04 Hierarchical Linear modeling

Shravan Vasishth / Bruno Nicenboim

25-29 March 2019

Understanding the independence assumption

Consider the reading time data from the experiment by Grodner and Gibson, 2005. This is the data from their Experiment 1. You can download the paper from **here**.

In this paper, the interest is in the reading time differences between object and subject relatives at the relative clause verb. The expectation from theory is that object relatives (objgap) have longer reading times than subject relatives (subjgap). The explanation for the longer reading times in objgap vs subjgap lies in working memory constraints: it is more difficult to figure out who did what to whom in object relatives than subject relatives.

Load and preprocess data

First, load the data-set provided, and do the preprocessing shown. This gives us the relevant data.

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

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

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

```
gg05e1 <- read.table("data/GrodnerGibson2005E1.csv",sep=",", header=T)
gge1 <- gg05e1 %>% filter(item != 0)

gge1 <- gge1 %>% mutate(word_positionnew = ifelse(item != 15 & word_position > 10,
                                                  word_position-1, word_position))
#there is a mistake in the coding of word position,
#all items but 15 have regions 10 and higher coded
#as words 11 and higher

## get data from relative clause verb:
gge1crit <- subset(gge1, ( condition == "objgap" & word_position == 6 ) |
            ( condition == "subjgap" & word_position == 4 ))
gge1crit<-gge1crit[,c(1,2,3,6)]
head(gge1crit)
```

```
##    subject item condition rawRT
## 6        1    1    objgap   320
## 19       1    2   subjgap   424
## 34       1    3    objgap   309
```

```
## 49       1    4   subjgap   274
## 68       1    5    objgap   333
## 80       1    6   subjgap   266
```

Always check what the data look like

Each of the 42 participants see multiple (eight) instances of subject and object relatives:

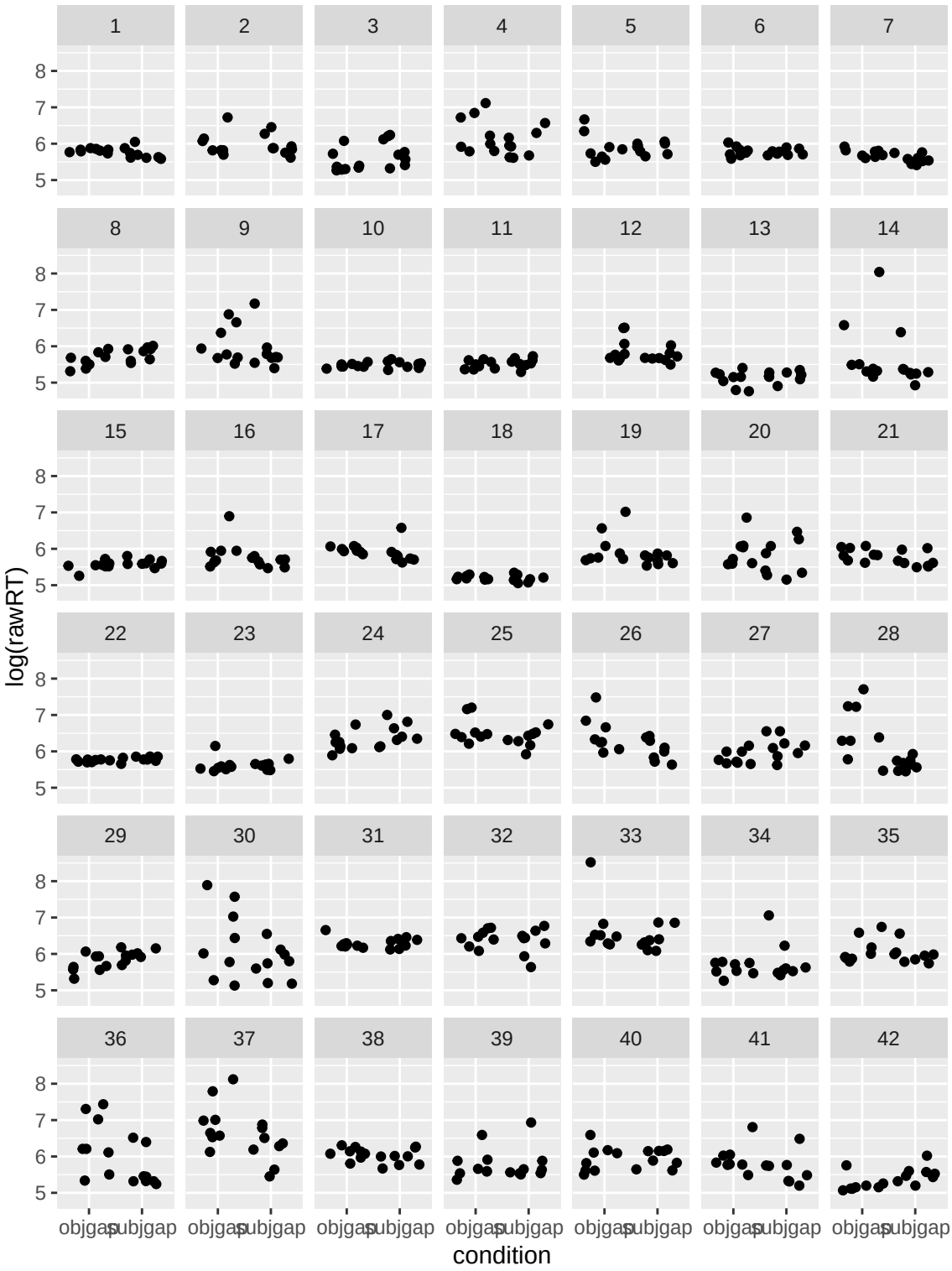
```
xtabs(~subject+condition,
          gge1crit)
```

```
##         condition
## subject objgap subjgap
##      1       8       8
##      2       8       8
##      3       8       8
##      4       8       8
##      5       8       8
##      6       8       8
##      7       8       8
##      8       8       8
##      9       8       8
##      10      8       8
##      11      8       8
##      12      8       8
##      13      8       8
##      14      8       8
##      15      8       8
##      16      8       8
##      17      8       8
##      18      8       8
##      19      8       8
##      20      8       8
##      21      8       8
##      22      8       8
##      23      8       8
##      24      8       8
##      25      8       8
##      26      8       8
##      27      8       8
##      28      8       8
##      29      8       8
##      30      8       8
##      31      8       8
##      32      8       8
##      33      8       8
##      34      8       8
##      35      8       8
##      36      8       8
##      37      8       8
##      38      8       8
##      39      8       8
##      40      8       8
##      41      8       8
```

```
##      42      8       8
```

So, from each participant, we have **repeated** measures, which are therefore **not** independent (because they come from the same subject).

```
library(ggplot2)
p <- ggplot(gge1crit, aes(x=condition, y=log(rawRT))) + geom_point(position="jitter")+
  facet_wrap( ~ subject, nrow=6)
p
```



Aggregate subject data

It is common in psychology to aggregate such data by subjects:

```
gge1bysubj<-aggregate(rawRT~subject+condition,mean,
                      data=gge1crit)
```

Look at the data:

```
head(gge1bysubj)
```

```
##   subject condition   rawRT
## 1       1    objgap 335.000
## 2       2    objgap 419.375
## 3       3    objgap 247.250
## 4       4    objgap 616.375
## 5       5    objgap 395.750
## 6       6    objgap 330.250
```

```
# one row for each subject for each condition:
dim(gge1bysubj)
```

```
## [1] 84  3
```

```
xtabs(~subject+condition,
      gge1bysubj)
```

```
##        condition
## subject objgap subjgap
##      1       1       1
##      2       1       1
##      3       1       1
##      4       1       1
##      5       1       1
##      6       1       1
##      7       1       1
##      8       1       1
##      9       1       1
##      10      1       1
##      11      1       1
##      12      1       1
##      13      1       1
##      14      1       1
##      15      1       1
##      16      1       1
##      17      1       1
##      18      1       1
##      19      1       1
##      20      1       1
##      21      1       1
##      22      1       1
##      23      1       1
##      24      1       1
##      25      1       1
##      26      1       1
##      27      1       1
##      28      1       1
##      29      1       1
##      30      1       1
##      31      1       1
```

```
##      32      1       1
##      33      1       1
##      34      1       1
##      35      1       1
##      36      1       1
##      37      1       1
##      38      1       1
##      39      1       1
##      40      1       1
##      41      1       1
##      42      1       1
```

Note that

- Each subject's responses are assumed to be **independent** of the others' (a debatable assumption!)
- We have *two* data points from each subject: one for objgap and one for subjgap. The responses from each subject are therefore *repeated* measures, hence correlated (not independent. We ignore this problem for now.

Generating fake data

We will now try to generate fake data resembling Grodner and Gibson's data:

Step 1: Fit a linear model to the aggregated data

```
## sum contrast:
gge1bysubj$so<-ifelse(gge1bysubj$condition=="objgap",-1,1)
m0<-lm(rawRT~so,gge1bysubj)
summary(m0)
```

```
##
## Call:
## lm(formula = rawRT ~ so, data = gge1bysubj)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -303.36 -116.35  -51.59   49.05  853.26
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   420.22      22.01  19.092   <2e-16 ***
## so            -51.14      22.01  -2.324   0.0226 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 201.7 on 82 degrees of freedom
## Multiple R-squared:  0.06177,    Adjusted R-squared:  0.05033
## F-statistic: 5.399 on 1 and 82 DF,  p-value: 0.02263
```

```
coefs<-summary(m0)$coefficients[,1]
sigma<-summary(m0)$sigma
```

Step 2: Generate fake data using estimates

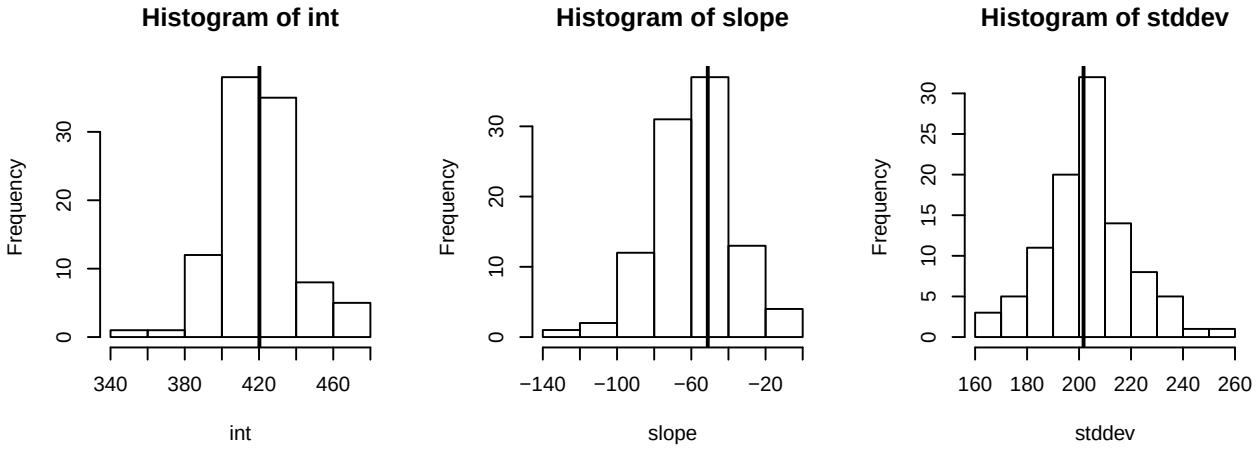
```
ncond<-2
n<-42*ncond
so<-gge1bysubj$so

y = coefs[1] + coefs[2]*so+rnorm(n,sigma)
fakedata<-data.frame(y=y,so=so)
head(fakedata)
```

```
##          y so
## 1 673.3046 -1
## 2 672.5361 -1
## 3 674.5356 -1
## 4 672.0265 -1
## 5 673.8216 -1
## 6 672.8650 -1
```

```
int<-slope<-stddev<-rep(NA,100)
for(i in 1:100){
y = coefs[1] + coefs[2]*so+rnorm(n,mean=0,sd=sigma)
fakedata<-data.frame(y=y,so=so)
m<-lm(y~so,fakedata)
int[i]<-summary(m)$coefficients[1,1]
slope[i]<-summary(m)$coefficients[2,1]
stddev[i]<-summary(m)$sigma
}
```

```
op<-par(mfrow=c(1,3),pty="s")
hist(int)
abline(v=coefs[1],lwd=2)
hist(slope)
abline(v=coefs[2],lwd=2)
hist(stddev)
abline(v=sigma,lwd=2)
```



Exercise 1

Load the following data and subset the relevant data:

```
chineseRC<-read.table("data/gibsonwu2012data.txt")
crit<-subset(chineseRC,region=="headnoun")
crit$region<-factor(crit$region)
head(crit[,c(1,2,3,7)])
```

```
##     subj item     type   rt
## 94     1   13  obj-ext 1561
## 221    1    6 subj-ext  959
## 341    1    5  obj-ext  582
## 461    1    9  obj-ext  294
## 621    1   14 subj-ext  438
## 753    1    4 subj-ext  286
```

```
crit<-crit[,c(1,2,3,7)]
head(crit)
```

```
##     subj item     type   rt
## 94     1   13  obj-ext 1561
## 221    1    6 subj-ext  959
## 341    1    5  obj-ext  582
## 461    1    9  obj-ext  294
## 621    1   14 subj-ext  438
## 753    1    4 subj-ext  286
```

Tasks

- Code the two levels of the two-level factor called type using sum coding (± 1); call the predictor x.
- Aggregate the data by subject
- Fit a linear model to the data investigating the effect of the predictor x (the predictor type that is now sum-coded)
- Using estimates of the parameters from the linear model as a starting point, generate fake data 100 times, and check that you can get realistic estimates of the parameters from the fake data, just as shown in the example above with the Grodner and Gibson data.

Understanding the False Discovery Rate (Type I error) of our Grodner and Gibson model

Type I error

```
nsim<-1000
tval<-rep(NA,nsim)
for(i in 1:nsim){
y = coefs[1]+rnorm(n,mean=0,sd=sigma)
fakedata<-data.frame(y=y,so=so)
m<-lm(y~so,fakedata)
tval[i]<-abs(summary(m)$coefficients[2,3])
}
## Type I error:
mean(tval>1.65)
```

```
## [1] 0.103
```

Understanding the True Discovery Rate (power=1-Type II error) of our model

For starters we will assume that the estimated relative clause effect is the true effect:

```
## slope
coefs[2]
```

```
##        so
## -51.14286
```

Under this assumption, we can now compute power:

```
nsim<-1000
tval<-rep(NA,nsim)
for(i in 1:nsim){
y = coefs[1]+coefs[2]*so+ rnorm(n,mean=0,sd=sigma)
fakedata<-data.frame(y=y,so=so)
m<-lm(y~so,fakedata)
tval[i]<-abs(summary(m)$coefficients[2,3])
}
## Power:
mean(tval>2)
```

```
## [1] 0.614
```

Understanding the repeated measures nature of the data

What's missing in the fake data simulation here? The fact that the *same* subject is giving us objgap and subjgap data!

```
subj<-rep(1:42,2)
fakedata$subj<-subj
fakedata
```

```
##             y so subj
## 1   572.71601 -1    1
## 2   422.76585 -1    2
## 3   425.40218 -1    3
## 4   316.07410 -1    4
## 5   179.84828 -1    5
## 6   291.25604 -1    6
## 7   575.00216 -1    7
## 8   526.36597 -1    8
## 9   693.15758 -1    9
## 10  873.48665 -1   10
## 11  540.54332 -1   11
## 12  437.83004 -1   12
## 13  461.48205 -1   13
## 14  547.24830 -1   14
## 15  499.18449 -1   15
## 16  450.89069 -1   16
## 17  861.40566 -1   17
## 18  266.36670 -1   18
## 19  691.72747 -1   19
## 20   30.40166 -1   20
## 21  353.47734 -1   21
```

```
## 22  459.95206 -1   22
## 23  257.14487 -1   23
## 24  498.76589 -1   24
## 25  307.94218 -1   25
## 26  476.59711 -1   26
## 27  478.34641 -1   27
## 28  311.72454 -1   28
## 29  514.09641 -1   29
## 30  412.21403 -1   30
## 31  492.61340 -1   31
## 32  510.61981 -1   32
## 33  913.28063 -1   33
## 34  642.25999 -1   34
## 35  613.59132 -1   35
## 36  497.16663 -1   36
## 37  230.89171 -1   37
## 38  555.53903 -1   38
## 39  226.67703 -1   39
## 40  633.65398 -1   40
## 41  337.06314 -1   41
## 42  576.92674 -1   42
## 43  758.10199  1    1
## 44  504.10074  1    2
## 45  502.04844  1    3
## 46  333.32242  1    4
## 47  283.18124  1    5
## 48  269.30736  1    6
## 49  458.18549  1    7
## 50  653.33232  1    8
## 51  272.72904  1    9
## 52 1034.19668  1   10
## 53  723.03575  1   11
## 54  182.97972  1   12
## 55  502.29116  1   13
## 56  452.53310  1   14
## 57  525.90035  1   15
## 58  345.92081  1   16
## 59  583.29593  1   17
## 60  274.44180  1   18
## 61  406.41378  1   19
## 62  265.79763  1   20
## 63  299.75548  1   21
## 64  428.10958  1   22
## 65  760.38994  1   23
## 66  565.11464  1   24
## 67  464.09540  1   25
## 68  381.33816  1   26
## 69  797.89744  1   27
## 70   86.03476  1   28
## 71  301.70140  1   29
## 72  633.05529  1   30
## 73  302.02116  1   31
## 74  -52.66738  1   32
## 75  418.33312  1   33
```

```
## 76  131.09121  1   34
## 77  715.11776  1   35
## 78  412.54131  1   36
## 79  545.09322  1   37
## 80  470.06698  1   38
## 81  566.58951  1   39
## 82  190.14112  1   40
## 83  630.96991  1   41
## 84  508.65003  1   42
```

Our linear model does not have any way to express the fact that we have repeated measures from each subject.

The linear model analysis is identical to the two-sample t-test (**both the linear model and the two-sample t-test are incorrect because they ignored the repeated measures nature of the data**):

```
t.test(rawRT~condition,gge1bysubj)
```

```
##
##  Welch Two Sample t-test
##
## data:  rawRT by condition
## t = 2.3235, df = 57.132, p-value = 0.02373
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##   14.13898 190.43245
## sample estimates:
##  mean in group objgap mean in group subjgap
##              471.3601              369.0744
```

The correct t-test would assume that the data from a given subject are **paired** (=repeated measures):

```
t.test(rawRT~condition,gge1bysubj,paired=TRUE)
```

```
##
##  Paired t-test
##
## data:  rawRT by condition
## t = 3.1093, df = 41, p-value = 0.003404
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##   35.85024 168.72119
## sample estimates:
## mean of the differences
##                102.2857
```

The correct linear model for the aggregated data here, which corresponds to the paired-test above, is the so-called **linear mixed model**:

```
library(lme4)
```

```
## Warning: package 'lme4' was built under R version 3.5.2
```

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

```
m1<-lmer(rawRT~so+(1|subject),gge1bysubj)
summary(m1)
```

```
## Linear mixed model fit by REML ['lmerMod']
```

```
## Formula: rawRT ~ so + (1 | subject)
##    Data: gge1bysubj
##
## REML criterion at convergence: 1103
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -1.2493 -0.5110 -0.1143  0.2429  3.4975
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  subject  (Intercept) 17970    134.1
##  Residual             22726    150.7
## Number of obs: 84, groups:  subject, 42
##
## Fixed effects:
##             Estimate Std. Error t value
## (Intercept)   420.22      26.43  15.901
## so            -51.14      16.45  -3.109
##
## Correlation of Fixed Effects:
##    (Intr)
## so 0.000
```

The model now is

$y = \beta_0 + \beta_1 \times so + u + \varepsilon$

where

- $\varepsilon \sim Normal(0, \sigma)$: Within-subject variability
- $u \sim Normal(0, \sigma_u)$: Between-subject variability

Assume $\sigma = 500$ and $\sigma_u = 200$, and $\beta_0 = 900$, and $\beta_1 = 50$. With these parameter values, we can generate fake data that has repeated measures for each of the two conditions.

Generating fake data for the linear mixed model

```
nsubj<-length(unique(gge1crit$subject))
N<-dim(gge1bysubj)[1]
rtfake<-rep(NA,N)
beta0<-900  ## Intercept
beta1<-50   ## slope
so<-gge1bysubj$so
## by subject adjustments:
u <- rnorm(nsubj,mean=0,sd=200)
sigma<-500

## Generate one fake data-set:
for(i in 1:N){
  rtfake[i] <- beta0 + beta1*so[i] + u[gge1bysubj[i,]$subject] +
    rnorm(1,mean=0,sd=sigma)
}
```

```
gge1bysubj$rtfake <- rtfake
summary(gge1bysubj$rtfake)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  -663.3   493.6   937.1   851.8  1219.9  2082.6
```

```
mfake<-lmer(rtfake~so+(1|subject),
            gge1bysubj)
summary(mfake)$coefficients
```

```
##               Estimate Std. Error   t value
## (Intercept) 851.847844   69.28148 12.295463
## so            6.023277   43.19373  0.139448
```

```
summary(mfake)$coefficients[2,3]
```

```
## [1] 0.139448
```

Exercise 2: Generate fake data (using the code shown above) to compute TDR

Generate fake data 100 times to compute TDR (power; here, you can assume that true $\beta_1 = 50$). Also assume that $\beta_0 = 600$, $\sigma = 300$ and $\sigma_u = 100$.

Exercise 3: Plot a power curve

Power for an experiment is not a single value. This is because power is a function of several variables: the standard deviation, the sample size, and the true effect size. Holding two of these variables constant (as shown in Exercise 2), we can compute power as a function of a *range* of possible/plausible effect sizes.

Concentrating only on True Discovery Rate or power now, compute power for a range of true values for β_1 ranging from 20-100 ms, in 10 ms increments.

Then plot a power curve, showing how power increases with increasing effect size.