[STAT 4400] HW-6

Michael Ghattas

4/6/2022

# Problem 1

library(lme4)  
library(lmerTest)

##   
## Attaching package: 'lmerTest'

## The following object is masked from 'package:lme4':  
##   
## lmer

## The following object is masked from 'package:stats':  
##   
## step

library(extraoperators)  
library(JWileymisc)

##   
## Attaching package: 'JWileymisc'

## The following object is masked from 'package:rstanarm':  
##   
## R2

library(multilevelTools)  
  
  
df <- read.csv(file = "/Users/Home/Documents/Michael\_Ghattas/School/CU\_Boulder/2022/Spring 2022/STAT - 4400/Data/ProfEvaltnsBeautyPublic.csv")  
head(df)

## tenured profnumber minority age beautyf2upper beautyflowerdiv beautyfupperdiv  
## 1 0 1 1 36 6 5 7  
## 2 1 2 0 59 2 4 4  
## 3 1 3 0 51 5 5 2  
## 4 1 4 0 40 4 2 5  
## 5 0 5 0 31 9 7 9  
## 6 1 6 0 62 5 6 6  
## beautym2upper beautymlowerdiv beautymupperdiv btystdave btystdf2u  
## 1 6 2 4 0.2015666 0.2893519  
## 2 3 2 3 -0.8260813 -1.6193560  
## 3 3 2 3 -0.6603327 -0.1878249  
## 4 2 3 3 -0.7663125 -0.6650018  
## 5 6 7 6 1.4214450 1.7208830  
## 6 6 5 5 0.5002196 -0.1878249  
## btystdfl btystdfu btystdm2u btystdml btystdmu class1 class2 class3  
## 1 0.4580018 0.8758139 0.6817153 -0.9000649 -0.1954181 0 0 1  
## 2 -0.0735065 -0.5770065 -1.1319040 -0.9000649 -0.6546507 0 0 0  
## 3 0.4580018 -1.5455530 -1.1319040 -0.9000649 -0.6546507 0 0 0  
## 4 -1.1365230 -0.0927330 -1.7364440 -0.3125226 -0.6546507 0 1 0  
## 5 1.5210190 1.8443610 0.6817153 2.0376470 0.7230470 0 0 0  
## 6 0.9895102 0.3915404 0.6817153 0.8625621 0.2638144 0 0 0  
## class4 class5 class6 class7 class8 class9 class10 class11 class12 class13  
## 1 0 0 0 0 0 0 0 0 0 0  
## 2 0 0 0 0 0 0 0 0 0 0  
## 3 1 0 0 0 0 0 0 0 0 0  
## 4 0 0 0 0 0 0 0 0 0 0  
## 5 0 0 0 0 0 0 0 0 0 0  
## 6 0 0 0 0 0 0 0 0 0 0  
## class14 class15 class16 class17 class18 class19 class20 class21 class22  
## 1 0 0 0 0 0 0 0 0 0  
## 2 0 0 0 0 0 0 0 0 0  
## 3 0 0 0 0 0 0 0 0 0  
## 4 0 0 0 0 0 0 0 0 0  
## 5 0 0 0 0 0 0 0 0 0  
## 6 0 0 0 0 0 0 0 0 0  
## class23 class24 class25 class26 class27 class28 class29 class30  
## 1 0 0 0 0 0 0 0 0  
## 2 0 0 0 0 0 0 0 0  
## 3 0 0 0 0 0 0 0 0  
## 4 0 0 0 0 0 0 0 0  
## 5 0 0 0 0 0 0 0 0  
## 6 0 0 0 0 0 0 0 0  
## courseevaluation didevaluation female formal fulldept lower multipleclass  
## 1 4.3 24 1 0 1 0 1  
## 2 4.5 17 0 0 1 0 0  
## 3 3.7 55 0 0 1 0 1  
## 4 4.3 40 1 0 1 0 1  
## 5 4.4 42 1 0 1 0 0  
## 6 4.2 182 0 1 1 0 0  
## nonenglish onecredit percentevaluating profevaluation students tenuretrack  
## 1 0 0 55.81395 4.7 43 1  
## 2 0 0 85.00000 4.6 20 1  
## 3 0 0 100.00000 4.1 55 1  
## 4 0 0 86.95652 4.5 46 1  
## 5 0 0 87.50000 4.8 48 1  
## 6 0 0 64.53901 4.4 282 1  
## blkandwhite btystdvariance btystdavepos btystdaveneg  
## 1 0 2.1298060 0.201567 0.000000  
## 2 0 1.3860810 0.000000 -0.826081  
## 3 0 2.5374350 0.000000 -0.660333  
## 4 0 1.7605770 0.000000 -0.766312  
## 5 0 1.6931000 1.421450 0.000000  
## 6 0 0.9447419 0.500220 0.000000

courses <- data.frame(df[,19:48])  
n <- nrow (df)  
J <- ncol (courses) + 1  
course.id <- rep (0, n)  
for (i in 1:n){  
 for (j in 1:30){  
 if (courses[i,j]==1) course.id[i] <- j  
 }  
}  
  
head(df)

## tenured profnumber minority age beautyf2upper beautyflowerdiv beautyfupperdiv  
## 1 0 1 1 36 6 5 7  
## 2 1 2 0 59 2 4 4  
## 3 1 3 0 51 5 5 2  
## 4 1 4 0 40 4 2 5  
## 5 0 5 0 31 9 7 9  
## 6 1 6 0 62 5 6 6  
## beautym2upper beautymlowerdiv beautymupperdiv btystdave btystdf2u  
## 1 6 2 4 0.2015666 0.2893519  
## 2 3 2 3 -0.8260813 -1.6193560  
## 3 3 2 3 -0.6603327 -0.1878249  
## 4 2 3 3 -0.7663125 -0.6650018  
## 5 6 7 6 1.4214450 1.7208830  
## 6 6 5 5 0.5002196 -0.1878249  
## btystdfl btystdfu btystdm2u btystdml btystdmu class1 class2 class3  
## 1 0.4580018 0.8758139 0.6817153 -0.9000649 -0.1954181 0 0 1  
## 2 -0.0735065 -0.5770065 -1.1319040 -0.9000649 -0.6546507 0 0 0  
## 3 0.4580018 -1.5455530 -1.1319040 -0.9000649 -0.6546507 0 0 0  
## 4 -1.1365230 -0.0927330 -1.7364440 -0.3125226 -0.6546507 0 1 0  
## 5 1.5210190 1.8443610 0.6817153 2.0376470 0.7230470 0 0 0  
## 6 0.9895102 0.3915404 0.6817153 0.8625621 0.2638144 0 0 0  
## class4 class5 class6 class7 class8 class9 class10 class11 class12 class13  
## 1 0 0 0 0 0 0 0 0 0 0  
## 2 0 0 0 0 0 0 0 0 0 0  
## 3 1 0 0 0 0 0 0 0 0 0  
## 4 0 0 0 0 0 0 0 0 0 0  
## 5 0 0 0 0 0 0 0 0 0 0  
## 6 0 0 0 0 0 0 0 0 0 0  
## class14 class15 class16 class17 class18 class19 class20 class21 class22  
## 1 0 0 0 0 0 0 0 0 0  
## 2 0 0 0 0 0 0 0 0 0  
## 3 0 0 0 0 0 0 0 0 0  
## 4 0 0 0 0 0 0 0 0 0  
## 5 0 0 0 0 0 0 0 0 0  
## 6 0 0 0 0 0 0 0 0 0  
## class23 class24 class25 class26 class27 class28 class29 class30  
## 1 0 0 0 0 0 0 0 0  
## 2 0 0 0 0 0 0 0 0  
## 3 0 0 0 0 0 0 0 0  
## 4 0 0 0 0 0 0 0 0  
## 5 0 0 0 0 0 0 0 0  
## 6 0 0 0 0 0 0 0 0  
## courseevaluation didevaluation female formal fulldept lower multipleclass  
## 1 4.3 24 1 0 1 0 1  
## 2 4.5 17 0 0 1 0 0  
## 3 3.7 55 0 0 1 0 1  
## 4 4.3 40 1 0 1 0 1  
## 5 4.4 42 1 0 1 0 0  
## 6 4.2 182 0 1 1 0 0  
## nonenglish onecredit percentevaluating profevaluation students tenuretrack  
## 1 0 0 55.81395 4.7 43 1  
## 2 0 0 85.00000 4.6 20 1  
## 3 0 0 100.00000 4.1 55 1  
## 4 0 0 86.95652 4.5 46 1  
## 5 0 0 87.50000 4.8 48 1  
## 6 0 0 64.53901 4.4 282 1  
## blkandwhite btystdvariance btystdavepos btystdaveneg  
## 1 0 2.1298060 0.201567 0.000000  
## 2 0 1.3860810 0.000000 -0.826081  
## 3 0 2.5374350 0.000000 -0.660333  
## 4 0 1.7605770 0.000000 -0.766312  
## 5 0 1.6931000 1.421450 0.000000  
## 6 0 0.9447419 0.500220 0.000000

### (a)

, for

### (b)

M1 <- lmer (courseevaluation ~ profevaluation + (1 + profevaluation | course.id) + students + (1 + students | course.id) + tenuretrack + (1 + tenuretrack | course.id) + tenured + (1 + tenured | course.id) + percentevaluating + (1 + percentevaluating | course.id), data = df)

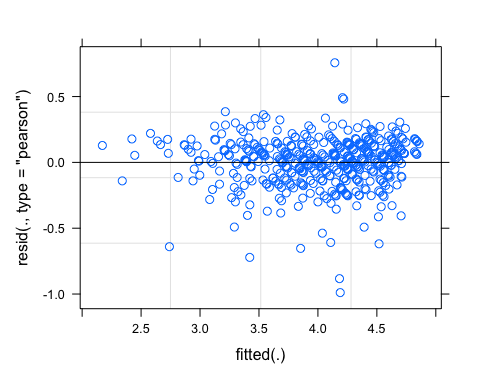
## boundary (singular) fit: see help('isSingular')

summary(M1)

## Linear mixed model fit by REML. t-tests use Satterthwaite's method [  
## lmerModLmerTest]  
## Formula:   
## courseevaluation ~ profevaluation + (1 + profevaluation | course.id) +   
## students + (1 + students | course.id) + tenuretrack + (1 +   
## tenuretrack | course.id) + tenured + (1 + tenured | course.id) +   
## percentevaluating + (1 + percentevaluating | course.id)  
## Data: df  
##   
## REML criterion at convergence: -158.4  
##   
## Scaled residuals:   
## Min 1Q Median 3Q Max   
## -5.1976 -0.5388 0.1175 0.6546 3.9758   
##   
## Random effects:  
## Groups Name Variance Std.Dev. Corr   
## course.id (Intercept) 5.199e-02 2.280e-01   
## profevaluation 2.345e-03 4.843e-02 -1.00  
## course.id.1 (Intercept) 8.580e-07 9.263e-04   
## students 9.228e-12 3.038e-06 -1.00  
## course.id.2 (Intercept) 2.396e-05 4.895e-03   
## tenuretrack 3.947e-05 6.282e-03 -1.00  
## course.id.3 (Intercept) 2.429e-03 4.929e-02   
## tenured 1.614e-03 4.017e-02 -1.00  
## course.id.4 (Intercept) 1.889e-02 1.374e-01   
## percentevaluating 2.221e-06 1.490e-03 -1.00  
## Residual 3.622e-02 1.903e-01   
## Number of obs: 463, groups: course.id, 30  
##   
## Fixed effects:  
## Estimate Std. Error df t value Pr(>|t|)   
## (Intercept) -9.450e-02 1.297e-01 4.402e+00 -0.729 0.5031   
## profevaluation 9.485e-01 2.736e-02 3.192e+00 34.673 3.19e-05 \*\*\*  
## students -6.214e-05 1.302e-04 1.684e+02 -0.477 0.6339   
## tenuretrack -6.614e-02 2.765e-02 7.418e+00 -2.392 0.0461 \*   
## tenured 5.425e-02 2.939e-02 5.508e+00 1.845 0.1189   
## percentevaluating 1.981e-03 8.925e-04 3.473e+00 2.220 0.1009   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Correlation of Fixed Effects:  
## (Intr) prfvlt stdnts tnrtrc tenurd  
## profevalutn -0.801   
## students -0.058 -0.082   
## tenuretrack -0.188 0.113 -0.057   
## tenured 0.069 -0.169 -0.079 -0.442   
## percntvltng -0.385 -0.198 0.194 -0.043 -0.010  
## optimizer (nloptwrap) convergence code: 0 (OK)  
## boundary (singular) fit: see help('isSingular')

### (c)

plot(M1)



# Problem 2

### (a)

I <- 100L  
J <- 10L  
W <- 3L  
tau <- 2  
sigma <- 1  
  
assignment <- matrix(0L,I,J)  
for (i in 1L:I) {  
 workload <- colSums(assignment)  
 available <- which (workload < W\*I/J)  
 if (i > 75L)  
 cat("Round ",i,": available = ",  
 paste(available,collapse=", "),"\n")  
 while (length(available) < W) {  
 slacker <- which.min(workload)  
 pswaps <- which(!assignment[1L:(i-1L),slacker])  
 swaprow <- sample(pswaps,1L)  
 swapcol <- sample(which(as.logical(assignment[swaprow,])),1L)  
 assignment[swaprow,swapcol] <- 0L  
 assignment[swaprow,slacker] <- 1L  
 workload <- colSums(assignment)  
 available <- which(workload < W\*I/J)  
 cat("Round ",i,"x: availble=",paste(available,collapse=", "),  
 "\n")  
 }   
 assignment[i,sample(available,W)] <- 1L  
}

## Round 76 : available = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10   
## Round 77 : available = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10   
## Round 78 : available = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10   
## Round 79 : available = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10   
## Round 80 : available = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10   
## Round 81 : available = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10   
## Round 82 : available = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10   
## Round 83 : available = 1, 2, 3, 4, 5, 6, 7, 8, 10   
## Round 84 : available = 1, 2, 3, 4, 5, 6, 7, 8, 10   
## Round 85 : available = 1, 2, 3, 4, 5, 6, 7, 8, 10   
## Round 86 : available = 1, 2, 3, 4, 5, 6, 7, 8, 10   
## Round 87 : available = 1, 2, 3, 4, 5, 6, 7, 8, 10   
## Round 88 : available = 1, 2, 3, 4, 5, 6, 7, 8, 10   
## Round 89 : available = 1, 3, 4, 5, 6, 7, 8, 10   
## Round 90 : available = 1, 3, 4, 5, 7, 8, 10   
## Round 91 : available = 1, 4, 5, 7, 8, 10   
## Round 92 : available = 1, 4, 7, 8, 10   
## Round 93 : available = 1, 4, 7, 8, 10   
## Round 94 : available = 4, 7, 8, 10   
## Round 95 : available = 4, 7, 8, 10   
## Round 96 : available = 4, 7, 8, 10   
## Round 97 : available = 4, 7, 8, 10   
## Round 98 : available = 4, 7, 8, 10   
## Round 99 : available = 4, 7, 10   
## Round 100 : available = 4, 7   
## Round 100 x: availble= 4, 6, 7

colSums(assignment)

## [1] 30 30 30 30 30 30 30 30 30 30

rowSums(assignment)

## [1] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3  
## [38] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3  
## [75] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3

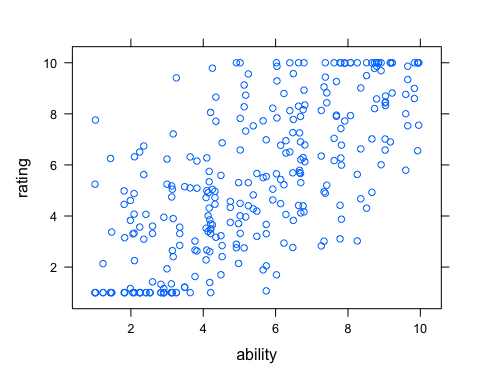
write.csv(assignment,"assignment.csv")  
  
ability <- runif(I,1,10)  
severity <- rnorm(J,0,tau)  
applicant <- rep(1L:I,each=W)  
rater <-   
 sapply(1L:I,  
 function (i)   
 which(as.logical(assignment[i,])))  
str(rater)

## int [1:3, 1:100] 3 9 10 2 6 10 3 5 7 3 ...

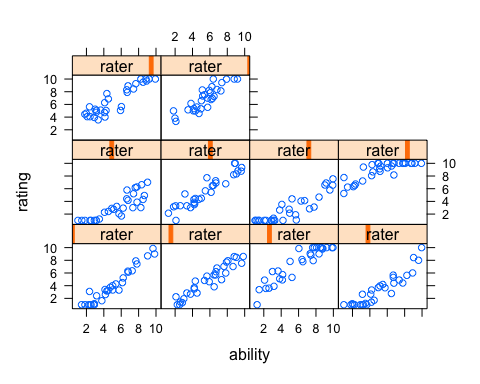
rating <- ability[applicant] + severity[rater] + rnorm(I\*W,0,sigma)  
rating <- pmax(1,pmin(rating,10))  
ratings.df <- data.frame(applicant=applicant, rater=as.vector(rater), rating=rating)  
ratings.df

## applicant rater rating  
## 1 1 3 4.877936  
## 2 1 9 4.071667  
## 3 1 10 3.325303  
## 4 2 2 4.722358  
## 5 2 6 4.459272  
## 6 2 10 4.966620  
## 7 3 3 9.014091  
## 8 3 5 6.626949  
## 9 3 7 4.674131  
## 10 4 3 10.000000  
## 11 4 5 4.306151  
## 12 4 9 9.498074  
## 13 5 1 8.698292  
## 14 5 2 8.464416  
## 15 5 6 8.325377  
## 16 6 1 1.000000  
## 17 6 5 1.000000  
## 18 6 9 4.064592  
## 19 7 1 6.126157  
## 20 7 2 6.782644  
## 21 7 10 8.346936  
## 22 8 3 5.744595  
## 23 8 6 4.310485  
## 24 8 10 5.341055  
## 25 9 4 6.006205  
## 26 9 6 10.000000  
## 27 9 9 9.688982  
## 28 10 2 3.490032  
## 29 10 4 3.317153  
## 30 10 8 8.055180  
## 31 11 4 2.668839  
## 32 11 9 6.275555  
## 33 11 10 4.991605  
## 34 12 2 4.947048  
## 35 12 3 9.053500  
## 36 12 5 3.011960  
## 37 13 1 3.089924  
## 38 13 8 6.741440  
## 39 13 9 5.622164  
## 40 14 1 1.000000  
## 41 14 5 1.000000  
## 42 14 6 3.149740  
## 43 15 3 3.371135  
## 44 15 5 1.000000  
## 45 15 7 1.000000  
## 46 16 7 6.901227  
## 47 16 8 10.000000  
## 48 16 10 10.000000  
## 49 17 6 3.654946  
## 50 17 8 9.787064  
## 51 17 9 5.041957  
## 52 18 3 6.313457  
## 53 18 4 1.000000  
## 54 18 9 5.100058  
## 55 19 2 7.018081  
## 56 19 3 10.000000  
## 57 19 5 4.925819  
## 58 20 5 2.895494  
## 59 20 7 2.756140  
## 60 20 8 10.000000  
## 61 21 1 1.160127  
## 62 21 4 1.000000  
## 63 21 5 1.000000  
## 64 22 3 5.088728  
## 65 22 7 2.631927  
## 66 22 10 6.151714  
## 67 23 2 3.827981  
## 68 23 3 7.779316  
## 69 23 8 9.574754  
## 70 24 1 1.000000  
## 71 24 2 1.000000  
## 72 24 7 1.000000  
## 73 25 1 1.000000  
## 74 25 7 1.000000  
## 75 25 9 3.949869  
## 76 26 4 1.000000  
## 77 26 9 4.455132  
## 78 26 10 4.983554  
## 79 27 1 3.166579  
## 80 27 8 8.654203  
## 81 27 9 7.707843  
## 82 28 5 4.412355  
## 83 28 8 10.000000  
## 84 28 9 8.301644  
## 85 29 4 5.663889  
## 86 29 6 6.901654  
## 87 29 9 7.896750  
## 88 30 3 3.566951  
## 89 30 4 1.000000  
## 90 30 8 6.506288  
## 91 31 2 4.627617  
## 92 31 9 5.050037  
## 93 31 10 8.219304  
## 94 32 4 8.003881  
## 95 32 6 8.763371  
## 96 32 7 5.793486  
## 97 33 1 5.225754  
## 98 33 5 2.938811  
## 99 33 10 8.795114  
## 100 34 2 2.252691  
## 101 34 6 1.000000  
## 102 34 8 6.319328  
## 103 35 2 5.692678  
## 104 35 4 2.766224  
## 105 35 10 7.264668  
## 106 36 1 6.285386  
## 107 36 4 4.395731  
## 108 36 8 8.163796  
## 109 37 4 2.645935  
## 110 37 7 1.000000  
## 111 37 9 5.048189  
## 112 38 1 6.346422  
## 113 38 7 2.829043  
## 114 38 10 8.132605  
## 115 39 2 5.992719  
## 116 39 5 3.875253  
## 117 39 6 6.775163  
## 118 40 1 4.483837  
## 119 40 6 5.429065  
## 120 40 10 6.767504  
## 121 41 1 7.422351  
## 122 41 3 10.000000  
## 123 41 5 6.269211  
## 124 42 2 7.717812  
## 125 42 8 10.000000  
## 126 42 10 9.975251  
## 127 43 1 2.602405  
## 128 43 4 1.396017  
## 129 43 6 3.847542  
## 130 44 2 8.585591  
## 131 44 3 9.860183  
## 132 44 10 10.000000  
## 133 45 4 1.161482  
## 134 45 9 4.608198  
## 135 45 10 3.828891  
## 136 46 2 6.205867  
## 137 46 4 5.305351  
## 138 46 7 4.121158  
## 139 47 6 10.000000  
## 140 47 8 10.000000  
## 141 47 9 10.000000  
## 142 48 1 3.305042  
## 143 48 5 2.048494  
## 144 48 10 5.544536  
## 145 49 1 3.227400  
## 146 49 4 1.700788  
## 147 49 10 5.589945  
## 148 50 1 3.750554  
## 149 50 6 4.337544  
## 150 50 10 4.569291  
## 151 51 2 5.501920  
## 152 51 6 7.727567  
## 153 51 7 1.890945  
## 154 52 3 9.860161  
## 155 52 8 9.290609  
## 156 52 10 7.840926  
## 157 53 2 7.039270  
## 158 53 3 10.000000  
## 159 53 5 6.169399  
## 160 54 3 8.821588  
## 161 54 5 5.205619  
## 162 54 9 8.432689  
## 163 55 1 9.864400  
## 164 55 2 7.533943  
## 165 55 6 9.342552  
## 166 56 1 4.009182  
## 167 56 6 4.492851  
## 168 56 8 10.000000  
## 169 57 2 1.421900  
## 170 57 3 3.603976  
## 171 57 6 3.316520  
## 172 58 5 1.000000  
## 173 58 7 1.000000  
## 174 58 8 7.757806  
## 175 59 3 10.000000  
## 176 59 6 8.815973  
## 177 59 9 10.000000  
## 178 60 1 4.278819  
## 179 60 2 4.825492  
## 180 60 10 7.530537  
## 181 61 3 10.000000  
## 182 61 4 10.000000  
## 183 61 7 7.554916  
## 184 62 1 7.905044  
## 185 62 3 7.949131  
## 186 62 9 9.265588  
## 187 63 1 8.996774  
## 188 63 2 8.600547  
## 189 63 3 10.000000  
## 190 64 4 1.000000  
## 191 64 5 1.000000  
## 192 64 7 1.344268  
## 193 65 2 1.932005  
## 194 65 3 6.228581  
## 195 65 9 5.244281  
## 196 66 2 7.932318  
## 197 66 3 10.000000  
## 198 66 8 10.000000  
## 199 67 1 3.395659  
## 200 67 3 4.909747  
## 201 67 9 4.009184  
## 202 68 7 6.556078  
## 203 68 8 10.000000  
## 204 68 9 10.000000  
## 205 69 2 5.697878  
## 206 69 5 5.790073  
## 207 69 6 7.253999  
## 208 70 1 1.630763  
## 209 70 2 2.666728  
## 210 70 6 3.018002  
## 211 71 2 2.849599  
## 212 71 6 3.309543  
## 213 71 9 3.563578  
## 214 72 1 2.408196  
## 215 72 3 3.899733  
## 216 72 8 7.215165  
## 217 73 1 1.000000  
## 218 73 2 1.000000  
## 219 73 5 1.000000  
## 220 74 3 1.000000  
## 221 74 6 2.135339  
## 222 74 7 1.000000  
## 223 75 4 4.190676  
## 224 75 5 3.204214  
## 225 75 6 5.663527  
## 226 76 5 4.164806  
## 227 76 8 10.000000  
## 228 76 9 8.927404  
## 229 77 2 2.842228  
## 230 77 5 2.409205  
## 231 77 9 6.866060  
## 232 78 6 6.465609  
## 233 78 7 4.058816  
## 234 78 10 6.949901  
## 235 79 5 1.696127  
## 236 79 8 10.000000  
## 237 79 9 5.652292  
## 238 80 4 5.627440  
## 239 80 5 3.022432  
## 240 80 9 10.000000  
## 241 81 5 1.000000  
## 242 81 6 5.175261  
## 243 81 9 4.744418  
## 244 82 5 2.284002  
## 245 82 7 3.522581  
## 246 82 9 4.719841  
## 247 83 3 5.304573  
## 248 83 7 4.401000  
## 249 83 8 9.562456  
## 250 84 3 8.125913  
## 251 84 6 6.507556  
## 252 84 10 10.000000  
## 253 85 4 1.204182  
## 254 85 5 1.217691  
## 255 85 10 5.148105  
## 256 86 2 3.689308  
## 257 86 6 3.486715  
## 258 86 7 1.000000  
## 259 87 5 2.748991  
## 260 87 8 9.128233  
## 261 87 10 8.278367  
## 262 88 1 1.000000  
## 263 88 2 1.326866  
## 264 88 7 1.000000  
## 265 89 3 9.778086  
## 266 89 6 8.205507  
## 267 89 8 10.000000  
## 268 90 1 3.319124  
## 269 90 3 7.821311  
## 270 90 10 6.582771  
## 271 91 4 8.430592  
## 272 91 5 7.014987  
## 273 91 7 6.582258  
## 274 92 4 1.000000  
## 275 92 7 1.000000  
## 276 92 8 9.409213  
## 277 93 1 1.000000  
## 278 93 7 1.000000  
## 279 93 8 6.253491  
## 280 94 4 3.963601  
## 281 94 8 8.730513  
## 282 94 10 7.326352  
## 283 95 4 4.890290  
## 284 95 8 10.000000  
## 285 95 10 9.437996  
## 286 96 4 1.000000  
## 287 96 7 1.000000  
## 288 96 8 5.244696  
## 289 97 4 3.700859  
## 290 97 7 2.141669  
## 291 97 10 5.305491  
## 292 98 4 4.418442  
## 293 98 7 3.106260  
## 294 98 8 10.000000  
## 295 99 4 3.670573  
## 296 99 7 1.067972  
## 297 99 10 6.983470  
## 298 100 4 1.000000  
## 299 100 6 3.300107  
## 300 100 7 1.000000

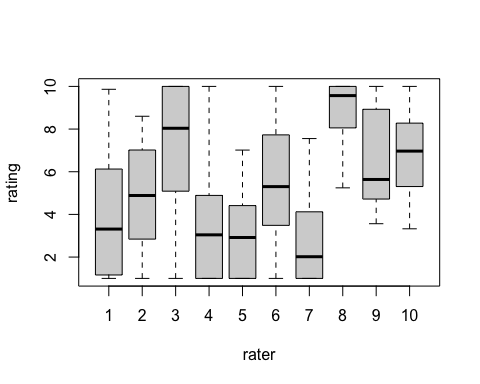
write.csv(ratings.df,"ratings.csv")  
  
library(lattice)  
ratings.df1 <- data.frame(ratings.df, ability=ability[applicant], severity=severity[rater])  
  
xyplot(rating~ability,data=ratings.df1)



xyplot(rating~ability|rater,data=ratings.df1)



boxplot(rating~rater,data=ratings.df1)



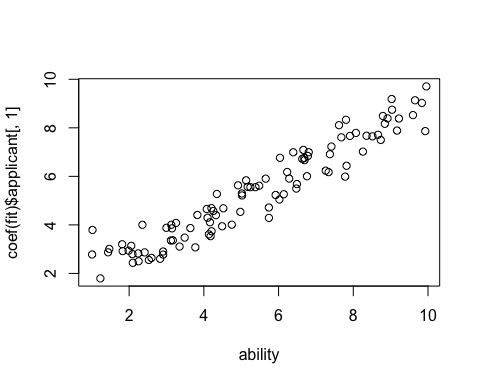
library(arm)  
fit <- lmer(rating ~ (1|applicant) + (1|rater), data=ratings.df)  
  
display(fit)

## lmer(formula = rating ~ (1 | applicant) + (1 | rater), data = ratings.df)  
## coef.est coef.se   
## 5.36 0.64   
##   
## Error terms:  
## Groups Name Std.Dev.  
## applicant (Intercept) 2.08   
## rater (Intercept) 1.92   
## Residual 0.94   
## ---  
## number of obs: 300, groups: applicant, 100; rater, 10  
## AIC = 1137.8, DIC = 1131.7  
## deviance = 1130.8

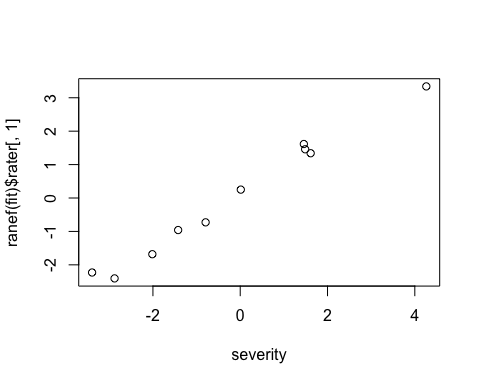
sqrt(9^2/12)

## [1] 2.598076

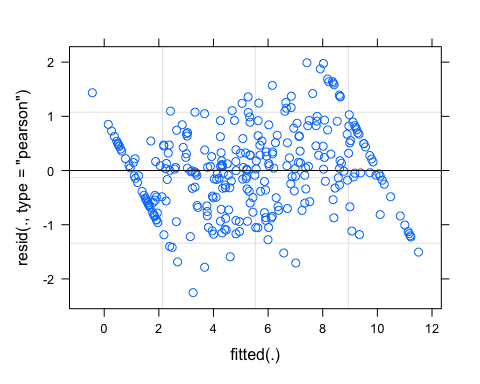
plot(ability,coef(fit)$applicant[,1])



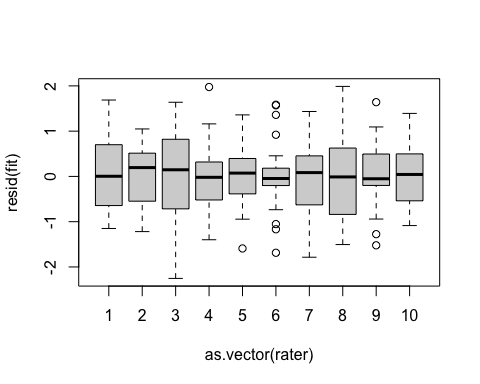
plot(severity,ranef(fit)$rater[,1])



plot(fit)

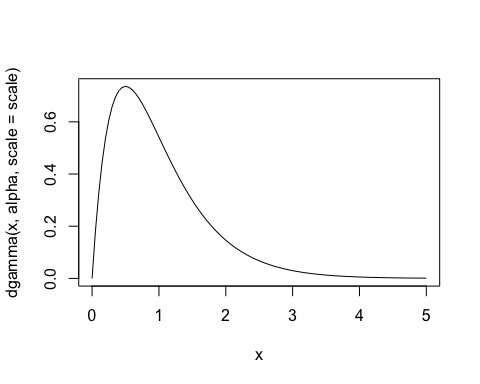


boxplot(resid(fit)~as.vector(rater))



### (b)

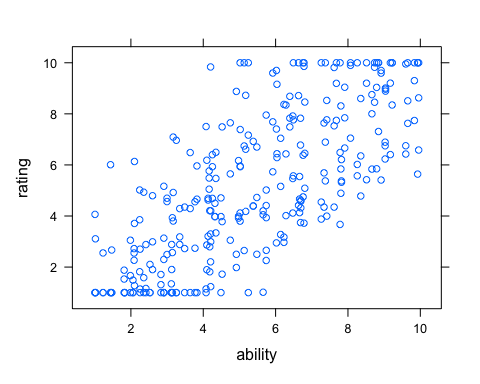
alpha <- 2  
scale <- .5  
curve(dgamma(x,alpha,scale=scale),xlim=c(0,5))



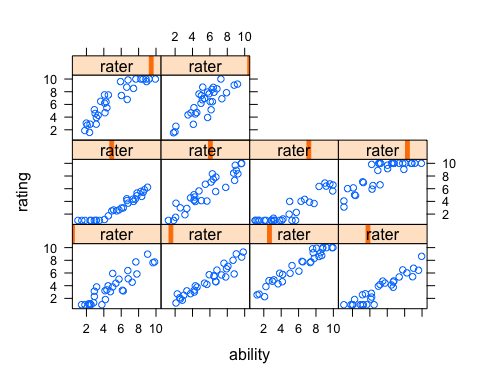
sigma2 <- rgamma(J,alpha,scale=scale)  
rating2 <- ability[applicant] + severity[rater] + rnorm(I\*W,0,sigma2[rater])  
rating2 <- pmax(1,pmin(rating2,10))  
ratings2.df <- data.frame(applicant=applicant, rater=as.vector(rater), rating=rating2, severity=severity[rater], ability=ability[applicant], sigma2=sigma2[rater])  
ratings2.df

## applicant rater rating severity ability sigma2  
## 1 1 3 2.265984 1.4877767 2.088463 0.7904247  
## 2 1 9 2.722904 1.6139248 2.088463 1.1312162  
## 3 1 10 2.551737 1.4578290 2.088463 1.2604318  
## 4 2 2 3.978599 -0.7929399 4.323272 0.6828268  
## 5 2 6 3.973206 0.0132853 4.323272 1.0132648  
## 6 2 10 4.496550 1.4578290 4.323272 1.2604318  
## 7 3 3 8.599447 1.4877767 8.353930 0.7904247  
## 8 3 5 4.782393 -2.8789604 8.353930 0.3411132  
## 9 3 7 6.352362 -3.3938644 8.353930 1.5997597  
## 10 4 3 9.198944 1.4877767 8.513316 0.7904247  
## 11 4 5 5.419206 -2.8789604 8.513316 0.3411132  
## 12 4 9 10.000000 1.6139248 8.513316 1.1312162  
## 13 5 1 8.965668 -1.4225642 9.037567 1.0591885  
## 14 5 2 9.009166 -0.7929399 9.037567 0.6828268  
## 15 5 6 8.889410 0.0132853 9.037567 1.0132648  
## 16 6 1 1.162306 -1.4225642 2.414470 1.0591885  
## 17 6 5 1.000000 -2.8789604 2.414470 0.3411132  
## 18 6 9 2.881571 1.6139248 2.414470 1.1312162  
## 19 7 1 5.086573 -1.4225642 6.807349 1.0591885  
## 20 7 2 6.454767 -0.7929399 6.807349 0.6828268  
## 21 7 10 8.461029 1.4578290 6.807349 1.2604318  
## 22 8 3 5.764736 1.4877767 4.163829 0.7904247  
## 23 8 6 5.049613 0.0132853 4.163829 1.0132648  
## 24 8 10 5.486119 1.4578290 4.163829 1.2604318  
## 25 9 4 5.409174 -2.0135056 8.916556 0.9744233  
## 26 9 6 9.697904 0.0132853 8.916556 1.0132648  
## 27 9 9 9.594964 1.6139248 8.916556 1.1312162  
## 28 10 2 3.296038 -0.7929399 4.202830 0.6828268  
## 29 10 4 2.205034 -2.0135056 4.202830 0.9744233  
## 30 10 8 9.835656 4.2636140 4.202830 1.4674738  
## 31 11 4 1.896995 -2.0135056 4.098462 0.9744233  
## 32 11 9 6.182575 1.6139248 4.098462 1.1312162  
## 33 11 10 2.865148 1.4578290 4.098462 1.2604318  
## 34 12 2 5.689450 -0.7929399 7.338838 0.6828268  
## 35 12 3 7.641267 1.4877767 7.338838 0.7904247  
## 36 12 5 4.346300 -2.8789604 7.338838 0.3411132  
## 37 13 1 1.000000 -1.4225642 2.355169 1.0591885  
## 38 13 8 4.931244 4.2636140 2.355169 1.4674738  
## 39 13 9 1.584704 1.6139248 2.355169 1.1312162  
## 40 14 1 1.000000 -1.4225642 1.826703 1.0591885  
## 41 14 5 1.000000 -2.8789604 1.826703 0.3411132  
## 42 14 6 1.000000 0.0132853 1.826703 1.0132648  
## 43 15 3 2.668110 1.4877767 1.468557 0.7904247  
## 44 15 5 1.000000 -2.8789604 1.468557 0.3411132  
## 45 15 7 1.000000 -3.3938644 1.468557 1.5997597  
## 46 16 7 6.404400 -3.3938644 9.171770 1.5997597  
## 47 16 8 10.000000 4.2636140 9.171770 1.4674738  
## 48 16 10 9.202186 1.4578290 9.171770 1.2604318  
## 49 17 6 4.689566 0.0132853 4.253096 1.0132648  
## 50 17 8 5.930066 4.2636140 4.253096 1.4674738  
## 51 17 9 6.399998 1.6139248 4.253096 1.1312162  
## 52 18 3 4.291185 1.4877767 3.639056 0.7904247  
## 53 18 4 1.000000 -2.0135056 3.639056 0.9744233  
## 54 18 9 6.483971 1.6139248 3.639056 1.1312162  
## 55 19 2 8.002506 -0.7929399 8.661620 0.6828268  
## 56 19 3 8.752655 1.4877767 8.661620 0.7904247  
## 57 19 5 5.836227 -2.8789604 8.661620 0.3411132  
## 58 20 5 2.500173 -2.8789604 4.915200 0.3411132  
## 59 20 7 1.981867 -3.3938644 4.915200 1.5997597  
## 60 20 8 8.878019 4.2636140 4.915200 1.4674738  
## 61 21 1 3.132281 -1.4225642 2.910086 1.0591885  
## 62 21 4 1.000000 -2.0135056 2.910086 0.9744233  
## 63 21 5 1.000000 -2.8789604 2.910086 0.3411132  
## 64 22 3 5.960817 1.4877767 3.825461 0.7904247  
## 65 22 7 1.000000 -3.3938644 3.825461 1.5997597  
## 66 22 10 4.658806 1.4578290 3.825461 1.2604318  
## 67 23 2 5.570780 -0.7929399 6.490266 0.6828268  
## 68 23 3 7.764521 1.4877767 6.490266 0.7904247  
## 69 23 8 10.000000 4.2636140 6.490266 1.4674738  
## 70 24 1 1.140621 -1.4225642 2.253875 1.0591885  
## 71 24 2 2.701198 -0.7929399 2.253875 0.6828268  
## 72 24 7 1.000000 -3.3938644 2.253875 1.5997597  
## 73 25 1 2.298491 -1.4225642 2.910328 1.0591885  
## 74 25 7 1.000000 -3.3938644 2.910328 1.5997597  
## 75 25 9 5.163721 1.6139248 2.910328 1.1312162  
## 76 26 4 1.000000 -2.0135056 1.813251 0.9744233  
## 77 26 9 1.878704 1.6139248 1.813251 1.1312162  
## 78 26 10 1.531699 1.4578290 1.813251 1.2604318  
## 79 27 1 3.337821 -1.4225642 4.349065 1.0591885  
## 80 27 8 6.493254 4.2636140 4.349065 1.4674738  
## 81 27 9 5.471387 1.6139248 4.349065 1.1312162  
## 82 28 5 4.417050 -2.8789604 6.634164 0.3411132  
## 83 28 8 10.000000 4.2636140 6.634164 1.4674738  
## 84 28 9 8.713403 1.6139248 6.634164 1.1312162  
## 85 29 4 3.741923 -2.0135056 6.688691 0.9744233  
## 86 29 6 7.827849 0.0132853 6.688691 1.0132648  
## 87 29 9 6.777405 1.6139248 6.688691 1.1312162  
## 88 30 3 3.853280 1.4877767 2.243513 0.7904247  
## 89 30 4 1.819021 -2.0135056 2.243513 0.9744233  
## 90 30 8 5.017635 4.2636140 2.243513 1.4674738  
## 91 31 2 5.730237 -0.7929399 5.921472 0.6828268  
## 92 31 9 9.598133 1.6139248 5.921472 1.1312162  
## 93 31 10 7.687680 1.4578290 5.921472 1.2604318  
## 94 32 4 6.425930 -2.0135056 9.597532 0.9744233  
## 95 32 6 9.953044 0.0132853 9.597532 1.0132648  
## 96 32 7 6.751721 -3.3938644 9.597532 1.5997597  
## 97 33 1 3.173758 -1.4225642 6.229940 1.0591885  
## 98 33 5 2.967219 -2.8789604 6.229940 0.3411132  
## 99 33 10 8.368607 1.4578290 6.229940 1.2604318  
## 100 34 2 1.272435 -0.7929399 2.098488 0.6828268  
## 101 34 6 3.708905 0.0132853 2.098488 1.0132648  
## 102 34 8 6.134369 4.2636140 2.098488 1.4674738  
## 103 35 2 5.466916 -0.7929399 6.475130 0.6828268  
## 104 35 4 4.115230 -2.0135056 6.475130 0.9744233  
## 105 35 10 7.915441 1.4578290 6.475130 1.2604318  
## 106 36 1 6.378768 -1.4225642 6.760145 1.0591885  
## 107 36 4 4.748522 -2.0135056 6.760145 0.9744233  
## 108 36 8 10.000000 4.2636140 6.760145 1.4674738  
## 109 37 4 2.568151 -2.0135056 3.146508 0.9744233  
## 110 37 7 1.000000 -3.3938644 3.146508 1.5997597  
## 111 37 9 3.932255 1.6139248 3.146508 1.1312162  
## 112 38 1 4.545484 -1.4225642 7.262176 1.0591885  
## 113 38 7 3.878847 -3.3938644 7.262176 1.5997597  
## 114 38 10 10.000000 1.4578290 7.262176 1.2604318  
## 115 39 2 5.373374 -0.7929399 7.820055 0.6828268  
## 116 39 5 5.320612 -2.8789604 7.820055 0.3411132  
## 117 39 6 6.208908 0.0132853 7.820055 1.0132648  
## 118 40 1 3.274473 -1.4225642 6.139561 1.0591885  
## 119 40 6 7.043255 0.0132853 6.139561 1.0132648  
## 120 40 10 5.159053 1.4578290 6.139561 1.2604318  
## 121 41 1 5.841576 -1.4225642 7.804232 1.0591885  
## 122 41 3 8.310068 1.4877767 7.804232 0.7904247  
## 123 41 5 4.889318 -2.8789604 7.804232 0.3411132  
## 124 42 2 6.661505 -0.7929399 7.910548 0.6828268  
## 125 42 8 9.040901 4.2636140 7.910548 1.4674738  
## 126 42 10 7.844730 1.4578290 7.910548 1.2604318  
## 127 43 1 1.815095 -1.4225642 4.179907 1.0591885  
## 128 43 4 2.795057 -2.0135056 4.179907 0.9744233  
## 129 43 6 4.208141 0.0132853 4.179907 1.0132648  
## 130 44 2 5.848825 -0.7929399 8.792783 0.6828268  
## 131 44 3 10.000000 1.4877767 8.792783 0.7904247  
## 132 44 10 9.034837 1.4578290 8.792783 1.2604318  
## 133 45 4 1.000000 -2.0135056 1.986082 0.9744233  
## 134 45 9 3.051181 1.6139248 1.986082 1.1312162  
## 135 45 10 1.666043 1.4578290 1.986082 1.2604318  
## 136 46 2 4.612726 -0.7929399 6.696083 0.6828268  
## 137 46 4 4.577345 -2.0135056 6.696083 0.9744233  
## 138 46 7 4.318521 -3.3938644 6.696083 1.5997597  
## 139 47 6 7.308289 0.0132853 8.844184 1.0132648  
## 140 47 8 10.000000 4.2636140 8.844184 1.4674738  
## 141 47 9 10.000000 1.6139248 8.844184 1.1312162  
## 142 48 1 5.016681 -1.4225642 5.741620 1.0591885  
## 143 48 5 2.655368 -2.8789604 5.741620 0.3411132  
## 144 48 10 3.935296 1.4578290 5.741620 1.2604318  
## 145 49 1 3.976630 -1.4225642 4.489395 1.0591885  
## 146 49 4 1.000000 -2.0135056 4.489395 0.9744233  
## 147 49 10 4.709164 1.4578290 4.489395 1.2604318  
## 148 50 1 3.052025 -1.4225642 4.747852 1.0591885  
## 149 50 6 5.648262 0.0132853 4.747852 1.0132648  
## 150 50 10 7.649129 1.4578290 4.747852 1.2604318  
## 151 51 2 4.201468 -0.7929399 5.654936 0.6828268  
## 152 51 6 4.048520 0.0132853 5.654936 1.0132648  
## 153 51 7 1.014419 -3.3938644 5.654936 1.5997597  
## 154 52 3 6.287783 1.4877767 6.037426 0.7904247  
## 155 52 8 9.156966 4.2636140 6.037426 1.4674738  
## 156 52 10 6.438349 1.4578290 6.037426 1.2604318  
## 157 53 2 7.531082 -0.7929399 7.617902 0.6828268  
## 158 53 3 9.815168 1.4877767 7.617902 0.7904247  
## 159 53 5 4.348453 -2.8789604 7.617902 0.3411132  
## 160 54 3 7.765720 1.4877767 7.414140 0.7904247  
## 161 54 5 3.994895 -2.8789604 7.414140 0.3411132  
## 162 54 9 8.525242 1.6139248 7.414140 1.1312162  
## 163 55 1 7.627407 -1.4225642 9.652258 1.0591885  
## 164 55 2 8.509741 -0.7929399 9.652258 0.6828268  
## 165 55 6 10.000000 0.0132853 9.652258 1.0132648  
## 166 56 1 3.798667 -1.4225642 5.018150 1.0591885  
## 167 56 6 4.117701 0.0132853 5.018150 1.0132648  
## 168 56 8 10.000000 4.2636140 5.018150 1.4674738  
## 169 57 2 1.900784 -0.7929399 2.597991 0.6828268  
## 170 57 3 4.793938 1.4877767 2.597991 0.7904247  
## 171 57 6 2.989017 0.0132853 2.597991 1.0132648  
## 172 58 5 1.000000 -2.8789604 1.019432 0.3411132  
## 173 58 7 1.000000 -3.3938644 1.019432 1.5997597  
## 174 58 8 3.106449 4.2636140 1.019432 1.4674738  
## 175 59 3 10.000000 1.4877767 9.221392 0.7904247  
## 176 59 6 8.347065 0.0132853 9.221392 1.0132648  
## 177 59 9 10.000000 1.6139248 9.221392 1.1312162  
## 178 60 1 4.385438 -1.4225642 5.381858 1.0591885  
## 179 60 2 4.395961 -0.7929399 5.381858 0.6828268  
## 180 60 10 6.922029 1.4578290 5.381858 1.2604318  
## 181 61 3 10.000000 1.4877767 9.954520 0.7904247  
## 182 61 4 8.625937 -2.0135056 9.954520 0.9744233  
## 183 61 7 6.583096 -3.3938644 9.954520 1.5997597  
## 184 62 1 7.744351 -1.4225642 7.680190 1.0591885  
## 185 62 3 9.194236 1.4877767 7.680190 0.7904247  
## 186 62 9 10.000000 1.6139248 7.680190 1.1312162  
## 187 63 1 7.738936 -1.4225642 9.836159 1.0591885  
## 188 63 2 9.299800 -0.7929399 9.836159 0.6828268  
## 189 63 3 10.000000 1.4877767 9.836159 0.7904247  
## 190 64 4 1.000000 -2.0135056 3.118112 0.9744233  
## 191 64 5 1.000000 -2.8789604 3.118112 0.3411132  
## 192 64 7 1.343082 -3.3938644 3.118112 1.5997597  
## 193 65 2 2.491874 -0.7929399 2.998104 0.6828268  
## 194 65 3 4.695982 1.4877767 2.998104 0.7904247  
## 195 65 9 4.564155 1.6139248 2.998104 1.1312162  
## 196 66 2 7.048009 -0.7929399 8.070390 0.6828268  
## 197 66 3 9.899033 1.4877767 8.070390 0.7904247  
## 198 66 8 10.000000 4.2636140 8.070390 1.4674738  
## 199 67 1 3.207271 -1.4225642 4.133904 1.0591885  
## 200 67 3 4.625902 1.4877767 4.133904 0.7904247  
## 201 67 9 4.692481 1.6139248 4.133904 1.1312162  
## 202 68 7 5.640096 -3.3938644 9.926123 1.5997597  
## 203 68 8 10.000000 4.2636140 9.926123 1.4674738  
## 204 68 9 10.000000 1.6139248 9.926123 1.1312162  
## 205 69 2 4.664207 -0.7929399 6.663364 0.6828268  
## 206 69 5 4.131881 -2.8789604 6.663364 0.3411132  
## 207 69 6 5.545681 0.0132853 6.663364 1.0132648  
## 208 70 1 1.000000 -1.4225642 3.771571 1.0591885  
## 209 70 2 2.737629 -0.7929399 3.771571 0.6828268  
## 210 70 6 4.559160 0.0132853 3.771571 1.0132648  
## 211 71 2 3.181820 -0.7929399 3.349516 0.6828268  
## 212 71 6 2.883867 0.0132853 3.349516 1.0132648  
## 213 71 9 4.289663 1.6139248 3.349516 1.1312162  
## 214 72 1 3.802950 -1.4225642 3.169514 1.0591885  
## 215 72 3 4.920800 1.4877767 3.169514 0.7904247  
## 216 72 8 7.090122 4.2636140 3.169514 1.4674738  
## 217 73 1 1.000000 -1.4225642 2.527244 1.0591885  
## 218 73 2 2.103419 -0.7929399 2.527244 0.6828268  
## 219 73 5 1.000000 -2.8789604 2.527244 0.3411132  
## 220 74 3 2.554078 1.4877767 1.229988 0.7904247  
## 221 74 6 1.000000 0.0132853 1.229988 1.0132648  
## 222 74 7 1.000000 -3.3938644 1.229988 1.5997597  
## 223 75 4 4.724665 -2.0135056 5.481126 0.9744233  
## 224 75 5 2.500596 -2.8789604 5.481126 0.3411132  
## 225 75 6 6.700413 0.0132853 5.481126 1.0132648  
## 226 76 5 3.719256 -2.8789604 6.784910 0.3411132  
## 227 76 8 10.000000 4.2636140 6.784910 1.4674738  
## 228 76 9 9.858765 1.6139248 6.784910 1.1312162  
## 229 77 2 3.780474 -0.7929399 4.521682 0.6828268  
## 230 77 5 1.721056 -2.8789604 4.521682 0.3411132  
## 231 77 9 7.489306 1.6139248 4.521682 1.1312162  
## 232 78 6 8.344831 0.0132853 6.285877 1.0132648  
## 233 78 7 4.014822 -3.3938644 6.285877 1.5997597  
## 234 78 10 6.432031 1.4578290 6.285877 1.2604318  
## 235 79 5 2.946874 -2.8789604 6.022823 0.3411132  
## 236 79 8 9.699309 4.2636140 6.022823 1.4674738  
## 237 79 9 7.405026 1.6139248 6.022823 1.1312162  
## 238 80 4 6.018396 -2.0135056 8.257412 0.9744233  
## 239 80 5 5.573616 -2.8789604 8.257412 0.3411132  
## 240 80 9 10.000000 1.6139248 8.257412 1.1312162  
## 241 81 5 1.000000 -2.8789604 3.127866 0.3411132  
## 242 81 6 1.903936 0.0132853 3.127866 1.0132648  
## 243 81 9 2.883881 1.6139248 3.127866 1.1312162  
## 244 82 5 1.144139 -2.8789604 4.078103 0.3411132  
## 245 82 7 1.000000 -3.3938644 4.078103 1.5997597  
## 246 82 9 7.502053 1.6139248 4.078103 1.1312162  
## 247 83 3 7.166909 1.4877767 5.245415 0.7904247  
## 248 83 7 1.000000 -3.3938644 5.245415 1.5997597  
## 249 83 8 10.000000 4.2636140 5.245415 1.4674738  
## 250 84 3 7.829607 1.4877767 6.392900 0.7904247  
## 251 84 6 7.488603 0.0132853 6.392900 1.0132648  
## 252 84 10 8.685398 1.4578290 6.392900 1.2604318  
## 253 85 4 2.725155 -2.0135056 3.485888 0.9744233  
## 254 85 5 1.000000 -2.8789604 3.485888 0.3411132  
## 255 85 10 4.349165 1.4578290 3.485888 1.2604318  
## 256 86 2 2.991369 -0.7929399 4.205899 0.6828268  
## 257 86 6 4.202519 0.0132853 4.205899 1.0132648  
## 258 86 7 1.234071 -3.3938644 4.205899 1.5997597  
## 259 87 5 2.648121 -2.8789604 5.131233 0.3411132  
## 260 87 8 10.000000 4.2636140 5.131233 1.4674738  
## 261 87 10 6.741864 1.4578290 5.131233 1.2604318  
## 262 88 1 1.310699 -1.4225642 2.829816 1.0591885  
## 263 88 2 1.708580 -0.7929399 2.829816 0.6828268  
## 264 88 7 1.000000 -3.3938644 2.829816 1.5997597  
## 265 89 3 9.822772 1.4877767 8.735545 0.7904247  
## 266 89 6 8.454078 0.0132853 8.735545 1.0132648  
## 267 89 8 10.000000 4.2636140 8.735545 1.4674738  
## 268 90 1 5.929252 -1.4225642 5.022025 1.0591885  
## 269 90 3 5.950469 1.4877767 5.022025 0.7904247  
## 270 90 10 7.385487 1.4578290 5.022025 1.2604318  
## 271 91 4 6.748669 -2.0135056 9.027522 0.9744233  
## 272 91 5 6.232131 -2.8789604 9.027522 0.3411132  
## 273 91 7 6.890102 -3.3938644 9.027522 1.5997597  
## 274 92 4 1.000000 -2.0135056 3.255177 0.9744233  
## 275 92 7 1.000000 -3.3938644 3.255177 1.5997597  
## 276 92 8 6.965140 4.2636140 3.255177 1.4674738  
## 277 93 1 1.000000 -1.4225642 1.437316 1.0591885  
## 278 93 7 1.000000 -3.3938644 1.437316 1.5997597  
## 279 93 8 6.007842 4.2636140 1.437316 1.4674738  
## 280 94 4 4.180819 -2.0135056 5.173128 0.9744233  
## 281 94 8 6.613593 4.2636140 5.173128 1.4674738  
## 282 94 10 8.724699 1.4578290 5.173128 1.2604318  
## 283 95 4 5.373512 -2.0135056 7.374701 0.9744233  
## 284 95 8 10.000000 4.2636140 7.374701 1.4674738  
## 285 95 10 6.891867 1.4578290 7.374701 1.2604318  
## 286 96 4 1.000000 -2.0135056 1.008387 0.9744233  
## 287 96 7 1.000000 -3.3938644 1.008387 1.5997597  
## 288 96 8 4.063008 4.2636140 1.008387 1.4674738  
## 289 97 4 3.918358 -2.0135056 4.975807 0.9744233  
## 290 97 7 3.987732 -3.3938644 4.975807 1.5997597  
## 291 97 10 6.169025 1.4578290 4.975807 1.2604318  
## 292 98 4 6.487007 -2.0135056 7.782192 0.9744233  
## 293 98 7 3.669940 -3.3938644 7.782192 1.5997597  
## 294 98 8 10.000000 4.2636140 7.782192 1.4674738  
## 295 99 4 4.410284 -2.0135056 5.739992 0.9744233  
## 296 99 7 2.258524 -3.3938644 5.739992 1.5997597  
## 297 99 10 7.951786 1.4578290 5.739992 1.2604318  
## 298 100 4 1.000000 -2.0135056 2.056819 0.9744233  
## 299 100 6 1.490828 0.0132853 2.056819 1.0132648  
## 300 100 7 1.000000 -3.3938644 2.056819 1.5997597

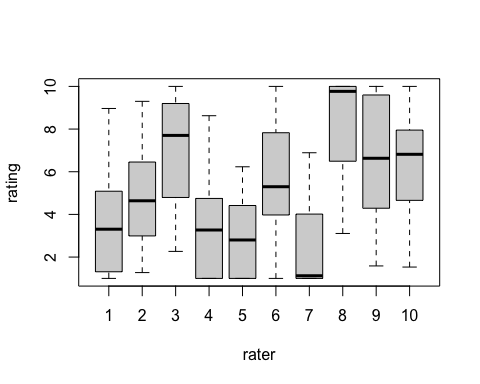
xyplot(rating~ability,data=ratings2.df)



xyplot(rating~ability|rater,data=ratings2.df)



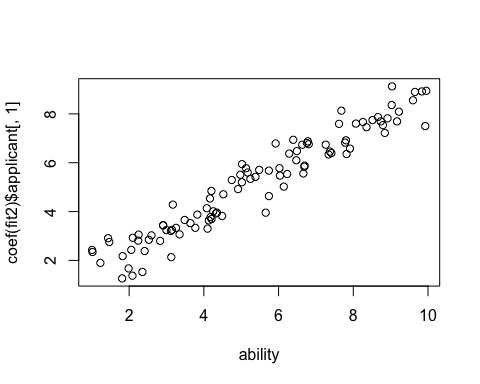
boxplot(rating~rater,data=ratings2.df)



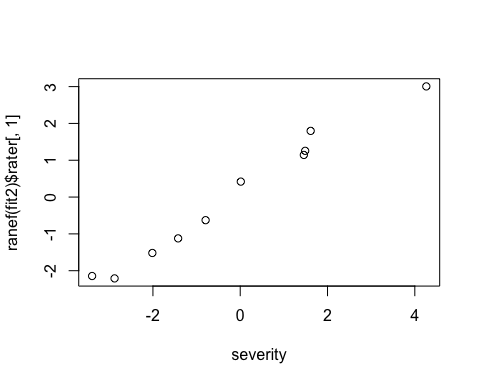
fit2 <- lmer(rating ~ (1|applicant) + (1|rater), data=ratings2.df)  
display(fit2)

## lmer(formula = rating ~ (1 | applicant) + (1 | rater), data = ratings2.df)  
## coef.est coef.se   
## 5.16 0.61   
##   
## Error terms:  
## Groups Name Std.Dev.  
## applicant (Intercept) 2.13   
## rater (Intercept) 1.80   
## Residual 0.97   
## ---  
## number of obs: 300, groups: applicant, 100; rater, 10  
## AIC = 1154.9, DIC = 1148.5  
## deviance = 1147.7

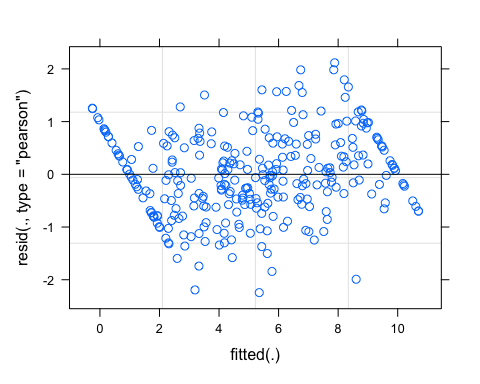
plot(ability,coef(fit2)$applicant[,1])



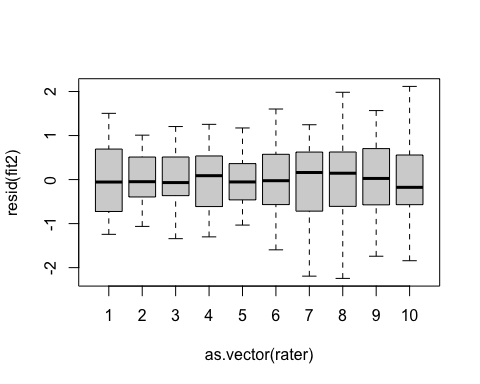
plot(severity,ranef(fit2)$rater[,1])



plot(fit2)



boxplot(resid(fit2)~as.vector(rater))



# Problem 3

library(reshape)

##   
## Attaching package: 'reshape'

## The following object is masked from 'package:data.table':  
##   
## melt

## The following object is masked from 'package:Matrix':  
##   
## expand

filename<- "http://www.stat.columbia.edu/~gelman/arm/examples/olympics/olympics1932.txt"  
olympics1932\_na<-read.fwf(filename,widths=c(2,14,9,9,9,9,9,9,9),skip=21,header = FALSE)  
colnames(olympics1932\_na)<- c("pair", "criterion", "judge\_1", "judge\_2", "judge\_3",  
 "judge\_4", "judge\_5" , "judge\_6", "judge\_7")  
olympics1932<-na.locf(olympics1932\_na)  
olympics1932$criterion<-str\_trim(olympics1932\_na$criterion)  
  
arr\_olym<-melt(data = olympics1932,id.vars=c("pair","criterion"),  
 measure.vars=c(colnames(olympics1932)[3:9]))  
  
olym\_984 <- rename(arr\_olym, c("pair"="skater\_ID", "variable"="judge\_ID"))  
olym\_984 <- olym\_984[order(olym\_984$judge\_ID),]  
olym\_984 <- olym\_984[c("criterion", "value", "skater\_ID", "judge\_ID")]  
  
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  
 )))))  
  
olym\_984

## criterion value skater\_ID judge\_ID SameCountry  
## 1 Program 5.6 1 judge\_1 0  
## 2 Performance 5.6 1 judge\_1 0  
## 3 Program 5.5 2 judge\_1 0  
## 4 Performance 5.5 2 judge\_1 0  
## 5 Program 6.0 3 judge\_1 0  
## 6 Performance 6.0 3 judge\_1 0  
## 7 Program 5.6 4 judge\_1 0  
## 8 Performance 5.6 4 judge\_1 0  
## 9 Program 5.4 5 judge\_1 0  
## 10 Performance 4.8 5 judge\_1 0  
## 11 Program 5.2 6 judge\_1 0  
## 12 Performance 4.8 6 judge\_1 0  
## 13 Program 4.8 7 judge\_1 0  
## 14 Performance 4.3 7 judge\_1 0  
## 15 Program 5.5 1 judge\_2 0  
## 16 Performance 5.5 1 judge\_2 0  
## 17 Program 5.2 2 judge\_2 0  
## 18 Performance 5.7 2 judge\_2 0  
## 19 Program 5.3 3 judge\_2 0  
## 20 Performance 5.5 3 judge\_2 0  
## 21 Program 5.3 4 judge\_2 0  
## 22 Performance 5.3 4 judge\_2 0  
## 23 Program 4.5 5 judge\_2 0  
## 24 Performance 4.8 5 judge\_2 0  
## 25 Program 5.1 6 judge\_2 0  
## 26 Performance 5.6 6 judge\_2 0  
## 27 Program 4.0 7 judge\_2 0  
## 28 Performance 4.6 7 judge\_2 0  
## 29 Program 5.8 1 judge\_3 0  
## 30 Performance 5.8 1 judge\_3 0  
## 31 Program 5.8 2 judge\_3 0  
## 32 Performance 5.6 2 judge\_3 0  
## 33 Program 5.8 3 judge\_3 0  
## 34 Performance 5.7 3 judge\_3 0  
## 35 Program 5.8 4 judge\_3 0  
## 36 Performance 5.8 4 judge\_3 0  
## 37 Program 5.8 5 judge\_3 0  
## 38 Performance 5.5 5 judge\_3 0  
## 39 Program 5.3 6 judge\_3 0  
## 40 Performance 5.0 6 judge\_3 0  
## 41 Program 4.7 7 judge\_3 0  
## 42 Performance 4.5 7 judge\_3 0  
## 43 Program 5.3 1 judge\_4 0  
## 44 Performance 4.7 1 judge\_4 0  
## 45 Program 5.8 2 judge\_4 0  
## 46 Performance 5.4 2 judge\_4 0  
## 47 Program 5.0 3 judge\_4 0  
## 48 Performance 4.9 3 judge\_4 0  
## 49 Program 4.4 4 judge\_4 0  
## 50 Performance 4.8 4 judge\_4 0  
## 51 Program 4.0 5 judge\_4 0  
## 52 Performance 4.4 5 judge\_4 0  
## 53 Program 5.4 6 judge\_4 0  
## 54 Performance 4.7 6 judge\_4 0  
## 55 Program 4.0 7 judge\_4 0  
## 56 Performance 4.0 7 judge\_4 0  
## 57 Program 5.6 1 judge\_5 0  
## 58 Performance 5.7 1 judge\_5 0  
## 59 Program 5.6 2 judge\_5 0  
## 60 Performance 5.5 2 judge\_5 0  
## 61 Program 5.4 3 judge\_5 0  
## 62 Performance 5.5 3 judge\_5 0  
## 63 Program 4.5 4 judge\_5 0  
## 64 Performance 4.5 4 judge\_5 0  
## 65 Program 5.5 5 judge\_5 0  
## 66 Performance 4.6 5 judge\_5 0  
## 67 Program 4.5 6 judge\_5 0  
## 68 Performance 4.0 6 judge\_5 0  
## 69 Program 3.7 7 judge\_5 0  
## 70 Performance 3.6 7 judge\_5 0  
## 71 Program 5.2 1 judge\_6 0  
## 72 Performance 5.3 1 judge\_6 0  
## 73 Program 5.1 2 judge\_6 0  
## 74 Performance 5.3 2 judge\_6 0  
## 75 Program 5.1 3 judge\_6 0  
## 76 Performance 5.2 3 judge\_6 0  
## 77 Program 5.0 4 judge\_6 0  
## 78 Performance 5.0 4 judge\_6 0  
## 79 Program 4.8 5 judge\_6 0  
## 80 Performance 4.8 5 judge\_6 0  
## 81 Program 4.5 6 judge\_6 0  
## 82 Performance 4.6 6 judge\_6 0  
## 83 Program 4.0 7 judge\_6 0  
## 84 Performance 4.0 7 judge\_6 0  
## 85 Program 5.7 1 judge\_7 0  
## 86 Performance 5.4 1 judge\_7 0  
## 87 Program 5.8 2 judge\_7 0  
## 88 Performance 5.7 2 judge\_7 0  
## 89 Program 5.3 3 judge\_7 0  
## 90 Performance 5.7 3 judge\_7 0  
## 91 Program 5.1 4 judge\_7 0  
## 92 Performance 5.5 4 judge\_7 0  
## 93 Program 5.5 5 judge\_7 0  
## 94 Performance 5.2 5 judge\_7 0  
## 95 Program 5.0 6 judge\_7 0  
## 96 Performance 5.2 6 judge\_7 0  
## 97 Program 4.8 7 judge\_7 0  
## 98 Performance 4.8 7 judge\_7 0

### (a)

library(dplyr)

## Warning: package 'dplyr' was built under R version 4.1.2

##   
## Attaching package: 'dplyr'

## The following object is masked from 'package:reshape':  
##   
## rename

## The following object is masked from 'package:car':  
##   
## recode

## The following object is masked from 'package:gridExtra':  
##   
## combine

## The following objects are masked from 'package:data.table':  
##   
## between, first, last

## The following object is masked from 'package:MASS':  
##   
## select

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

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. t-tests use Satterthwaite's method [  
## lmerModLmerTest]  
## 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 df t value Pr(>|t|)   
## (Intercept) 5.1347 0.1954 9.5399 26.28 3.2e-10 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

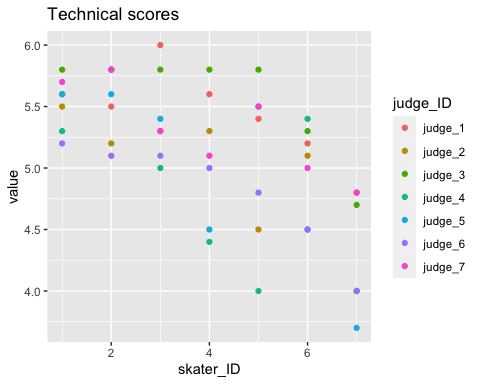
### (b)

reg\_art <- lmer(value ~ 1 + (1|skater\_ID) + (1|judge\_ID),data=data\_art)  
summary(reg\_tech)

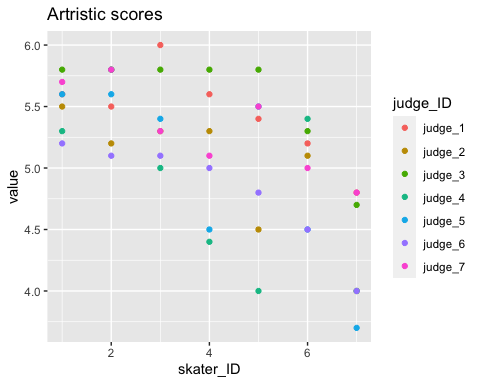
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [  
## lmerModLmerTest]  
## Formula: value ~ 1 + (1 | skater\_ID) + (1 | judge\_ID)  
## Data: data\_tech  
##   
## REML criterion at convergence: 60  
##   
## Scaled residuals:   
## Min 1Q Median 3Q Max   
## -2.51025 -0.45646 -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 df t value Pr(>|t|)   
## (Intercept) 5.1347 0.1954 9.5399 26.28 3.2e-10 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

### (c)

ggplot(data\_tech,aes(x=skater\_ID,y=value,color=judge\_ID))+geom\_point()+  
 ggtitle("Technical scores")

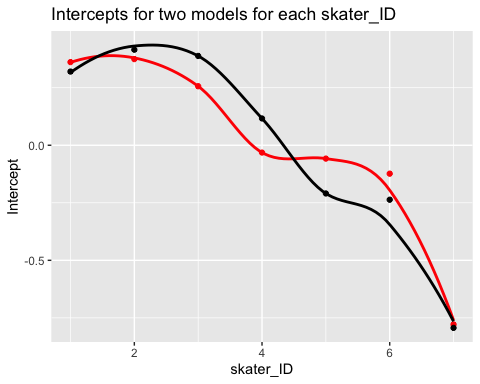


ggplot(data\_tech,aes(x=skater\_ID,y=value,color=judge\_ID))+geom\_point()+  
 ggtitle("Artristic scores")



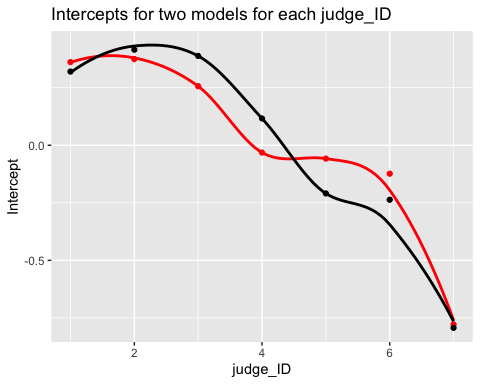
inter\_skate <- as.data.frame(cbind(unlist(ranef(reg\_tech))[1:7],unlist(ranef(reg\_art))[1:7]))  
inter\_skate$skater\_ID <-c(1:7)   
ggplot(data=inter\_skate)+  
 geom\_point(col="red",aes(x=skater\_ID,y=V1))+geom\_smooth(col="red",aes(x=skater\_ID,y=V1),se=FALSE)+  
 geom\_point(col="black",aes(x=skater\_ID,y=V2))+geom\_smooth(col="black",aes(x=skater\_ID,y=V2),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'



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'



### (d)

Please see graphs above.

# Problem 4

library(ggplot2)  
library(bayesplot)

## Warning: package 'bayesplot' was built under R version 4.1.2

## This is bayesplot version 1.9.0

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

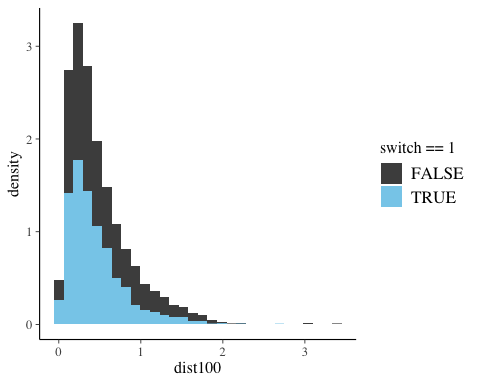
## - bayesplot theme set to bayesplot::theme\_default()

## \* Does \_not\_ affect other ggplot2 plots

## \* See ?bayesplot\_theme\_set for details on theme setting

library(rstanarm)  
theme\_set(bayesplot::theme\_default())  
  
data(wells)  
wells$dist100 <- wells$dist / 100  
  
ggplot(wells, aes(x = dist100, y = ..density.., fill = switch == 1)) +  
 geom\_histogram() +  
 scale\_fill\_manual(values = c("gray30", "skyblue"))

## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.



### (a)

t\_prior <- student\_t(df = 7, location = 0, scale = 2.5)  
fit1 <- stan\_glm(switch ~ dist100, data = wells,  
 family = binomial(link = "logit"),  
 prior = t\_prior, prior\_intercept = t\_prior,  
 cores = 2, seed = 12345)

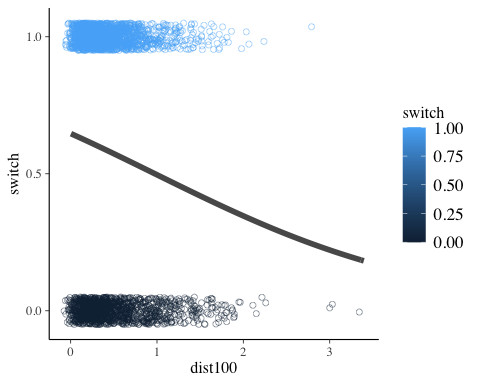
### (b)

round(posterior\_interval(fit1, prob = 0.5), 2)

## 25% 75%  
## (Intercept) 0.57 0.65  
## dist100 -0.69 -0.56

### (c)

# Predicted probability as a function of x  
pr\_switch <- function(x, ests) plogis(ests[1] + ests[2] \* x)  
# A function to slightly jitter the binary data  
jitt <- function(...) {  
 geom\_point(aes\_string(...), position = position\_jitter(height = 0.05, width = 0.1),  
 size = 2, shape = 21, stroke = 0.2)  
}  
ggplot(wells, aes(x = dist100, y = switch, color = switch)) +  
 scale\_y\_continuous(breaks = c(0, 0.5, 1)) +  
 jitt(x="dist100") +  
 stat\_function(fun = pr\_switch, args = list(ests = coef(fit1)),  
 size = 2, color = "gray35")



# Problem 5

df <- read.table("/Users/Home/Documents/Michael\_Ghattas/School/CU\_Boulder/2022/Spring 2022/STAT - 4400/Data/rodents.dat")  
df$race <- factor(df$race, labels=c("White (non-hispanic)", "Black (non-hispanic)", "Puerto Rican", "Other Hispanic", "Asian/Pacific Islander", "Amer-Indian/Native Alaskan", "Two or more races"))  
df$unitflr2 <- as.factor(df$unitflr2)  
df$numunits <- as.factor(df$numunits)  
df$stories <- as.factor(df$stories)  
df$extwin4\_2 <- as.factor(df$extwin4\_2)  
df$extflr5\_2 <- as.factor(df$extflr5\_2)  
df$borough <- factor(df$borough, labels=c("Bronx", "Brooklyn", "Manhattan", "Queens", "Staten Island"))  
df$cd <- as.factor(df$cd)  
df$intcrack2 <- as.factor(df$intcrack2)  
df$inthole2 <- as.factor(df$inthole2)  
df$intleak2 <- as.factor(df$intleak2)  
df$intpeel\_cat <- as.factor(df$intpeel\_cat)  
df$help <- as.factor(df$help)  
df$old <- as.factor(df$old)  
df$dilap <- as.factor(df$dilap)  
df$regext <- as.factor(df$regext)  
df$poverty <- as.factor(df$poverty)  
df$povertyx2 <- as.factor(df$povertyx2)  
df$housing <- factor(df$housing, labels=c("public", "rent controlled/stabilized", "owned", "other rentals"))  
df$board2 <- as.factor(df$board2)  
df$subsidy <- as.factor(df$subsidy)  
df$under6 <- as.factor(df$under6)  
# Missing values  
missingNA <- sapply(df, function(x) sum(is.na(x)))  
df <- na.omit(df)

### (a)

model.14.3A <- glmer(rodent2 ~ 1+race+personrm +intcrack2 + inthole2 + intleak2 +  
 struct +regext+extflr5\_2 +  
 # old+dilap+intpeel\_cat+extwin4\_2+housing +  
 (1|bldg),  
 data=df,  
 family=binomial(link="logit"),  
 control=glmerControl(  
 optimizer="bobyqa",  
 optCtrl=list(maxfun=200000))   
 )  
summary(model.14.3A)

## Generalized linear mixed model fit by maximum likelihood (Laplace  
## Approximation) [glmerMod]  
## Family: binomial ( logit )  
## Formula: rodent2 ~ 1 + race + personrm + intcrack2 + inthole2 + intleak2 +   
## struct + regext + extflr5\_2 + (1 | bldg)  
## Data: df  
## Control: glmerControl(optimizer = "bobyqa", optCtrl = list(maxfun = 2e+05))  
##   
## AIC BIC logLik deviance df.resid   
## 757.5 826.6 -363.7 727.5 729   
##   
## Scaled residuals:   
## Min 1Q Median 3Q Max   
## -2.2234 -0.4474 -0.2733 0.4820 3.4320   
##   
## Random effects:  
## Groups Name Variance Std.Dev.  
## bldg (Intercept) 1.065 1.032   
## Number of obs: 744, groups: bldg, 491  
##   
## Fixed effects:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.89276 0.41929 -4.514 6.36e-06 \*\*\*  
## raceBlack (non-hispanic) 1.05538 0.32064 3.291 0.000997 \*\*\*  
## racePuerto Rican 0.93419 0.37042 2.522 0.011670 \*   
## raceOther Hispanic 1.14651 0.33310 3.442 0.000578 \*\*\*  
## raceAsian/Pacific Islander 0.08936 0.53164 0.168 0.866517   
## raceAmer-Indian/Native Alaskan 1.42349 1.28831 1.105 0.269190   
## raceTwo or more races 1.04408 1.08211 0.965 0.334618   
## personrm 0.80507 0.28081 2.867 0.004144 \*\*   
## intcrack21 1.13764 0.31495 3.612 0.000304 \*\*\*  
## inthole21 0.92155 0.39585 2.328 0.019909 \*   
## intleak21 0.50604 0.25998 1.947 0.051594 .   
## struct -1.18173 0.24889 -4.748 2.05e-06 \*\*\*  
## regext1 -0.33257 0.22167 -1.500 0.133544   
## extflr5\_21 1.11165 0.57035 1.949 0.051288 .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

##   
## Correlation matrix not shown by default, as p = 14 > 12.  
## Use print(x, correlation=TRUE) or  
## vcov(x) if you need it

### (b)

model.14.3B <- glmer(rodent2 ~ 1+race+personrm +intcrack2 + inthole2 + intleak2 +  
 struct +regext+extflr5\_2 +  
 # old+dilap+intpeel\_cat+extwin4\_2+housing +  
 (1|bldg)+  
 (1|cd),  
 data=df,  
 family=binomial(link="logit"),  
 # increase convergence iterations  
 control=glmerControl(  
 optimizer="bobyqa",  
 optCtrl=list(maxfun=200000))   
 )  
summary(model.14.3B)

## Generalized linear mixed model fit by maximum likelihood (Laplace  
## Approximation) [glmerMod]  
## Family: binomial ( logit )  
## Formula: rodent2 ~ 1 + race + personrm + intcrack2 + inthole2 + intleak2 +   
## struct + regext + extflr5\_2 + (1 | bldg) + (1 | cd)  
## Data: df  
## Control: glmerControl(optimizer = "bobyqa", optCtrl = list(maxfun = 2e+05))  
##   
## AIC BIC logLik deviance df.resid   
## 758.7 832.5 -363.3 726.7 728   
##   
## Scaled residuals:   
## Min 1Q Median 3Q Max   
## -2.1093 -0.4523 -0.2703 0.4710 3.4232   
##   
## Random effects:  
## Groups Name Variance Std.Dev.  
## bldg (Intercept) 0.9167 0.9574   
## cd (Intercept) 0.1313 0.3624   
## Number of obs: 744, groups: bldg, 491; cd, 55  
##   
## Fixed effects:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.8858 0.4214 -4.475 7.64e-06 \*\*\*  
## raceBlack (non-hispanic) 1.0288 0.3282 3.135 0.001719 \*\*   
## racePuerto Rican 0.8610 0.3824 2.252 0.024334 \*   
## raceOther Hispanic 1.0946 0.3416 3.204 0.001353 \*\*   
## raceAsian/Pacific Islander 0.1302 0.5331 0.244 0.807097   
## raceAmer-Indian/Native Alaskan 1.4623 1.2740 1.148 0.251035   
## raceTwo or more races 0.9959 1.0878 0.916 0.359907   
## personrm 0.8326 0.2815 2.957 0.003104 \*\*   
## intcrack21 1.1008 0.3157 3.487 0.000488 \*\*\*  
## inthole21 0.9186 0.3934 2.335 0.019548 \*   
## intleak21 0.4901 0.2606 1.880 0.060079 .   
## struct -1.1613 0.2492 -4.659 3.18e-06 \*\*\*  
## regext1 -0.3549 0.2230 -1.592 0.111479   
## extflr5\_21 1.0929 0.5673 1.926 0.054043 .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

##   
## Correlation matrix not shown by default, as p = 14 > 12.  
## Use print(x, correlation=TRUE) or  
## vcov(x) if you need it

### (c)

anova\_logit.14 <- anova(model.14.3B,model.14.3A); anova\_logit.14

## Data: df  
## Models:  
## model.14.3A: rodent2 ~ 1 + race + personrm + intcrack2 + inthole2 + intleak2 + struct + regext + extflr5\_2 + (1 | bldg)  
## model.14.3B: rodent2 ~ 1 + race + personrm + intcrack2 + inthole2 + intleak2 + struct + regext + extflr5\_2 + (1 | bldg) + (1 | cd)  
## npar AIC BIC logLik deviance Chisq Df Pr(>Chisq)  
## model.14.3A 15 757.46 826.64 -363.73 727.46   
## model.14.3B 16 758.67 832.46 -363.33 726.67 0.786 1 0.3753