Chapter 15

Hamiltonian Monte Carlo

15.1 Cerebral malaria: coding up samplers

Suppose you work for the WHO researching malaria. In particular, it is your job to produce a model for the number of cases of cerebral malaria in a large country. Cerebral malaria is one of the most severe complications resulting from infection with Plasmodium falciparum malaria, and without treatment invariably causes death. However, even for patients receiving treatment there is still a significant chance of permanent cognitive impairment.

You decide to model the number of cases of cerebral malaria ($X = 5$) as being from a joint normal distribution along with the number of all malaria cases ($Y = 20$). The mean number of cases of cerebral malaria is μ_c, and the mean cases of all malaria is μ_t. If we assume an (improper) uniform prior distribution on these quantities and assume that the correlation between cerebral and total cases is known ($\rho = 0.8$) the posterior is:

$$\begin{pmatrix} \mu_t \\ \mu_c \end{pmatrix} \sim \mathcal{N}\left[\begin{pmatrix} 20 \\ 5 \end{pmatrix}, \quad \begin{pmatrix} 2 & 0.8 \\ 0.8 & 0.5 \end{pmatrix} \right]$$

where all quantities are measured in units of "000s".

Note that this example does not test your ability to do Bayesian inference (because we have already provided the exact form of the posterior distribution). Rather its purpose is allow you to compare the performance of a number of different sampling algorithms.

Problem 15.1.1. Use your statistical software of choice generate 100 independent samples of (μ_t, μ_c). Draw a scatter plot of your (μ_t, μ_c) samples, with lines connecting consecutive points. How close are the sample-estimated means to the true means? (Hint: to do this in R you will need to use the MASS package:

```
library(MASS)
```

```
Sigma <- matrix(c(2, 0.8, 0.8, 0.5), 2, 2)
mvrnorm(n=100, c(20, 5), Sigma)
```

)

The independent sampler is the gold standard sampling routine here. Its samples quickly traverse the posterior space (Figure 15.2), meaning that we get an accurate (and unbiased) estimate of the posterior mean for only 100 samples (Figure 15.1).

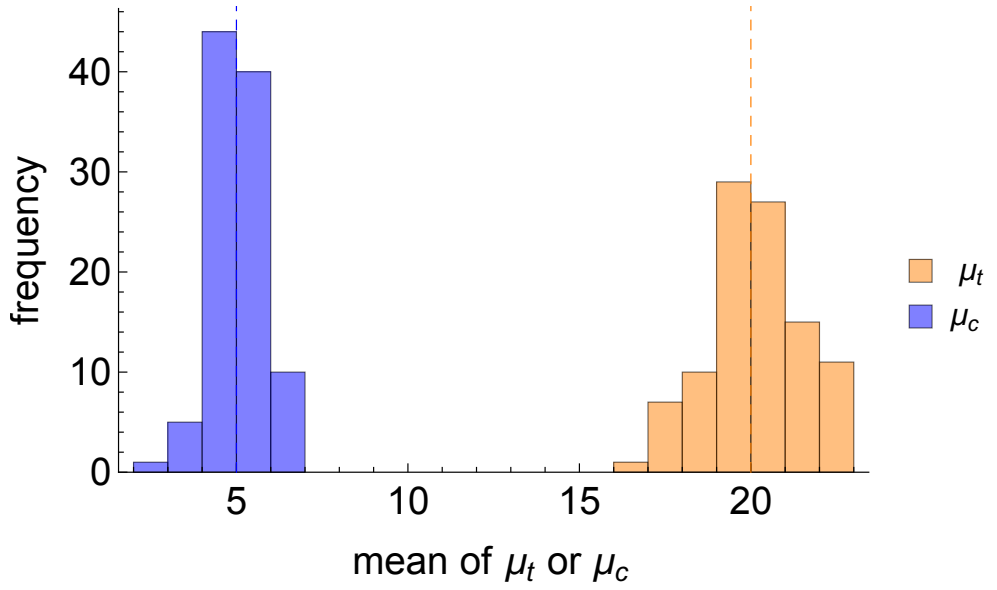

Figure 15.1: The sampling distribution for 100 samples from the posterior using the independent sampler.

Problem 15.1.2. Code up a Random Walk Metropolis sampler for this example. This is composed of the following steps:

1. Create a proposal function that takes a current value of $\theta = (\mu_t, \mu_c)$ and outputs a proposed value of these using a multivariate normal centred on the current estimates. (Here use a multivariate normal proposal with an identity covariance matrix.)

2. Create a function which takes as inputs $\theta^{current}$ and $\theta^{proposed}$, and outputs the ratio of the posteriors of the proposed value to the current one (Hint: to do this in R you will need to use the following to calculate the value of the posterior at (x, y):

```
library(mvtnorm)

Sigma <- matrix(c(2, 0.8, 0.8, 0.5), 2, 2)
dmvnorm(c(x, y), c(20, 5), Sigma)
```

).

3. Create an accept/reject function which takes as inputs $\theta^{current}$ and $\theta^{proposed}$, then uses the above ratio function to find: $r = \frac{\theta^{proposed}}{\theta^{current}}$; then compares r with a uniformly-distributed random number u between 0 and 1. If $r > u \implies$ output $\theta^{proposed}$; otherwise output $\theta^{current}$.

4. Combine the proposal function along with the accept/reject function to make a function that takes as input $\theta^{current}$, proposes a new value of θ, then based on r moves to that new point or stays in the current position.

5. Create a function called "RWMetropolis" that takes a starting value of θ and runs for n steps.

Use your "RWMetropolis" function to generate 100 samples from the posterior starting from $(\mu_t, \mu_c) = (10, 5)$. Draw a line plot of your (μ_t, μ_c) samples. How do your estimates of the posterior mean from Random Walk Metropolis compare with the true values? Why is there a bias in your estimates, and how could this be corrected?

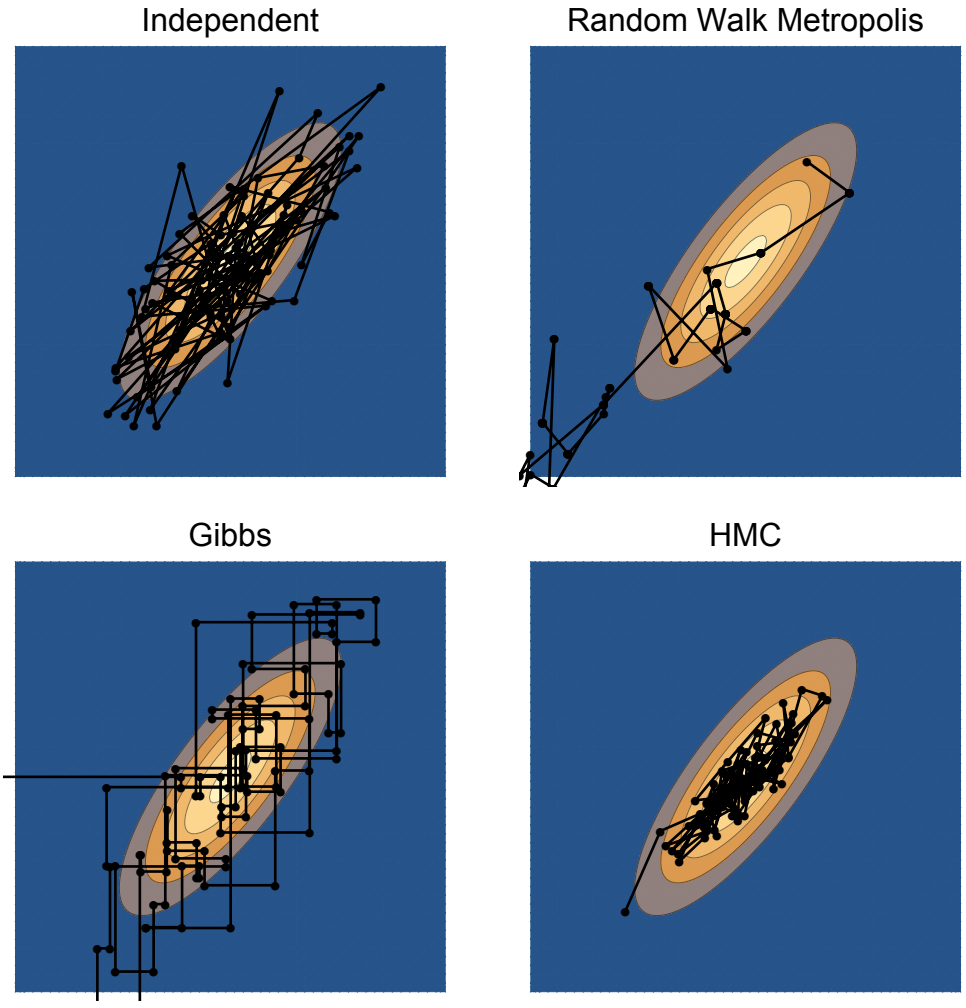

Figure 15.2: 100 samples from the posterior for the malaria example using four different sampling algorithms.

The Random Walk Metropolis sampler is significantly less efficient at exploring posterior space (Figure 15.2) than the independent sampler. This is because of its random walk nature; it essentially

ignores the posterior geometry! It is also because dependent sampling, in general, will never match the performance of an independent sampler.

The estimates of the mean of μ_t should be downwardly-biased (Figure 15.3) because we started the sampler at $\mu_t = 10 < 20$. What we should do is remove the first half (or so) iterations of the chain, where it has yet to converge to a stationary distribution.

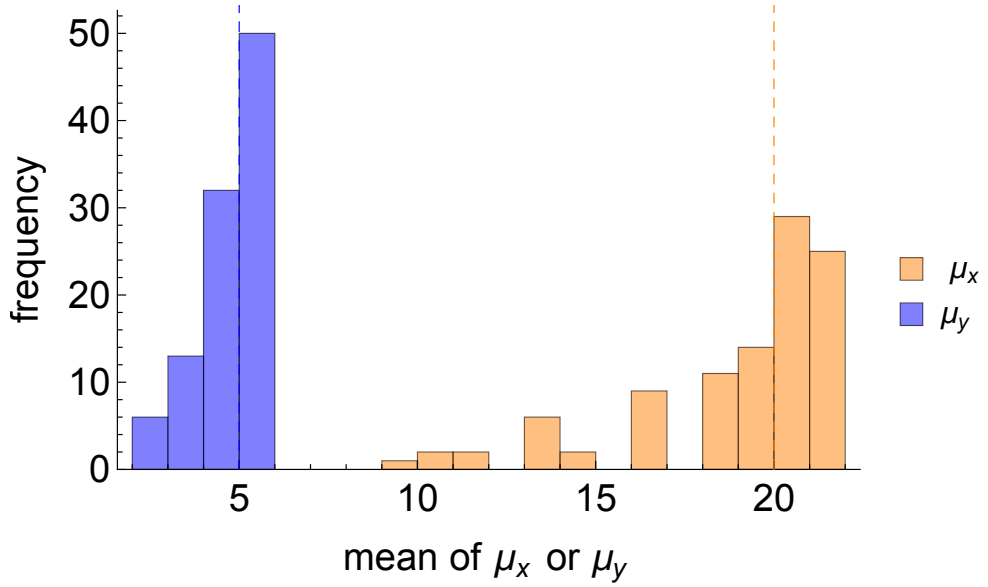

Figure 15.3: The sampling distribution for 100 samples from the posterior using Random Walk Metropolis sampler.

Problem 15.1.3. For your 100 samples using Random Walk Metropolis calculate the percentage of accepted steps.

This is about 30% (Figure 15.4). This relatively low acceptance rate (although not far from optimality for RWM) means that the sampler is slow to explore the posterior.

Problem 15.1.4. Create a function that calculates Gelman's $\hat{R}$ for each of (μ_t, μ_c) using:

$$\hat{R}(t) = \sqrt{\frac{W(t) + \frac{1}{T}(B(t) - W(t))}{W(t)}} \tag{15.1}$$

where,

$$W(t) = \frac{1}{m}\sum_{j=1}^{m} s(t)_j^2 \tag{15.2}$$

measures the within-chain variance at time t averaged over m chains, and $s(t)_j^2$ is the sample variance of chain j. And:

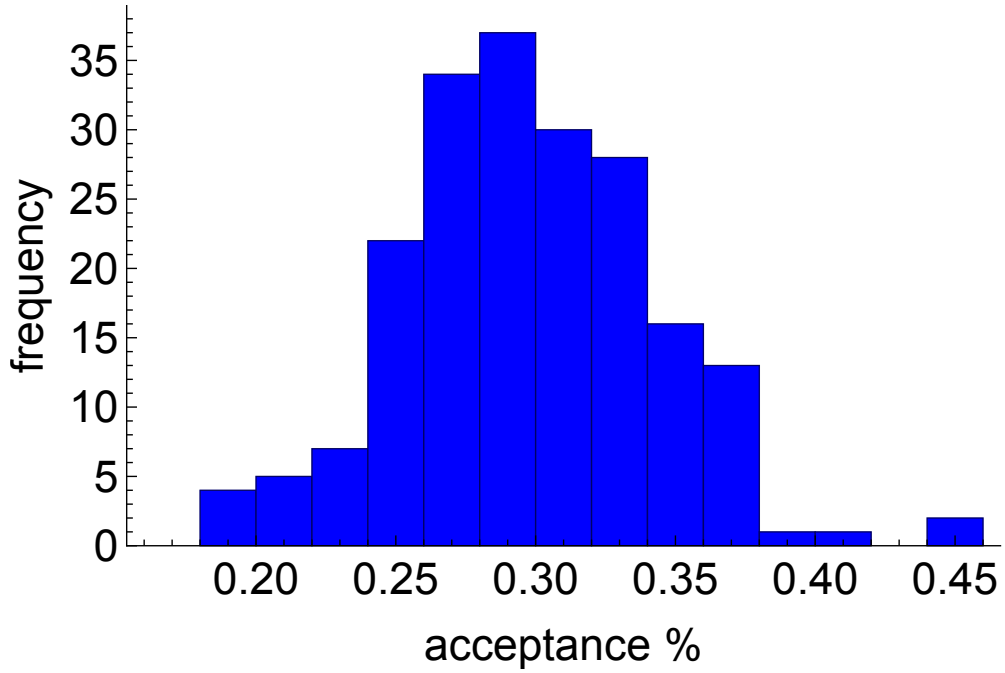

Figure 15.4: The acceptance rate across 200 Random Walk Metropolis Markov Chains, in each case using 100 steps.

$$B(t) = \frac{t}{m-1}\sum_{j=1}^{m}(\overline{\theta(t)}_j - \overline{\theta(t)})^2 \tag{15.3}$$

measures the between-chain variance at time t. Here $\overline{\theta(t)}_j$ is the average value of a parameter in chain j, and $\overline{\theta(t)}$ is the average value of a parameter across all chains. (Hint 1: first create two separate functions that calculate the within and between chain variance. Hint 2: you will obtain a value of $\hat{R}$ for each of (μ_t, μ_c).)

To do this first create a function in R that calculates the within chain variance,

```
fWithin <- function(lSamples){
  return(mean(sapply(lSamples, var)))
}

## Testing it
lSamples <- lapply(seq(1, 10, 1), function(i) rbinom(100, 100, 0.5))
fWithin(lSamples)
```

Then another that calculates the between chain variance,

```
fBetween <- function(lSamples){
  lMean <- sapply(lSamples, mean)
  aMean <- mean(lMean)
```

```
  m <- length(lSamples)
  t <- length(lSamples[[1]])
  return((t / (m - 1)) * sum((lMean - aMean) ^ 2))
}
```

Then putting these together we get a function to calculate $\hat{R}$,

```
fRhat <- function(lSamples){
  W <- fWithin(lSamples)
  B <- fBetween(lSamples)
  t <- length(lSamples[[1]])
  return(sqrt((W + (1 / t) * (B - W)) / W))
}
```

Problem 15.1.5. Start all eight chains at $(\mu_t, \mu_c) = (20, 5)$ and calculate $\hat{R}$ for a per chain sample size of 5. Does this mean we have reached convergence?

If the chains all begin in the same area of parameter space $\implies$ we get a false impression of convergence. This is why it's so important to start them at over-dispersed locations.

Problem 15.1.6. Using eight chains calculate $\hat{R}$ for each of (μ_t, μ_c) for a sample size of 100. This time make sure to start your chains in overdispersed positions in parameter space. Use a random number from a multivariate normal centred on the posterior means with a covariance matrix of 40 times the identity matrix.

The chains should be starting at dispersed locations in parameter space (Figure 15.5), otherwise the value of $\hat{R}$ obtained will be biased downwards. After a sample size of 100 the chains should be nearing convergence, and should have a value of $\hat{R}$ of about 1.05-1.06 (Figure 15.6).

Problem 15.1.7. After approximately how many iterations does Random Walk Metropolis reach $\hat{R} < 1.1$?

After about 150-250 iterations the chains should have roughly reached $\hat{R} < 1.1$ (Figure 15.6).

Problem 15.1.8. The conditional distributions of each variable are given by:

$$\mu_t \sim \mathcal{N}(20 + 1.6(\mu_c - 5), (1 - 0.8^2)2)$$
$$\mu_c \sim \mathcal{N}(5 + 0.4(\mu_t - 20), (1 - 0.8^2)0.5)$$

Use this information to code up a Gibbs sampler, again starting at $(\mu_t, \mu_c) = (10, 5)$. (Hint: in R use `rnorm`, or equivalent to create two functions: one that produces draws of μ_t given μ_c; and the other that produces draws of μ_c given μ_t. Then create a function that cycles between these updates. Make sure to always draw samples using the most recent values of (μ_t, μ_c)).

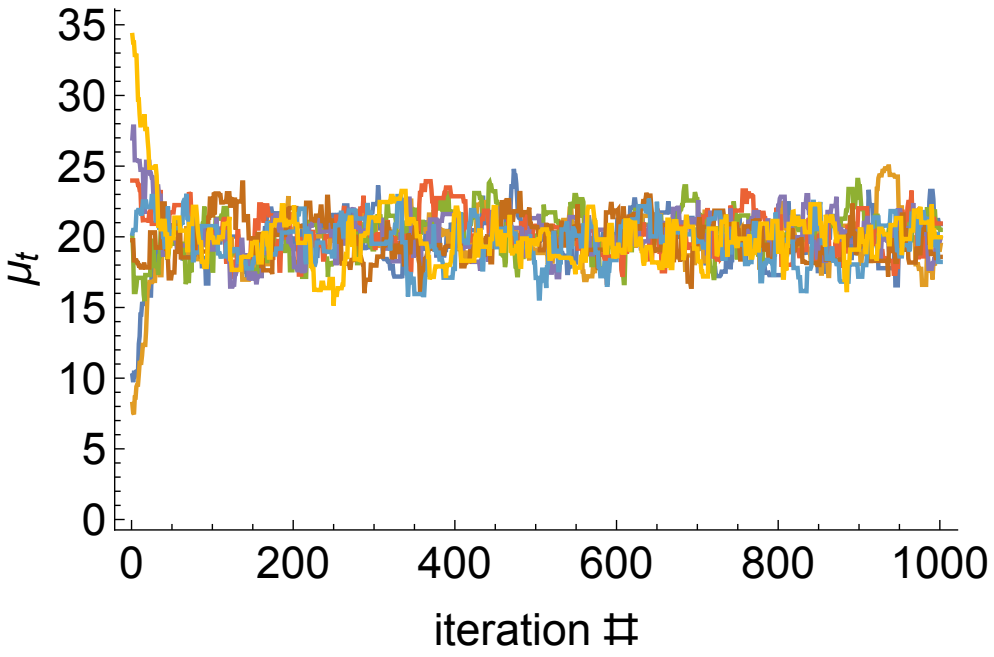

Figure 15.5: The path taken by each chain in μ_t space.

Problem 15.1.9. Use your Gibbs sampler to draw 100 samples. Draw a scatter plot of your (μ_t, μ_c) samples with lines connecting consecutive points. Discarding the first 50 observations, how do the estimates of the mean of each parameter compare with their true values?

After a short warm-up period the samples from the Gibbs algorithm quickly converge towards the posterior (Figure 15.2), with the resultant estimates of the mean reflecting this (Figure 15.7).

Problem 15.1.10. Generate 200 samples from each of your Random Walk Metropolis and Gibbs samplers. Discard the first 100 observations of each as warm-up. For each calculate the error in estimating the posterior mean of μ_t. Repeat this exercise 40 times; each time recording the error. How does their error compare to the independent sampler?

Each of the three algorithms is essentially unbiased after we discard the warm-up (Figure 15.8). I obtained an error of about 0.12 for the independent sampler; 0.51 for Random Walk Metropolis; and 0.40 for Gibbs.

Problem 15.1.11. Repeat Problem 15.10 to obtain the average error in estimating the posterior mean of μ_t across a range of sample sizes $n = 5$ to $n = 200$.

The error from Random Walk Metropolis and Gibbs is always higher than the independent sampler (Figure 15.9). After a sample size of about 20, the Gibbs outperforms Random Walk Metropolis.

Problem 15.1.12. Using the results from the previous question estimate the effective sample size for 150 observations of the Random Walk Metropolis and Gibbs samplers.

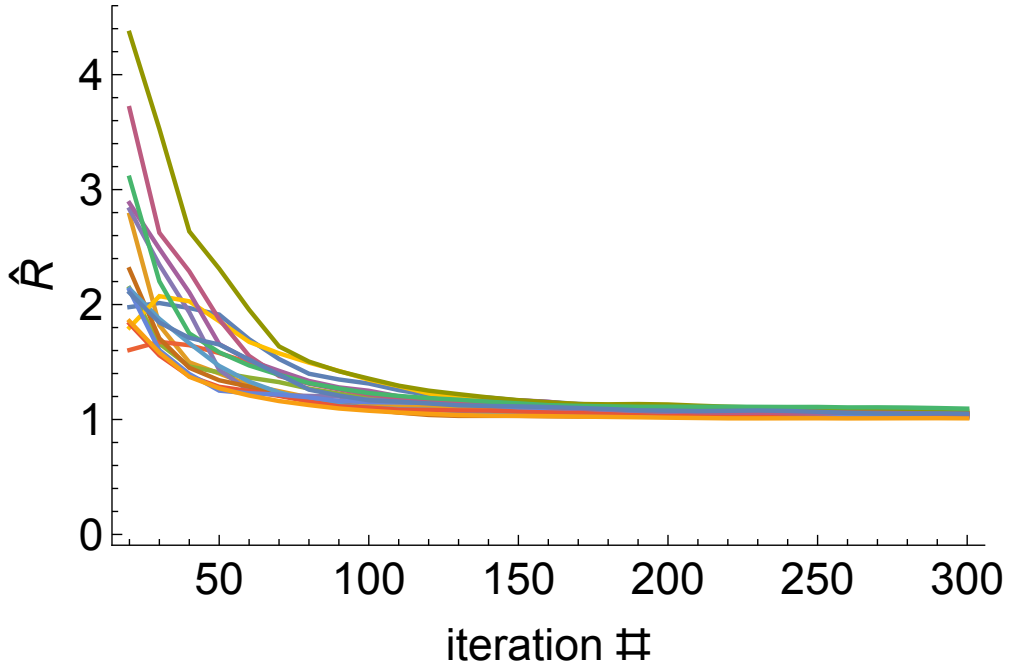

Figure 15.6: The values of $\hat{R}$ for μ_t across 16 different replicates, each with 8 chains running in parallel.

These estimates will be somewhat noisy, but I obtained an equivalent sample size of 11 for the Gibbs and 7 for RWM (Figure 15.9).

Problem 15.1.13. What do the above results tell you about the relative efficiency of each of the three samplers?

The Gibbs is more efficient than RWM. Although both are quite inefficient compared to the independent sampler; each with an effective sample size far less than 10% actual sample size.

Problem 15.1.14. Code up a Hamiltonian Monte Carlo sampler for this problem. (Alternatively, use the functions provided in the R file "HMC_scripts.R" adapted from [1]). Use a standard deviation of the momentum proposal distribution (normal) of 0.18, along with a step size $\epsilon = 0.18$ and $L = 10$ individual steps per iteration to simulate 100 samples from the posterior. How does the estimate of the mean compare with that from the Independent, Random Walk Metropolis and Gibbs samplers?

The performance of HMC is comparable to that of the independent sampler (Figure 15.9), and hence considerably more efficient at estimating the posterior mean.

Problem 15.1.15. What is the acceptance rate for HMC? How does this compare with RWM?

Based on 1,000 samples I obtain an acceptance rate of above 99% for HMC.

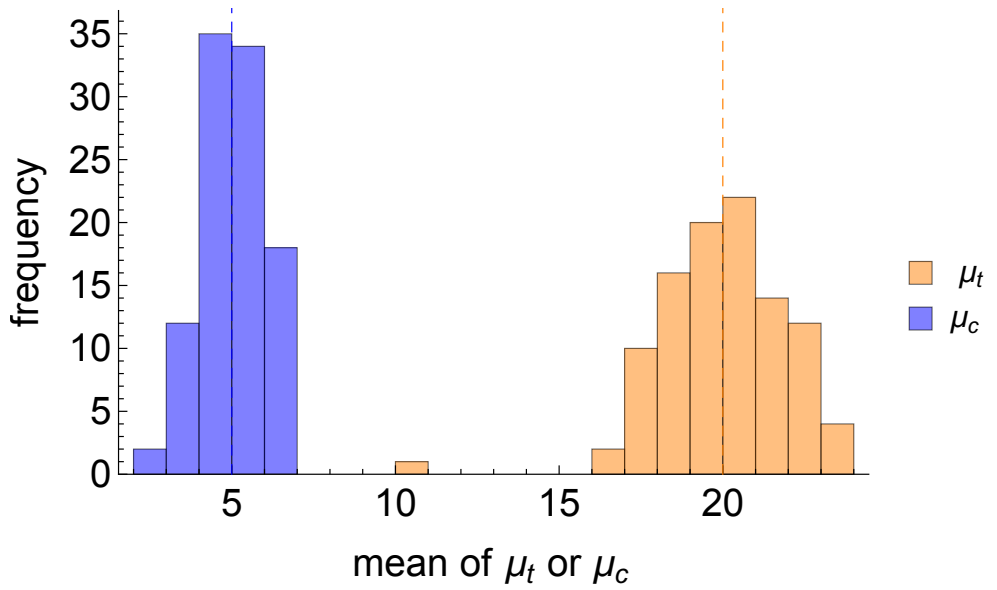

Figure 15.7: The sampling distribution for 100 samples from the posterior using a Gibbs sampler.

Problem 15.1.16. Gibbs sampling has an acceptance rate of 100%. How can HMC be more efficient than Gibbs given that its acceptance rate is less than 100%?

The Gibbs sampler moves in steps that are either vertical or horizontal (Figure 15.2). This is an inefficient way to explore the posterior which has a diagonal orientation. HMC tends to move in the diagonal with an acceptance rate comparable to that of Gibbs. By taking account of the posterior geometry it is much more efficient at exploring the typical set.

Problem 15.1.17. You receive new data that results in a change in the posterior to:

$$\begin{pmatrix}\mu_t\\ \mu_c\end{pmatrix} \sim \mathcal{N}\left[\begin{pmatrix}20\\ 5\end{pmatrix}, \quad \begin{pmatrix}2 & 0.99\\ 0.99 & 0.5\end{pmatrix}\right]$$

Using your Random Walk Metropolis sampler calculate $\hat{R}$ for 8 chains; each generating 100 samples for each.

With the higher correlation between the parameters the rate of convergence is slower. Intuitively this is because the model is poorly identified; it is impossible to disentangle one parameter's effect from another's. This causes problems with statistical inference, and hence with the sampling. This is an example of Gelman's "Folk Theorem" which states that any problem with MCMC is generally a problem with the underlying model.

In this case we can still get convergence although it occurs at a much slower rate than before (Figure 15.10), meaning that a sample size of about 500 is required.

Problem 15.1.18. Estimate the value of $\hat{R}$ for HMC on the posterior from the new data, for a sample size of 100. How does it compare to Random Walk Metropolis?

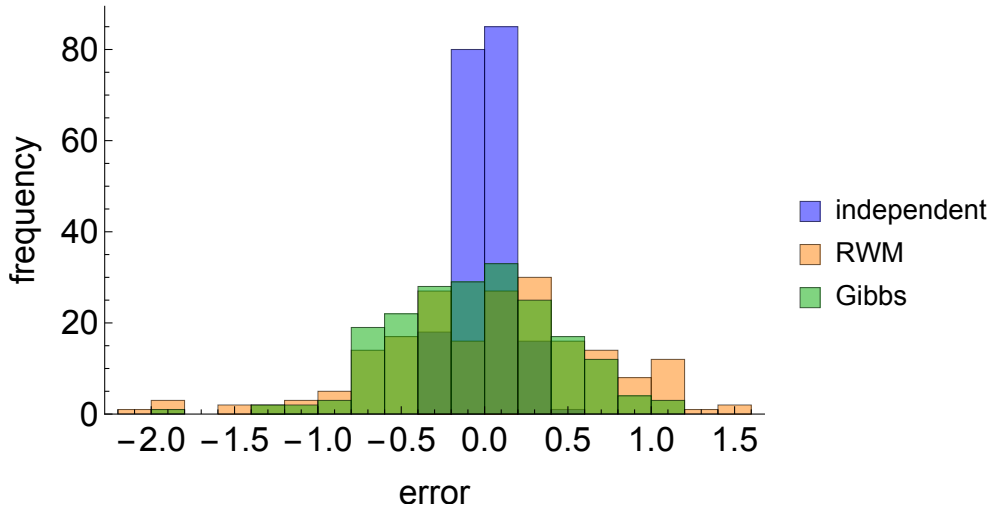

Figure 15.8: The sampling distribution in estimating the posterior mean of μ_t for three different sampling algorithms.

The rate of convergence will be significantly faster for HMC, and so we expect a value of $\hat{R}$ that is less than that for Random Walk Metropolis.

15.2 HMC and U-Turns

The code in `HMC_UTurn.R` uses simulates Hamiltonian dynamics for a single particle on the distribution described in the previous question:

$$\begin{pmatrix}\mu_t \\ \mu_c\end{pmatrix} \sim \mathcal{N}\left[\begin{pmatrix}20 \\ 5\end{pmatrix}, \quad \begin{pmatrix}2 & 0.8 \\ 0.8 & 0.5\end{pmatrix}\right]$$

In this question we will see how the efficiency of HMC depends on choice of the number of intermediate steps. In particular we investigate the propensity of a particle undergoing Newtonian dynamics to perform U-Turns.

Problem 15.2.1. Simulate a single particle starting at $(20, 5)$ for $L = 10$ steps with the following parameters $\epsilon = 0.18$ (step size), $\sigma = 0.18$ (momentum proposal distribution width). Plot the path in parameter space.

The particle seems to move without turning round (Figure 15.11).

Problem 15.2.2. Now try $L = 20, 50, 100$ steps, again plotting the results what do you notice about the paths?

As the number of steps taken increases the particle becomes more predisposed to U-turns (Figure 15.11).

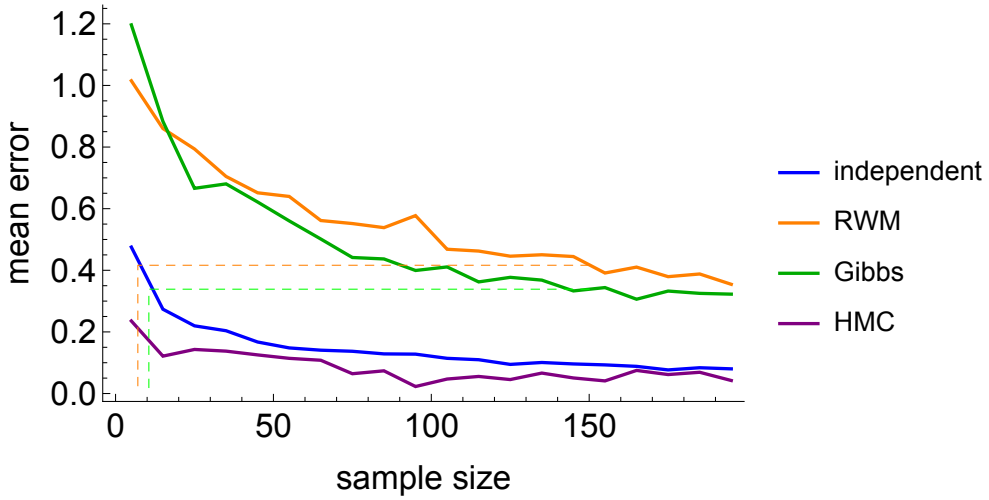

Figure 15.9: The mean error in estimating the posterior mean of μ_t for four different sampling algorithms. The dotted lines indicate the effective sample sizes for an actual sample size of 100 for the Gibbs and Random Walk Metropolis algorithms.

Problem 15.2.3. Simulate 100 iterations of the particle starting at $(20, 5)$, with each particle running for $L = 100$ steps. Examine the motion of the particle in one of the parameter dimensions, and hence determine an optimal number of steps for this distribution.

This can be done with the following R code,

```
nReplicates <- 100
nStep <- 100
mAll <- matrix(ncol=nReplicates, nrow=nStep)
for(i in 1:nReplicates){
  lTest <- HMC_keep(c(20, 5), U, grad_U, 0.18, nStep, 0.18)
  lTemp <- lTest$pos[, 1]
  aLen <- length(lTemp)
  mAll[, i] <- lTemp[1:(aLen - 1)]
}

library(reshape2)
mAll <- melt(mAll)
library(ggplot2)
ggplot(mAll, aes(x=Var1, colour=as.factor(Var2), y=value)) + geom_path() +
  theme(legend.position="none") +
  ylab('mu_t') + xlab('number of steps')
```

which should produce a plot qualitatively similar to Figure 15.12. From this we can see that we explore the most distance in this parameter dimension for about $L = 13 - 14$. Any longer and the particle turns round on its self.

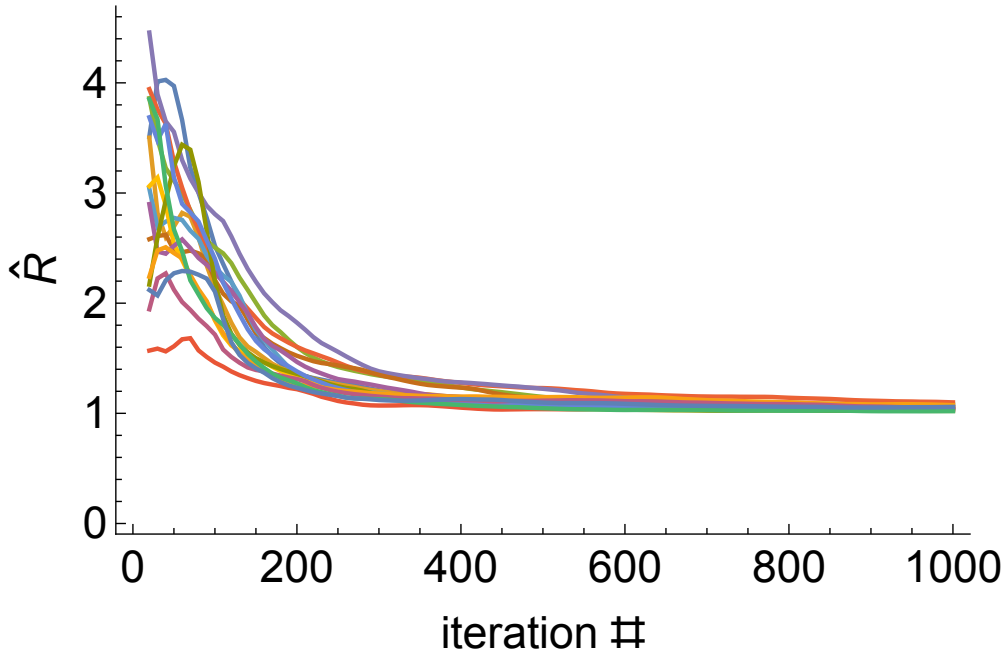

Figure 15.10: The values of $\hat{R}$ for 16 different replicates, each with 8 chains running in parallel for the case where $\rho = 0.99$.

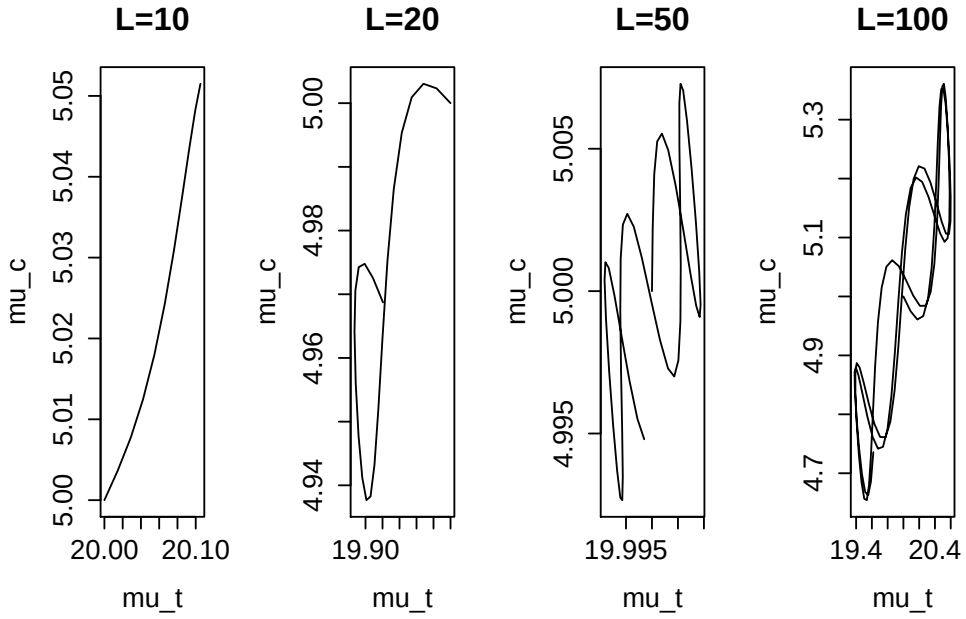

Figure 15.11: The path of a particle in parameter space for differing number of steps, L.

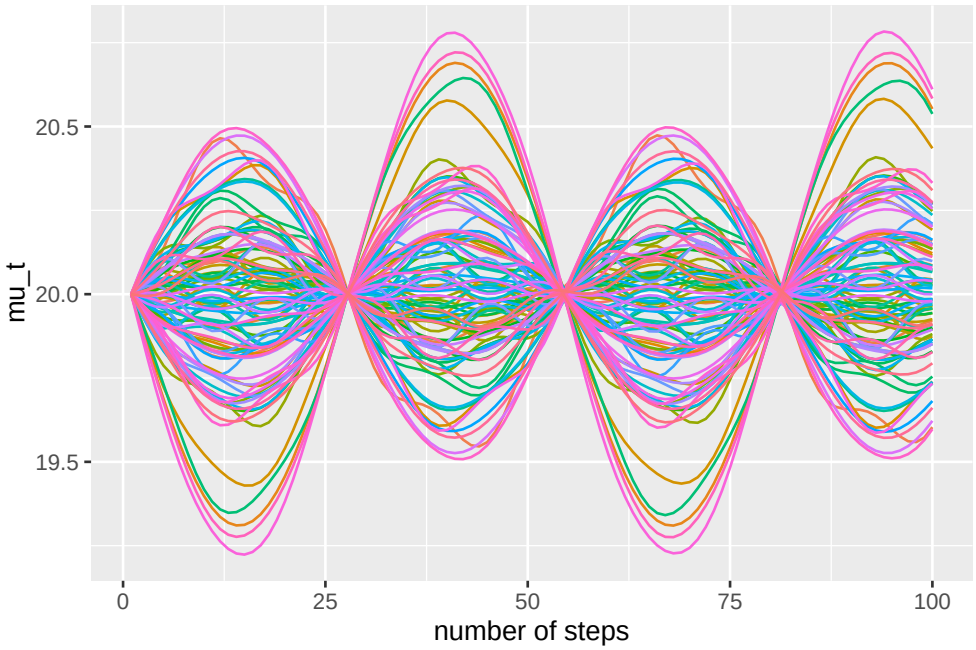

Figure 15.12: The path of 100 particle replicates over in the μ_t dimension.

Bibliography

[1] Radford M Neal et al. Mcmc using hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo*, 2:113–162, 2011.