Course Business

- LOTS of data on CourseWeb for this week
  - Cognitive Tutor use in schools
  - Word processing ("lexical decision") task

- Course evaluation (OMET) survey available
  - E-mailed to you and also on CourseWeb
Week 13: Data Management & Level-2 Variables

- Follow-Up & Distributed Practice
- Data Management in R
  - `rbind()`
  - `merge()`
  - `melt()`
- Level-2 Fixed & Random Effects
  - What do level-2 variables do?
  - Continuous or categorical?
    - Median splits
    - Extreme groups design
  - Good measurement
    - Reliability
    - Validity
Follow-Up

• Empirical logit last week—why didn’t everyone get model convergence error?
  • I had applied effects coding

Source Confusion Model

• Let’s model what causes people to make source confusions:
  • `model.Source <- glmer(SourceConfusion ~ AssocStrength*Strategy + (1+AssocStrength|Subject) + (1+Strategy|WordPair), data=sourceconfusion, family=binomial)`

• This looks bad! 😞
Follow-Up

- Empirical logit last week—why didn’t everyone get model convergence error?
- With default treatment coding, model does “converge” but produces nonsensical outcomes

\[
\exp(19.215) = \text{Odds of a source memory error are 221293404 times greater w/ maintenance rehearsal}
\]

- Again, because NO source errors in one condition
- Basically, infinity times more likely in the other
- So, bad model either way
- Always check your output—make sure it’s sensible!
A positive psychology lab is examining how feelings of subjective well being (SWB) vary over the course of the typical workweek. 60 participants come to the lab on Monday to participate in the first session and to get an app for their phones. We then use the app to poll the participants on their SWB (rated 1 to 7) once each of the remaining days of the week.

We run the following model:

```r
model1 <- lmer(SWB ~ 1 + DayOfWeek + SessionNumber + (1 + DayOfWeek + SessionNumber | Subject), data=positivepsych)
```

However, we receive the following error message:

fixed-effect model matrix is rank deficient so dropping 1 column / coefficient
Distributed Practice: The Final Chapter

• The lab disagrees on what we should do:
  • Andre says, “Let’s increase the `maxfun` parameter to allow the model more chances to converge.”
  • Bill says, “`DayOfWeek` and `SessionNumber` are perfectly confounded; we can fix this error by removing one of them.”
  • Caitlin says, “The random-effects structure is probably too complex. Let’s simplify it by removing the correlation parameters by using `l|Subject` instead of `l|Subject`”
  • Donghee says, “We can deal with the low sample size by computing the empirical logit and using that as our new DV.”
The lab disagrees on what we should do:

• Andre says, “Let’s increase the maxfun parameter to allow the model more chances to converge.”
  • This isn’t a failure to converge.
  • And, simply adding more iterations often does not fix convergence errors.

• Bill says, “DayOfWeek and SessionNumber are perfectly confounded; we can fix this error by removing one of them.”

• Caitlin says, “The random-effects structure is probably too complex. Let’s simplify it by removing the correlation parameters by using |Subject instead of ||Subject”
  • This would indeed simplify the random effects structure, but there is no reason to think it’s a problem—it’s not what the error message is about

• Donghee says, “We can deal with the low sample size by computing the empirical logit and using that as our new DV.”
  • The empirical logit is only relevant for a binomial DV
Distributed Practice: The Final Chapter

- An I/O psychologist models EmployeeBurnout as a function of YearsOnJob, tracked longitudinally for each of 500 employees.
- Which figure below corresponds to the assumptions made by each of these model formulae?
  - EmployeeBurnout ~ 1 + YearsOnJob + (1|Employee)
  - EmployeeBurnout ~ 1 + poly(YearsOnJob, degree=2) + (1|Employee)
  - EmployeeBurnout ~ 1 + YearsOnJob + (1 + YearsOnJob|Employee)
An I/O psychologist models EmployeeBurnout as a function of YearsOnJob, tracked longitudinally for each of 500 employees.

Which figure below corresponds to the assumptions made by each of these model formulae?:

- \( \text{EmployeeBurnout} \sim 1 + \text{YearsOnJob} + (1|\text{Employee}) \) (B)
- \( \text{EmployeeBurnout} \sim 1 + \text{poly}(\text{YearsOnJob}, \text{degree}=2) + (1|\text{Employee}) \) (C)
- \( \text{EmployeeBurnout} \sim 1 + \text{YearsOnJob} + (1 + \text{YearsOnJob}|\text{Employee}) \) (A)
Week 13: Data Management & Level-2 Variables

- Follow-Up & Distributed Practice
- Data Management in R
  - `rbind()`
  - `merge()`
  - `melt()`
- Level-2 Fixed & Random Effects
  - What do level-2 variables do?
  - Continuous or categorical?
    - Median splits
    - Extreme groups design
  - Good measurement
    - Reliability
    - Validity
Week 13: Data Management & Level-2 Variables

- Lots of data today on CourseWeb today (we’ll be talking about how to combine it):
  - **school1.csv**
  - **school2.csv**
  - **school3.csv**
    - Student math performance in three different schools
  - **tutoruse.csv**
    - Whether each classroom used a computer adaptive math tutor or not. Stored in a separate file so the experimenter is blind to this
  - **lexicaldecision.csv**
    - Cognitive task measuring word processing. See a string of letters, decide if it’s
  - **subtlexus.csv**
**`rbind()`**

- Paste together the rows from two (or more) dataframes to create a new one:
  
  ```r
  allschools <- rbind(school1, school2, school3)
  ```

- Useful when observations are spread across files
  
  - Or, to create a dataframe that consists of 2 subsets

- Requires these to have the same columns
  
  - Do before calculating new variables

- “More of the same”
Week 13: Data Management & Level-2 Variables

- Follow-Up & Distributed Practice
- Data Management in R
  - `rbind()`
  - `merge()`
  - `melt()`
- Level-2 Fixed & Random Effects
  - What do level-2 variables do?
  - Continuous or categorical?
    - Median splits
    - Extreme groups design
  - Good measurement
    - Reliability
    - Validity
merge()

- Sometimes different files/dataframes contain different variables relevant to the same observations
- Common scenario in mixed effects models context: Level-2 variables are in a different file than Level-1 measurements

### allschools: 1 row per student

<table>
<thead>
<tr>
<th>School</th>
<th>Classroom</th>
<th>Student</th>
<th>HoursOfStudy</th>
<th>StudentSES</th>
<th>Pretest</th>
<th>Posttest</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jefferson</td>
<td>C001</td>
<td>S0001</td>
<td>1</td>
<td>0.7803573</td>
<td>0.5137800</td>
<td>1.5399431</td>
</tr>
<tr>
<td>Jefferson</td>
<td>C001</td>
<td>S0002</td>
<td>3</td>
<td>-0.2153623</td>
<td>0.2634907</td>
<td>1.3080398</td>
</tr>
<tr>
<td>Jefferson</td>
<td>C001</td>
<td>S0003</td>
<td>0</td>
<td>0.1290432</td>
<td>0.5232901</td>
<td>1.4550667</td>
</tr>
<tr>
<td>Jefferson</td>
<td>C001</td>
<td>S0004</td>
<td>3</td>
<td>1.6873593</td>
<td>0.3640230</td>
<td>0.6022264</td>
</tr>
<tr>
<td>Jefferson</td>
<td>C001</td>
<td>S0005</td>
<td>3</td>
<td>0.2196517</td>
<td>0.7866884</td>
<td>1.2517459</td>
</tr>
<tr>
<td>Jefferson</td>
<td>C001</td>
<td>S0006</td>
<td>5</td>
<td>-0.2931509</td>
<td>1.2862659</td>
<td>1.6046956</td>
</tr>
</tbody>
</table>

### tutoruse.csv:

<table>
<thead>
<tr>
<th>Classroom</th>
<th>Tutor</th>
</tr>
</thead>
<tbody>
<tr>
<td>C001</td>
<td>No</td>
</tr>
<tr>
<td>C002</td>
<td>Yes</td>
</tr>
<tr>
<td>C003</td>
<td>No</td>
</tr>
<tr>
<td>C004</td>
<td>Yes</td>
</tr>
<tr>
<td>C005</td>
<td>No</td>
</tr>
<tr>
<td>C006</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Each class has only one row—did this class use the tutor or not?
merge()

- Sometimes different files/dataframes contain different variables relevant to the same observations
- Common scenario in mixed effects models context: Level-2 variables are in a different file than Level-1 measurements

<table>
<thead>
<tr>
<th>Subject</th>
<th>Word</th>
<th>PrevTrials</th>
<th>RT</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>panther</td>
<td>0</td>
<td>703.877</td>
</tr>
<tr>
<td>S1</td>
<td>drive</td>
<td>1</td>
<td>532.387</td>
</tr>
<tr>
<td>S1</td>
<td>monorail</td>
<td>2</td>
<td>731.882</td>
</tr>
<tr>
<td>S13</td>
<td>peony</td>
<td>0</td>
<td>808.392</td>
</tr>
<tr>
<td>S13</td>
<td>monorail</td>
<td>1</td>
<td>489.479</td>
</tr>
<tr>
<td>S13</td>
<td>aardvark</td>
<td>2</td>
<td>875.799</td>
</tr>
</tbody>
</table>

**lexicaldecision.csv:**
1 row per **trial**
Each word appears in multiple rows

<table>
<thead>
<tr>
<th>Word</th>
<th>WordFreq</th>
</tr>
</thead>
<tbody>
<tr>
<td>the</td>
<td>6.1766</td>
</tr>
<tr>
<td>to</td>
<td>6.0632</td>
</tr>
<tr>
<td>a</td>
<td>6.0175</td>
</tr>
<tr>
<td>you</td>
<td>6.3293</td>
</tr>
<tr>
<td>and</td>
<td>5.8343</td>
</tr>
<tr>
<td>it</td>
<td>5.9839</td>
</tr>
</tbody>
</table>

**subtlexus.csv:**
Each word has only **one** row with its frequency
merge()

- Sometimes different files/dataframes contain different variables relevant to the same observations.
- Common scenario in mixed effects models context: Level-2 variables are in a different file than Level-1 measurements.

1 row per trial
Each subject has multiple rows

<table>
<thead>
<tr>
<th>SUBJECT</th>
<th>ITEM</th>
<th>CONDITION</th>
<th>CORRECT</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Sentence1</td>
<td>Active</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>Sentence2</td>
<td>Passive</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>Sentence3</td>
<td>Active</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>Sentence4</td>
<td>Passive</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>Sentence1</td>
<td>Active</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>Sentence2</td>
<td>Passive</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>Sentence3</td>
<td>Active</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>Sentence4</td>
<td>Passive</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SUBJECT</th>
<th>READINGSPAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
</tr>
</tbody>
</table>

Each subject has only one row with his or her Reading Span score.
merge()

- "Look up word frequency from the other dataframe"
- We can combine these dataframes if they have at least one column in common
- **Word** tells us which word was presented on an individual trial, and it also identifies the word in our database of word frequency

**lexicaldecision.csv:**
- 1 row per **trial**
- Each word appears in multiple rows

**subtlexus.csv:**
- Each word has only **one** row with its frequency
merge()

- \texttt{lexdec2 <- merge(lexicaldecision, subtlexus, by='Word')}
  - New dataframe has both the columns from \texttt{lexicaldecision} (Subject, PrevTrials, RT) and the columns from \texttt{subtlexus} (WordFreq)
  - Matches the observations using the \texttt{Word} column
merge()

- `lexdec2 <- merge(lexicaldecision, subtlexus, by='Word')`
  - New dataframe has both the columns from `lexicaldecision` (Subject, PrevTrials, RT) and the columns from `subtlexus` (WordFreq)
  - Matches the observations using the **Word** column
merge() – Renaming Columns

- What if the columns have different names?
- **Item** in `lexicaldecision` tells us which **Word** to look for in `subtlexus` ... but R doesn't know that!
- Easy solution is to rename the column
  ```r
  colnames(lexicaldecision)[colnames(lexicaldecision) == 'Item'] <- 'Word'
  ```
- Look at the column names for `lexicaldecision`
- Find the one called “Item”
- Replace that name with “Word”

Then do the `merge()`
merge() – all.x and all.y

- nrow(lexicaldecision) 2040
- nrow(lexdec2) 1800
- Six words don’t have a frequency measurement
- Default behavior of `merge()` is to drop rows that can’t be matched (inner join)
- `lexdec2 <- merge(lexicaldecision, subtlexus, by='WORD', all.x=TRUE)`

Keep the rows in `lexicaldecision` where we can’t find the matching WORD in `subtlexus`

WordFreq will be NA in these rows
merge() – all.x and all.y

- nrow(lexicaldecision) 2040
- nrow(lexdec2) 1800
- Six words don’t have a frequency measurement
- Default behavior of `merge()` is to drop rows that can’t be matched (inner join)
- `lexdec2 <- merge(lexicaldecision, subtlexus, by='WORD', all.x=TRUE, all.y=TRUE)`

Adding all.y=TRUE would also include rows for all of the words in the word frequency database, even the words that weren’t used in our experiment.

We DON’T need or want that.
merge() – Matching by Multiple Columns

- Sometimes, one column isn’t enough to uniquely match things across files/dataframes
- Can use multiple columns in `merge()`
  ```r
  lexdec2 <- merge(
    lexicaldecision, subtlexus,
    by=c('Word', 'Country'))
  ```
- This is a logical AND. Has to match both Word and Country

Imagine doing our task in both the US and UK. Word frequency differs somewhat between American English & British English, so now we need both **Word** and **Country** to look up the frequency.
merge() – Troubleshooting

- If you leave out `by=`:
  - R tries to figure out the matching columns on its own
- If you leave out `by=` and NO columns match:
  - R creates a massive dataframe in which every row in dataframe 1 is paired with every row in dataframe 2
    - `nrow(trials) * nrow(subtlexus)`
- Symptoms:
  - You end up with an enormous dataframe with tens of thousands of observations
  - The `merge()` takes so long that it seems like your computer has frozen
- Hit STOP and check your `merge()` call
**merge() – Practice!**

- Remember our math tutoring data?:

<table>
<thead>
<tr>
<th>School</th>
<th>Classroom</th>
<th>Student</th>
<th>HoursOfStudy</th>
<th>StudentSES</th>
<th>Pretest</th>
<th>Posttest</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jefferson</td>
<td>C001</td>
<td>S0001</td>
<td>1</td>
<td>0.7803573</td>
<td>0.5137800</td>
<td>1.5399431</td>
</tr>
<tr>
<td>Jefferson</td>
<td>C001</td>
<td>S0002</td>
<td>3</td>
<td>-0.2153623</td>
<td>0.2634907</td>
<td>1.3080398</td>
</tr>
<tr>
<td>Jefferson</td>
<td>C001</td>
<td>S0003</td>
<td>0</td>
<td>0.1290432</td>
<td>0.5232901</td>
<td>1.4550667</td>
</tr>
<tr>
<td>Jefferson</td>
<td>C001</td>
<td>S0004</td>
<td>3</td>
<td>1.6873593</td>
<td>0.3640230</td>
<td>0.6022264</td>
</tr>
<tr>
<td>Jefferson</td>
<td>C001</td>
<td>S0005</td>
<td>3</td>
<td>0.2196517</td>
<td>0.7866884</td>
<td>1.2517459</td>
</tr>
<tr>
<td>Jefferson</td>
<td>C001</td>
<td>S0006</td>
<td>5</td>
<td>-0.2931509</td>
<td>1.2862659</td>
<td>1.6046956</td>
</tr>
</tbody>
</table>

- Use `merge()` to add the tutor data from `tutoruse` to `allschools`
merge() – Practice!

- Remember our math tutoring data?:

Use `merge()` to add the tutor data from `tutoruse` to `allschools`

```
allschools <- merge(tutoruse, allschools, by='Classroom')
```
Week 13: Data Management & Level-2 Variables

- Follow-Up & Distributed Practice
- Data Management in R
  - `rbind()`
  - `merge()`
  - `melt()`
- Level-2 Fixed & Random Effects
  - What do level-2 variables do?
  - Continuous or categorical?
    - Median splits
    - Extreme groups design
  - Good measurement
    - Reliability
    - Validity
melt()

- Need to install package **reshape2**

- For **lmer()**, each observation needs its own row
  - “long” format
    - Time 1 row
    - Time 2 gets a separate row

- Sometimes data comes to us in “wide” format
  - Each repeated measure is a different *column* in the same row

<table>
<thead>
<tr>
<th>SUBJECT</th>
<th>SESSION</th>
<th>MATHSCORE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Pre-test</td>
<td>-1.27</td>
</tr>
<tr>
<td>1</td>
<td>Post-test</td>
<td>0.33</td>
</tr>
<tr>
<td>2</td>
<td>Pre-test</td>
<td>0.25</td>
</tr>
<tr>
<td>2</td>
<td>Post-test</td>
<td>1.11</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Student</th>
<th>Pretest</th>
<th>Posttest</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>S0001</td>
<td>0.5137800</td>
</tr>
<tr>
<td>2</td>
<td>S0002</td>
<td>0.2634907</td>
</tr>
</tbody>
</table>
melt()

- Need to install package `reshape2`
  - Then do `library(reshape2)`

- `melt()` turns “wide” data into “long” data

- `melteddata <- melt(allschools, measure.vars=c('Pretest', 'Posttest'), Pretest and Posttest are the columns that we want to convert into separate observations (often, repeated measures on the same individual)
melt()

- Need to install package `reshape2`
  - Then do `library(reshape2)`

- `melt()` turns “wide” data into “long” data

- `melteddata <- melt(allschools, measure.vars=c('Pretest', 'Posttest'))`,

But, we need some way to preserve student, school, & classroom IDs and SES/hours of study.

Don’t want to treat Student SES as though it were the outcome from a 3rd session!
melt()

- Need to install package reshape2
  - Then do `library(reshape2)`

- `melt()` turns “wide” data into “long” data

```r
melteddata <- melt(allschools,
  measure.vars=c('Pretest', 'Posttest'),
  id.vars=c('Student', 'Classroom', 'School',
            'StudentSES', 'HoursOfStudy'),
)
```

**id.vars** are columns that should stay as separate columns:
- IDs for students, classrooms, schools
- Between-subjects variables that are constant: StudentSES and HoursOfStudy
melt()

- Need to install package `reshape2`
  - Then do `library(reshape2)`
- `melt()` turns “wide” data into “long” data
  ```r
  melteddata <- melt(allschools, measure.var=c('Pretest', 'Posttest'), id.var=c('Student', 'Classroom', 'School', 'StudentSES', 'HoursOfStudy'), variable.name='Session')
  ```
  We’re creating a new variable to distinguish between the pretest & posttest sessions

Let’s call it `Session` (but could be anything you want)
melt(): The Results

- summary(melteddata)

<table>
<thead>
<tr>
<th>School</th>
<th>Classroom</th>
<th>Student</th>
<th>Session</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jefferson:600</td>
<td>C001 : 60</td>
<td>S0001 : 2</td>
<td>Pretest</td>
<td>:900</td>
</tr>
<tr>
<td>Hoover :600</td>
<td>C002 : 60</td>
<td>S0002 : 2</td>
<td>Posttest:900</td>
<td></td>
</tr>
<tr>
<td>Harding :600</td>
<td>C003 : 60</td>
<td>S0003 : 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>C004 : 60</td>
<td>S0004 : 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>C005 : 60</td>
<td>S0005 : 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>C006 : 60</td>
<td>S0006 : 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(Other):1440</td>
<td>(Other):1788</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Now we have 2 rows per student: A “Pretest” row and a “Posttest” row
- Can now include Session as a predictor variable in lmer
- This column is named Session because that’s what we set the variable.name argument to:
  - melteddata <- melt(allschools, measure.vars=c('Pretest', 'Posttest'), id.vars=c('Student', 'Classroom', 'School', 'StudentSES', 'HoursOfStudy'), variable.name='Session')
melt(): The Results

- `summary(melteddata)`

- DV is just called `value` by default because R has no way of knowing what it represents

- We can change that:
  ```r
  melteddata <- melt(allschools, 
                    measure.vars=c('Pretest', 'Posttest'), 
                    id.vars=c('Student', 'Classroom', 'School', 'StudentSES', 'HoursOfStudy'), 
                    variable.name='Session', 
                    value.name='MathScore')
  ```
melt(): Extra Practice!

- If you completed the `merge()` practice earlier, `allschools` will also have a `Tutor` column that we want to preserve when we convert to long format.
- Old `melt()` was:
  ```r
  melteddata <- melt(allschools, 
    measure.vars=c('Pretest', 'Posttest'),
    id.vars=c('Student', 'Classroom', 'School', 'StudentSES', 'HoursOfStudy'),
    variable.name='Session',
    value.name='MathScore')
  ```
- Where should we add `Tutor` in the `melt()` call?
melt(): Extra Practice!

- If you completed the `merge()` practice earlier, `allschools` will also have a **Tutor** column that we want to preserve when we convert to long format.
- New `melt()` is:
  
melteddata <- melt(allschools, 
  measure.vars=c('Pretest', 'Posttest'),
  id.vars=c('Student', 'Classroom', 'School',
   'StudentSES', 'HoursOfStudy', 'Tutor'),
  variable.name='Session',
  value.name='MathScore')

  *id.vars are the columns that should stay as-is*
melt()

- Need to install package **reshape2**
  - Then do `library(reshape2)`
- `melt()` turns “wide” data into “long” data
- Also a corresponding function, `cast()`, to turn “long” format data into “wide” format data
  - Analogy: Casting molten steel
- Other, newer package for reshaping data: **dplyr**
Summary

- Data is already in one data frame but you need to rearrange it:

- Same variables in more than one file:

- Different variables in more than one file:
Summary

- Data is already in one data frame but you need to rearrange it:
  - `melt()`

- Same variables in more than one file:
  - `rbind()`

- Different variables in more than one file:
  - `merge()`
Week 13: Data Management & Level-2 Variables

- Follow-Up & Distributed Practice
- Data Management in R
  - `rbind()`
  - `merge()`
  - `melt()`
- Level-2 Fixed & Random Effects
  - What do level-2 variables do?
  - Continuous or categorical?
    - Median splits
    - Extreme groups design
  - Good measurement
    - Reliability
    - Validity
Let’s consider one model of our lexical decision data:

- `model1 <- lmer(RT ~ 1 + PrevTrials + (1|Subject) + (1|Word), data=lexdec2)`

Hierarchical linear model notation for this:

- **Lv.2 (Item):** $B_k = u_{00(0k)}$
  - Level 2 model predicts the effect of item $k$
  - Could substitute random intercept into the level 1 model

- **Lv.2 (Subj.):** $B_j = u_{00(j0)}$

- **Lv.1 (Trial):** $Y_{i(jk)} = \gamma_{000} + \gamma_{100} \text{PrevTrials} + B_j + B_k + \epsilon_{i(jk)}$
  - Intercept
  - # of previous trials seen
  - Subject
  - Item
  - Error
Level-2 Fixed and Random Effects

- Now let's add a fixed effect of word frequency:
  - `model2 <- lmer(RT ~ 1 + PrevTrials + WordFreq + (1|Subject) + (1|Word), data=lexdec2)`

- Which level does this characterize?:
  - Lv.2 (Item): $B_k = u_{00(0k)}$
  - Lv.2 (Subj.): $B_j = u_{00(j0)}$
  - Lv.1 (Trial): $Y_{i(jk)} = \gamma_{000} + \gamma_{100} \text{PrevTrials} + B_j + B_k + e_{i(jk)}$

  Level 2 model predicts the effect of item $k$
  Could substitute random intercept into the level 1 model
Level-2 Fixed and Random Effects

- Now let’s add a fixed effect of word frequency:
  - `model2 <- lmer(RT ~ 1 + PrevTrials + WordFreq + (1|Subject) + (1|Word), data=lexdec2)`

- Which level does this characterize?:
  - Lv.2 (Item): $B_k = \gamma_{200} \text{WordFreq} + u_{00(0k)}$
  - Lv.2(Subj.): $B_j = u_{00(j0)}$
  - Lv.1(Trial): $Y_{i(jk)} = \gamma_{000} + \gamma_{100} \text{PrevTrials} + B_j + B_k + e_{i(jk)}$
    - Intercept
    - # of previous trials seen
    - Subject
    - Item
    - Error
What Changes?

- Random item variance is greatly reduced.
- Word frequency accounts for a lot of the variance among items.
- Word frequency explains a lot of the “Item k” effect we’re substituting into the level 1 equation. No longer just a random intercept.
What Didn’t Change?

- Level 1 fixed effect (PrevTrials) and error term essentially unchanged.
- Doesn’t matter what explains the “Item k” effect; still substituting into the same Lv 1 model.
- Note that WordFreq & PrevTrials effects are slightly correlated (due to random sampling of item orders); otherwise, there’d be no change.
What Didn’t Change?

- Estimated variance in subject intercept also essentially the same
- Explaining where the “Item k” effect comes from doesn’t change the “Subject j” effect
Week 13: Data Management & Level-2 Variables

- Follow-Up & Distributed Practice
- Data Management in R
  - rbind()
  - merge()
  - melt()
- Level-2 Fixed & Random Effects
  - What do level-2 variables do?
  - Continuous or categorical?
    - Median splits
    - Extreme groups design
  - Good measurement
    - Reliability
    - Validity
Continuous or Categorical Predictors?

- In ANOVA, subject & item differences typically examined as **categorical** variables
- e.g. **median split**:
  - `median(lexcdec2$WordFreq, na.rm=TRUE) = 3.30`
  - Word frequencies above the median are in category A and words below it are in category B
Continuous or Categorical Predictors?

- In ANOVA, subject & item differences typically examined as **categorical** variables
- e.g. **median split**:
  - \( \text{median(lexcdec2$WordFreq, na.rm=TRUE)} = 3.30 \)
  - Word frequencies above the median are in category A and words below it are in category B
Evaluating Median Splits

- Median splits are **noisy** and **discard info.**
- Ignores all within-category variation

`pomegranate` (WF: 1.1461) and `glasses` (WF: 3.2279) are considered equally "low-frequency" words by the median split, as they are both on the same side of a median split in the graph.
Evaluating Median Splits

- Median splits are noisy and discard info.
  - Ignores all within-category variation
  - High probability of misclassification

If our measures of word frequency were even slightly off, these words could have ended up in the opposite categories!

- glasses (WF: 3.2279)
- chair (WF: 3.400)
Evaluating Median Splits

- Median splits are **noisy** and **discard info**.
  - Ignores all within-category variation
  - High probability of misclassification
- Greatly reduces power and estimated effect size (Cohen, 1983)
Median splits are noisy and discard info.
- Ignores all within-category variation
- High probability of misclassification
- Greatly reduces power and estimated effect size (Cohen, 1983)

Also, comparing two categories can’t tell us about the form of the relationship (as polynomial contrasts can)

If continuous variation (in word frequency, second language proficiency, etc.) measured, better to include it in the model
Week 13: Data Management & Level-2 Variables

- Follow-Up & Distributed Practice
- Data Management in R
  - `rbind()`
  - `merge()`
  - `melt()`
- Level-2 Fixed & Random Effects
  - What do level 2 variables do?
  - Continuous or categorical?
    - Median splits
    - Extreme groups design
  - Good measurement
    - Reliability
    - Validity
Extreme Group Designs

- In some cases, we might deliberately sample only very low- and very high-frequency words
  - Extreme group design
- Now, we don’t know what the full relation is

Here there be dragons
In some cases, we might deliberately sample only very low- and very high-frequency words

- Extreme group design

Now, we don’t know what the full relation is
In some cases, we might deliberately sample only very low- and very high-frequency words

- Extreme group design

Now, we don’t know what the full relation is
**Extreme Group Designs**

- In some cases, we might deliberately sample only very low- and very high-frequency words
  - Extreme group design
- Now, we don’t know what the full relation is
  - *Should* treat this as a categorical variable (reflects design)
**Extreme Group Designs: Evaluation**

- May **overestimate** effect size
- Still, better than median splits if you want to do a categorical design (Conway et al., 2005)
  - e.g., you only care whether a difference exists (not its size / shape)

Here there be dragons
**Breakpoints**

- When you have a continuous variable, but you think there’s a qualitative shift at some point in the range
  - e.g., below vs above the poverty line
- Add a categorical variable that represents whether or not you’re above the point at which the shift happens

- **Main effect of breakpoint only** – single shift downward but same slope
- **Main effect of breakpoint & interaction** – slopes also change
Week 13: Data Management & Level-2 Variables

- Follow-Up & Distributed Practice
- Data Management in R
  - `rbind()`
  - `merge()`
  - `melt()`
- Level-2 Fixed & Random Effects
  - What do level 2 variables do?
  - Continuous or categorical?
    - Median splits
    - Extreme groups design
- Good measurement
  - Reliability
  - Validity
Good Measurement: Reliability

- Suppose we find that a measure of working memory is unrelated to people’s moral judgments
  - Maybe these are truly unrelated
  - Or, maybe we just failed to accurately measure WM and/or moral reasoning

- Not all measures are good measures
  - Measures may be noisy
  - Measures may not measure a stable or meaningful characteristic of people/items/schools
Good Measurement: Reliability

- Good measures produce consistent scores
  - Across times (test-retest reliability)
  - Across items (internal consistency)
  - Across judges (inter-rater reliability)
- Shows you’re measuring something real

- If measures can’t even predict themselves, they can’t predict anything else!

$r = .77$ Good!

$r = .16$ Bad!
Week 13: Data Management & Level-2 Variables

- Follow-Up & Distributed Practice
- Data Management in R
  - `rbind()`
  - `merge()`
  - `melt()`
- Level-2 Fixed & Random Effects
  - What do level 2 variables do?
  - Continuous or categorical?
    - Median splits
    - Extreme groups design
- Good measurement
  - Reliability
  - Validity
Good Measurement: Validity

- Even if we have a *reliable* measure, no guarantee it measures the thing we *think* it measures
  - You’re measuring *something*, but what is it?
  - *Examples of tests that produce consistent results but don’t measure what we want:*
Good Measurement: Validity

- Valid measures should show (among other things):
  - **Convergent validity**: Correlate with *other* measures of this construct

Operation Span task: Remember words while verifying equations

3 x 4 = 12
(T / F)?

Reading Span task: Remember words while verifying sentences

An official who manages a state is called a governor.
(T / F)?

Here, two tasks designed to measure working memory correlate
Good Measurement: Validity

- Valid measures should show (among other things):
  - **Convergent validity:** Correlate with *other* measures of this construct
  - **Divergent validity:** *Don’t* correlate with things that are supposed to be different
    - If “working memory” task correlates with years of education or socioeconomic status, might not be measuring what we thought
Good Measurement: Validity

- Valid measures should show (among other things):
  - **Convergent validity**: Correlate with other measures of this construct
  - **Divergent validity**: Don’t correlate with things that are supposed to be different
    - Do higher Working Memory scores predict second language learning just because subjects who are “smarter” or more motivated do well on both tasks?
    - Or is this unique to WM?
    - Measuring only 1 construct makes it difficult to tell where the locus of an effect lies