DEGGENDORF INSTITUTE OF TECHNOLOGY

Data Visualization

01

Prof. Dr. Phillipp Torkler

Data Visualization Provides Data Exploration and Communication

Data Visualization serves **two** major purposes:

1. **Data Exploration:**
   - Familiarize with a data set
   - Look for patterns in data
   - Patterns or regularities in the data set are expected but not known in advance
   - Data exploration can guide the scientific process
   - New scientific ideas can emerge from visualizations from (laboratory) data. Thus, data visualization is often more than ‘just showing data’.

2. **Communication / Presentation:**
   - Interesting findings have been made and need to get communicated clearly to readers
   - Focus to tell a clear message
   - A reader does not need to get through the same process as a researcher that tries to find patterns during data exploration. In contrast, key findings should be communicated without overwhelming a reader with unnecessary data

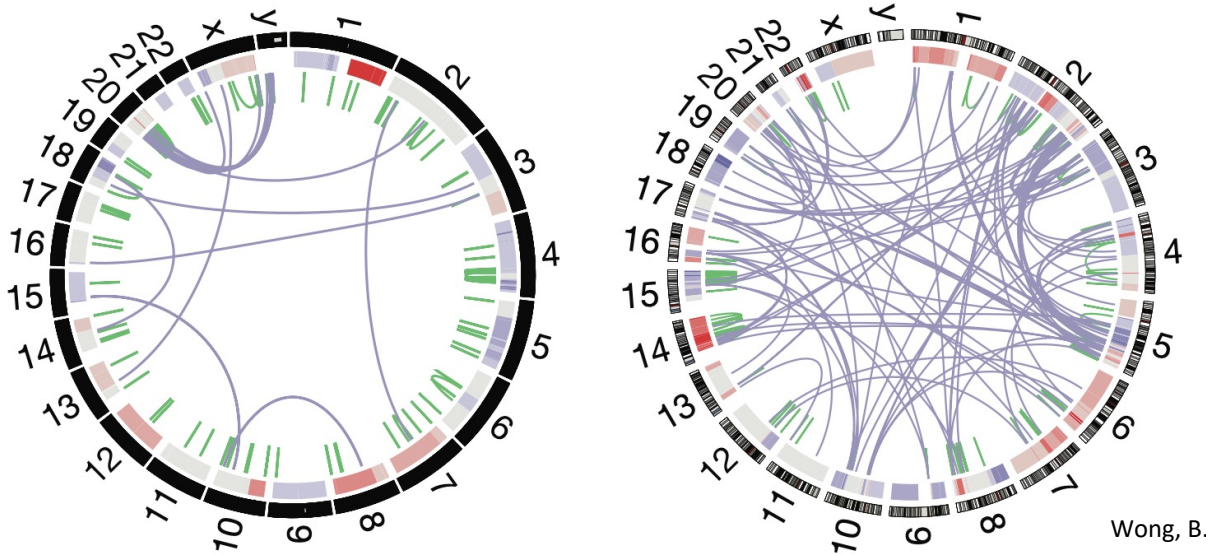
In either case, the purpose of any figure is to transport a message!

Why Do We Need Exploration and Communication?

Today, the challenge for researchers is to take benefit from large data sets without getting drawn in too much data. Thus, both exploration and communication is key for finding and sharing new insights.

Goals of data visualization:

- enable researches to explore and explain their data through (interactive) visualizations
- take advantage of the human's ability to recognize patterns
- data types and research questions evolve rapidly in the scientific community. Likewise, data visualizations need to adapt to new techniques to provide insights.
- Visualization as a complement for algorithmic approaches to provide a mental image of what happens (see example)



Wong, B. "Visualzing biological data" *Nature Methods* (2012)

Technical and Design Aspects of Data Visualization

A figure can be **any** visual representation: graphs, photos, drawings, schematics, cartoons, maps etc.. Despite their variety, the purpose of any figure is to support a message. **Good figures will show the data AND the message** you want to tell!

Figure creation consists of two major building blocks:

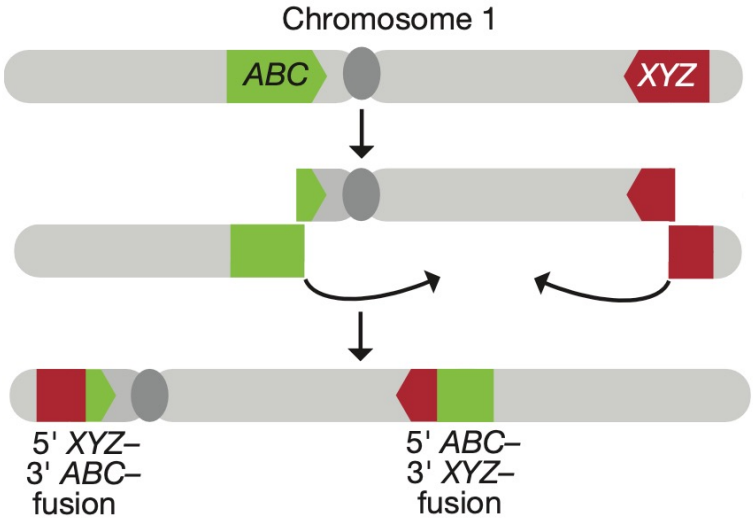
1. The ’technical’ generation of a figure (the doing):
   - e.g. use of a programming language and corresponding graphic library
   - choice and usage of graphics software
   - photography etc…

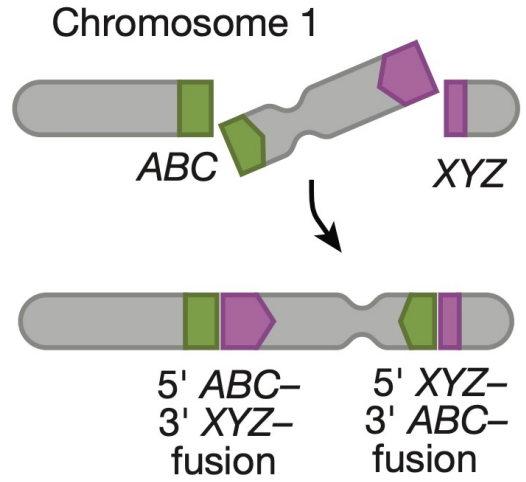
2. The ‘design’ of a figure (the theory and idea):
   - use graphic design principles
   - biology of the human visual system
   - psychology

In this course we want to train ’the doing’ and introduce a couple of design principles to generate meaningful and clear figures.

Examples For Clearly Structured Figures (1)

Visualizing chromosomal inversion that results in two fusion genes:

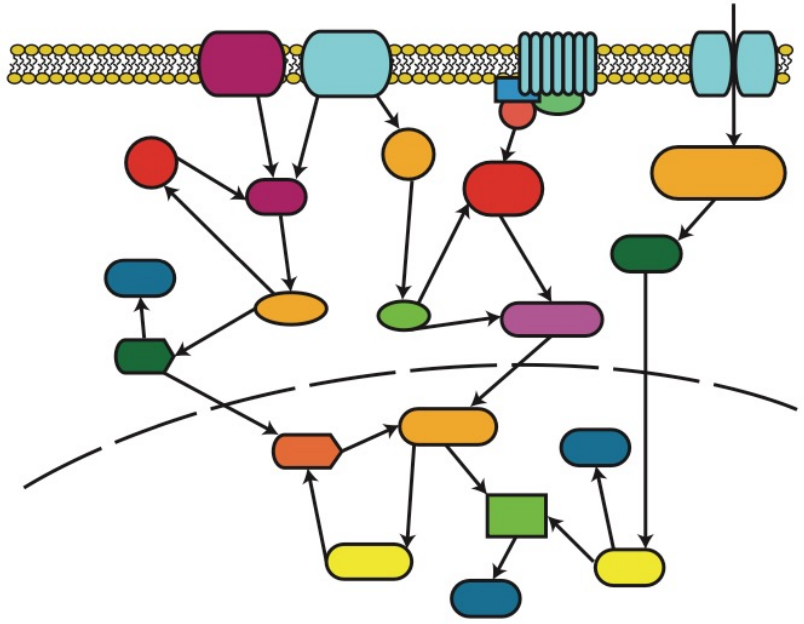


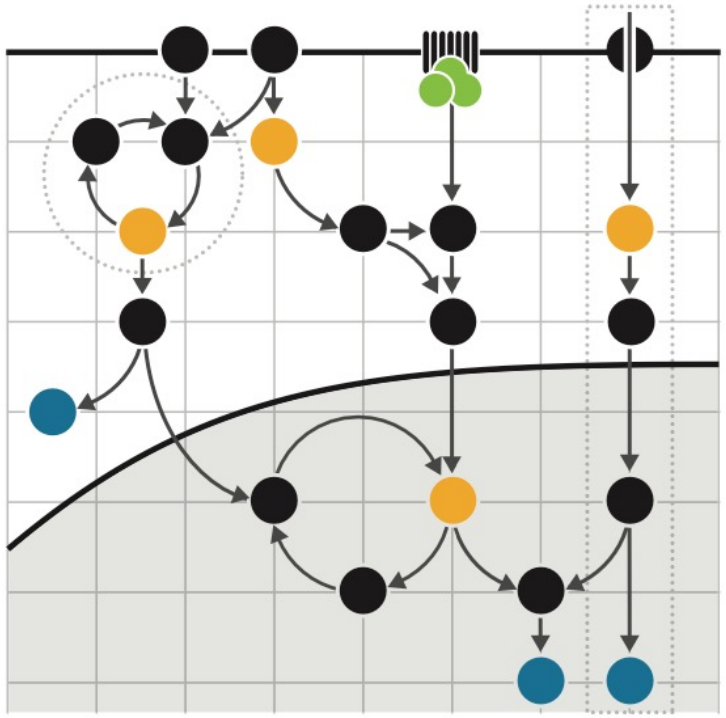


Wong, B. “Simplify to clarify” *Nature Methods* (2011)

Examples For Clearly Structured Figures (2)

Examples for pathway diagrams





- redundant visual encodings removed and main points emphasized by visual grouping
- Color and shape variations have been removed except for those highlighting a molecule of interest (orange), the products of the pathway (blue)

Wong, B. “Pathways” *Nature Methods* (2016)

Reading Exercise

- What is salience?
- Why do we need to consider the concept of salience when designing (scientific) visualizations?

Notes

TECHNISCHE HOCHSCHULE DEGGENDORF THD

Figure Generation

Vector vs. Raster Graphics

https://inkscape.org/~ozant/★art-bot

Vector vs. Raster Graphics

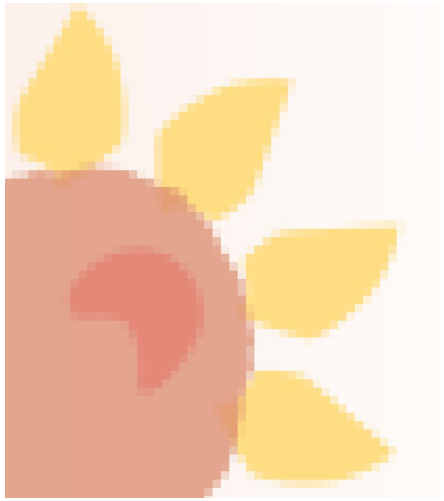
https://inkscape.org/~ozant/★art-bot

Raster Graphic

Vector Graphic

Vector vs. Raster Graphics

https://inkscape.org/~ozant/★art-bot



A raster graphic of width n and height m is described by an mn array where each position i, j stores the color information of the corresponding pixel of the graphic.

In contrast, vector graphics are described by graphical primitives like lines (described by two points) and circles (described by center and radius). Primitives can be described by mathematical functions, and they can be combined to build complex objects.

Objects in a vector graphic are fully described by their primitives and as a consequence of that vector graphics can be scaled without loss in quality.

Take home message: schematic drawings or data visualizations are ideally generated in a vector format.

Tools For Creating and Editing Vector Graphics

TECHNISCHE HOCHSCHULE DEGGENDORF THD

There are many tools available, but two prominent examples are Adobe Illustrator (if you have money and want to pay) and Inkscape (free and open-source). Both programs can be used to create or edit vector graphics.

https://inkscape.org

https://www.adobe.com/products/illustrator.html

Knowledge in these tools can be quite helpful even as a computer scientist. Composing bigger figures from multiple single plots, quickly adjust or align colors, add explanations or explanatory drawings can help to create better figures. Depending on the scenario, compositions can be done more easily in these programs rather than doing these edits via programming.

Figure Generation via Programming

Of course, as computer scientists we want to generate figures from data automatically via programming languages. In the first part of the course, you have already been introduced into *base R* and *ggplot2.* For the sake of comparison, let's look at a few others:

Base R [https://cran.r-project.org]

matplotlib (python) [https://matplotlib.org]

D3 (JavaScript) [https://d3js.org]

ggplot2 (R) [https://ggplot2.tidyverse.org]

seaborn (Python) [https://seaborn.pydata.org/examples/index.html]

plotly (R, Python, JavaScript) [https://plotly.com]

bokeh (Python) [https://docs.bokeh.org/en/latest/index.html]

and so on…. there are so many…

If There Are So Many, Which Should I Learn?

TECHNISCHE HOCHSCHULE DEGGENDORF THD

Typically, it makes sense to focus on a plotting library that belongs to the language that is used in the project. Since most data science is performed via R or Python it is beneficial to have experience in both. For R good starting points are base R and ggplot2 for Python matplotlib is very popular.

Comparing the left and right list, what do you think is the difference between them?

Base R [https://cran.r-project.org]

matplotlib (python) [https://matplotlib.org]

D3 (JavaScript) [https://d3js.org]

ggplot2 (R) [https://ggplot2.tidyverse.org]

seaborn (Python) [https://seaborn.pydata.org/examples/index.html]

plotly (R, Python, JavaScript) [https://plotly.com]

bokeh (Python) [https://docs.bokeh.org/en/latest/index.html]

If There Are So Many, Which Should I Learn?

TECHNISCHE HOCHSCHULE DEGGENDORF THD

Typically, it makes sense to focus on a plotting library that belongs to the language that is used in the project. Since most data science is performed via R or Python it is beneficial to have experience in both. For R good starting points are base R and ggplot2 for Python matplotlib is very popular.

Comparing the left and right list, what do you think is the difference between them?

Base R [https://cran.r-project.org]

matplotlib (python) [https://matplotlib.org]

D3 (JavaScript) [https://d3js.org]

ggplot2 (R) [https://ggplot2.tidyverse.org]

seaborn (Python) [https://seaborn.pydata.org/examples/index.html]

plotly (R, Python, JavaScript) [https://plotly.com]

bokeh (Python) [https://docs.bokeh.org/en/latest/index.html]

It’s hard to draw a clear line, but some libraries are more low-level (e.g. base R, D3, matplotlib), whereas others are more high-level (e.g. seaborn, ggplot2). It’s a trade-off. The more high-level a library is, the more you need to stick to the decisions and the offered functionality. Extending or changing is often cumbersome. Low-level offers more flexibility at the cost of spending more time to create visual pleasing plots.

Technical Requirements

Conda, Python, Matplotlib

TECHNISCHE HOCHSCHULE DEGGENDORF THD

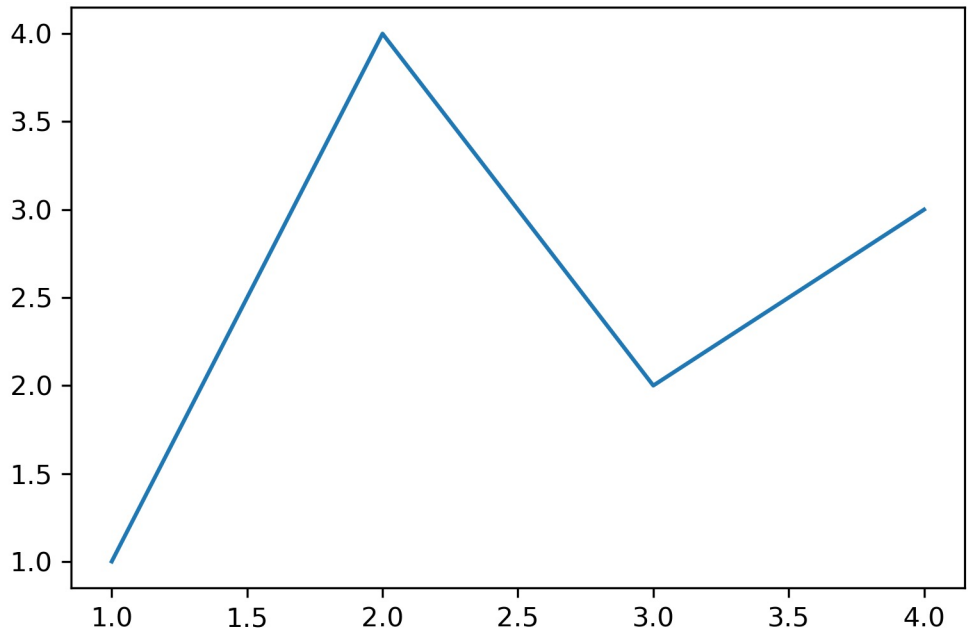
In this course we will use Python to generate data visualizations. The LSI students should have a running system since this setup is used in other courses as well.

Follow the instructions on the DataVis_Setup.pdf to install the required software.

Matplotlib

```python
import matplotlib.pyplot as plt
import numpy as np

fig, ax = plt.subplots()  # Create a figure containing a single axes.
ax.plot([1, 2, 3, 4], [1, 4, 2, 3])  # Plot some data on the axes.
```



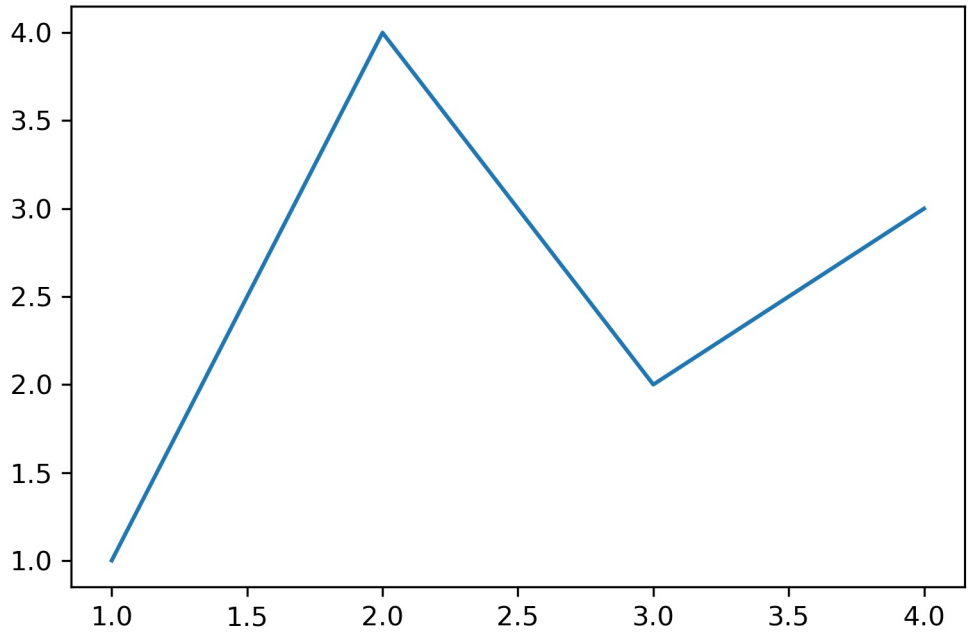
matplotlib user guide:

https://matplotlib.org/stable/tutorials/introductory/usage.html

Matplotlib

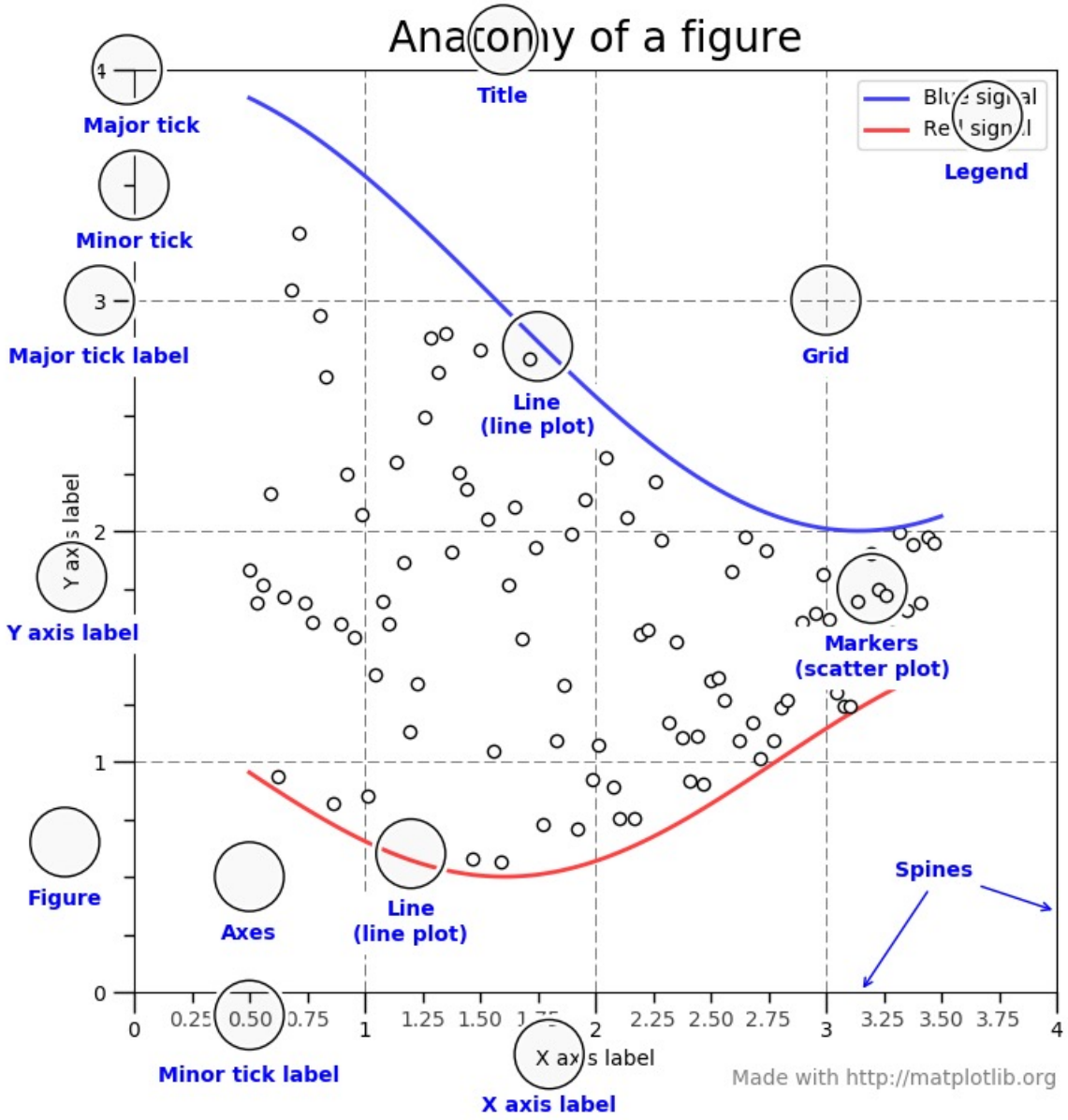
```python
import matplotlib.pyplot as plt
import numpy as np

fig, ax = plt.subplots()  # Create a figure containing a single axes.
ax.plot([1, 2, 3, 4], [1, 4, 2, 3])  # Plot some data on the axes.
```



matplotlib user guide:

https://matplotlib.org/stable/tutorials/introductory/usage.html



Programming Exercise

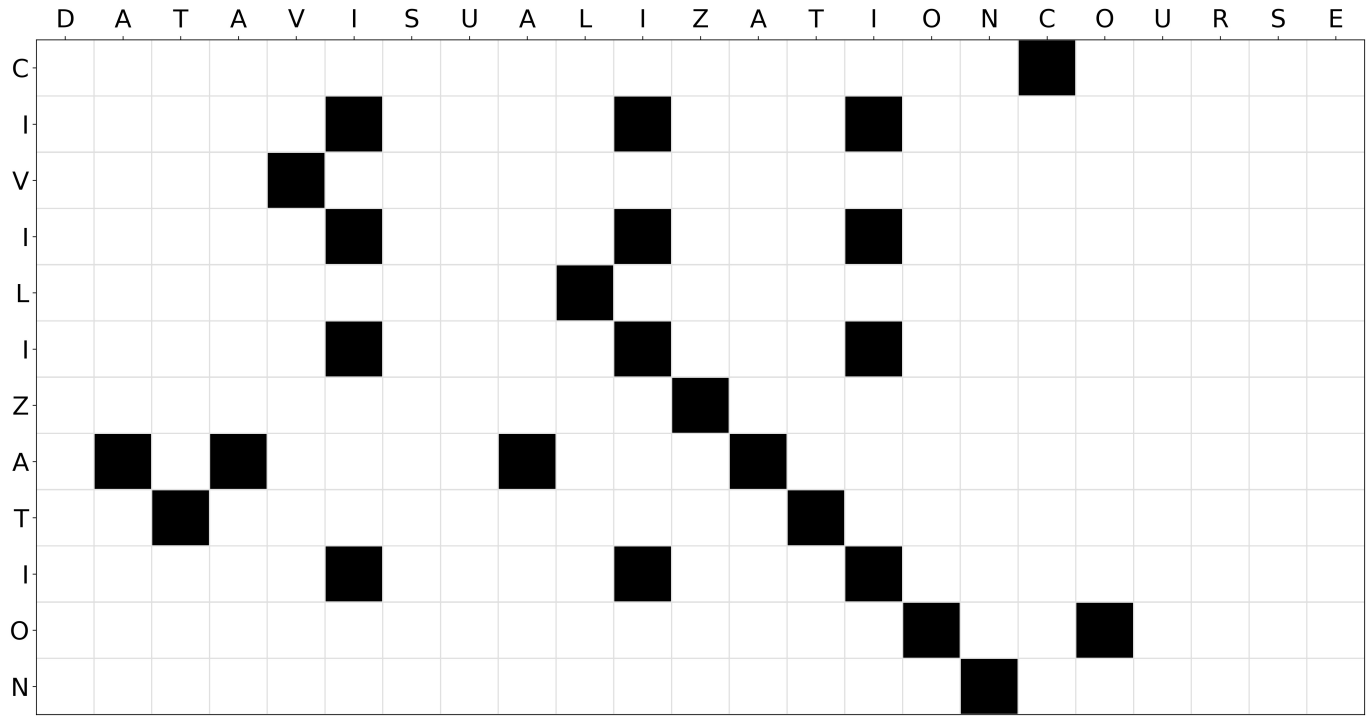
- Create A Dot Plot From Scratch Using Python/matplotlib

A Dot Plot Visualizes Sequence Similarity

A simple and visual way of comparing two sequences s_1 and s_2 of length n and m with each other is a dot plot. The sequences s_1 and s_2 span a $n \times m$ matrix M where a dot is plotted for $M_{i,j}$ if $s_1[i] = s_2[j]$.

$$M_{i,j} = \begin{cases} 1, \text{if } s_1[i] = s_2[j] \\ 0, \text{ if } s_1[i] \neq s_2[j] \end{cases}$$

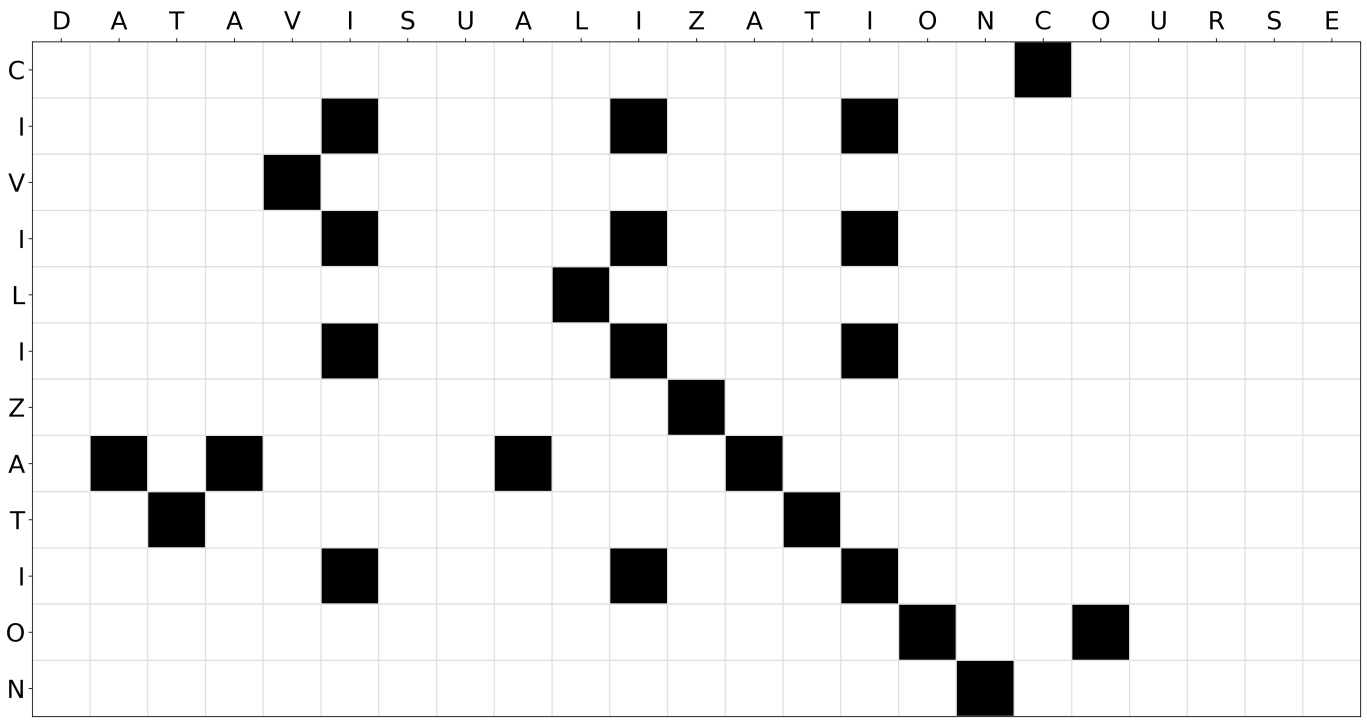
The figure below shows a dot plot for $s_1 = CIVILIZATION$ and $s_2 = DATAVISUALIZATIONCOURSE$.

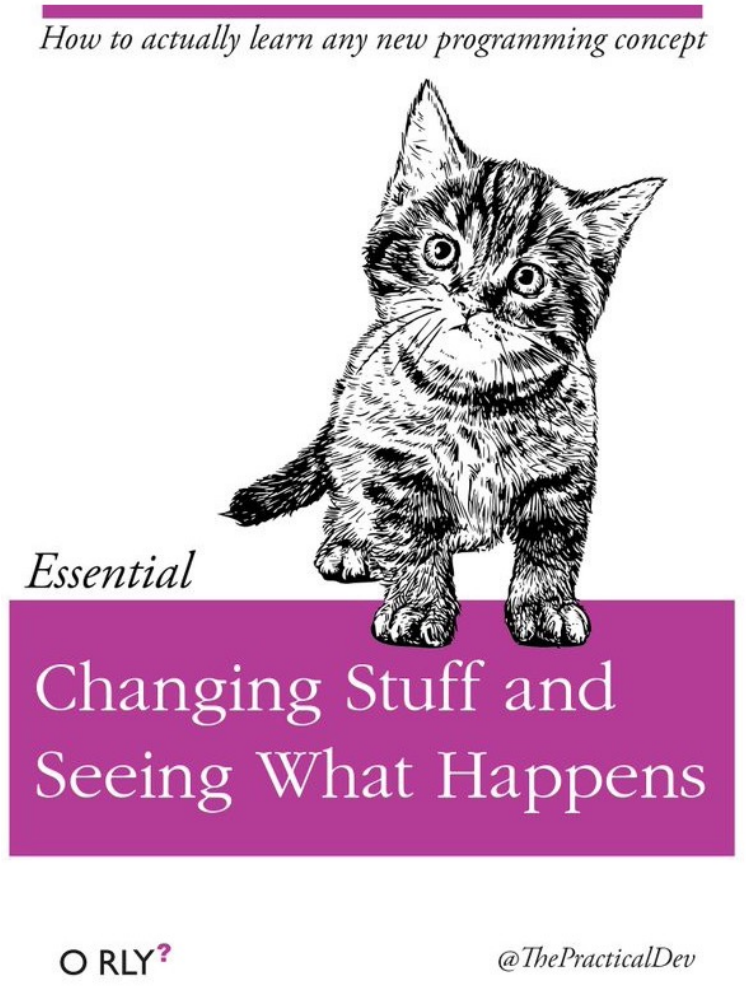


A Dot Plot Visualizes Sequence Similarity

Matplotlib user guide:

https://matplotlib.org/stable/tutorials/introductory/usage.html





https://twitter.com/ThePracticalDev/status/720257210161311744