From drab to fab

Teaching visualisation via incremental improvements

Mine Çetinkaya-Rundel and Maria Tackett

February 2020

In our September 2014 column titled we wrote about five concrete reasons your students should be learning to analyse data in the reproducible paradigm. One of the reasons we gave was that "[t]his workflow establishes the norm that data analysis and science in general should be an iterative process". Just like data analysis in general, creating an effective data visualisation is also an iterative process.

It can be very inspirational to share highly polished data visualisations with students. A few of our favourite visualisations to bring to the classroom are text analysis of Donald Trump's tweets by David Robinbson (varianceexplained.org/r/trump-tweets), the Dialect Survey by Joshua Katz (joshkatz.co/dialect.html), and map of America's weather related disasters by Tim Meko (washingtonpost.com/graphics/2019/national/mapping-disasters). We also love the NY Times series What's Going on in This Graph? (nytimes.com/column/whats-going-on-in-this-graph) as well as just about any visualisation from the Flowing Data blog by Nathan Yau (flowingdata.com), The Pudding (pudding.cool), The Functional Art blog by Alberto Cairo (thefunctionalart.com), and Chartable (blog.datawrapper.de). For example, the stories on how the American work day changed over 15 years (flowingdata.com/2019/07/16/how-the-american-work-day-changed-in-15-years) and on gender tropes in film with screen direction (pudding.cool/2017/08/screen-direction) both feature in-depth and enlightening but simple visualisations that tell a meaningful story. We like these examples because their message is clear enough to glean quickly, but one can extract deeper insights by looking at them for a bit longer, a bit more closely.

All of these visualisations share one thing in common – they are incredibly polished. It is possible for a student just starting off to think "well, I could never do that!". Or perhaps worse, to think that they can, and not get to something as impressive on their first or second try and get discouraged. We believe that it's important to show students not only the end result, but also the process of building a meaningful data visualisation. This journey can be incredibly educational in its own, as each step of the improvement is an opportunity to teach a new data visualisation or wrangling technique.

In this column we present three examples of "from drab to fab" type class activities, i.e. improving data visualisations. The first example is on types of staff involved with instruction at American universities and how their distributions have changed over time. We walk through this example in detail, showing some of the code and step-by-step improvements on the visualisations. We then present two more examples of such activities, one on electric skateboards and the other on fisheries. The data for all three examples along with R code to produce the visualisations presented here can be found on the GitHub repository accompanying this column at bit.ly/taking-a-chance.

The drab

The data come from The American Association of University Professors (AAUP), which is a nonprofit membership association of faculty and other academic professionals. In 2013, the AAUP published a report on Trends in Instructional Staff Employment Status. The plot shown in Figure 1 is very similar to a plot presented in this report.

The visualisation is a bit difficult to make sense of, and this is the point. We show this visualisation to the students at the beginning of the class, with some clarification on what the y-axis means (various levels of staff at the university who teach classes, ranging from graduate students to full-time tenured faculty) and ask them to interpret the visualisation in teams. Eventually students are able to identify the main message of the

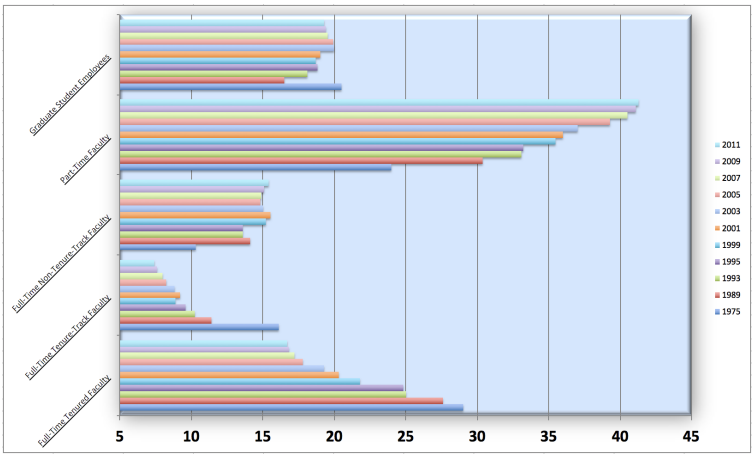


Figure 1: Reproduction of visualisation included in the 2013 AAUP report on titled Trends in Instructional Staff Employment Status

story: the highest proportion of instructors are part-time faculty, and this proportion has been increasing from 1975 to 2011. Along with this main message, students also naturally come to the conclusion that this is not the best visualisation of these data, and in fact, the visualisation hides the main message instead of making it loud and clear.

At this point it's a worthwhile exercise to ask students to start thinking about how this visualisation can be improved. Instead of just talking through it, we ask students to sketch the visualisations on a piece of paper. They're not expected to actually create an accurate visualisation via a sketch, but we recommend they label their axes, mark their axis ranges, and decide on which geometric object(s) they plan on using to represent their data (points, lines, bars, etc.). Figure 2 is an example of such a sketch.

Having thought through what the end result should look like, and especially having put it down on paper with a sketch, gives the students a path to follow in the next step when they're using software to build this improved visualisation.

Onto the fab

Next we walk through the steps for improving this visualisation. One can do this during class, with interactive exercises and just-in-time teaching sprinkled throughout, or give it as an assignment students work on outside of class. For the latter, we have had success in assigning this as a team exercise in which students write a critique of the original visualization and then create an improved one. After a few days, teams present their visualizations during class while describing how they have improved upon the original one. If the class size allows, it's also a great idea to then have students present their process (both technical and thought processes) for improving the visualization, so they can see the various approaches each team has taken.

In the walk through of a possible path to "fab" (an improved visualisation) that we present below, we highlight opportunities for instruction along the way, using R as the programming language and the **tidyverse** suite of packages. We also present a series of questions that can be used as part of a class activity or assignment. We deliberately keep steps simple, improving only one or two features of the visualisation at a time, in an effort to emphasize that creating an effective data visualisation is an iterative process. The visualizations in this path is representative of the variety of those created by students across multiple semesters of doing this assignment in introductory data science courses.

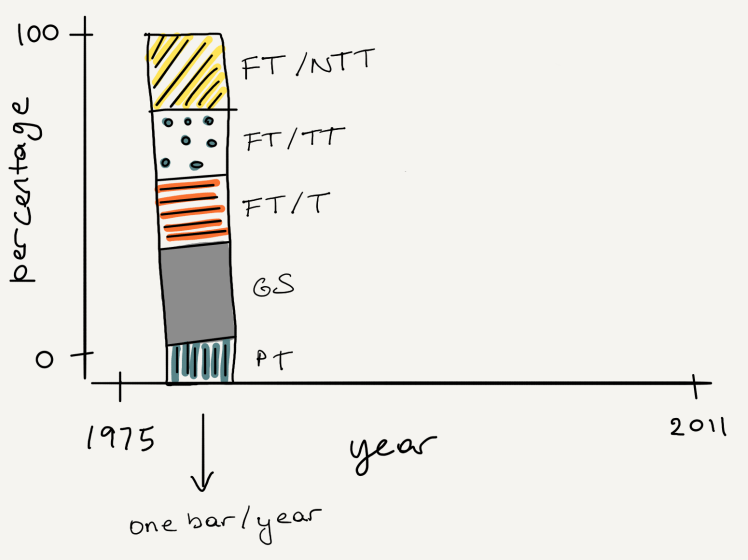


Figure 2: Example of sketch of redesigned visualisation.

Step 1. Get the data

The dataset provided in the AAUP report is in the following format, and this is exactly how we recommend showing it to the students before distributing it.

faculty_type	1975	1989	1993	1995	1999	2001	2003	2005	2007	2009	2011
Full-Time Tenured Faculty	29.0	27.6	25.0	24.8	21.8	20.3	19.3	17.8	17.2	16.8	16.7
Full-Time Tenure-Track Faculty	16.1	11.4	10.2	9.6	8.9	9.2	8.8	8.2	8.0	7.6	7.4
Full-Time Non-Tenure-Track Faculty	10.3	14.1	13.6	13.6	15.2	15.5	15.0	14.8	14.9	15.1	15.4
Part-Time Faculty	24.0	30.4	33.1	33.2	35.5	36.0	37.0	39.3	40.5	41.1	41.3
Graduate Student Employees	20.5	16.5	18.1	18.8	18.7	19.0	20.0	19.9	19.5	19.4	19.3

> ***Question 1.*** *What does each row in the dataset represent? What does each column represent? What do the numbers represent? What does each column add up to, and why?*

We can load the data frame using the following, and call it `staff_wide`.

```
library(tidyverse)
staff <- read_csv("http://bit.ly/chance-staff")
```

Step 2. Reshape the data

Our goal is to create a visualisation similar to the sketch in Figure 2. In order to do this we need a column for year, a column for percentage, and a column for faculty staff type. However, our dataset is currently not in this format. This presents a great opportunity for some just-in-time teaching for reshaping data from wide to long format. This data wrangling step is quite common when working with time series and with repeated measures data, and hence it's a useful skill for students to have.

> ***Q2.*** *If the long data will have a row for each year/faculty type combination, and there are 5 faculty types and 11 years of data, how many rows will the data have?*

We can accomplish this task using the `pivot_longer()` function, which also has a counterpart `pivot_wider()` for going from long to wide datasets. In the next code chunk we start with the wide dataset and we pivot all

of its columns except for faculty type (`cols = -faculty_type`), placing column names in a new variable called `year` and placing the values in the original data frame in a new variable called `percentage`.

```
staff_long <- staff_wide %>%
  pivot_longer(cols = -faculty_type, names_to = "year", values_to = "percentage")
```

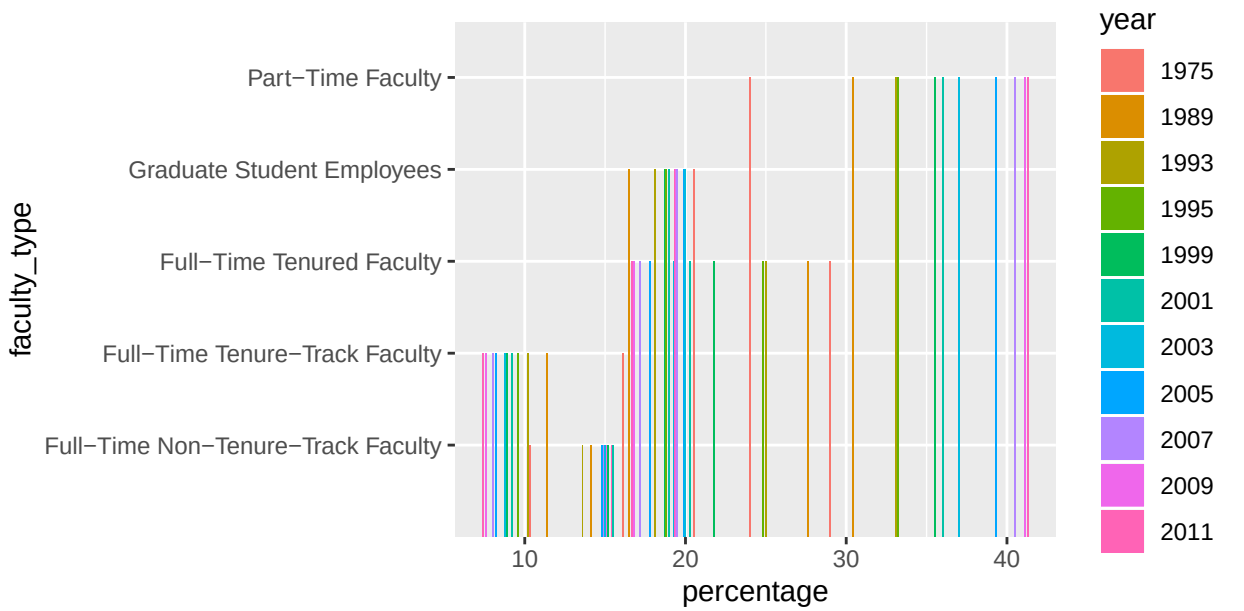
```
## # A tibble: 55 x 3
##   faculty_type              year  percentage
##   <chr>                     <chr>      <dbl>
## 1 Full-Time Tenured Faculty 1975        29
## 2 Full-Time Tenured Faculty 1989        27.6
## 3 Full-Time Tenured Faculty 1993        25
## 4 Full-Time Tenured Faculty 1995        24.8
## 5 Full-Time Tenured Faculty 1999        21.8
## # ... with 50 more rows
```

Q3. *What does each row represent in the long dataset? How does this differ from what each row represents in the wide dataset?*

Step 3. Recreate the plot

A good way to get started with improving a visualisation is to first recreate it. We can do that by placing `percentage` on the x-axis and `faculty_type` on the y-axis and filling the bars by `year`.

```
ggplot(data = staff_long, aes(x = percentage, y = faculty_type, fill = year)) +
  geom_col(position = "dodge")
```

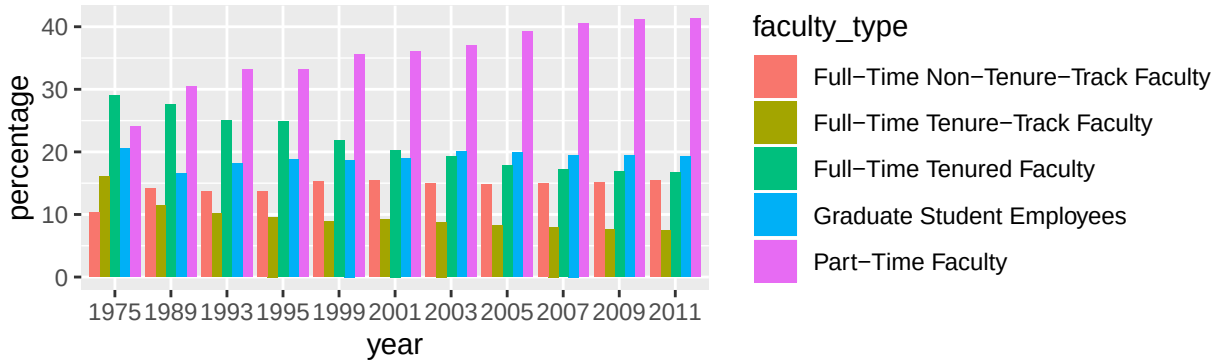


Now that we know we can match out starting point (we haven't matched it exactly, but we're close enough to be confident that we're working with the same data), we can start making improvements towards our sketch.

Step 4. Recreate the sketch

We'll start by changing variable mappings to certain aesthetics of the plot, like the `x` and `y` axes and which variable determines how we `fill` the bars.

```
ggplot(data = staff_long, aes(x = year, y = percentage, fill = faculty_type)) +
  geom_col(position = "dodge")
```

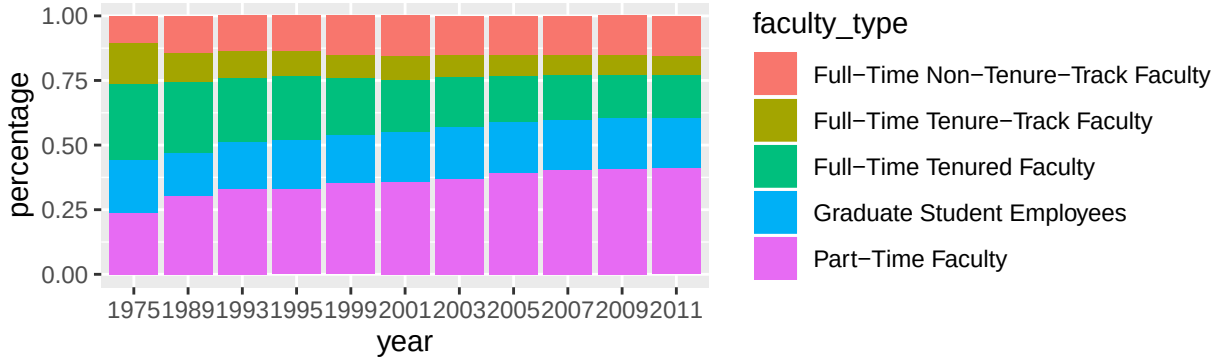


Q4. *In what way(s) is this plot an improvement on the original "drab" visualization?*

Q5. *In what way(s) is the main message still unclear / hidden?*

Next, let's convert it to a segmented bar plot by changing the position of the columns from `dodged` to `fill`. This not only looks like our sketch, but it also is starting to tell the story of increasing percentages of part-time faculty instructional faculty over the years.

```
ggplot(data = staff_long, aes(x = year, y = percentage, fill = faculty_type)) +
  geom_col(position = "fill")
```



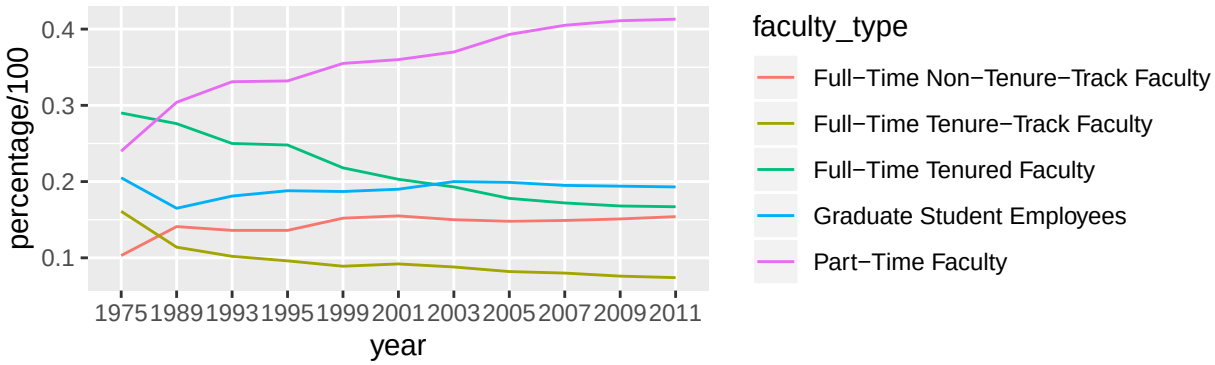
Q6. *In what way(s) is the segmented bar plot an improvement on the the side-by-side bar plot from the previous step?*

Q7. *In what way(s) is the main message still unclear / hidden?*

Step 5. Improve further

Most students will get to this point and call it a day once they have successfully created the plot they envisioned in their sketch using R. This presents a good opportunity to get them to think about alternate geometric objects to represent the data. For example, line plots are common in time series data, so they are worth a try.

```
ggplot(staff_long, aes(x = year, y = percentage/100, group = faculty_type, color = faculty_type)) +
  geom_line()
```



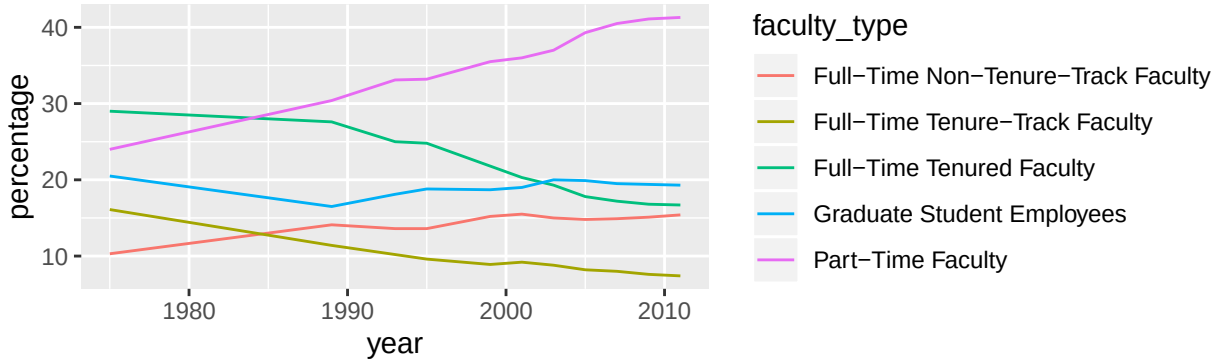
The lines are an improvement over the bars, but let's take a look at the x-axis. When the data was reshaped in Step 1, `year` was stored as a character variable. Therefore, the years are sorted alphabetically (which happens to match the numerical sorting) and are equally spaced in our visualization, giving the impression that the data were collected at equally spaced time intervals.

Let's first transform `year` to be a numerical variable:

```
staff_long <- staff_long %>% mutate(year = as.numeric(year))
```

and then re-plot the data.

```
ggplot(staff_long, aes(x = year, y = percentage, group = faculty_type, color = faculty_type)) +
  geom_line()
```

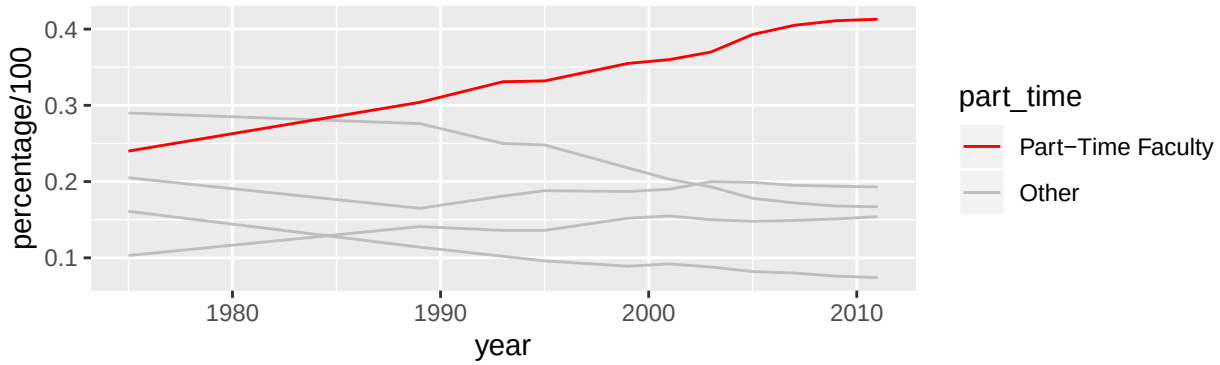


Q8. *If the main goal of this visualisation is to draw attention to the increasing percentage of part-time faculty over the years, how can we make this message stronger?*

One option is to use colors to draw attention to the part-time faculty line. We can accomplish this by creating a binary variable (`part_time`) that indicates whether the faculty type is part-time or something else. The `fct_other()` function is one quick and straightforward way of accomplishing this. Then, we can use red to indicate part-time and a much more muted colour (grey) to indicate all other categories

```
staff_long <- staff_long %>%
  mutate(part_time = fct_other(faculty_type, keep = "Part-Time Faculty"))

ggplot(staff_long, aes(x = year, y = percentage/100, group = faculty_type, color = part_time)) +
  geom_line() +
  scale_color_manual(values = c("red", "gray"))
```



Now the main message of the story is clear: the highest proportion of instructors are part-time faculty, and this proportion has been increasing from 1975 to 2011. In the previous line plots, it was unclear where one should focus their attention since each faculty type was a distinct and vibrant colour. By making the lines for the other faculty levels grey, one is still able to see the trends for those faculty types without distracting from the trend of the primary faculty type of interest in this story, part-time faculty.

Step 6. Formatting and annotation

There are still a number of features of this plot that could use improving:

1. Percentages are missing the % sign (we'll use the `percent_format()` function from the **scales** package for this)
2. The labels are poorly formatted and the data source is not indicated
3. The grey lines on grey background don't offer much contrast
4. Legend is taking up too much space on the right hand side

Suppose we store the plot from the previous step as `p`. Then, we can add the necessary layers to `p` to address these updates.

```
library(scales)
p +
  scale_y_continuous(labels = percent_format(accuracy = 1)) +              # %s
  labs(                                                                    # labels
    title = "Percentage of part-time faculty instructors is on the rise",
    x = "Year", y = "Percentage", color = "",
    caption = "Source: bit.ly/taking-a-chance"
  ) +
  theme_minimal() +                                                        # background
  theme(legend.position = "bottom")                                        # legend
```

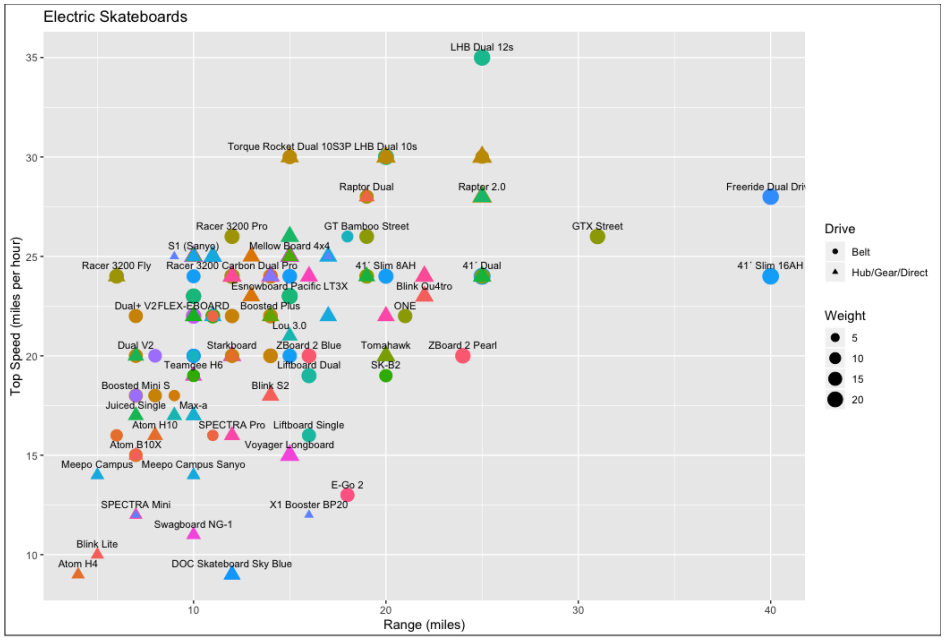
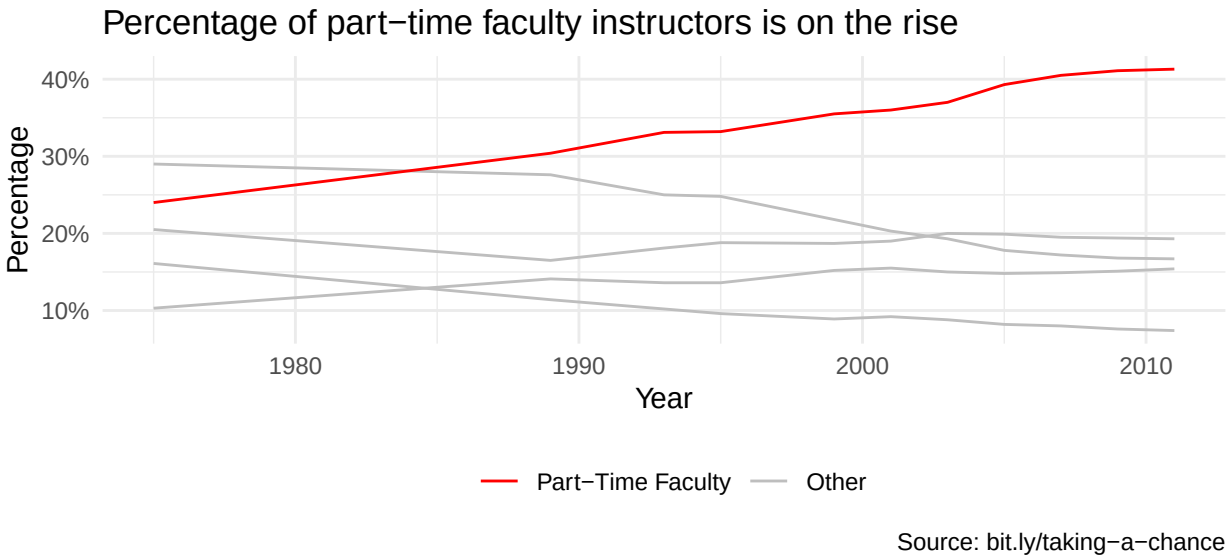


Figure 3: Comparison of specifications for electric skateboards.



Similar exercises

Electric skateboars

Electric Skateboard HQ is a review and discussion site on, you guessed it, electric skateboards. They summarise their comparison of various electric skateboards using the plot shown in Figure **??** is an example of such a sketch. The primary issue with this visualisation is that it's trying to convey too much information at once, and hence it is a good example to discuss the appropriate level of information any single visualisation should convey.

The data for used to create this plot may be found at bit.ly/chance-skateboard. A class activity / assignment around this example should begin by having the students list of all the variables included in this visualisation

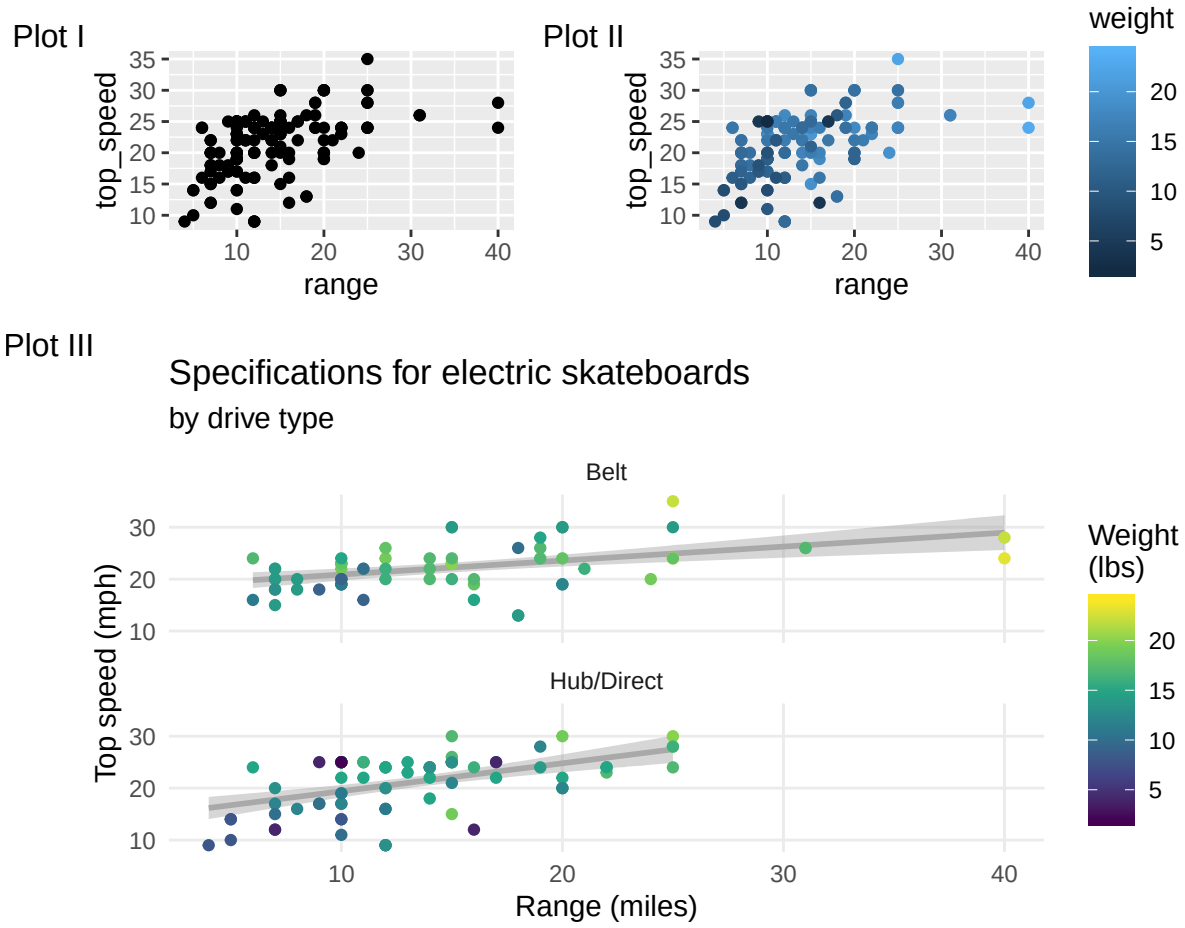


Figure 4: Plot I: Top speed vs. range, 2. Plot II: I + coloring by weight, 3. Plot III: II + faceting and formatting

and how they are displayed in the scatterplot:

- Relationship between the top speed and the range (points on x and y axis)
- Manufacturer (colour of points)
- Weight of electric skateboard (size of point)
- Type of drive (dotted vs. solid line on point)
- Name of electric skateboard (labels for each point)

Figure 4 shows the evolution of visualisations we can make in improving this plot. The most clear information we get from this visualization is the relationship between the top speed and the range. Therefore, we can start by making a plot that just examines the relationship between these two variables (Plot I). Then, we add colour to add information on weights (Plot II), and finally, we can use we can facet the plot based on the drive (belt vs. hub/direct) as well as use proper labels and an informative title (Plot III).

Fisheries of the world

The final example is on fishing harvest of countries around the world. The data come from Wikipedia article titled *Fishing industry by country*. Figure 5 shows the "drab" starting point for this exercise.

The fishery data for used to create this plot may be found at bit.ly/chance-fishery. In this dataset each row represents a country, and for each country we have fishing harvest data for 2016 from capture, aquaculture

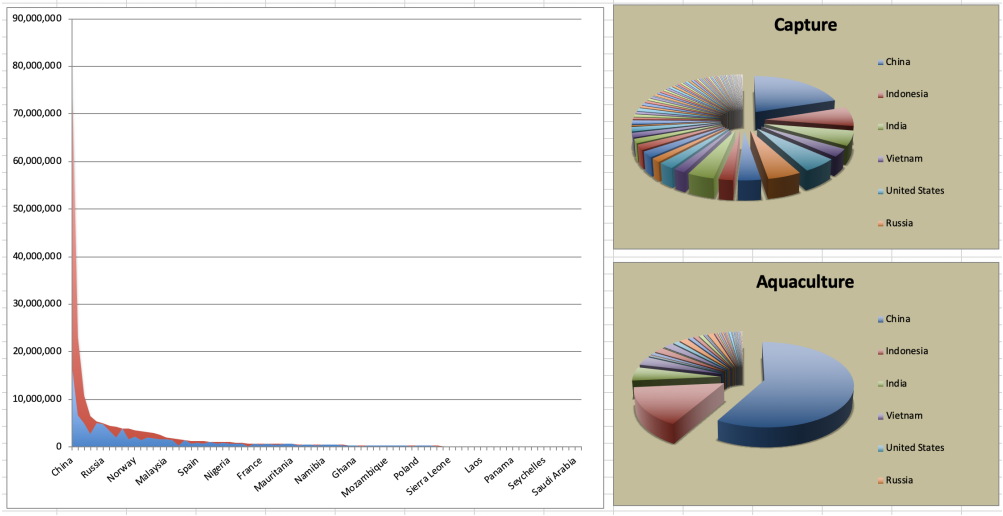
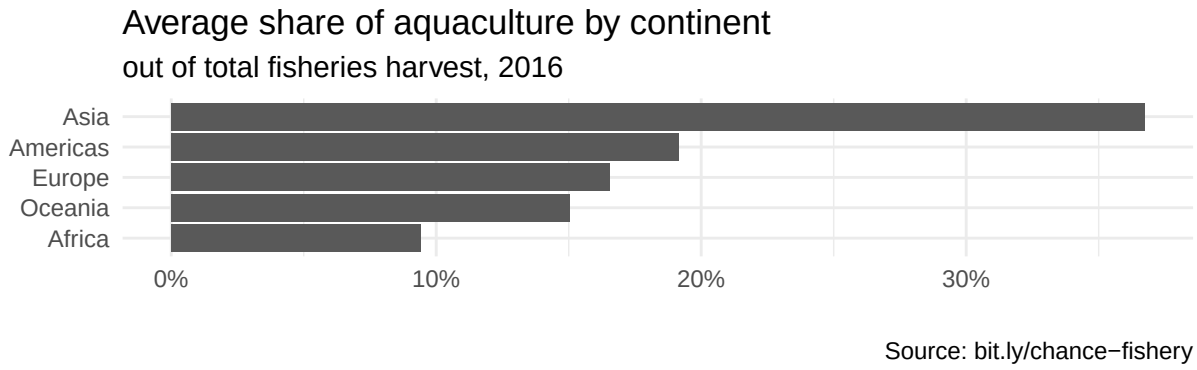


Figure 5: Starting point for the fisheries exercise.

(fish farming), and the total harvest, in metric tonnes. This exercise presents an opportunity for teaching how to join data frames, which is an incredibly useful skill for working with data coming from multiple sources. If wanting to keep the exercise simple, we can join the fishery data with continent information (we have provide the continent data at bit.ly/chance-continent). This also presents an opportunity to talk about how one might need to make contextual decisions while preparing their data for analysis – some of the countries in the fishery dataset (e.g. Hong Kong) do not appear on the continent lookup dataset. The discussion around how to address this is an opportunity to convey to students concerns around ethics in data science and how decisions we make in data wrangling and modelling might sometimes be subjective. Figure **??** presents an example of a simple but improved data visualisation for a continent level analysis of these data.



For a slightly more advanced activity, we map these data. This will require a discussion around basics of making maps in R and shapefiles. Fortunately, there is plenty of built-in functionality in the **gplot2** package for making country level maps. Despite the code required for mapping not being too complicated, this exercise can still ignite plenty of discussion around missing data, specifically how to represent countries for which we don't have any data on a map. Figure **??** presents an example of a choropleth map of the distribution of percentage of aquaculture across the globe.

This activity can also be restructured as a web scraping exercise, where students first scrape the data from Wikipedia, and then dive into visualisation.

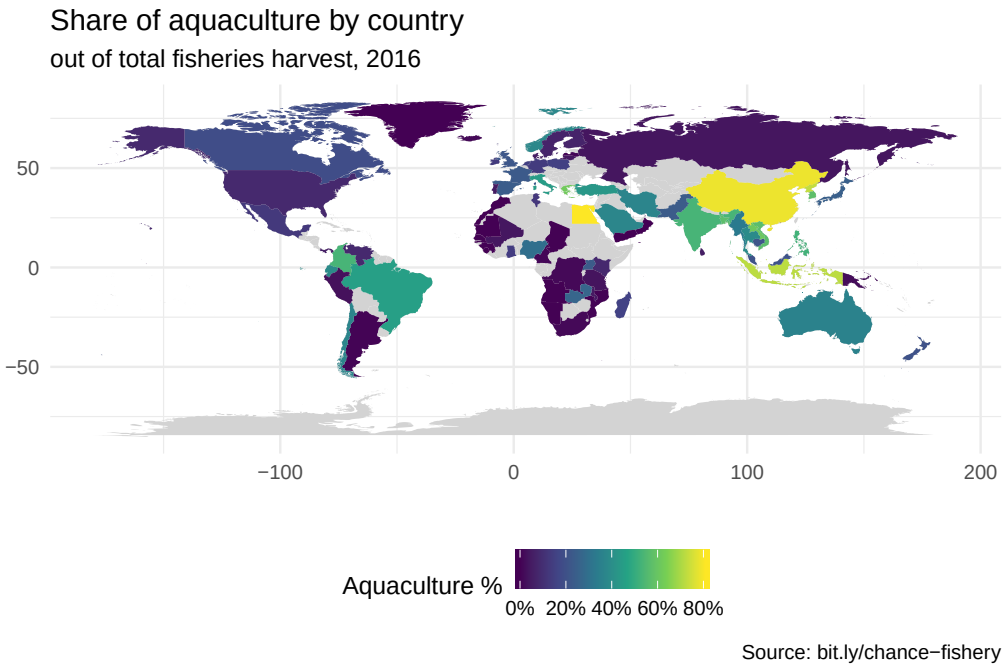


Figure 6: Choropleth map of percentage of aquaculture.

Conclusion

In this column we gave three examples of "from drab to fab" visualisation activities, for each one we highlighted other topics you can introduce along with data visualisation. You can find the data as well as the code for reproducing all of the figures we presented in the GitHub repository for this column as well as links to the additional resources we mentioned at bit.ly/taking-a-chance.

Further reading

1. Wickham et al., (2019). Welcome to the tidyverse. Journal of Open Source Software, 4(43), 1686, https://doi.org/10.21105/joss.01686.
2. "Fishing industry by country." Wikipedia, Wikimedia Foundation, 12 February 2020, en.wikipedia.org/wiki/Fishing_indus