Bayesian Hierarchical Linear Model with Complex Variance Structures

Tianyuan Zhou

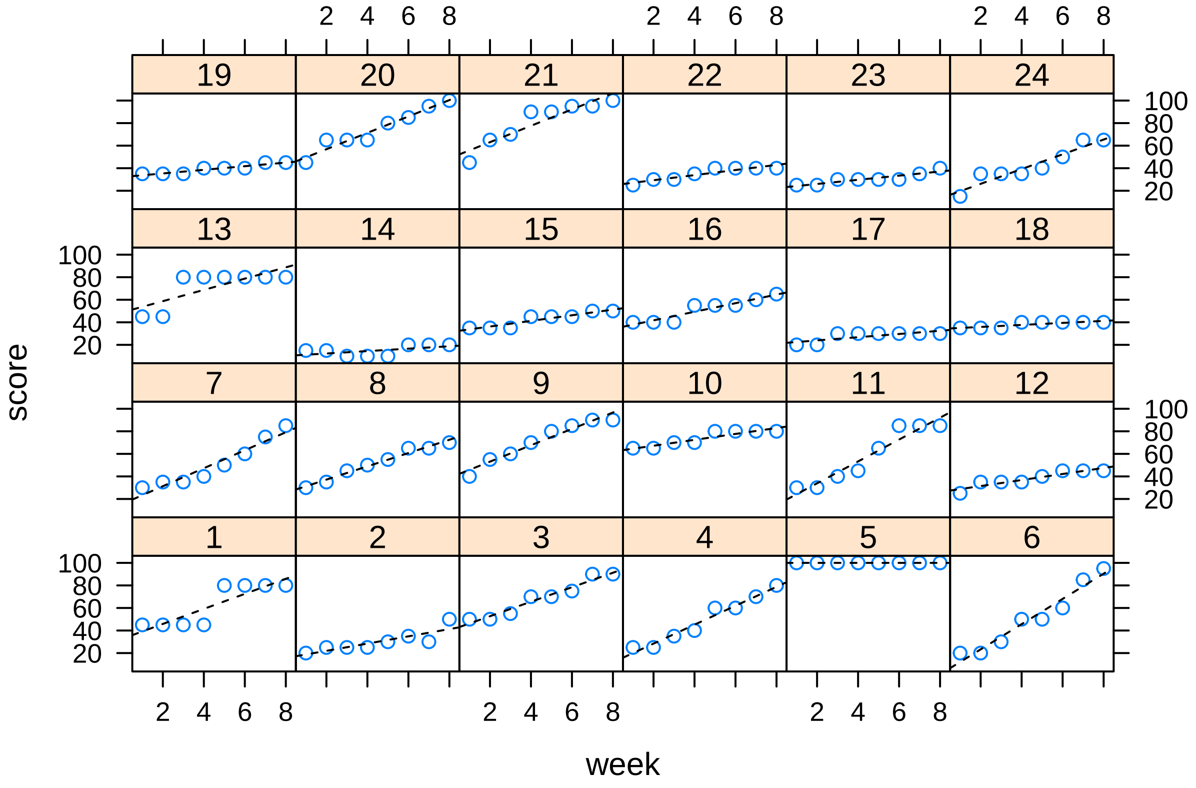
December 6, 2015

In many real world applications, it is often advisable to use a hierarchical structure when we try to model data because different predictors might be affect response differently, and each predictor could have different levels of variation. In Gelman, Carlin, and Stern's book, they gave a school example to illustrate the specific data structure in which a hierarchical linear model would be more appropriate. In their example, the standardized score of students in each school district could be affected by variations in each students, but could also be affected by variations among each school it selves. Since many students attend same school, it is better to model the data under a hierarchical structure that taken account into information regarding both variations between different students and variations between different schools. In this project, we would also discuss the applications of hierarchical linear models in statistical modeling.

For our project, we will take a look at the data "stroke." This data record 24 subjects, each suffered a stroke, and their recoveries in each week under different treatments. The 24 subjects are divided into 3 groups: group A patients are using the new treatment, group B patients are using the usual treatment, and group C are using the usual treatment but undergoing the treatment in different hospital. In each week, the patients are giving a score after taking a test regarding how well they can perform the ordinary tasks, with 100 being the highest (meaning they can perform all the tasks extremely well) and 0 being the lowest (meaning they can't perform tasks at all). They measure these scores for 8 weeks and record them, forming our data. Our main point of interest is to see whether the new treatment is more effective at helping stroke patients to recover compare to the old treatment.

Before we begin our fitting and analysis, it is good to take a look on how patients recover after . We can plot the trend for each individual subject during these 8 weeks using package *lattice*, and the plot is shown below.

```
## Loading required package: Rcpp
## rstan (Version 2.8.0, packaged: 2015-09-19 14:48:38 UTC, GitRev: 05c3d0058b6a)
## For execution on a local, multicore CPU with excess RAM we recommend calling
## rstan_options(auto_write = TRUE)
## options(mc.cores = parallel::detectCores())
```



In the data set, subjects 1 through 8 belong to group A, subjects 9 through 16 belong to group B, and subjects 17 through 24 belong to group C. It is then quickly apparent from the picture that each subjects response to the treatment differently even if they are in the same group. For instance, subject 9 and subject 14 both belong to group B. However, subject 9 recovers really well, going from a score of 40 to a score of 90 in 8 weeks; subject 14, on the other hand, does not seem to recover from the stroke at all, going from a score 15 to a score of 20, including two weeks (week 3 and 4) that his performance score actually dropped. Similar observations can be made in group A: subject 3, 4, 7, and 8 all seem to have a roughly linear recovery rate, while subject 1 seems to make a jump in performance score in week 5 (going from 45 to 80) but otherwise stagnant. Then there is subject 5, who has a score of 100 through out 8 weeks which would led us question if he is misdiagnosed from the start. There are plenty reasonable explanations for this results. One such explanation could be that a patient who has a low starting score, such as subject 14, probably suffered a more serious stroke than subject with a high starting score, such as subject 10, which then would require more time to recover and thus would response to treatment more slowly. This suggests that just regress score on each group could be misleading and we should taken account into the variations between each patients in the group, and with this motivation we began to explore the hierarchical linear model we mentioned earlier.

In the hierarchical setup, our lower hierarchy is the performance score (this is our y) on each subject and each week (these are our βs). Then, at the higher hierarchy, we regress β on week, each group, and the increases for each group per week (we denote these as X_β). Since we are trying to build a linear model, we will assume that our underlying distributions are normal, that is, $y|\beta, \sigma^2 \sim N(X\beta, \sigma^2 I_n)$. Then, for each β_j, it would also be normally distributed. During the class, we assume that the variance structure for β would be a scalar, denoted by τ^2, which follows an scaled-inverse-chi-square distribution. multiplied by an identity matrix I_{P_J}. However, in our problem, the assumption of independence between different covariates broke down at the β level: Group A and Group B are performed in the same hospital, and Group B and Group C uses the same treatment, so intuitively there should be some dependencies between different groups. Therefore, for the purpose of this project, we wish to explore a more general covariance structure on the group-level coefficients. Under such setup, instead a random scalar (as in the case of τ^2), we have a random matrix, which change the conjugate distribution from a scaled-inverse-chi-square to an inverse-Wishart distribution, or $\beta_j|\alpha, \Sigma_\beta \sim N((X_\beta)_j\alpha, \Sigma_\beta)$, for $j = 1, 2, 3, ..., J$, where $\Sigma_\beta \sim Inv-Wishart(\nu_\beta, \Lambda_\beta^{-1})$, $\Lambda_\beta = \xi_\beta^{-1}I_p$. ξ_β is a scale here. Then, for the whole β matrix, it follows a normal distribution with $X_\beta\alpha$ as its mean and $\Sigma_\beta \otimes I_J$ as its variance. Finally, we also need to define a prior for α and σ^2, for these two we will just use the principal of conjugacy and define $\alpha \sim N(\alpha_0, \Sigma_\alpha)$, and $\sigma^2 \sim inv-\chi^2(\nu_0, \xi_0)$.

Because all of our priors are defined through the principal of conjugacy, our posterior will be from the same family as our prior distribution. However, because the computation of normalization constant of the posterior is very difficult, if not impossible in most of these cases, we need to use MCMC methods to sample from the conditional posterior of each parameter to approximate the posterior distribution. In this question specifically, we would devise a Gibbs sampler to sample β, α, σ^2, and Σ_β each iteration and constantly update them until convergences. In order to accomplish that, we need conditional posterior for all of the parameters, with some calculations (shown in appendix), it can be shown that the conditional posterior for each variable, respectively, is

$$\beta|y, \sigma^2, \alpha, \Sigma_\beta \sim N(\tilde{\Sigma_\beta}(\frac{1}{\sigma^2}X^TY + (\Sigma_\beta \otimes I_j)^{-1}X_\beta\alpha), \tilde{\Sigma_\beta} = \frac{1}{\sigma^2}X^TX + (\Sigma_\beta \otimes I_J)^{-1})$$

$$\alpha|y, \sigma^2, \beta, \Sigma_\beta \sim N(\tilde{\Sigma}_\alpha(X_\beta^T(\Sigma_\beta \otimes I_j)^{-1}\beta + \Sigma_\alpha^{-1}\alpha_0), \tilde{\Sigma}_\alpha = X_\beta^T(\Sigma_\beta \otimes I_j)^{-1}X_\beta + \Sigma_\alpha^{-1})$$

$$\sigma^2|y, \beta, \alpha, \Sigma_\beta \sim Inv-\chi^2(n+\nu_0, \frac{\nu_0\xi_0 + (Y - X\beta)^T(Y - X\beta)}{n+\nu_0})$$

$$\Sigma_\beta \sim Inv-W(\nu_\beta + J, \sum_j(\beta_j - X_{\beta_j}\alpha)(\beta_j - X_{\beta_j}\alpha)^T + \xi_\beta I_p)$$

Having devised all the conditional posterior, we could now sample from it. However, R does not have a build in scaled-inverse-χ^2 or inverse-Wishart sampler, therefore we need to write our own based on *rgamma* and *rWishart* function, the random sampler is shown below

```r
rinvchisq <- function (n, nu, nu.tau2) 1 / rgamma(n, nu / 2, nu.tau2 / 2)
rinvWishart <- function(n, df, S) {
  sample <- rWishart(n, df, chol2inv(chol(S)))
  dimension <- dim(sample)
  sample.inv <- array(0, dim = c(dimension[1], dimension[2], dimension[3]))
  for (i in 1:n) {
    sample.inv[,,i] <- chol2inv(chol(sample[,,i]))
  }
  return(drop(sample.inv))
}
```

These two function take care of sampling for α and Σ_β. To sample for β and α, we can use the *bslm.sample1* function provided by Professor Carvalho. However, in the *bslm.sample1* function, by default the parameter prior.disp take in a list of scalar value, but in our case, one of the parameter in prior.disp of α is $\Sigma_\beta \otimes I_J$, a matrix, which would break down the code. Therefore, we need to do some manipulation so that the parameter passed in for alpha.prior becomes a scalar. We accomplish this by letting the precision $\Sigma_\beta \otimes I_J$ be C^TC, and instead of regressing β on X_β, we will be regressing $C^{-T}\beta$ on $C^{-T}X_\beta$, which will make its variance (and its precision) becomes an identity matrix I_p, thus solves our issue. We then use 4 independent chains, each with 1000 samples, with 500 samples in each chain as warm-up to form our sampler. (Which means we have 2000 total samples) Because we don't have any prior information regarding the recovery speed of stroke patient or the effectiveness of treatment, we'll set our prior to be non-informative. That is, we'll set the coefficient of prior parameter for α, σ^2, and Σ_β to be zeros (for Σ_β it means its "scale" is a zero matrix). We acquired our starting point for the Gibbs sampling procedure by running a frequentist *lm* command, first regress Y on X, then regress X on $X_\beta\alpha$, we take the coefficients from these frequentist linear models to be the starting values for α and β, and the variance for y as the starting point for σ^2. We don't need a starting point for Σ_β as its conditional posterior parameters involves α and β and we can start sampling using their initial values. The sampler code and its output is shown below:

```
#Define priors
alpha.prior <- list(mean=rep(0, p.alpha), precision=rep(0, p.alpha)) # 'fixed'
sigma2.prior <- list(df=0, scale=0)
sigmaBeta.prior <- list(df=0, S = matrix(c(0, 0, 0, 0), 2, 2))


# [ Hierarchical model with sigmaBeta]
###################################################################
nsamples <- 1000
nchains <- 4
sims <- mcmc.init.array(nsamples, nchains,
                        c(colnames(x.beta), colnames(x), 'sigma2', 'sigmaInt', 'sigmaWeek', 'corr'))

bl <- lm(y ~ x - 1)
beta <- coef(bl)                 #starting value for beta
sigma2 <- deviance(bl) / n       #starting value for sigma2
al <- lm(beta ~ x.beta - 1)
alpha <- coef(al)                #starting value for alpha
for (chain in 1:nchains) {
  for (iter in 1:nsamples) {
    #sample sigmaBeta
    sum <- matrix(0, 2, 2)
    for(i in 1:24) {
      sum <- sum + (rbind(beta[i], beta[i+24])-rbind(x.beta[i,], x.beta[24+i,])%*%alpha)%*%t(rbind(beta
    }
    sigmaBeta <- rinvWishart(1, sigmaBeta.prior$df + nsubject, sum+sigmaBeta.prior$S)

    #sample alpha
    C <- chol((kronecker(sigmaBeta, diag(nsubject))))
    CinvTBeta <- backsolve(C, beta, transpose = TRUE)
    cinvTxBeta <- backsolve(C, x.beta, transpose = TRUE)
    alpha <- bslm.sample1(CinvTBeta, cinvTxBeta,
                          prior.disp=list(1e6, 1), dispersion=1,
                          prior.coef = alpha.prior)
    #sample beta
    beta <- bslm.sample1(y, x,
                 prior.disp=sigma2.prior, dispersion=sigma2,
                 prior.coef=list(mean=x.beta %*% alpha,
                 precision=chol2inv(crossprod(C))))

    #sample sigma^2
    sigma2 <- rinvchisq(1, sigma2.prior$df + n,
                        sigma2.prior$df * sigma2.prior$scale +
                          crossprod(y - x %*% beta))

    sims[iter, chain, ] <- c(alpha, beta, sigma2, sqrt(sigmaBeta[1, 1]), sqrt(sigmaBeta[2, 2]), sigmaBe
  }
}
monitor(sims[,,c(1:p.alpha, p.alpha + p.beta + 1:4)], digits=2)
```

```
## Inference for the input samples (4 chains: each with iter=1000; warmup=500):
##
##                        mean se_mean    sd   2.5%    25%    50%    75% 97.5%
## coefintercept         29.92    0.19  8.56  13.35 24.50 29.96 35.37 46.91
## coefweek               6.19    0.03  1.23   3.83  5.39  6.19  6.98  8.68
## coefintercept:groupB  3.06    0.27 12.11 -20.32 -4.91  3.16 11.15 26.88
## coefweek:groupB      -1.85    0.04  1.72  -5.32 -2.94 -1.85 -0.77  1.64
## coefintercept:groupC -0.16    0.27 11.87 -23.97 -8.03 -0.31  7.49 23.09
## coefweek:groupC      -2.58    0.04  1.74  -6.16 -3.72 -2.55 -1.43  0.88
## sigma2               27.71    0.09  3.26  21.95 25.37 27.43 29.84 34.84
## sigmaInt             23.49    0.10  4.04  17.07 20.75 22.97 25.53 32.70
## sigmaWeek             3.34    0.02  0.60   2.41  2.92  3.27  3.67  4.73
## corr                 -0.42    0.01  0.19  -0.73 -0.56 -0.44 -0.29  0.01
##                      n_eff Rhat
## coefintercept         2000    1
## coefweek              1977    1
## coefintercept:groupB  2000    1
## coefweek:groupB       2000    1
## coefintercept:groupC  2000    1
## coefweek:groupC       2000    1
## sigma2                1211    1
## sigmaInt              1609    1
## sigmaWeek             1509    1
## corr                  1432    1
##
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

From our sample (the numbers could change due to re-running the code to perform random sampling, the summary below is the number for one particular sample I ran.) The coefficient of intercept is 30.23, which means the mean score on week 1 of patients in Group A is 30.23. coefweek is 6.23, which means the average increase in scores for patients in Group A is 6.23. The coefintercept:groupB and coefintercept:groupC is 2.92 and -0.27, respectively, which signifies that the scores of patients in group B on week one is, on average, 2.92 points higher than that of patients in group A, while the scores of patients in group C, on average, is .27 points lower than that of patients in group A. Finally, the coefweek:groupB and coefweek:groupC is -1.91 and -2.61, which signifies that patients in group B and group C's increase in scores per week, on average, is 1.91 points and 2.61 points less than that of patients in the group A. The monitor also gave the 95% posterior interval, thus there is .95 probability that the score for group A patients on week one is between 13.62 and 47.17, weekly increase in scores for all patients is between 3.69 and 8.55, difference in scores between group A patients and group B patiens on week one is between -20.87 and 26.43, difference in scores between group A patients and group B patiens on week one is between -23.73 and 24.58, difference of mean increase in scores per week between group A patients and group B patients is between -5.31 to 1.62, and difference of mean increase in scores per week between group A patients and group C patients is between -5.97 to 0.89. Judging by the mean of sample, it seems that this model indicates that the new treatment for stroke is more effective than the old treatment for stroke since both group B and Group C's patients (which use the old treatments in different hospitals) have less average increase in scores per week. We can also calculate the variances of slopes and intercepts (they are the diagonal entries of Σ_β matrix), as well as correlation between the βs as $\rho = \frac{\rho_{X,Y}}{\sigma_X \sigma_Y}$. I store the standard deviation instead of variance for the convenience of comparing models later, the standard deviation of intercept has .95 probability to be in between 17.04 and 32.36 with a mean of 23.29, while the standard deviation of intercept has .95 probability to be in between 2.37 to 4.76 with mean of 3.37. This output is reasonable as intuitively it makes sense that different individuals are highly variable while the weekly increases, given a subject, is probably the same. (We can refer back to the xyplot that most patients shows a linear improvement in scores) Thus the variations between individual is probably higher than the variations between the weekly increases in scores. The estimated correlation has a .95 probability -.73 and -.02 with a mean around -.40.

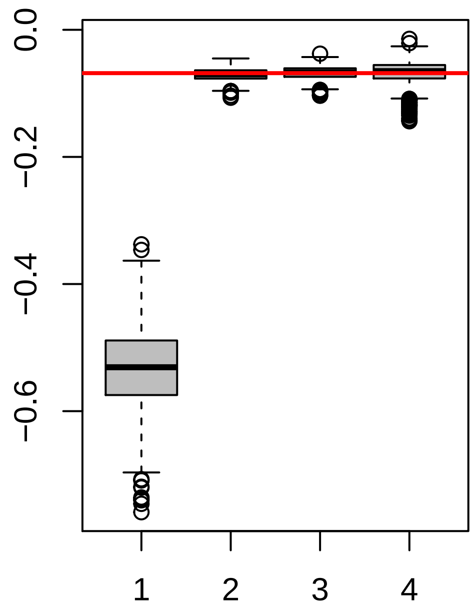
The other way we can approach this problem is to build a mixed effects model, which in general takes the form $y_{ij} = \alpha + \delta_j + e_{ij}$, where α is the predictor coefficients, which are fixed effects, while e_{ij}, which follows a $N(0, \sigma^2)$ distribution, is the variations within the subject groups, and δ_j, which follows a $N(0, \tau^2)$ distribution, is the variations between the subject groups, are random effects. There is a nice package in R called *lme4* that can efficiently build a mixed effects model, its model output is shown below:

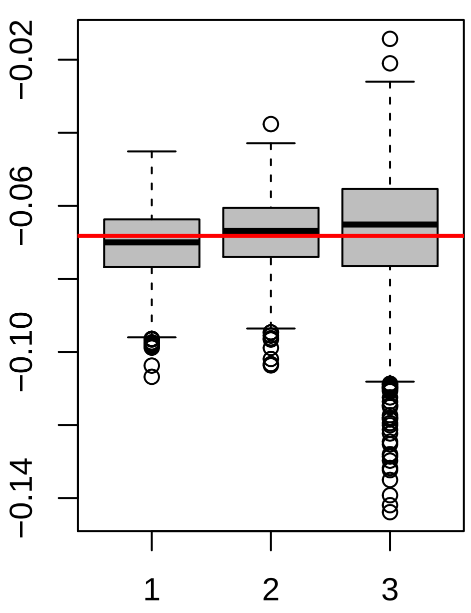
```
## Loading required package: Matrix

## Linear mixed model fit by REML ['lmerMod']
## Formula: score ~ group * week + (week | subject)
##    Data: stroke
## REML criterion at convergence: 1327.747
## Random effects:
##  Groups   Name        Std.Dev. Corr
##  subject  (Intercept) 21.034
##           week         2.937   -0.37
##  Residual              5.220
## Number of obs: 192, groups:  subject, 24
## Fixed Effects:
## (Intercept)       groupB       groupC         week  groupB:week
##     30.1339       3.0357      -0.3348       6.2202      -1.8899
## groupC:week
##     -2.5818
```

Overall, there doesn't seem to be much difference between the mixed effects model and the hierarchical linear model. The three coefficients of mean weekly increases in score are particularly close, differ by only about .02 for every one of them. Moreover, the square of standard deviation of residuals is 27.2484, which is pretty close to σ^2 of our sample from the hierarchical linear models (usually has a mean around 27 with a .95 posterior interval from about 22 to about 35). The standard deviations of β for the hierarchical linear model is slightly higher, which could results from that the Inverse-Wishart distribution offered higher variability on the covariance structure. The estimated correlation ρ is -.37, which is reasonably close to the sampled ρ from the hierarchical linear model.

Our main motivation of using an Inverse-Wishart prior on Σ_β is that we can captured the correlations between the responses more fully, so it's worthwhile to check if that is actually the case. To do this, we will compare three models: one "naive" model with only fixed effects from our covariates, the hierarchical linear model we build, the hierarchical linear model we build in class that uses $\tau^2 I_p$ as the variance of β, and the mixed effects model we build in class. For each model, we used our sampled parameters from our Gibbs sampler to now simulate y. That is, we sampled y based on $N(X\beta, \sigma^2 I_n)$ distribution, where β and σ^2 is now our sampled posteriors. We then define a function called *cor.test* which takes in a vector of numbers as parameter, and calculated the log of estimated of correlation between these numbers. We applied this function to each one of our replicated data sets to get an estimated correlation for each one, and here are the results:





The red line in the graph is the actual correlation ρ of the data. The first boxplot is the estimated ρ from the naive model based only on fixed effects; the second boxplot is the estimated ρ from our hierarchical linear model; the third boxplot is the estimated ρ from the in-class hierarchical linear model; the last boxplot is the estimated ρ from the mixed effects model. In the second graph, I plot the boxplots in the same order but with the naive model removed. It is very apparent from the first plot that the naive model based on fixed effects does not capture the auto-correlation at all as the red line is above all the points from the naive model. All the other models seem to do quite well on capturing the correlations from the model. I then plot the second plots to take a closer look, and indeed all the other models capture the autocorrelation pretty well as their boxplots all have medians close to the real value.

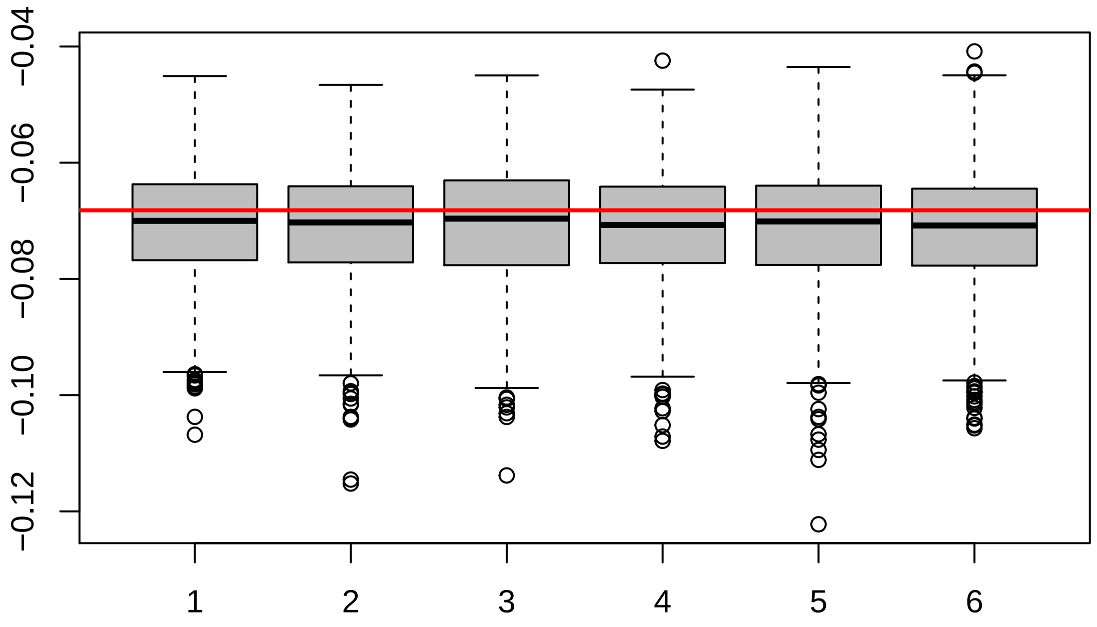
The above plots could be turn into a Bayesian "p-value" as well. Since the boxplot is the autocorrelation of our sample and red line ρ is the actual value, we can calculate how does the estimated mean autocorrelation compare to the actual autocorrelation. Specifically, we obtain this p-value by finding the expectation of an indicator function, $E[I(ct > \rho)]$, where ct is our sample autocorrelations. As the expectation of an indicator function equals to the probability of the event specified in the indicator function, we will get a p-value. Because the boxplots are assumed to be roughly symmetric around the true correlation, if our estimated autocorrelation agrees with the actual value, we would expect a p-value around .5.

```
pValue <- c(mean(ct0>rho), mean(ct>rho), mean(ct1>rho), mean(ctMix>rho))
pValue
```

```
## [1] 0.000 0.428 0.550 0.571
```

These p-value matches with what we roughly expected from the boxplots. The p-value of 0 on the naive model signifies that the models does not capture the true autocorrelation, while other three models all have p-value reasonably close to 0.5 to suggests that they captured correlations in the response.

So far, we have use the noninformative priors as my hyper-priors for Σ_β, and the results are very reasonable. However, if we have some prior information on one or more parameters, then the result would change accordingly. Therefore, we wish to conduct a sensitive analysis to analyze the how does the changes in prior of the Σ_β affect the other parameter or the ability of predictive sample to cover the true autocorrelation of the data. We will first change the value of ξ_β. Under our current setup, $\Lambda_\beta^{-1} = \xi\beta I_p$, meaning our prior assumption for the variance structure is that the intercept and weekly increase in score for each individual subjects are independent and that their variances are identical. A change in ξ does not change this assumption but would change the scale of the variations. I ran 5 different sampler, each with differnt ξ: $1/100$, 1, 5, 100, 10^{10}.

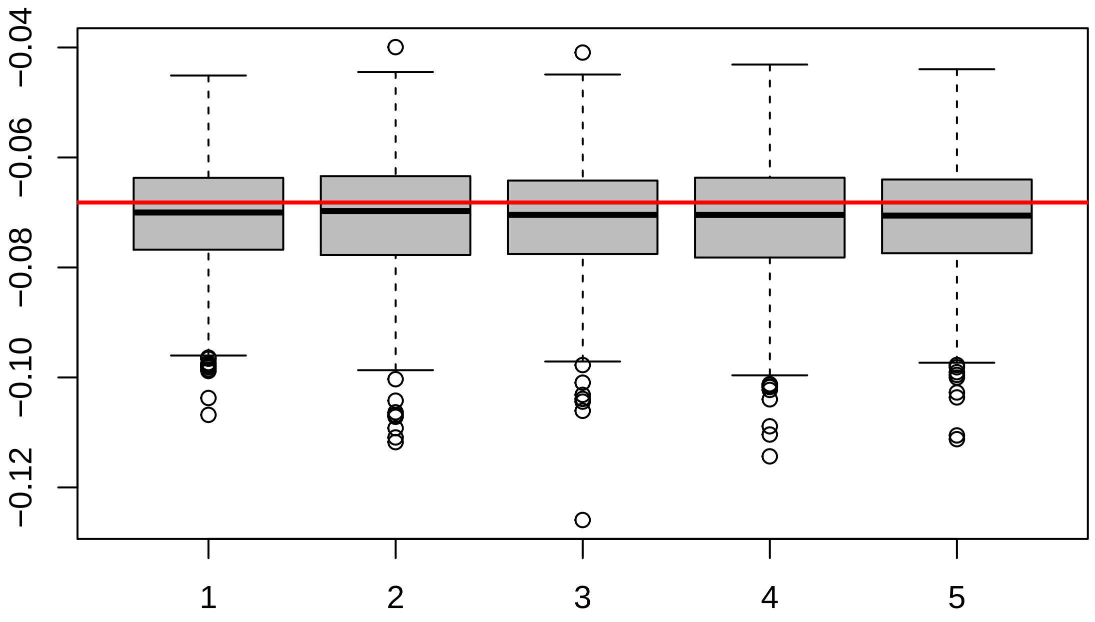


```
## [1] 0.428 0.414 0.437 0.388 0.415 0.397
```

The output of the monitor function for one of the sampler can be seen in the appendix. (example shown is $\xi = 5$) If the ξ_β is large enough (large is relative here, but suppose it is goes to ∞), it'll cause $\Sigma_\beta \otimes I_j$ to be large, and thus cause the estimations of α and β to be unreliable. Otherwise, the data is pretty stable and doesn't affected by the changes in ξ_β too much, and as you can see in the boxplots and p-values, the changes in ξ_β does not seem to have an effect on how well the posterior samples capture the autocorrelations. In the boxplots, the first boxplot is the estimated ρ for my original model, and the subsequent boxplots are estimated ρ for $\xi = 1/100, 1, 5, 100, 10^{10}$. All of the boxplots have median close to the real ρ and all of the hierarchical linear models have p-value around 0.4, which suggests they call captured the autocorrelations of data quite well.

We can further complicate the variance by taking more complex strucuture for Λ_β. Our previous assumption that the variations between the patients and variations between the weeks are the same is highly debatable. In fact, our previous analysis based on this assumption produced a posterior that suggested the opposite. Therefore, the diagonals could have different entries. Furthermore, it is unclear that the relationship between patients and weekly increases in scores are truly independent. As a consequence, our Λ_β does not necessarily need to take a form of a scalar multiplied by an identity matrix. Thus, I made four more samplers, this time not putting

Λ_β as a diagonal matrix. When doing that, I need to make sure that the columns are not linear combinations of each other so that the matrix would be singular. I then perform the same analysis on these samples as before.



```
## [1] 0.428 0.441 0.418 0.419 0.405
```

As with the last section, the monitor output of one sampler is in the appendix (the entry for Λ_β^{-1} is 5, 7, 7, 2 by column), the first boxplot is the estimated ρ for the original hierarchical linear model, and the subsequent boxplot corresponds to the estimated ρ for simulations 7, 8, and 9, each with a different Λ_β. (The exact value can be seen from the appendix) All of the posterior sample are pretty close to each other, all of the boxplots have median around the true ρ, and all p-value is reasonably close to 0.5, which means the autocorrelations of response are well captured. This result, combined with last section's results, suggests that more a more complex structure and different scales on Λ_β does not affect the posterior of Σ_β, α, β, or predicted y that much. The reason is that the conditional posterior of Σ_β involves the summation of $(\beta_j - X\beta_j\alpha)(\beta_j - X\beta_j\alpha)^T$, which will overwhelm the prior matrix Λ_β unless its entries are huge. Then, since the posterior of β and α depends on Σ_β, they would not change much either.

In this *stroke* data set, we made inquiries into several different models in attempt to understand the effect of different treatment on the recovery of stroke. While all the models give out the similiar answers (The new treatments is more effective than the old treatments), the hierarchical linear models were able to capture both the variations in the score and variations in each β (subject and week), and as a consequence can capture the autocorrelation in the response well and can simulate replicated data. The last point is extremely important as if you give it a new data point, its prediction would be more accurate and more precise than a simple model based on fixed effects only. Understand and correctly apply the hierarchical linear model, therefore, would help us to capture more informations about the data, and could provide valuable informations in the future studies.