Intro to ggvis

Hadley Wickham

@hadleywickham

Chief Scientist, RStudio

July 2015

1. Warmups
2. Basic plots
3. Layering
4. =~:=
5. Interactivity

Warmups

Your turn

What are the five most important types of plot? How do you draw them with ggplot2?

Basic plots

```
library(ggvis)
data("mpg", package = "ggplot2")
data("economics", package = "ggplot2")
```

```
?ggplot2::mpg
str(mpg)
```

```
ggvis(mpg, x = ~displ, y = ~hwy)
```

Data frame

Variable name

```
ggvis(mpg, x = ~displ, y = ~hwy)
```

Visual property

```
ggvis(mpg, ~displ, ~hwy)
```

Your turn

What other plot types can `ggvis()` make?

What happens if you plot a single continuous variable? A single categorical variable? What about continuous & categorical? Or categorical & continuous? Or date time?

```
ggvis(mpg, ~displ)
ggvis(mpg, ~displ, ~hwy)

ggvis(economics, ~date, ~psavert)

ggvis(mpg, ~drv)
ggvis(mpg, ~drv, ~hwy)

# Obviously still some work to do
ggvis(mpg, ~hwy, ~drv)
ggvis(mpg, ~drv, ~class)
```

```
# Other visual properties
ggvis(mpg, ~displ, ~hwy, fill = ~class)
ggvis(mpg, ~displ, ~hwy, shape = ~drv)
ggvis(mpg, ~displ, ~hwy, size = ~cty)
```

Your turn

Experiment with the visual properties.

What happens if you map a continuous variable to shape? Or colour? What happens if you map a categorical variable to size?

Layering

```
# ggvis uses a functional interface: you
# call functions to modify a plot

layer_points(ggvis(mpg, ~displ, ~hwy))

layer_smooths(
  layer_points(ggvis(mpg, ~displ, ~hwy))
)

# This gets tedious pretty fast, so ggvis
# uses %>% from magrittr

# (Interesting historical note: this is how
#  ggplot (not 2) worked!)
```

```
mpg %>%
  ggvis(~displ, ~hwy) %>%
  layer_points() %>%
  layer_smooths()
```

Layers

- Primitives: `points`, `paths` (`lines`), `rects`, `text`
- Distributions: `boxplots`, `densities`, `freqpolys`, `histograms`
- Models: `smooths`, `model_predictions`

Your turn

What are the parameters of `layer_smooths()`? How do you make the curve wigglier? How do you add standard errors?

```
library(dplyr)

mpg %>%
  filter(year == 1999) %>%
  ggvis(~displ, ~hwy) %>%
  layer_points() %>%
  layer_smooths()
```

```
mpg %>%
  group_by(cyl) %>%
  ggvis(~hwy) %>%
  layer_histograms(fill = ~cyl)

# A little bit of smarts for doing this
# automatically
mpg %>%
  group_by(drv) %>%
  ggvis(~hwy, stroke = ~drv)
```

Your turn

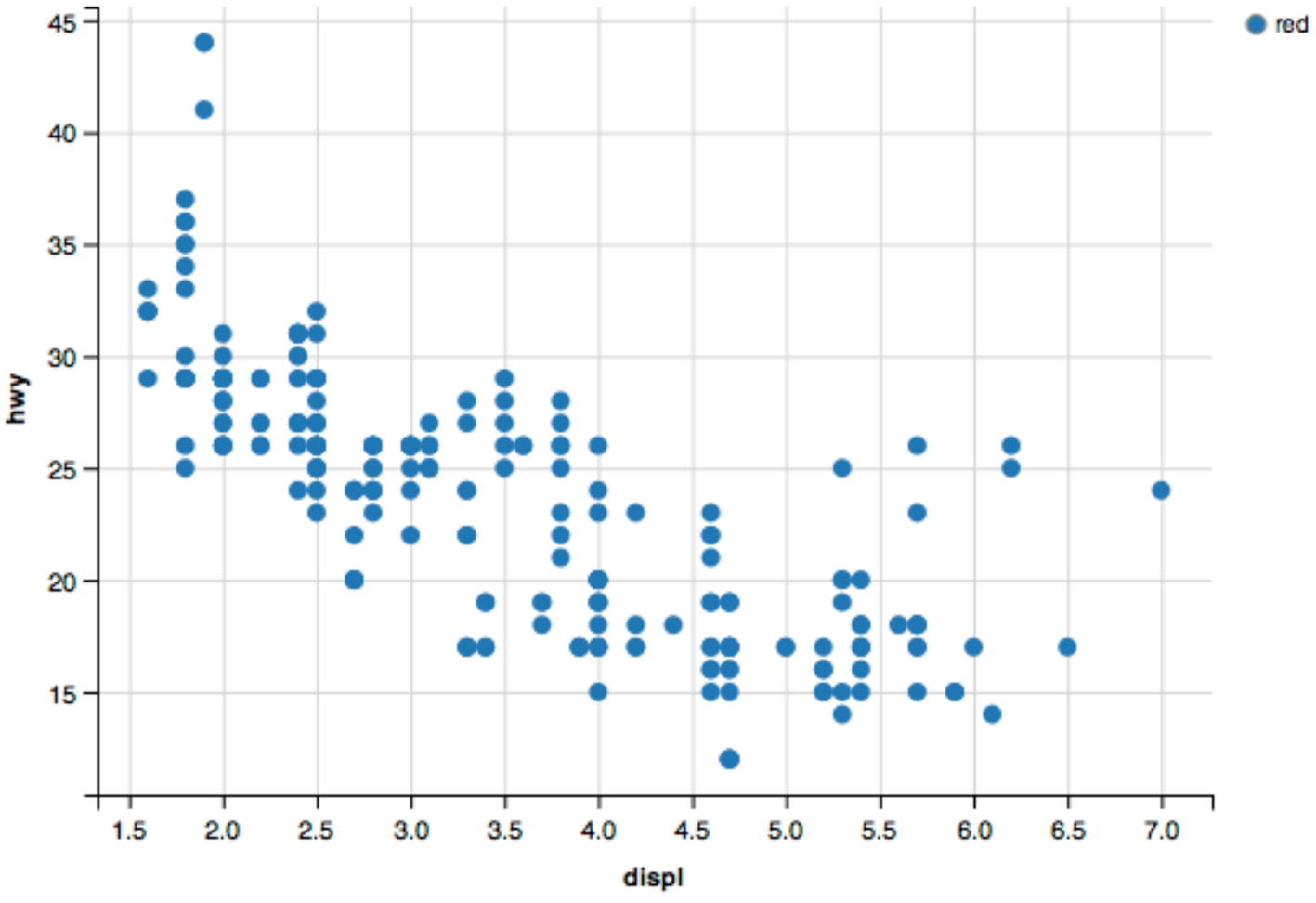
Read `?cocaine`. Use visualisations to explore the major determinants of the cost of cocaine in the US.

Which layers do you find most useful?

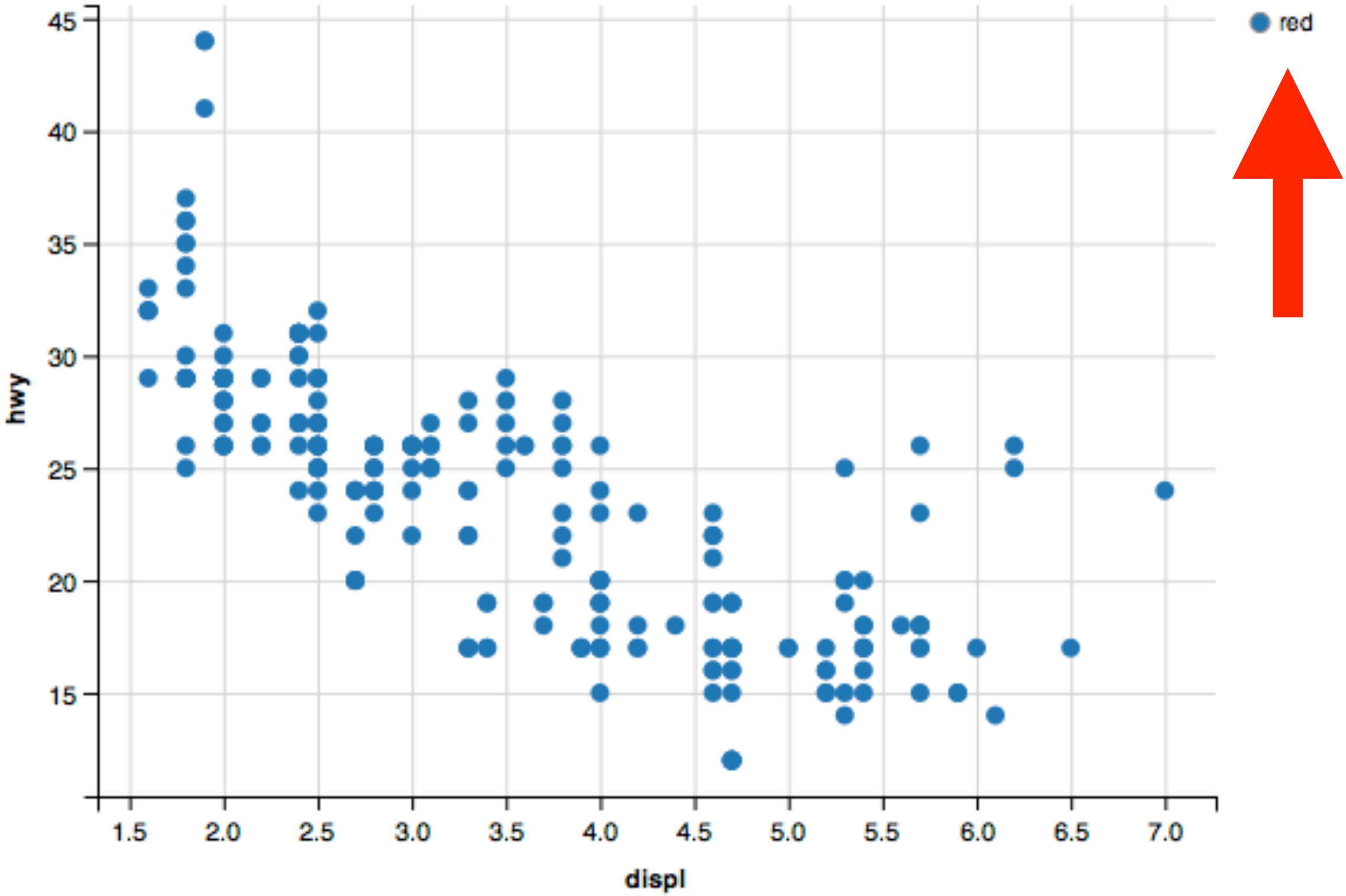
$= \quad := \quad \sim$

```
mpg %>%
  ggvis(~displ, ~hwy) %>%
  layer_points(fill = "red")
```

What do you think this plot will look like?



?!?!


red
45
40
35
30
25
20
15
hwy
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
5.5
6.0
6.5
7.0
displ

The RHS needs to be
scaled to a visual value

```
# A useful application

mpg %>%
  ggvis(~displ, ~hwy) %>%
  layer_points() %>%
  layer_model_predictions(model = "lm",
    stroke = "lm") %>%
  layer_smooths(stroke = "loess")
```

The RHS needs is already
a visual value

```
# To actually make the points red
mpg %>%
  ggvis(~displ, ~hwy) %>%
  layer_points(fill := "red")
```

```
# Sometimes the data is already scaled
df <- data.frame(
  x = 1:3,
  y = 1:3,
  col = c("red", "green", "blue")
)

ggvis(df, ~x, ~y, fill = ~col)
ggvis(df, ~x, ~y, fill := ~col)
```

	:=	=
~	`fill := ~colour`	`fill := "red"`
	`fill = ~class`	`fill = "name"`

	:=	=
~	–	fill = "red"
	aes(fill = class)	aes(fill = "name")

```
df <- data.frame(
  x = 1:3,
  y = 1:3
)

xvar <- ~x
yvar <- ~y

ggvis(df, xvar, yvar)
```

Interaction

```
# The coolest thing about ggvis is that plot
# parameters don't need to be static: they
# can be interactive

input_text() # text box
input_slider() # numeric slider
input_select() # dropdown box
input_checkbox() # checkbox
```

```
mpg %>%
  ggvis(~displ, ~hwy) %>%
  layer_points() %>%
  layer_smooths(
    span = input_slider(0.2, 1)
  )
```

Console ~/Dropbox (RStudio)/15-uzurich/code-slides/1-ggvis/

```
> mpg %>%
+   ggvis(~displ, ~hwy) %>%
+   layer_points() %>%
+   layer_smooths(
+     span = input_slider(0.2, 1)
+   )
Showing dynamic visualisation. Press Escape/Ctrl + C to stop.
```

Your turn

Add a dropdown that allows you to pick the colour of the line.

Draw a density plot with bandwidth connected to a slider

Why doesn't this work?

```
mpg %>%
  ggvis(~displ, ~hwy) %>%
  layer_points(fill = input_select(names(mpg)))
```

```
mpg %>%
  ggvis(~displ, ~hwy) %>%
  layer_points(fill = input_select(names(mpg),
    map = as.name))

# map = as.name turns a string into a name
# of a variable
```

Your turn

What does `waggle()` do? What does `up_down()` do?

Create a scatterplot where size is controlled by up and down arrows, and opacity is controlled by left and right arrows.