
EuroPython

Python and Data Storytelling to create and deliver better presentations

Sebastián Flores, EuroPyCon 2025

Part 1 - Block 4

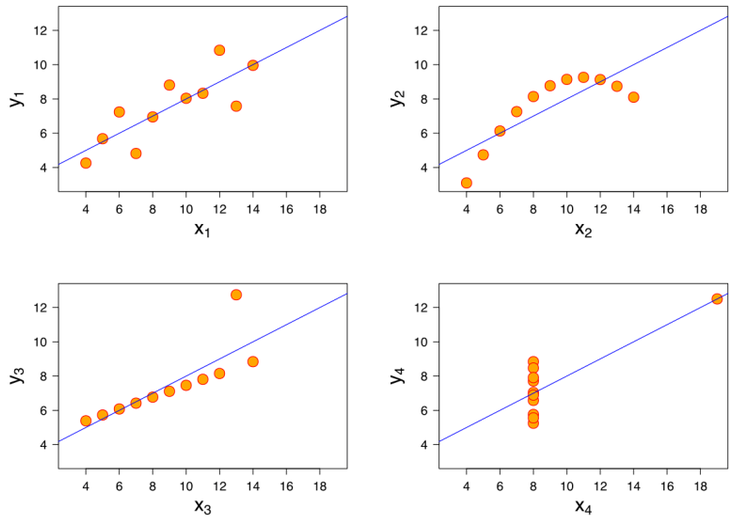
Visualization Quick Wins

Why visualizations?

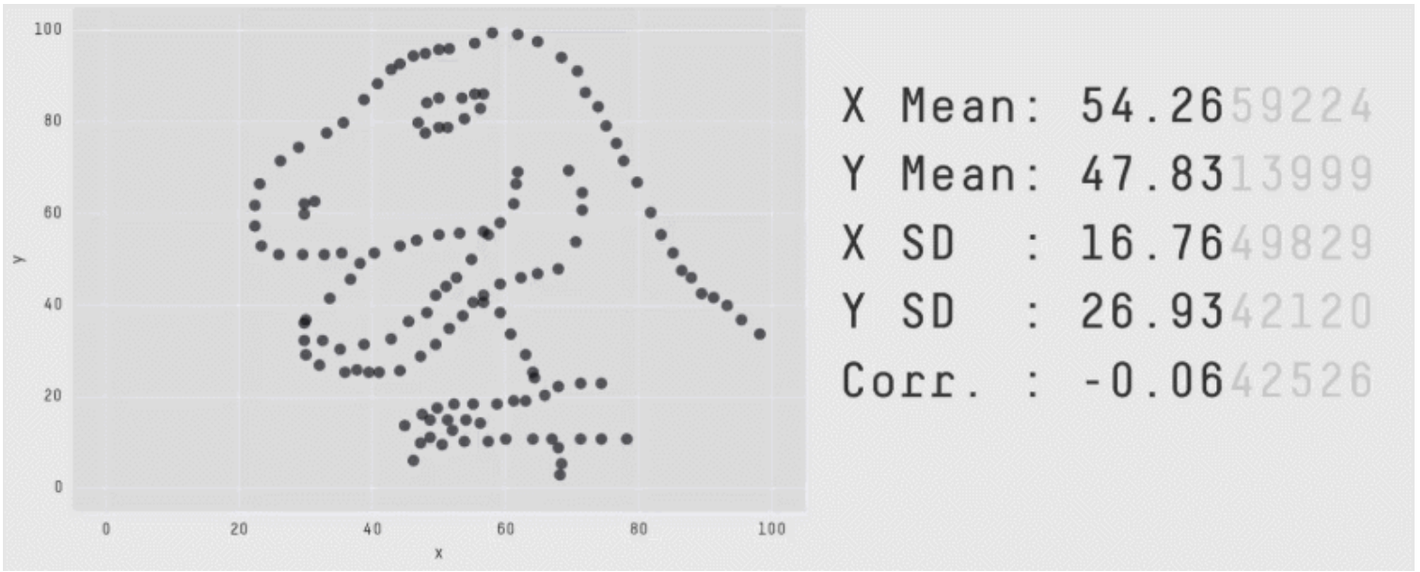
Because of Anscombe’s quartet and the datasaurus.

Why visualizations?

Anscombe's quartet



Datasaurus Dozen

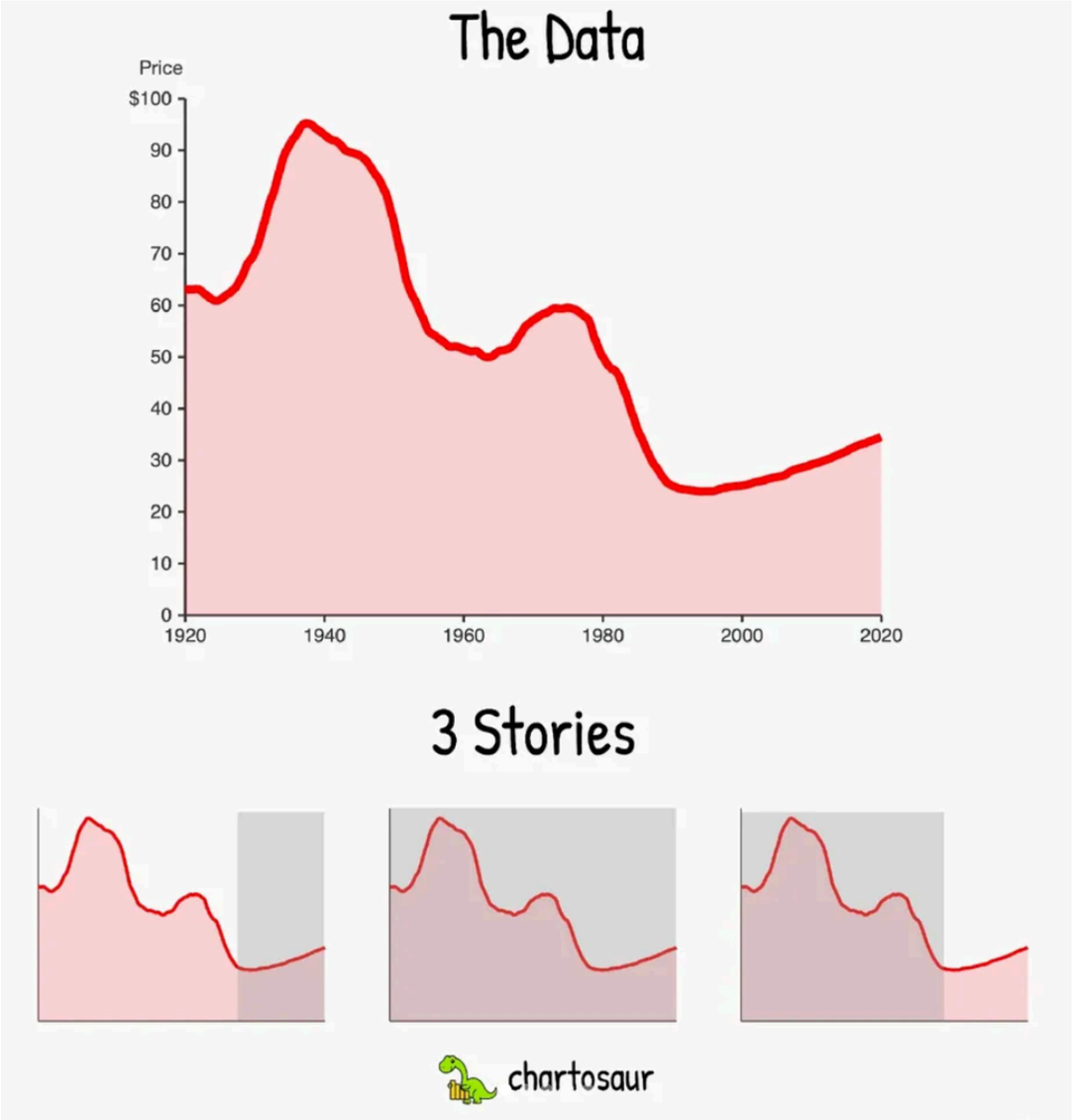


Quick wins

Let’s see some quick wins and code before the coffee break.

Visualization is never "agnostic/neutral"

There is always more than one story within the data:



Select the story within your data

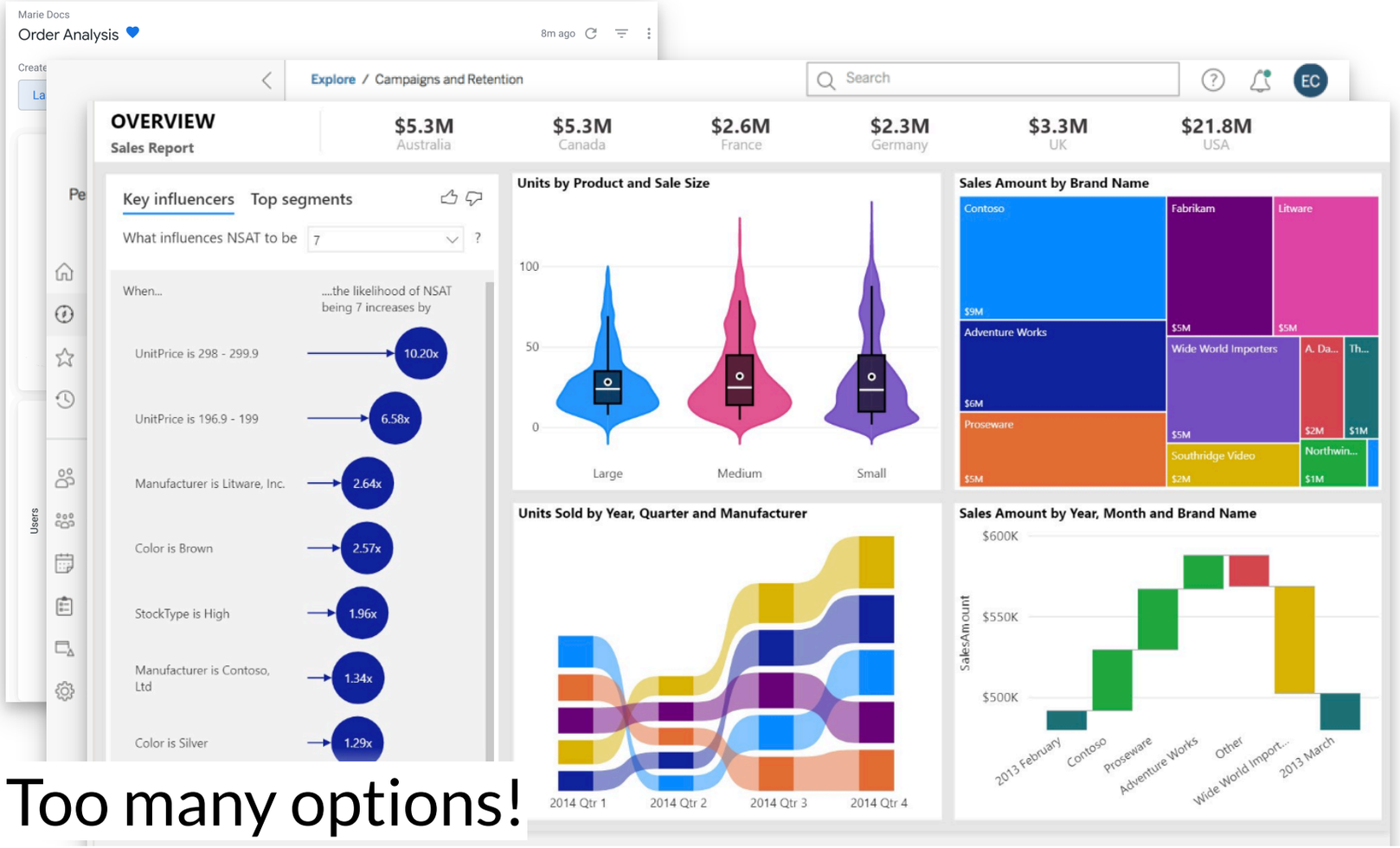
Be fair and truthful.
Be aware you’re not neutral, and that you are showing part of the data in a given way.

Exploratory Viz != Explanatory Viz

Craft what **YOU** want your audience to see.

Avoid exploratory graphs on your presentation.

Data Exploratory Viz



Too many options!

User **cannot** be responsible of finding your insights.

Remember: Never use exploratory visualizations on a presentation!

Simplicity is the ultimate sophistication.

Leonardo da Vinci

My references:

- **Storytelling with data** (book), by Cole Nussbaumer Knaflic
- **Matplotlib Journey** (online course), by Yan Holtz and Joseph Barbier

Simple ideas with high impact

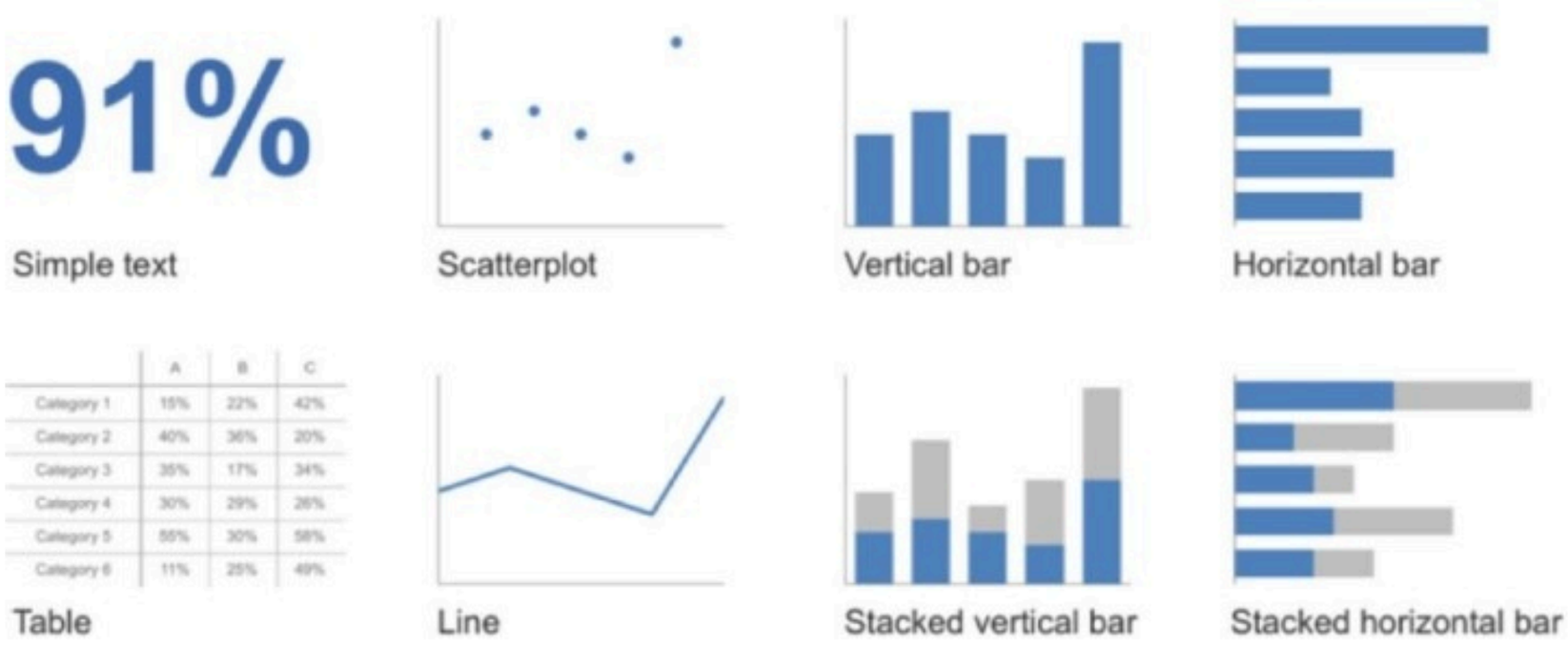
0. Use simpler plots
1. Apply a mis-en-place
2. Annotate your plots
3. Use better colors
4. Develop a progression

0. Use simpler plots

Avoid using complex/custom plots.

Most of the time, you can use just 8 types of plots.

Top 8 plots according to Cole Nussbaumer Knaflic's Storytelling with Data:



- Number: *directly on your slideware*
- Scatterplot: `plt.scatter`
- Vertical bar: `plt.bar`
- Horizontal bar: `plt.barh`
- Table: *directly on your slideware*
- Line: `plt.plot`
- Stacked Vertical bar: *Iterate over* `plt.bar`
- Stacked Horizontal bar: *Iterate over* `plt.barh`

1. Mis-en-place

Prepare your data before plotting

- Create a clean dataset
- If necesary: cast, group and simplify the data before plotting.

2. Mixed bag of tricks

- Learn the difference between fig and ax
- Prefer to use `ax.text` and `fig.text` over `title`, `subtitle`, `suptitle`, `xlabel`, `ylabel`, etc.
- Remove any unnecesary bounding box or line (spine of ax)
- Simplify axes and ticks
- Avoid vertical text

3. Annotate your plots

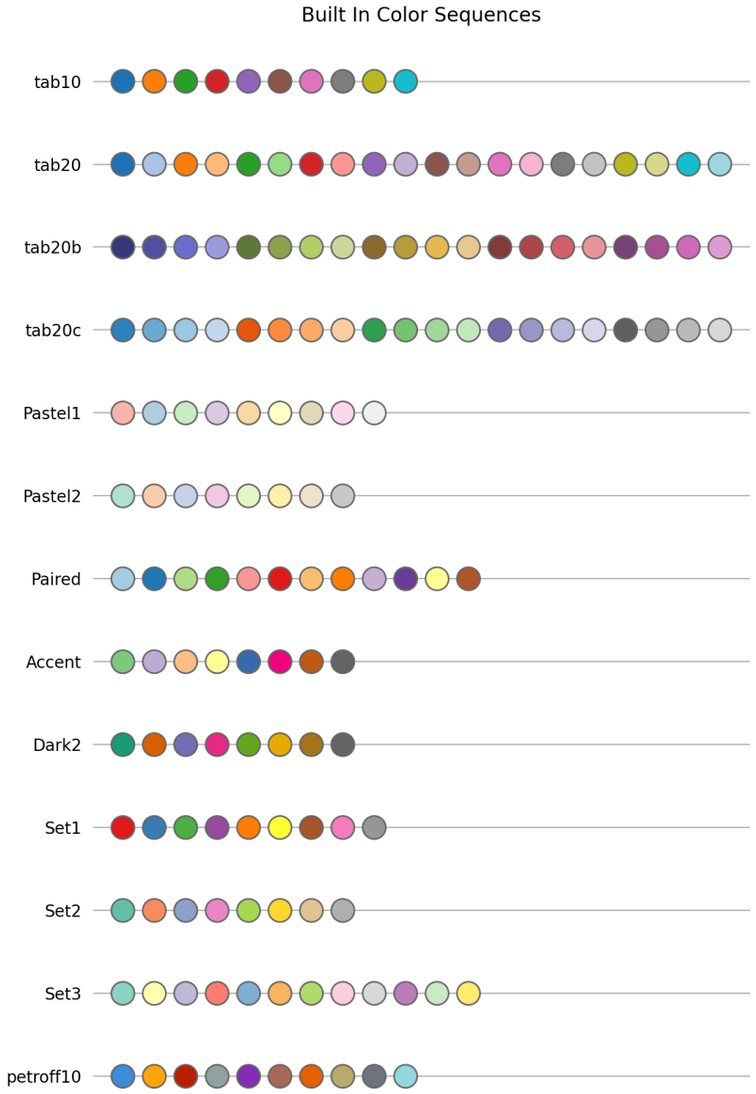
- Remove the “legend” (if possible): annotate contextually.
- Use “highligth_text” to mix font sizes, weigths and colors.

4. Use better colors

- Don’t use regular colors: use a palette
- Options:
  - Matplotlib’s palette: 13 options
  - Pypalette: +2600 curated palettes
- “Inspire” yourself from great examples

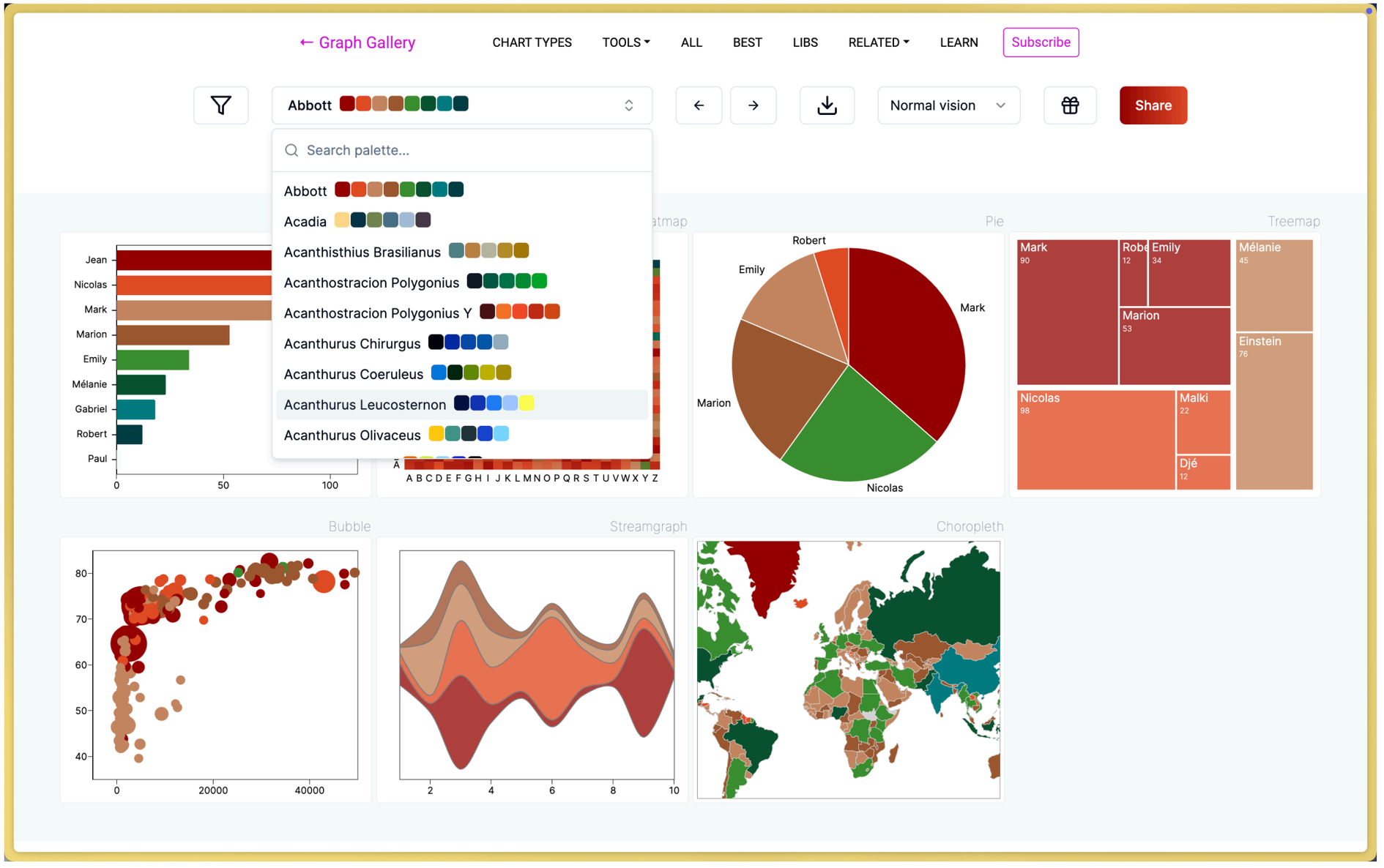
4. Colors: Matplotlib

https://matplotlib.org/stable/gallery/color/color_sequences.html:


Built In Color Sequences
tab10
tab20
tab20b
tab20c
Pastel1
Pastel2
Paired
Accent
Dark2
Set1
Set2
Set3
petroff10

4. Colors: Pypalette

https://python-graph-gallery.com/color-palette-finder/:


← Graph Gallery
CHART TYPES
TOOLS
ALL
BEST
LIBS
RELATED
LEARN
Subscribe
Abbott
Normal vision
Share
Search palette...
Abbott
Acadia
Acanthisthius Brasilianus
Acanthostracion Polygonius
Acanthostracion Polygonius Y
Acanthurus Chirurgus
Acanthurus Coeruleus
Acanthurus Leucosternon
Acanthurus Olivaceus
Jean
Nicolas
Mark
Marion
Emily
Mélanie
Gabriel
Robert
Paul
0
50
100
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
Pie
Robert
Emily
Mark
Marion
Nicolas
Treemap
Mark
90
Robe
12
Emily
34
Marion
53
Mélanie
45
Einstein
76
Nicolas
98
Malki
22
Djé
12
Bubble
80
70
60
50
40
0
20000
40000
Streamgraph
2
4
6
8
10
Choropleth

5. Develop a progression

- Don’t reveal the final plot inmediately
- Use a progression to train the audience into how to read
- Revel the plot step by step and explain each step
- Keep the control of the attention at all steps

Let’s code!