that we want to maximize and τ is the temperature that allows us to make our density flatter (for large τ) or more peaked (for small τ). It is ideal to have a large τ at the start of our chain as it allows us to span a larger area in our target distribution. As we advance through the chain, we constantly decrease τ to make our density more peaked and allow us to converge to the highest mode of our density. There are multiple ways we could update τ throughout the series and in this paper we will explore *exponential cooling* where we decrease τ exponentially and *linear cooling* where we decrease τ linearly for each iteration. For this simulation, a final temperature (τ_n) of 0.002 and initial temperature (τ_0) of 100 seems to give us decent results.

3 Results

In the following subsections, we show the results of our MCMC algorithms, compare the estimates of our parameters and investigate their performances.

3.1 Expected value of the parameters using Metropolis Algorithm

We can see from figure 3 below that the trace plot of ν seems to have good *mixing* where it tends to show constant mean and variance throughout the chain. We can see from figure 4 that the ACF of ν seems to go down quickly and is very minimal after 150 lags which is a good sign that our algorithm has a small *varfact* and standard error. The other 2 parameters showcase similar characteristics as can be seen in the Appendix. To obtain our final estimate, a burn-in of

20,000 seems to be sufficient based on the trace plots.

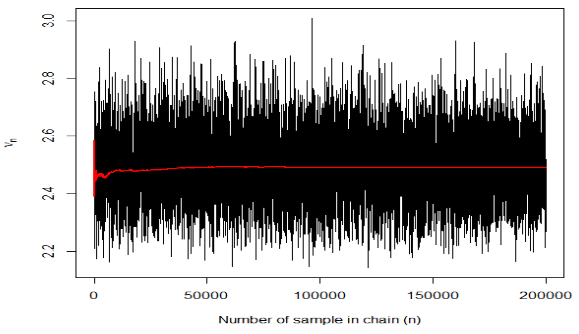

Figure 3: This figure shows the trace plot of our chain for ν

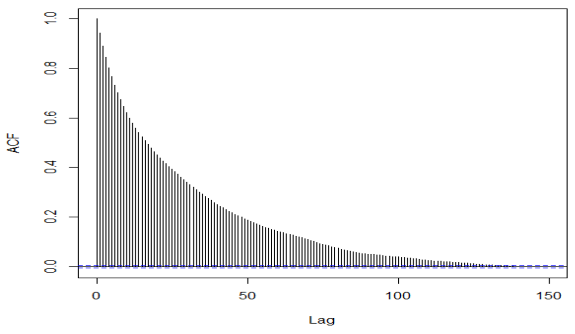

Figure 4: This figure shows the ACF plot of our chain for ν

After re-running our algorithm 20 times, we can see our final Monte-Carlo estimate for our parameters below and see that our algorithm seems to work well in the fact that we have a small standard error for all 3 parameters:

Using 20 chains of 200,000 iterations each		
Parameter	Final MC estimate	Approximate SE
ν	2.4899	0.0006
μ	0.00065	0.0000002
σ	0.0069	0.0000008

Table 1: This table shows our Final Monte-Carlo Estimate using Componentwise Metropolis Algorithm together with its approximate Standard Error

3.2 Finding the parameters that maximizes our Posterior using Simulated Annealing

We can see from below trace plots that both exponential and linear cooling seems to work well and give us similar results. In particular, we can see that our chain do not get stuck towards a point and seems to converge to the highest mode of our density as we go through the chain. In this case, exponential cooling seems to converge faster (as expected) than linear cooling. Given our current parameters, it seems that exponential cooling slightly edges linear cooling in terms of efficiency and confidence in the estimate.

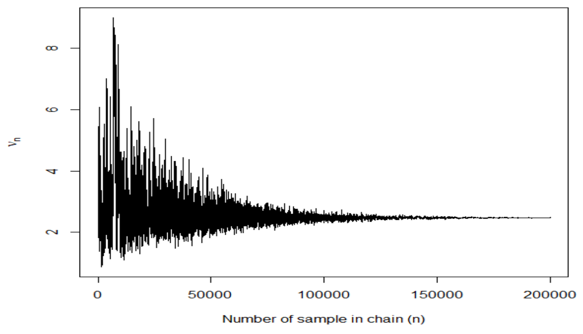

Figure 5: This figure shows the trace plot of our chain for ν using Exponential Cooling

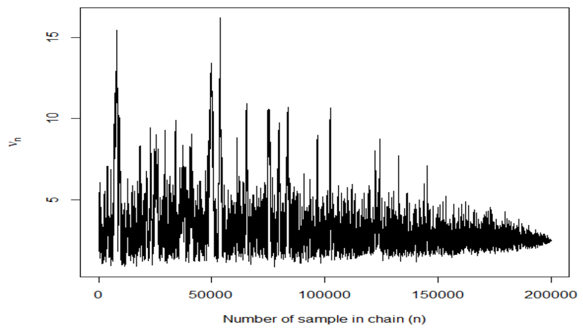

Figure 6: This figure shows the trace plot of our chain for ν using Linear Cooling

In order to show that our estimates are not due to chance, we re-run our algorithm 20 times and calculate the sample standard deviation of the 20 estimates. We can see from the small standard deviation for both methods that our algorithm seems to work well and does not depend on the starting point of our chain. We can see from table 2 that the sample standard deviation of the estimates for exponential cooling seems to be smaller than the sample standard deviation for linear cooling which again hints that exponential cooling is slightly better for this problem.

Parameter	Exp Cooling MC	Exp Cooling SD	Linear Cooling MC	Linear Cooling SD
ν	2.488	0.0057	2.478	0.0268
μ	0.000649	0.0000055	0.00066	0.0000207
σ	0.0069	0.000006	0.07	0.000032

Table 2: This table shows our estimates for the parameters using Exponential and Linear cooling together with the standard deviation of re-running the algorithm 20 times

3.3 Comparison of VaR, CVaR and Expected returns from the different models and Historical approach

We could see from figure 7, that our model using the estimated parameters of the t-distribution of our MC algorithms tend to show similar density as compared to historical data. Moreover, it seems that the 3 MC approaches provide similar models but exponential cooling seems to be the closest to historical data.

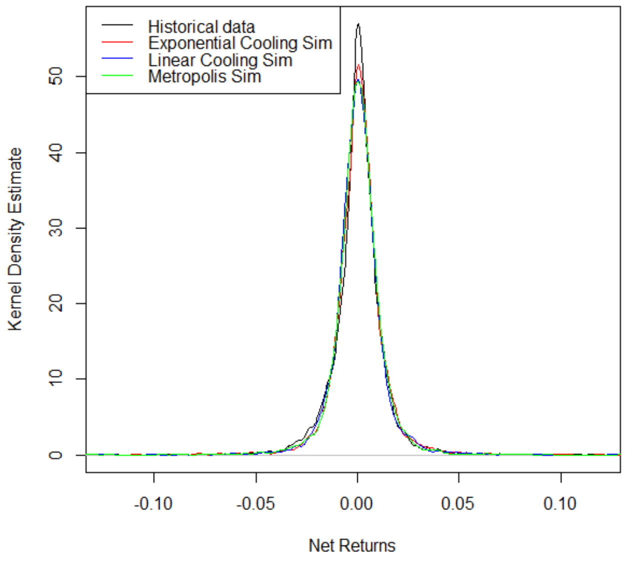

Figure 7: This figure illustrates the KDE of the historical data and simulating using the estimated parameters of the location-scale t-distribution from our MCMC algorithms

Finally, we can see from table 3 that the VaR, CVaR and Expected returns using the estimated parameters of the different algorithms tends to show similar results and also seems to agree with our historical data. This seems to show that all of our algorithms worked well in estimating the parameters of our model with Exponential Cooling and simple Componentwise Metropolis providing the closest estimates compared to historical data.

Method of Estimation	$VaR_{0.05}$	$CVaR_{0.05}$	Expected Returns
Historical Data	0.019	0.0303	0.00059
Exponential Cooling	0.0171	0.0314	0.000649
Linear Cooling	0.0173	0.0317	0.000662
Simple Metropolis	0.0171	0.0314	0.000647

Table 3: This table shows our estimate for VaR, CVaR (at a 5% level) and Expected Returns using the estimated parameters from our MCMC algorithms

4 Conclusions

In this paper, we illustrated how we can model asset returns using a Bayesian approach by assuming a heavy tailed distribution to our historical net returns dataset. Moreover, we have shown that even though our target density has fat-tails and does not have the guarantees of geometric ergodicity, we are still able to get decent results using the Metropolis and Simulated Annealing algorithms. When generating our chains using MCMC algorithms, it is important to consider the range of values of the different parameters as using a smaller spread for the proposal distributions of parameters with smaller values seems to provide better mixing and convergence of our algorithm. We have shown that estimating the parameters of our model using a simple Metropolis Algorithm shows similar results to Simulated Annealing when modeling the net-returns of S&P 500 using a t-distribution. Exponential Cooling for Simulated Annealing seems to show great results in our model with quick convergence and small deviations when generating multiple estimates from the algorithm. Overall, using a Bayesian model for our asset-returns and estimating the parameters using MCMC seems to show reasonable estimates for VaR, CVaR and Expected Returns which agrees with our historical dataset.

A Appendix and Source Code

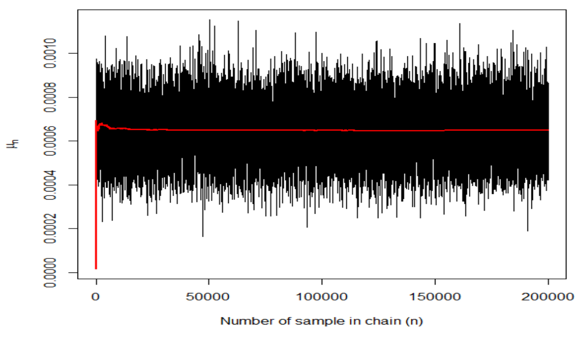

Figure 8: This figure shows the trace plot of our chain for μ

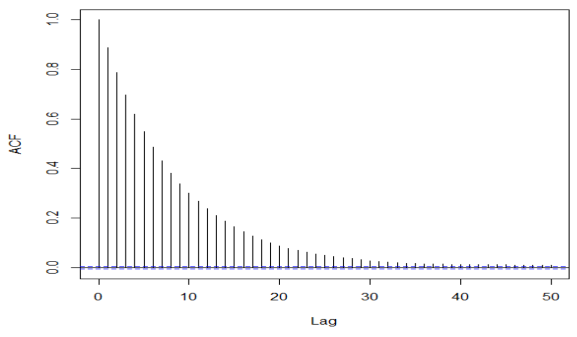

Figure 9: This figure shows the ACF plot of our chain for μ

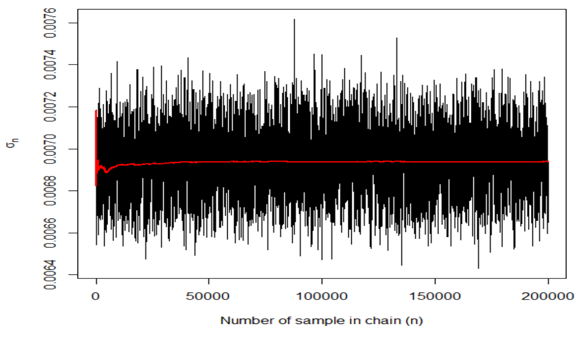

Figure 10: This figure shows the trace plot of our chain for σ

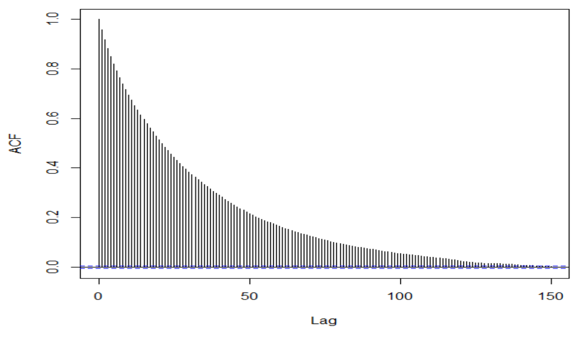

Figure 11: This figure shows the ACF plot of our chain for σ

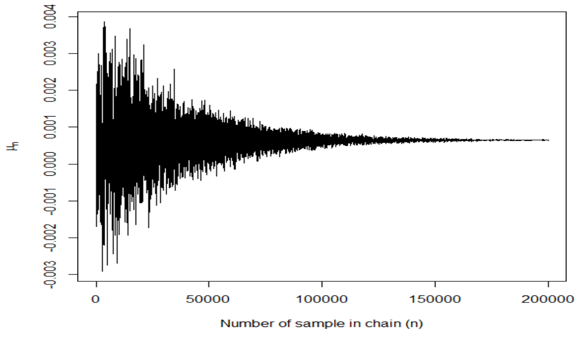

Figure 12: This figure shows the trace plot of our chain for μ using Exponential Cooling

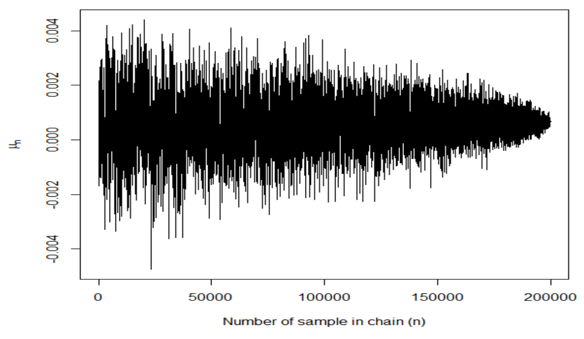

Figure 13: This figure shows the trace plot of our chain for μ using Linear Cooling

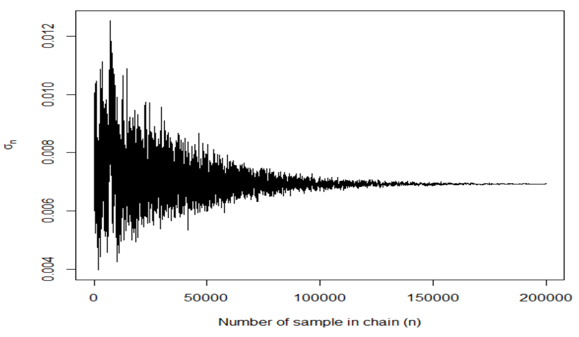

Figure 14: This figure shows the trace plot of our chain for σ using Exponential Cooling

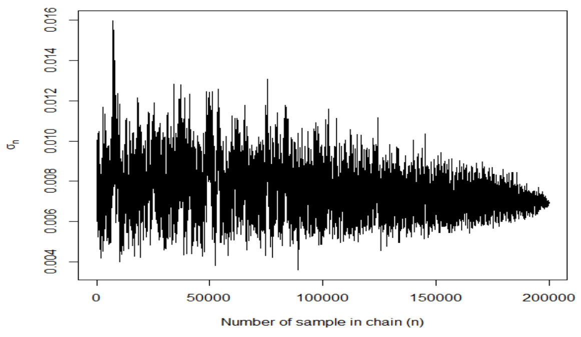

Figure 15: This figure shows the trace plot of our chain for σ using Linear Cooling