homework 07

Chaoqun Yin

November 1, 2018

Data analysis

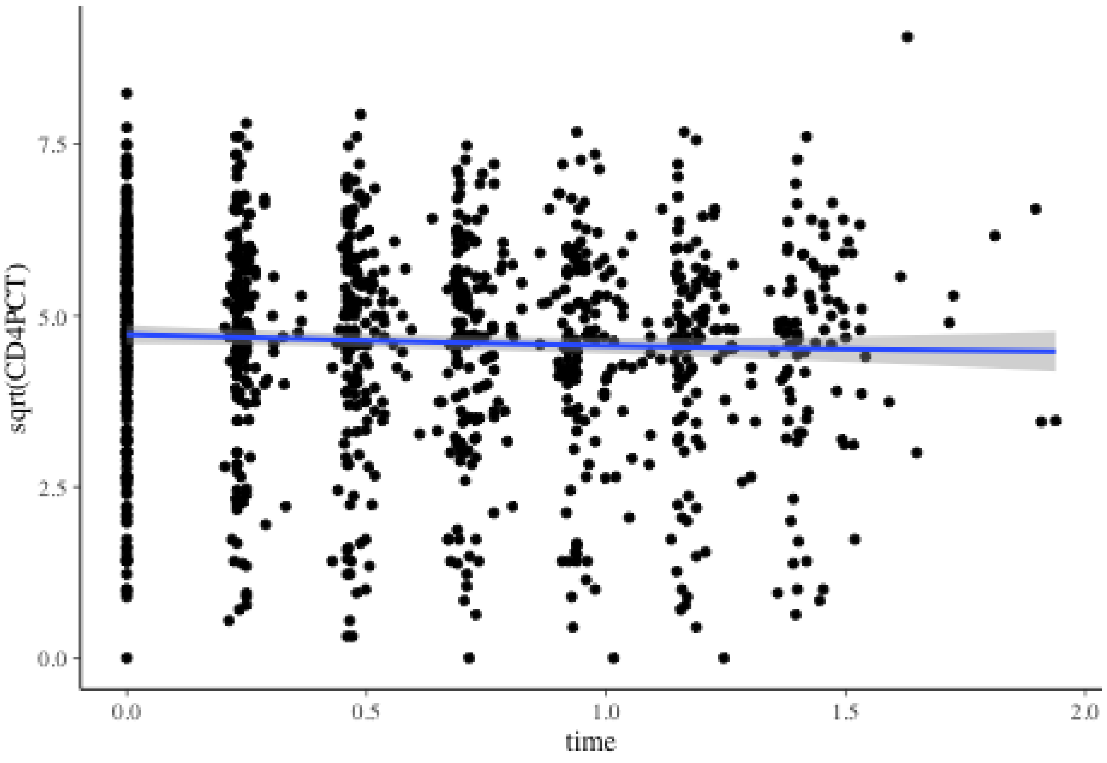
CD4 percentages for HIV infected kids

The folder `cd4` has CD4 percentages for a set of young children with HIV who were measured several times over a period of two years. The dataset also includes the ages of the children at each measurement.

1. Graph the outcome (the CD4 percentage, on the square root scale) for each child as a function of time.

```r
library(ggplot2)
ggplot(data = hiv.data, aes(y = sqrt(CD4PCT), x = time)) +
  geom_point() +
  geom_smooth()
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```



2. Each child's data has a time course that can be summarized by a linear fit. Estimate these lines and plot them for all the children.

```r
reg.1 <- lmer(y ~ 1 + time + (1|newpid),data = hiv.data)
```

```
## Warning: 'rBind' is deprecated.
##  Since R version 3.2.0, base's rbind() should work fine with S4 objects
```

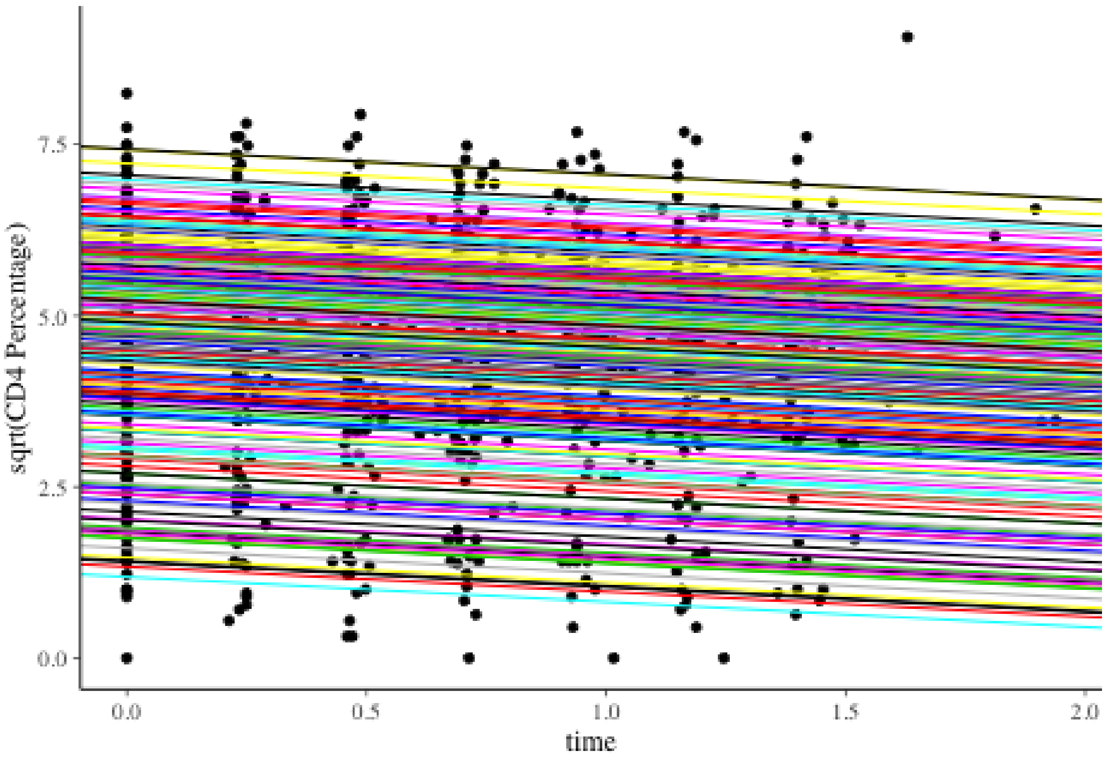
```r
display(reg.1)
```

```
## lmer(formula = y ~ 1 + time + (1 | newpid), data = hiv.data)
##             coef.est coef.se
## (Intercept)  4.76     0.10
## time        -0.37     0.05
##
## Error terms:
##  Groups   Name        Std.Dev.
##  newpid   (Intercept) 1.40
##  Residual             0.77
## ---
## number of obs: 1072, groups: newpid, 250
## AIC = 3148.8, DIC = 3126.9
## deviance = 3133.9
```

```r
#fixef(reg.1)
#ranef(reg.1)

reg.1_coef <- coef(reg.1)
reg.1_coef <- data.frame(reg.1_coef$newpid)
colnames(reg.1_coef) <- c("intercept","time")
reg.1_coef$newpid <- c(1:250)

ggplot(data=hiv.data) +
  geom_point(aes(x=time, y=y)) +
  geom_abline(intercept = reg.1_coef$intercept,
              slope=reg.1_coef$time, color=reg.1_coef$newpid)+
  labs(y="sqrt(CD4 Percentage)")
```



3. Set up a model for the children's slopes and intercepts as a function of the treatment and age at baseline. Estimate this model using the two-step procedure–first estimate the intercept and slope separately for each child, then fit the between-child models using the point estimates from the first step.

```
child_mat <- matrix(0,nrow=254,ncol = 3)
colnames(child_mat) <- c("newpid","intercept","slope")

for (i in unique(hiv.data$newpid)){
  child_lm <- lm(y ~ time, hiv.data[newpid == i,c("y","time")])
  child_mat[i,1] <- i
  child_mat[i,2] <- coef(child_lm)[1]
  child_mat[i,3] <- coef(child_lm)[2]
}

hiv.data.use <- hiv.data[,list(age.baseline=unique(age.baseline),treatment=unique(treatment)), by=newpi
hiv.data.use <- merge(child_mat,hiv.data.use,by="newpid")

lm(intercept~ age.baseline+factor(treatment),data = hiv.data.use)
```

```
##
## Call:
## lm(formula = intercept ~ age.baseline + factor(treatment), data = hiv.data.use)
##
## Coefficients:
##        (Intercept)        age.baseline  factor(treatment)2
##             5.1179             -0.1210              0.1236
```

```
lm(slope~ age.baseline+factor(treatment),data=hiv.data.use)
```

```
##
## Call:
## lm(formula = slope ~ age.baseline + factor(treatment), data = hiv.data.use)
##
## Coefficients:
##        (Intercept)        age.baseline  factor(treatment)2
##           -0.26568            -0.04223            -0.13926
```

4. Write a model predicting CD4 percentage as a function of time with varying intercepts across children. Fit using `lmer()` and interpret the coefficient for time.

```
reg.2 <- lmer(y ~ time + (1|newpid),data = hiv.data)
summary(reg.2)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: y ~ time + (1 | newpid)
##    Data: hiv.data
##
## REML criterion at convergence: 3140.8
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -4.7379 -0.4379  0.0024  0.4324  5.0017
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  newpid   (Intercept) 1.9569   1.3989
##  Residual             0.5968   0.7725
## Number of obs: 1072, groups:  newpid, 250
##
## Fixed effects:
##             Estimate Std. Error t value
## (Intercept)  4.76341    0.09648   49.37
## time        -0.36609    0.05399   -6.78
##
## Correlation of Fixed Effects:
##      (Intr)
## time -0.278
```

5. Extend the model in (4) to include child-level predictors (that is, group-level predictors) for treatment and age at baseline. Fit using `lmer()` and interpret the coefficients on time, treatment, and age at baseline.

```
reg.3 <- lmer(y ~ time + treatment + age.baseline + (1|newpid),data = hiv.data)
summary(reg.3)
```

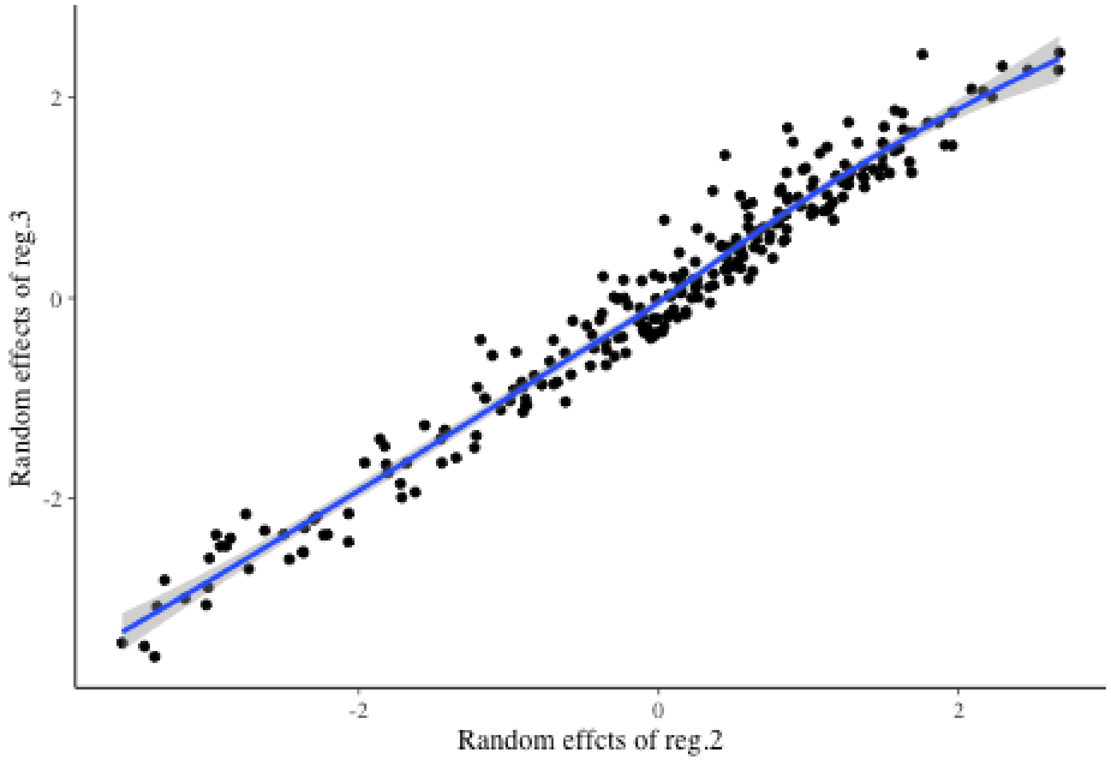
```
## Linear mixed model fit by REML ['lmerMod']
## Formula: y ~ time + treatment + age.baseline + (1 | newpid)
##    Data: hiv.data
##
## REML criterion at convergence: 3137.2
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -4.7490 -0.4392  0.0097  0.4282  5.0141
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  newpid   (Intercept) 1.8897   1.3747
##  Residual             0.5969   0.7726
## Number of obs: 1072, groups:  newpid, 250
##
## Fixed effects:
##              Estimate Std. Error t value
## (Intercept)   4.90606    0.31684  15.485
## time         -0.36216    0.05399  -6.708
## treatment     0.18008    0.18262   0.986
## age.baseline -0.11945    0.04000  -2.986
##
## Correlation of Fixed Effects:
##             (Intr) time   trtmnt
## time        -0.086
## treatment   -0.850  0.010
## age.baselin -0.430 -0.017 -0.003
```

6. Investigate the change in partial pooling from (4) to (5) both graphically and numerically.

```
data_plot <- as.data.frame(cbind(unlist(ranef(reg.2)),unlist(ranef(reg.3))))
colnames(data_plot) <- c("reg.2","reg.3")

ggplot(data=data_plot,aes(x=reg.2,y=reg.3))+geom_point()+geom_smooth()+
  xlab("Random effcts of reg.2")+
  ylab("Random effects of reg.3")
```

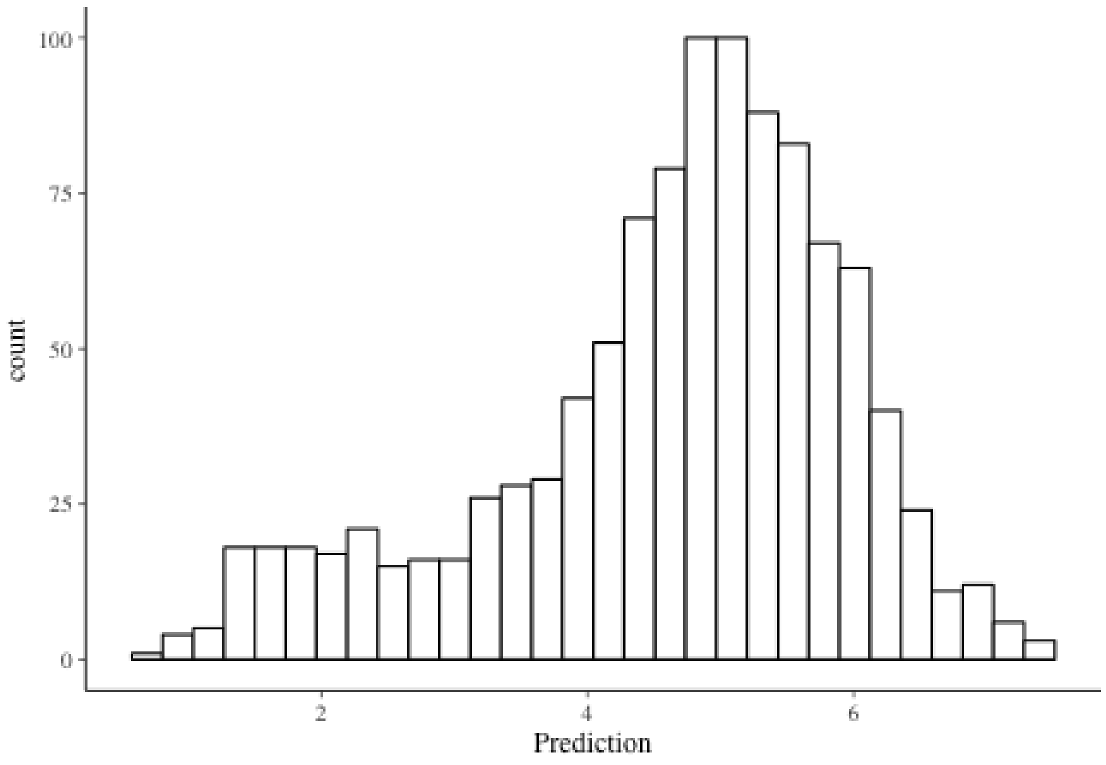
```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



7. Use the model fit from (5) to generate simulation of predicted CD4 percentages for each child in the dataset at a hypothetical next time point.

```r
library(dplyr)
predict_data <- hiv.data %>%
  filter(is.na(hiv.data$treatment)==FALSE) %>%
  filter(is.na(age.baseline)==FALSE) %>%
  select(time,treatment,age.baseline,newpid)
predict_new <- predict(reg.3,newdata=predict_data)
predict_cmb <- cbind(predict_new,predict_data)
colnames(predict_cmb)[1] <- c("Prediction")
ggplot(predict_cmb,aes(x=Prediction))+geom_histogram(color="black", fill="white")
```

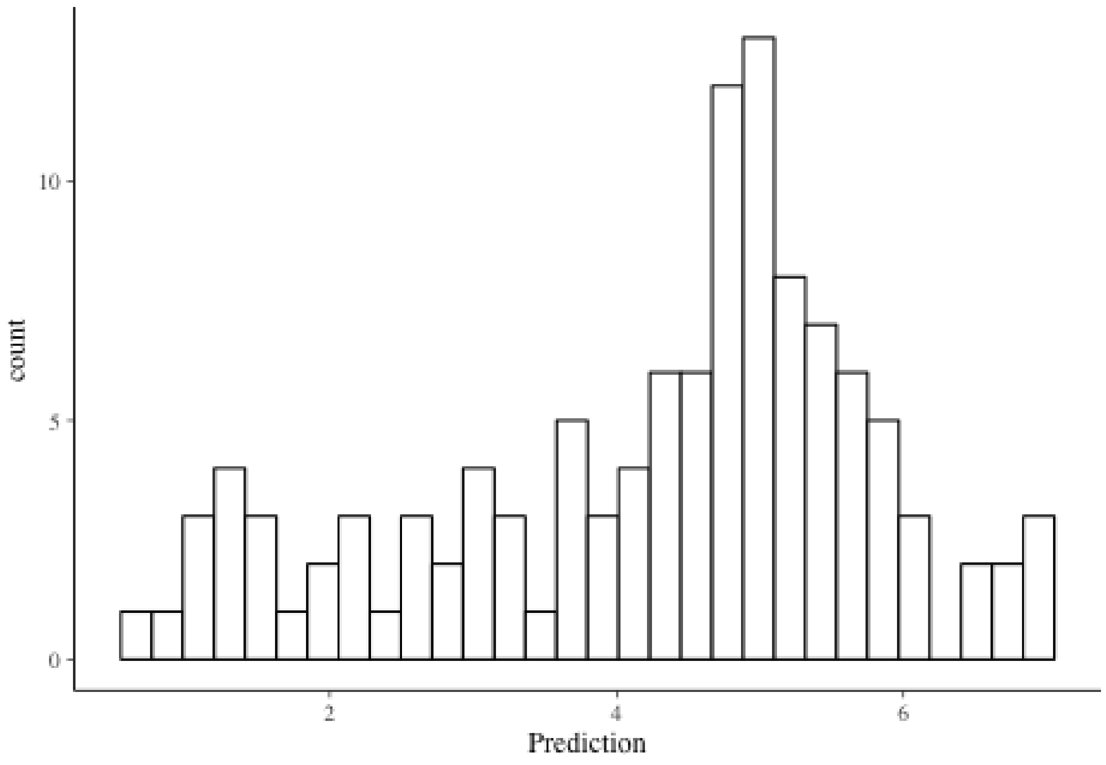
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



8. Use the same model fit to generate simulations of CD4 percentages at each of the time periods for a new child who was 4 years old at baseline.

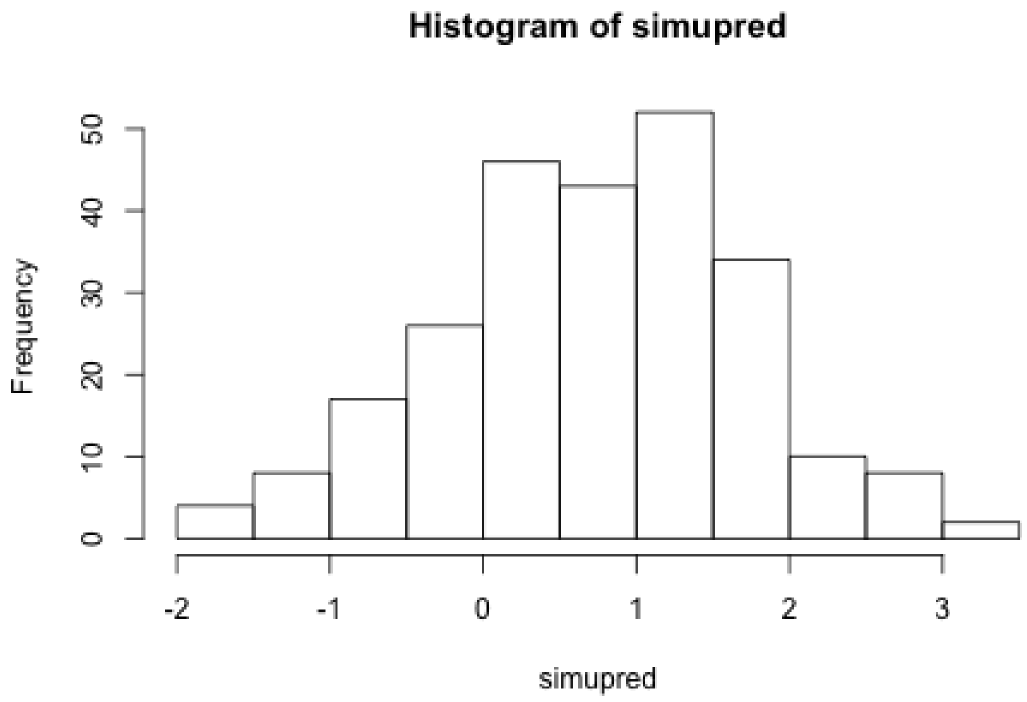
```
predict_data2 <- hiv.data %>%
  filter(is.na(hiv.data$treatment)==FALSE) %>%
  filter(is.na(age.baseline)==FALSE) %>%
  select(time,treatment,age.baseline,newpid,CD4CNT) %>%
  filter(round(age.baseline)==4)
predict_new2 <- predict(reg.3,newdata=predict_data2)
predict_cmb2 <- cbind(predict_new2,predict_data2)
colnames(predict_cmb2)[1] <- c("Prediction")
ggplot(predict_cmb2,aes(x=Prediction))+geom_histogram(color="black", fill="white")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



9. Posterior predictive checking: continuing the previous exercise, use the fitted model from (5) to simulate a new dataset of CD4 percentages (with the same sample size and ages of the original dataset) for the final time point of the study, and record the average CD4 percentage in this sample. Repeat this process 1000 times and compare the simulated distribution to the observed CD4 percentage at the final time point for the actual data.

```
pred_new_9 <- hiv.data[,list(time=max(time),age.baseline=unique(age.baseline),
                       treatment=unique(treatment)),by =newpid]
cm3<-coef(reg.3)$newpid
sigy<-sigma.hat(reg.3)$sigma$data
predy<-cm3[,1]+cm3[,2]*pred_new_9$time+cm3[,3]*pred_new_9$age.baseline+cm3[,4]*(pred_new_9$treatment-1)
avg.pred.CD4PCT<-NULL
simupred<-matrix(NA,nrow(pred_new_9),1000)
for (i in 1:1000){
  ytilde<-rnorm(predy,sigy)
  simupred[,1]<-ytilde
}
hist(simupred)
```



10. Extend the modelto allow for varying slopes for the time predictor.

```
hiv_reg_vslope <- lmer(y~time+factor(treatment)+age.baseline+(1+time|newpid), data = hiv.data)
summary(hiv_reg_vslope)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: y ~ time + factor(treatment) + age.baseline + (1 + time | newpid)
##    Data: hiv.data
##
## REML criterion at convergence: 3107
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -5.0998 -0.4057  0.0174  0.4030  5.0157
##
## Random effects:
##  Groups   Name        Variance Std.Dev. Corr
##  newpid   (Intercept) 1.8464   1.3588
##           time        0.3374   0.5808   -0.04
##  Residual             0.5145   0.7173
## Number of obs: 1072, groups:  newpid, 250
##
## Fixed effects:
##                     Estimate Std. Error t value
## (Intercept)          5.10850    0.18594  27.474
## time                -0.35258    0.06763  -5.214
## factor(treatment)2  0.15952    0.18137   0.880
## age.baseline        -0.12423    0.03971  -3.128
##
## Correlation of Fixed Effects:
```

```
##              (Intr) time   fct()2
## time         -0.114
## fctr(trtm)2 -0.463  0.010
## age.baselin -0.729 -0.013 -0.004
```

11. Next fit a model that does not allow for varying slopes but does allow for different coefficients for each time point (rather than fitting the linear trend).

```
hiv_reg <- lmer(y~factor(time)+(1|newpid), data = hiv.data)
summary(hiv_reg)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: y ~ factor(time) + (1 | newpid)
##    Data: hiv.data
##
## REML criterion at convergence: 2170.5
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.9506 -0.2738  0.0000  0.2631  4.1995
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  newpid   (Intercept) 1.9868   1.4096
##  Residual             0.4951   0.7036
## Number of obs: 1072, groups:  newpid, 250
##
## Fixed effects:
##                               Estimate Std. Error t value
## (Intercept)                   4.769568   0.100065   47.66
## factor(time)0.205            -1.226876   0.667895   -1.84
## factor(time)0.209999999999999  0.207180   0.889535    0.23
## factor(time)0.213333333333333  0.158952   0.944335    0.17
## factor(time)0.213333333333334 -1.204186   0.944335   -1.28
## factor(time)0.215833333333332  1.474403   0.896474    1.64
## factor(time)0.215833333333334 -0.247137   0.844335   -0.29
## factor(time)0.216666666666667 -0.347741   0.803115   -0.43
## factor(time)0.218333333333334  0.067513   0.896474    0.08
## factor(time)0.219166666666667 -0.478445   0.852896   -0.56
## factor(time)0.221666666666667  0.192599   0.944335    0.20
## factor(time)0.224166666666666  1.651562   0.856205    1.93
## factor(time)0.224166666666667 -1.533347   0.629556   -2.44
## factor(time)0.226666666666667  1.415890   0.585295    2.42
## factor(time)0.227499999999999 -1.563378   0.887929   -1.76
## factor(time)0.2275            0.072451   0.460800    0.16
## factor(time)0.229999999999999 -0.357796   0.585602   -0.61
## factor(time)0.23             -0.108025   0.123504   -0.87
## factor(time)0.2325           -0.586847   0.397658   -1.48
## factor(time)0.233333333333333  0.018469   0.839056    0.02
## factor(time)0.235000000000001 -1.957440   0.798740   -2.45
## factor(time)0.235833333333333  0.043869   0.294715    0.15
## factor(time)0.235833333333334  0.183556   0.618437    0.30
## factor(time)0.2375            1.438741   0.888147    1.62
## factor(time)0.238333333333333 -0.274278   0.486002   -0.56
## factor(time)0.238333333333334  0.853827   0.817299    1.04
## factor(time)0.240833333333333 -0.211997   0.784471   -0.27
## factor(time)0.240833333333334  0.341076   0.592169    0.58
## factor(time)0.243333333333333 -0.509639   0.889708   -0.57
## factor(time)0.244166666666667  0.088744   0.478383    0.19
## factor(time)0.245833333333333  0.093406   0.432676    0.22
## factor(time)0.245833333333334 -0.248513   0.604743   -0.41
## factor(time)0.246666666666666 -0.506662   0.647659   -0.78
## factor(time)0.246666666666667  0.381414   0.854785    0.45
## factor(time)0.249166666666666  0.151166   0.385297    0.39
## factor(time)0.249166666666667 -0.479340   0.190045   -2.52
## factor(time)0.251666666666667  0.250594   0.430045    0.58
## factor(time)0.251666666666668  0.307869   0.803546    0.38
## factor(time)0.2525            -0.051823   0.944335   -0.05
## factor(time)0.254166666666667 -0.627409   0.840636   -0.75
## factor(time)0.255              0.325945   0.800623    0.41
## factor(time)0.256666666666667  0.292102   0.631505    0.46
## factor(time)0.257499999999999  0.089346   0.843495    0.11
## factor(time)0.2575             0.450784   0.630964    0.71
## factor(time)0.2625            -0.038563   0.842884   -0.05
## factor(time)0.265             -0.159570   0.845421   -0.19
## factor(time)0.265833333333333 -0.360456   0.838630   -0.43
## factor(time)0.268333333333333 -0.339997   0.590420   -0.58
## factor(time)0.268333333333334  0.071720   0.489864    0.15
## factor(time)0.2875             0.502035   0.525423    0.96
## factor(time)0.289999999999999 -0.774070   0.944335   -0.82
## factor(time)0.293333333333333 -0.282857   0.944335   -0.30
## factor(time)0.304166666666667  0.309641   0.889195    0.35
## factor(time)0.306666666666666 -0.276897   0.889031   -0.31
## factor(time)0.306666666666667  0.216413   0.585991    0.37
## factor(time)0.325833333333334 -0.158816   0.891552   -0.18
## factor(time)0.328333333333333 -0.873166   0.944335   -0.92
## factor(time)0.331666666666667  0.048151   0.944335    0.05
## factor(time)0.358333333333333  0.572240   0.857006    0.67
## factor(time)0.364166666666667 -0.069913   0.607607   -0.12
## factor(time)0.429166666666666 -0.288824   0.944335   -0.31
## factor(time)0.429166666666667 -0.438378   0.896474   -0.49
## factor(time)0.438333333333333 -0.851991   0.896474   -0.95
## factor(time)0.440833333333333 -0.097299   0.772169   -0.13
## factor(time)0.443333333333332  0.154974   0.847647    0.18
## factor(time)0.449166666666667 -0.080601   0.792136   -0.10
## factor(time)0.454166666666666  0.269002   0.896474    0.30
## factor(time)0.454166666666667  0.094124   0.856205    0.11
## factor(time)0.455             -1.452938   0.887929   -1.64
## factor(time)0.456666666666667  0.239063   0.944335    0.25
## factor(time)0.4575            -0.137210   0.490456   -0.28
## factor(time)0.459166666666667  0.267761   0.619176    0.43
## factor(time)0.459999999999999 -0.187366   0.458238   -0.41
## factor(time)0.46              -0.267033   0.170802   -1.56
## factor(time)0.460000000000001 -0.273311   0.313779   -0.87
## factor(time)0.462499999999999 -0.773148   0.480801   -1.61
## factor(time)0.4625             0.428628   0.411262    1.04
## factor(time)0.463333333333333 -0.700609   0.810605   -0.86
## factor(time)0.465             -0.880672   0.480259   -1.83
## factor(time)0.465833333333333  0.321130   0.586093    0.55
## factor(time)0.465833333333334 -0.604840   0.828224   -0.73
## factor(time)0.4675             1.612699   0.609019    2.65
## factor(time)0.468333333333335 -0.796282   0.852896   -0.93
## factor(time)0.470833333333333 -0.358993   0.463422   -0.77
## factor(time)0.470833333333334 -0.455022   0.607535   -0.75
## factor(time)0.473333333333333 -0.170787   0.620439   -0.28
## factor(time)0.473333333333334  0.453871   0.791748    0.57
## factor(time)0.474166666666666 -0.665417   0.820271   -0.81
## factor(time)0.474166666666667 -0.810403   0.802688   -1.01
## factor(time)0.475833333333333 -0.457044   0.803094   -0.57
## factor(time)0.475833333333334 -1.408959   0.577288   -2.44
## factor(time)0.476666666666667  0.605763   0.404257    1.50
## factor(time)0.479166666666666 -0.235713   0.579617   -0.41
## factor(time)0.479166666666667 -0.032123   0.251706   -0.13
## factor(time)0.481666666666666  0.114945   0.586544    0.20
## factor(time)0.481666666666667  0.218026   0.427140    0.51
## factor(time)0.484166666666667 -2.714651   0.805427   -3.37
## factor(time)0.485              0.899936   0.622062    1.45
## factor(time)0.487499999999999 -0.227278   0.769321   -0.30
## factor(time)0.4875             1.796125   0.820889    2.19
## factor(time)0.487500000000001  1.764383   0.803546    2.20
## factor(time)0.489999999999999  0.094412   0.854785    0.11
## factor(time)0.495             -0.144600   0.597975   -0.24
## factor(time)0.495833333333333 -0.472833   0.859802   -0.55
## factor(time)0.495833333333334 -0.123628   0.944335   -0.13
## factor(time)0.498333333333333 -0.588715   0.387469   -1.52
## factor(time)0.498333333333334 -0.580897   0.377044   -1.54
## factor(time)0.500833333333333 -0.261680   0.793174   -0.33
## factor(time)0.500833333333334  0.053607   0.813317    0.07
## factor(time)0.501666666666667 -1.183614   0.614362   -1.93
## factor(time)0.503333333333334  0.593039   0.838630    0.71
## factor(time)0.504166666666666  0.243929   0.517747    0.47
## factor(time)0.505833333333333 -1.532777   0.818758   -1.87
## factor(time)0.509166666666666 -0.421767   0.896474   -0.47
## factor(time)0.511666666666667 -1.100328   0.873457   -1.26
## factor(time)0.514166666666666  0.119718   0.622831    0.19
## factor(time)0.515              0.161511   0.944335    0.17
## factor(time)0.5175            -0.420848   0.384095   -1.10
## factor(time)0.517500000000001 -1.128677   0.820580   -1.38
## factor(time)0.533333333333333 -0.271452   0.643410   -0.42
## factor(time)0.533333333333334 -1.071042   0.866200   -1.24
## factor(time)0.534166666666667  0.221594   0.616684    0.36
## factor(time)0.536666666666667 -0.479104   0.436860   -1.10
## factor(time)0.555833333333333 -0.219958   0.615504   -0.36
## factor(time)0.558333333333333  1.735017   0.944335    1.84
## factor(time)0.564166666666667 -0.771678   0.893563   -0.86
## factor(time)0.575             -0.347199   0.633479   -0.55
## factor(time)0.580833333333333 -0.280191   0.944335   -0.30
## factor(time)0.5825             0.640225   0.944335    0.68
## factor(time)0.594166666666667 -0.482295   0.891552   -0.54
## factor(time)0.610833333333333 -1.086082   0.944335   -1.15
## factor(time)0.6375             1.801451   0.896474    2.01
## factor(time)0.648333333333333 -2.110579   0.887959   -2.38
## factor(time)0.651666666666666 -1.281269   0.874578   -1.47
## factor(time)0.6575            -1.006976   0.944335   -1.07
## factor(time)0.67              -0.425508   0.850078   -0.50
## factor(time)0.670833333333333 -0.991365   0.491383   -2.02
## factor(time)0.673333333333333  0.051111   0.944335    0.05
## factor(time)0.675833333333333 -0.149936   0.667895   -0.22
## factor(time)0.684166666666667  0.534412   0.629675    0.85
## factor(time)0.685              0.613652   0.837774    0.73
## factor(time)0.6875            -1.589771   0.607402   -2.62
## factor(time)0.689166666666666 -0.089993   0.889535   -0.10
## factor(time)0.689166666666667  0.162698   0.806937    0.20
## factor(time)0.69              -0.177252   0.159701   -1.11
## factor(time)0.6925             0.521706   0.553126    0.94
## factor(time)0.692500000000001  0.904909   0.835260    1.08
## factor(time)0.693333333333334 -0.478445   0.852896   -0.56
## factor(time)0.695              0.370779   0.844335    0.44
## factor(time)0.695833333333333 -0.315443   0.855954   -0.37
## factor(time)0.695833333333334 -1.682770   0.831727   -2.02
## factor(time)0.695833333333335  0.817961   0.594681    1.38
## factor(time)0.697500000000001 -1.490182   0.788534   -1.89
## factor(time)0.698333333333332  0.048322   0.836309    0.06
## factor(time)0.698333333333333 -0.605469   0.588373   -1.03
## factor(time)0.700833333333333  1.558778   0.486150    3.21
## factor(time)0.703333333333333 -0.890523   0.576335   -1.55
## factor(time)0.703333333333334 -1.065773   0.817299   -1.30
## factor(time)0.704166666666667 -4.769574   0.802688   -5.94
## factor(time)0.705833333333333 -0.476319   0.488425   -0.98
## factor(time)0.705833333333334 -0.449623   0.770690   -0.58
## factor(time)0.706666666666667  2.360514   0.797411    2.96
## factor(time)0.709166666666666 -0.110098   0.796240   -0.14
## factor(time)0.709166666666667  0.227515   0.276843    0.82
## factor(time)0.711666666666666  0.101433   0.839961    0.12
## factor(time)0.711666666666667 -0.465341   0.814847   -0.57
## factor(time)0.711666666666668 -0.880563   0.568408   -1.55
## factor(time)0.714166666666667 -1.430035   0.576376   -2.48
## factor(time)0.714999999999999 -0.719300   0.793317   -0.91
## factor(time)0.715000000000001 -0.615551   0.858566   -0.72
## factor(time)0.7175            -1.012083   0.805427   -1.26
## factor(time)0.72              -4.316305   0.779875   -5.53
## factor(time)0.725             -0.654615   0.599541   -1.09
## factor(time)0.725833333333332  0.350514   0.843495    0.42
## factor(time)0.725833333333333  0.722607   0.829229    0.87
## factor(time)0.725833333333334  0.328194   0.944335    0.35
## factor(time)0.728333333333333 -0.256206   0.289837   -0.88
## factor(time)0.730833333333333  0.078620   0.505267    0.16
## factor(time)0.733333333333333 -0.737440   0.842927   -0.87
## factor(time)0.734166666666666 -2.809390   0.806103   -3.49
## factor(time)0.735833333333333 -0.930860   0.667895   -1.39
## factor(time)0.736666666666666  0.049558   0.847647    0.06
## factor(time)0.736666666666667  1.506491   0.842679    1.79
## factor(time)0.7425            -0.481665   0.944335   -0.51
## factor(time)0.744166666666667  0.150697   0.573413    0.26
## factor(time)0.745             -0.385140   0.822975   -0.47
## factor(time)0.7475            -0.358934   0.365447   -0.98
## factor(time)0.752500000000001 -1.651959   0.838630   -1.97
## factor(time)0.758333333333334  0.222837   0.857006    0.26
## factor(time)0.761666666666667 -0.590758   0.807723   -0.73
## factor(time)0.763333333333333 -0.134067   0.814749   -0.16
## factor(time)0.763333333333335 -0.016868   0.866200   -0.02
## factor(time)0.764166666666666 -0.450129   0.859802   -0.52
## factor(time)0.765833333333333 -1.215075   0.873457   -1.39
## factor(time)0.766666666666667 -0.444238   0.385404   -1.15
## factor(time)0.775             -1.571834   0.809510   -1.94
## factor(time)0.78               0.887286   1.578578    0.56
## factor(time)0.783333333333333  1.054204   0.889195    1.19
## factor(time)0.785             -0.231228   0.828690   -0.28
## factor(time)0.785833333333333  0.443859   0.801492    0.55
## factor(time)0.788333333333333 -0.742195   0.944335   -0.79
## factor(time)0.794166666666667 -0.655414   0.816973   -0.80
## factor(time)0.8025            -0.561678   0.576533   -0.97
## factor(time)0.805             -1.579665   0.820580   -1.93
## factor(time)0.805000000000001  0.076655   0.788670    0.10
## factor(time)0.807500000000001 -0.323903   0.893563   -0.36
## factor(time)0.824166666666667 -0.009591   0.633479   -0.02
## factor(time)0.8625            -0.449827   0.608076   -0.74
## factor(time)0.8675             0.559600   0.896474    0.62
## factor(time)0.878333333333334 -0.231051   0.887959   -0.26
## factor(time)0.881666666666666  0.938199   0.944335    0.99
## factor(time)0.895833333333333 -0.725147   0.887929   -0.82
## factor(time)0.900833333333333 -0.493064   0.850078   -0.58
## factor(time)0.900833333333334  0.301367   0.584032    0.52
## factor(time)0.903333333333333 -1.029916   0.764785   -1.35
## factor(time)0.903333333333334  2.250866   0.944335    2.38
## factor(time)0.905833333333334  1.261612   0.896474    1.41
## factor(time)0.908333333333334 -0.438378   0.896474   -0.49
## factor(time)0.909166666666666  2.462470   0.944335    2.61
## factor(time)0.909166666666667 -0.295149   0.810605   -0.36
## factor(time)0.911666666666667 -0.259061   0.787408   -0.33
## factor(time)0.914166666666667  0.576386   0.586101    0.98
## factor(time)0.9175            -0.489931   0.837774   -0.58
## factor(time)0.919166666666667  0.051386   0.476409    0.11
## factor(time)0.919999999999998 -0.940748   0.785511   -1.20
## factor(time)0.92              -0.951061   0.266692   -3.57
## factor(time)0.920000000000001 -0.146212   0.461200   -0.32
## factor(time)0.9225            -0.530739   0.586742   -0.90
## factor(time)0.925833333333333 -1.738964   0.590085   -2.95
## factor(time)0.925833333333334  0.340248   0.835260    0.41
## factor(time)0.928333333333333  0.037925   0.786996    0.05
## factor(time)0.928333333333334 -0.935158   0.828224   -1.13
## factor(time)0.930833333333333 -0.722569   0.557367   -1.30
## factor(time)0.930833333333334 -1.969320   0.809787   -2.43
## factor(time)0.933333333333332  0.431358   0.824949    0.52
## factor(time)0.933333333333333  0.013104   0.587975    0.02
## factor(time)0.934166666666664 -0.823625   0.836309   -0.98
## factor(time)0.934166666666667 -1.034583   0.802688   -1.29
## factor(time)0.935833333333333 -0.521517   0.590830   -0.88
## factor(time)0.935833333333334 -0.733569   0.826636   -0.89
## factor(time)0.936666666666667  0.135900   0.785600    0.17
## factor(time)0.938333333333334  0.033973   0.803301    0.04
## factor(time)0.939166666666666 -0.186732   0.333447   -0.56
## factor(time)0.939166666666667  0.037504   0.330737    0.11
## factor(time)0.939166666666668 -1.718841   0.815748   -2.11
## factor(time)0.941666666666666 -0.295776   0.406572   -0.73
## factor(time)0.941666666666667 -0.493855   0.814847   -0.61
## factor(time)0.944166666666667  0.123153   0.590848    0.21
## factor(time)0.9475             0.658797   0.585171    1.13
## factor(time)0.952500000000001 -0.446713   0.791748   -0.56
## factor(time)0.955              1.561105   0.839961    1.86
## factor(time)0.955000000000001 -0.472901   0.813317   -0.58
## factor(time)0.955833333333333 -1.057318   0.847647   -1.25
## factor(time)0.9575             0.386676   0.788979    0.49
## factor(time)0.958333333333333  0.112486   0.361580    0.31
## factor(time)0.958333333333334 -0.141636   0.460195   -0.31
## factor(time)0.960833333333333 -1.469735   0.819513   -1.79
## factor(time)0.964166666666666 -0.014802   0.843495   -0.02
## factor(time)0.964166666666667  0.498011   0.829229    0.60
## factor(time)0.965833333333333  0.049558   0.847647    0.06
## factor(time)0.976666666666667 -0.514257   0.840636   -0.61
## factor(time)0.977499999999999 -3.223603   0.806103   -4.00
## factor(time)0.9775            -0.388406   0.361563   -1.07
## factor(time)0.977500000000001  0.583422   0.858566    0.68
## factor(time)0.982499999999999 -0.310236   0.821371   -0.38
## factor(time)0.9825            -0.084077   0.842927   -0.10
## factor(time)0.983333333333333 -0.024900   0.877530   -0.03
## factor(time)0.985833333333333 -0.411304   0.944335   -0.44
## factor(time)0.996666666666666 -0.611878   0.565856   -1.08
## factor(time)0.996666666666667 -0.945729   0.419655   -2.25
## factor(time)0.999166666666667 -0.977614   0.780572   -1.25
## factor(time)1                 -0.989028   0.839056   -1.18
## factor(time)1.00166666666667  -0.360456   0.838630   -0.43
## factor(time)1.0025            -1.298607   0.842679   -1.54
## factor(time)1.01083333333333  -0.319069   0.807723   -0.40
## factor(time)1.0125            -0.750908   0.814749   -0.92
## factor(time)1.01583333333333  -0.955228   0.605094   -1.58
## factor(time)1.02083333333333  -1.287823   0.852052   -1.51
## factor(time)1.02333333333333  -0.572698   0.896474   -0.64
## factor(time)1.0325             0.786867   0.889195    0.88
## factor(time)1.035             -0.366242   0.386699   -0.95
## factor(time)1.04833333333333  -1.287006   0.873457   -1.47
## factor(time)1.05333333333333  -0.133114   0.828690   -0.16
## factor(time)1.05416666666667  -0.539067   0.579938   -0.93
## factor(time)1.07583333333333  -1.001225   0.846718   -1.18
## factor(time)1.08666666666667   0.129411   1.578578    0.08
## factor(time)1.09              -1.528740   0.944335   -1.62
## factor(time)1.0925            -0.633171   0.499895   -1.27
## factor(time)1.11416666666667   0.099585   0.853916    0.12
## factor(time)1.11666666666667   0.938199   0.944335    0.99
## factor(time)1.13083333333333   0.282810   0.850078    0.33
## factor(time)1.13583333333333  -2.799413   0.944335   -2.96
## factor(time)1.13916666666667   0.114333   0.785600    0.15
## factor(time)1.14166666666667  -0.477028   0.574995   -0.83
## factor(time)1.14416666666667   1.060697   0.856205    1.24
## factor(time)1.145             -0.423028   0.887929   -0.48
## factor(time)1.1475            -1.346340   0.837774   -1.61
## factor(time)1.14916666666667  -0.670284   0.574019   -1.17
## factor(time)1.15              -0.406816   0.218973   -1.86
## factor(time)1.1525            -0.918511   0.457969   -2.01
## factor(time)1.15583333333333  -1.165888   0.489397   -2.38
## factor(time)1.1575            -0.140648   0.462527   -0.30
## factor(time)1.15833333333333  -1.634679   0.593283   -2.76
## factor(time)1.16083333333333  -1.023765   0.408913   -2.50
## factor(time)1.16333333333333   0.861129   0.594373    1.45
## factor(time)1.16416666666667   0.313932   0.836309    0.38
## factor(time)1.16583333333333  -1.575529   0.831662   -1.89
## factor(time)1.16833333333333  -0.315058   0.803301   -0.39
## factor(time)1.16916666666667  -0.519501   0.308375   -1.68
## factor(time)1.17166666666667  -1.390847   0.574315   -2.42
## factor(time)1.17666666666667  -1.308144   0.826636   -1.58
## factor(time)1.1775            -0.096984   0.803094   -0.12
## factor(time)1.18              -0.450033   0.796240   -0.57
## factor(time)1.18833333333333  -0.103804   0.219879   -0.47
## factor(time)1.19666666666667   0.323759   0.629187    0.51
## factor(time)1.20166666666667   0.087381   0.478257    0.18
## factor(time)1.20416666666667  -0.619201   0.593045   -1.04
## factor(time)1.20666666666667   0.298968   0.788979    0.38
## factor(time)1.2075            -1.068470   0.471992   -2.26
## factor(time)1.2125             0.101327   0.842927    0.12
## factor(time)1.22416666666667  -0.306556   0.858566   -0.36
## factor(time)1.22583333333333   0.020885   0.859929    0.02
## factor(time)1.22666666666667  -0.017402   0.592221   -0.03
## factor(time)1.22916666666667  -0.120776   0.589096   -0.21
## factor(time)1.23166666666667  -1.420232   0.803115   -1.77
## factor(time)1.2325            -0.123764   0.797411   -0.16
## factor(time)1.24583333333333  -1.099417   0.479220   -2.29
## factor(time)1.24833333333333  -1.653424   0.842679   -1.96
## factor(time)1.25333333333333  -0.991393   0.896474   -1.11
## factor(time)1.26166666666667  -0.421045   0.814749   -0.52
## factor(time)1.265             -0.193560   0.471145   -0.41
## factor(time)1.2675            -0.255216   0.839056   -0.30
## factor(time)1.28416666666667  -2.296963   0.859802   -2.67
## factor(time)1.3025            -0.690644   0.873457   -0.79
## factor(time)1.30333333333333  -0.336692   0.569163   -0.59
## factor(time)1.31166666666667  -1.863853   0.809510   -2.30
## factor(time)1.34166666666667  -1.112087   0.785600   -1.42
## factor(time)1.35              -0.550790   0.874578   -0.63
## factor(time)1.35833333333333  -1.662568   0.837774   -1.98
## factor(time)1.36              -0.089993   0.889535   -0.10
## factor(time)1.36083333333333  -0.053401   0.891552   -0.06
## factor(time)1.36583333333333  -0.746767   0.787408   -0.95
## factor(time)1.37166666666667  -0.346994   0.566445   -0.61
## factor(time)1.37416666666667  -0.691187   0.785511   -0.88
## factor(time)1.375             -0.831183   0.789534   -1.05
## factor(time)1.37666666666667  -3.027990   0.763248   -3.97
## factor(time)1.37916666666667   0.033907   0.779263    0.04
## factor(time)1.38              -0.416891   0.323762   -1.29
## factor(time)1.3825            -1.524985   0.550030   -2.77
## factor(time)1.38583333333333  -0.782336   0.592215   -1.32
## factor(time)1.38583333333334   0.170415   0.828224    0.21
## factor(time)1.3875             -0.222944   0.565738   -0.39
## factor(time)1.38833333333333  -1.465144   0.799007   -1.83
## factor(time)1.39083333333333  -0.582446   0.571575   -1.02
## factor(time)1.39583333333333  -0.573371   0.597476   -0.96
## factor(time)1.39666666666667   0.315484   0.766329    0.41
## factor(time)1.39833333333333  -0.332316   0.824752   -0.40
## factor(time)1.39916666666667  -0.472081   0.309771   -1.52
## factor(time)1.40166666666667  -2.155714   0.603021   -3.57
## factor(time)1.41               -1.281746   0.552646   -2.32
## factor(time)1.4125             -0.691036   0.826636   -0.84
## factor(time)1.415              -0.176664   0.574657   -0.31
## factor(time)1.41833333333333  -0.135883   0.455992   -0.30
## factor(time)1.42083333333333  -0.742195   0.944335   -0.79
## factor(time)1.42416666666667   0.386892   0.785885    0.49
## factor(time)1.42583333333333  -0.188695   0.813317   -0.23
## factor(time)1.42916666666667   0.671162   0.867880    0.77
## factor(time)1.43166666666667  -0.125068   0.582487   -0.21
## factor(time)1.43666666666667   0.454994   0.807445    0.56
## factor(time)1.4375             -0.881048   0.786381   -1.12
## factor(time)1.44583333333333  -3.386943   0.806103   -4.20
## factor(time)1.45333333333333  -0.852592   0.896474   -0.95
## factor(time)1.45416666666667  -0.439392   0.858566   -0.51
## factor(time)1.45583333333333  -0.562055   0.788979   -0.71
## factor(time)1.45666666666667  -0.186101   0.410081   -0.45
## factor(time)1.4625              0.314905   0.797411    0.39
## factor(time)1.47                0.183589   0.582874    0.31
## factor(time)1.4725              0.125183   0.821371    0.15
## factor(time)1.475               0.271834   0.842927    0.32
## factor(time)1.48166666666667  -1.522995   0.842679   -1.81
## factor(time)1.48333333333333  -0.601486   0.896474   -0.67
## factor(time)1.4925             -2.585387   0.803115   -3.22
## factor(time)1.495               0.196603   0.482137    0.41
## factor(time)1.4975              0.426067   0.629187    0.68
## factor(time)1.5                -0.302003   0.814749   -0.37
## factor(time)1.50583333333333  -0.834775   0.893535   -0.93
## factor(time)1.51416666666667   0.248172   0.788670    0.31
## factor(time)1.51666666666667  -1.936565   0.839056   -2.31
## factor(time)1.51916666666667  -3.133959   0.859802   -3.64
## factor(time)1.53                0.508131   0.845421    0.60
## factor(time)1.53083333333333   0.105473   0.889195    0.12
## factor(time)1.53333333333333  -0.133904   0.576054   -0.23
## factor(time)1.54166666666667  -0.908948   0.809510   -1.12
## factor(time)1.59083333333333  -1.281269   0.874578   -1.47
## factor(time)1.615              -0.729764   0.828690   -0.88
## factor(time)1.62916666666667   3.592710   0.800623    4.49
## factor(time)1.64833333333333  -1.521420   0.847647   -1.79
## factor(time)1.71666666666667   0.004166   0.843995    0.00
## factor(time)1.725              -0.376405   0.788670   -0.48
## factor(time)1.81166666666667   0.347990   0.845421    0.41
## factor(time)1.89666666666667  -0.408127   0.824752   -0.49
## factor(time)1.90833333333333  -0.732068   0.857006   -0.85
## factor(time)1.93833333333333  -0.883644   0.944335   -0.94
```

```
##
## Correlation matrix not shown by default, as p = 403 > 12.
## Use print(x, correlation=TRUE)  or
##   vcov(x)     if you need it
```

12. Compare the results of these models both numerically and graphically.

```
anova(reg.3,hiv_reg_vslope,hiv_reg,reg.2)
```

```
## refitting model(s) with ML (instead of REML)

## Data: hiv.data
## Models:
## reg.2: y ~ time + (1 | newpid)
## reg.3: y ~ time + treatment + age.baseline + (1 | newpid)
## hiv_reg_vslope: y ~ time + factor(treatment) + age.baseline + (1 + time | newpid)
## hiv_reg: y ~ factor(time) + (1 | newpid)
##                 Df    AIC    BIC  logLik deviance    Chisq Chi Df
## reg.2            4 3141.9 3161.8 -1566.9   3133.9
## reg.3            6 3136.1 3165.9 -1562.0   3124.1   9.7956      2
## hiv_reg_vslope   8 3110.3 3150.1 -1547.1   3094.3  29.7893      2
## hiv_reg        405 3244.5 5260.3 -1217.3   2434.5 659.7525    397
##                Pr(>Chisq)
## reg.2
## reg.3            0.007463 **
## hiv_reg_vslope  3.399e-07 ***
## hiv_reg         2.261e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Figure skate in the 1932 Winter Olympics

The folder olympics has seven judges' ratings of seven figure skaters (on two criteria: "technical merit" and "artistic impression") from the 1932 Winter Olympics. Take a look at http://www.stat.columbia.edu/~gelman/arm/examples/olympics/olympics1932.txt

1. Construct a $7 \times 7 \times 2$ array of the data (ordered by skater, judge, and judging criterion).

```
arr.olym <- melt(data = olympics1932,id.vars=c("pair","criterion"),
               measure.vars=c(colnames(olympics1932)[3:9]))
arr.olym
```

```
##    pair   criterion variable value
## 1     1     Program  judge_1   5.6
## 2     1 Performance  judge_1   5.6
## 3     2     Program  judge_1   5.5
## 4     2 Performance  judge_1   5.5
## 5     3     Program  judge_1   6.0
## 6     3 Performance  judge_1   6.0
## 7     4     Program  judge_1   5.6
## 8     4 Performance  judge_1   5.6
## 9     5     Program  judge_1   5.4
## 10    5 Performance  judge_1   4.8
## 11    6     Program  judge_1   5.2
## 12    6 Performance  judge_1   4.8
## 13    7     Program  judge_1   4.8
## 14    7 Performance  judge_1   4.3
## 15    1     Program  judge_2   5.5
## 16    1 Performance  judge_2   5.5
## 17    2     Program  judge_2   5.2
## 18    2 Performance  judge_2   5.7
## 19    3     Program  judge_2   5.3
## 20    3 Performance  judge_2   5.5
## 21    4     Program  judge_2   5.3
## 22    4 Performance  judge_2   5.3
## 23    5     Program  judge_2   4.5
## 24    5 Performance  judge_2   4.8
## 25    6     Program  judge_2   5.1
## 26    6 Performance  judge_2   5.6
## 27    7     Program  judge_2   4.0
## 28    7 Performance  judge_2   4.6
## 29    1     Program  judge_3   5.8
## 30    1 Performance  judge_3   5.8
## 31    2     Program  judge_3   5.8
## 32    2 Performance  judge_3   5.6
## 33    3     Program  judge_3   5.8
## 34    3 Performance  judge_3   5.7
## 35    4     Program  judge_3   5.8
## 36    4 Performance  judge_3   5.8
## 37    5     Program  judge_3   5.8
## 38    5 Performance  judge_3   5.5
## 39    6     Program  judge_3   5.3
## 40    6 Performance  judge_3   5.0
## 41    7     Program  judge_3   4.7
## 42    7 Performance  judge_3   4.5
## 43    1     Program  judge_4   5.3
## 44    1 Performance  judge_4   4.7
## 45    2     Program  judge_4   5.8
## 46    2 Performance  judge_4   5.4
## 47    3     Program  judge_4   5.0
## 48    3 Performance  judge_4   4.9
## 49    4     Program  judge_4   4.4
## 50    4 Performance  judge_4   4.8
## 51    5     Program  judge_4   4.0
## 52    5 Performance  judge_4   4.4
## 53    6     Program  judge_4   5.4
## 54    6 Performance  judge_4   4.7
## 55    7     Program  judge_4   4.0
## 56    7 Performance  judge_4   4.0
## 57    1     Program  judge_5   5.6
## 58    1 Performance  judge_5   5.7
## 59    2     Program  judge_5   5.6
## 60    2 Performance  judge_5   5.5
## 61    3     Program  judge_5   5.4
## 62    3 Performance  judge_5   5.5
## 63    4     Program  judge_5   4.5
## 64    4 Performance  judge_5   4.5
## 65    5     Program  judge_5   5.5
## 66    5 Performance  judge_5   4.6
## 67    6     Program  judge_5   4.5
## 68    6 Performance  judge_5   4.0
## 69    7     Program  judge_5   3.7
## 70    7 Performance  judge_5   3.6
## 71    1     Program  judge_6   5.2
## 72    1 Performance  judge_6   5.3
## 73    2     Program  judge_6   5.1
## 74    2 Performance  judge_6   5.3
## 75    3     Program  judge_6   5.1
## 76    3 Performance  judge_6   5.2
## 77    4     Program  judge_6   5.0
## 78    4 Performance  judge_6   5.0
## 79    5     Program  judge_6   4.8
## 80    5 Performance  judge_6   4.8
## 81    6     Program  judge_6   4.5
## 82    6 Performance  judge_6   4.6
## 83    7     Program  judge_6   4.0
## 84    7 Performance  judge_6   4.0
## 85    1     Program  judge_7   5.7
## 86    1 Performance  judge_7   5.4
## 87    2     Program  judge_7   5.8
## 88    2 Performance  judge_7   5.7
## 89    3     Program  judge_7   5.3
## 90    3 Performance  judge_7   5.7
## 91    4     Program  judge_7   5.1
## 92    4 Performance  judge_7   5.5
## 93    5     Program  judge_7   5.5
## 94    5 Performance  judge_7   5.2
## 95    6     Program  judge_7   5.0
## 96    6 Performance  judge_7   5.2
## 97    7     Program  judge_7   4.8
## 98    7 Performance  judge_7   4.8
```

2. Reformulate the data as a 98 × 4 array (similar to the top table in Figure 11.7), where the first two columns are the technical merit and artistic impression scores, the third column is a skater ID, and the fourth column is a judge ID.

```
library("plyr")
```

```
## ------------------------------------------------------------------------------

## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:
## library(plyr); library(dplyr)

## ------------------------------------------------------------------------------

##
## Attaching package: 'plyr'

## The following objects are masked from 'package:dplyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize

## The following objects are masked from 'package:reshape':
##
##     rename, round_any
```

```
d <- data.frame(alpha=1:3, beta=4:6, gamma=7:9)
plyr::rename(d, c("beta"="two", "gamma"="three"))
```

```
##   alpha two three
## 1     1   4     7
## 2     2   5     8
## 3     3   6     9
```

```
olym.984 <- rename(arr.olym, c("pair"="skater_ID", "criterion"="criterion", "variable"="judge_ID","valu
olym.984 <- olym.984[order(olym.984$judge_ID),]
olym.984 <- olym.984[c("criterion", "value", "skater_ID", "judge_ID")]
```

3. Add another column to this matrix representing an indicator variable that equals 1 if the skater and judge are from the same country, or 0 otherwise.

```
olym.984$SameCountry <-ifelse(olym.984[,3] == " 1"&olym.984[,4] == "judge_5",1,
  ifelse(olym.984[,3] == " 2"&olym.984[,4] == "judge_7",1,
  ifelse(olym.984[,3] == " 3"&olym.984[,4] == "judge_1",1,
  ifelse(olym.984[,3] == " 4"&olym.984[,4] == "judge_1",1,
  ifelse(olym.984[,3] == " 7"&olym.984[,4] == "judge_7",1,0
  )))))
```

4. Write the notation for a non-nested multilevel model (varying across skaters and judges) for the technical merit ratings and fit using lmer().

```
data.tech <- olym.984 %>%
  filter(criterion=="Program")
data.art <- olym.984 %>%
  filter(criterion=="Performance")

reg.tech <- lmer(value ~ 1 + (1|skater_ID) + (1|judge_ID),data=data.tech)
summary(reg.tech)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: value ~ 1 + (1 | skater_ID) + (1 | judge_ID)
##    Data: data.tech
##
## REML criterion at convergence: 60
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -2.51025 -0.45646 -0.05459  0.63866  1.89709
##
## Random effects:
##  Groups    Name        Variance Std.Dev.
##  skater_ID (Intercept) 0.17488  0.4182
##  judge_ID  (Intercept) 0.07664  0.2768
##  Residual              0.11057  0.3325
## Number of obs: 49, groups:  skater_ID, 7; judge_ID, 7
##
## Fixed effects:
##             Estimate Std. Error t value
## (Intercept)   5.1347     0.1954   26.28
```

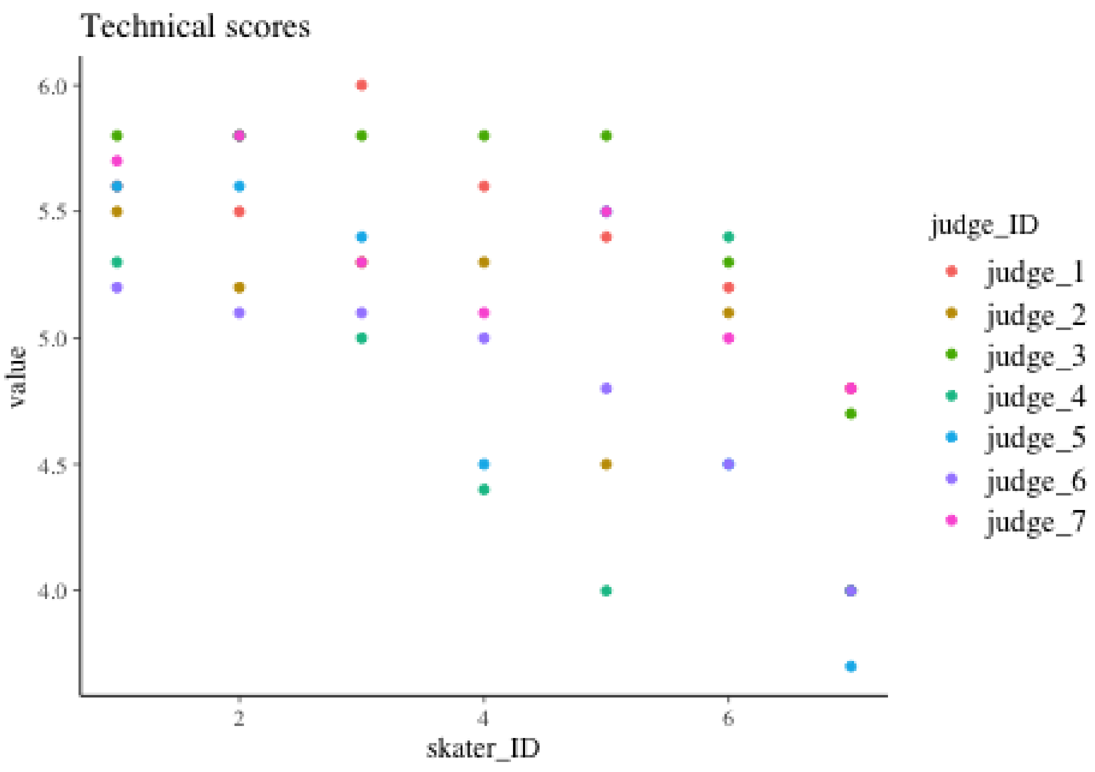
5. Fit the model in (4) using the artistic impression ratings.

```r
reg.art <- lmer(value ~ 1 + (1|skater_ID) + (1|judge_ID),data=data.art)
summary(reg.art)
```

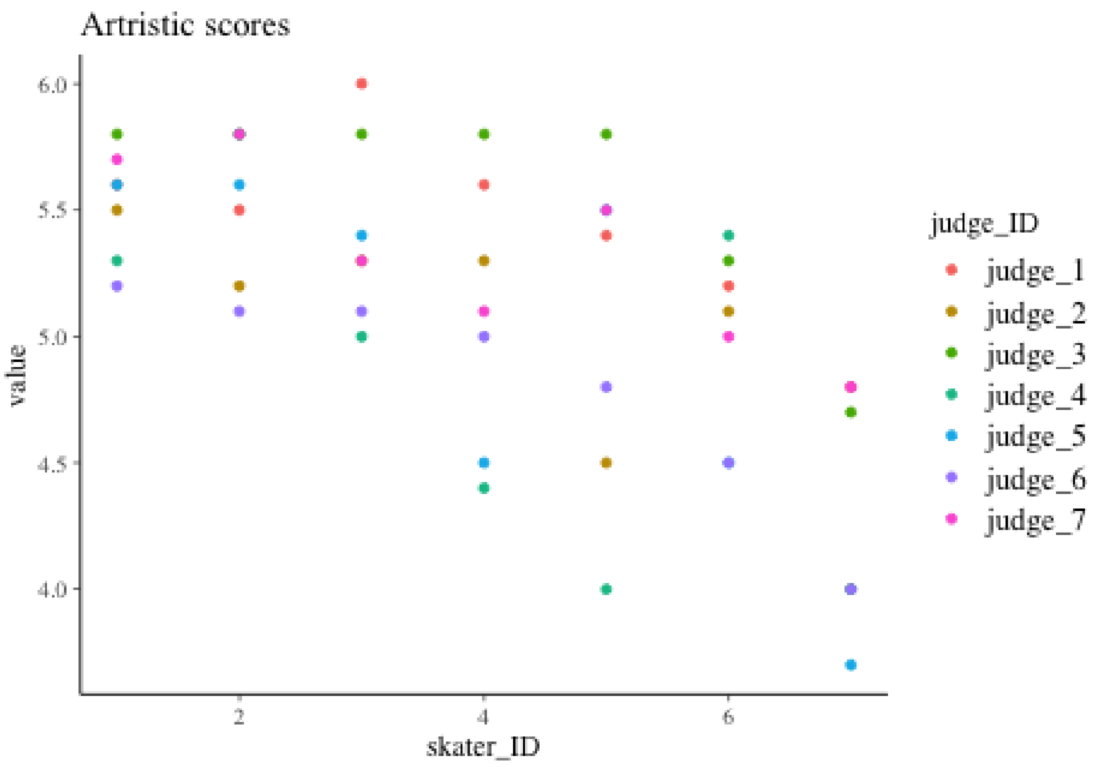
```
## Linear mixed model fit by REML ['lmerMod']
## Formula: value ~ 1 + (1 | skater_ID) + (1 | judge_ID)
##    Data: data.art
## 
## REML criterion at convergence: 46.2
## 
## Scaled residuals: 
##      Min       1Q   Median       3Q      Max 
## -2.10128 -0.50469 -0.09884  0.40875  2.10489 
## 
## Random effects:
##  Groups    Name        Variance Std.Dev.
##  skater_ID (Intercept) 0.20486  0.4526  
##  judge_ID  (Intercept) 0.07759  0.2785  
##  Residual              0.07446  0.2729  
## Number of obs: 49, groups:  skater_ID, 7; judge_ID, 7
## 
## Fixed effects:
##             Estimate Std. Error t value
## (Intercept)   5.0918     0.2046   24.88
```

6. Display your results for both outcomes graphically.

```r
ggplot(data.tech,aes(x=skater_ID,y=value,color=judge_ID))+geom_point()+
  ggtitle("Technical scores")
```

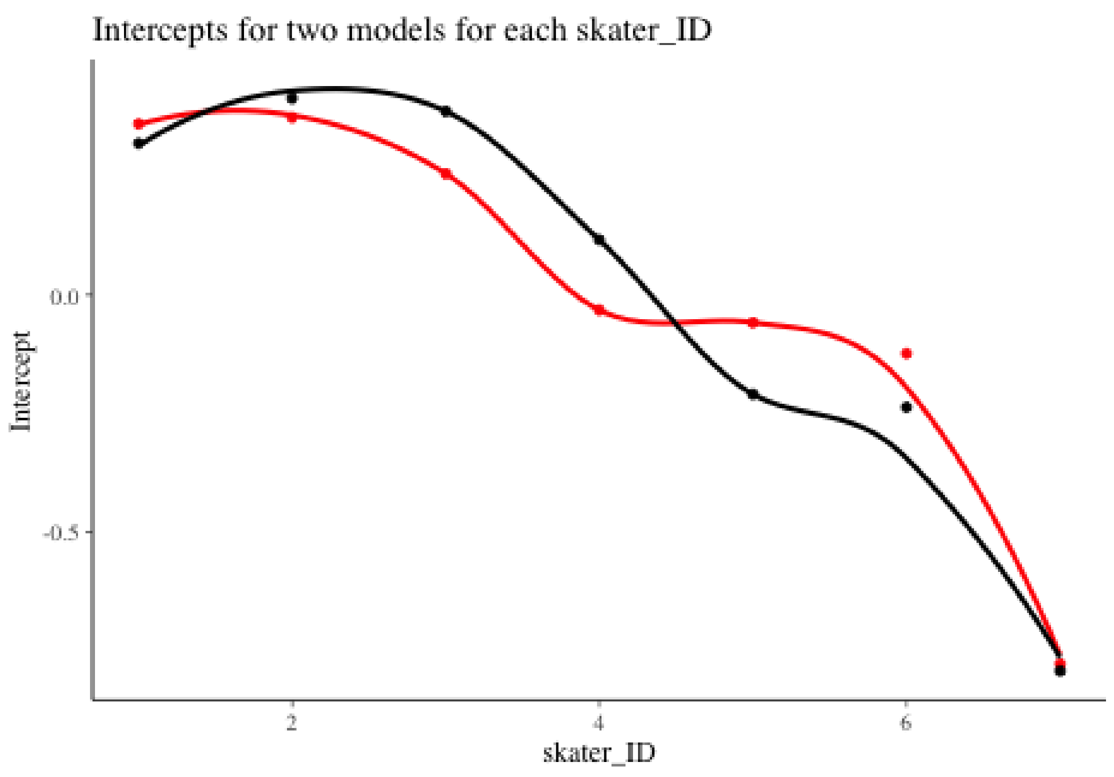


```r
ggplot(data.tech,aes(x=skater_ID,y=value,color=judge_ID))+geom_point()+
  ggtitle("Artristic scores")
```



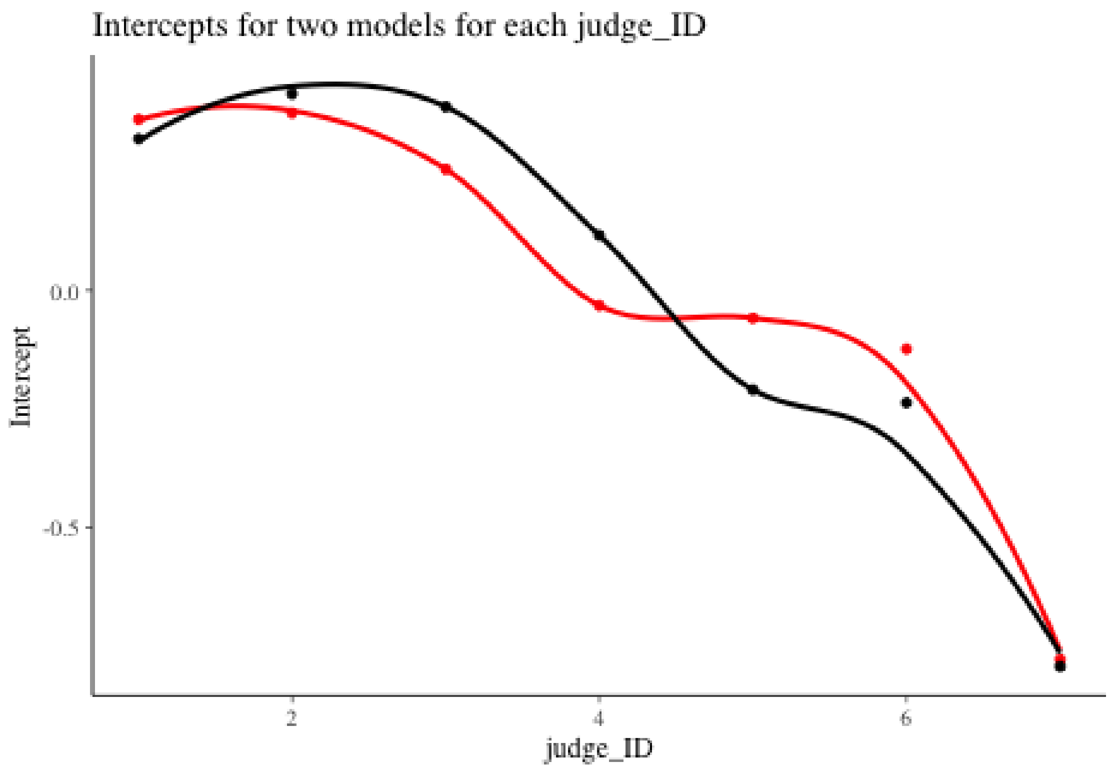
```r
#A plot displaying Intercepts for two models for each skater_ID
inter.skate <- as.data.frame(cbind(unlist(ranef(reg.tech))[1:7],unlist(ranef(reg.art))[1:7]))
inter.skate$skater_ID <-c(1:7)
ggplot(data=inter.skate)+
  geom_point(col="red",aes(x=skater_ID,y=V1))+geom_smooth(col="red",aes(x=skater_ID,y=V1),se=FALSE)+
  geom_point(col="black",aes(x=skater_ID,y=V2))+geom_smooth(col="black",aes(x=skater_ID,y=V2),se=FALSE)
  ggtitle("Intercepts for two models for each skater_ID")+
  ylab("Intercept")
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



```r
##A plot displaying Intercepts for two models for each judge_ID
inter.judge <- as.data.frame(cbind(unlist(ranef(reg.tech))[1:7],unlist(ranef(reg.art))[1:7]))
inter.judge$judge_ID <-c(1:7)
ggplot(data=inter.judge)+
  geom_point(col="red",aes(x=judge_ID,y=V1))+geom_smooth(col="red",aes(x=judge_ID,y=V1),se=FALSE)+
  geom_point(col="black",aes(x=judge_ID,y=V2))+geom_smooth(col="black",aes(x=judge_ID,y=V2),se=FALSE)+
  ggtitle("Intercepts for two models for each judge_ID")+
  ylab("Intercept")
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



7. (optional) Use posterior predictive checks to investigate model fit in (4) and (5).

Different ways to write the model:

Using any data that are appropriate for a multilevel model, write the model in the five ways discussed in Section 12.5 of Gelman and Hill.

1st method: Allowing regression coefficeints to vary accross groups

$y = 4.91 + X_{itime} * (-0.36) + X_{itreatment} * (-0.12) + X_{iage.base} * 0.18 + 0.77$

$\alpha_j \sim$ N$(0, 1.37^2)$

2nd method: Combining separate local regressions

$y \sim N(4.91 + X_{itime} * (-0.36) + X_{itreatment} * (-0.12) + X_{iage.base} * (0.18), 0.77^2)$

$\alpha_j \sim$ N$(RandomIntercept, 1.37^2)$

3rd method: Modeling the coefficients of a large regression model

$y_i \sim N(4.91 + X_{itime} * (-0.36) + X_{itreatment} * (-0.12) + X_{iage.base} * (0.18), 0.77^2)$

$\beta_j \sim N(0, 1.37^2)$

4th method: Regression with multiple error terms

$y_i \sim \ N(4.91 + X_{itime} * (-0.36) + X_{itreatment} * (-0.12) + X_{iage.base} * (0.18) + 1.37^2, 0.77^2)$

5th method: Large regression with correlated errors

$y_i \sim \ N(4.91 + X_{itime} * (-0.36) + X_{itreatment} * (-0.12) + X_{iage.base} * (0.18), 1.37^2 + 0.77^2)$

Models for adjusting individual ratings:

A committee of 10 persons is evaluating 100 job applications. Each person on the committee reads 30 applications (structured so that each application is read by three people) and gives each a numerical rating between 1 and 10.

1. It would be natural to rate the applications based on their combined scores; however, there is a worry that different raters use different standards, and we would like to correct for this. Set up a model for the ratings (with parameters for the applicants and the raters). $y_{score} = \alpha_{j[i]} + \beta_{cadidate} X_{iCadidate} + \beta_{rater} X_{iRater} + U_{RandomEffect-Rater}$
2. It is possible that some persons on the committee show more variation than others in their ratings. Expand your model to allow for this. lmer(rating~applicants+raters+(1+raters|raters))