Some useful and simple tables and plots, and some model diagnostic plots besides

Luke Miratrix and Lily Bliznashka

2019-11-26

In this document we give a few simple plots and summary tables that may be useful for final projects and other things as well. This includes a few simple model diagnostic plots to check for extreme outliers and whatnot.

It is a bit of a hodge-podge, but skimming to get some ideas is definitely worthwhile.

National Youth Survey Example

Our running example is the National Youth Survey (NYS) data as described in Raudenbush and Bryk, page 190. This data comes from a survey in which the same students were asked yearly about their acceptance of 9 "deviant" behaviors (such as smoking marijuana, stealing, etc.). The study began in 1976, and followed two cohorts of children, starting at ages 11 and 14 respectively. We will analyze the first 5 years of data.

At each time point, we have measures of:

- ATTIT, the attitude towards deviance, with higher numbers implying higher tolerance for deviant behaviors.
- EXPO, the "exposure", based on asking the children how many friends they had who had engaged in each of the "deviant" behaviors.

Both of these variables have been transformed to a logarithmic scale to reduce skew.

For each student, we have:

- Gender (binary)
- Minority status (binary)
- Family income, in units of $10K (this can be either categorical or continuous).

Getting the data ready

We'll focus on the first cohort, from ages 11-15. First, let's read the data. Note that this data frame is in "wide format". That is, there is only one row for each student, with all the different observations for that student in different columns of that one row.

```
nyswide = read.csv("nyswide.csv")
head(nyswide)
```

```
##   ID ATTIT.11 EXPO.11 ATTIT.12 EXPO.12 ATTIT.13 EXPO.13 ATTIT.14 EXPO.14
## 1  3     0.11   -0.37     0.20   -0.27     0.00   -0.37     0.00   -0.27
## 2  8     0.29    0.42     0.29    0.20     0.11    0.42     0.51    0.20
## 3  9     0.80    0.47     0.58    0.52     0.64    0.20     0.75    0.47
## 4 15     0.44    0.07     0.44    0.32     0.89    0.47     0.75    0.26
## 5 33     0.20   -0.27     0.64   -0.27     0.69   -0.27       NA      NA
## 6 45     0.11    0.26     0.37   -0.17     0.37    0.14     0.37    0.14
##   ATTIT.15 EXPO.15 FEMALE MINORITY INCOME
## 1     0.11   -0.17      1        0      3
```

```
## 2     0.69    0.20      0        0      4
## 3     0.98    0.47      0        0      3
## 4     0.80    0.47      0        0      4
## 5     0.11    0.07      1        0      4
## 6     0.69    0.32      1        0      4
```

For our purposes, we want it in "long format", i.e. each student has multiple rows for the different observations. The `reshape()` command does this for us (reshape allows us to reshape two variables at once, as compared to `gather()` from tidyverse which does not.

```
  nys1.na = reshape(nyswide, direction="long", #we want it in long format
                    varying=list(ATTIT=paste("ATTIT",11:15,sep="."),
                    EXPO=paste("EXPO",11:15,sep=".") ),
                    v.names=c("ATTIT","EXPO"), idvar="ID", timevar="AGE", times=11:15)

# drop missing ATTIT values
  nys1 = nys1.na[!is.na(nys1.na$ATTIT),]

  head( nys1 )
```

```
##       ID FEMALE MINORITY INCOME AGE ATTIT  EXPO
## 3.11   3      1        0      3  11  0.11 -0.37
## 8.11   8      0        0      4  11  0.29  0.42
## 9.11   9      0        0      3  11  0.80  0.47
## 15.11 15      0        0      4  11  0.44  0.07
## 33.11 33      1        0      4  11  0.20 -0.27
## 45.11 45      1        0      4  11  0.11  0.26
```

Note, the `paste` command makes the sequence `c("ATTIT.12", "ATTIT.13", ...)` to autogenerate our variable names to reshape.

```
  paste("ATTIT",11:15,sep=".")
```

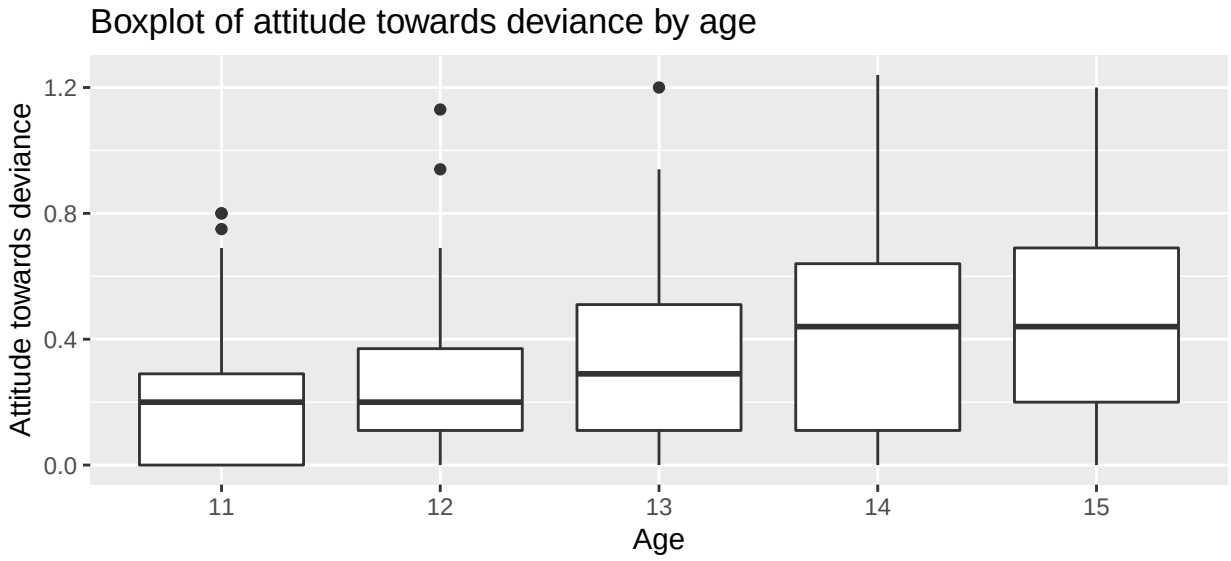
```
## [1] "ATTIT.11" "ATTIT.12" "ATTIT.13" "ATTIT.14" "ATTIT.15"
```

We may wish to make our age variable a factor so it is treated appropriately as an indicator of what wave the data was collected in.

```
  nys1$agefac = as.factor(nys1$AGE)
```

Just to get a sense of the data, let's plot each age as a boxplot

```
  ggplot(nys1, aes(agefac, ATTIT)) +
    geom_boxplot() +
    labs(title = "Boxplot of attitude towards deviance by age",
         x = "Age", y = "Attitude towards deviance")
```



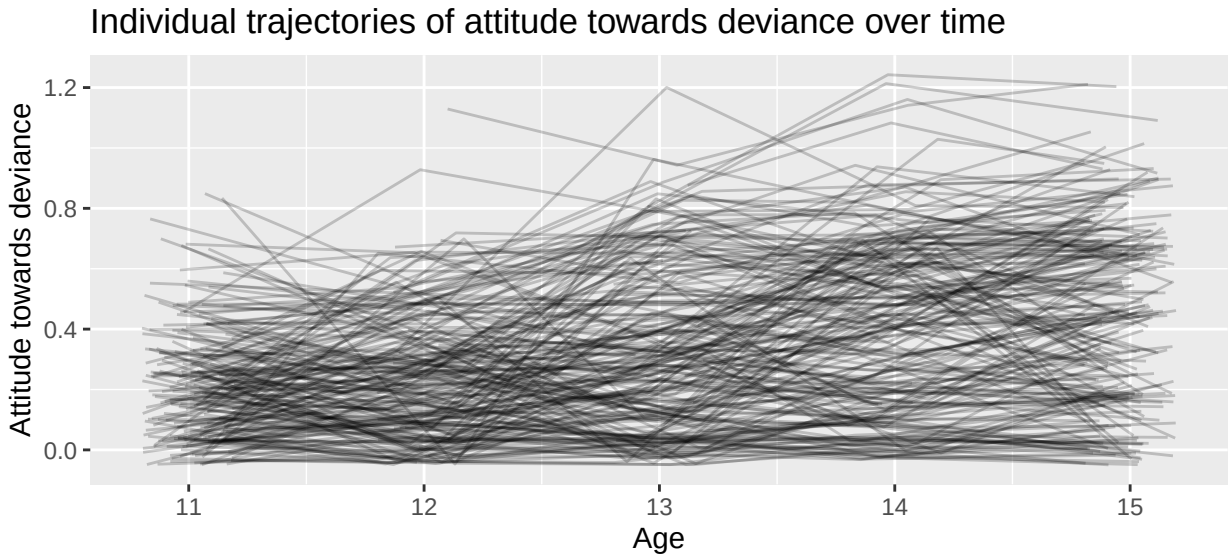
Note some features of the data:

- First, we see that ATTIT goes up over time.
- Second, we see the variation of points also goes up over time. This is evidence of heteroskedasticity.

If we plot individual lines we have:

```
nys1$AGEjit = jitter(nys1$AGE)
nys1$ATTITjit = jitter(nys1$ATTIT, amount=0.05)

ggplot(filter(nys1, complete.cases(nys1)), aes(AGEjit, ATTITjit, group=ID)) +
  geom_line(alpha=0.2) +
  labs(title = "Individual trajectories of attitude towards deviance over time",
       x = "Age", y ="Attitude towards deviance")
```



Note how we have correlation of residuals: some students have systematically lower trajectories and some students have systematically higher trajectories (although there is a lot of bouncing around).

Tabulating data (Categorical variables)

We can tabulate data as so:

```
  table(nys1$AGE)
```

```
##
##  11  12  13  14  15
## 202 209 230 220 218
```

or

```
  table(nys1$MINORITY, nys1$AGE)
```

```
##
##      11  12  13  14  15
##   0 159 165 182 175 175
##   1  43  44  48  45  43
```

Interestingly, we have more observations for later ages.

We can make "proportion tables" as well:

```
prop.table( table( nys1$MINORITY, nys1$INCOME  ), margin=1 )
```

```
##
##           1       2       3       4       5       6       7       8       9
##   0 0.06075 0.13551 0.18341 0.18107 0.14369 0.10981 0.06893 0.05257 0.00935
##   1 0.28251 0.41704 0.12556 0.05830 0.05830 0.02242 0.01345 0.00000 0.00000
##
##          10
##   0 0.05491
##   1 0.02242
```

The margin determines what adds up to 100%.

Summary stats (continuous variables)

The `tableone` package is useful:

```
  library(tableone)
```

```
## Warning: package 'tableone' was built under R version 3.5.2
```

```
# sample mean
  CreateTableOne(data = nys1,
                 vars = c("ATTIT"))
```

```
##
##                      Overall
##   n                  1079
##   ATTIT (mean (SD)) 0.33 (0.27)
```

```
# you can also stratify by a variables of interest
  CreateTableOne(data = nys1,
                 vars = c("ATTIT"),
                 strata = c("FEMALE"))
```

```
##                     Stratified by FEMALE
##                      0           1           p      test
##   n                   559         520
##   ATTIT (mean (SD)) 0.37 (0.27) 0.29 (0.27) <0.001
```

```
# you can also include both binary variables
  CreateTableOne(data = nys1,
                 vars = c("ATTIT", "agefac"),  # include both binary and continuous variables here
                 factorVars = c("agefac"), # include only binary variables here
                 strata = c("FEMALE"))
```

```
##                     Stratified by FEMALE
##                      0            1            p      test
##   n                   559          520
##   ATTIT (mean (SD)) 0.37 (0.27)  0.29 (0.27)  <0.001
##   agefac (%)                                   0.991
##      11               106 (19.0)    96 (18.5)
##      12               105 (18.8)   104 (20.0)
##      13               119 (21.3)   111 (21.3)
##      14               115 (20.6)   105 (20.2)
##      15               114 (20.4)   104 (20.0)
```

Table of summary stats

You can easily make pretty tables using the `stargazer` package:

```
  library(stargazer)
```

```
##
## Please cite as:
```

```
##  Hlavac, Marek (2018). stargazer: Well-Formatted Regression and Summary Statistics Tables.
```

```
##  R package version 5.2.2. https://CRAN.R-project.org/package=stargazer
```

```
# to include all variables
  stargazer(nys1, header = FALSE)
```

Table 1:

Statistic	N	Mean	St. Dev.	Min	Pctl(25)	Pctl(75)	Max
ID	1,079	841.000	484.000	3	422	1,242	1,720
FEMALE	1,079	0.482	0.500	0	0	1	1
MINORITY	1,079	0.207	0.405	0	0	0	1
INCOME	1,079	4.100	2.350	1	2	5	10
AGE	1,079	13.000	1.400	11	12	14	15
ATTIT	1,079	0.330	0.272	0.000	0.110	0.510	1.240
EXPO	1,079	−0.002	0.301	−0.370	−0.270	0.200	1.040
AGEjit	1,079	13.000	1.410	10.800	11.900	14.100	15.200
ATTITjit	1,079	0.330	0.273	−0.050	0.101	0.526	1.240

You can include only some of the variables and omit stats that are not of interest:

```
# to include only variables of interest
  stargazer(nys1[2:7], header=FALSE,
```

```
            omit.summary.stat = c("p25", "p75", "min", "max"), # to omit percentiles
            title = "Table 1: Descriptive statistics")
```

Table 2: Table 1: Descriptive statistics

Statistic	N	Mean	St. Dev.
FEMALE	1,079	0.482	0.500
MINORITY	1,079	0.207	0.405
INCOME	1,079	4.100	2.350
AGE	1,079	13.000	1.400
ATTIT	1,079	0.330	0.272
EXPO	1,079	−0.002	0.301

See the **stargazer** help file for how to set/change more of the options: https://cran.r-project.org/web/packages/stargazer/stargazer.pdf

High School and Beyond Example

For this part, we'll use the HSB data to summarize variables by group/school.

```
# load data
  # read student data
  dat = read.spss( "hsb1.sav", to.data.frame=TRUE )
  # read school data
  sdat = read.spss( "hsb2.sav", to.data.frame=TRUE )
```

```
## re-encoding from CP1252
```

```
  # merge
  data = merge( dat, sdat, by="id" )
```

Summarizing by group

To plot summaries by group, first aggregate your data, and plot the results. Like so:

```
aggdat = data %>% group_by(id, sector) %>%
  summarize( per.fem = mean( female ) )
head( aggdat )
```

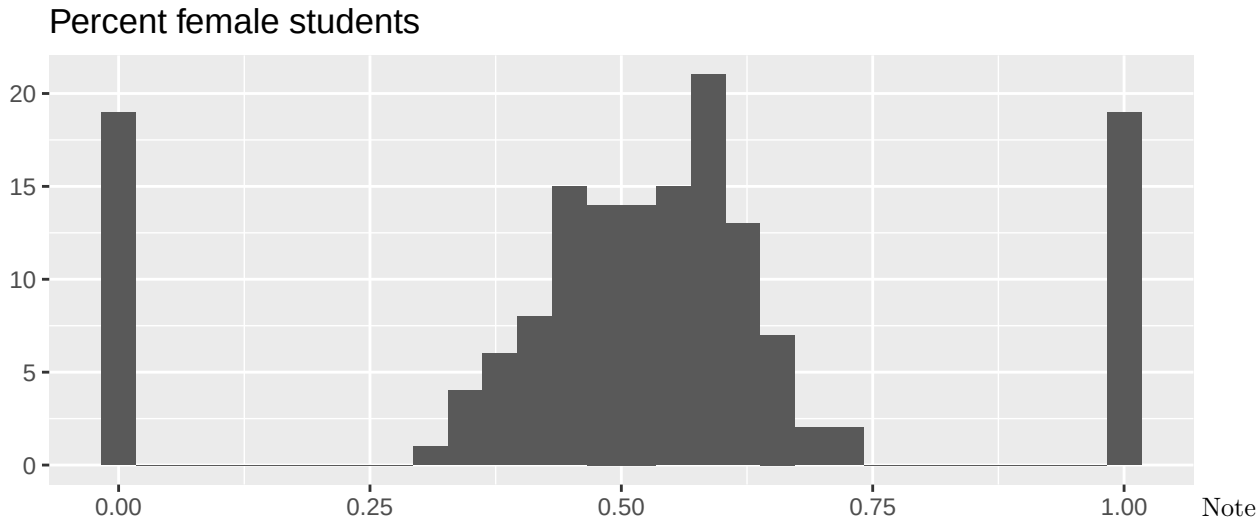
```
## # A tibble: 6 x 3
## # Groups:   id [6]
##    id    sector per.fem
##   <fct>  <dbl>   <dbl>
## 1 1224       0   0.596
## 2 1288       0   0.44
## 3 1296       0   0.646
## 4 1308       1   0
## 5 1317       1   1
## 6 1358       0   0.367
```

The including sector (a level 2 variable) is a way to ensure it gets carried through to the aggregated results. Neat trick.

Anyway, we then plot:

```
qplot( aggdat$per.fem,
       main = "Percent female students",
       xlab = "")
```

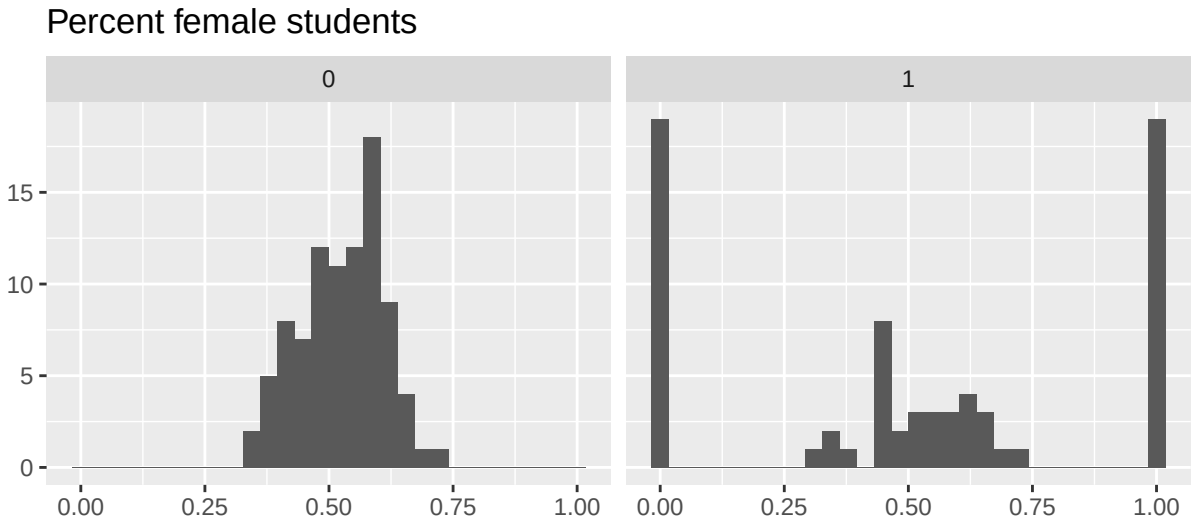
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Note the single sex (catholic) schools. We can facet to see both groups:

```
qplot( per.fem, data=aggdat,
       main = "Percent female students",
       xlab = "") +
  facet_wrap( ~ sector )
```

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



Diagnostic plots

We can also make some disagnostic plots for our model. first, let's fit a random intercept model.

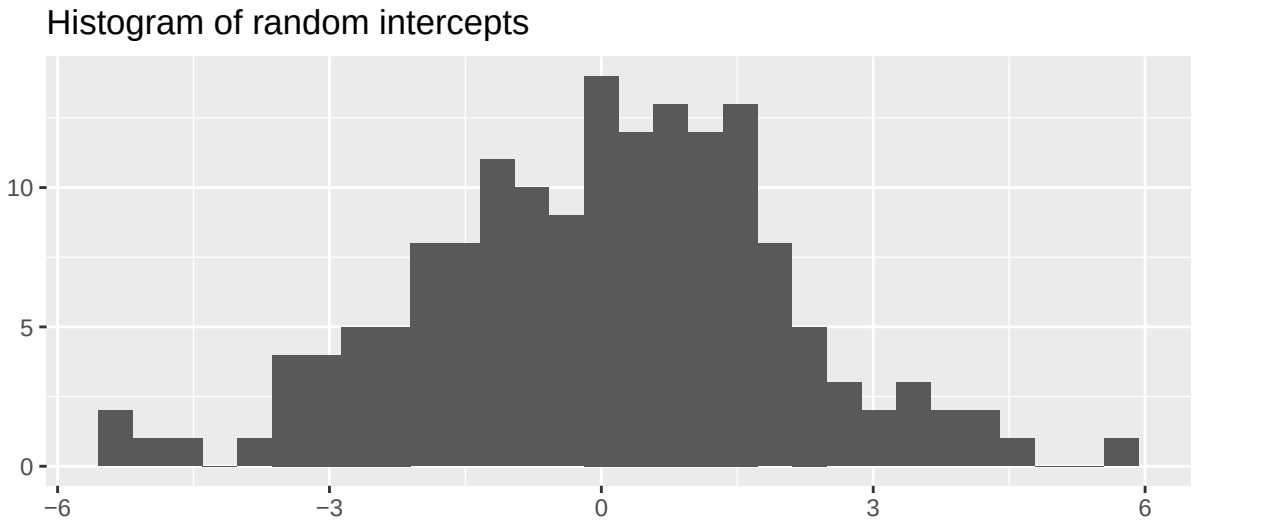
```
m1 = lmer(mathach ~ 1 + ses + (1|id), data=dat)
arm::display(m1)
```

```
## lmer(formula = mathach ~ 1 + ses + (1 | id), data = dat)
##             coef.est coef.se
## (Intercept) 12.66     0.19
## ses          2.39     0.11
##
## Error terms:
##  Groups   Name        Std.Dev.
##  id       (Intercept) 2.18
##  Residual             6.09
## ---
## number of obs: 7185, groups: id, 160
## AIC = 46653.2, DIC = 46637
## deviance = 46641.0
```

We can check if some of our assumptions are being grossly violated, i.e. residuals at all levels are normally distributed.

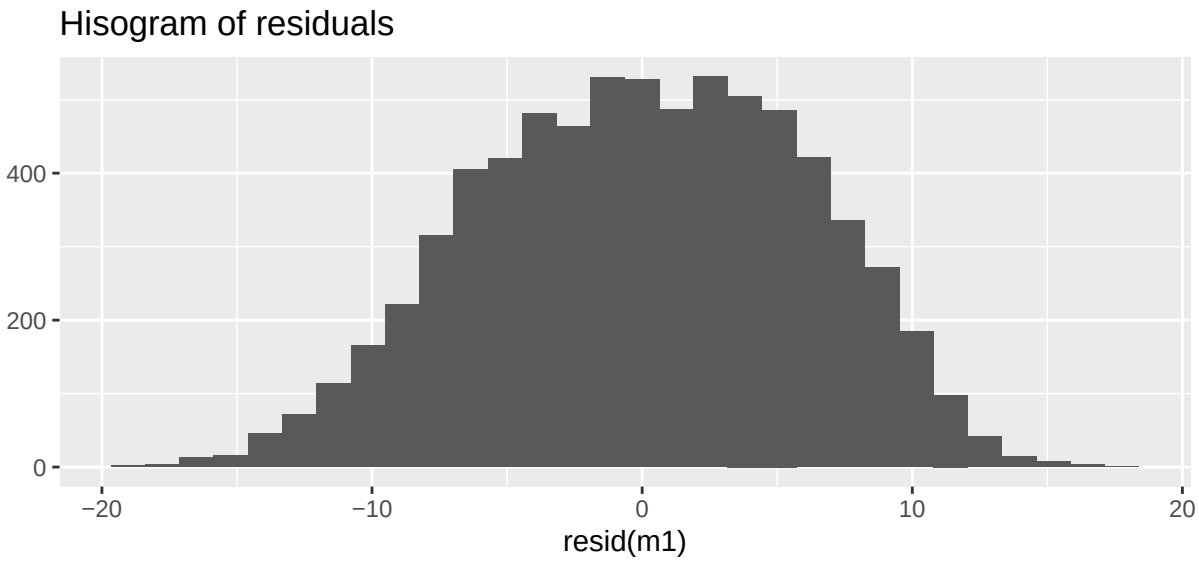
```
qplot(ranef(m1)$id[,1],
      main = "Histogram of random intercepts", xlab="")
```

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




```
qplot(resid(m1),
      main = "Hisogram of residuals")
```

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

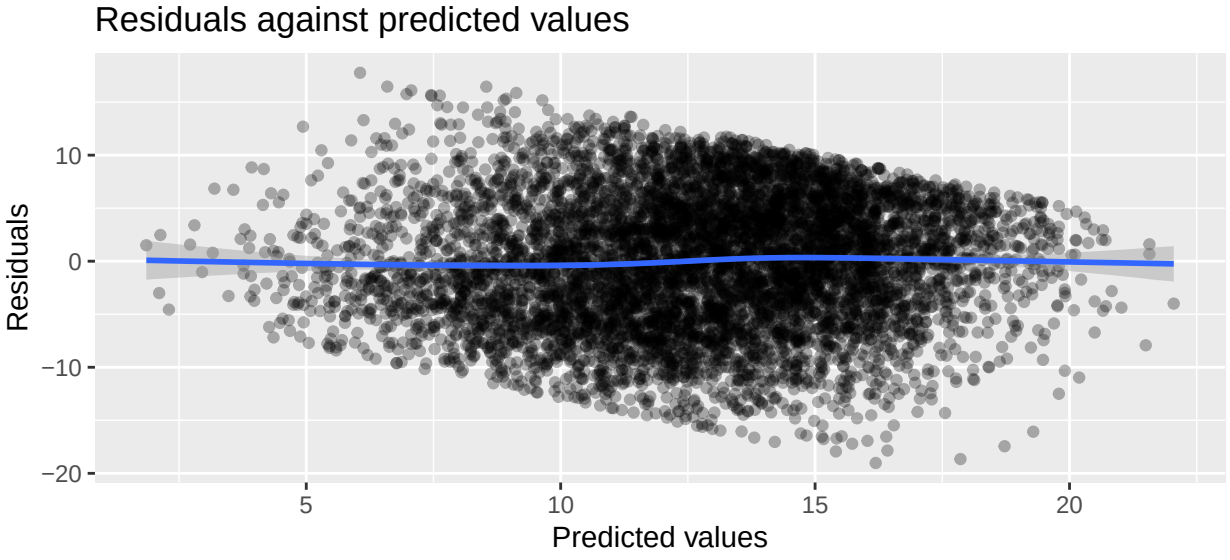


We can check for heteroskedasticity by plotting residuals against predicted values

```
data$yhat  = predict(m1)
data$resid = resid(m1)

ggplot(data, aes(yhat, resid)) +
    geom_point(alpha=0.3) +
    geom_smooth() +
    labs(title = "Residuals against predicted values",
         x = "Predicted values", y ="Residuals")
```

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



It looks reasonable (up to the discrete and bounded nature of our data). No major weird curves in the loess line through the residuals means linearity is a reasonable assumption. That being said, our nominal SEs around our loess line are tight, so the mild curve is probably evidence of *some* model misfit.

We can also look at the distribution of random effects using the `lattice` package

```
library(lattice)
qqmath(ranef(m1, condVar=TRUE), strip=FALSE)
```

```
## $id
```