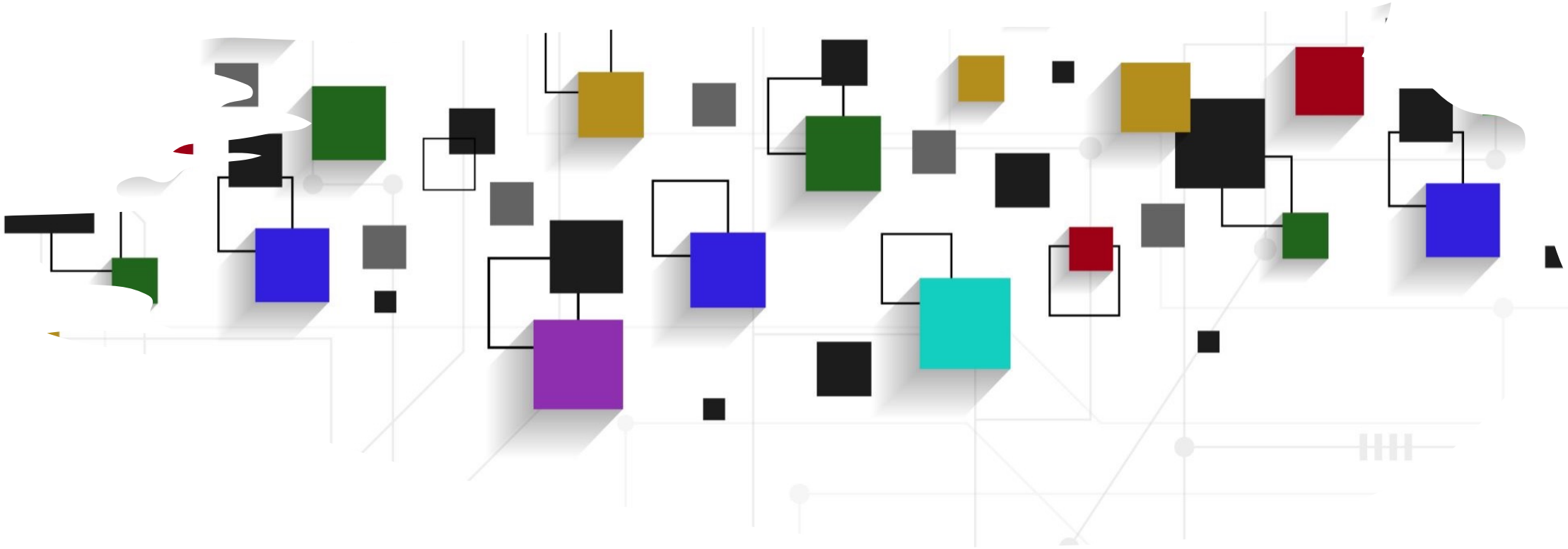


CogLab: Manipulate/Infer

WEEK 10

what's coming up

Week 10 / Making Inferences

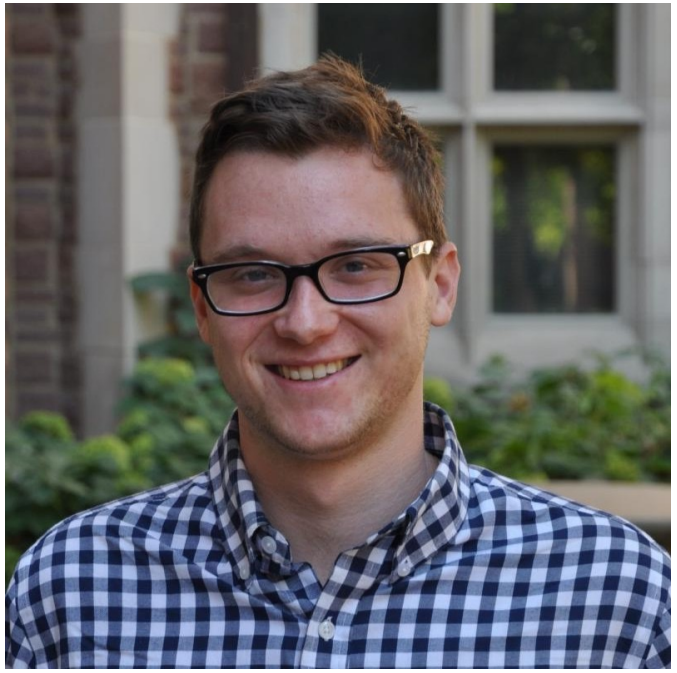
- **Class survey (October: Extra Credit)**
  Nov 5 | 1 pts
- **W10 Quiz**
  Nov 5 | 5 pts
- **Formative Assignment #2 (R Descriptive)**
  Nov 5 | 20 pts
- **Milestone 6: Pre-Registration**
  Nov 5 | 20 pts
- **Milestone 6: Self/Peer Assessment**
  Nov 5 | 1 pts

logistics: formative assignment #2

- descriptive statistics and plotting in R
- you will need to use tidyverse functions
- due Nov 5 (first draft worth 2%, second worth 8%)

Nov 7: guest speaker

- Dr. Kyle Featherston
- Ph.D., Psychological and Brain Sciences
- Research Program Director, Columbia University School of Nursing
- available for one-on-one career meetings:
  - 9 am – 10 am
  - 1 pm – 3 pm
- sign up here



pre-registration

- due Nov 5
- plan your data collection + analyses
- submit pilot data

1. **Data Collection**: Have any data been collected for this study already?
2. **Main Question**: What is the main question being asked or hypothesis being tested in this study?
3. **Dependent Variable(s)**: Describe the key dependent variable(s) specifying how they will be measured.
4. **Condition(s)**: How many and which conditions will participants be assigned to? Please include an example trial of each type of condition you have in your experiment. Please also specify which independent variable will be within-participants or between-participants.
5. **Analyses**: Specify exactly which analyses you will conduct to examine the main question/hypothesis.
6. **Outliers & Exclusions**: Describe exactly how outliers will be defined and handled, and your precise rule(s) for excluding observations.
7. **Predicted Plot**: Please submit a predicted plot for your study based on what you expect the pattern to look like for your main hypothesis.
8. **Sample Size**: How many observations will be collected or what will determine sample size? No need to justify the decision, but be precise about exactly how the number will be determined.
9. **Exploratory details**: Anything else you would like to pre-register? (e.g., secondary analyses, variables collected for exploratory purposes, unusual analyses planned?)

recap: Oct 24, 2023

- what we covered:
  - tidyverse verbs
- your to-do's were:
  - *complete*: data cleaning + readying experiment for piloting
  - *prep*: complete the Tidy your Data primer
  - *prep*: start formative assignment #2

today's agenda

- reviewing tidyverse verbs through your data
- learning a few more verbs/functions

open your RStudio project

- open the project and your .Rmd file
- run all chunks
- create new heading # load revised class data
- download & import revised data
- change data types for a few columns

```
# load revised class data

```{r}
savic = read_csv("final_class_data.csv") %>%
  mutate(rt = as.numeric(rt),
         relatedness = as.factor(relatedness),
         type = as.factor(type))
```
```

basic descriptives

- we first want to understand some basic information about this dataset
- how many total trials are in your dataset?
- how many levels does the relatedness variable have?
- what if you wanted the count the number of total trials per participant?

```
# basic descriptives

```{r}

nrow(savic)


```
```

```
levels(savic$relatedness)
```

tidyverse: count()

- count() is another useful descriptive function like summarize() that tallies up counts of different things in your dataset while respecting groupings
- group by ID and count the trials
- how many target trials per ID?

```
# basic descriptives

```{r}

nrow(savic)

savic %>% group_by(ID) %>% count()

```
```

```
# A tibble: 47 × 2
# Groups:   ID [47]
          ID     n
       <dbl> <int>
 1   5418680   588
 2  46356924   588
 3  52271504   595
 4  59881077   604
 5 161705773   594
 6 223076836   592
 7 275998227   691
 8 276772242   591
 9 291529588   617
10 317312681   602
# … with 37 more row
# ℹ Use `print(n = .
```

```
savic %>% filter(typeoftrial == "target") %>%
  group_by(ID) %>% count()

```
```

```
> savic %>% filter(typeoftrial == "target") %>%
+   group_by(ID) %>% count()
# A tibble: 45 × 2
# Groups:   ID [45]
          ID     n
       <dbl> <int>
 1   5418680   104
 2  46356924   104
 3  52271504   104
 4  59881077   104
 5 161705773   104
 6 223076836   104
 7 275998227   104
```
```

tidyverse: pull()

- pull() is a convenient function that allows you to get the values inside one specific column as a vector
- extract the RTs from target trials

```r
savic %>%
  filter(typeoftrial == "target") %>%
  pull(rt)
```

```
> savic %>%
+   filter(typeoftrial == "target") %>%
+   pull(rt)
  [1]   NA   NA   75   NA   NA   NA   NA  214   NA   34
 [17]  242   NA  136   NA   NA  277   NA  294   NA   NA
 [33]  176  208  177   NA  254   NA   NA   NA  150   64
 [49]   NA   NA  227  223  302   NA  230   NA   88  253
 [65]   75  224   81   NA   98   NA  187   NA   NA   NA
 [81]   NA  133   NA   NA   NA   NA   NA   NA  246  214
```

tidyverse: unique()

- unique() lets you see how many unique values are inside a particular column or vector
- how many subjects did the experiment?
- the length of a vector can be obtained using length()

```
savic %>%
  pull(ID) %>% unique()
```

```
 [1]  823806428  275998227  399427091  617108779  438881597   59881077  685459176  789634588  896801386  636502299  500683496
[12] 9793241458 7008638007 7840701730 6636886676 1804967059 1017677598  958013040  223076836  633750828  713138680  560469737
[23]   52271504  983568245  510710057  343557222   46356924    5418680  463680341  276772242  823472278  291529588  969332346
[34]  964433595  648738364  653349550  161705773  858059169  317312681  492031667  768759264  607043191  794294242  393616702
[45]  988749039  366197048  662530864
```

```
savic %>%
  pull(ID) %>% unique() %>% length()
```

wrangling your data

phase	measure	type	exclusion criteria
attention	accuracy	descriptive	< 0.75
association	proportion of responses	descriptive	
priming	$RT_{related}$ vs. $RT_{unrelated}$ for direct and shared pairs	inferential (mixed effects model / ANOVA)	RT < 200 ms and RT > 1500 ms correct responses

attention

- create new heading # attention
- define a new dataframe attention_trials that only consists of the attention check rows
- which columns are most relevant?
- only keep the relevant columns
- view this data

```
# attention

```{r}

attention_trials = savic %>% filter(typeoftrial == "attention")
```

```
attention_trials = savic %>% filter(typeoftrial == "attention") %>%
  select(ID, revised_response, novel1, novel2, novel3, revised_correct)
```

ID	revised_response	novel1	novel2	novel3	revised_correct
823806428	Any	foobly	mipp	NOT_FOUND	0
275998227	apple	foobly	mipp	NOT_FOUND	0
399427091	apple	foobly	NOT_FOUND	NOT_FOUND	0
617108779	apple	foobly	NOT_FOUND	NOT_FOUND	0
438881597	apple	foobly	NOT_FOUND	NOT_FOUND	0
59881077	Apple	foobly	NOT_FOUND	NOT_FOUND	0
685459176	apple	foobly	Zimziland	NOT_FOUND	0
```

summarizing accuracy

- calculate the mean accuracy and standard deviation across all attention trials
- how do we do this for each participant?

```
## mean

attention_trials %>%
  summarize(mean_accuracy = mean(revised_correct),
            sd_accuracy = sd(revised_correct))
```

```
## summarize participant accuracy
subject_attention_accuracy = attention_trials %>%
  group_by(ID) %>%
  summarize(mean_accuracy = mean(revised_correct))
```

first-R-notebook.Rmd × | subject_attention_accuracy ×

1.0000000 | Filter

	ID	mean_accuracy
1	5418680	0.8888889
2	46356924	1.0000000
3	52271504	0.7777778
4	59881077	0.2222222
5	161705773	0.1111111
6	223076836	0.8888889
7	275998227	0.5555556
8	276772242	1.0000000
9	291529588	0.6666667
10	317312681	1.0000000
11	343557222	0.8888889
12	366197048	0.0000000

excluding participants

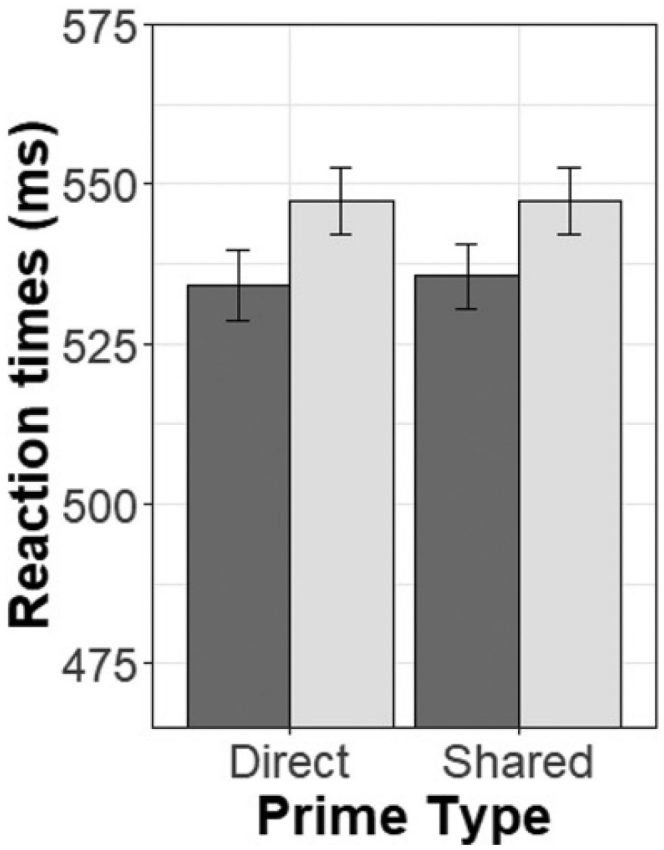
- what was our exclusion criteria for attention check?
- how do we find IDs that have accuracy less than 75%?
- storing these IDs in a vector will be useful later on

```
## find IDs that have less than 75% accuracy

low_acc_IDs = subject_attention_accuracy %>%
  filter(mean_accuracy < 0.75) %>%
  pull(ID)
```

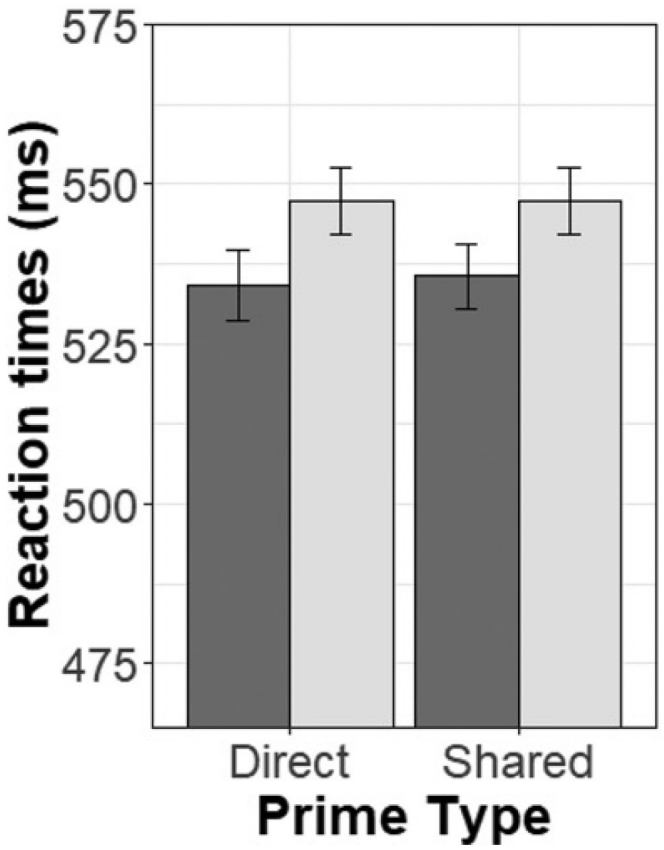
priming data plan

- list out all the steps we will need to take to get to our intended plot from the raw data
- also list the tidyverse functions you may need to use for each step



priming data plan

- filter for target trials
- select only relevant columns
- apply exclusions: filter, %in%
  - remove NA trials
  - RT > 200 and RT < 1500
  - correct responses
  - non-practice trials
  - relatedness is related/unrelated
  - type is direct/shared
  - remove low accuracy IDs
- compute means per condition
  - summarize



priming trials: filtering

- filter for target trials

```
priming_data = savic %>% filter(typeoftrial == "target")
```

priming trials: selecting

- filter for target trials
- select only relevant columns

```
priming_data = savic %>% filter(typeoftrial == "target") %>%
  select(ID, rt, relatedness, prime, response, type, correct, block_number, target, correct_key)
```

priming trials: filtering/exclusions

- filter for target trials
- select only relevant columns
- apply exclusions: filter, %in%
  - remove NA trials
  - RT > 200 and RT < 1500
  - correct responses
  - non-practice trials
  - relatedness is related/unrelated
  - type is direct/shared
  - remove low accuracy IDs

```
priming_data = savic %>% filter(typeoftrial == "target") %>%
  select(ID, rt, relatedness, prime, response, type, correct, block_number, target, correct_key)%>%
  filter(!is.na(rt))
```

```
  filter(!is.na(rt), rt > 200 , rt < 1500)
```

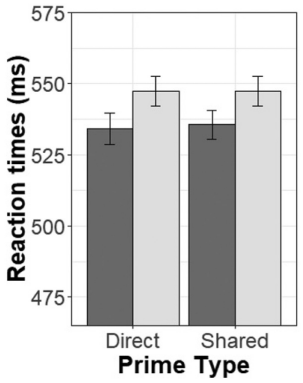
```
filter(!is.na(rt), rt > 200 , rt < 1500, correct == "TRUE")
```

```
filter(!is.na(rt), rt > 200 , rt < 1500, correct == "TRUE", block_number == 1)
```

```
filter(!is.na(rt), rt > 200 , rt < 1500, correct == "TRUE", block_number == 1) %>%
filter(relatedness %in% c("related", "unrelated")  & type %in% c("direct", "shared"))
```

```
priming_data = savic %>% filter(typeoftrial == "target") %>%
  select(ID, rt, relatedness, prime, response, type, correct,
         block_number, target, correct_key)%>%
  filter(!is.na(rt), rt > 200 , rt < 1500, correct == "TRUE", block_number == 1) %>%
  filter(relatedness %in% c("related", "unrelated")  & type %in% c("direct", "shared")) %>%
  filter(!ID %in% low_acc_IDs)
```

priming trials: compute means



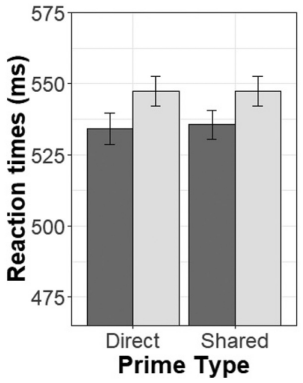
- use the priming data to compute means for each condition

```
## plot

```{r}
priming_data %>%
  group_by(type, relatedness) %>%
  summarise(mean_rt = mean(rt))
```
```

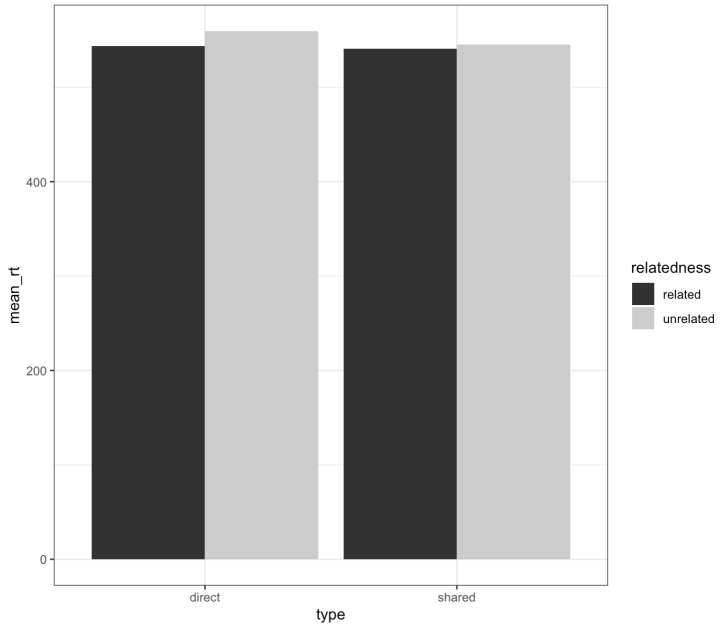
```
  type   relatedness mean_rt
  <fct>  <fct>         <dbl>
1 direct related        544.
2 direct unrelated      560.
3 shared related        541.
4 shared unrelated      545.
```

plot priming data



- directly pass the data from the means into ggplot()
- interpretation?

```
priming_data %>%
  group_by(type, relatedness) %>%
  summarise(mean_rt = mean(rt)) %>%
  ggplot() +
  geom_col(mapping = aes(x= type, y = mean_rt,
                         group = relatedness, fill = relatedness),
           position = "dodge")+
  theme_bw()+
  scale_fill_grey()
```



association data

- how do we evaluate whether participants responded with the correct triads?
- how do we evaluate what is a direct or indirect association?

In the free association task, participants were asked to respond to the prompt word with one of the training triad words. They responded as instructed on an average 96% of the free association trials presented at the end of training. In addition, they tended to respond with training words that had directly co-occurred with the prompt word. Whereas 81% of participants' responses were based on direct co-occurrence, only 2% were based on shared co-occurrence regularities.[3]

[3] Please note that here and in all subsequent experiments the proportion of responses congruent with direct and shared co-occurrence regularities was corrected for guessing. This was needed to more accurately reflect true learning and differentiate it from high proportions of congruent responses that could spuriously result from simple guessing given that the number of possible responses was restricted to six words. Complete data, coding schema and steps in analyses of attention check questions and free association data are available at https://osf.io/dt84u/?view_only=84eda92478e34cda98fe4adcf2417339.

creating a scoring sheet

- four possible cues were presented
- each cue has six possible valid responses
- each response can be congruent / incongruent for a given cue
- the type of association can be direct / shared / random for a given cue-response

dodish | foobly | geck | mipp

apple | dodish | foobly

geck | horse | mipp

read in scoring sheet

- new heading # association
- read in the scoring sheet and view the dataframe
- what are congruent responses?
- what is a direct association?
- what is a random association?

```
# association

```{r}

scoring = read_csv("association_scoring.csv")%>%
  arrange(cue, response)
```

cue	response	congruence	type_of_association	cue_type
dodish	apple	incongruent	direct	adjective
dodish	dodish	repeat	random	adjective
dodish	foobly	incongruent	random	adjective
dodish	geck	congruent	direct	adjective
dodish	horse	congruent	direct	adjective
dodish	mipp	incongruent	direct	adjective
foobly	apple	congruent	direct	adjective
foobly	dodish	incongruent	random	adjective
foobly	foobly	repeat	random	adjective
foobly	geck	incongruent	direct	adjective
```

merging two dataframes

- we want to merge our association data with this scoring sheet
- first, filter for association trials
- select relevant columns
- compare association trials to scoring data
- to merge, we need at least one shared column between two dataframes
- potential problems?

```
association_trials = savic %>%
  filter(typeoftrial == "association")
```

```
association_trials = savic %>%
  filter(typeoftrial == "association") %>%
  select(ID, revised_response, cue)
```

first-R-notebook.Rmd* | association_trials

Filter

	ID	revised_response	cue
1	5418680	Apple	foobly
2	5418680	Apple	foobly
3	5418680	Dodish	geck
4	5418680	Dodish	geck
5	5418680	Dodish	geck

cue	response	congruence	type_of_association	cue_type
dodish	apple	incongruent	direct	adjective
dodish	dodish	repeat	random	adjective
dodish	foobly	incongruent	random	adjective
dodish	geck	congruent	direct	adjective
dodish	horse	congruent	direct	adjective
dodish	mipp	incongruent	direct	adjective
foobly	apple	congruent	direct	adjective
foobly	dodish	incongruent	random	adjective
foobly	foobly	repeat	random	adjective
foobly	geck	incongruent	direct	adjective

setting up for merging

- rename() the response column
- convert to lowercase

```
association_trials = savic %>%
  filter(typeoftrial == "association") %>%
  select(ID, revised_response, cue) %>%
  rename(response = "revised_response")%>%
  mutate(response = tolower(response))
```

first-R-notebook.Rmd* | association_trials

Filter

	ID	revised_response	cue
1	5418680	Apple	foobly
2	5418680	Apple	foobly
3	5418680	Dodish	geck
4	5418680	Dodish	geck
5	5418680	Dodish	geck

first-R-notebook.Rmd | association_trials

Filter

	ID	response	cue
1	5418680	apple	foobly
2	5418680	apple	foobly
3	5418680	dodish	geck
4	5418680	dodish	geck
5	5418680	dodish	geck
6	5418680	dodish	geck
7	5418680	dodish	geck

tidyverse: left_join()

- left_join() allows you to merge additional columns from a different dataframe to your dataframe, by matching on common column names and values

```
association_trials = savic %>%
  filter(typeoftrial == "association") %>%
  select(ID, revised_response, cue) %>%
  rename(response = "revised_response")%>%
  mutate(response = tolower(response)) %>%
  left_join(scoring)
```

ID	response	cue	congruence	type_of_association	cue_type
5418680	apple	foobly	congruent	direct	adjective
5418680	apple	foobly	congruent	direct	adjective
5418680	dodish	geck	congruent	direct	noun
5418680	dodish	geck	congruent	direct	noun
5418680	dodish	geck	congruent	direct	noun
5418680	dodish	geck	congruent	direct	noun
5418680	dodish	geck	congruent	direct	noun

computing congruence

- first, we remove NA trials
- keep only congruent/incongruent trials
- keep only direct/shared associations

```
congruence_trials = association_trials %>%
  filter(!is.na(congruence))%>%
  filter(congruence %in% c("congruent", "incongruent")) %>%
  filter(type_of_association %in% c("direct", "shared"))
```

congruence counts

- create new dataframe called congruence_counts
- group by ID, congruent, association type, and cue type and compute a count

```
congruence_counts = congruence_trials %>%
  group_by(ID, cue_type,  congruence,type_of_association) %>%
  count()
```

ID	cue_type	congruence	type_of_association	n
5418680	adjective	congruent	direct	18
5418680	noun	congruent	direct	18
46356924	adjective	congruent	direct	15
46356924	adjective	incongruent	direct	2
46356924	noun	congruent	direct	5
46356924	noun	incongruent	direct	12
46356924	noun	incongruent	shared	1

congruence proportions

ID	cue_type	congruence	type_of_association	n
5418680	adjective	congruent	direct	18
5418680	noun	congruent	direct	18
46356924	adjective	congruent	direct	15
46356924	adjective	incongruent	direct	2
46356924	noun	congruent	direct	5
46356924	noun	incongruent	direct	12
46356924	noun	incongruent	shared	1

- next, group by ID and cue type and compute a proportion

```
congruence_counts = congruence_trials %>%
  group_by(ID, cue_type,  congruence,type_of_association) %>%
  count() %>%
  group_by(ID, cue_type) %>%
  mutate(proportion = n / sum(n))
```

ID	cue_type	congruence	type_of_association	n	proportion
5418680	adjective	congruent	direct	18	1.00000000
5418680	noun	congruent	direct	18	1.00000000
46356924	adjective	congruent	direct	15	0.88235294
46356924	adjective	incongruent	direct	2	0.11764706
46356924	noun	congruent	direct	5	0.27777778
46356924	noun	incongruent	direct	12	0.66666667
46356924	noun	incongruent	shared	1	0.05555556

correcting for guessing

- we could just look at the proportion of trials that were congruent
- but this doesn't account for incongruent trials (or guessing)
- we want to subtract the proportion of incongruent trials from congruent trials

```
congruence_counts %>%
  filter(congruence == "congruent") %>%
  ungroup()%>%
  summarise(mean_prop = mean(proportion))
```

```
A tibble: 1 × 1
  mean_prop
      <dbl>
      0.860
```

long vs. wide data

- data is often in 2 main formats:
  - long
  - wide
- long data has multiple rows indicating each observation
- wide data has multiple columns indicating each observation

long

ID	cue_type	congruence	type_of_association	n	proportion
5418680	adjective	congruent	direct	18	1.00000000
5418680	noun	congruent	direct	18	1.00000000
46356924	adjective	congruent	direct	15	0.88235294
46356924	adjective	incongruent	direct	2	0.11764706
46356924	noun	congruent	direct	5	0.27777778
46356924	noun	incongruent	direct	12	0.66666667
46356924	noun	incongruent	shared	1	0.05555556

ID	cue_type	type_of_association	congruent	incongruent
5418680	adjective	direct	1.00000000	NA
5418680	noun	direct	1.00000000	NA
46356924	adjective	direct	0.88235294	0.11764706
46356924	noun	direct	0.27777778	0.66666667

wide

converting to wide format

ID	cue_type	congruence	type_of_association	n	proportion
5418680	adjective	congruent	direct	18	1.00000000
5418680	noun	congruent	direct	18	1.00000000
46356924	adjective	congruent	direct	15	0.88235294
46356924	adjective	incongruent	direct	2	0.11764706
46356924	noun	congruent	direct	5	0.27777778
46356924	noun	incongruent	direct	12	0.66666667
46356924	noun	incongruent	shared	1	0.05555556

- select relevant columns
- pivot_wider()
- specifies which columns to make wide and where to get the values from

```
wide_counts = congruence_counts %>%
  select(ID, cue_type,congruence, type_of_association, proportion) %>%
  pivot_wider(names_from = congruence, values_from = proportion)
```

ID	cue_type	type_of_association	congruent	incongruent
5418680	adjective	direct	1.00000000	NA
5418680	noun	direct	1.00000000	NA
46356924	adjective	direct	0.88235294	0.11764706
46356924	noun	direct	0.27777778	0.66666667

filling empty columns

- use mutate() to fill up NA values with 0s
- create new proportion column that computes difference between congruent and incongruent proportions
- mean of prop column?

ID	cue_type	type_of_association	congruent	incongruent
5418680	adjective	direct	1.00000000	NA
5418680	noun	direct	1.00000000	NA
46356924	adjective	direct	0.88235294	0.11764706
46356924	noun	direct	0.27777778	0.66666667

```
wide_counts = congruence_counts %>%
  select(ID, cue_type,congruence, type_of_association, proportion) %>%
  pivot_wider(names_from = congruence, values_from = proportion)%>%
  mutate(incongruent = ifelse(is.na(incongruent), 0, incongruent),
         congruent = ifelse(is.na(congruent), 0, congruent))
```

	ID	cue_type	type_of_association	congruent	incongruent
1	5418680	adjective	direct	1.00000000	0.00000000
2	5418680	noun	direct	1.00000000	0.00000000
3	46356924	adjective	direct	0.88235294	0.11764706
4	46356924	noun	direct	0.27777778	0.66666667

```
wide_counts = congruence_counts %>%
  select(ID, cue_type,congruence, type_of_association, proportion) %>%
  pivot_wider(names_from = congruence, values_from = proportion)%>%
  mutate(incongruent = ifelse(is.na(incongruent), 0, incongruent),
         congruent = ifelse(is.na(congruent), 0, congruent))%>%
  mutate(prop = congruent - incongruent)
```

ID	cue_type	type_of_association	congruent	incongruent	prop
5418680	adjective	direct	1.00000000	0.00000000	1.00000000
5418680	noun	direct	1.00000000	0.00000000	1.00000000
46356924	adjective	direct	0.88235294	0.11764706	0.76470588

```
mean(wide_counts$prop)
```

going back to the analysis description

- what proportion of trials are congruent?

In the free association task, participants were asked to respond to the prompt word with one of the training triad words. They responded as instructed on an average 96% of the free association trials presented at the end of training. In addition, they tended to respond with training words that had directly co-occurred with the prompt word. Whereas 81% of participants' responses were based on direct co-occurrence, only 2% were based on shared co-occurrence regularities.[3]

```
> mean(wide_counts$prop)
[1] 0.7194747
```

HW: computing association proportions

- write, run, and interpret the code

first-R-notebook.Rmd × association_type_occurrence ×

Filter

	ID	cue_type	direct	shared
1	5418680	adjective	1.0000000	NA
2	5418680	noun	1.0000000	NA
3	46356924	adjective	0.7647059	NA
4	46356924	noun	-0.3888889	-0.05555556
5	52271504	adjective	1.0000000	NA
6	52271504	noun	1.0000000	NA
7	59881077	adjective	0.6470588	NA
8	59881077	noun	1.0000000	NA
9	161705773	adjective	1.0000000	NA
10	161705773	noun	1.0000000	NA

```r
## counts by type of association

association_type_occurrence = wide_counts %>%
  select(ID, cue_type, type_of_association, prop) %>%
  pivot_wider(names_from = type_of_association, values_from = prop) %>%
  mutate(shared = ifelse(is.na(shared), 0, shared),
         direct = ifelse(is.na(direct), 0, direct))

mean(association_type_occurrence$direct)
mean(association_type_occurrence$shared)
```

```
> mean(association_type_occurrence$direct)
[1] 0.8387088
> mean(association_type_occurrence$shared)
[1] -0.009946785
```

next class

- **before** class
  - *prep:* interpret/understand association code (HW)
  - *prep:* Hypothesis Testing
  - *apply:* formative assignment #2 (R descriptive)
- **during** class
  - making statistical inferences from data!