Programming Principles in Python (CSCI 503/490)

Visualization

Dr. David Koop
Exploring Data through Visualization
Exploring Data through Visualization
Why do we visualize data?

Why Graphics?

Figures are richer; provide more information with less clutter and in less space.

Figures provide the gestalt effect: they give an overview; make structure more visible.

Figures are more accessible, easier to understand, faster to grasp, more comprehensible, more memorable, more fun, and less formal.

List adapted from: [Stasko et al. 1998]
# Why Visual?

<table>
<thead>
<tr>
<th></th>
<th>I</th>
<th></th>
<th>II</th>
<th></th>
<th>III</th>
<th></th>
<th>IV</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>x</td>
<td>y</td>
<td></td>
<td>x</td>
<td>y</td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>---</td>
<td>------</td>
<td>---</td>
<td>-------</td>
<td>---</td>
<td>-------</td>
<td>---</td>
<td>-------</td>
<td>---</td>
</tr>
<tr>
<td>10.0</td>
<td>8.04</td>
<td>10.0</td>
<td>9.14</td>
<td>10.0</td>
<td>7.46</td>
<td>8.0</td>
<td>6.58</td>
<td></td>
</tr>
<tr>
<td>8.0</td>
<td>6.95</td>
<td>8.0</td>
<td>8.14</td>
<td>8.0</td>
<td>6.77</td>
<td>8.0</td>
<td>5.76</td>
<td></td>
</tr>
<tr>
<td>13.0</td>
<td>7.58</td>
<td>13.0</td>
<td>8.74</td>
<td>13.0</td>
<td>12.74</td>
<td>8.0</td>
<td>7.71</td>
<td></td>
</tr>
<tr>
<td>9.0</td>
<td>8.81</td>
<td>9.0</td>
<td>8.77</td>
<td>9.0</td>
<td>7.11</td>
<td>8.0</td>
<td>8.84</td>
<td></td>
</tr>
<tr>
<td>11.0</td>
<td>8.33</td>
<td>11.0</td>
<td>9.26</td>
<td>11.0</td>
<td>7.81</td>
<td>8.0</td>
<td>8.47</td>
<td></td>
</tr>
<tr>
<td>14.0</td>
<td>9.96</td>
<td>14.0</td>
<td>8.10</td>
<td>14.0</td>
<td>8.84</td>
<td>8.0</td>
<td>7.04</td>
<td></td>
</tr>
<tr>
<td>6.0</td>
<td>7.24</td>
<td>6.0</td>
<td>6.13</td>
<td>6.0</td>
<td>6.08</td>
<td>8.0</td>
<td>5.25</td>
<td></td>
</tr>
<tr>
<td>4.0</td>
<td>4.26</td>
<td>4.0</td>
<td>3.10</td>
<td>4.0</td>
<td>5.39</td>
<td>19.0</td>
<td>12.50</td>
<td></td>
</tr>
<tr>
<td>12.0</td>
<td>10.84</td>
<td>12.0</td>
<td>9.13</td>
<td>12.0</td>
<td>8.15</td>
<td>8.0</td>
<td>5.56</td>
<td></td>
</tr>
<tr>
<td>7.0</td>
<td>4.82</td>
<td>7.0</td>
<td>7.26</td>
<td>7.0</td>
<td>6.42</td>
<td>8.0</td>
<td>7.91</td>
<td></td>
</tr>
<tr>
<td>5.0</td>
<td>5.68</td>
<td>5.0</td>
<td>4.74</td>
<td>5.0</td>
<td>5.73</td>
<td>8.0</td>
<td>6.89</td>
<td></td>
</tr>
</tbody>
</table>

[F. J. Anscombe]
### Why Visual?

<table>
<thead>
<tr>
<th></th>
<th>I</th>
<th>II</th>
<th>III</th>
<th>IV</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>y</td>
<td>x</td>
<td>y</td>
<td>x</td>
</tr>
<tr>
<td>10.0</td>
<td>8.04</td>
<td>10.0</td>
<td>9.14</td>
<td>10.0</td>
</tr>
<tr>
<td>8.0</td>
<td>6.95</td>
<td>8.0</td>
<td>8.14</td>
<td>8.0</td>
</tr>
<tr>
<td>13.0</td>
<td>7.58</td>
<td>13.0</td>
<td>8.74</td>
<td>13.0</td>
</tr>
<tr>
<td>9.0</td>
<td>8.81</td>
<td>9.0</td>
<td>8.77</td>
<td>9.0</td>
</tr>
<tr>
<td>11.0</td>
<td>8.33</td>
<td>11.0</td>
<td>9.26</td>
<td>11.0</td>
</tr>
<tr>
<td>14.0</td>
<td>9.96</td>
<td>14.0</td>
<td>8.10</td>
<td>14.0</td>
</tr>
<tr>
<td>6.0</td>
<td>7.24</td>
<td>6.0</td>
<td>6.13</td>
<td>6.0</td>
</tr>
<tr>
<td>4.0</td>
<td>4.26</td>
<td>4.0</td>
<td>3.10</td>
<td>4.0</td>
</tr>
<tr>
<td>12.0</td>
<td>10.84</td>
<td>12.0</td>
<td>9.13</td>
<td>12.0</td>
</tr>
<tr>
<td>7.0</td>
<td>4.82</td>
<td>7.0</td>
<td>7.26</td>
<td>7.0</td>
</tr>
<tr>
<td>5.0</td>
<td>5.68</td>
<td>5.0</td>
<td>4.74</td>
<td>5.0</td>
</tr>
</tbody>
</table>

- **Mean of x**: 9
- **Variance of x**: 11
- **Mean of y**: 7.50
- **Variance of y**: 4.122
- **Correlation**: 0.816

[F. J. Anscombe]
Why Visual?

Why Visual?

[F. J. Anscombe]
Why Visual?

<table>
<thead>
<tr>
<th></th>
<th>Mean of x</th>
<th>Variance of x</th>
<th>Mean of y</th>
<th>Variance of y</th>
<th>Correlation</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1</td>
<td>9</td>
<td>11</td>
<td>7.50</td>
<td>4.122</td>
<td>0.816</td>
</tr>
<tr>
<td>x2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

[F. J. Anscombe]
Visualization Goals

• "The purpose of visualization is **insight**, not pictures" – B. Schneiderman

• Identify patterns, trends
• Spot outliers
• Find similarities, correlation
Data is Encoded via Visual Channels

- **Position**
  - Horizontal
  - Vertical
  - Both

- **Color**

- **Shape**

- **Tilt**

- **Size**
  - Length
  - Area
  - Volume

[Munzner (ill. Maguire), 2014]
matplotlib

• Strengths:
  - Designed like Matlab
  - Many rendering backends
  - Can reproduce almost any plot
  - Proven, well-tested

• Weaknesses:
  - API is imperative
  - Not originally designed for the web
  - Dated styles
pandas Integration

• Can call many of these methods directly from pandas
• Handled through kind kwarg or .plot accessor
• It will try to guess a reasonable visualization, but may fail:
  - `fruit.plot()`
• Instead, specify x and y and other parameters:
  - `fruit.plot(kind='bar', x='name', y='price')`
  - `plt.bar(x='name', height='price', data=fruit) # SIMILAR`
  - `fruit.plot.scatter(x='price', y='count', c='name') # ERROR`
  - `colors = {'Apple': 'red', 'Orange': 'orange',
              'Banana': 'yellow', 'Pear': 'green'}
    fruit.plot.scatter(x='price', y='count',
                       c=fruit['name'].map(colors))`
Assignment 8

• Due Monday
• Data and Visualization
• Pokemon Data
Final Exam

- Wednesday, May 10, **10:00-11:50am** in PM 110
- **More comprehensive** than Test 2
- Expect questions from topics covered on Test 1 and 2
- Expect questions from the last four weeks of class (data, visualization, machine learning)
- Similar format
The Python Visualization Landscape
The Python Visualization Landscape

JavaScript
- cufflinks
- toyplot
- bokeh
- datashader
- holoviews
- datashader

OpenGL
- vispy
- glumpy
- visvis
- galry

PyQtGraph
- GR Framework
- mayavi
- pyglet
- ipyvolume
- ipyleaflet
- pythreejs
- bqplot
- mpld3
- d3po
- d3js
- altair
- vega-lite
- vega
- pandas
- seaborn
- matplotlib
- json
- d3js
- OpenGL

Mayavi
- pyglet

Galry
- mayavi

Ipyvolume
- ipyleaflet

Pydeck
- ipyvolume

Networking
- networkx
- basemap
- cartopy
- graph-tool
- graphviz
- yellowbrick
- scikit-plot
- plotnine

Visualization Libraries
- Holoviews
- Datashader
- Bokeh
- Bqplot
- Mpld3
- D3js
- Altair
- Vega-

Plotting Libraries
- Matplotlib
- Plotly
- Ipyvolume
- Cuflinks
- Toyplot

GLFW
- GLFW
- Inks

Datashader
- Datashader
- D3po
- D3js

Lighting
- Lighting
- Ipyvolume
- Ipyleaflet
- Pythreejs

GLFW
- GLFW
- Inks

Matplotlib
- Matplotlib
- Javascript

OpenGL
- OpenGL
- D3js

Visualization Tools
- Glumpy
- Vispy
- Glueviz
- YT

Ipython
- Ipython
- Pydeck

Pygraphviz
- Pygraphviz
- Pyqtgraph

Ipython
- Ipython
- Pydeck

Scientific Computing
- Scikit-plot
- Yellowbrick
- Graph-tool
- Graphviz

Web Technologies
- Ipyvolume
- Ipyleaflet
- Pythreejs
History of Vega-Lite & Altair

- "Grammar of Graphics", L. Wilkinson
- "A Layered Grammar of Graphics", H. Wickham
- ggplot: plotting library for R
- Vega: similar idea for Javascript/JSON (U. Washington, A. Satyanarayan)
  - "Declarative language for creating, saving, and sharing interactive visualization designs"
  - More focus on interaction and reactive signals
  - Separation between specification and runtime
- Vega-Lite: higher-level language than Vega (U. Washington, D. Moritz)
  - uses carefully designed rules to default settings
History of Vega-Lite & Altair

- Altair: Python interface to Vega-Lite (J. VanderPlas)
  - "spend more time understanding your data and its meaning"
  - Specify the what, minimize the amount of code directing the how
  - Python can write JSON specification just as well as any other language
  - Bindings make it more Python-friendly, integrate with pandas, add support for Jupyter, etc.

- Vega Fusion (J. Mease)
  - Scaling to larger datasets
  - Serverside scaling
Basic Example

- import altair as alt
  import pandas as pd
  data = pd.DataFrame({'x': [1,3,4,6,10], 'y': [1,5,2,7,3]})
  alt.Chart(data).mark_line().encode(x='x', y='y')

- Easiest to use data from a pandas data frame
  - Another option is a csv or json file
  - Can support geo_interface, too

- Chart is the basic unit
- Mark: .mark_*() indicates the geometry created for each data item
- Encode: .encode() allows visual properties to be set to data attributes
Visual Marks

- **Marks** are the basic graphical elements in a visualization
- Marks classified by dimensionality:
  - Points
  - Lines
  - Areas
- Also can have surfaces, volumes
- Think of marks as a mathematical definition, or if familiar with tools like Adobe Illustrator or Inkscape, the path & point definitions
- Altair: area, bar, circle, geoshape, image, line, point, rect, rule, square, text, tick
  - Also compound marks: boxplot, errorband, errorbar
Encode via Visual Channels

- **Position**
  - Horizontal
  - Vertical
  - Both

- **Color**

- **Shape**

- **Tilt**

- **Size**
  - Length
  - Area
  - Volume

[Munzner (ill. Maguire), 2014]
Easily Explore Different Encodings

<table>
<thead>
<tr>
<th>'age'</th>
<th>'weight'</th>
<th>'zoo_area'</th>
<th>'num_scoops'</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>20</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>50</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>25</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>6</td>
<td>55</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>10</td>
<td>125</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>

```python
data = pd.DataFrame({
    'age': [1,3,4,6,10],
    'weight': [20,50,25,55,125],
    'zoo_area': [1,3,3,1,2],
    'num_scoops': [3,2,4,2,3]
})
alt.Chart(data).mark_point(filled=True, size=50,
    stroke='black',strokeWidth=1
).encode(
    x='age',
    y='weight',
    color='zoo_area'
)```
Problem: zoo_area is not a continuous value, nor is it ordered in any way!
Data Attributes and Altair Types

- Categorical
- Ordered
- Ordinal
- Quantitative

[Munzner (ill. Maguire), 2014]
Data Attributes and Altair Types

- **Categorical data** = Nominal (N)
- **Ordinal data** = Ordinal (O)
- **Quantitative data** = Quantitative (Q)
- **Temporal data** = Temporal (T)

[Munzner (ill. Maguire), 2014]
Specifying the Type

\texttt{zoo\_area:O} \hspace{1cm} \texttt{zoo\_area:N}
Different Channels for Different Attribute Types

**Magnitude Channels: Ordered Attributes**
- Position on common scale
- Position on unaligned scale
- Length (1D size)
- Tilt/angle
- Area (2D size)
- Depth (3D position)
- Color luminance
- Color saturation
- Curvature
- Volume (3D size)

**Identity Channels: Categorical Attributes**
- Spatial region
- Color hue
- Motion
- Shape

Altair will use its rules to pick whether to use color hue or saturation based on the type.

[Altair (ill. Maguire), 2014]
Multiple Views in Visualization
Multiple Views in Visualization

[Improvise, Weaver, 2004]
Multiple Views in Visualization
Altair Supports Concatenation, Layering, & Repetition

- **Layering:**
  - + Operator

- **Concatenation:**
  - Horizontal: | operator
  - Vertical: & operator

- **Repetition**
  - Use of .repeat for layout
  - Reference repeated variables in the encoding
[Rock 'N' Roll is Here to Pay, R. Garofalo, 1977 (via Tufte)]
Also Visualization, but with Interaction

[Music Timeline, Google Research (no working version)]
Interaction

- Grammar of Graphics, why not Grammar of Interaction?
- Vega-Lite/Altair is about **interactive** graphics
- Types of Interactions:
  - Selection
  - Zoom
  - Brushing
Selection

- Selection is often used to initiate other changes
- User needs to select something to drive the next change
- What can be a selection target?
  - Items, links, attributes, (views)
- How?
  - mouse click, mouse hover, touch
  - keyboard modifiers, right/left mouse click, force
- Selection modes:
  - Single, multiple
  - Contiguous?
Highlighting

- Selection is the user action
- Feedback is important!
- How? Change selected item's visual encoding
  - Change color: want to achieve visual popout
  - Add outline mark: allows original color to be preserved
  - Change size (line width)
  - Add motion: marching ants
Highlighting

- Selection is the user action
- Feedback is important!
- How? Change selected item's visual encoding
  - Change color: want to achieve visual popout
  - Add outline mark: allows original color to be preserved
  - Change size (line width)
  - Add motion: marching ants
Altair's Interactive Charts

Weather Selection: Rain vs. Sun

Seattle Weather: 2012-2015

Weather Selection: Rain vs. Sun

D. Koop, CSCI 503/490, Spring 2023
Date Selection: July-September Sun

Seattle Weather: 2012-2015

Date: Jan 01 Feb 01 Mar 01 Apr 01 May 01 Jun 01 Jul 01 Aug 01 Sep 01 Oct 01 Nov 01 Dec 01

Maximum Daily Temperature (°C)

-5 0 5 10 15 20 25 30 35 40

Count of Records

0 10 20 30 40 50

Date Selection: July-September Sun