Homework 2

Stat 597a: Spatial Models

Claire Kelling

Due September 28, 2017

Problem 1:

First, read over this section. There is one key change we will make compared to Wikle's analysis, which is that I have first transformed the coordinates from longitude and latitude to UTM coordinates. This is because the geoRglm package only uses Euclidean distances. Using it with longitude and latitude will not give accurate distances between the points. Another way we could do it, if we were coding this from scratch, would be to keep longitude and latitude, but work with great circle distances when calculating the distance matrix.

Given this new coordinate system, the prior for the range parameter ϕ that we'll use is a discrete uniform distribution from 500 to 300,000, in increments of 500. The reasons for the discrete prior are again due to the constraints of the geoRglm package.

With this change, write down the three layers of the hierarchical model as defined on slide 8 of Lecture 8. That is, what distributions make up the data model, process model, and prior model?

From the lecture, we know there are going to be 3 components to our model: the data model, the process model, and the parameter model. That is, $f(\eta, \theta|\mathbf{Y}) \propto f(\mathbf{Y}, \eta, \theta) \times f(\eta|\theta) \times \pi(\theta)$.

According to the textbook, Hierarchical Model with Spatial Data, we will model this data according to the genearlized linear spatial modeling framework.

So, the data model is as follows:

$$f(\mathbf{Y}|\lambda(s_i), \phi) \sim Poisson(\lambda(s_i))$$

where we employ a Gaussian spatial process model to describe the spatial variation in $\lambda(s_i)$ and use the canonical log-link function. So, $log(\lambda(s_i)) = \beta + \eta(s_i)$ where $\eta(s_i)$ is a Guassian process with mean 0, variance σ_n^2 and correlation function $r(s_i, s_j; \phi)$. Specifically, $r_\eta(s_i, s_j; \phi) = exp(-||s_i - s_j||/\phi)$. We were told in the problem statement that ϕ followes a discrete uniform distribution from 500 to 300,000, in increments of 500. Therefore, the process model is as follows:

$$log(\lambda(s_i)) = \beta + \eta(s_i) \text{ where}$$

$$f(\eta(s_i)|\phi) \sim \text{GP}(mean = 0, \text{variance} = \sigma_n^2, \text{correlation function} = r_\eta(s_i, s_j; \phi) = exp(-||s_i - s_j||/\phi)$$

and the parameter model is as follows:

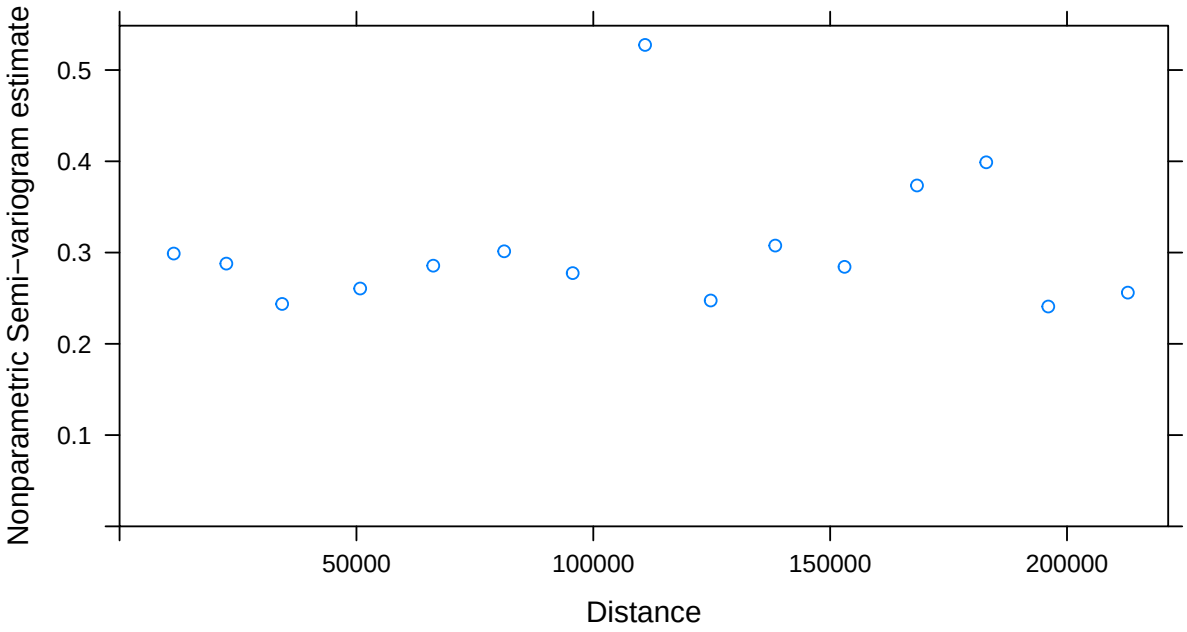
$$\pi(\phi) \sim Uniform(500, 300{,}000) \text{ with breaks as described above.}$$

Problem 2

Transform the original observations using $Z_i = log(Y_i)$ and use classical geostatistical techniques to get preliminary estimates of σ^2 and ϕ by treating the Z_i as normally distributed given η (i.e. fit a model with a nugget, and extract just the estimates for σ^2 and ϕ).

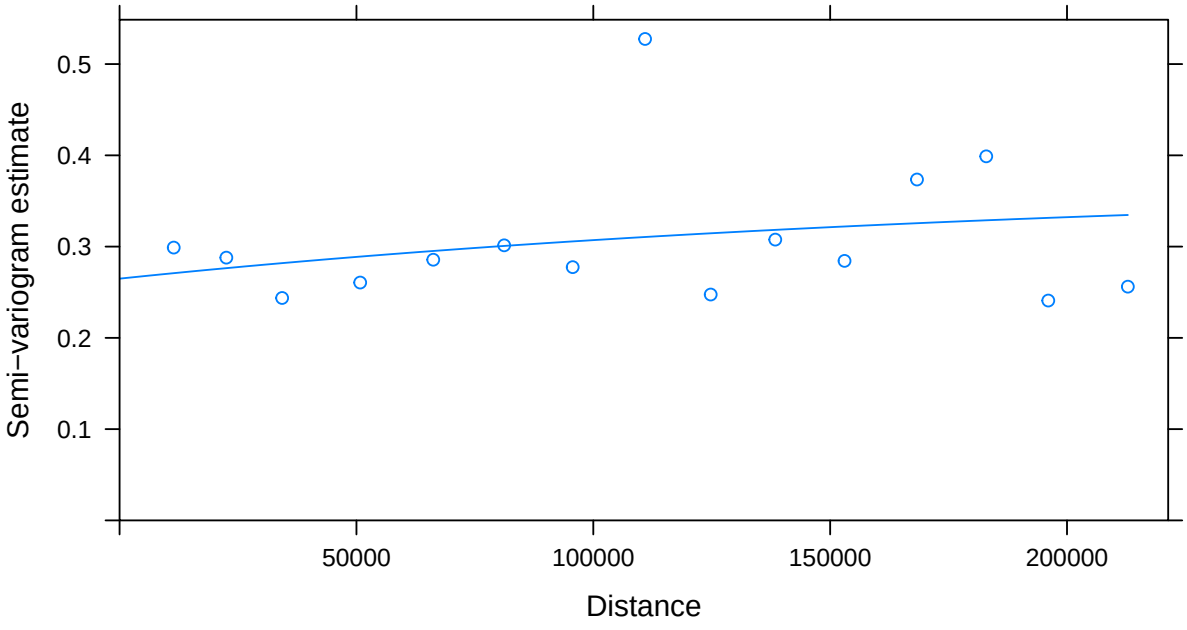
```
##
## Call:
## lm(formula = z ~ 1, data = dove)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.2009 -0.3038  0.1017  0.3767  1.1072
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.29951    0.08421   39.18   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5773 on 46 degrees of freedom
```

```
##    np      dist     gamma dir.hor dir.ver   id
## 1   1  11430.00 0.2989113       0       0 var1
## 2   4  22535.47 0.2879151       0       0 var1
## 3   8  34315.57 0.2437084       0       0 var1
## 4  29  50790.42 0.2606114       0       0 var1
## 5  34  66219.10 0.2856123       0       0 var1
## 6  41  81185.97 0.3013776       0       0 var1
## 7  38  95650.94 0.2774728       0       0 var1
## 8  41 110932.50 0.5275067       0       0 var1
## 9  40 124783.97 0.2474626       0       0 var1
## 10 50 138418.78 0.3076691       0       0 var1
## 11 43 153035.99 0.2843422       0       0 var1
## 12 58 168332.32 0.3735157       0       0 var1
## 13 45 182952.55 0.3989139       0       0 var1
## 14 50 196046.65 0.2408701       0       0 var1
## 15 61 212845.51 0.2560925       0       0 var1
```




```
## [1] 47
```

```
##   model     psill    range
## 1   Nug 0.2648920      0.0
## 2   Exp 0.1044543 193204.2
```



Problem 3

Use model.glm.control, prior.glm.control, mcmc.control, and pois.krige.bayes to run an initial MCMC chain, fixing ϕ at your estimate from (1). The goal here is to experiment with changing S.scale to achieve an acceptance rate of about 60% for the process samples, which is optimal for the algorithm pois.krige.bayes is using to sample the vector of process values.

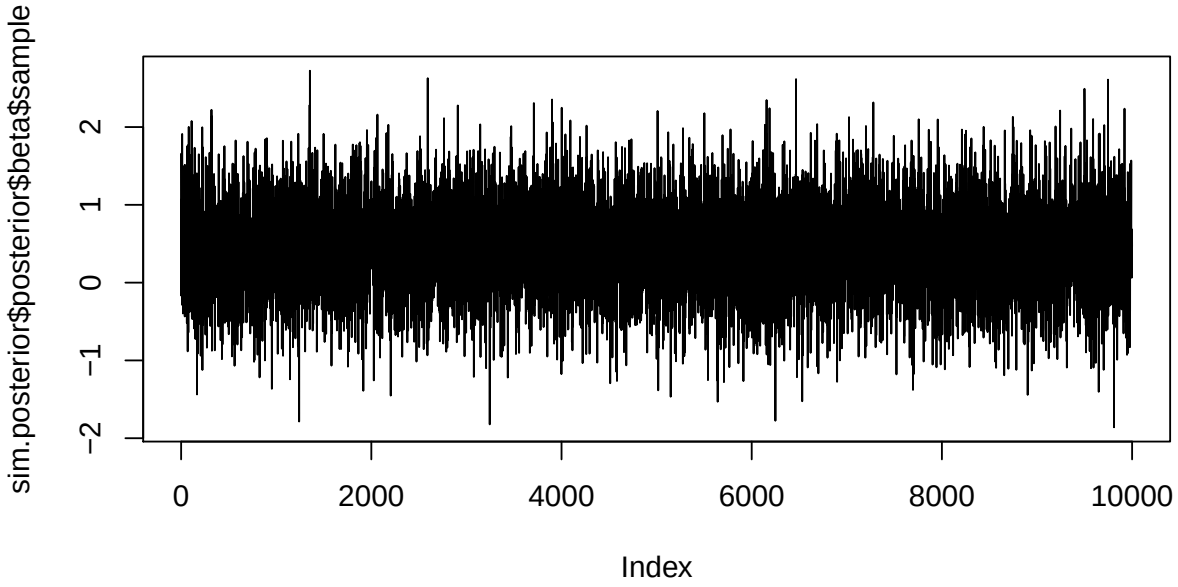
The choice of niter = 100000 and thin = 10 is just to keep things manageable at this stage. We will eventually run a much longer chain. I suggest you always keep burn.in = 0 and then discard the initial samples yourself after seeing the results.

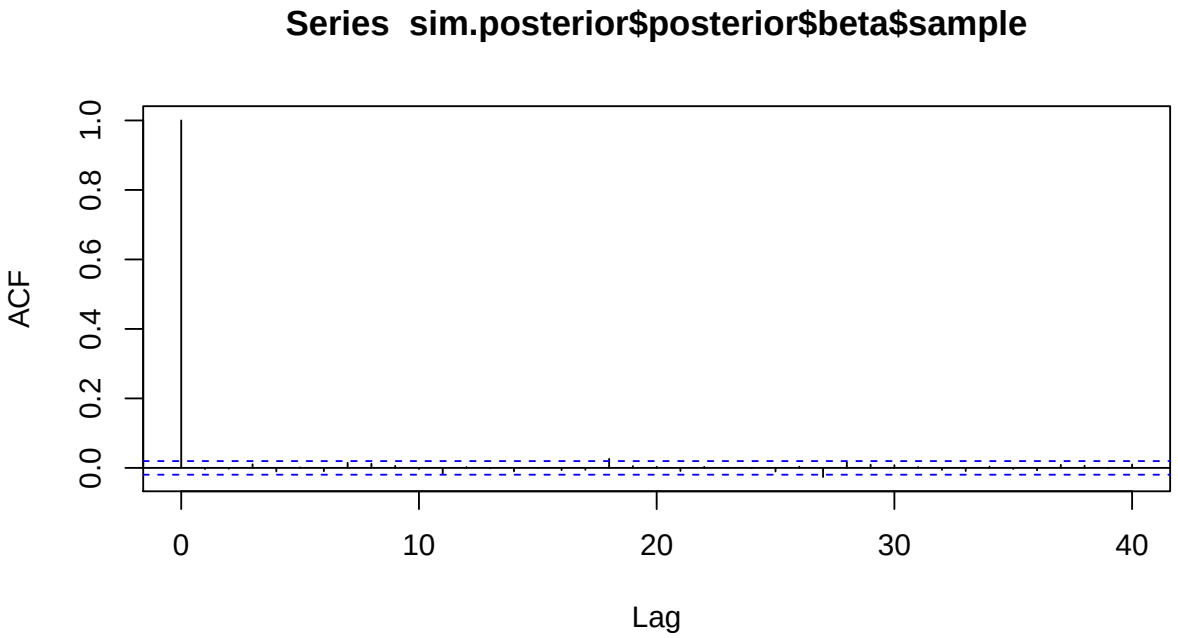
```
## pois.krige.bayes: model with mean being constant
## iter. numb. 1000 ; Acc.-rate = 0.00
## iter. numb. 2000 ; Acc.-rate = 0.00
## iter. numb. 3000 ; Acc.-rate = 0.00
## iter. numb. 4000 ; Acc.-rate = 0.00
## iter. numb. 5000 ; Acc.-rate = 0.00
## iter. numb. 6000 ; Acc.-rate = 0.00
## iter. numb. 7000 ; Acc.-rate = 0.00
## iter. numb. 8000 ; Acc.-rate = 0.00
## iter. numb. 9000 ; Acc.-rate = 0.00
## iter. numb. 10000 ; Acc.-rate = 0.00
## iter. numb. 11000 ; Acc.-rate = 0.00
## iter. numb. 12000 ; Acc.-rate = 0.00
## iter. numb. 13000 ; Acc.-rate = 0.00
## iter. numb. 14000 ; Acc.-rate = 0.00
## iter. numb. 15000 ; Acc.-rate = 0.00
## iter. numb. 16000 ; Acc.-rate = 0.00
## iter. numb. 17000 ; Acc.-rate = 0.00
## iter. numb. 18000 ; Acc.-rate = 0.00
## iter. numb. 19000 ; Acc.-rate = 0.00
## iter. numb. 20000 ; Acc.-rate = 0.00
## iter. numb. 21000 ; Acc.-rate = 0.00
## iter. numb. 22000 ; Acc.-rate = 0.00
## iter. numb. 23000 ; Acc.-rate = 0.00
## iter. numb. 24000 ; Acc.-rate = 0.00
## iter. numb. 25000 ; Acc.-rate = 0.00
## iter. numb. 26000 ; Acc.-rate = 0.00
## iter. numb. 27000 ; Acc.-rate = 0.00
## iter. numb. 28000 ; Acc.-rate = 0.00
## iter. numb. 29000 ; Acc.-rate = 0.00
## iter. numb. 30000 ; Acc.-rate = 0.00
## iter. numb. 31000 ; Acc.-rate = 0.00
## iter. numb. 32000 ; Acc.-rate = 0.00
## iter. numb. 33000 ; Acc.-rate = 0.00
## iter. numb. 34000 ; Acc.-rate = 0.00
## iter. numb. 35000 ; Acc.-rate = 0.00
## iter. numb. 36000 ; Acc.-rate = 0.00
## iter. numb. 37000 ; Acc.-rate = 0.00
## iter. numb. 38000 ; Acc.-rate = 0.00
## iter. numb. 39000 ; Acc.-rate = 0.00
## iter. numb. 40000 ; Acc.-rate = 0.00
## iter. numb. 41000 ; Acc.-rate = 0.00
## iter. numb. 42000 ; Acc.-rate = 0.00
## iter. numb. 43000 ; Acc.-rate = 0.00
```

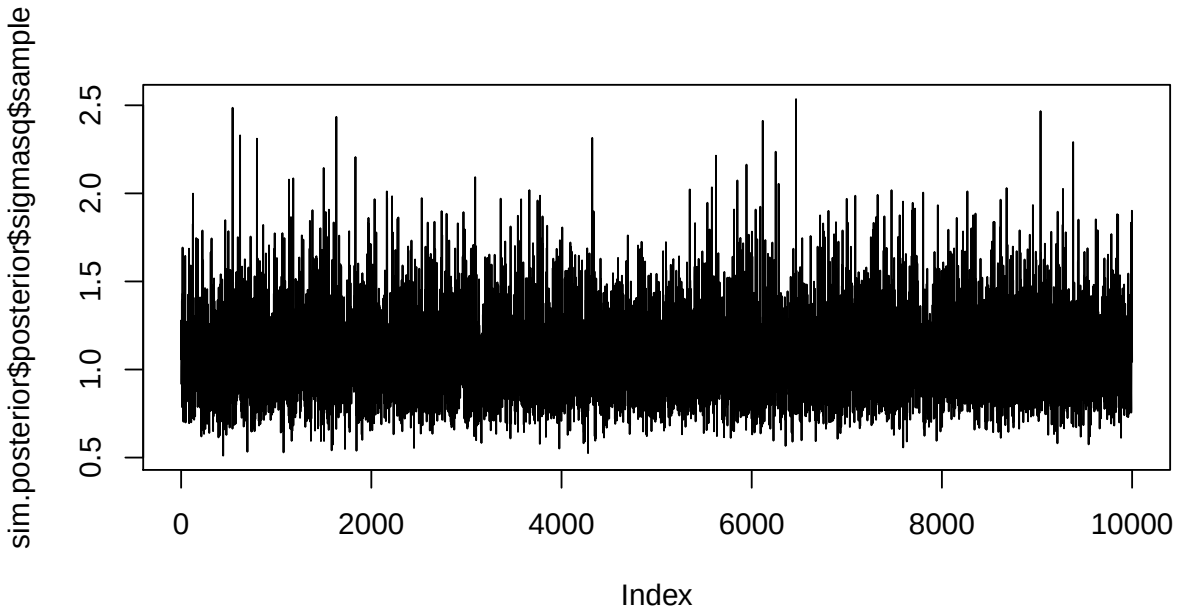
```
## iter. numb. 44000 ; Acc.-rate = 0.00
## iter. numb. 45000 ; Acc.-rate = 0.00
## iter. numb. 46000 ; Acc.-rate = 0.00
## iter. numb. 47000 ; Acc.-rate = 0.00
## iter. numb. 48000 ; Acc.-rate = 0.00
## iter. numb. 49000 ; Acc.-rate = 0.00
## iter. numb. 50000 ; Acc.-rate = 0.00
## iter. numb. 51000 ; Acc.-rate = 0.00
## iter. numb. 52000 ; Acc.-rate = 0.00
## iter. numb. 53000 ; Acc.-rate = 0.00
## iter. numb. 54000 ; Acc.-rate = 0.00
## iter. numb. 55000 ; Acc.-rate = 0.00
## iter. numb. 56000 ; Acc.-rate = 0.00
## iter. numb. 57000 ; Acc.-rate = 0.00
## iter. numb. 58000 ; Acc.-rate = 0.00
## iter. numb. 59000 ; Acc.-rate = 0.00
## iter. numb. 60000 ; Acc.-rate = 0.00
## iter. numb. 61000 ; Acc.-rate = 0.00
## iter. numb. 62000 ; Acc.-rate = 0.00
## iter. numb. 63000 ; Acc.-rate = 0.00
## iter. numb. 64000 ; Acc.-rate = 0.00
## iter. numb. 65000 ; Acc.-rate = 0.00
## iter. numb. 66000 ; Acc.-rate = 0.00
## iter. numb. 67000 ; Acc.-rate = 0.00
## iter. numb. 68000 ; Acc.-rate = 0.00
## iter. numb. 69000 ; Acc.-rate = 0.00
## iter. numb. 70000 ; Acc.-rate = 0.00
## iter. numb. 71000 ; Acc.-rate = 0.00
## iter. numb. 72000 ; Acc.-rate = 0.00
## iter. numb. 73000 ; Acc.-rate = 0.00
## iter. numb. 74000 ; Acc.-rate = 0.00
## iter. numb. 75000 ; Acc.-rate = 0.00
## iter. numb. 76000 ; Acc.-rate = 0.00
## iter. numb. 77000 ; Acc.-rate = 0.00
## iter. numb. 78000 ; Acc.-rate = 0.00
## iter. numb. 79000 ; Acc.-rate = 0.00
## iter. numb. 80000 ; Acc.-rate = 0.00
## iter. numb. 81000 ; Acc.-rate = 0.00
## iter. numb. 82000 ; Acc.-rate = 0.00
## iter. numb. 83000 ; Acc.-rate = 0.00
## iter. numb. 84000 ; Acc.-rate = 0.00
## iter. numb. 85000 ; Acc.-rate = 0.00
## iter. numb. 86000 ; Acc.-rate = 0.00
## iter. numb. 87000 ; Acc.-rate = 0.00
## iter. numb. 88000 ; Acc.-rate = 0.00
## iter. numb. 89000 ; Acc.-rate = 0.00
## iter. numb. 90000 ; Acc.-rate = 0.00
## iter. numb. 91000 ; Acc.-rate = 0.00
## iter. numb. 92000 ; Acc.-rate = 0.00
## iter. numb. 93000 ; Acc.-rate = 0.00
## iter. numb. 94000 ; Acc.-rate = 0.00
## iter. numb. 95000 ; Acc.-rate = 0.00
## iter. numb. 96000 ; Acc.-rate = 0.00
## iter. numb. 97000 ; Acc.-rate = 0.00
```

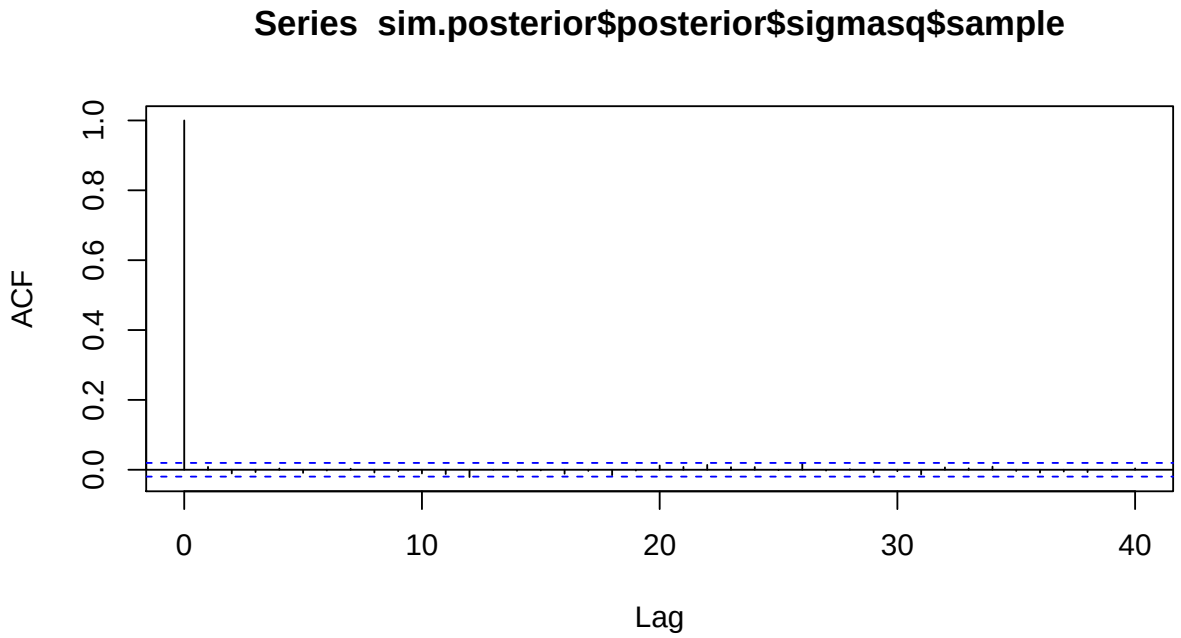
```
## iter. numb. 98000 ; Acc.-rate = 0.00
## iter. numb. 99000 ; Acc.-rate = 0.00
## iter. numb. 100000 ; Acc.-rate = 0.00
## MCMC performed: n.iter. =  1e+05 ; thinning =  10 ; burn.in =  0
## Only Bayesian estimation of model parameters
```

```
## iter.numb  Acc.rate
##     1e+05     0e+00
```




Series sim.posterior$posterior$beta$sample
ACF
0.0
0.2
0.4
0.6
0.8
1.0
0
10
20
30
40
Lag


sim.posterior$posterior$sigmasq$sample
0.5
1.0
1.5
2.0
2.5
0
2000
4000
6000
8000
10000
Index



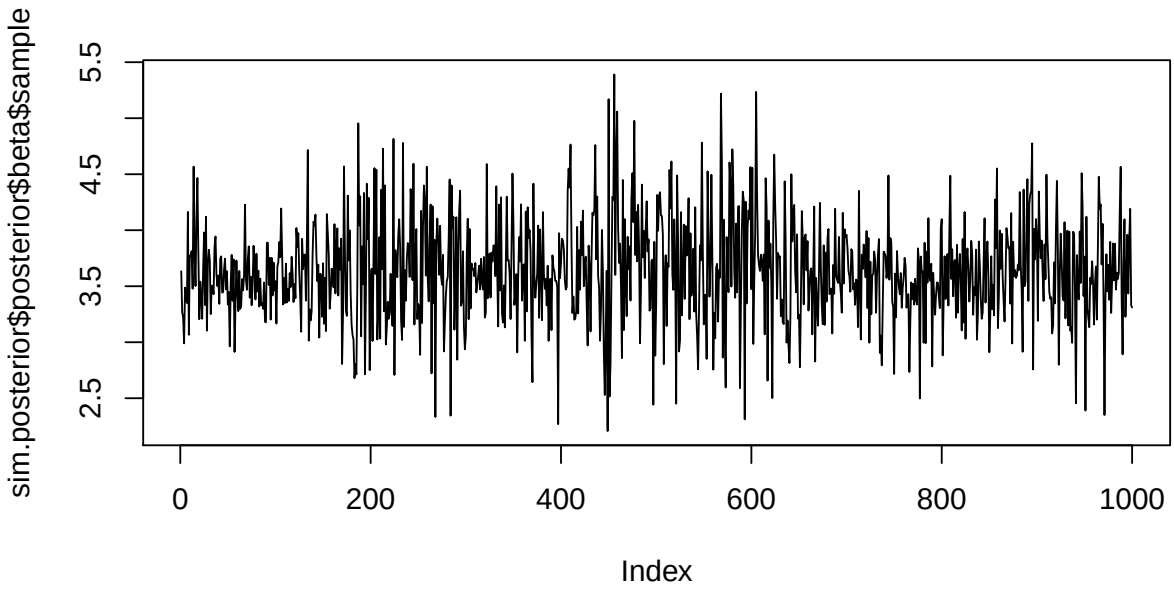
Problem 4

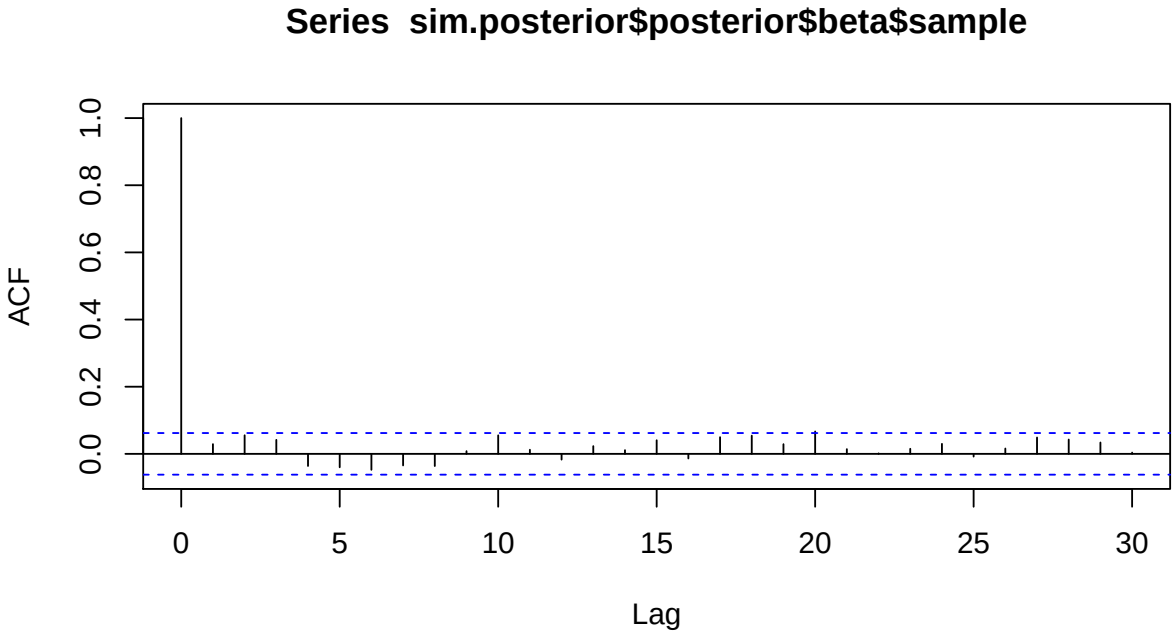
Now duplicate and modify the code from (3) to include sampling phi. Experiment with changing S.scale and phi.scale to get acceptance rates of about 60% and 25%, respectively. (Note: I found the acceptance rates fluctuated over the course of my chain. Just aim to get in the right ballpark.) I suggest you also take thin = 100 and n.iter = 100000 here. Make trace plots and ACF plots of the parameters $\sigma^2, \phi,$ and β. You will likely see a LOT of autocorrelation.

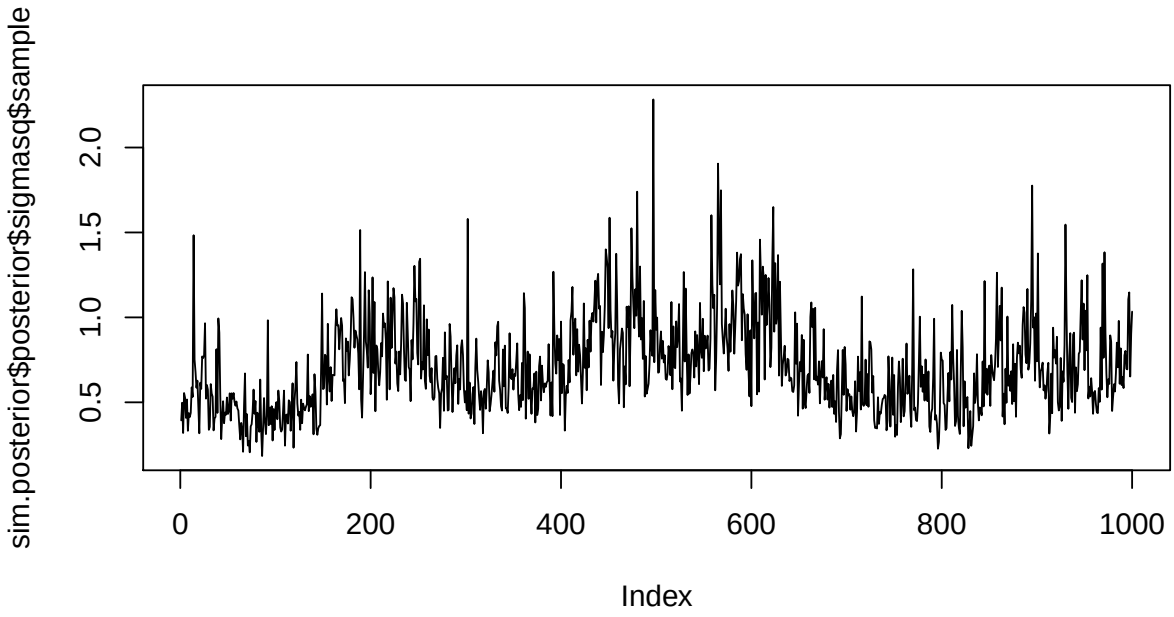
```
## pois.krige.bayes: model with mean being constant
## iter. numb. 1000 ; Acc.-rate = 0.41 ; Acc-rate-phi = 0.91
## iter. numb. 2000 ; Acc.-rate = 0.35 ; Acc-rate-phi = 0.90
## iter. numb. 3000 ; Acc.-rate = 0.38 ; Acc-rate-phi = 0.92
## iter. numb. 4000 ; Acc.-rate = 0.45 ; Acc-rate-phi = 0.89
## iter. numb. 5000 ; Acc.-rate = 0.47 ; Acc-rate-phi = 0.87
## iter. numb. 6000 ; Acc.-rate = 0.50 ; Acc-rate-phi = 0.86
## iter. numb. 7000 ; Acc.-rate = 0.58 ; Acc-rate-phi = 0.82
## iter. numb. 8000 ; Acc.-rate = 0.56 ; Acc-rate-phi = 0.84
## iter. numb. 9000 ; Acc.-rate = 0.61 ; Acc-rate-phi = 0.81
## iter. numb. 10000 ; Acc.-rate = 0.54 ; Acc-rate-phi = 0.88
## iter. numb. 11000 ; Acc.-rate = 0.54 ; Acc-rate-phi = 0.84
## iter. numb. 12000 ; Acc.-rate = 0.51 ; Acc-rate-phi = 0.87
## iter. numb. 13000 ; Acc.-rate = 0.48 ; Acc-rate-phi = 0.86
## iter. numb. 14000 ; Acc.-rate = 0.44 ; Acc-rate-phi = 0.91
## iter. numb. 15000 ; Acc.-rate = 0.36 ; Acc-rate-phi = 0.92
## iter. numb. 16000 ; Acc.-rate = 0.29 ; Acc-rate-phi = 0.95
## iter. numb. 17000 ; Acc.-rate = 0.28 ; Acc-rate-phi = 0.93
## iter. numb. 18000 ; Acc.-rate = 0.27 ; Acc-rate-phi = 0.95
## iter. numb. 19000 ; Acc.-rate = 0.24 ; Acc-rate-phi = 0.95
## iter. numb. 20000 ; Acc.-rate = 0.26 ; Acc-rate-phi = 0.95
```

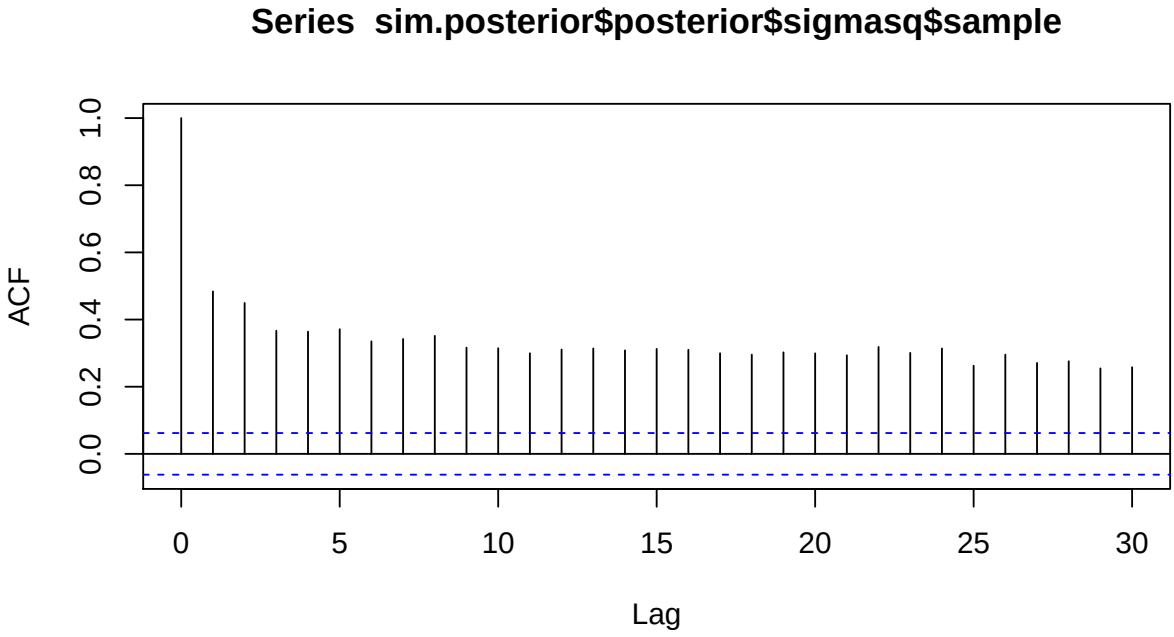
```
## iter. numb. 21000 ; Acc.-rate = 0.27 ; Acc-rate-phi = 0.95
## iter. numb. 22000 ; Acc.-rate = 0.26 ; Acc-rate-phi = 0.96
## iter. numb. 23000 ; Acc.-rate = 0.26 ; Acc-rate-phi = 0.95
## iter. numb. 24000 ; Acc.-rate = 0.26 ; Acc-rate-phi = 0.95
## iter. numb. 25000 ; Acc.-rate = 0.28 ; Acc-rate-phi = 0.95
## iter. numb. 26000 ; Acc.-rate = 0.26 ; Acc-rate-phi = 0.95
## iter. numb. 27000 ; Acc.-rate = 0.25 ; Acc-rate-phi = 0.93
## iter. numb. 28000 ; Acc.-rate = 0.32 ; Acc-rate-phi = 0.94
## iter. numb. 29000 ; Acc.-rate = 0.34 ; Acc-rate-phi = 0.92
## iter. numb. 30000 ; Acc.-rate = 0.27 ; Acc-rate-phi = 0.94
## iter. numb. 31000 ; Acc.-rate = 0.33 ; Acc-rate-phi = 0.93
## iter. numb. 32000 ; Acc.-rate = 0.39 ; Acc-rate-phi = 0.91
## iter. numb. 33000 ; Acc.-rate = 0.35 ; Acc-rate-phi = 0.91
## iter. numb. 34000 ; Acc.-rate = 0.40 ; Acc-rate-phi = 0.89
## iter. numb. 35000 ; Acc.-rate = 0.38 ; Acc-rate-phi = 0.91
## iter. numb. 36000 ; Acc.-rate = 0.36 ; Acc-rate-phi = 0.91
## iter. numb. 37000 ; Acc.-rate = 0.34 ; Acc-rate-phi = 0.94
## iter. numb. 38000 ; Acc.-rate = 0.31 ; Acc-rate-phi = 0.92
## iter. numb. 39000 ; Acc.-rate = 0.32 ; Acc-rate-phi = 0.92
## iter. numb. 40000 ; Acc.-rate = 0.35 ; Acc-rate-phi = 0.95
## iter. numb. 41000 ; Acc.-rate = 0.28 ; Acc-rate-phi = 0.94
## iter. numb. 42000 ; Acc.-rate = 0.30 ; Acc-rate-phi = 0.94
## iter. numb. 43000 ; Acc.-rate = 0.25 ; Acc-rate-phi = 0.96
## iter. numb. 44000 ; Acc.-rate = 0.24 ; Acc-rate-phi = 0.94
## iter. numb. 45000 ; Acc.-rate = 0.23 ; Acc-rate-phi = 0.95
## iter. numb. 46000 ; Acc.-rate = 0.26 ; Acc-rate-phi = 0.96
## iter. numb. 47000 ; Acc.-rate = 0.25 ; Acc-rate-phi = 0.96
## iter. numb. 48000 ; Acc.-rate = 0.26 ; Acc-rate-phi = 0.96
## iter. numb. 49000 ; Acc.-rate = 0.25 ; Acc-rate-phi = 0.95
## iter. numb. 50000 ; Acc.-rate = 0.27 ; Acc-rate-phi = 0.96
## iter. numb. 51000 ; Acc.-rate = 0.25 ; Acc-rate-phi = 0.96
## iter. numb. 52000 ; Acc.-rate = 0.21 ; Acc-rate-phi = 0.96
## iter. numb. 53000 ; Acc.-rate = 0.23 ; Acc-rate-phi = 0.97
## iter. numb. 54000 ; Acc.-rate = 0.24 ; Acc-rate-phi = 0.96
## iter. numb. 55000 ; Acc.-rate = 0.27 ; Acc-rate-phi = 0.94
## iter. numb. 56000 ; Acc.-rate = 0.26 ; Acc-rate-phi = 0.97
## iter. numb. 57000 ; Acc.-rate = 0.25 ; Acc-rate-phi = 0.96
## iter. numb. 58000 ; Acc.-rate = 0.23 ; Acc-rate-phi = 0.96
## iter. numb. 59000 ; Acc.-rate = 0.26 ; Acc-rate-phi = 0.97
## iter. numb. 60000 ; Acc.-rate = 0.26 ; Acc-rate-phi = 0.96
## iter. numb. 61000 ; Acc.-rate = 0.27 ; Acc-rate-phi = 0.95
## iter. numb. 62000 ; Acc.-rate = 0.28 ; Acc-rate-phi = 0.95
## iter. numb. 63000 ; Acc.-rate = 0.29 ; Acc-rate-phi = 0.95
## iter. numb. 64000 ; Acc.-rate = 0.32 ; Acc-rate-phi = 0.94
## iter. numb. 65000 ; Acc.-rate = 0.32 ; Acc-rate-phi = 0.92
## iter. numb. 66000 ; Acc.-rate = 0.32 ; Acc-rate-phi = 0.93
## iter. numb. 67000 ; Acc.-rate = 0.35 ; Acc-rate-phi = 0.93
## iter. numb. 68000 ; Acc.-rate = 0.34 ; Acc-rate-phi = 0.94
## iter. numb. 69000 ; Acc.-rate = 0.30 ; Acc-rate-phi = 0.93
## iter. numb. 70000 ; Acc.-rate = 0.33 ; Acc-rate-phi = 0.94
## iter. numb. 71000 ; Acc.-rate = 0.40 ; Acc-rate-phi = 0.90
## iter. numb. 72000 ; Acc.-rate = 0.39 ; Acc-rate-phi = 0.91
## iter. numb. 73000 ; Acc.-rate = 0.38 ; Acc-rate-phi = 0.91
## iter. numb. 74000 ; Acc.-rate = 0.36 ; Acc-rate-phi = 0.91
```

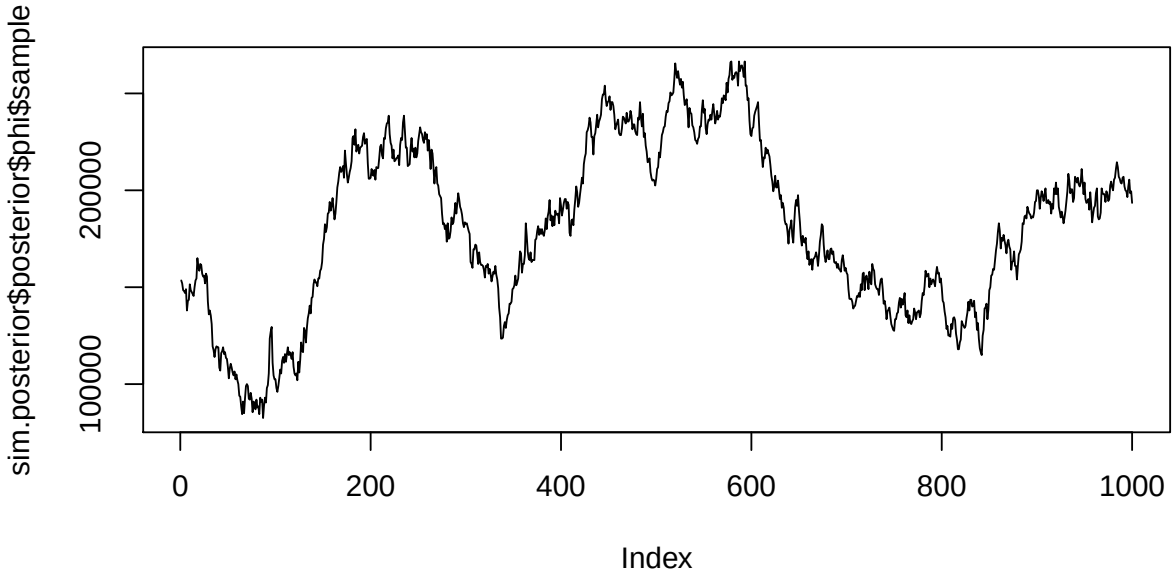
```
## iter. numb. 75000 ; Acc.-rate = 0.39 ; Acc-rate-phi = 0.90
## iter. numb. 76000 ; Acc.-rate = 0.43 ; Acc-rate-phi = 0.92
## iter. numb. 77000 ; Acc.-rate = 0.40 ; Acc-rate-phi = 0.89
## iter. numb. 78000 ; Acc.-rate = 0.42 ; Acc-rate-phi = 0.92
## iter. numb. 79000 ; Acc.-rate = 0.36 ; Acc-rate-phi = 0.90
## iter. numb. 80000 ; Acc.-rate = 0.34 ; Acc-rate-phi = 0.91
## iter. numb. 81000 ; Acc.-rate = 0.40 ; Acc-rate-phi = 0.91
## iter. numb. 82000 ; Acc.-rate = 0.43 ; Acc-rate-phi = 0.89
## iter. numb. 83000 ; Acc.-rate = 0.44 ; Acc-rate-phi = 0.88
## iter. numb. 84000 ; Acc.-rate = 0.43 ; Acc-rate-phi = 0.89
## iter. numb. 85000 ; Acc.-rate = 0.43 ; Acc-rate-phi = 0.90
## iter. numb. 86000 ; Acc.-rate = 0.34 ; Acc-rate-phi = 0.92
## iter. numb. 87000 ; Acc.-rate = 0.35 ; Acc-rate-phi = 0.93
## iter. numb. 88000 ; Acc.-rate = 0.33 ; Acc-rate-phi = 0.91
## iter. numb. 89000 ; Acc.-rate = 0.30 ; Acc-rate-phi = 0.92
## iter. numb. 90000 ; Acc.-rate = 0.29 ; Acc-rate-phi = 0.93
## iter. numb. 91000 ; Acc.-rate = 0.28 ; Acc-rate-phi = 0.94
## iter. numb. 92000 ; Acc.-rate = 0.31 ; Acc-rate-phi = 0.94
## iter. numb. 93000 ; Acc.-rate = 0.31 ; Acc-rate-phi = 0.93
## iter. numb. 94000 ; Acc.-rate = 0.29 ; Acc-rate-phi = 0.95
## iter. numb. 95000 ; Acc.-rate = 0.24 ; Acc-rate-phi = 0.94
## iter. numb. 96000 ; Acc.-rate = 0.30 ; Acc-rate-phi = 0.93
## iter. numb. 97000 ; Acc.-rate = 0.28 ; Acc-rate-phi = 0.93
## iter. numb. 98000 ; Acc.-rate = 0.26 ; Acc-rate-phi = 0.93
## iter. numb. 99000 ; Acc.-rate = 0.28 ; Acc-rate-phi = 0.94
## iter. numb. 100000 ; Acc.-rate = 0.29 ; Acc-rate-phi = 0.93
## MCMC performed: n.iter. =  1e+05 ; thinning =  100 ; burn.in =  0
## Only Bayesian estimation of model parameters

## [1] 100

##    iter.numb     Acc.rate Acc.rate.phi
##     1.00e+05     2.91e-01     9.32e-01
```

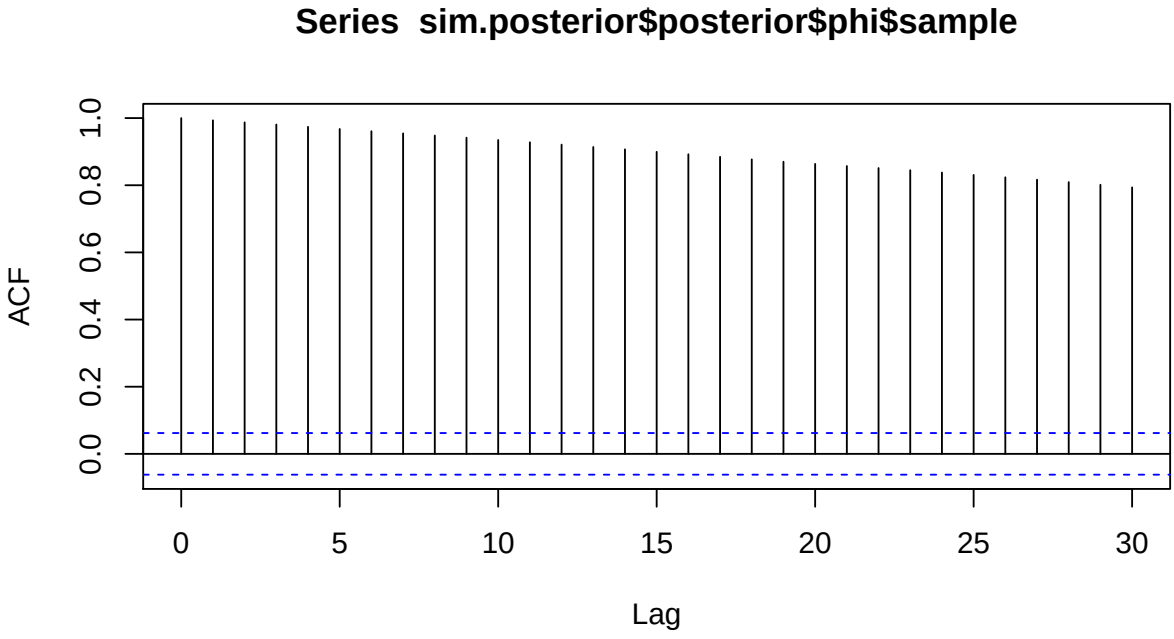

sim.posterior$posterior$beta$sample
2.5
3.5
4.5
5.5
0
200
400
600
800
1000
Index


Series sim.posterior$posterior$beta$sample
ACF
0.0
0.2
0.4
0.6
0.8
1.0
0
5
10
15
20
25
30
Lag


sim.posterior$posterior$sigmasq$sample
0.5
1.0
1.5
2.0
0
200
400
600
800
1000
Index


Series sim.posterior$posterior$sigmasq$sample
ACF
0.0
0.2
0.4
0.6
0.8
1.0
0
5
10
15
20
25
30
Lag





Problem 5

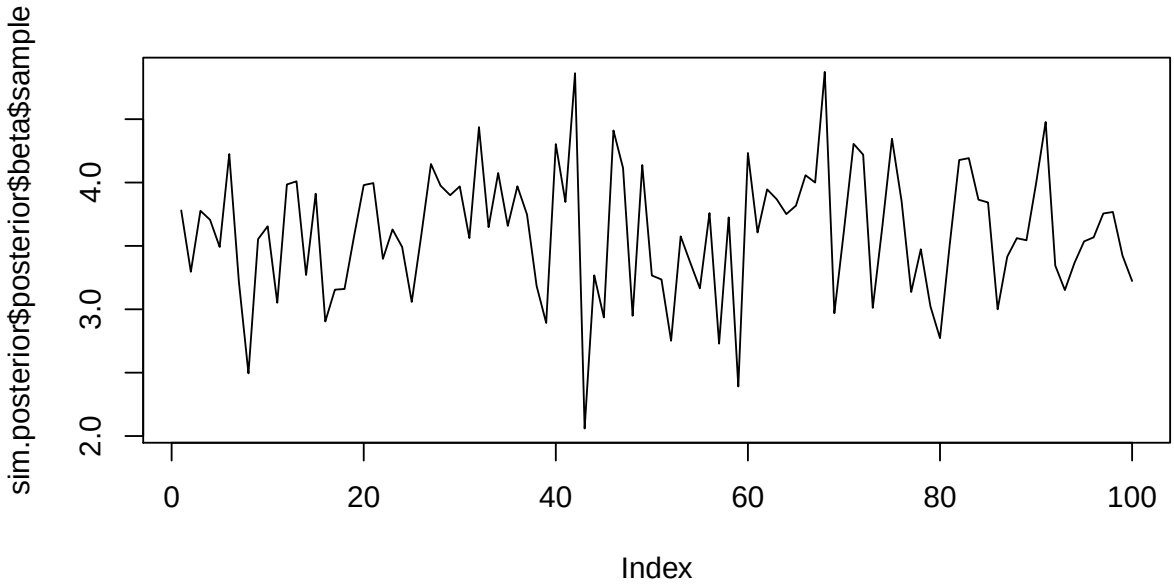
There is not much beyond changing S.scale and phi.scale that we can do to reduce the autocorrelation. So we will run a long chain and subsample it. Run the same code as in part (4), increasing to thin = 1e4 and n.iter = 1e7. Now do the following with your sampled parameters.

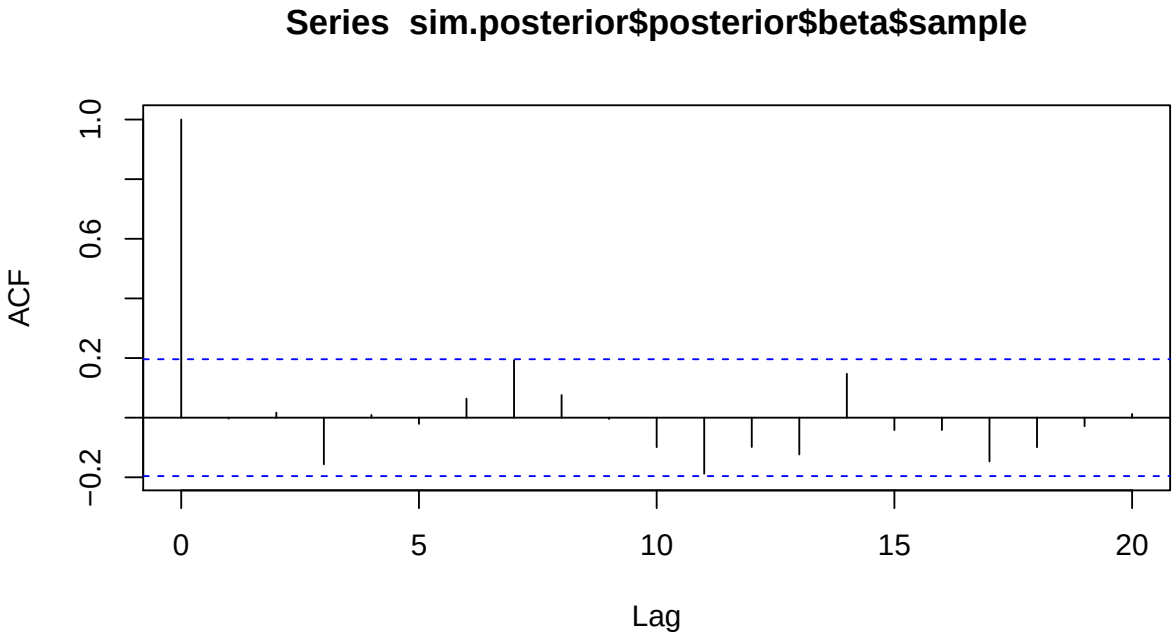
- **Make trace plots and ACF plots. Choose a burn-in to discard and make them again.**
- **Calculate the effective sample size you have for each parameter. If they are not at least 100 for each parameter, go back and run a longer chain.**
- **Plot the marginal posterior distributions for each parameter using histograms and/or kernel density estimators.**

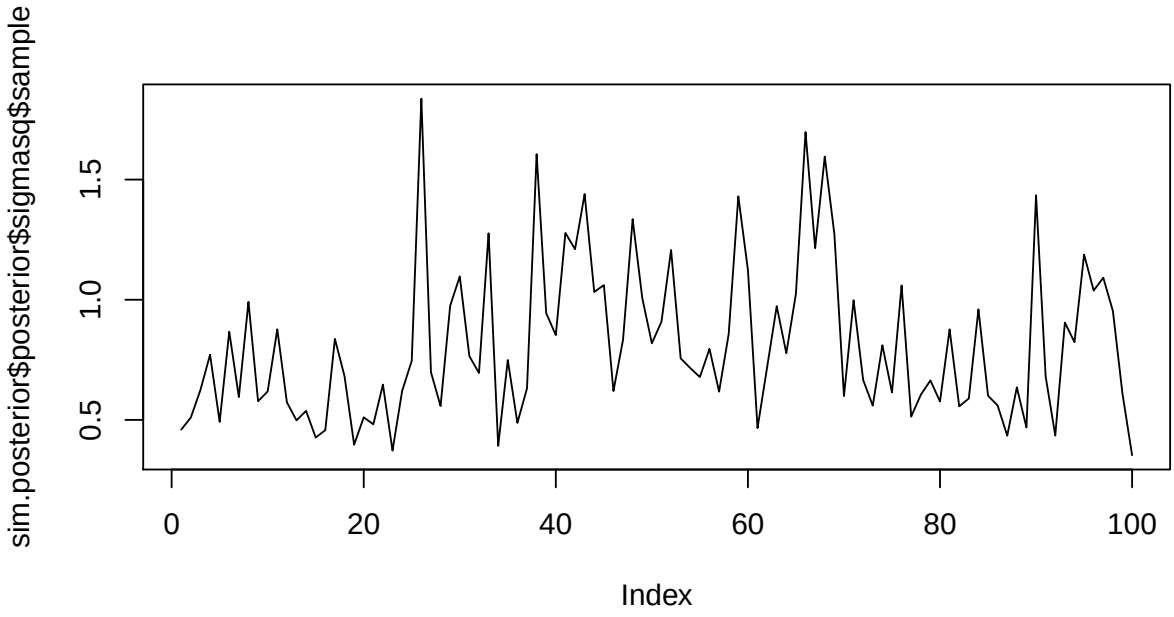
```
## pois.krige.bayes: model with mean being constant
## iter. numb. 1000 ; Acc.-rate = 0.37 ; Acc-rate-phi = 0.92
## iter. numb. 2000 ; Acc.-rate = 0.38 ; Acc-rate-phi = 0.92
## iter. numb. 3000 ; Acc.-rate = 0.36 ; Acc-rate-phi = 0.91
## iter. numb. 4000 ; Acc.-rate = 0.38 ; Acc-rate-phi = 0.91
## iter. numb. 5000 ; Acc.-rate = 0.40 ; Acc-rate-phi = 0.92
## iter. numb. 6000 ; Acc.-rate = 0.37 ; Acc-rate-phi = 0.91
## iter. numb. 7000 ; Acc.-rate = 0.30 ; Acc-rate-phi = 0.92
## iter. numb. 8000 ; Acc.-rate = 0.32 ; Acc-rate-phi = 0.93
## iter. numb. 9000 ; Acc.-rate = 0.35 ; Acc-rate-phi = 0.93
## iter. numb. 10000 ; Acc.-rate = 0.38 ; Acc-rate-phi = 0.92
## iter. numb. 11000 ; Acc.-rate = 0.43 ; Acc-rate-phi = 0.90
## iter. numb. 12000 ; Acc.-rate = 0.39 ; Acc-rate-phi = 0.92
## iter. numb. 13000 ; Acc.-rate = 0.41 ; Acc-rate-phi = 0.91
## iter. numb. 14000 ; Acc.-rate = 0.41 ; Acc-rate-phi = 0.92
## iter. numb. 15000 ; Acc.-rate = 0.29 ; Acc-rate-phi = 0.93
## iter. numb. 16000 ; Acc.-rate = 0.31 ; Acc-rate-phi = 0.93
## iter. numb. 17000 ; Acc.-rate = 0.29 ; Acc-rate-phi = 0.94
## iter. numb. 18000 ; Acc.-rate = 0.33 ; Acc-rate-phi = 0.93
## iter. numb. 19000 ; Acc.-rate = 0.36 ; Acc-rate-phi = 0.92
## iter. numb. 20000 ; Acc.-rate = 0.34 ; Acc-rate-phi = 0.93
## iter. numb. 21000 ; Acc.-rate = 0.27 ; Acc-rate-phi = 0.93
## iter. numb. 22000 ; Acc.-rate = 0.31 ; Acc-rate-phi = 0.95
## iter. numb. 23000 ; Acc.-rate = 0.35 ; Acc-rate-phi = 0.93
## iter. numb. 24000 ; Acc.-rate = 0.33 ; Acc-rate-phi = 0.91
## iter. numb. 25000 ; Acc.-rate = 0.38 ; Acc-rate-phi = 0.92
## iter. numb. 26000 ; Acc.-rate = 0.31 ; Acc-rate-phi = 0.93
## iter. numb. 27000 ; Acc.-rate = 0.28 ; Acc-rate-phi = 0.93
## iter. numb. 28000 ; Acc.-rate = 0.26 ; Acc-rate-phi = 0.96
## iter. numb. 29000 ; Acc.-rate = 0.30 ; Acc-rate-phi = 0.96
## iter. numb. 30000 ; Acc.-rate = 0.29 ; Acc-rate-phi = 0.94
## iter. numb. 31000 ; Acc.-rate = 0.28 ; Acc-rate-phi = 0.95
## iter. numb. 32000 ; Acc.-rate = 0.26 ; Acc-rate-phi = 0.96
## iter. numb. 33000 ; Acc.-rate = 0.24 ; Acc-rate-phi = 0.96
## iter. numb. 34000 ; Acc.-rate = 0.25 ; Acc-rate-phi = 0.96
## iter. numb. 35000 ; Acc.-rate = 0.22 ; Acc-rate-phi = 0.95
## iter. numb. 36000 ; Acc.-rate = 0.19 ; Acc-rate-phi = 0.97
## iter. numb. 37000 ; Acc.-rate = 0.22 ; Acc-rate-phi = 0.97
## iter. numb. 38000 ; Acc.-rate = 0.20 ; Acc-rate-phi = 0.96
## iter. numb. 39000 ; Acc.-rate = 0.20 ; Acc-rate-phi = 0.96
## iter. numb. 40000 ; Acc.-rate = 0.20 ; Acc-rate-phi = 0.97
## iter. numb. 41000 ; Acc.-rate = 0.22 ; Acc-rate-phi = 0.94
## iter. numb. 42000 ; Acc.-rate = 0.23 ; Acc-rate-phi = 0.96
## iter. numb. 43000 ; Acc.-rate = 0.21 ; Acc-rate-phi = 0.96
## iter. numb. 44000 ; Acc.-rate = 0.24 ; Acc-rate-phi = 0.96
## iter. numb. 45000 ; Acc.-rate = 0.24 ; Acc-rate-phi = 0.97
## iter. numb. 46000 ; Acc.-rate = 0.22 ; Acc-rate-phi = 0.96
```

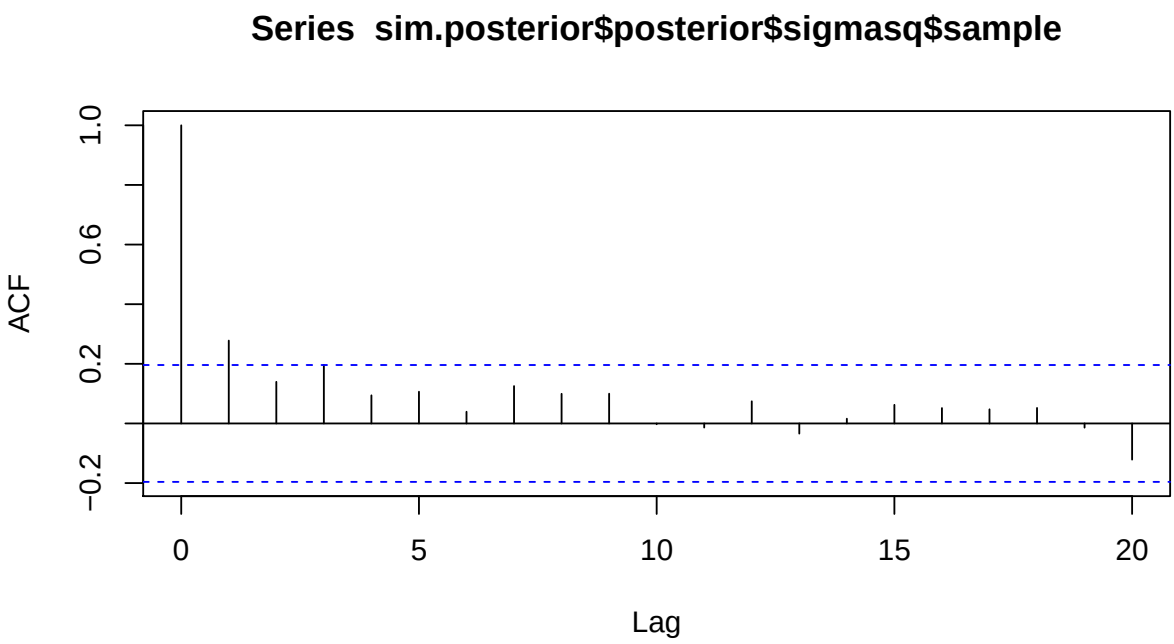
```
## iter. numb. 47000 ; Acc.-rate = 0.21 ; Acc-rate-phi = 0.95
## iter. numb. 48000 ; Acc.-rate = 0.20 ; Acc-rate-phi = 0.97
## iter. numb. 49000 ; Acc.-rate = 0.19 ; Acc-rate-phi = 0.91
## iter. numb. 50000 ; Acc.-rate = 0.22 ; Acc-rate-phi = 0.97
## iter. numb. 51000 ; Acc.-rate = 0.19 ; Acc-rate-phi = 0.96
## iter. numb. 52000 ; Acc.-rate = 0.23 ; Acc-rate-phi = 0.96
## iter. numb. 53000 ; Acc.-rate = 0.19 ; Acc-rate-phi = 0.93
## iter. numb. 54000 ; Acc.-rate = 0.24 ; Acc-rate-phi = 0.96
## iter. numb. 55000 ; Acc.-rate = 0.19 ; Acc-rate-phi = 0.93
## iter. numb. 56000 ; Acc.-rate = 0.21 ; Acc-rate-phi = 0.98
## iter. numb. 57000 ; Acc.-rate = 0.21 ; Acc-rate-phi = 0.96
## iter. numb. 58000 ; Acc.-rate = 0.22 ; Acc-rate-phi = 0.97
## iter. numb. 59000 ; Acc.-rate = 0.23 ; Acc-rate-phi = 0.97
## iter. numb. 60000 ; Acc.-rate = 0.21 ; Acc-rate-phi = 0.97
## iter. numb. 61000 ; Acc.-rate = 0.22 ; Acc-rate-phi = 0.96
## iter. numb. 62000 ; Acc.-rate = 0.20 ; Acc-rate-phi = 0.97
## iter. numb. 63000 ; Acc.-rate = 0.22 ; Acc-rate-phi = 0.96
## iter. numb. 64000 ; Acc.-rate = 0.23 ; Acc-rate-phi = 0.96
## iter. numb. 65000 ; Acc.-rate = 0.22 ; Acc-rate-phi = 0.96
## iter. numb. 66000 ; Acc.-rate = 0.23 ; Acc-rate-phi = 0.97
## iter. numb. 67000 ; Acc.-rate = 0.20 ; Acc-rate-phi = 0.95
## iter. numb. 68000 ; Acc.-rate = 0.22 ; Acc-rate-phi = 0.96
## iter. numb. 69000 ; Acc.-rate = 0.24 ; Acc-rate-phi = 0.97
## iter. numb. 70000 ; Acc.-rate = 0.26 ; Acc-rate-phi = 0.97
## iter. numb. 71000 ; Acc.-rate = 0.22 ; Acc-rate-phi = 0.96
## iter. numb. 72000 ; Acc.-rate = 0.26 ; Acc-rate-phi = 0.96
## iter. numb. 73000 ; Acc.-rate = 0.25 ; Acc-rate-phi = 0.95
## iter. numb. 74000 ; Acc.-rate = 0.27 ; Acc-rate-phi = 0.95
## iter. numb. 75000 ; Acc.-rate = 0.27 ; Acc-rate-phi = 0.96
## iter. numb. 76000 ; Acc.-rate = 0.30 ; Acc-rate-phi = 0.94
## iter. numb. 77000 ; Acc.-rate = 0.33 ; Acc-rate-phi = 0.92
## iter. numb. 78000 ; Acc.-rate = 0.29 ; Acc-rate-phi = 0.93
## iter. numb. 79000 ; Acc.-rate = 0.30 ; Acc-rate-phi = 0.94
## iter. numb. 80000 ; Acc.-rate = 0.32 ; Acc-rate-phi = 0.92
## iter. numb. 81000 ; Acc.-rate = 0.35 ; Acc-rate-phi = 0.92
## iter. numb. 82000 ; Acc.-rate = 0.35 ; Acc-rate-phi = 0.93
## iter. numb. 83000 ; Acc.-rate = 0.37 ; Acc-rate-phi = 0.91
## iter. numb. 84000 ; Acc.-rate = 0.33 ; Acc-rate-phi = 0.93
## iter. numb. 85000 ; Acc.-rate = 0.33 ; Acc-rate-phi = 0.94
## iter. numb. 86000 ; Acc.-rate = 0.35 ; Acc-rate-phi = 0.92
## iter. numb. 87000 ; Acc.-rate = 0.40 ; Acc-rate-phi = 0.91
## iter. numb. 88000 ; Acc.-rate = 0.40 ; Acc-rate-phi = 0.91
## iter. numb. 89000 ; Acc.-rate = 0.35 ; Acc-rate-phi = 0.92
## iter. numb. 90000 ; Acc.-rate = 0.34 ; Acc-rate-phi = 0.92
## iter. numb. 91000 ; Acc.-rate = 0.33 ; Acc-rate-phi = 0.92
## iter. numb. 92000 ; Acc.-rate = 0.27 ; Acc-rate-phi = 0.94
## iter. numb. 93000 ; Acc.-rate = 0.30 ; Acc-rate-phi = 0.94
## iter. numb. 94000 ; Acc.-rate = 0.29 ; Acc-rate-phi = 0.95
## iter. numb. 95000 ; Acc.-rate = 0.29 ; Acc-rate-phi = 0.94
## iter. numb. 96000 ; Acc.-rate = 0.31 ; Acc-rate-phi = 0.94
## iter. numb. 97000 ; Acc.-rate = 0.29 ; Acc-rate-phi = 0.93
## iter. numb. 98000 ; Acc.-rate = 0.36 ; Acc-rate-phi = 0.94
## iter. numb. 99000 ; Acc.-rate = 0.36 ; Acc-rate-phi = 0.91
## iter. numb. 100000 ; Acc.-rate = 0.41 ; Acc-rate-phi = 0.89
```

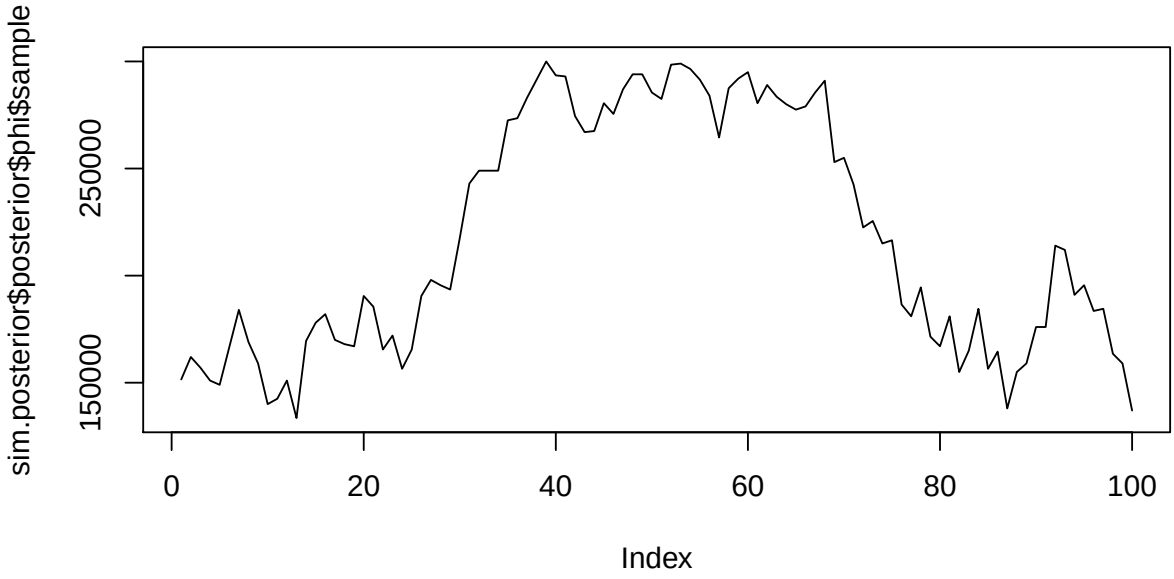
```
## MCMC performed: n.iter. =  1e+05 ; thinning =  1000 ; burn.in =  0
## Only Bayesian estimation of model parameters

## [1] 100

##    iter.numb     Acc.rate Acc.rate.phi
##     1.00e+05     4.15e-01     8.90e-01
```

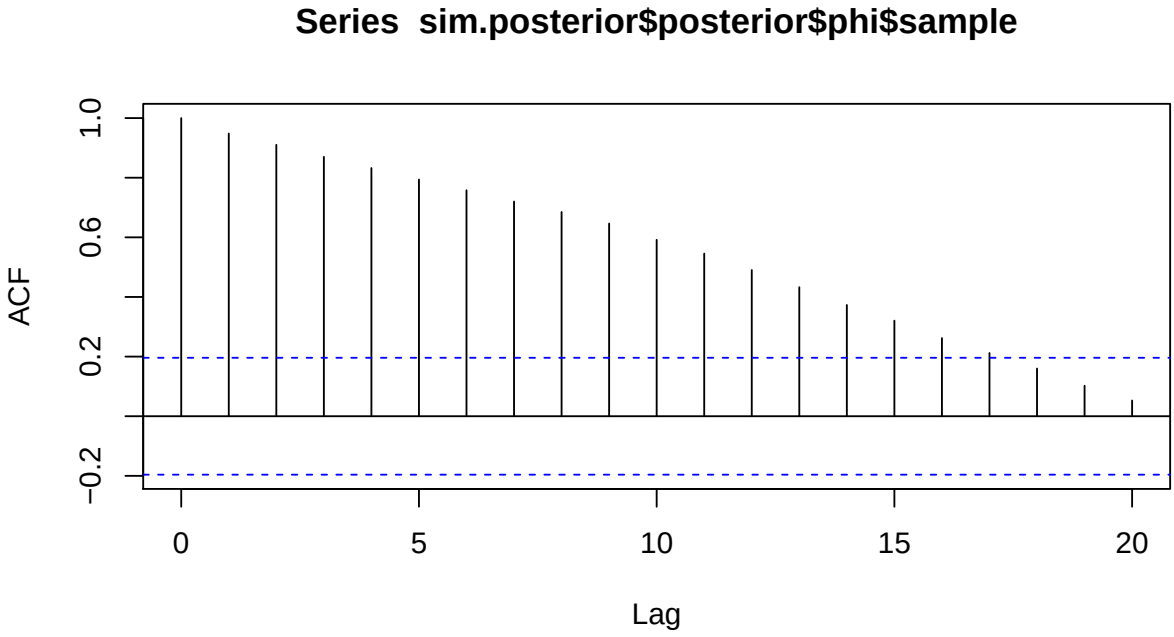


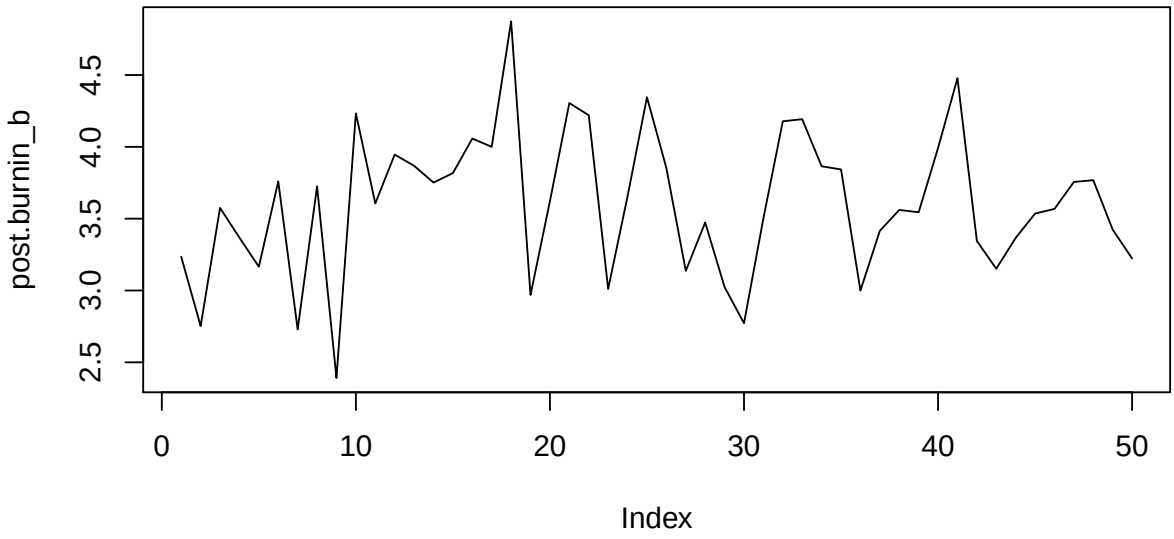


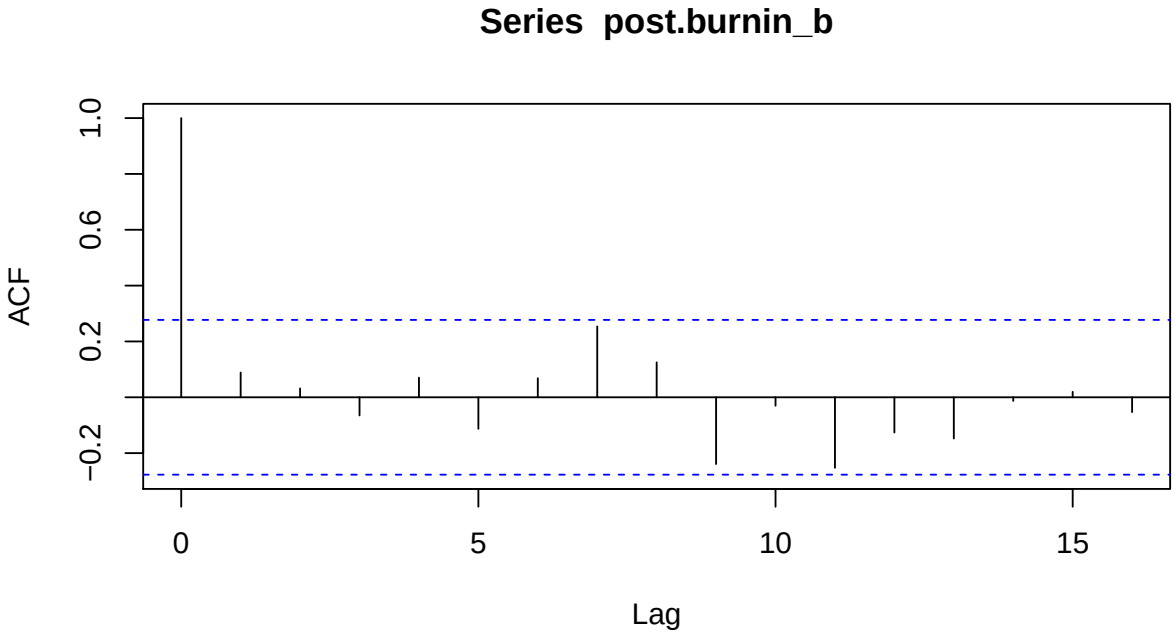

sim.posterior$posterior$sigmasq$sample
0.5
1.0
1.5
0
20
40
60
80
100
Index

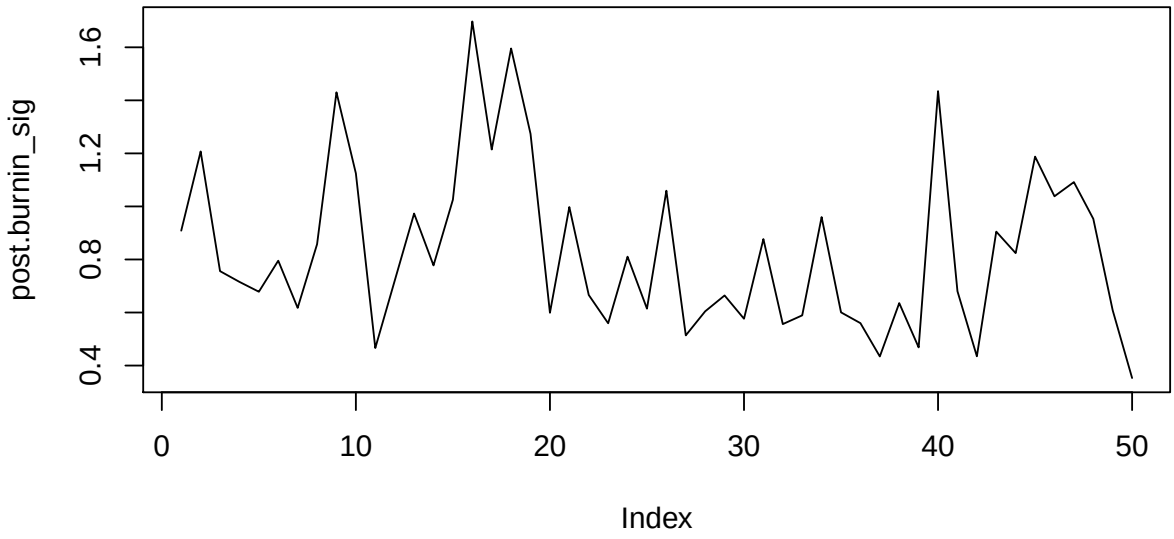

Series sim.posterior$posterior$sigmasq$sample
ACF
−0.2
0.2
0.6
1.0
0
5
10
15
20
Lag

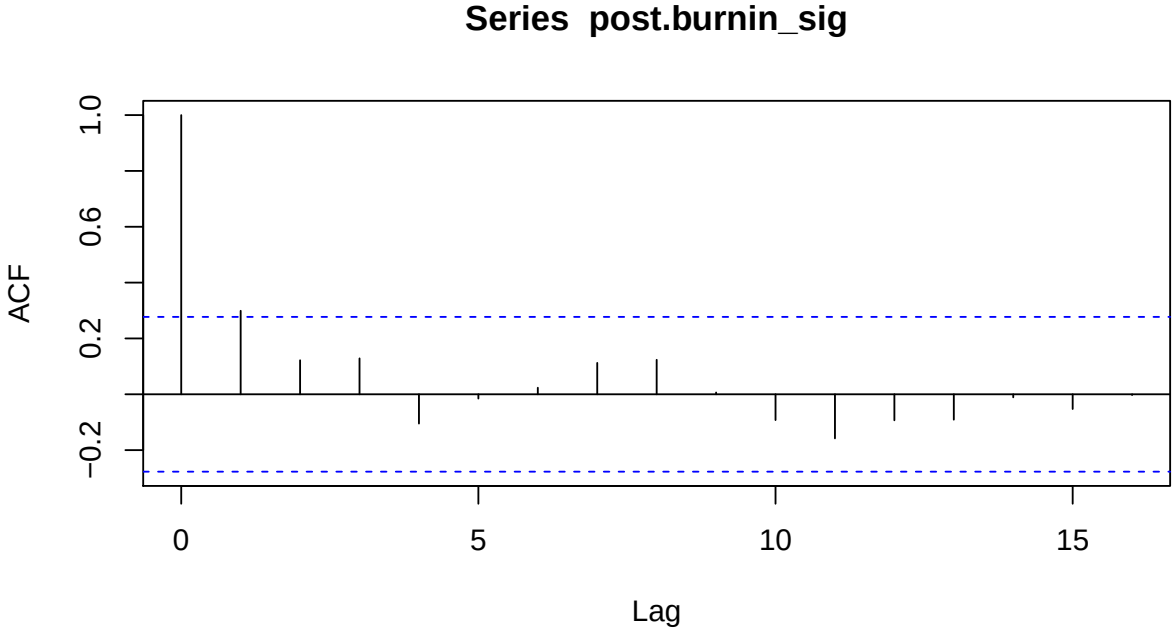

sim.posterior$posterior$phi$sample
150000
250000
0
20
40
60
80
100
Index

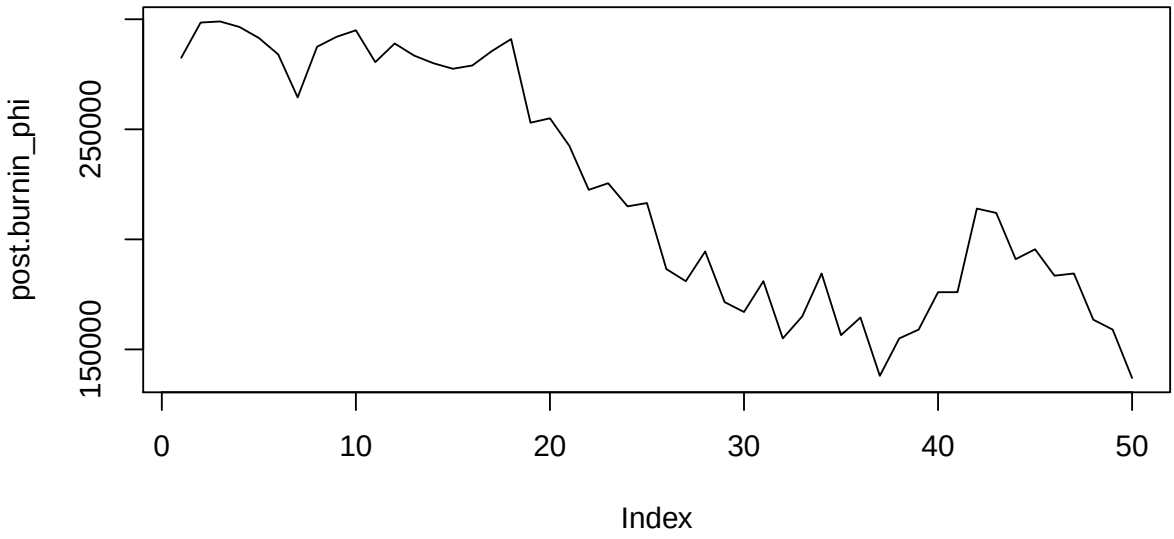

Series sim.posterior$posterior$phi$sample
ACF
−0.2
0.2
0.6
1.0
0
5
10
15
20
Lag

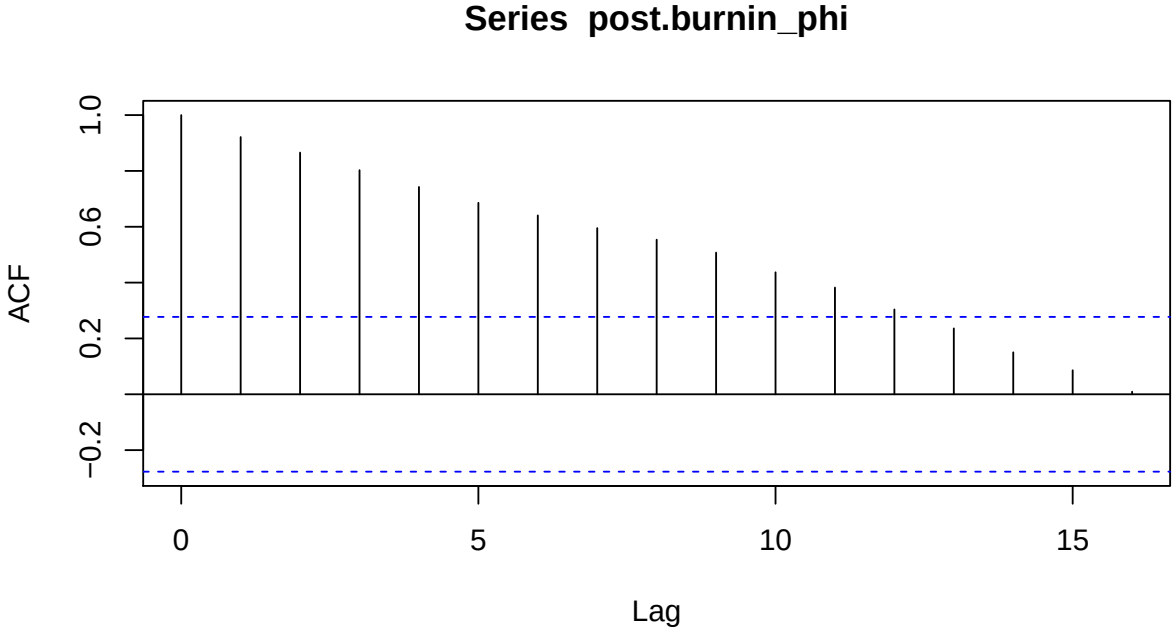

post.burnin_b
2.5
3.0
3.5
4.0
4.5
0
10
20
30
40
50
Index


Series post.burnin_b
ACF
−0.2
0.2
0.6
1.0
0
5
10
15
Lag


post.burnin_sig
0.4
0.8
1.2
1.6
0
10
20
30
40
50
Index


Series post.burnin_sig
ACF
−0.2
0.2
0.6
1.0
0
5
10
15
Lag

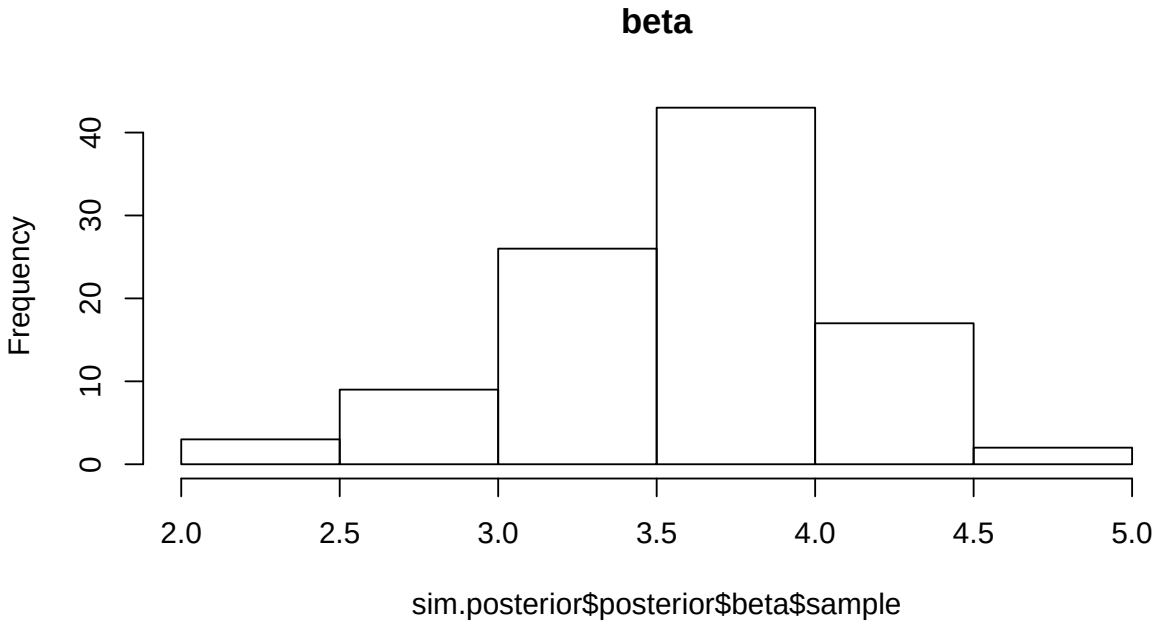


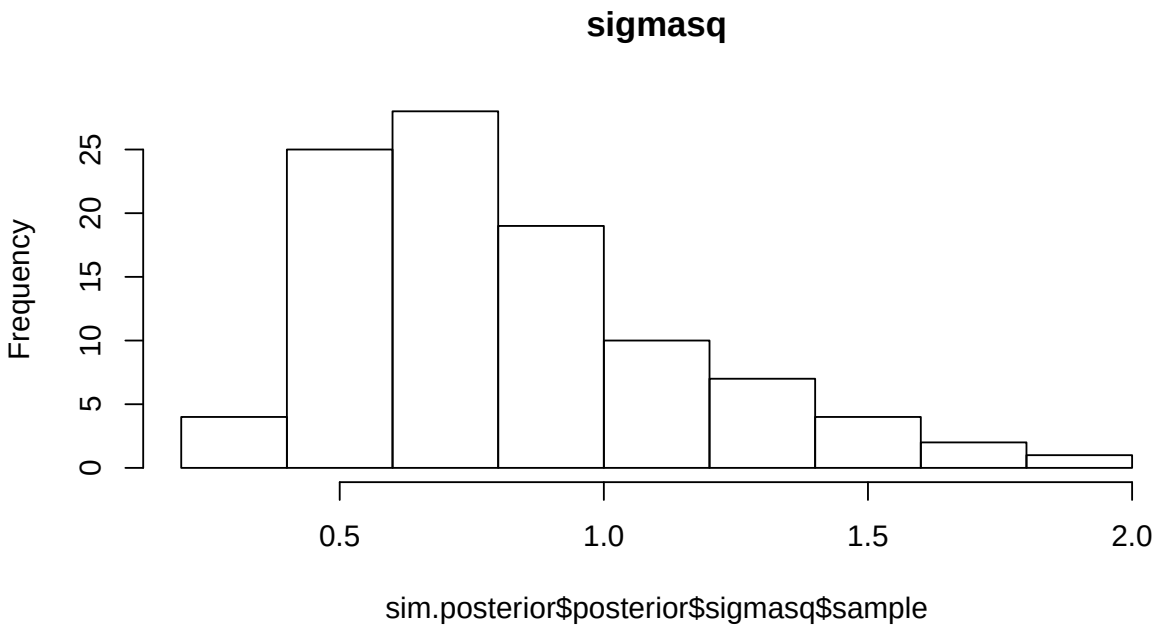


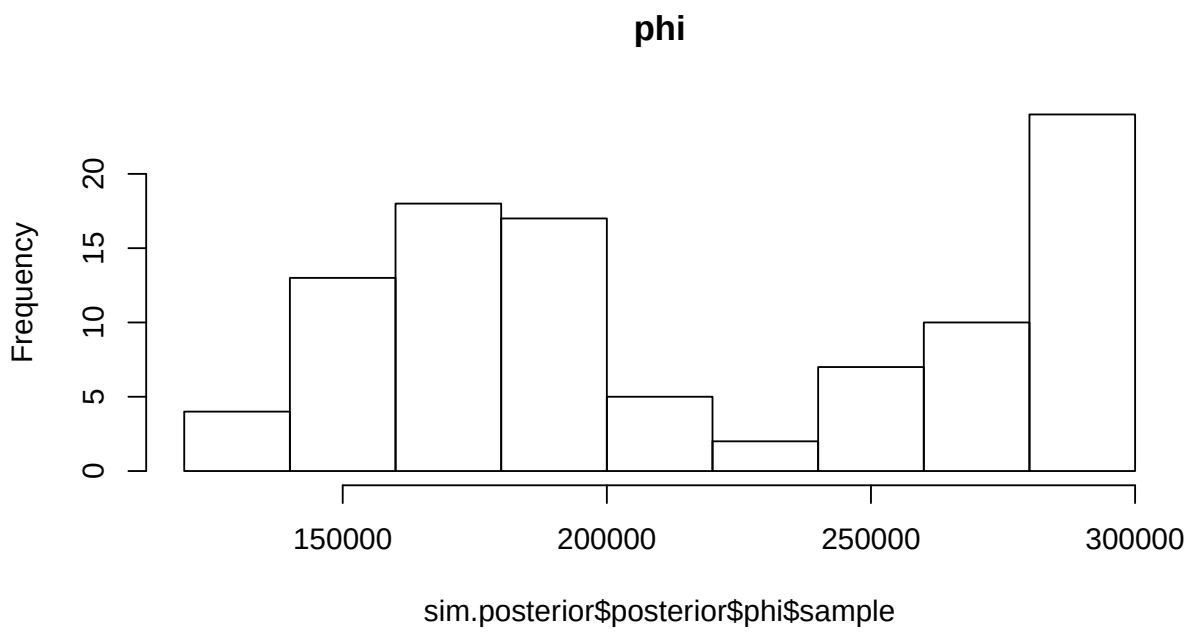

```
##       se
## 18.75106
```

```
##       se
## 7.025655
```

```
##       se
## 38.39557
```


beta
Frequency
0
10
20
30
40
2.0
2.5
3.0
3.5
4.0
4.5
5.0
sim.posterior$posterior$beta$sample


sigmasq
Frequency
0
5
10
15
20
25
0.5
1.0
1.5
2.0
sim.posterior$posterior$sigmasq$sample



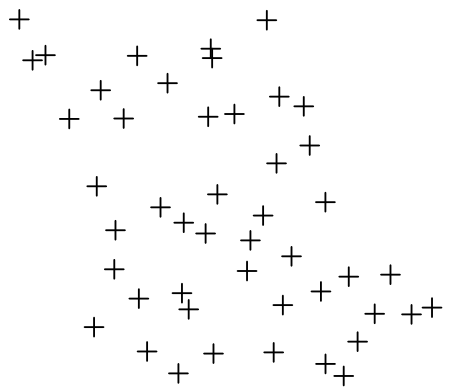
Problem 6

Do one final run of the chain, the same as in (5), but this time specifying the burn-in value you chose in (5) and also modifying the code to include prediction at the locations in dove.grid. Create two color or grayscale plots, showing the posterior mean and standard deviation for the underlying mean surface (exp η) at each location in dove.grid.

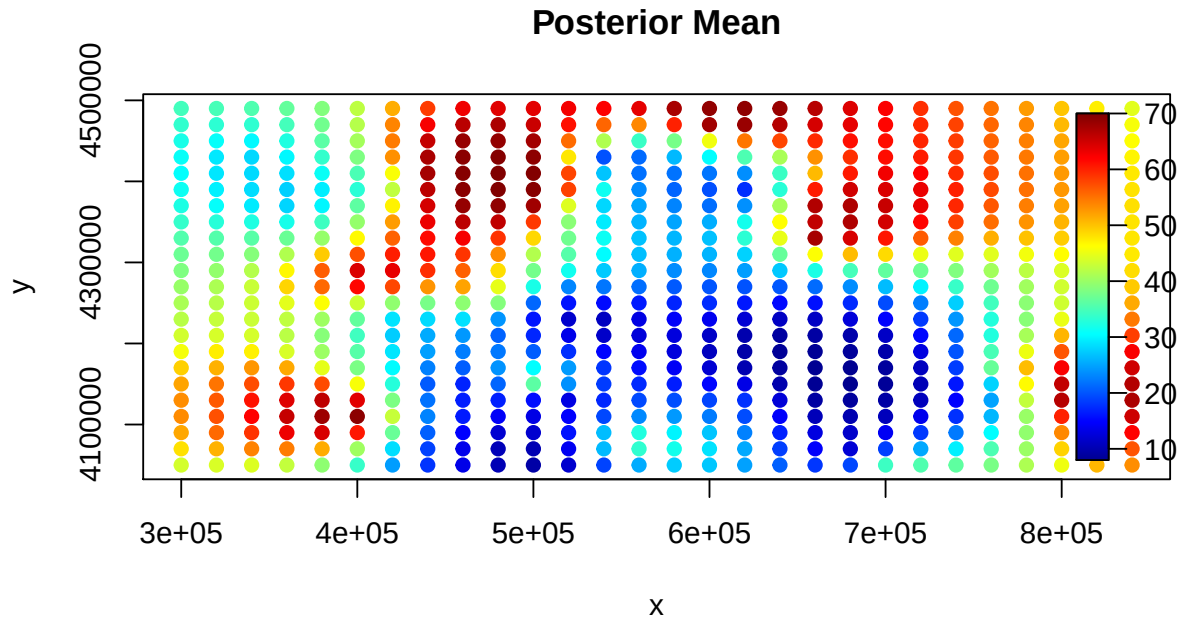
```
## pois.krige.bayes: model with mean being constant
## burn-in =  50 is finished; Acc.-rate = 0.60 ; Acc-rate-phi = 0.80
## iter. numb. 1050 ; Acc.-rate = 0.58 ; Acc-rate-phi = 0.91
## iter. numb. 2050 ; Acc.-rate = 0.59 ; Acc-rate-phi = 0.90
## iter. numb. 3050 ; Acc.-rate = 0.58 ; Acc-rate-phi = 0.90
## iter. numb. 4050 ; Acc.-rate = 0.59 ; Acc-rate-phi = 0.90
## iter. numb. 5050 ; Acc.-rate = 0.55 ; Acc-rate-phi = 0.89
## iter. numb. 6050 ; Acc.-rate = 0.53 ; Acc-rate-phi = 0.94
## iter. numb. 7050 ; Acc.-rate = 0.56 ; Acc-rate-phi = 0.91
## iter. numb. 8050 ; Acc.-rate = 0.55 ; Acc-rate-phi = 0.92
## iter. numb. 9050 ; Acc.-rate = 0.58 ; Acc-rate-phi = 0.90
## iter. numb. 10050 ; Acc.-rate = 0.61 ; Acc-rate-phi = 0.90
## iter. numb. 11050 ; Acc.-rate = 0.58 ; Acc-rate-phi = 0.92
## iter. numb. 12050 ; Acc.-rate = 0.55 ; Acc-rate-phi = 0.93
## iter. numb. 13050 ; Acc.-rate = 0.57 ; Acc-rate-phi = 0.90
## iter. numb. 14050 ; Acc.-rate = 0.64 ; Acc-rate-phi = 0.90
## iter. numb. 15050 ; Acc.-rate = 0.61 ; Acc-rate-phi = 0.91
## iter. numb. 16050 ; Acc.-rate = 0.57 ; Acc-rate-phi = 0.91
## iter. numb. 17050 ; Acc.-rate = 0.62 ; Acc-rate-phi = 0.90
## iter. numb. 18050 ; Acc.-rate = 0.58 ; Acc-rate-phi = 0.90
## iter. numb. 19050 ; Acc.-rate = 0.59 ; Acc-rate-phi = 0.90
## iter. numb. 20050 ; Acc.-rate = 0.58 ; Acc-rate-phi = 0.92
## iter. numb. 21050 ; Acc.-rate = 0.59 ; Acc-rate-phi = 0.91
```

```
## iter. numb. 22050 ; Acc.-rate = 0.67 ; Acc-rate-phi = 0.88
## iter. numb. 23050 ; Acc.-rate = 0.68 ; Acc-rate-phi = 0.85
## iter. numb. 24050 ; Acc.-rate = 0.69 ; Acc-rate-phi = 0.86
## iter. numb. 25050 ; Acc.-rate = 0.67 ; Acc-rate-phi = 0.85
## iter. numb. 26050 ; Acc.-rate = 0.62 ; Acc-rate-phi = 0.90
## iter. numb. 27050 ; Acc.-rate = 0.61 ; Acc-rate-phi = 0.91
## iter. numb. 28050 ; Acc.-rate = 0.64 ; Acc-rate-phi = 0.90
## iter. numb. 29050 ; Acc.-rate = 0.65 ; Acc-rate-phi = 0.88
## iter. numb. 30050 ; Acc.-rate = 0.65 ; Acc-rate-phi = 0.88
## iter. numb. 31050 ; Acc.-rate = 0.70 ; Acc-rate-phi = 0.84
## iter. numb. 32050 ; Acc.-rate = 0.71 ; Acc-rate-phi = 0.83
## iter. numb. 33050 ; Acc.-rate = 0.69 ; Acc-rate-phi = 0.86
## iter. numb. 34050 ; Acc.-rate = 0.71 ; Acc-rate-phi = 0.84
## iter. numb. 35050 ; Acc.-rate = 0.69 ; Acc-rate-phi = 0.88
## iter. numb. 36050 ; Acc.-rate = 0.63 ; Acc-rate-phi = 0.88
## iter. numb. 37050 ; Acc.-rate = 0.64 ; Acc-rate-phi = 0.90
## iter. numb. 38050 ; Acc.-rate = 0.63 ; Acc-rate-phi = 0.86
## iter. numb. 39050 ; Acc.-rate = 0.64 ; Acc-rate-phi = 0.89
## iter. numb. 40050 ; Acc.-rate = 0.69 ; Acc-rate-phi = 0.84
## iter. numb. 41050 ; Acc.-rate = 0.64 ; Acc-rate-phi = 0.87
## iter. numb. 42050 ; Acc.-rate = 0.59 ; Acc-rate-phi = 0.88
## iter. numb. 43050 ; Acc.-rate = 0.62 ; Acc-rate-phi = 0.90
## iter. numb. 44050 ; Acc.-rate = 0.61 ; Acc-rate-phi = 0.88
## iter. numb. 45050 ; Acc.-rate = 0.67 ; Acc-rate-phi = 0.87
## iter. numb. 46050 ; Acc.-rate = 0.68 ; Acc-rate-phi = 0.85
## iter. numb. 47050 ; Acc.-rate = 0.67 ; Acc-rate-phi = 0.88
## iter. numb. 48050 ; Acc.-rate = 0.64 ; Acc-rate-phi = 0.88
## iter. numb. 49050 ; Acc.-rate = 0.68 ; Acc-rate-phi = 0.88
## iter. numb. 50050 ; Acc.-rate = 0.80 ; Acc-rate-phi = 0.81
## iter. numb. 51050 ; Acc.-rate = 0.80 ; Acc-rate-phi = 0.77
## iter. numb. 52050 ; Acc.-rate = 0.83 ; Acc-rate-phi = 0.71
## iter. numb. 53050 ; Acc.-rate = 0.84 ; Acc-rate-phi = 0.64
## iter. numb. 54050 ; Acc.-rate = 0.87 ; Acc-rate-phi = 0.64
## iter. numb. 55050 ; Acc.-rate = 0.86 ; Acc-rate-phi = 0.72
## iter. numb. 56050 ; Acc.-rate = 0.84 ; Acc-rate-phi = 0.70
## iter. numb. 57050 ; Acc.-rate = 0.84 ; Acc-rate-phi = 0.68
## iter. numb. 58050 ; Acc.-rate = 0.84 ; Acc-rate-phi = 0.61
## iter. numb. 59050 ; Acc.-rate = 0.85 ; Acc-rate-phi = 0.62
## iter. numb. 60050 ; Acc.-rate = 0.87 ; Acc-rate-phi = 0.64
## iter. numb. 61050 ; Acc.-rate = 0.82 ; Acc-rate-phi = 0.74
## iter. numb. 62050 ; Acc.-rate = 0.79 ; Acc-rate-phi = 0.79
## iter. numb. 63050 ; Acc.-rate = 0.70 ; Acc-rate-phi = 0.85
## iter. numb. 64050 ; Acc.-rate = 0.68 ; Acc-rate-phi = 0.85
## iter. numb. 65050 ; Acc.-rate = 0.74 ; Acc-rate-phi = 0.85
## iter. numb. 66050 ; Acc.-rate = 0.76 ; Acc-rate-phi = 0.84
## iter. numb. 67050 ; Acc.-rate = 0.78 ; Acc-rate-phi = 0.78
## iter. numb. 68050 ; Acc.-rate = 0.67 ; Acc-rate-phi = 0.82
## iter. numb. 69050 ; Acc.-rate = 0.69 ; Acc-rate-phi = 0.86
## iter. numb. 70050 ; Acc.-rate = 0.72 ; Acc-rate-phi = 0.86
## iter. numb. 71050 ; Acc.-rate = 0.72 ; Acc-rate-phi = 0.85
## iter. numb. 72050 ; Acc.-rate = 0.70 ; Acc-rate-phi = 0.87
## iter. numb. 73050 ; Acc.-rate = 0.68 ; Acc-rate-phi = 0.84
## iter. numb. 74050 ; Acc.-rate = 0.68 ; Acc-rate-phi = 0.88
## iter. numb. 75050 ; Acc.-rate = 0.64 ; Acc-rate-phi = 0.87
```

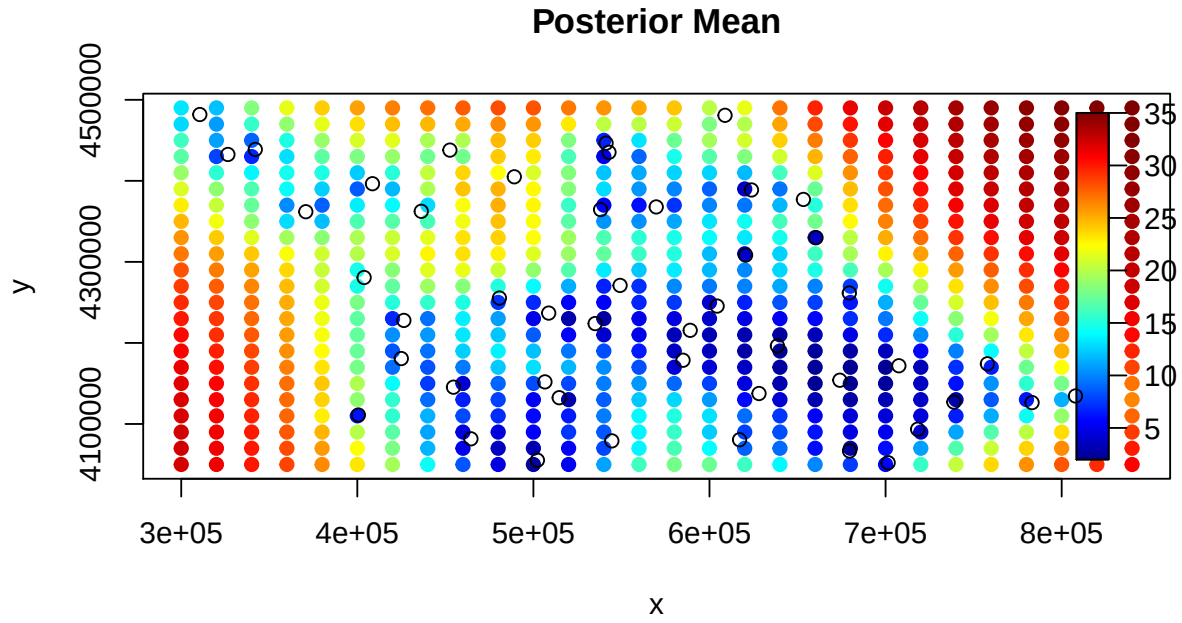
```
## iter. numb. 76050 ; Acc.-rate = 0.68 ; Acc-rate-phi = 0.88
## iter. numb. 77050 ; Acc.-rate = 0.67 ; Acc-rate-phi = 0.86
## iter. numb. 78050 ; Acc.-rate = 0.67 ; Acc-rate-phi = 0.86
## iter. numb. 79050 ; Acc.-rate = 0.69 ; Acc-rate-phi = 0.88
## iter. numb. 80050 ; Acc.-rate = 0.67 ; Acc-rate-phi = 0.87
## iter. numb. 81050 ; Acc.-rate = 0.69 ; Acc-rate-phi = 0.86
## iter. numb. 82050 ; Acc.-rate = 0.72 ; Acc-rate-phi = 0.82
## iter. numb. 83050 ; Acc.-rate = 0.76 ; Acc-rate-phi = 0.82
## iter. numb. 84050 ; Acc.-rate = 0.66 ; Acc-rate-phi = 0.87
## iter. numb. 85050 ; Acc.-rate = 0.70 ; Acc-rate-phi = 0.85
## iter. numb. 86050 ; Acc.-rate = 0.68 ; Acc-rate-phi = 0.88
## iter. numb. 87050 ; Acc.-rate = 0.70 ; Acc-rate-phi = 0.84
## iter. numb. 88050 ; Acc.-rate = 0.73 ; Acc-rate-phi = 0.85
## iter. numb. 89050 ; Acc.-rate = 0.70 ; Acc-rate-phi = 0.86
## iter. numb. 90050 ; Acc.-rate = 0.62 ; Acc-rate-phi = 0.88
## iter. numb. 91050 ; Acc.-rate = 0.58 ; Acc-rate-phi = 0.88
## iter. numb. 92050 ; Acc.-rate = 0.58 ; Acc-rate-phi = 0.91
## iter. numb. 93050 ; Acc.-rate = 0.58 ; Acc-rate-phi = 0.90
## iter. numb. 94050 ; Acc.-rate = 0.61 ; Acc-rate-phi = 0.90
## iter. numb. 95050 ; Acc.-rate = 0.61 ; Acc-rate-phi = 0.89
## iter. numb. 96050 ; Acc.-rate = 0.65 ; Acc-rate-phi = 0.88
## iter. numb. 97050 ; Acc.-rate = 0.64 ; Acc-rate-phi = 0.90
## iter. numb. 98050 ; Acc.-rate = 0.65 ; Acc-rate-phi = 0.86
## iter. numb. 99050 ; Acc.-rate = 0.67 ; Acc-rate-phi = 0.87
## iter. numb. 100050 ; Acc.-rate = 0.69 ; Acc-rate-phi = 0.85
## MCMC performed: n.iter. =  1e+05 ; thinning =  1000 ; burn.in =  50
## pois.krige.bayes: Prediction performed
```



```
## [1]  8.67046 68.03207
```




```
## [1]  2.924302 37.248705
```




```
##
#Problem 2
##

#load the data
```

```r
load("C:/Users/ckell/OneDrive/Penn State/2017-2018/597/spatial_statistics_597/Homework 2/data/dove.RData

#transform the original observations
dove$z <- log(dove$counts)

#use classical geostatistical techniques to get preliminary estimates of $\sigma^2$ and $\phi$
#I'm supposing that this means we are not using the generalized linear spatial modeling framework yet.

#Fitting the preliminary linear model
linmod <- lm(z~coords.x1+coords.x2, data=dove)
linmod <- lm(z~1, data=dove)
summary(linmod)
#storing the residuals so I can plot them
dove$resid <- linmod$resid
View(as.data.frame(dove))

##  Nonparametric estimation of the variogram
vg <- variogram(resid ~ 1, data = dove)#, width=75)
print(vg)
plot(vg, xlab = "Distance", ylab = "Nonparametric Semi-variogram estimate", width=5)

##  Fitting the variogram parametrically
##  range in this case is the number of datapoints, according to lecture code
nrow(dove)
##  nugget =1 includes a nugget term in the model
# fitvg <- fit.variogram(vg, vgm(1, "Exp", range=47, nugget=1), fit.method = 2)
# print(fitvg)
# fitvg.2 <- fit.variogram(vg, vgm(1, "Exp", 47, 0.05))
# print(fitvg.2)
fitvg.3 <- fit.variogram(vg, vgm("Exp"))
print(fitvg.3)
plot(vg, fitvg.3, xlab = "Distance", ylab = "Semi-variogram estimate", layout=c(2,1))

s2.hat <- fitvg.3$psill[2]
phi.hat <- fitvg.3$range[2]
tau2.hat <- fitvg.3$psill[1]
```

```r
##
#Problem 3
##

dove_mcmc <- as.geodata(dove)

#run an initial MCMC chain, fixing phi at your estimate from
#achieve an acceptance rate of about 60% for the process samples
sim.model <- model.glm.control(cov.model= "exponential")#, kappa = NA, not required for exponential
sim.prior <- prior.glm.control(phi.prior = "fixed", phi = phi.hat)
sim.mcmc <- mcmc.control(S.scale = 0.017, Htrunc = "default", S.start = "random",
                         burn.in = 0, thin = 10, n.iter = 100000)
sim.posterior <- pois.krige.bayes(dove_mcmc, model=sim.model, prior=sim.prior,
                                  mcmc.input=sim.mcmc)#, keep.mcmc.sim=TRUE)
sim.posterior$posterior$acc.rate[100,]

#ACF and trace plots
```

```
plot(sim.posterior$posterior$beta$sample, type = "l")
acf(sim.posterior$posterior$beta$sample)

plot(sim.posterior$posterior$sigmasq$sample, type = "l")
acf(sim.posterior$posterior$sigmasq$sample)
```

```
##
#Problem 4
##

#run an initial MCMC chain, fixing phi at your estimate from
sim.model <- model.glm.control(cov.model= "exponential")#, kappa = NA, not required for exponetnial
sim.prior <- prior.glm.control(phi.prior = "uniform", phi.discrete = seq(500,300000,500))
# Experiment with changing S.scale and phi.scale to get acceptance rates of about 60%
# and 25%, respectively.
# S.scale 0.007
sim.mcmc <- mcmc.control(S.scale = 0.01, phi.scale=0.12, Htrunc = "default",
                         S.start = "random",
                         burn.in = 0, thin = 100, n.iter = 100000)
sim.posterior <- pois.krige.bayes(dove_mcmc, model=sim.model, prior=sim.prior,
                                  mcmc.input=sim.mcmc)

nrow(sim.posterior$posterior$acc.rate)
sim.posterior$posterior$acc.rate[100,]

#acf and trace plots for beta
plot(sim.posterior$posterior$beta$sample, type = "l")
acf(sim.posterior$posterior$beta$sample)

#acf and trace plots for sigma^2
plot(sim.posterior$posterior$sigmasq$sample, type = "l")
acf(sim.posterior$posterior$sigmasq$sample)

#acf and trace plots for phi
plot(sim.posterior$posterior$phi$sample, type = "l")
acf(sim.posterior$posterior$phi$sample)
```

```
##
#Problem 5
##

#Run the same code as in part (4), increasing to thin = 1e4 and n.iter = 1e7.
#run an initial MCMC chain, fixing phi at your estimate from
sim.model <- model.glm.control(cov.model= "exponential")#, kappa = NA, not required for exponetnial
sim.prior <- prior.glm.control(phi.prior = "uniform", phi.discrete = seq(500,300000,500))
# Experiment with changing S.scale and phi.scale to get acceptance rates of about 60%
# and 25%, respectively.
# S.scale 0.007
sim.mcmc <- mcmc.control(S.scale = 0.01, phi.scale=0.12, Htrunc = "default",
                         S.start = "random",
                         burn.in = 0, thin = 1e3, n.iter = 1e5)
sim.posterior <- pois.krige.bayes(dove_mcmc, model=sim.model, prior=sim.prior,
                                  mcmc.input=sim.mcmc)
```

```
nrow(sim.posterior$posterior$acc.rate)
sim.posterior$posterior$acc.rate[100,]

#acf and trace plots for beta
plot(sim.posterior$posterior$beta$sample, type = "l")
acf(sim.posterior$posterior$beta$sample)

#acf and trace plots for sigma^2
plot(sim.posterior$posterior$sigmasq$sample, type = "l")
acf(sim.posterior$posterior$sigmasq$sample)

#acf and trace plots for phi
plot(sim.posterior$posterior$phi$sample, type = "l")
acf(sim.posterior$posterior$phi$sample)

#choose a burnin to discard and make them again
burnin <- 1:50
test <- sim.posterior$posterior$sigmasq$sample
post.burnin_sig <- sim.posterior$posterior$sigmasq$sample[-c(burnin)]
post.burnin_b <- sim.posterior$posterior$beta$sample[-c(burnin)]
post.burnin_phi <- sim.posterior$posterior$phi$sample[-c(burnin)]

#acf and trace plots for beta
plot(post.burnin_b, type = "l")
acf(post.burnin_b)

#acf and trace plots for sigma^2
plot(post.burnin_sig, type = "l")
acf(post.burnin_sig)

#acf and trace plots for phi
plot(post.burnin_phi, type = "l")
acf(post.burnin_phi)


#calculate the effective sample size for each parameter
##if they are not all at least 100 for each parameter, go back and run a longer chain
ess(post.burnin_sig)
ess(post.burnin_phi)
ess(post.burnin_b)

#plot the marginal and posterior distributions for each parameter using histograms and/or kernel densit
#par(mfrow = c(1,3))
hist(sim.posterior$posterior$beta$sample, main ="beta")
hist(sim.posterior$posterior$sigmasq$sample, main = "sigmasq")
hist(sim.posterior$posterior$phi$sample, main = "phi")
```

```
##
#Problem 6
##
#Do one final run of the chain, the same as in (5), but this time specifying the burn-in value you chos
sim.model <- model.glm.control(cov.model= "exponential")#, kappa = NA, not required for exponetnial
sim.prior <- prior.glm.control(phi.prior = "uniform", phi.discrete = seq(500,300000,500))
#S.scale = 0.5, 0.2, 0.7
```

```r
# Experiment with changing S.scale and phi.scale to get acceptance rates of about 60%
# and 25%, respectively.
sim.mcmc <- mcmc.control(S.scale = 0.007, phi.scale=100, Htrunc = "default",
                         S.start = "random",
                         burn.in = 50, thin = 1e3, n.iter =  1e5)
sim.posterior <- pois.krige.bayes(dove_mcmc, model=sim.model, prior=sim.prior,
                                  mcmc.input=sim.mcmc, locations=dove.grid)

post_med <- sim.posterior$predictive$median
uncertainty <- sim.posterior$predictive$uncertainty


ploteqc <- function(spobj, z, breaks, ...){
  pal <- tim.colors(length(breaks)-1)
  fb <- classIntervals(z, n = length(pal),
                       style = "fixed", fixedBreaks = breaks)
  col <- findColours(fb, pal)
  plot(spobj, col = col, ...)
  image.plot(legend.only = TRUE, zlim = range(breaks), col = pal)
}


plot(dove)

range(post_med)
breaks <- seq(8, 70, by = 0.01)
ploteqc(dove.grid, post_med, breaks, pch = 19)
#map("county", region = "missouri", add = TRUE)
title(main = "Posterior Mean")


range(uncertainty)
breaks <- seq(2, 35, by = 0.01)
ploteqc(dove.grid, uncertainty, breaks, pch = 19)
#map("county", region = "missouri", add = TRUE)
title(main = "Posterior Mean")
points(dove)
```

\end{document}