Problem 1

(a)

First, I derived the posterior distribution $\pi(\beta_0 \,|\, \boldsymbol{Y}, \boldsymbol{X})$. To do this, I used the fact that the posterior is equal to the product of the likelihood and the prior distributions (up to a constant, which we can disregard). For numerical stability, I worked in log-scale, so that the log-posterior is proportional to the sum of the log-likelihood and the log-prior. Once these are correctly defined, the Metropolis-Hastings algorithm is as follows:

```
chain[1] = initial state
for i in 2:length(chain)
    p = draw from proposal distribution
    prob = exp(posterior(p) - posterior(chain[i]))
    if a draw from standard uniform < prob
        chain[i+1] = p                              (Metropolis update)
    else
        chain[i+1] = chain[i]
```

For my algorithm I used a symmetric random walk proposal, and settled on a variance of 1. This proposal ended up having an acceptance rate of about 35%, which is pretty good. I simulated a chain with 100,000 observations.[1] To obtain an idea of a starting value to use, I plotted the log-likelihood function. Since we are only dealing with a univariate problem, this is an easy way to get a rough idea of the maximum likelihood estimator of the data. Based on that (see R code for plot), it looks like 7.5 is a pretty good initial guess. I also tried starting values of 5 and 10, just in case the posterior distribution has some weird bimodality issues.
Here are the density functions that needed to be defined (keeping in mind that we are told $\beta_0 = 5$, $\lambda = 0.4$, $\sigma = 1$ for all i):

$$\log f(\boldsymbol{Y} \,|\, \beta_1, \boldsymbol{X}) \propto 0.4\beta_1 \sum_{i=1}^{n} x_i - 0.4 \sum_{i=1}^{n} y_i + \sum_{i=1}^{n} \log \operatorname{erfc}\left(\frac{5.4 + \beta_1 x_i - y_i}{\sqrt{2}}\right) \qquad \text{(log-likelihood)}$$

$$\log g(\beta_1) \propto -\frac{\beta_1^2}{200} \qquad \text{(log-prior)}$$

The log-posterior function is simply the sum of these two. See the corresponding R code for the exact implementation of my algorithm.

(b)

Results summarized in Table 1.

Table 1: Results of M-H runs, for each starting value.

Starting value	$\widehat{\mathrm{E}[\beta_1]}$	MCMCse
5	7.3398	0.002194
7.5	7.3418	0.002117
10	7.3409	0.001961

[1] I did perform a run with 1,000,000 observations, which improved the MCMC standard errors from 0.002 to 0.00066. However, the estimates and credible intervals were nearly identical, and plots showed that the estimates appeared to converge pretty sufficiently by 100,000 samples, so I elected to omit this from my final code in the interest of run time.

Problem 1 continued on next page...

(c)

Credible intervals given in Table 2.

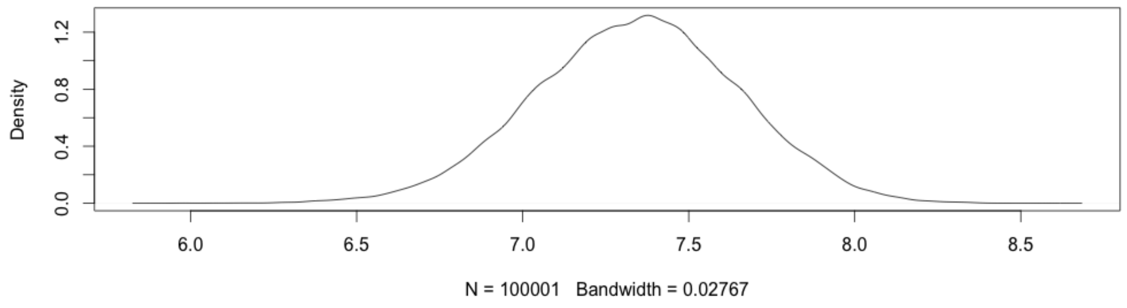
Table 2: Credible intervals for $\widehat{\mathrm{E}[\beta_1]}$, for each starting value.

Starting value	Credible interval
5	$(6.7166, 7.9270)$
7.5	$(6.7277, 7.9277)$
10	$(6.7166, 7.9362)$

(d)

The smoothed density plots of the three realizations are very similar, so I have only included the one corresponding with an initial value of 7.5. See Figure 1.

Figure 1: Smoothed plot of the estimated posterior pdf of β_1.



(e)

All three realizations have very similar estimates and credible intervals, which is a good sign that the posterior distribution is unimodal. The MCMC standard errors are all fairly small as well. Figure 2 shows that the estimates for each realization appear to be converging and "settling down." Also, the autocorrelations fall off nicely, as seen in Figure 3. This indicates that there is not too much dependence in the Markov chain, and thus we are seeing enough independent samples to trust our results. This is further supported by the effective sample size, which was calculated to be about 20,000.

Figure 2: Estimates as a function of sample size. Black line is starting value of 5, blue is 7.5, red is 10.

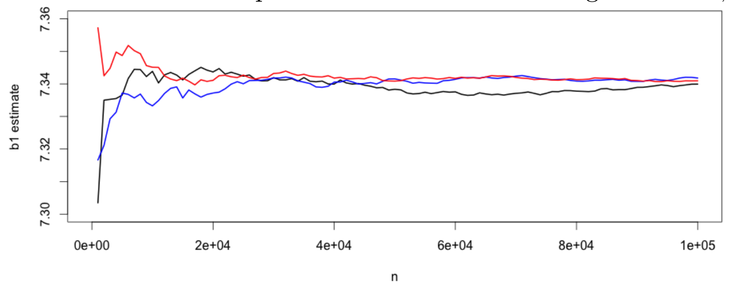
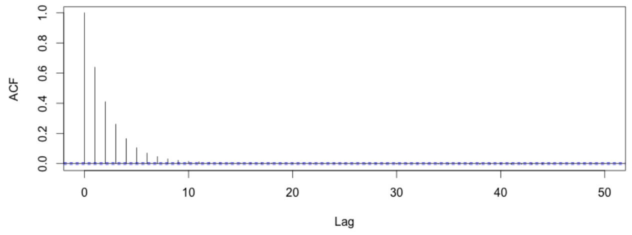


Figure 3: Autocorrelation plot for the realization with starting value 7.5. The others were more or less identical and therefore were omitted.



Problem 2

(a)

At first I used a Metropolis algorithm with a simple random walk proposal. However, this proved to be pretty inefficient, with about 500,000 iterations needed just to get an effective sample size of about 5,000 for each parameter. Authors Chib and Greenberg (1994) recommend an alternative: use a fixed multivariate normal proposal centered at the mode of the target (in this case, the MLE) and with the inverse of the Hessian evaluated at the mode as the covariance matrix. This is what I ended up using, with significantly improved results. Using the Newton-Raphson method, I found the MLE to be about $(2.35, 3.46, 0.803)$. The Hessian is available in the code but is omitted here.

```
chain[,1] = initial states
for i in 2:length(chain)
    for j in 1:3
        p = draw from proposal distribution of jth parameter
        let ‘‘posterior’’ be the full conditional posterior of jth parameter
        prob = exp(posterior(p, cur val of all other params) - posterior(cur val of all other params))
        if a draw from standard uniform < prob
            next value of jth parameter = p
        else
            next value of jth parameter = current value of jth parameter
```

By simple algebra, the full conditional density functions are as follows:

$$\pi(\beta_0 \,|\, \beta_1, \lambda, \boldsymbol{Y}, \boldsymbol{X}) \propto n\lambda\beta_0 + \sum_{i=1}^{n} \log \operatorname{erfc}\left(\frac{\beta_0 + \beta_1 x_i + \lambda - y_i}{\sqrt{2}}\right) - \frac{\beta_0^2}{200}$$

$$\pi(\beta_1 \,|\, \beta_0, \lambda, \boldsymbol{Y}, \boldsymbol{X}) \propto n\lambda\beta_1 \sum_{i=1}^{n} x_i + \sum_{i=1}^{n} \log \operatorname{erfc}\left(\frac{\beta_0 + \beta_1 x_i + \lambda - y_i}{\sqrt{2}}\right) - \frac{\beta_1^2}{200}$$

$$\pi(\lambda \,|\, \beta_0, \beta_1, \boldsymbol{Y}, \boldsymbol{X}) \propto (n - 0.99)\log\lambda + n\lambda\beta_0 + \lambda\beta_1 \sum_{i=1}^{n} x_i + \frac{1}{2}n\lambda^2 - n\lambda \sum_{i=1}^{n} y_i + \sum_{i=1}^{n} \log \operatorname{erfc}\left(\frac{\beta_0 + \beta_1 x_i + \lambda - y_i}{\sqrt{2}}\right) - \frac{\lambda}{100}$$

(b)

See Table 3.

(c)

I found the correlation between the simulated posterior distributions for β_0 and for β_1 to be -0.701.

Problem 2 continued on next page. . .

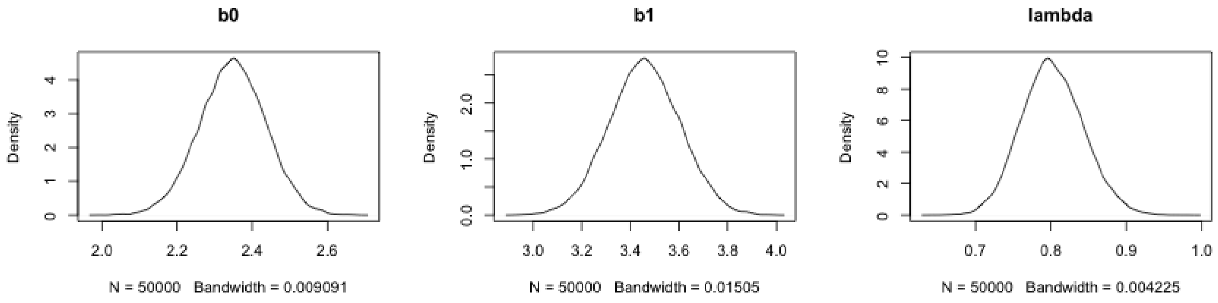
Table 3: Results for Problem 2.

Parameter	Posterior Mean	MCMC standard error	95% credible interval
β_0	2.3473	0.0012	$(2.1728, 2.5173)$
β_1	3.4595	0.0017	$(3.1734, 3.7482)$
λ	0.8021	0.0003	$(0.7225, 0.8852)$

(d)

See Figure 4 below.

Figure 4: Smoothed approximate density plots for marginal distributions



(e)

Figure 5 shows that the autocorrelation falls off nicely. The effective sample sizes are 5400 for β_0, 6800 for β_1, and 16,800 for λ, pretty good considering I only ran 50,000 iterations. This is a good sign for the accuracy of the estimates. Furthermore, a plot (Figure 6) of the data along with the line $y = \beta_0 + \beta_1 x$ indicates that the data probably could have been generated using these parameters. Finally, because I have little confidence in myself, I ran a small simulation study in which I generated data using $\beta_0 = 4$, $\beta_1 = 1.5$, and $\lambda = 0.7$, to see if my algorithm would give me those values back. We see from Table 4 that the algorithm performed nicely, which leads me to believe the results for the given problem are pretty accurate as well.

Table 4: Results from simulation study. True values were $\beta_0 = 4$, $\beta_1 = 1.5$, $\lambda = 0.7$.

Parameter	Posterior Mean	MCMC standard error	95% credible interval
β_0	4.0729	0.0013	$(3.8875, 4.2577)$
β_1	1.6241	0.0019	$(1.3164, 1.9393)$
λ	0.7398	0.0003	$(0.6684, 0.8127)$

Problem 3

(a)

I am assuming here that the priors and likelihood functions are all the same as they were for problem 2. See Table 5 for results.

Problem 3 [(a)] continued on next page...

Figure 5: Autocorrelation plots for problem 2. β_0, β_1, and λ respectively.

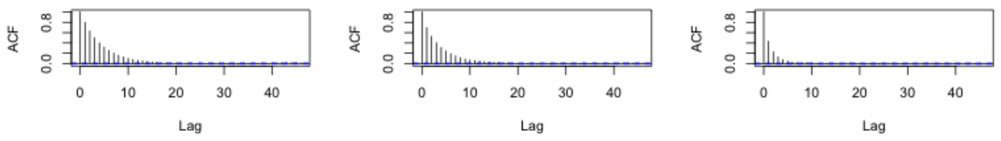


Figure 6: Scatterplot of data with fitted line using β_0, β_1.

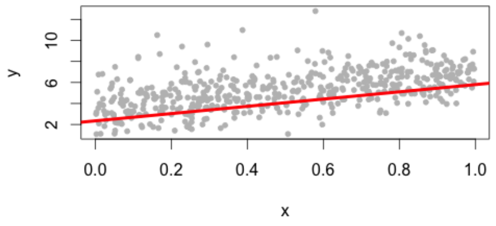


Table 5: Results for problem 3

Parameter	Posterior Mean	MCMC standard error	95% credible interval
β_0	0.1513	0.0021	$(-0.1712, 0.4614)$
β_1	2.4795	0.0033	$(1.9367, 3.005)$
λ	0.1612	0.00013	$(0.1499, 0.1718)$

(b)

See Figure 7 below.

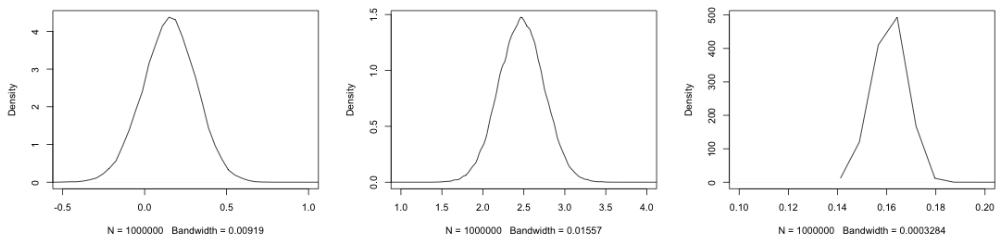


Figure 7: Smoothed approximate density plots for marginal distributions

(c)

Looking at a scatterplot of the data, it appears as though we might have a bimodal distribution. There are two pretty distinct parallel clusters of points, which suggests that the data might be taken from two densities, perhaps having the same β_1 with different β_0 values. Theoretically, a Metropolis algorithm with a symmetric random walk proposal density should be able to detect two peaks in the β_0 distribution, or whichever distributions are bimodal. However, despite increasing the variance of the proposal distribution and moving the initial point around, I was unable to detect more than one peak for any parameter, even when running 1,000,000 iterations to get an acceptable ESS. I am unsure whether my algorithm is poorly constructed or if my hypothesis about the bimodality is just wrong.