Introduction to visualization using `ggplot()`

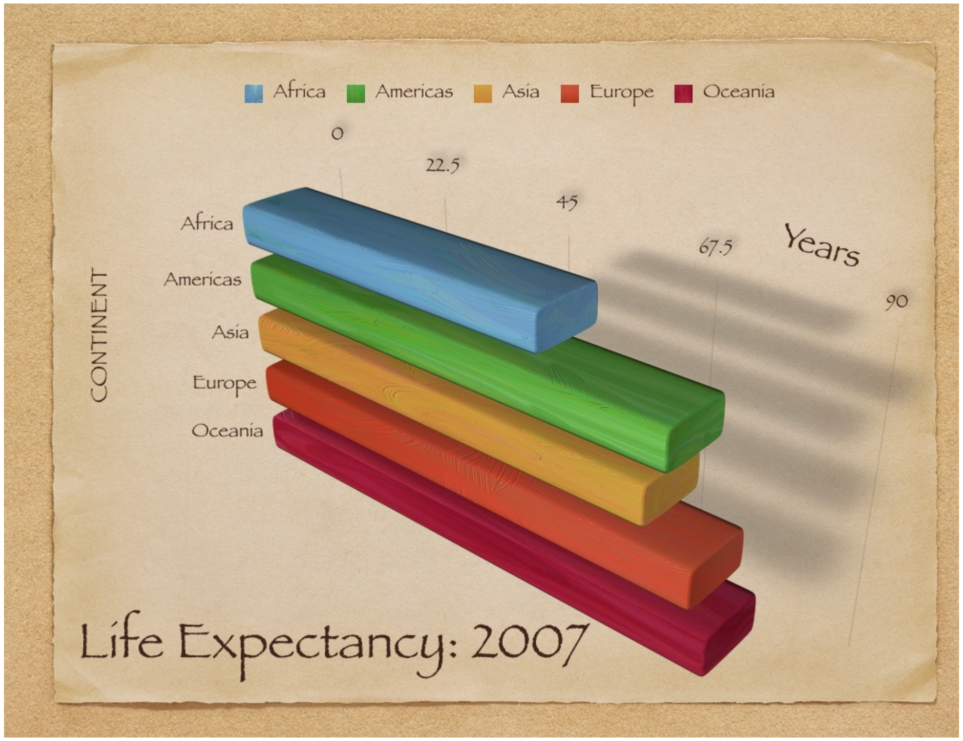
Yaqian Huang

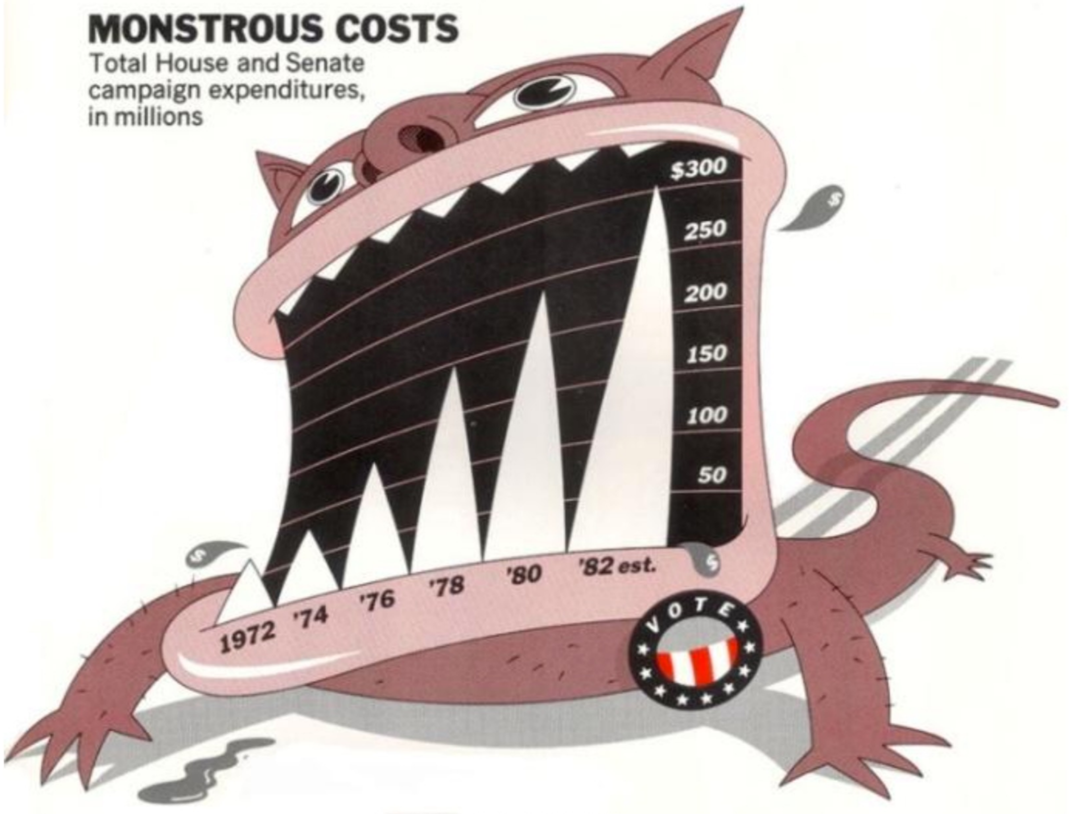
yaqian.huang@oeaw.ac.at

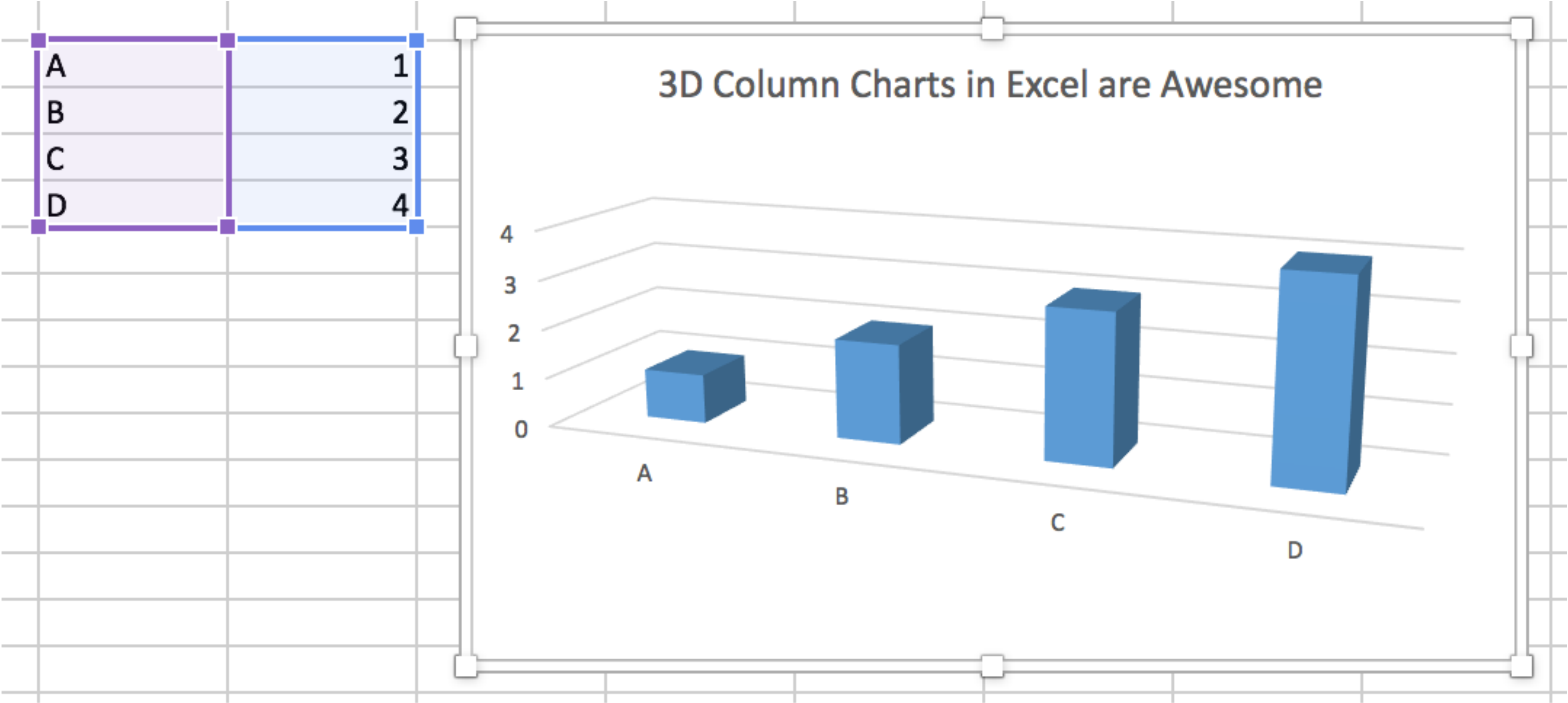
September 26, 2023

University of Canterbury | NZILBB

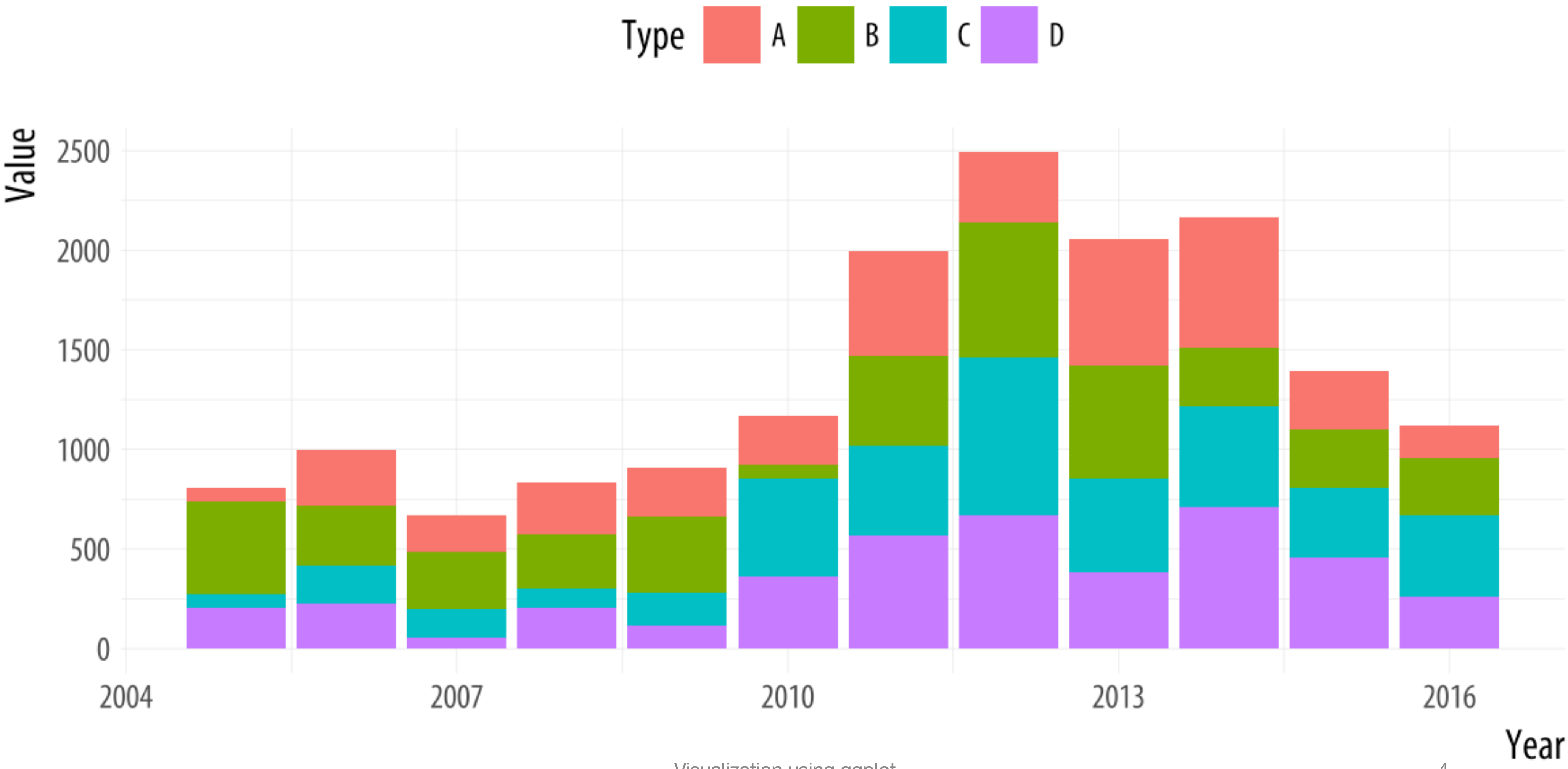
What’s wrong with these graphs?



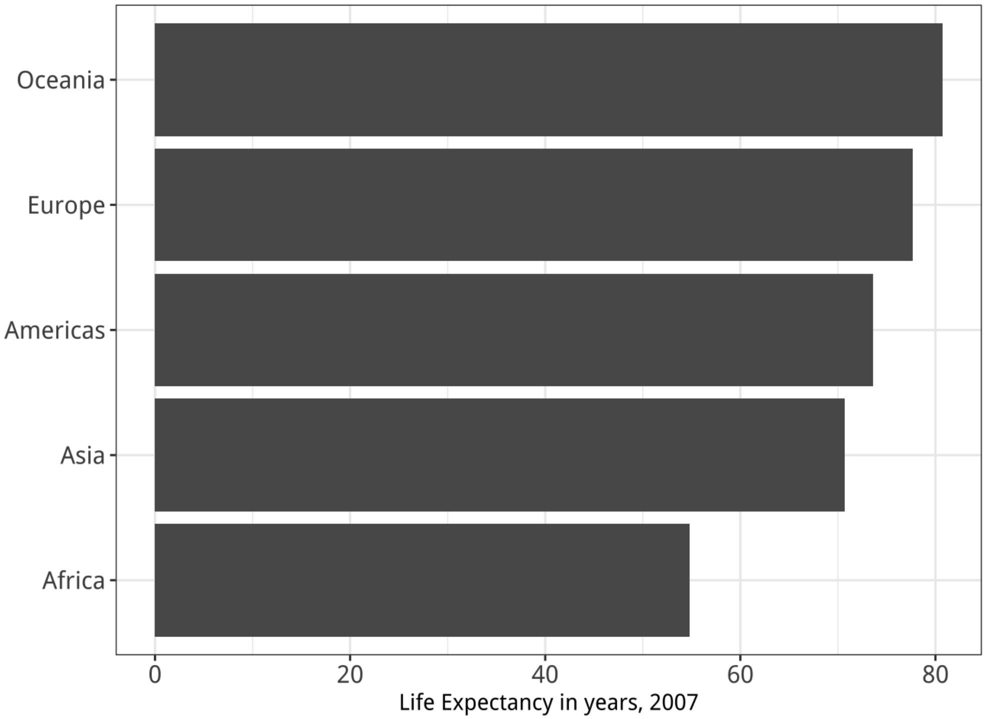


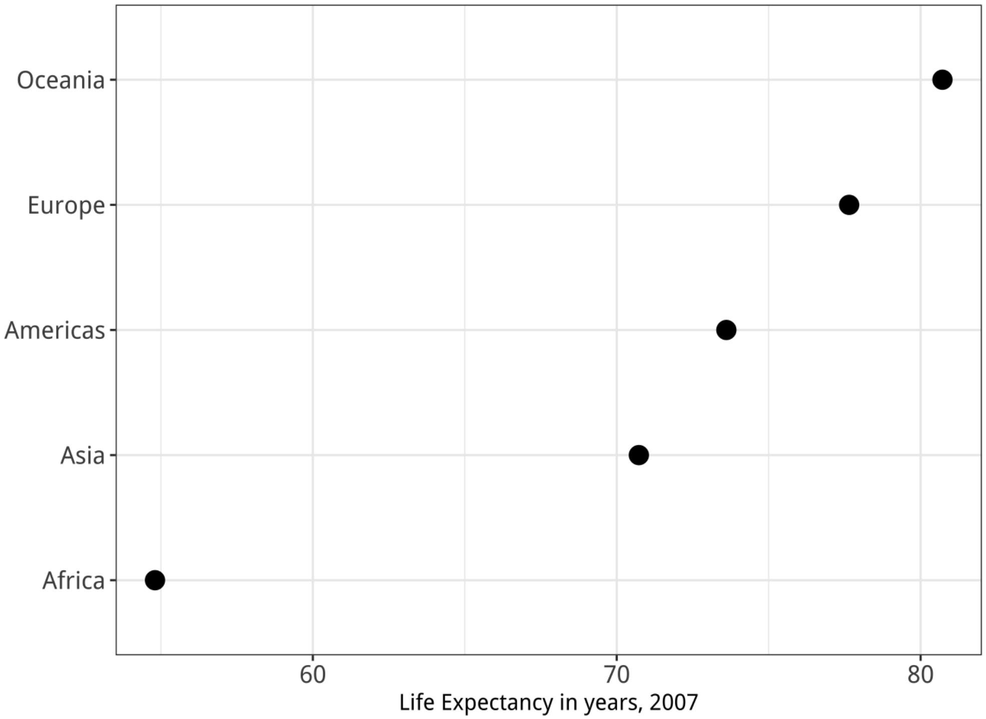


A	1
B	2
C	3
D	4


Type
A
B
C
D
Value
2500
2000
1500
1000
500
0
2004
2007
2010
2013
2016
Year

Which plot is better?





Class Objectives

1. Learn how to make good graphs

2. Learn how to achieve objective 1 with `ggplot()`

Graphs should …

- have labels and be interpretable without consulting a figure caption or having to solve a puzzle
- facilitate relevant quantitative interpretation and comparisons
- represent variability and uncertainty to permit inferential statistics by eye
- follow conventions for the kind of information/data being presented
- be visually accurate and consistent
- not waste ink and should otherwise look pretty

Remove
to improve
(the **data-ink** ratio)

`ggplot()`

- ‘grammar of graphics’ to organize and make sense of different elements
  - kind of plot, scales, title, labels, legends, colors, shapes, …

- breaks up the task of making a graph into a series of distinct tasks by adding them as layers

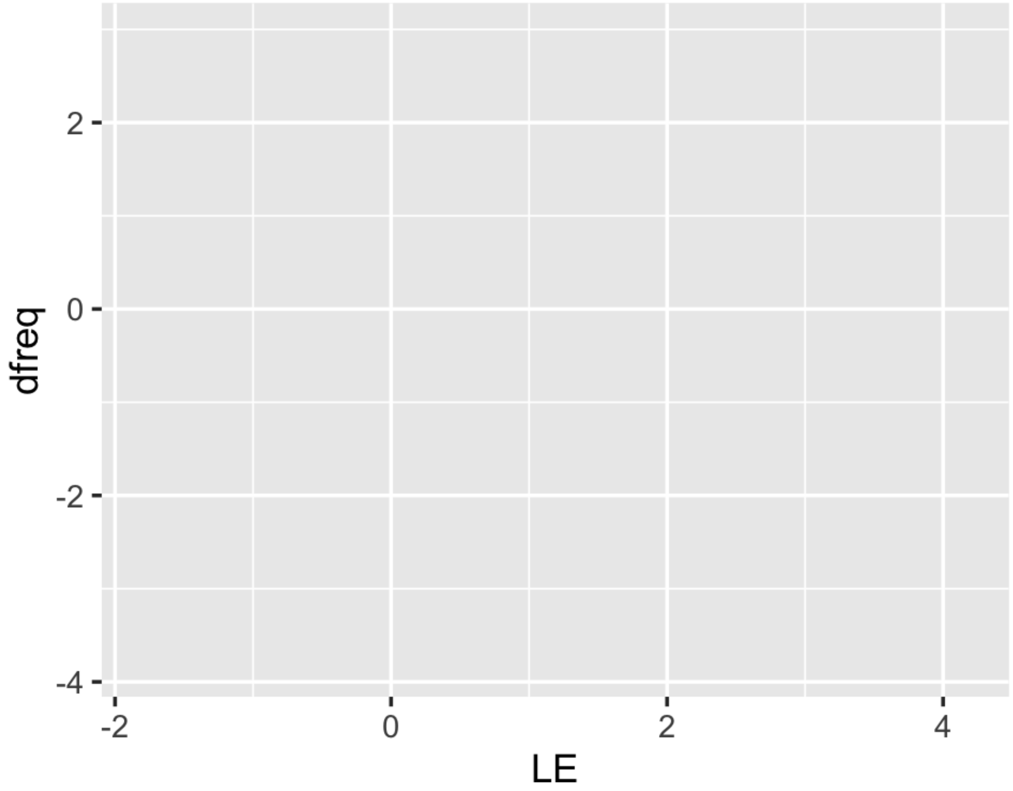
How to **`ggplot()`**

1. Put a tidy data frame in `ggplot()`
2. Map variables onto different `aes`(thetic variables) (e.g., x, y, color, shape, alpha, etc.).
3. Draw some `geom`(etric entity) according to that mapping (e.g., point, line, smooth, etc.)

```
> library(ggplot2)

> fig <- ggplot(data=..., mapping = aes(x=…, y=…, color=…)) + geom_*() +

 # the lines above form the basis of the plot,

 # the lines below make it look nicer

      facet_*() +

      scale_*() +

      theme*()
```

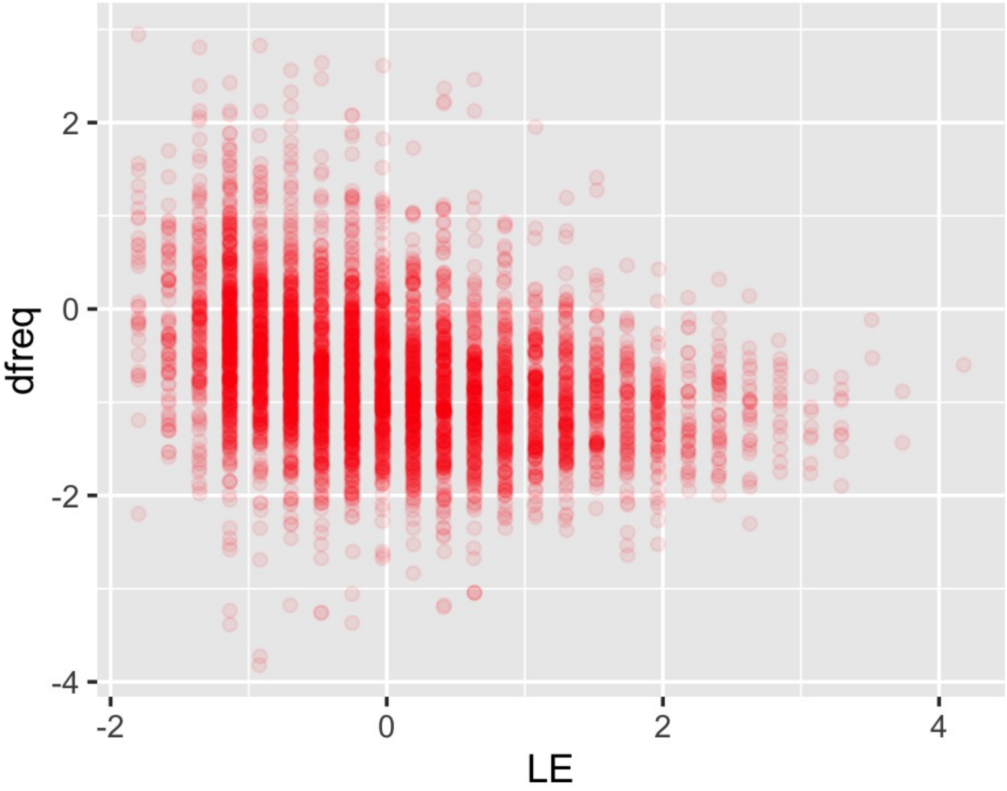
How to **ggplot()**

```
> ggplot()
```



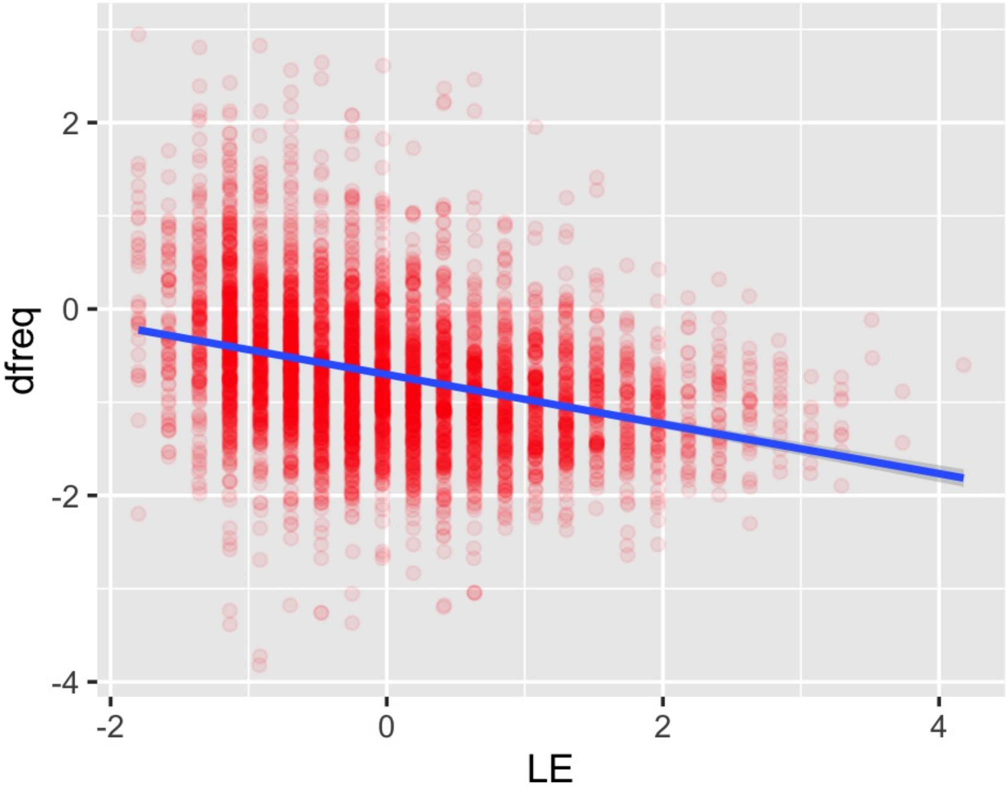
How to **ggplot()**

```
> ggplot()+
geom_point()
```



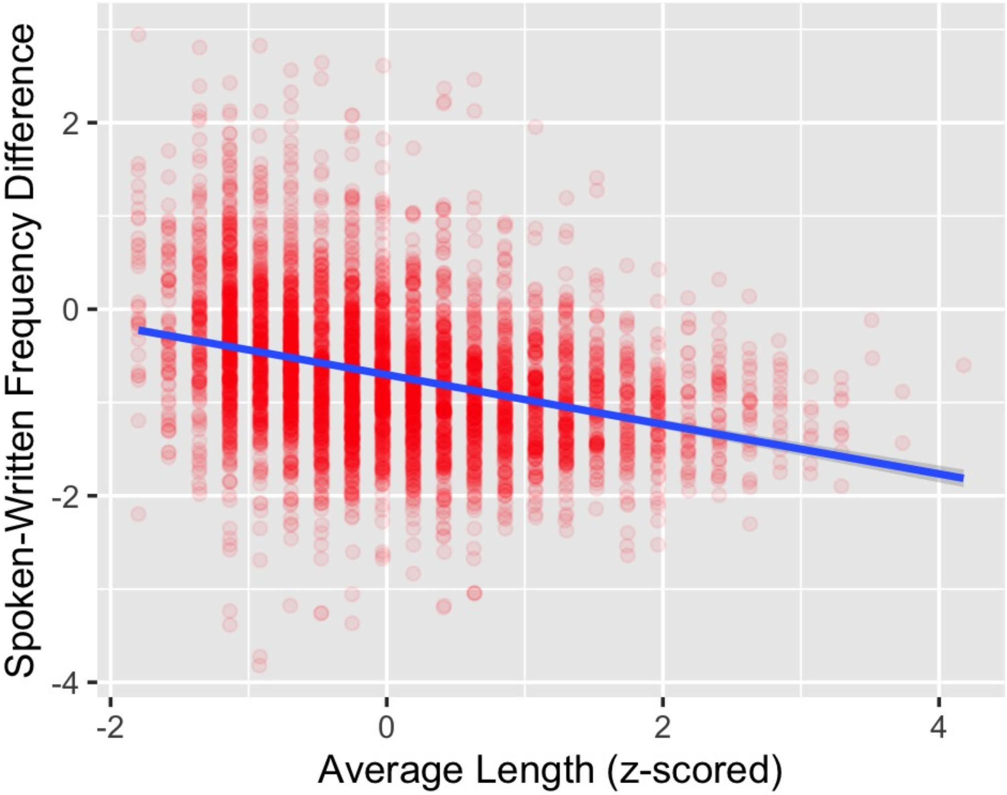
How to **ggplot()**

```
> ggplot()+
geom_point()+
geom_smooth()
```



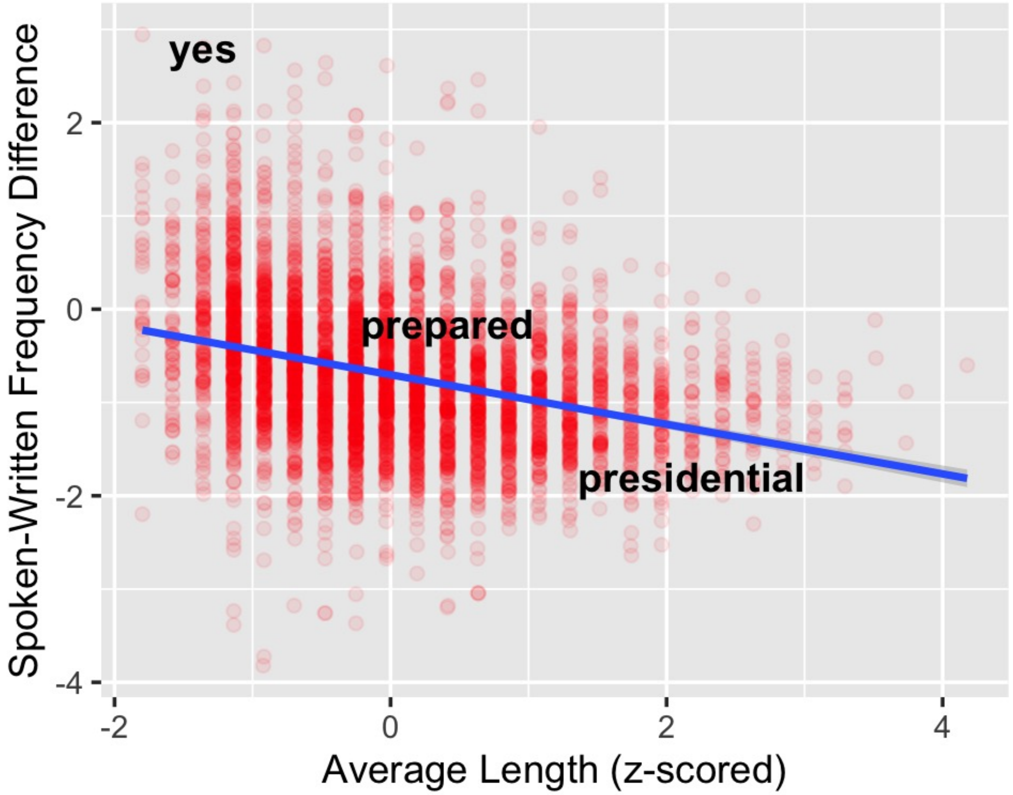
How to **ggplot()**

```
> ggplot()+
geom_point()+
geom_smooth()+
xlab()+ylab()
```



How to `ggplot()`

```
> ggplot()+
geom_point()+
geom_smooth()+
xlab()+ylab()+
geom_text()x3
```



Lau & Huang et al. (2019)

How to `ggplot()`

Goal: show how response/dependent variable(s) change with explanatory/independent variable(s).

What kind of variables? Categorical? Numerical?

Think of it as an abstract formula, e.g.,:

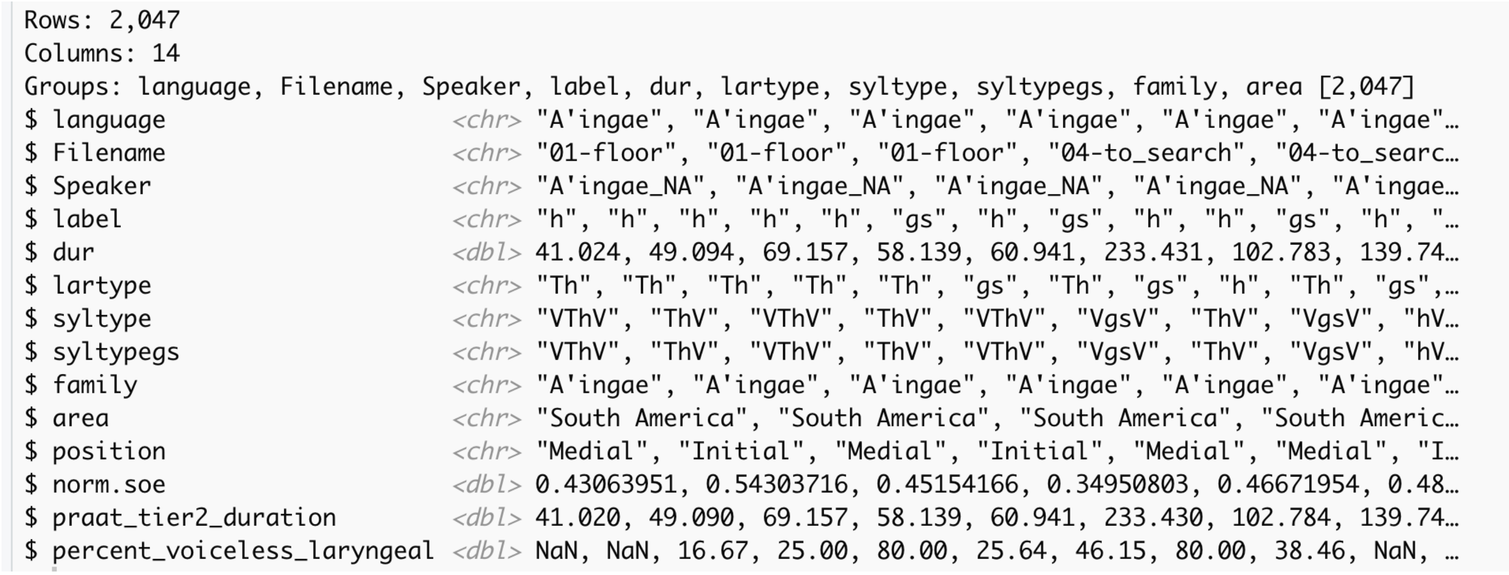
How does *VOT* vary across *gender*, *language*, and *speech rate*:

```
numerical ~ 2*categorical + numerical
```

Example: voicing in glottal sounds

Voicing intensity (SoE) and % voicing measures in glottal sounds ([ʔ, h, ɦ], creaky and breathy vowels) (Garellek et al., 2021)

```
> glimpse(df)
```

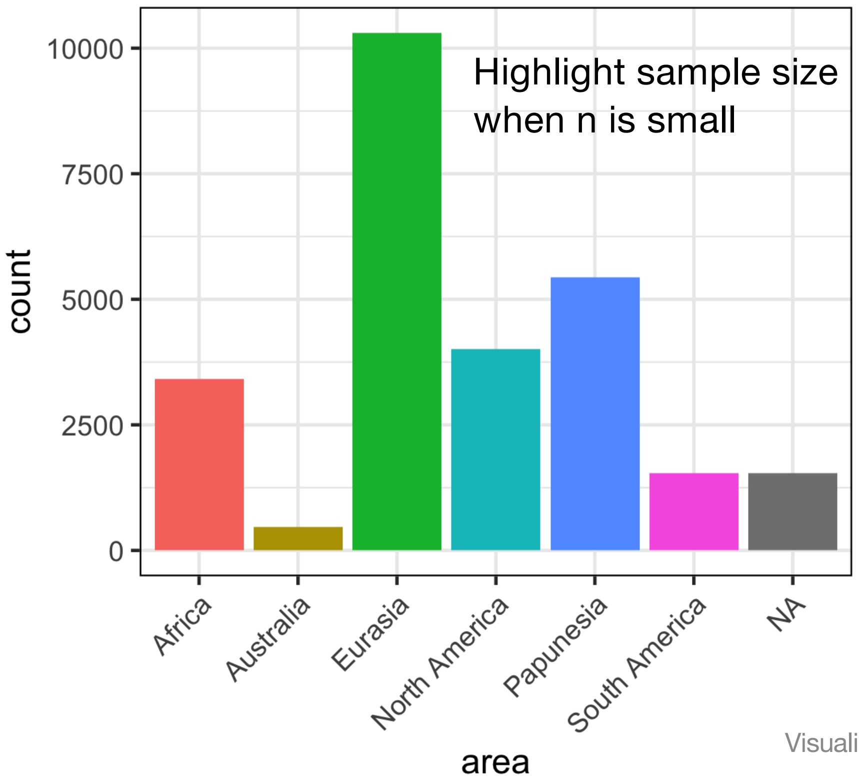


```
Rows: 2,047
Columns: 14
Groups: language, Filename, Speaker, label, dur, lartype, syltype, syltypegs, family, area [2,047]
$ language                  <chr> "A'ingae", "A'ingae", "A'ingae", "A'ingae", "A'ingae", "A'ingae"…
$ Filename                  <chr> "01-floor", "01-floor", "01-floor", "04-to_search", "04-to_searc…
$ Speaker                   <chr> "A'ingae_NA", "A'ingae_NA", "A'ingae_NA", "A'ingae_NA", "A'ingae…
$ label                     <chr> "h", "h", "h", "h", "h", "gs", "h", "gs", "h", "h", "gs", "h", "…
$ dur                       <dbl> 41.024, 49.094, 69.157, 58.139, 60.941, 233.431, 102.783, 139.74…
$ lartype                   <chr> "Th", "Th", "Th", "Th", "Th", "gs", "Th", "gs", "h", "Th", "gs",…
$ syltype                   <chr> "VThV", "ThV", "VThV", "ThV", "VThV", "VgsV", "ThV", "VgsV", "hV…
$ syltypegs                 <chr> "VThV", "ThV", "VThV", "ThV", "VThV", "VgsV", "ThV", "VgsV", "hV…
$ family                    <chr> "A'ingae", "A'ingae", "A'ingae", "A'ingae", "A'ingae", "A'ingae"…
$ area                      <chr> "South America", "South America", "South America", "South Americ…
$ position                  <chr> "Medial", "Initial", "Medial", "Initial", "Medial", "Medial", "I…
$ norm.soe                  <dbl> 0.43063951, 0.54303716, 0.45154166, 0.34950803, 0.46671954, 0.48…
$ praat_tier2_duration      <dbl> 41.020, 49.090, 69.157, 58.139, 60.941, 233.430, 102.784, 139.74…
$ percent_voiceless_laryngeal <dbl> NaN, NaN, 16.67, 25.00, 80.00, 25.64, 46.15, 80.00, 38.46, NaN, …
```

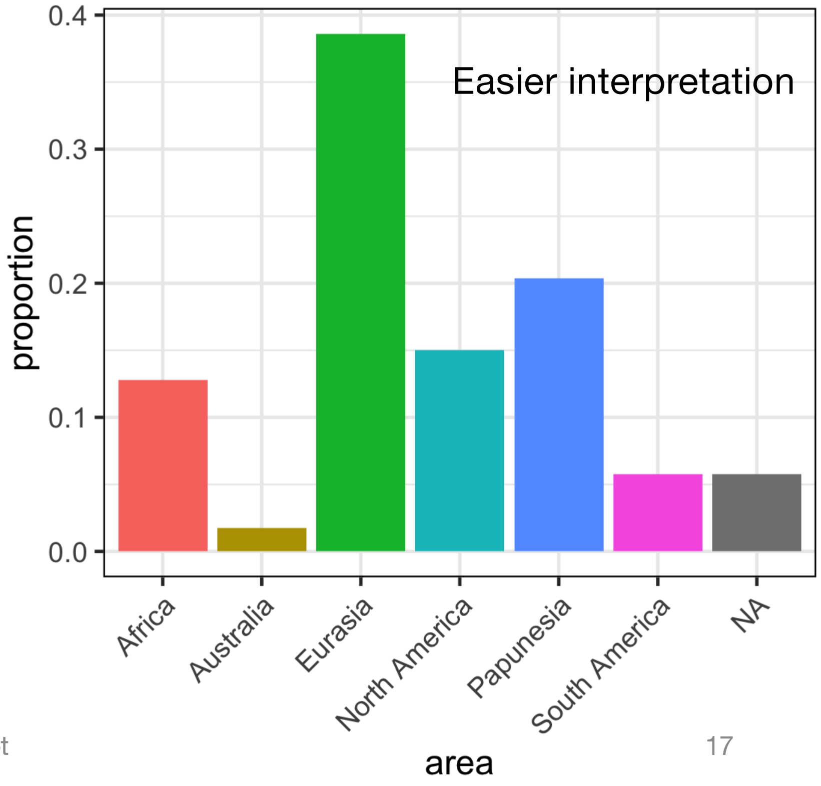
Categorical ~ 0

```
> ggplot(df, aes(x=area,
fill=area)) + geom_bar()
```

Code can be found in RMD



Highlight sample size when n is small



Easier interpretation

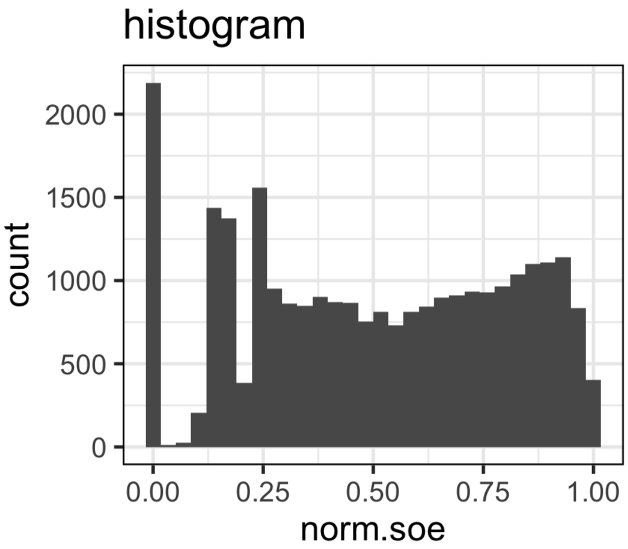
Numerical ~ 0

```
> ggplot(df,
aes(x=norm.soe))+
```

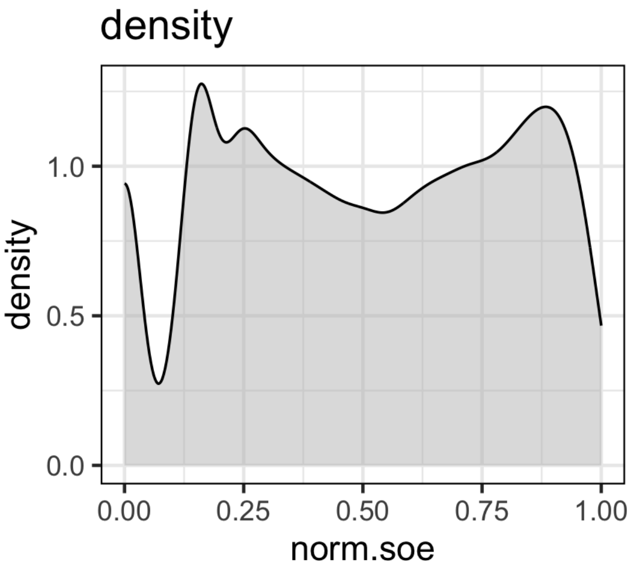
`geom_histogram()`

`OR`

`geom_density(fill=``'gray', alpha=.5)`



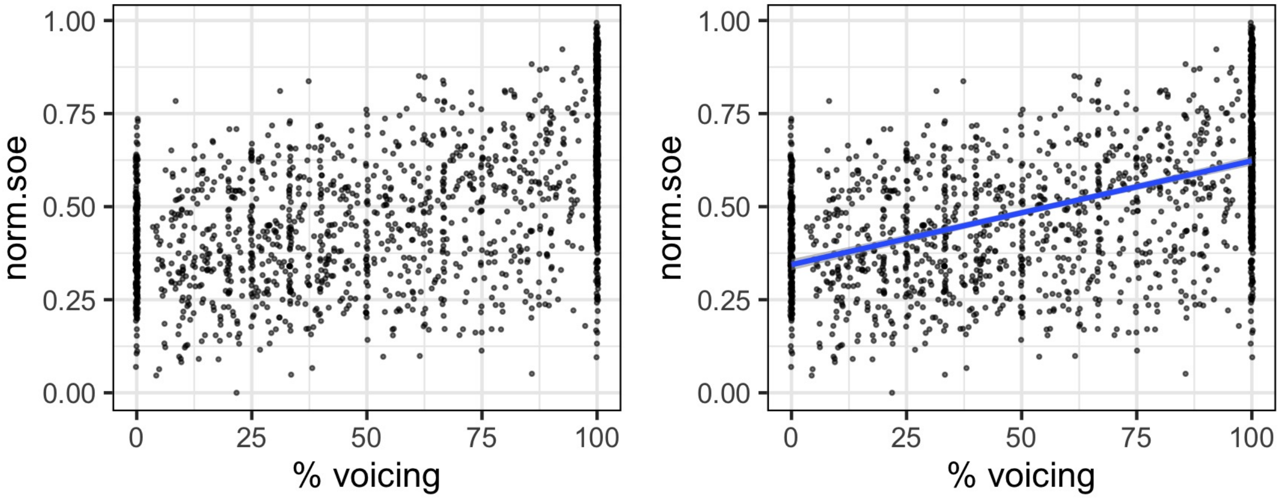
+ Portrays noisiness
- Impression sensitive to bins



- Obscures noisiness
+ not too sensitive to reasonable kernel width

Numerical ~ Numerical

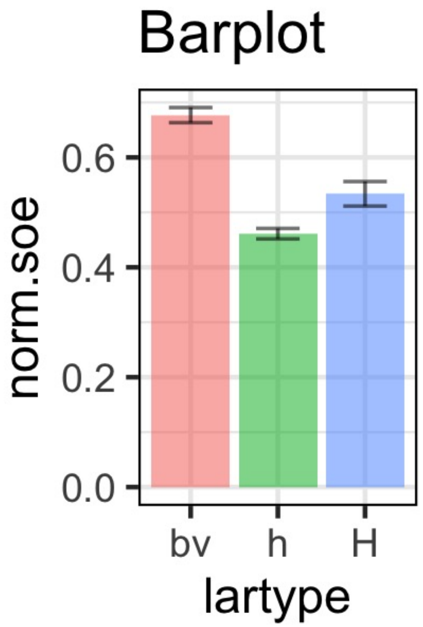
```
> ggplot(df.means,
aes(x=percent_voici
ng, y=norm.soe)) +
geom_point()

+
geom_smooth(method
= 'lm',na.rm = T)
```

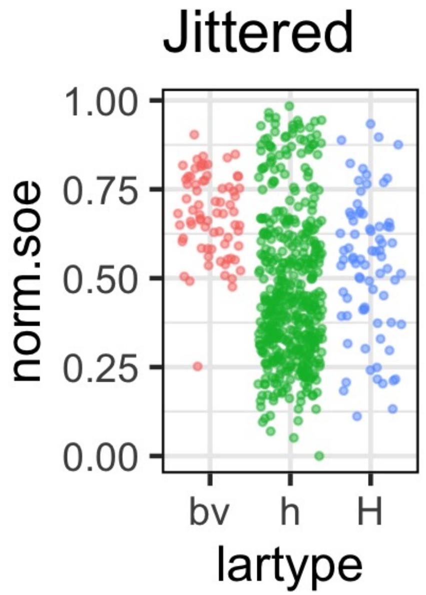


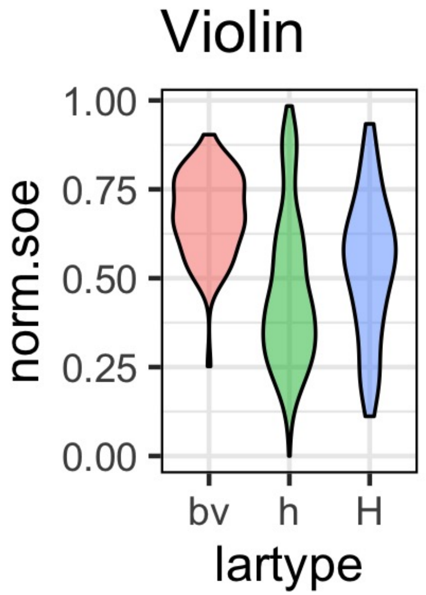
Show data & fitted linear model

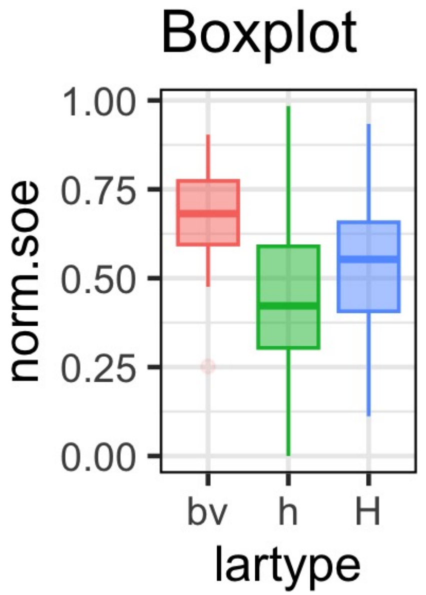
Numerical ~ Categorical

```
> ggplot(df,aes(x=lartype, color=lartype, fill=lartype, y=norm.soe))+
  stat_summary(fun.y = mean, geom="bar") OR geom_jitter() OR geom_violin()
OR geom_boxplot()
```







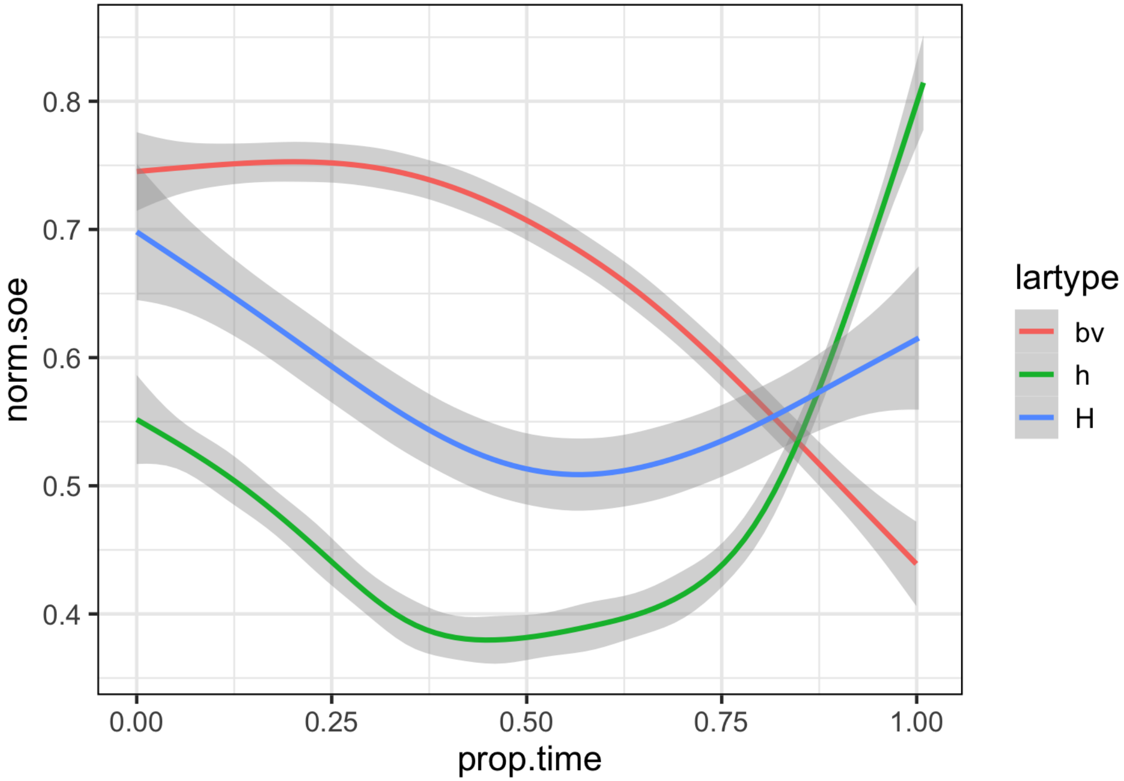


Notes on Numerical ~ Categorical

- Always put error bars on bar plots (std. error or CI are fine)
- Look at raw data (e.g., strip plots) before going to more compressed plots
- By removing the solid bar from a bar plot, we can add a good visualization of data distribution. This is better.

Numerical ~ Numerical + Categorical

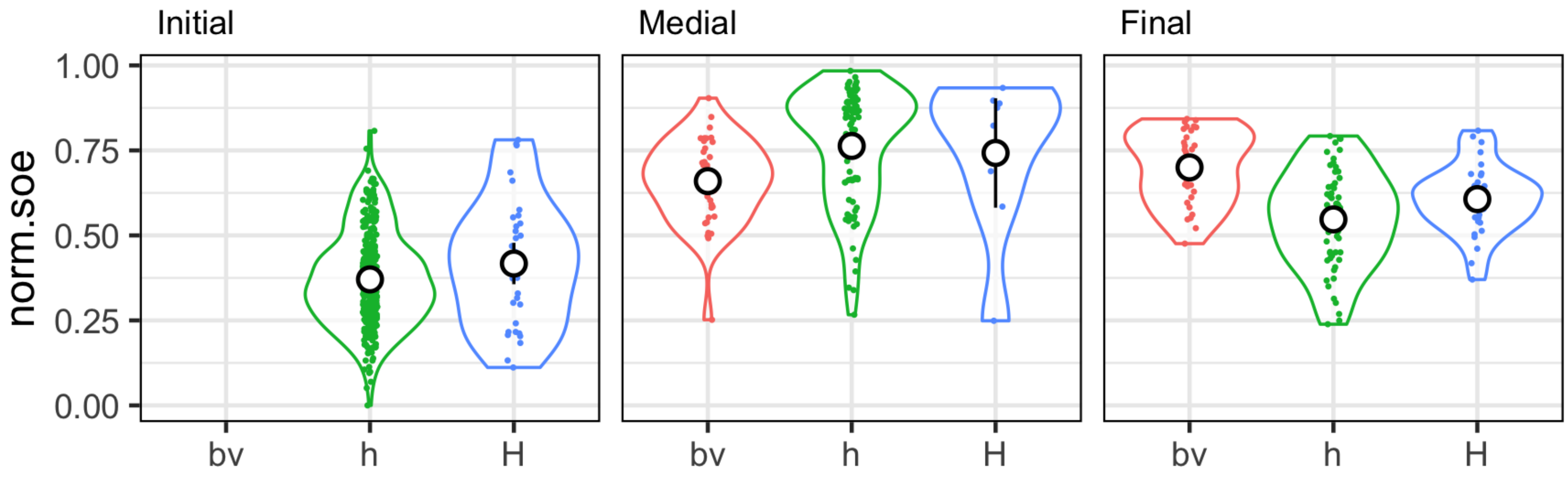
```
> ggplot(df,
aes(x=prop.time,
y=norm.soe,
color = lartype))+
geom_smooth()
```



Numerical ~ Categorical + Categorical

```
> ggplot(df,aes(x=lartype,y=norm.soe,color=lartype))+
      geom_violin()+ geom_jitter()+
      stat_summary(fun.data = mean_ci,  geom="pointrange")+
      facet_wrap(~position)
```

- Violin
- Mean+error bar
- Jitter



Wrap-up: what we’ve learned

- Visualization principles: dos and don’ts
- Implementation using `ggplot()`
- How to choose a plot type

Wrap-up: what to learn next

- More complicated graphs; fine-tuning parameters
- Reproducible figure schemes: theme list, color schemes
- Tailor to different purposes: presentation, papers, posters
- Interface with other useful tools, e.g., `ShinyR`

References

- Lau, S. H., Huang, Y., Ferreira, V. S., & Vul, E. (2019). Perceptual features predict word frequency asymmetry across modalities. *Attention, Perception, & Psychophysics, 81*, 1076-1087.
- Garellek, M., Chai, Y., Huang, Y., & Van Doren, M. (2023). Voicing of glottal consonants and non-modal vowels. *Journal of the International Phonetic Association, 53*(2), 305-332.
- Healy, K. (2018). Data visualization: a practical introduction. *Princeton University Press*.
- Wickham, H. (2011). ggplot2. *Wiley interdisciplinary reviews: computational statistics, 3*(2), 180-185.
- *ggplot2* cheat sheet: https://rstudio.github.io/cheatsheets/html/data-visualization.html

Questions?

All the materials can be accessed:
https://github.com/yaqianhuang/Stats-workshops