6 - Polishing your plots

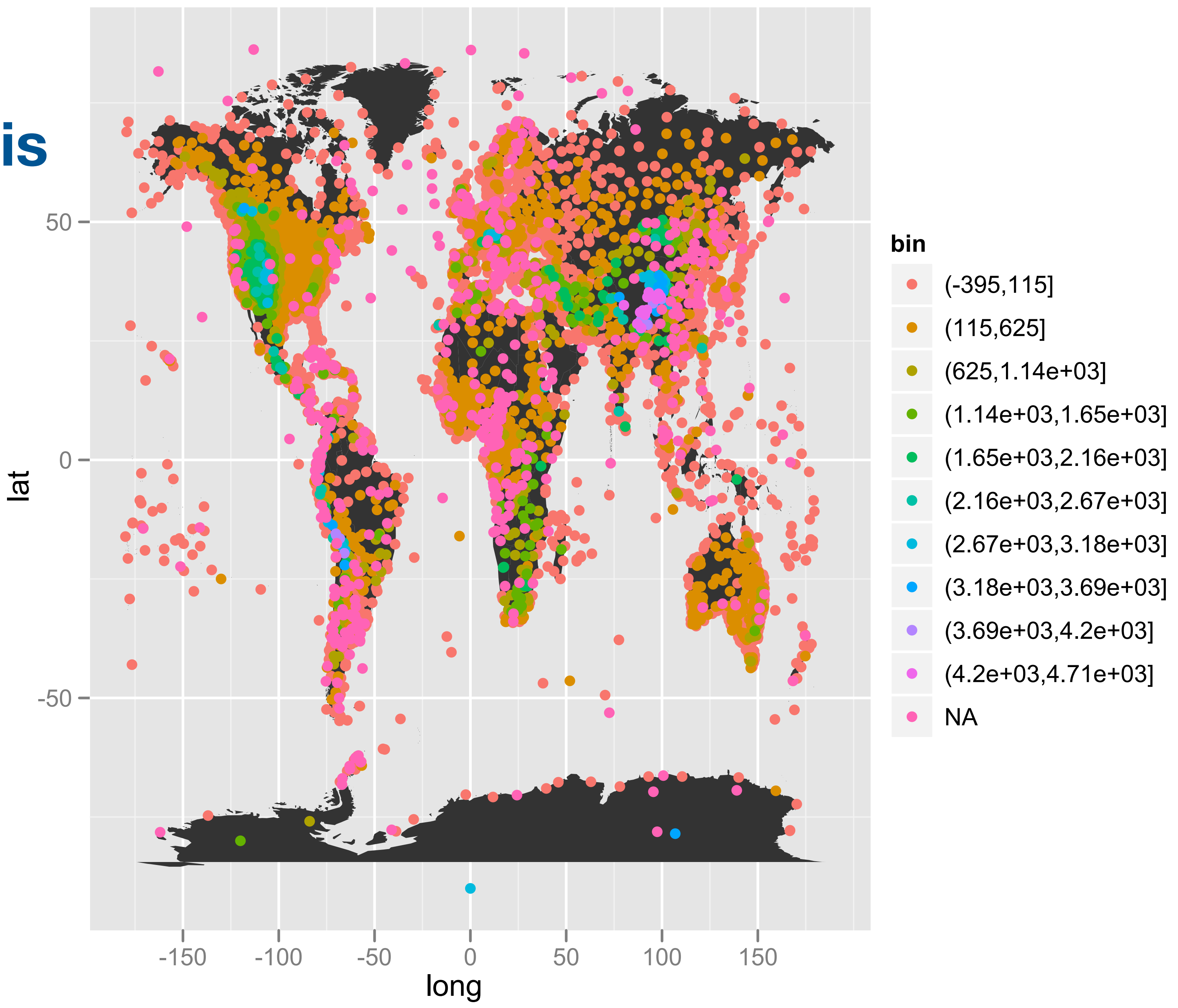
R workshop

- Advanced Graphics -

Outline

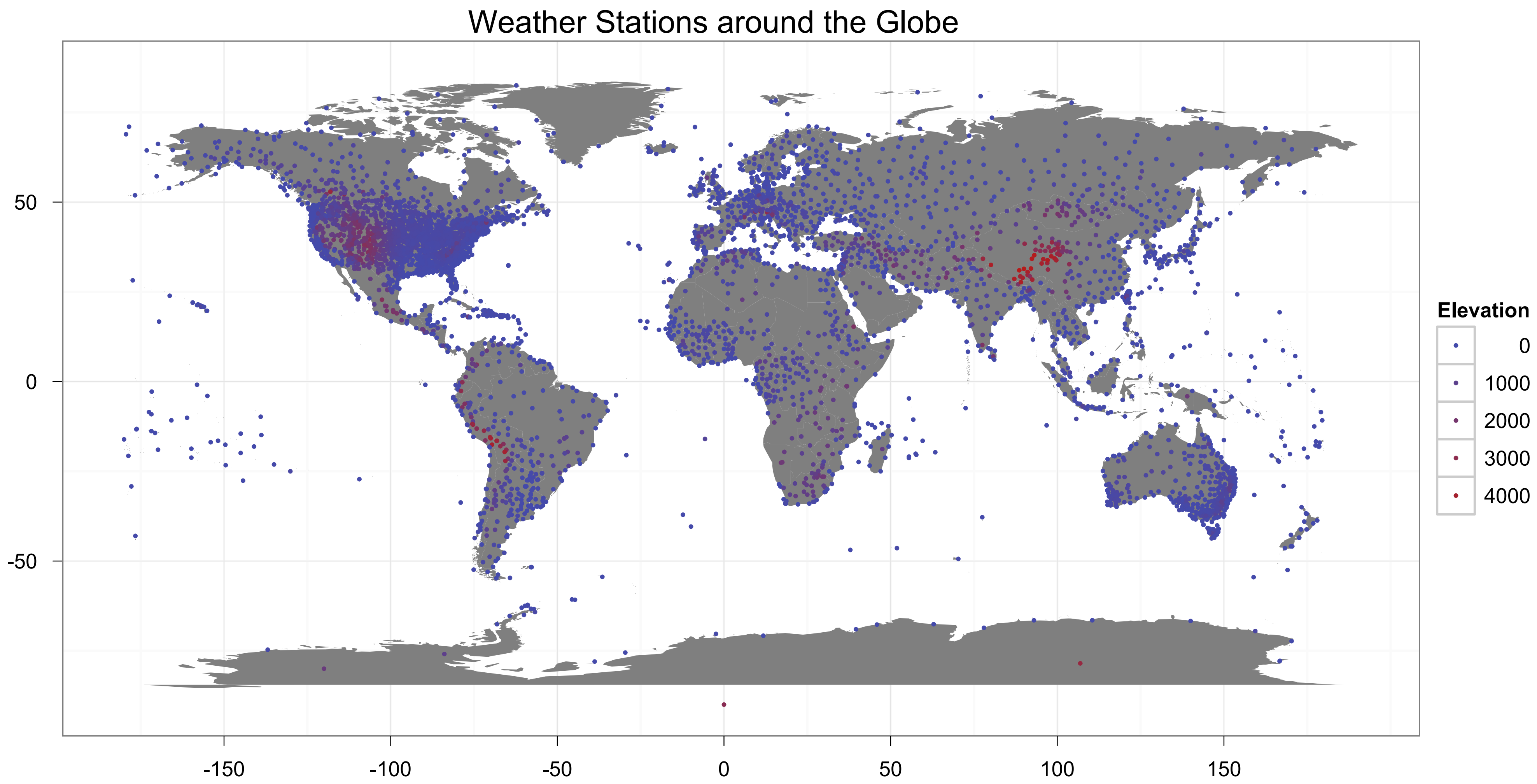
- Finishing Touches
  - options
  - themes
  - saving plots

What's wrong with this plot?



Some problems

- Incorrect coordinate system (aspect ratio)
- Bad color scheme
- Unnecessary axis labels
- Legend needs improvement: better title and better key labels
- No title



Weather Stations around the Globe
50
0
-50
-150
-100
-50
0
50
100
150
Elevation
0
1000
2000
3000
4000

Outline

- **Themes**: control presentation of non-data elements.
- **Saving your work**: to include in reports, presentations, etc.

Visual appearance

So far have only discussed how to get the data displayed the way you want, focussing on the essence of the plot.

Themes give you a huge amount of control over the appearance of the plot, the choice of background colours, fonts and so on.

```
# Two built in themes.  The default:
> qplot(carat, price, data = diamonds)

# And a theme with a white background:
> qplot(carat, price, data = diamonds) +
theme_bw()

# Use theme_set if you want it to apply to
every
# future plot.
> theme_set(theme_bw())

# This is the best way of seeing all the
default
# options
> theme_bw()
> theme_grey()
```

Plot title

You can change this for an individual plot by

```
labs(title = "My title")

ggtitle("My title")
```

Elements

You can also make your own theme, or modify and existing.

Themes are made up of elements which can be one of:

`element_line`, `element_text`, `element_rect`, `element_blank`

Gives you a lot of control over plot appearance.

Elements

Axis: `axis.line, axis.text.x, axis.text.y, axis.ticks, axis.title.x, axis.title.y`

Legend: `legend.background, legend.key, legend.text, legend.title`

Panel: `panel.background, panel.border, panel.grid.major, panel.grid.minor`

Strip: `strip.background, strip.text.x, strip.text.y`

```
# To modify a plot
> p + theme(plot.title =
  theme_text(size = 12, face = "bold"))
> p + theme(plot.title =
   element_text(colour = "red"))
> p + theme(plot.title =
   element_text(angle = 45))
> p + theme(plot.title =
   element_text(hjust = 1))
```

```
# If we want, we could also remove
the axes:
> last_plot() + theme(
+   axis.text.x = element_blank(),
+   axis.text.y = element_blank(),
+   axis.title.x = element_blank(),
+   axis.title.y = element_blank(),
+   axis.ticks.length = unit(0,
"cm"),
+   axis.ticks.margin = unit(0,
"cm"))
```

Saving your work

Raster	Vector
pixel-based	instruction-based
png	pdf
for plots with many points	for all other plots
ms office, web	latex

Output

```
> qplot(price, carat, data =
diamonds)
> ggsave("diamonds.png")

# Selects graphics device based
on extension
> ggsave("diamonds.png")
> ggsave("diamonds.pdf")
```

```
# Uses on-screen device size, or override
with
# width & height (to be reproducible)
> ggsave("diamonds.png", width = 6, height
= 6)

# Outputs last plot by default, override
# with plot:
> dplot <- qplot(carat, price, data =
diamonds)
> ggsave("diamonds.png", plot = dplot)

# Defaults to 300 dpi for png
> ggsave("diamonds.png", dpi = 72)
```

Your turn

Save a pdf of a scatterplot of price vs carat. Open it up in adobe acrobat.

Save a png of the same scatterplot and embed it into word.