The Data
Who's in the data?
Learning individual words
Hardest and Easier Words
More Practice

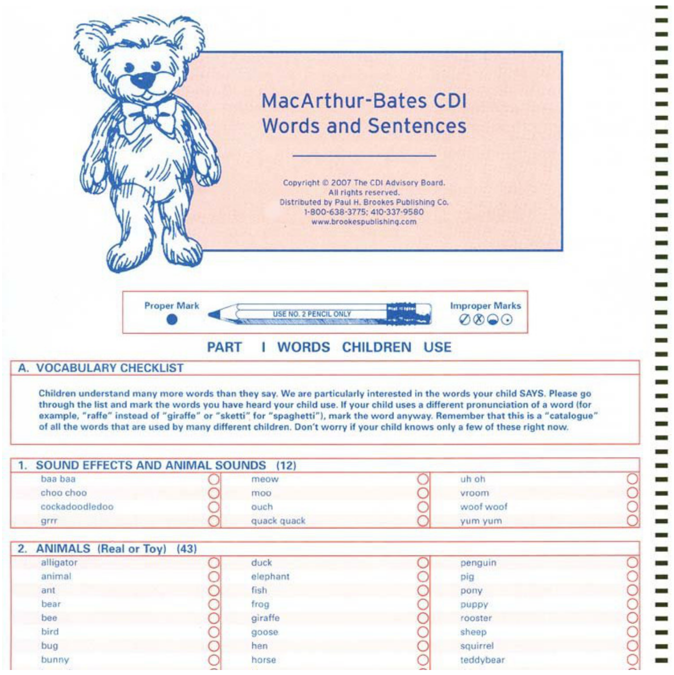
Introduction to the Tidyverse and Functions

Your reproducible lab report: Before you get started, download the R Markdown template for this lab. Remember all of your code and answers go in this document:

```
download.file("https://dyurovsky.github.io/85309/post/rmd/lab3.Rmd",
              destfile = "lab3.Rmd")
```

Over the course of the previous labs, you've gotten a chance to get your feet with some of the data munging and plotting tools that are in the Tidyverse. The goal of this is to give you a better understanding for what these tools do, and to give you practice using them. A second purpose is to introduce you to writing your own functions in R. This will be important for understanding the simulation lectures and labs, and also will make your R-writing life way smoother and more flexible.

We're going to be working with some data from Wordbank (http://wordbank.stanford.edu/), an open database of parent-report measures of their children's vocabularies. The data come from a standardized form called the MacArthur-Bates Communicative Development Inventory (https://mb-cdi.stanford.edu/) that researchers in the US and dozens of other countries have been using for over 40 years to study the processes of language learning and their relationship to other aspects of children's development. The forms consist of a checklist of words that children typically learn at different ages, and parents are asked to indicate which of them their children already know. We're going to look at data from the English form for older children– you can see what the first page looks like below.



MacArthur-Bates CDI
Words and Sentences

Copyright © 2007 The CDI Advisory Board.
All rights reserved.
Distributed by Paul H. Brookes Publishing Co.
1-800-638-3775; 410-337-9580
www.brookespublishing.com

Proper Mark ●
USE NO. 2 PENCIL ONLY
Improper Marks

PART I WORDS CHILDREN USE

A. VOCABULARY CHECKLIST

Children understand many more words than they say. We are particularly interested in the words your child SAYS. Please go through the list and mark the words you have heard your child use. If your child uses a different pronunciation of a word (for example, "raffe" instead of "giraffe" or "sketti" for "spaghetti"), mark the word anyway. Remember that this is a "catalogue" of all the words that are used by many different children. Don't worry if your child knows only a few of these right now.

1. SOUND EFFECTS AND ANIMAL SOUNDS (12)

baa baa	○	meow	○	uh oh	○
choo choo	○	moo	○	vroom	○
cockadoodledoo	○	ouch	○	woof woof	○
grrr	○	quack quack	○	yum yum	○

2. ANIMALS (Real or Toy) (43)

alligator	○	duck	○	penguin	○
animal	○	elephant	○	pig	○
ant	○	fish	○	pony	○
bear	○	frog	○	puppy	○
bee	○	giraffe	○	rooster	○
bird	○	goose	○	sheep	○
bug	○	hen	○	squirrel	○
bunny	○	horse	○	teddybear	○

In this lab we will explore the data using the `dplyr` package and visualize it using the `ggplot2` package for data visualization. These are both parts of the `tidyverse` ecosystem.

Let's load the `tidyverse` package.

```
library(tidyverse)
```

The Data

The dataset consists of parents' judgments about whether their children knew each of the words on MCDI. Let's load it and take a look at it.

```
wordbank <- read_csv("https://dyurovsky.github.io/85309/data/lab3/wordbank.csv")
```

The data set `wordbank` that shows up in your workspace is a **tibble**, with each row representing an *observation* and each column representing a *variable*. For this data set, each *observation* is a single parent judgment.

To view the names of the variables, type the command

```
names(wordbank)
```

The **codebook** (description of the variables) is included below.

- `id` : A unique identifier for each child
- `age` : The child's age in months
- `gender` : The child's gender as reported by the parent
- `category` : The type of the word being judged (e.g. animals, people, time_words)
- `word` : The word being judged

- `knows` : A logical (TRUE/FALSE) indicating whether the parent reported that their child produces this word

Who’s in the data?

Let’s find out how many children we have data about. To do this, we’ll use three important functions from the tidyverse: `group_by` , `summarise` , and `distinct` . Intuitively, what I want to do is ask a question like “How many children of each age and gender are in my dataset?” You can think of `group_by` as defining who I want to ask my question about, and `summarise` as defining what question I want to ask. One easy way to ask “how many” is to use the `n` function which takes no arguments and just counts how many rows I have within each group.

The problem is, I have multiple judgments for each child, so if I use just ask “how many do I have of each age and gender” I will find out how many judgments I have–not how many children. To solve this, I’ll first use `distinct` to say I only want to keep 1 row for each id.

So the code looks like this:

```
age_gender_count <- wordbank %>%
  group_by(age, gender) %>%
  distinct(id) %>%
  summarise(count = n())

age_gender_count
```

Can you tell how many of each age/gender are in the dataset?

Exercise 1 Use this same logic to find out the answer to 3 more descriptive questions: How many words are on the form? How many categories? Which category has the most words in it?

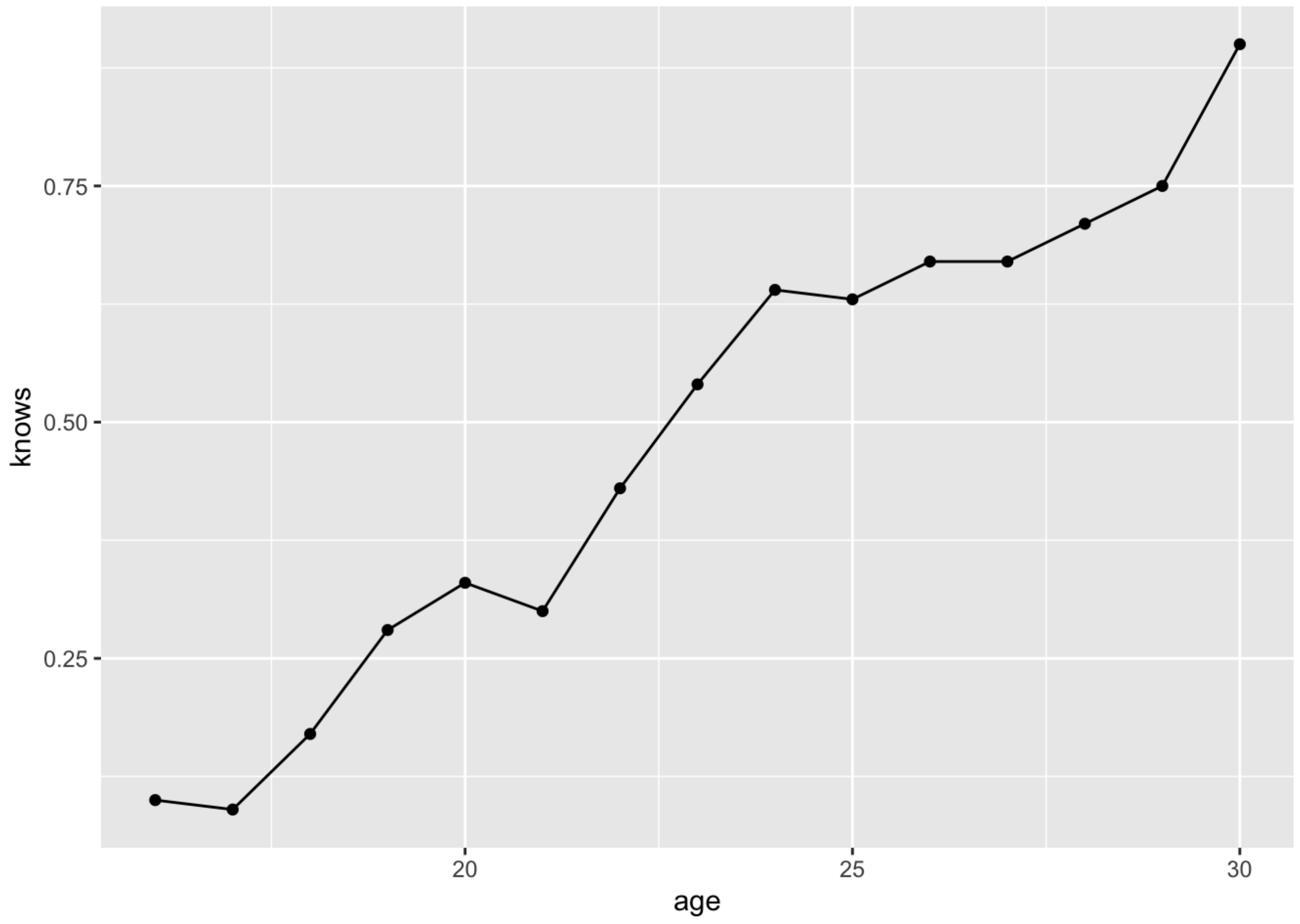
Learning individual words

We can use this dataset to figure out which words are easier to learn and which words are harder. Let’s try this out with the word “tiger.” To do this, we’ll need another tidyverse function: `filter` . With this function, we can keep just the rows in a dataset that match a logical expression. In this case, I want to filter to just the rows where the *word* column has the value “tiger”.

Once I do that, I can make a plot of proportion of children at each age who know the word I’m interested in. I’ll do that with the `ggplot` function, adding a the point and line geoms to get an acquisition curve.

```
tiger <- wordbank %>%
  filter(word == "tiger") %>%
  group_by(age) %>%
  summarise(knows = mean(knows))

ggplot(tiger, aes(x = age, y = knows)) +
  geom_point() +
  geom_line()
```



Exercise 2 See if you can find two other words on the form–one that is easier to learn than “tiger” and one that is harder to learn. Prove it to me by making a plot that has learning trajectories for all 3 words (maybe using the `color` aesthetic). Make sure you think about what you want to `group_by` and what you want to `filter` to. You might find the `distinct` function helpful for seeing what words are on the form.

Hardest and Easier Words

Now that you saw that some words are easier and some are harder, let’s find the easiest and hardest words. To do that, we want to find the word that is known by the smallest (hardest) or largest (easiest) proportion of children.

Exercise 3 Make a tibble called `word_difficulty` that has a row for each word and column called `prop` that tells you what proportion of the children in the sample know it. What's the hardest word? What's the easiest word? Do these make sense to you? You might find that `arrange` is helpful for sorting the `prop` column to make reading it easier.

Now let's write a function called `hardest_word` that can tell us the hardest word a particular child knows. To write a function in R, you define a new variable just like you've done before (say for `word_difficulty`) but you assign to it a function and specify the parameters it takes. Our function will take a parameter called `child` that will be one of the `id`s in our `wordbank`s dataset. The curly braces specify that everything inside them is part of the function. That looks likes this:

```
hardest_word <- function(child) {

}
```

What we want this function to do is to start with the `wordbank` dataset, `filter` to just the rows for this child and the words that this child knows, and then find the word from that set that is the hardest according to our `word_difficulty` tibble. This will require us to use two new functions from `tidyverse`: `left_join` and `pull`. `left_join` takes two tables and joins them together by all of the rows that match on a set of specified columns. We'll need this to add the difficulties from `word_difficulty` to the filtered table. `pull` takes a column out of a tibble–we'll need this at the end to get the word out. Your template has a skeleton for this function with the `left_join` and `pull` already written–you just need to write the rest.

When you have it working, it should look like this:

```
hardest_word("129250")
```

```
## [1] "if"
```

```
hardest_word("130375")
```

```
## [1] "basement"
```

Exercise 4 Write the `hardest_word` function and show me that it works by trying it for two children: 129277 who is 19 months old and 129579 who is 30 months old. Do their hardest words make sense?

More Practice

Exercise 5 A well known result in the language acquisition literature is that girls tend to learn language faster than boys. Can you see if this is true in the wordbank data? Make a plot like the one you did for Exercise 2 but with the acquisition trajectories for boys and girls rather than for 3 words.

Exercise 6 Are there any children in the wordbank dataset who know every word? If there is, tell me the age and id of the youngest one. Otherwise, tell me the age and id of the child who knows the most words.

Exercise 7 Write a `youngest_age` function that works the same way as your `hardest_word` function but instead takes in a word and returns the age of the youngest child in the database who knows that word. You won't need `left_join` for this one, but you will need `pull` Tell me the age of the youngest child who knows "wish".