Final_Exam_Problem2

```
library(rstan)
```

```
## Loading required package: StanHeaders
```

```
## Loading required package: ggplot2
```

```
## rstan (Version 2.19.2, GitRev: 2e1f913d3ca3)
```

```
## For execution on a local, multicore CPU with excess RAM we recommend calling
## options(mc.cores = parallel::detectCores()).
## To avoid recompilation of unchanged Stan programs, we recommend calling
## rstan_options(auto_write = TRUE)
```

```
library(bayesplot)
```

```
## This is bayesplot version 1.7.0
```

```
## - Online documentation and vignettes at mc-stan.org/bayesplot
```

```
## - bayesplot theme set to bayesplot::theme_default()
```

```
##    * Does _not_ affect other ggplot2 plots
```

```
##    * See ?bayesplot_theme_set for details on theme setting
```

```
library(matrixStats)
source("~/Documents/BU_2019_Fall/bslm.R")

# Data
stroke <- within(read.csv("~/Documents/BU_2019_Fall/stroke.csv"), subject <- factor(subj
ect))
```

setup design matrices

```r
n = 8 #no of observations per subject
J = 24 #no of subjects
p = 2 #no of predictors b0 (intercept), b1 (slope)
Y <- matrix(0,nrow = n,ncol = J)
X <- matrix(0,nrow = n, ncol = p*J)
subjects_unique <- unique(stroke$subject)
col_n = 1
for (i in seq(1,24,by=1)){
  Y[ ,i] <- stroke$score[stroke$subject==subjects_unique[i]] # size: n*J
  X[ ,col_n] <- rep(c(1),n) #intercept
  X[ ,col_n+1] <- c(1,2,3,4,5,6,7,8) #slope
  col_n = col_n + 2
}
```

Initialize Parameters

```r
# Using Priors
mod <- lm(Y[ ,1] ~ X[ ,1:2] - 1)
beta0 <- mod$coefficients
sigma0 <- diag(530,nrow = 2,ncol = 2)
nu <- 2
tau2 <- 4

beta <- rnorm(2,mean=beta0,sqrt(sigma0))
sigma2 <- 10
rho <- 0.8
```

```r
R_rho <- function(rho){
  In = diag(1,n,n)
  ((1-rho)*In) + (rho * matrix(1,n,n))
}

R_rho_inverse <- function(rho){
  In <- diag(1,n,n)
  (1/(1-rho)) * (In - (rho/(1+(n-1)*rho)) * matrix(c(1),n,n))
}

R_rho_determinant <- function(rho){
  In <- diag(1,n,n)
  (1-rho)^(n-1) * (1+((n-1)*rho))
}

Jeffrey_Prior_rho <- function(rho){
  (((n-1)/(1-(1/n * (1+(n-1)*rho)))^2) + (1/(1/n*(1+(n-1)*rho))^2))^(1/2)
}

# Set up grid for Numerical Integration Sampling of Rho
rho_grid <- seq((-1/(n-1))+0.001, 1-0.001, by = 0.01)
```

Conditional Posteriors

```
# Samples Sigma2 from its Conditional Posterior distribution
CondPost_sigma2 <- function(beta,rho, sigma2){
  df_sigma2 = (n*J+nu) # Degrees of Freedom
  scale_sigma2 <- 0
  col_n <- 1
  for (j in 1:J) {
    scale_sigma2 <- scale_sigma2 + t((Y[ ,j] - X[ , col_n:( col_n+p-1)] %*% beta)) %*% R
_rho_inverse(rho) %*% (Y[ ,j] - X[ ,col_n:( col_n+p-1)] %*% beta)
  col_n <- col_n + 2
  }
  scale_sigma2 <- (scale_sigma2 + nu*tau2)/(n*J + nu) # Scale
  v <- (df_sigma2 *  scale_sigma2)/rchisq(1,df_sigma2) # Scaled Inv-Chisquared
 return(as.vector(v))
 # return(300)
}

# Samples Beta from its Conditional Posterior distribution
  # Beta_Star: From Centering Trick
beta_star <- function(X,Y, rho){
  solve(t(X) %*% R_rho_inverse(rho) %*% X) %*% (t(X) %*% R_rho_inverse(rho) %*% Y)
}

CondPost_beta <- function(beta,sigma2,rho){
  V_inv <- solve(sigma0)
  col_n <- 1
  for (j in 1:J){
    V_inv <- V_inv + (as.vector(1/sigma2) * t(X[ ,col_n:( col_n+p-1)]) %*% R_rho_inverse
(rho) %*% X[ ,col_n:( col_n+p-1)])
    col_n <- col_n + 2
  } # Variance
  mu <- solve(sigma0) %*% beta0
  col_n <- 1
  for (j in 1:J){
    mu <- mu + (as.vector(1/sigma2) * t(X[ ,col_n:( col_n+p-1)]) %*% R_rho_inverse(rho)
 %*% X[ ,col_n:( col_n+p-1)])%*% beta_star(X[ ,col_n:( col_n+p-1)],Y[ ,j],rho)
    col_n <- col_n + 2
  }
  mu <- solve(V_inv) %*% mu # Mean
  beta <- rnorm(2,mean = mu, sd = chol(solve(V_inv)))
  return(beta)
}

# Rho: Computes Log Conditional Posterior distribution Values over the grid values
logCondPost_rho <- function(sigma2, beta, rho){
  rho_Post <- log(R_rho_determinant(rho)^(-J/2))
  col_n <- 1
  for (j in 1:J) {
    rho_Post <- rho_Post - (as.vector(1/(2*sigma2))*t((Y[ ,j] - X[ ,col_n:( col_n+p-1)]
 %*% beta)) %*% R_rho_inverse(rho) %*% (Y[ ,j] - X[ ,col_n:( col_n+p-1)] %*% beta))
    col_n <- col_n + 2
  }
  rho_Post <- rho_Post + log(Jeffrey_Prior_rho(rho))
  return(rho_Post)
}

# Samples Rho using Numerical Integration based on a grid
rho_sample <- function(sigma2, beta){
   rho_logPost = rep(0,1,length(rho_grid))
    for (i in 1:length(rho_grid)){
      rho_logPost[i] <- logCondPost_rho(sigma2, beta, rho_grid[i])
    }
    # Normalize Rho Posterior distribution values over the grid
    rho_log_normalize <- rho_logPost - rep(logSumExp(rho_logPost),1,length(rho_grid))
    # Sample from Multinomial
    rho_s <- rmultinom(1,size = 1,prob = exp(rho_log_normalize))
    index_rho_s <- match(1,rho_s)
    rho <- rho_grid[index_rho_s]
    return(rho)
}
```

- b. Gibbs Sampling

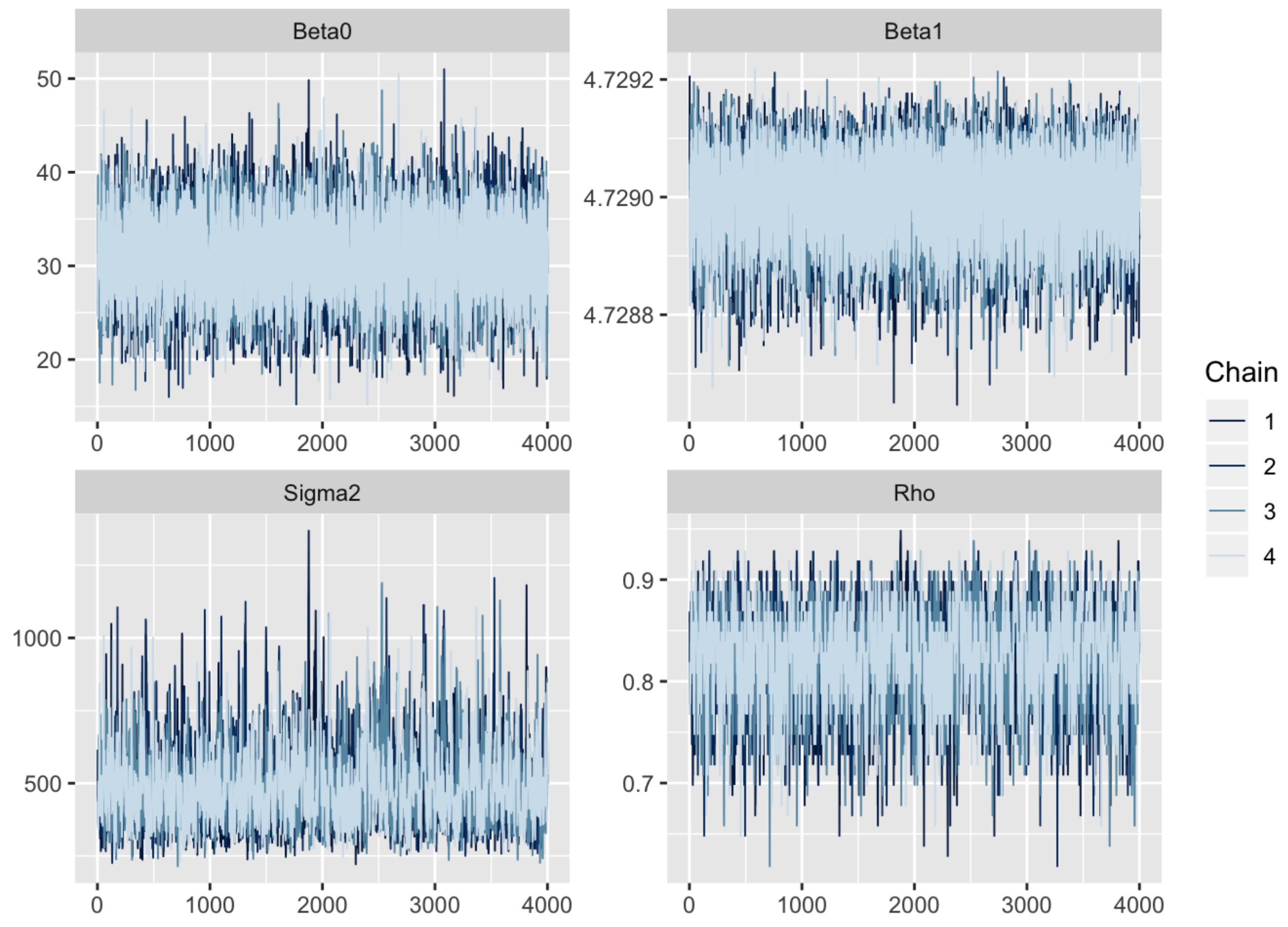
```
nchains = 4
nsims = 5000
warmup = 1000

gibbs_chain <- matrix(0,ncol = 4,nrow = nsims - warmup)
param_names <-  c("Beta0","Beta1", "Sigma2","Rho")
colnames(gibbs_chain)<- param_names
nparams <- length(param_names)
sims <- mcmc_array(nsims - warmup, nchains, param_names)
for (chain in 1:nchains) {

  for (it in 1:nsims) {
    #print(it)
    # 1. Sample from posterior on beta
    beta <- CondPost_beta(beta,sigma2,rho)
    # 2. Sample from sigma2
    sigma2 <- CondPost_sigma2(beta,rho,sigma2)
    # 3. Sample from Rho
    rho <- rho_sample(sigma2, beta)

    if(it > warmup){
      sims[it - warmup, chain, ] <- c(beta,sigma2,rho)
    }
  }

}
```

Plot Convergence Statistics

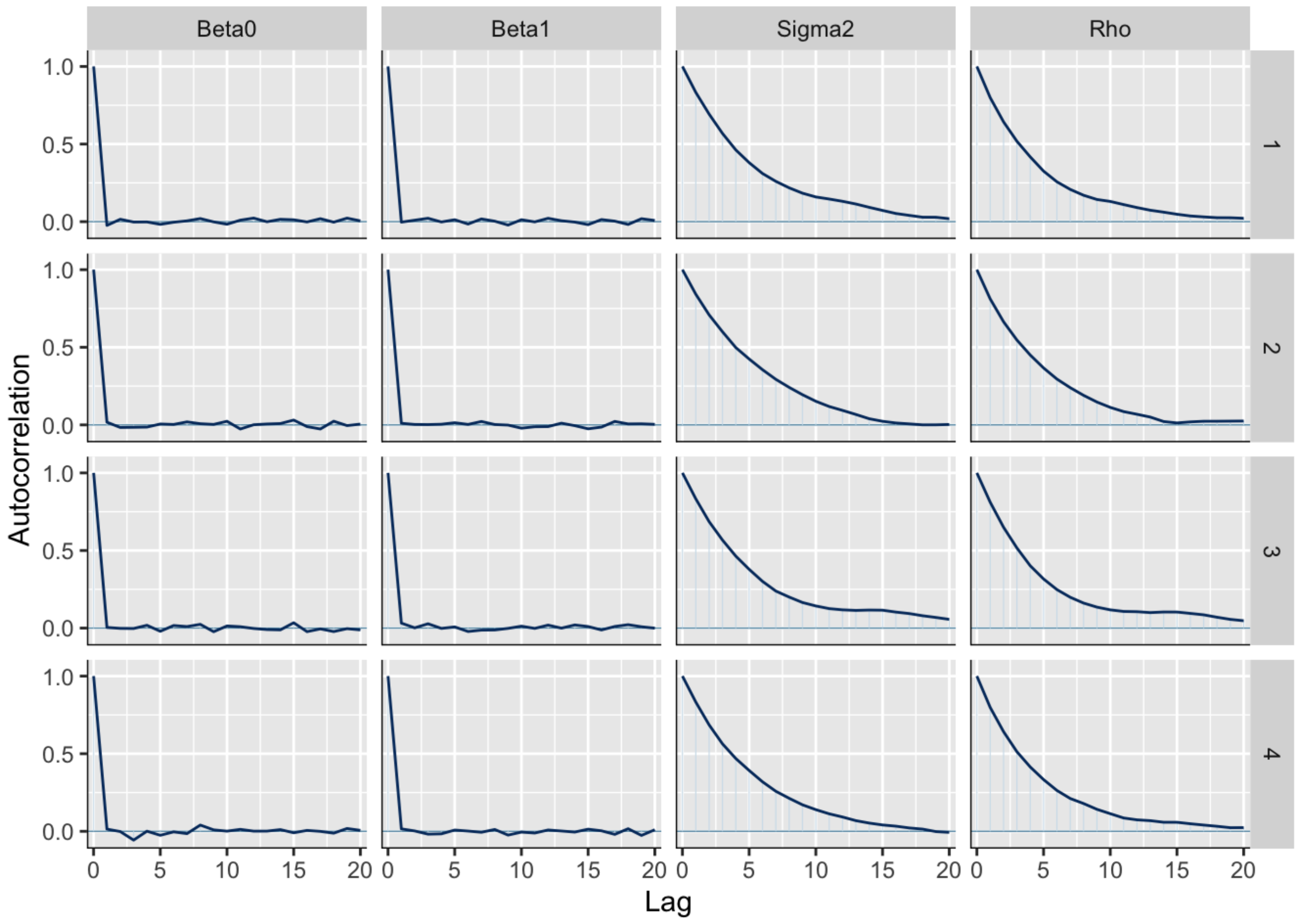
```
monitor(sims)
```

```
## Inference for the input samples (4 chains: each with iter = 4000; warmup = 2000):
##
##          Q5   Q50   Q95  Mean    SD  Rhat Bulk_ESS Tail_ESS
## Beta0   24.0  31.1  38.1  31.1   4.3     1     8061     6879
## Beta1    4.7   4.7   4.7   4.7   0.0     1     7235     7615
## Sigma2 324.8 467.9 737.0 491.9 129.9     1      704     1374
## Rho      0.7   0.8   0.9   0.8   0.0     1      694     1685
##
## For each parameter, Bulk_ESS and Tail_ESS are crude measures of
## effective sample size for bulk and tail quantities respectively (an ESS > 100
## per chain is considered good), and Rhat is the potential scale reduction
## factor on rank normalized split chains (at convergence, Rhat <= 1.05).
```

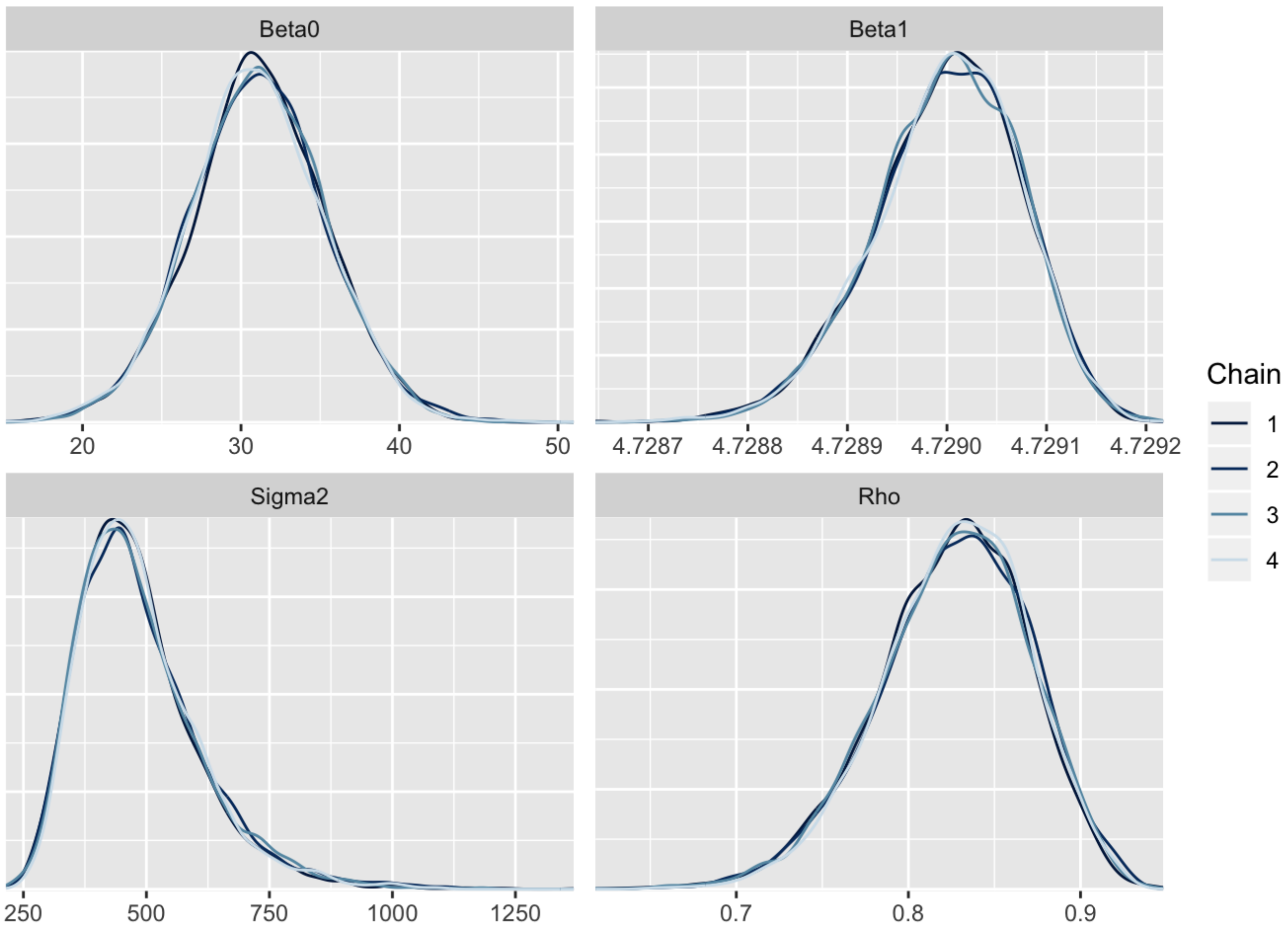
```
mcmc_trace(sims)
```



```
mcmc_acf(sims)
```



```
mcmc_dens_overlay(sims)
```



```
t(apply(sims[,2,],2, function(x) quantile(x, c(.025,.25,.5,.75,.975))))
```

```
##
## parameters       2.5%        25%        50%        75%      97.5%
##      Beta0 22.6200232 28.2045611 31.0920807 33.8532842 39.6272441
##      Beta1  4.7288409  4.7289540  4.7290053  4.7290534  4.7291320
##     Sigma2 301.4328279 396.6756925 464.8822819 556.8924935 828.9156644
##        Rho  0.7281429  0.7981429  0.8281429  0.8581429  0.9081429
```

Convergence Statistics: Convergence of the parameters was assessed using Trace plots, Autocorrelation plots, Density overlays and Rhat, ESS for 4 chains. All chains for all parameters appear to be mixing well in the Trace plots. Autocorrelation plots show that beta0 and sigma2 have very low autocorrelation, however beta1 and rho appear to be dependent on the past showing high autocorrelation with non-zero lags. Density overlays show similar estimated densities for all parameters for the 4 chains. Rhat and ESS are under the reasonable limits, suggesting reasonable convergence of the parameters.

- b. Comparison with Mixed Model

```
library(lme4)
```

```
## Loading required package: Matrix
```

```
# E[y_{ijk}] = (alpha + beta_j + gamma * w_k + theta_j * w_k)
#              + (phi_i + psi_i * w_k)
re <- lmer(score ~ week + (1 | subject), data = stroke) # equi-correlation structure
summary(re)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: score ~ week + (1 | subject)
##    Data: stroke
##
## REML criterion at convergence: 1468.3
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -2.1794 -0.6118  0.0208  0.6453  3.0440
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  subject  (Intercept) 392.71   19.817
##  Residual              79.67    8.926
## Number of obs: 192, groups:  subject, 24
##
## Fixed effects:
##             Estimate Std. Error t value
## (Intercept)  31.0342     4.2870   7.239
## week          4.7297     0.2811  16.824
##
## Correlation of Fixed Effects:
##      (Intr)
## week -0.295
```

As can be seen from the Linear mixed model summary above, Fixed effects intercept and slope (effect of week on score) estimates are very close to the Beta values observed using Gibbs Sampler. Additionally, Random effects Variance estimate from linear mixed model reasonably matches the sigma2 estimated using Gibbs Sampler.

- c. Posterior Predictive Checks and Uncorrelated Outlier analysis

```
library(expm)
```

```
##
## Attaching package: 'expm'
```

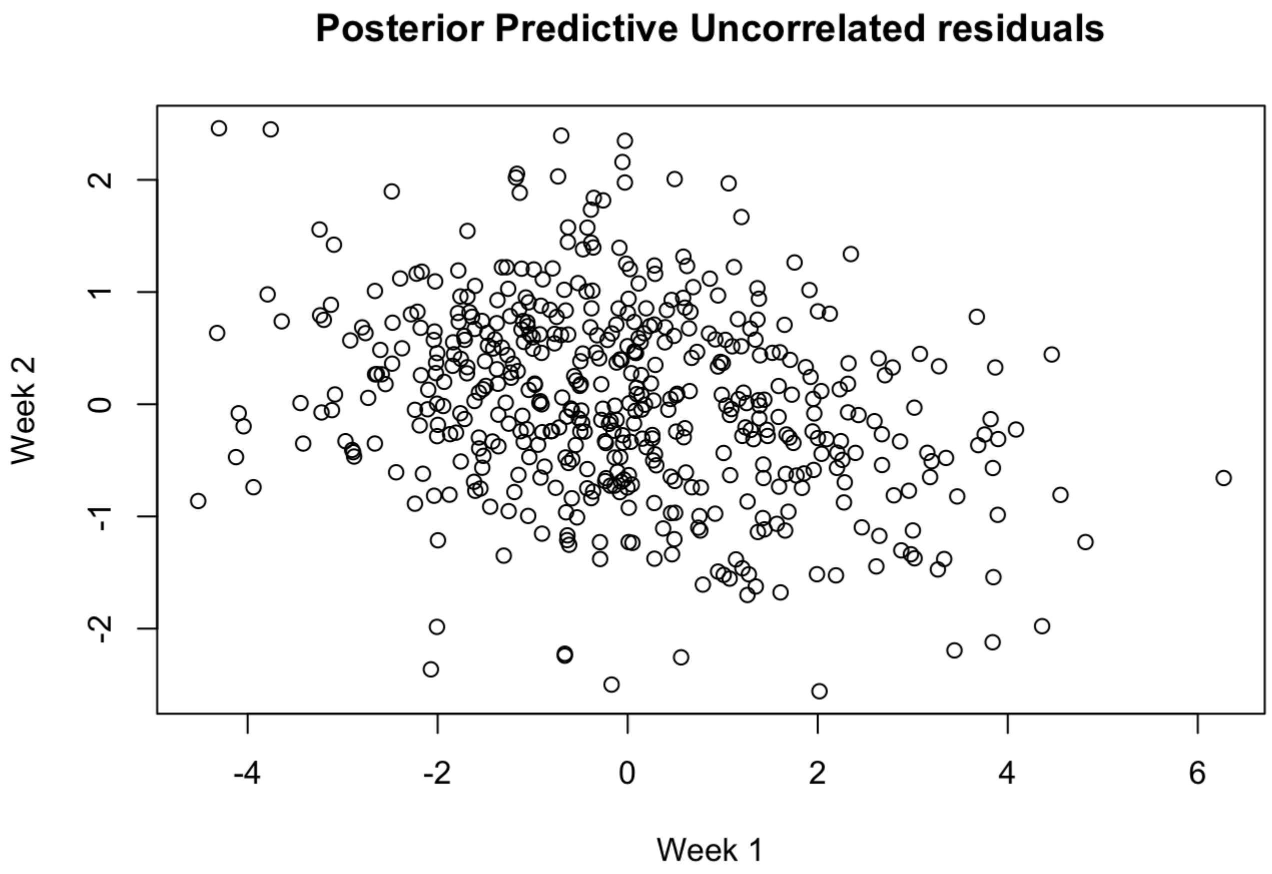
```
## The following object is masked from 'package:Matrix':
##
##     expm
```

```
Post_pred <- function(beta_pred, sigma2_pred, rho_pred){
   mu <- X[ ,1:2] %*% beta_pred
   std <- sqrt(sigma2_pred) * sqrtm(R_rho(rho_pred))
  y_rep_s <- rnorm(8,mean = mu, sd = std)
  return(y_rep_s)
}
```

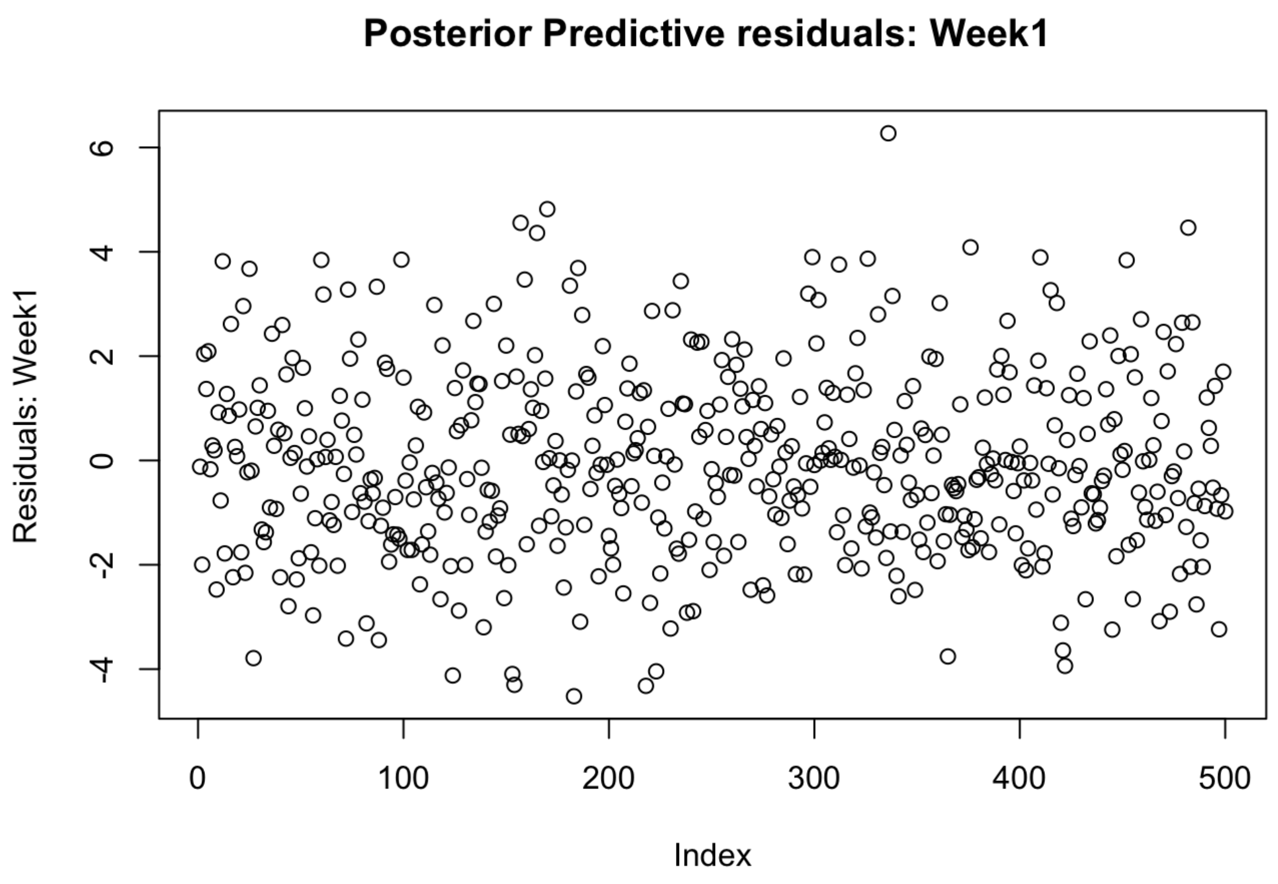
Computing Posterior Predictive and Original Data Residuals

```
# Posterior Predictive Residuals
residuals_rep <- matrix(NA,8,500)
for (i in 1:500) {
  #print(it)
    # 1. Sample from posterior on beta
    beta_s <- CondPost_beta(beta,sigma2,rho)
    # 2. Sample from sigma2
    sigma2_s <- CondPost_sigma2(beta,rho,sigma2)
    # 3. Sample from Rho
    rho_sm <- rho_sample(sigma2, beta)
    # Posterior Predictive distribution Samples
    y_rep <- Post_pred(beta_s, sigma2_s, rho_sm)
    # Compute normalized residuals
    residuals_rep[,i] <-sqrt(1/sigma2_s) * sqrtm(R_rho_inverse(rho_sm)) %*% (y_rep - X[
 ,1:2] %*% beta_s)
}
# Original Residuals
residuals_orig <- matrix(NA,8,J)
for (i in 1:J){
  residuals_orig[,i] <-sqrt(1/sigma2) * sqrtm(R_rho_inverse(rho)) %*% (Y[,i] - X[ ,1:2]
 %*% beta)
}
```

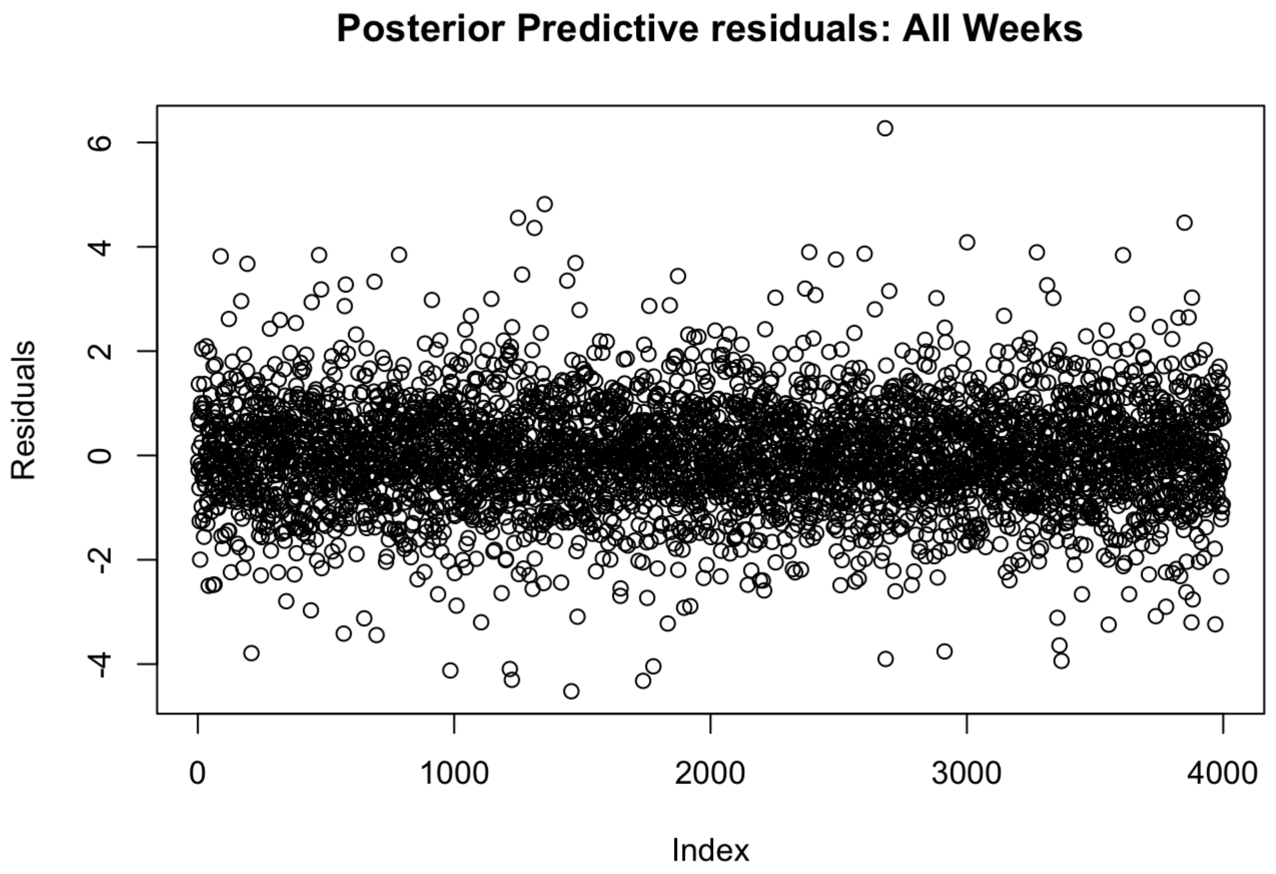
```
plot(residuals_rep[1,],residuals_rep[2,], xlab = "Week 1", ylab = "Week 2", main = "Post
erior Predictive Uncorrelated residuals")
```



```
plot(residuals_rep[1,], main = "Posterior Predictive residuals: Week1 ", ylab = "Residua
ls: Week1")
```



```
#plot(residuals_n[,1], main = "Posterior Predictive Week1 residuals")

plot(as.vector(residuals_rep), main = "Posterior Predictive residuals: All Weeks", ylab
 = "Residuals")
```



As can be seen from the residual plots above, residuals across the weeks are uncorrelated.

Proportion of Outliers : Posterior Predictive

```
# Identifying a residual as an outlier if it is beyond 3*standard deviation of the norma
lized residuals
  outlier_bound <- 3*sd(as.vector(residuals_rep))
  res <- as.vector(residuals_rep)
  outlier <- res[res > outlier_bound | res < -outlier_bound]
  print(length(outlier)/length(res))
```

```
## [1] 0.012
```

```
  # Original Y
  outlier_bound_orig <- 3*sd(as.vector(residuals_orig))
  res_orig <- as.vector(residuals_orig)
  outlier_orig <- res_orig[res_orig > outlier_bound_orig | res_orig < -outlier_bound_ori
g]
  print(length(outlier_orig)/length(res_orig))
```

```
## [1] 0.005208333
```

Above values are the proportion of Residual Outliers for the posterior predicted Yrep and original Y repectively. Proportion of outliers seems to reasonably match, suggesting that our model fits the data well.

- d. Comparing Linear Model with (H1) and without (H0) interaction term (group:week)

```
modH0 <- lm(score ~ week + group, data = stroke)
summary(modH0)
```

```
##
## Call:
## lm(formula = score ~ week + group, data = stroke)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -49.210 -13.942  -4.618  15.118  58.429
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  36.8415     3.9663   9.289  < 2e-16 ***
## week          4.7297     0.6611   7.155 1.82e-11 ***
## groupB       -5.4688     3.7102  -1.474   0.1422
## groupC      -11.9531     3.7102  -3.222   0.0015 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 20.99 on 188 degrees of freedom
## Multiple R-squared:  0.2468, Adjusted R-squared:  0.2348
## F-statistic: 20.53 on 3 and 188 DF,  p-value: 1.494e-11
```

```
modH1 <- lm(score ~ week + group + group:week, data = stroke)
summary(modH1)
```

```
##
## Call:
## lm(formula = score ~ week + group + group:week, data = stroke)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -47.812 -13.560  -5.253  13.289  63.646
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  30.1339     5.7706   5.222 4.71e-07 ***
## week          6.2202     1.1427   5.443 1.64e-07 ***
## groupB        3.0357     8.1608   0.372    0.710
## groupC       -0.3348     8.1608  -0.041    0.967
## week:groupB  -1.8899     1.6161  -1.169    0.244
## week:groupC  -2.5818     1.6161  -1.598    0.112
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 20.95 on 186 degrees of freedom
## Multiple R-squared:  0.2577, Adjusted R-squared:  0.2377
## F-statistic: 12.91 on 5 and 186 DF,  p-value: 8.677e-11
```

Model H1 shows non-significant co-efficients for the interaction terms, suggesting those terms should not be included in the model since group:week does not have significant effect on scores. Therefore H0 is a better model than H1.