

Towards Interactivity

HES 505 Fall 2023: Session 31

Matt Williamson

Objectives

3 Categories of data visualization



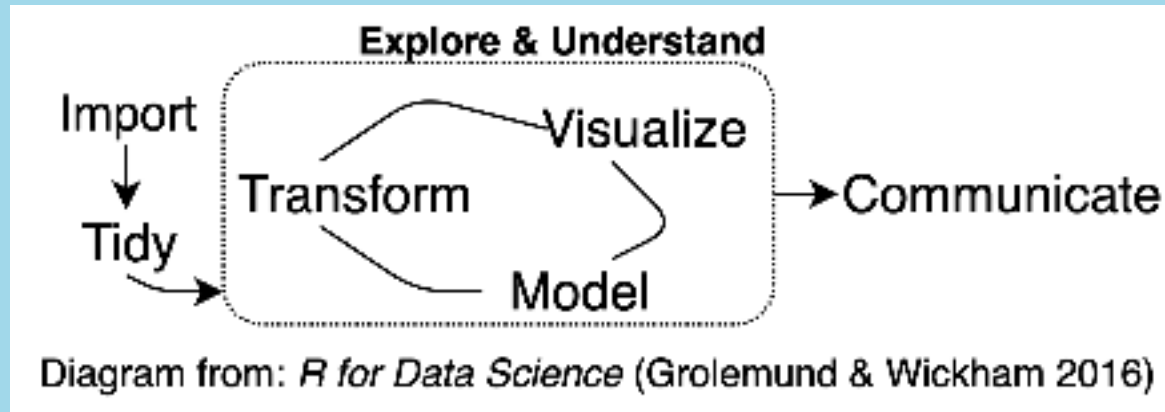
- Static
- Interactive
- Dynamic



dynamic

Why Move Beyond Static Maps?

Dealing with complex datasets



- Identifying structure that might otherwise be hidden
- Diagnosing models and interpreting results
- Aiding the sense-making process

Clarity in presentation

- Zooming allows the user to determine scale of presentation
- Hovering allows more information to be displayed 'on-demand'
- Subsetting facilitates ease of interpretation

Designing for the User

Who is your audience?

- Your advisor and colleagues?
- An external collaborator?
- The general public?
 - User archetypes

Iteration



From Usability.gov

- Feedback is critical
- Ideation: What *specifically* does the user need?
- Meaning: Are the data clearly defined and explained? Are the conclusions obvious?
- Function: Given the usecases, will the application (visualization) actually perform?

Building interactive visualizations in R

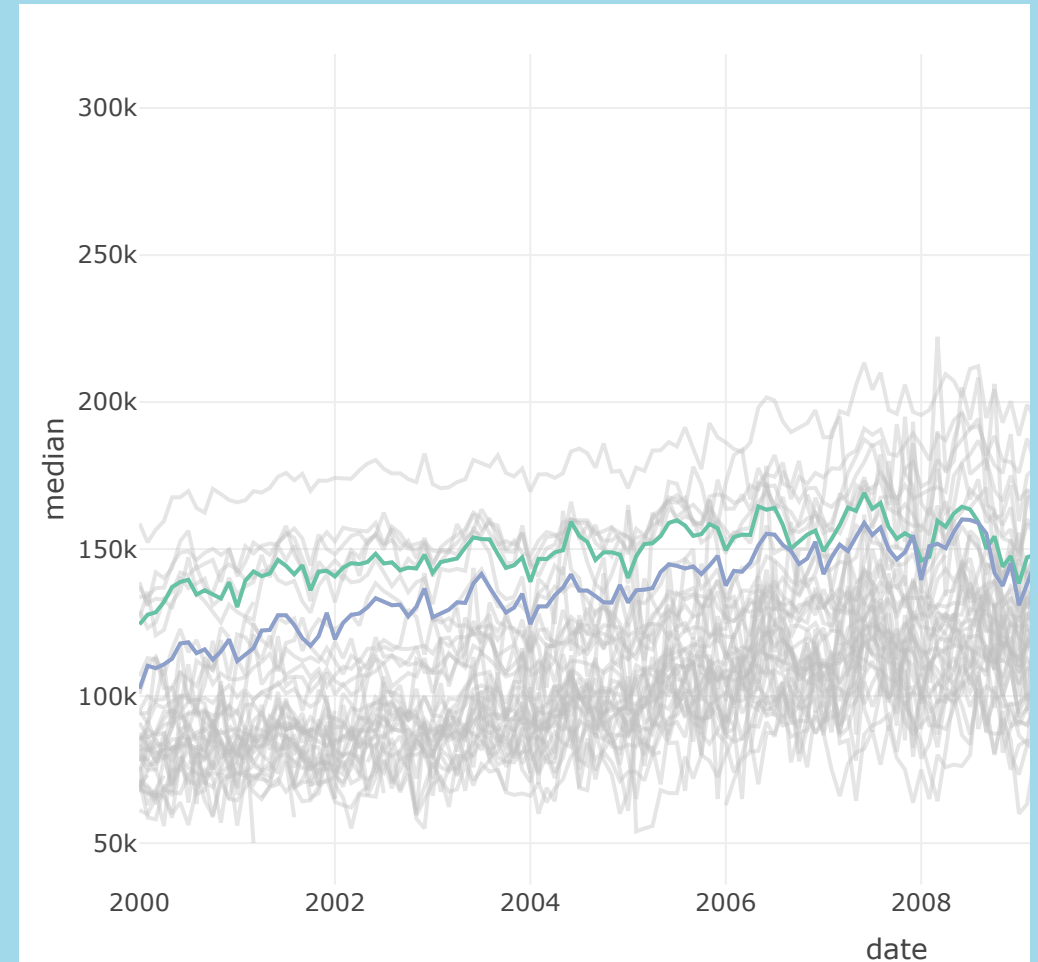
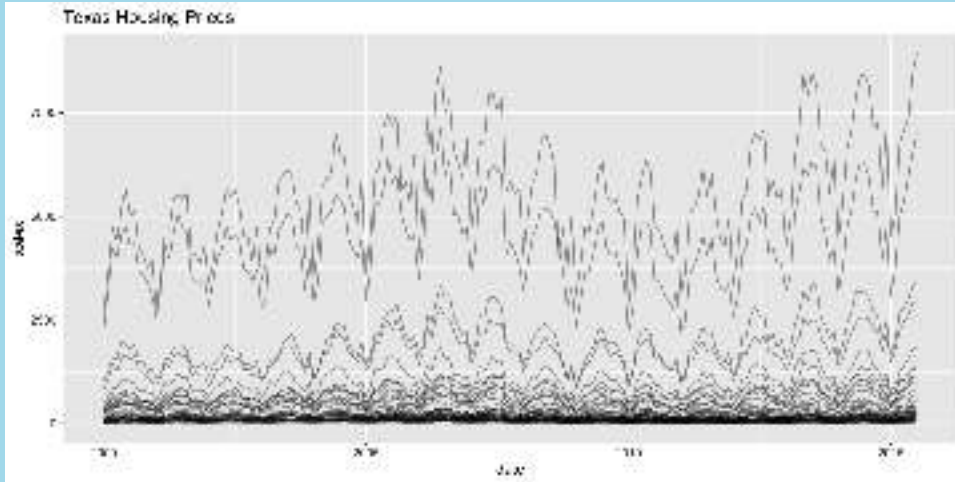
A note about APIs

- API: Application Programming Interface
- A software intermediary that allows two applications to “communicate”
- Lots of **R** packages rely on APIs to access data on the web (e.g., **tidycensus**)
- Facilitates reproducibility and powerful web applications built on **R** analyses
- May require “keys” and additional parsing (Mapbox and Google)

Interactive maps with **mapview** and **tmap**

- Easy extension of your existing
- Class Demo

Clarity in presentation (revisited)



Using `plotly`

- Syntax is similar to `ggplot`
- `hoverinfo` describes which elements you'd like to make interactive
- Other plot elements available (see `?plot_ly`)

Using **plotly**

```
1 g <- txhousing %>%
2   # group by city
3   group_by(city) %>%
4   # initiate a plotly object with date on x and median on y
5   plotly::plot_ly(x = ~date, y = ~median) %>%
6   # add a line plot for all texan cities
7   plotly::add_lines(name = "Texan Cities", hoverinfo = "none",
8                     type = "scatter", mode = "lines",
9                     line = list(color = 'rgba(192,192,192,0.4)')) %>%
10  # plot separate lines for Dallas and Houston
11  plotly::add_lines(name = ~city,
12                   data = filter(txhousing,
13                                 city %in% c("Dallas", "Houston")),
14                   hoverinfo = ~city,
15                   line = list(color = c("red", "blue")),
16                   color = ~city)
```


Animated maps with **tmap** and **gganimate**

::: columns ::: {.column width="40%"}

```
1 urb_anim = tm_shape(world) + tm_polygons() +  
2   tm_shape(urban_agglomerations) + tm_dots(size = "population_millions") +  
3   tm_facets(along = "year", free.coords = FALSE)  
4 tmap_animation(urb_anim, filename = "urb_anim.gif", delay = 25)
```

::: :: {column width="60%"}

1950

