Interactive R

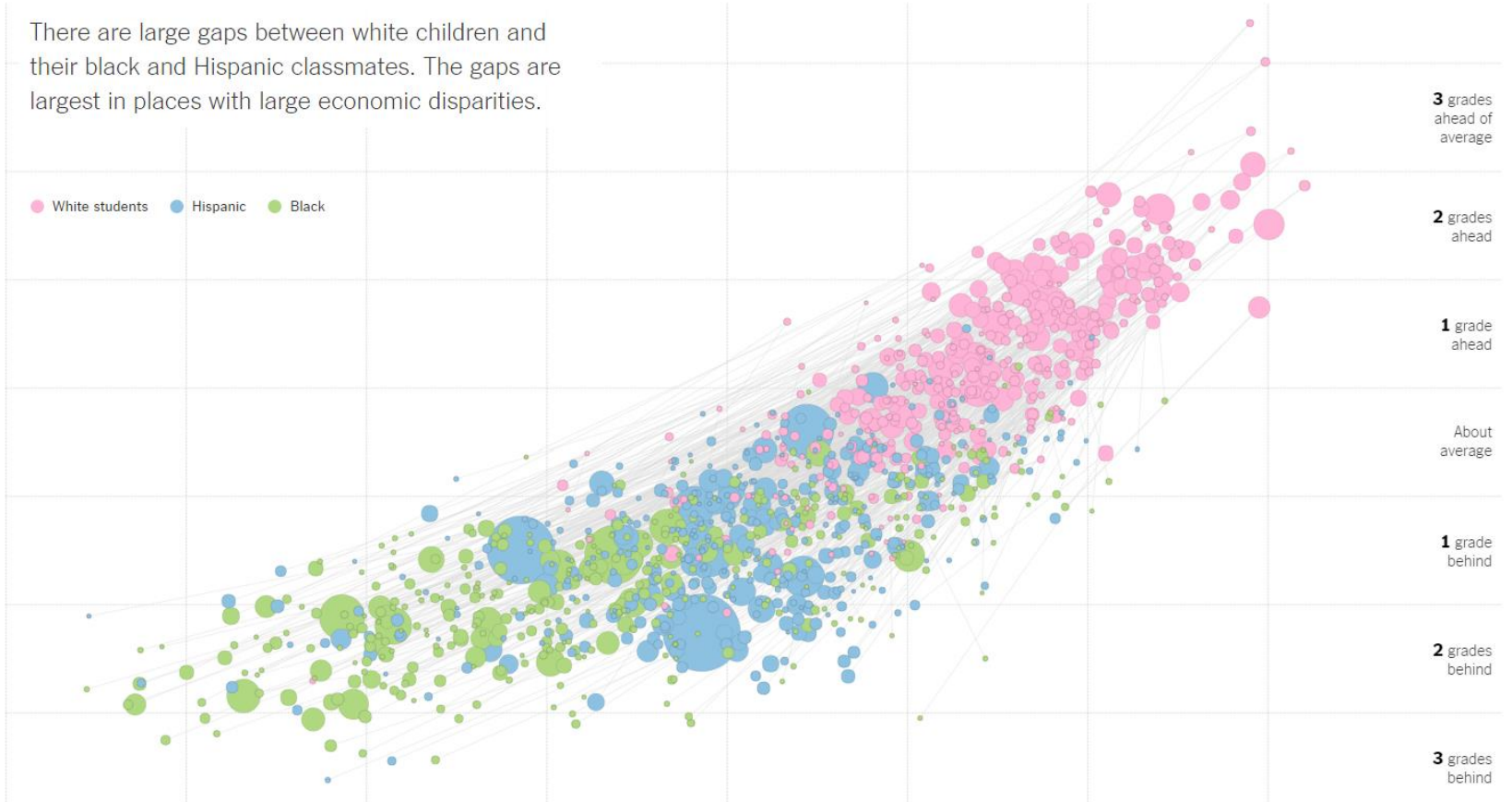
Bringing graphs to life

- Conveys an additional dimension of information
- Facilitates exploration and presentation of data

- Can be distracting
- Takes a while to learn and implement

Blame the Germans

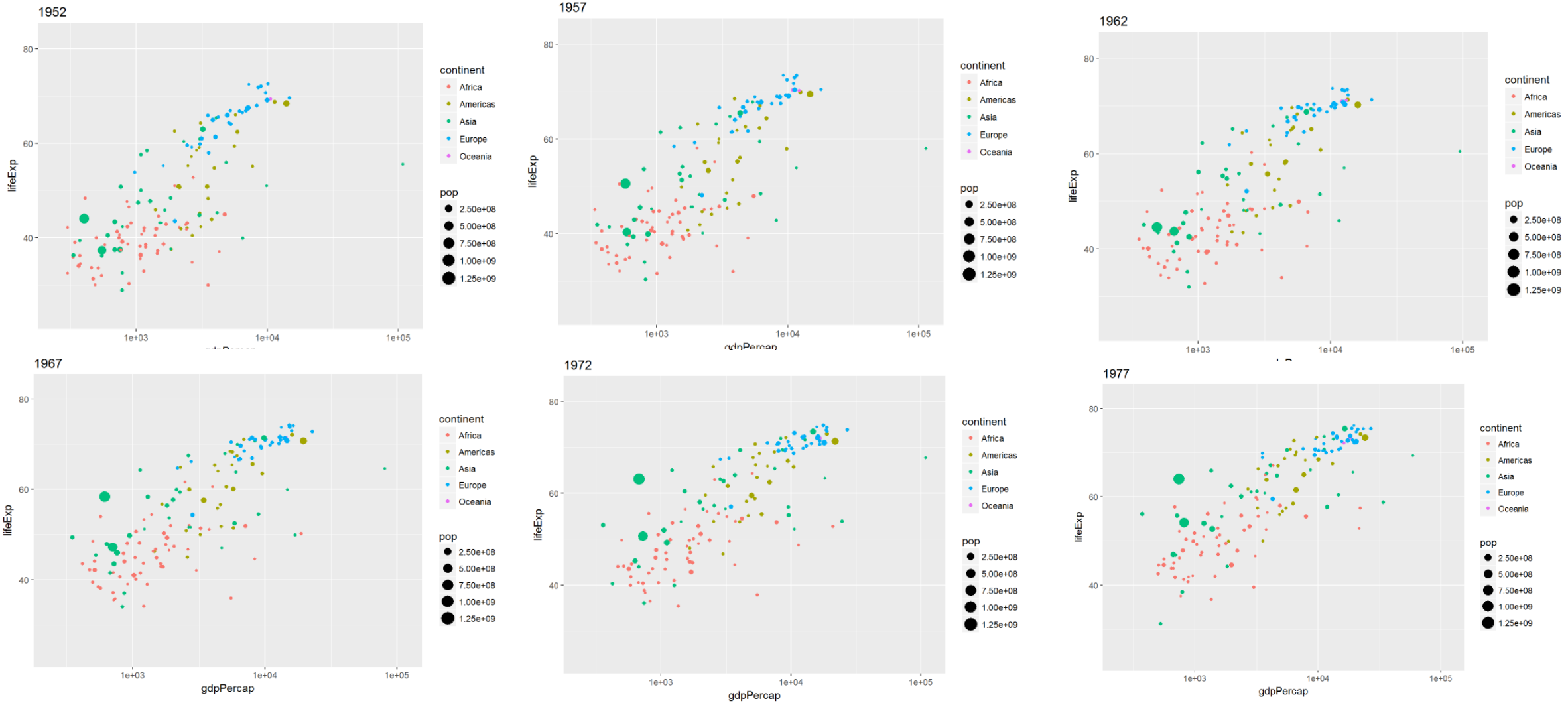
https://www.youtube.com/watch?v=jbkSRLYSojo&t=30



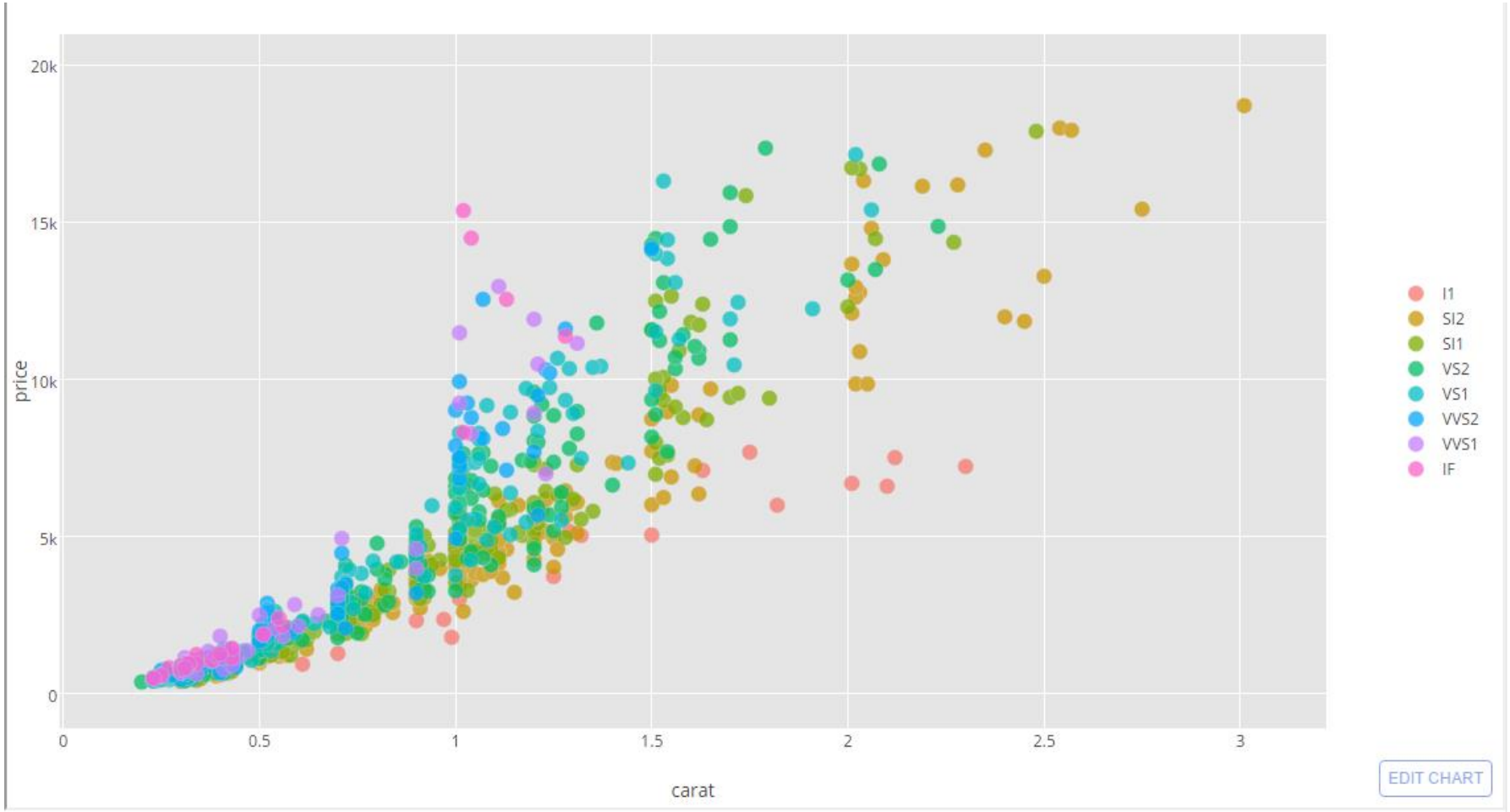
http://www.nytimes.com/interactive/2016/04/29/upshot/money-race-and-success-how-your-school-district-compares.html

Let’s get started: assembling frames

```
library(gganimate)
p <- ggplot(gapminder, aes(gdpPercap, lifeExp, size = pop, color = continent, frame = year)) +
  geom_point() +
  scale_x_log10()
gg_animate(p)
```



Adding details on graphs is easier than ever



https://plot.ly/ggplot2/

But the offer is confusing and uneven

- Constant steam of new packages enhancing R graphs with interactive capabilities
- Few remain maintained long enough to be worth the trouble
- The easier a package is to pick up, the more reliant it is on strong documentation and maintenance

How does it work? Short answer: it doesn’t

“There’s nothing particularly special about R expect for being the greatest software on earth”
_Amanda Cox

Or at least, not in R

R is great, but it needs the strengths from other ecosystems to offer all the tools we need

Or at least, not in R

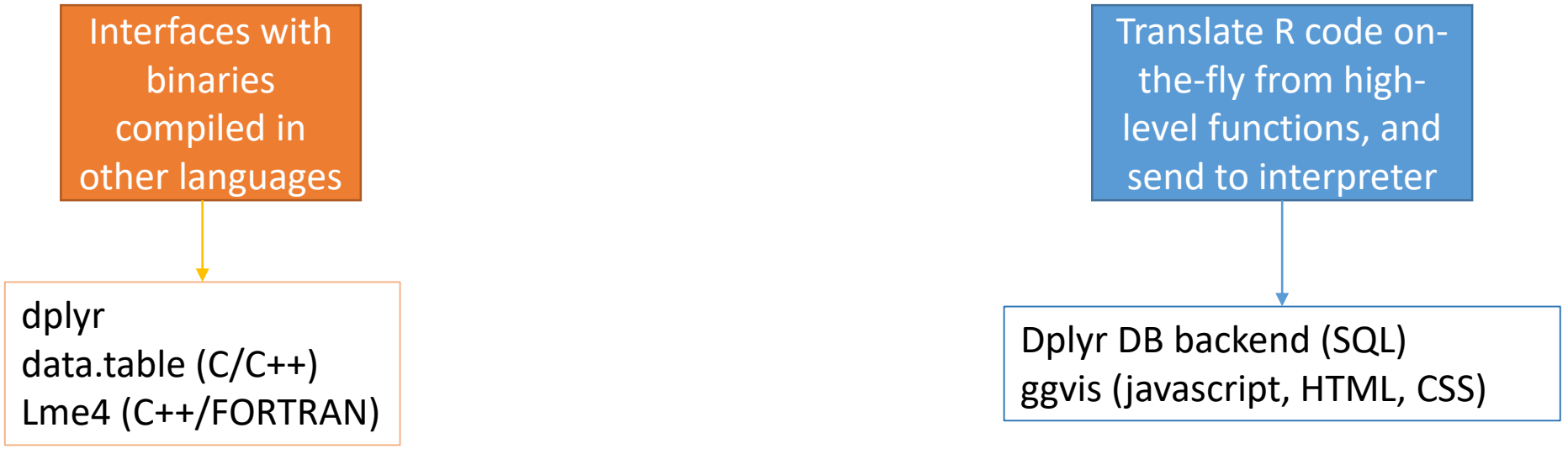
R is great, but it needs the strengths from other ecosystems to offer all the tools we need

Intercaces with binaries compiled in other languages

Translate R code on-the-fly from high-level functions, and send to interpreter

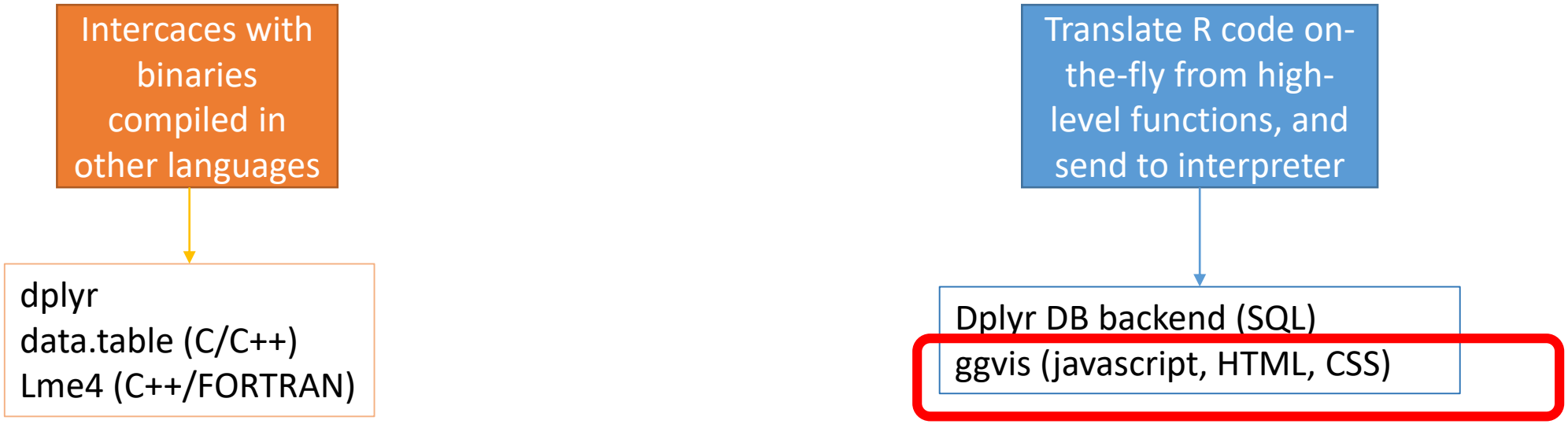
Or at least, not in R

R is great, but it needs the strengths from other ecosystems to offer all the tools we need

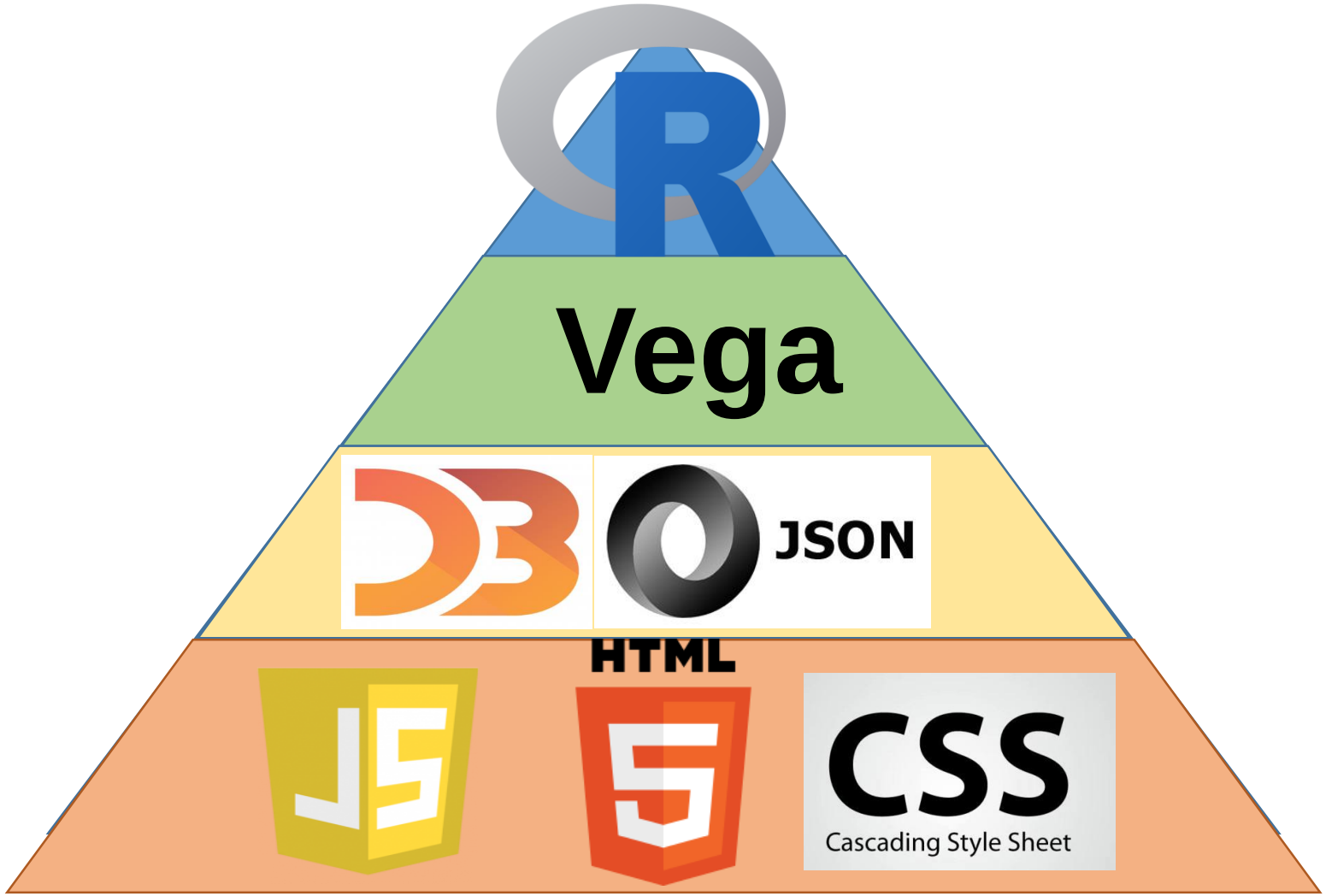


Or at least, not in R

R is great, but it needs the strengths from other ecosystems to offer all the tools we need



Long story short



This way to machine-level hellscape

(screams of the damned)

ggvis in 5 seconds

- Written by Hadley Wickham
- Superior performance vs. ggplot, but still under development
  - No faceting ☹
- Like ggplot, follows the “grammar of graphics”
- dplyr-like syntax, extensively uses piping
- Documentation is adequate, but functions are still changing

Anatomy of ggvis.

Data. Nearly all ggvis arguments are evaluated in the dt environment

```
filter(dt, year %in% c(1952, 2007)) %>%
  ggvis(y=~lifeExp, x=~gdpPercap, fill=~continent, shape=~factor(year)) %>%
  layer_points(opacity:=0.5)
```

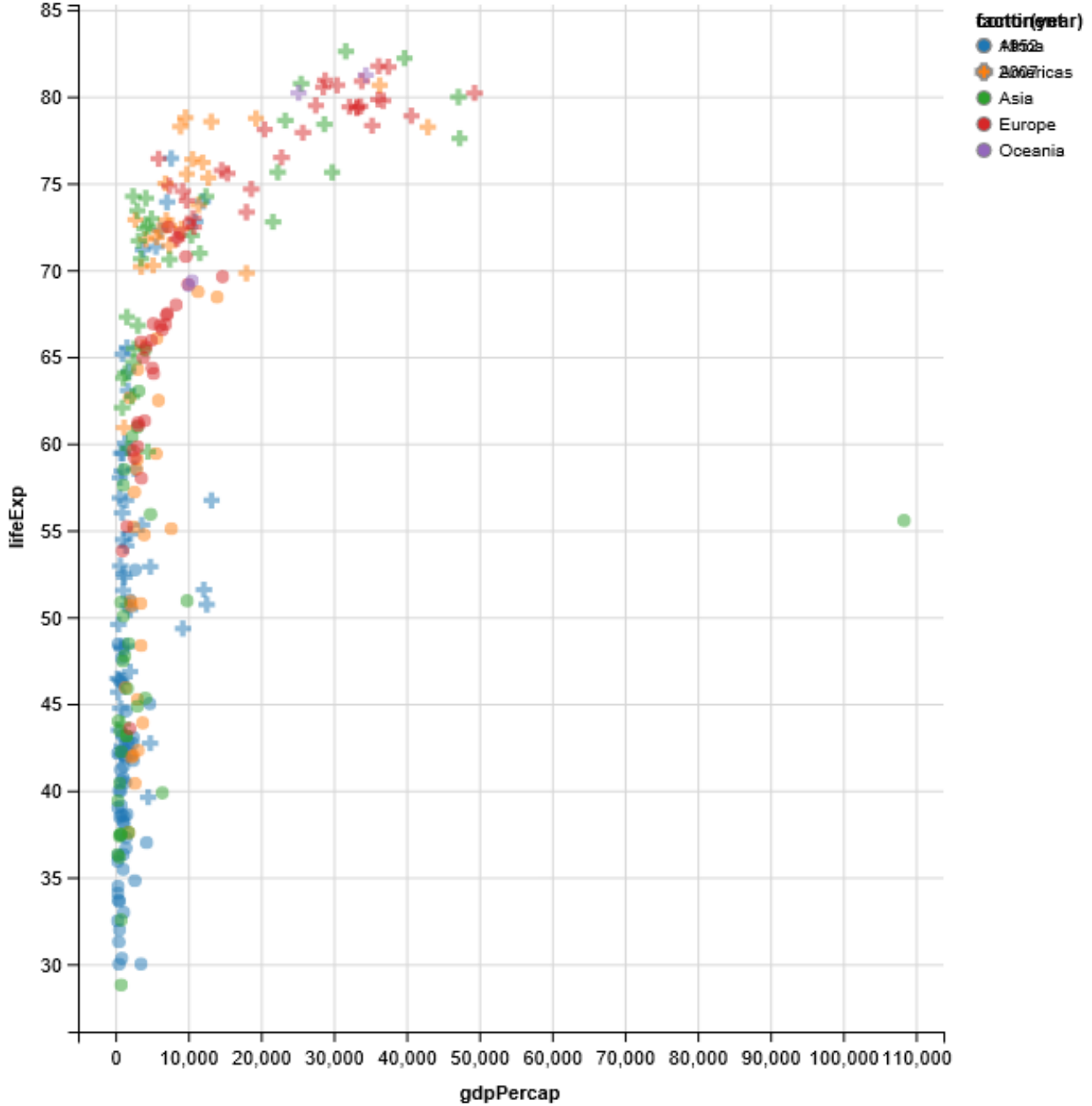
Visual properties ("props")

Layers can have their own visual properties, and they inherit from the ggvis() call by default.

```
filter(dt, year %in% c(1952, 2007)) %>%
  ggvis(y=~lifeExp, x=~gdpPercap, fill=~continent, shape=~factor(year)) %>%
  layer_points(opacity:=0.5)
```

Mapped variable in dt, use =~
Constant variable in dt, use :=~

Constant number, use :=


lifeExp
gdpPercap
85
80
75
70
65
60
55
50
45
40
35
30
0
10,000
20,000
30,000
40,000
50,000
60,000
70,000
80,000
90,000
100,000
110,000
Asia
Europe
Oceania

Scales are assigned to properties

```
dt %>% ggvis(y=~lifeExp, x=~log10(gdpPercap), fill=~continent) %>%
  scale_numeric("y", domain = c(30,80)) %>% scale_numeric("x", domain = c(2,5)) %>%
  filter(year == eval(input_slider(1952, 2007, step = 5, label = "Year"))) %>%
  layer_points(opacity:=0.5)
```

Evaluated interactive inputs can be given as parameters