Advanced Institute for Artificial Intelligence

# Data Visualization for Exploration

https://advancedinstitute.ai

# Data Visualization

## Exploratory Data Analysis with Python

# References

## References and Image Sources

- Python Data Science Handbook: Essential Tools for Working with Data (Book)
- Fundamentals of Data Visualization: A Primer on Making Informative and Compelling Figures (Book)
- Claus O. Wilke - Fundamentals of Data Visualization (Free online)
- Edward Tufte - The Visual Display of Quantitative Information (Book)
- Matplotlib Gallery
- Seaborn Gallery

# Part 1

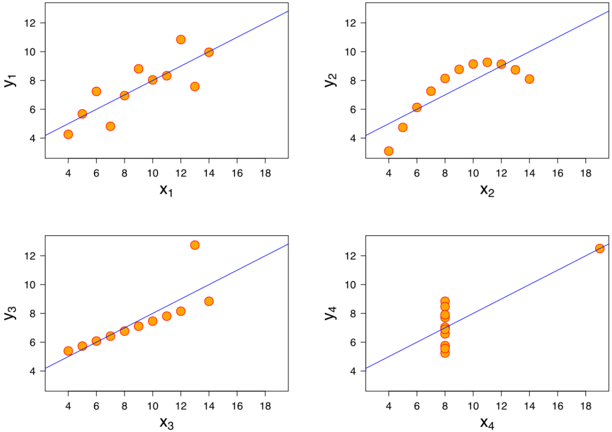
## Why Visualization Matters

# The Power of Visualization

## Anscombe's Quartet (1973)

- Four datasets with **identical statistical properties**:
  - Same mean of X and Y
  - Same variance of X and Y
  - Same correlation
  - Same linear regression line
- But they look **completely different** when plotted!
- **Lesson:** Summary statistics alone can be misleading
- **Always visualize your data!**

## Anscombe's Quartet (1973)



# Why Visualize Data?

## The Role of Visualization in Data Exploration

- **Pattern Discovery:**
  - Identify trends, clusters, outliers, and relationships
  - See what summary statistics cannot reveal
- **Data Quality Assessment:**
  - Spot missing values, errors, and anomalies
  - Understand data distributions
- **Hypothesis Generation:**
  - Visual exploration leads to questions and insights
- **Communication:**
  - Share findings with stakeholders
  - One picture can replace thousands of numbers

# Exploratory vs Explanatory Visualization

## Two Different Purposes

- **Exploratory** (our focus today):
  - For **you** to understand the data
  - Quick, iterative, many plots
  - Aesthetics are secondary to insight
  - Interactive exploration is valuable
- **Explanatory:**
  - For **others** to understand your findings
  - Polished, carefully designed
  - Clear message and narrative
  - Aesthetics and clarity are critical
- Today: focus on **exploratory** visualization for data analysis

# Data Visualization Principles

## Fundamental Guidelines

- **Accuracy:** Represent data truthfully
  - Visual encoding must match the data
  - If a value is 2x another, it should **look** 2x larger
- **Clarity:** Make the message obvious
  - Avoid clutter and chartjunk
  - Use appropriate chart types
- **Accessibility:** Design for all audiences
  - Consider color blindness (8% of men, 0.5% of women)
  - Readable fonts and labels
- **Honesty:** Don't mislead
  - Start axes at zero (for bar charts)
  - Use appropriate scales

## How Humans Process Visual Information

- We are **better** at perceiving:
  - **Position** along a common scale (most accurate)
  - **Length** and **direction**
  - **Angle** and **slope**
- We are **worse** at perceiving:
  - **Area** and **volume**
  - **Color saturation** and **density**
  - **Curvature**
- **Implication:** Choose encodings that align with our perceptual strengths
  - Bar charts $>$ Pie charts (position vs. angle)

## Introduction - Part Art, Part Science

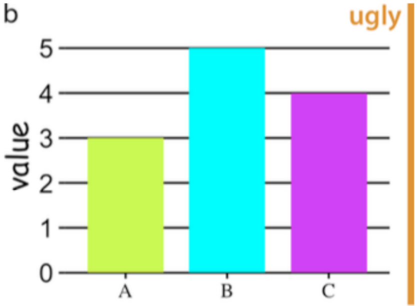
- Get the art right **without getting the science wrong** and vice versa
- Communicate data with **accuracy**
  - Do not deceive or distort data
  - *If a number* ***is twice as large as another****, but in the visualization they* ***appear to be almost equal****, then the* ***visualization is wrong***
- Must be **visually pleasing**
  - *If a figure contains* ***dissonant colors****,* ***unbalanced visual elements****, or other* ***distracting features****, the observer will have* ***more difficulty interpreting the figure correctly***

# Common Mistakes

## Ugly, Bad, and Wrong Figures

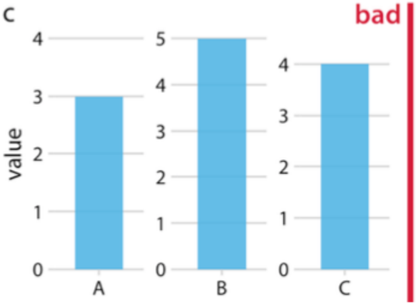
## Introduction - Ugly, bad, and wrong figures

- Ugly figures:
  - Aesthetic problems, but clear and informative
  - Can still communicate effectively
  - Example: default matplotlib with poor color choices



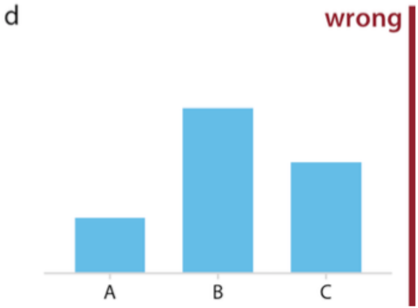
## Introduction - Ugly, bad, and wrong figures

- Bad figures:
  - Perception-related problems
  - Unclear, confusing, overly complicated, or misleading
  - Makes it difficult to extract the correct information



## Introduction - Ugly, bad, and wrong figures

- Wrong figures:
  - Mathematical or factual problems
  - Objectively incorrect
  - Misrepresents the data



## When Pie Charts Go Wrong

- **Problem:** Humans are poor at judging angles and areas
- **When pie charts fail:**
  - Too many slices ($>$ 5-7 categories)
  - 3D effects that distort perception
  - Similar-sized slices (hard to compare)
  - Exploded slices without reason
- **Better alternative:** Bar chart
  - Position along common scale is easier to judge
- **When pie charts are OK:**
  - 2-3 categories
  - Showing parts of a whole
  - One slice is clearly dominant ($>$ 50%)

# Bad Practice #2: Truncated Y-Axis

## Misleading Bar Charts

- **Problem:** Bar charts with Y-axis not starting at zero
- **Why it's wrong:**
  - Bar length should be proportional to value
  - Truncating exaggerates differences
  - Example: Value of 98 vs 100 looks 10x different if axis starts at 95
- **Rule:** Bar charts should **always start at zero**
- **Exception:** Line charts can have non-zero baselines
  - We judge slope and position, not length
- **If differences are too small to see with zero baseline:**
  - Reconsider if bar chart is appropriate
  - Consider showing differences or percentage changes

# Bad Practice #3: Dual Y-Axes

## Two Scales, Double Trouble

- **Problem:** Two different Y-axes on same plot
- **Why it's problematic:**
  - Can make any correlation appear by adjusting scales
  - Confusing to readers
  - Which axis goes with which line?
- **Better alternatives:**
  - Use two separate plots (small multiples)
  - Normalize data to same scale
  - Plot one variable vs. the other (scatter)
- **Acceptable use case:**
  - When both variables share clear relationship (e.g., temperature in °C and °F)

# Bad Practice #4: Chartjunk

## Less is More

- **Chartjunk:** Non-data ink that doesn't enhance understanding
- **Examples:**
  - Unnecessary 3D effects
  - Decorative backgrounds and patterns
  - Excessive grid lines
  - Redundant labels and legends
  - Clipart and decorations
- **Edward Tufte's principle:**
  - Maximize **data-ink ratio**: ink used for data / total ink
  - Remove everything that doesn't add information
- **Goal:** Let the data speak

# Bad Practice #5: Rainbow Color Maps

## Color Matters

- **Problem:** Rainbow/jet colormap for continuous data
- **Why it's bad:**
  - Not perceptually uniform (yellow appears brighter than blue)
  - Creates artificial boundaries in continuous data
  - Misleading for colorblind viewers (8% of men)
  - Does not print well in grayscale
- **Better alternatives:**
  - **Sequential:** viridis, plasma, cividis (perceptually uniform)
  - **Diverging:** RdBu, coolwarm (for data with meaningful center)
  - **Qualitative:** tab10, Set2 (for categorical data)
- **Rule:** Use colorblind-friendly palettes

## Overplotting and Complexity

- **Problem:** Cramming too much into one plot
- **Symptoms:**
  - Overlapping points (overplotting)
  - Too many lines on one chart
  - Tiny, unreadable labels
  - More than 7-8 categories
- **Solutions:**
  - **Reduce opacity** for overlapping points
  - **Sample** large datasets
  - **Use small multiples** (faceting) instead of one crowded plot
  - **Interactive plots** for exploration
  - **Aggregate** data appropriately

# Bad Practice #7: No Context

## Missing Critical Information

- **Problem:** Plots without proper labels and context
- **What's missing:**
  - Axis labels (what are we plotting?)
  - Units (dollars? thousands? percentages?)
  - Title (what is this showing?)
  - Legend (what do colors/shapes mean?)
  - Source (where did this data come from?)
- **Minimum requirements for any plot:**
  - Descriptive axis labels with units
  - Clear title or caption
  - Legend when using multiple series
- **Remember:** Your plot should be self-explanatory

# Choosing the Right Chart

## Visualization Directory

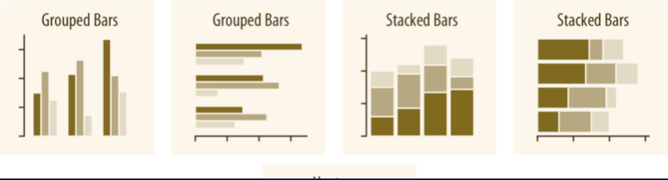
# The Chart Selection Process

## What Do You Want to Show?

- **Comparison:** How do values compare?
  - Bar charts, grouped bars, dot plots
- **Distribution:** How are values distributed?
  - Histograms, box plots, violin plots, density plots
- **Relationship:** How do variables relate?
  - Scatter plots, line charts, heatmaps
- **Composition:** What are the parts of the whole?
  - Stacked bars, area charts, treemaps
- **Time series:** How does it change over time?
  - Line charts, area charts
- **Choose based on your question, not preference!**

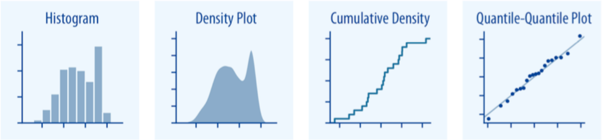
## Quantities

- **Numerical values** shown for a set of categories
- **Best practices:**
  - Vertical or horizontal bars
  - Start Y-axis at zero
  - Order by value (descending) unless natural order exists
  - Limit to 10-15 categories
- **Use cases:**
  - Comparing quantities across categories; Showing rankings; and Highlighting differences



## Distributions

- **Histograms and density plots:** most intuitive
  - Show shape: normal, skewed, bimodal? and Identify outliers
- **Box plots:** show summary statistics
  - Median, quartiles, outliers
  - Good for comparing multiple distributions
- **Violin plots:** combine box plot + density
- **Q-Q plots:** test for normality
  - More technical, harder to interpret
- **Tip:** Always check distribution before analysis!



## Choosing the Right Distribution Visualization

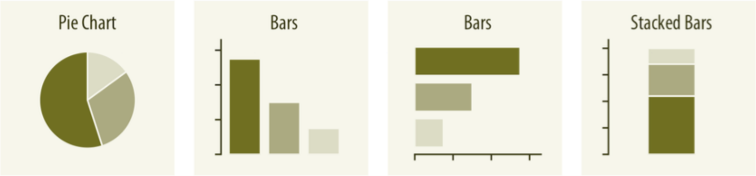
- **Histogram:**
  - Pros: Simple, intuitive, shows frequency
  - Cons: Bin size affects appearance
  - Use: Single distribution, count-based
- **Density plot (KDE):**
  - Pros: Smooth, doesn't depend on bins
  - Cons: Can be misleading if bandwidth is wrong
  - Use: Comparing multiple distributions

## Choosing the Right Distribution Visualization

- **Box plot:**
  - Pros: Shows statistics, compact, multiple groups
  - Cons: Hides distribution shape
  - Use: Comparing many distributions side-by-side
- **Violin plot:**
  - Pros: Shows distribution + statistics
  - Use: When you need both density and summary

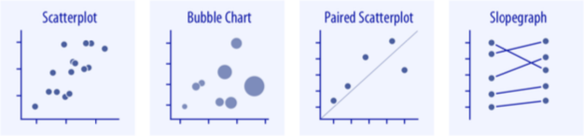
## Proportions

- **Showing parts of a whole**
- **Pie charts:** use sparingly!
  - OK for 2-3 categories
  - Avoid 3D, explosions, too many slices
- **Stacked bar charts:** often better than pie
  - Easier to compare and can show multiple groups
- **Treemap:** hierarchical proportions
  - Good for many nested categories



## X-Y Relationships

- **Scatter plots:** relationship between two quantitative variables
  - Identify correlation, clusters, outliers
  - Add regression line to show trend
- **Bubble charts:** add third variable as size
  - Don't overuse - hard to judge area
- **Line charts:** for continuous data (especially time series): shows trends and patterns
- **Heatmaps:** for correlation matrices
  - Shows many pairwise relationships
- **Tip:** Add transparency if points overlap

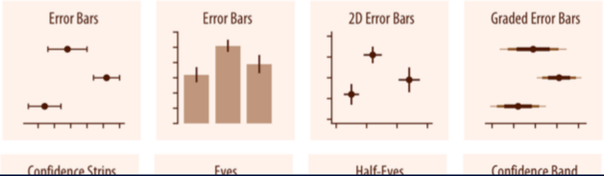


## Showing Change Over Time

- **Line charts:** standard for time series
  - Shows trends, seasonality, cycles
  - Multiple lines for comparison
- **Area charts:** emphasize magnitude
  - Stacked areas show composition over time
- **Best practices:**
  - Put time on X-axis (left to right)
  - Don't connect discrete events with lines
  - Show enough context (not just recent data)
  - Annotate important events
- **Common mistakes:**
  - Too many lines (hard to distinguish)
  - Inconsistent time intervals

## Representing Uncertainties

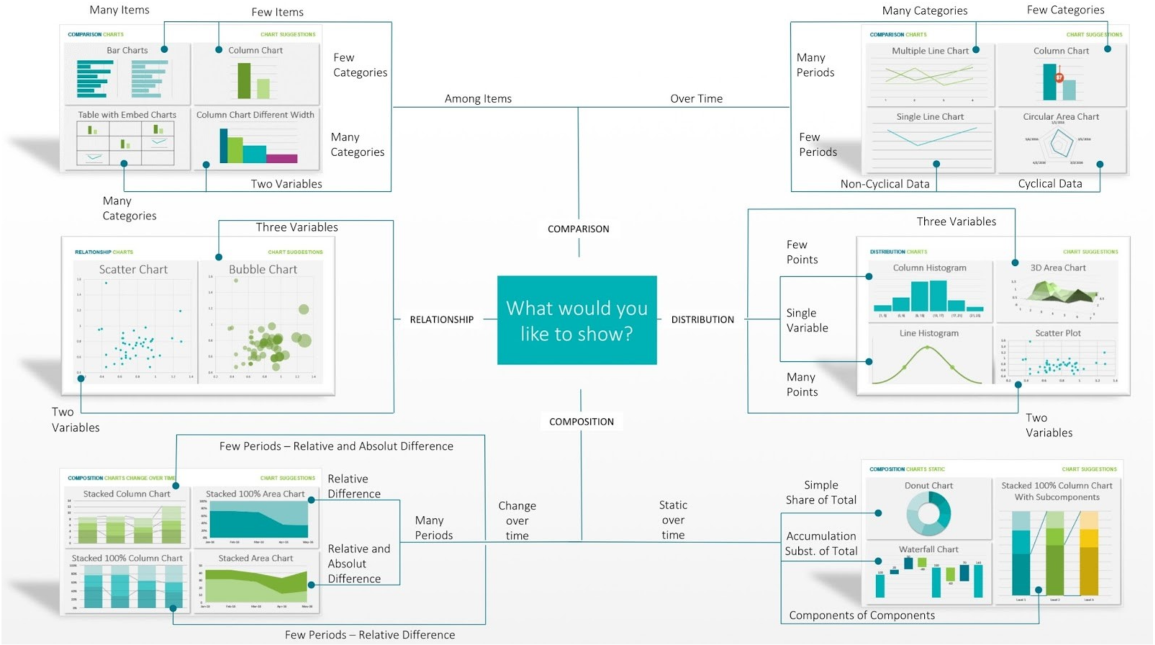
- **Error bars:** range of likely values
  - Standard deviation, standard error, confidence intervals
- **Always specify what error bars represent!**
- **Confidence bands:** for regression lines
- **Transparency/shading:** show uncertainty range
- **Common mistakes:**
  - Not explaining what the bars mean
  - Error bars on bar charts (use dot plots instead)



# Small Multiples (Faceting)

## The Power of Repetition

- **Small multiples:** same chart structure, different data subsets
- **Advantages:**
  - Compare patterns across groups
  - Avoid overplotting
  - Clearer than many lines on one plot
  - Eye can easily compare same structure
- **When to use:**
  - Comparing across categories (cities, products, years)
  - Time series for multiple groups
  - Distribution across subgroups
- **Keep consistent:**
  - Same axis scales
  - Same colors/styles
  - Logical ordering


What would you like to show?
COMPARISON
Among Items
Over Time
Many Items
Few Items
COMPARISON CHARTS
CHART SUGGESTIONS
Bar Charts
Column Chart
Few Categories
Table with Embed Charts
Column Chart Different Width
Many Categories
Two Variables
Many Categories
Many Categories
Few Categories
Many Periods
Multiple Line Chart
Column Chart
Single Line Chart
Circular Area Chart
Few Periods
Non-Cyclical Data
Cyclical Data
RELATIONSHIP
Three Variables
RELATIONSHIP CHARTS
CHART SUGGESTIONS
Scatter Chart
Bubble Chart
Two Variables
DISTRIBUTION
Three Variables
Few Points
DISTRIBUTION CHARTS
CHART SUGGESTIONS
Column Histogram
3D Area Chart
Single Variable
Line Histogram
Scatter Plot
Many Points
Two Variables
COMPOSITION
Few Periods – Relative and Absolut Difference
COMPOSITION CHARTS CHANGE OVER TIME
CHART SUGGESTIONS
Stacked Column Chart
Stacked 100% Area Chart
Relative Difference
Many Periods
Change over time
Static over time
Stacked 100% Column Chart
Stacked Area Chart
Relative and Absolute Difference
Few Periods – Relative Difference
COMPOSITION CHARTS STATIC
CHART SUGGESTIONS
Simple Share of Total
Donut Chart
Stacked 100% Column Chart With Subcomponents
Accumulation Subst. of Total
Waterfall Chart
Components of Components

# Python Visualization Ecosystem

## Tools for Exploration

# Python Visualization Libraries

## Overview

- **Matplotlib:** The foundation
  - Low-level control, highly customizable
  - Verbose syntax
  - Static plots
- **Seaborn:** Statistical visualization
  - Built on matplotlib, higher-level API
  - Beautiful defaults, statistical functions
  - Great for exploration
- **Pandas plotting:** Quick and dirty
  - df.plot() for rapid exploration
  - Limited customization
- **Plotly:** Interactive plots
  - Hover, zoom, pan
  - Good for dashboards

# When to Use Which Library?

## Decision Guide

- **Quick exploration:** Pandas .plot()
  - Fast, minimal code
  - Limited customization
- **Statistical analysis:** Seaborn
  - Distribution plots, regression, categorical
  - Beautiful defaults, less code
- **Full control:** Matplotlib
  - Custom plots, publication-quality
  - More verbose, steeper learning curve
- **Interactive exploration:** Plotly
  - Large datasets, need to explore interactively
  - Dashboards and web applications
- **For today: Focus on Matplotlib + Seaborn**

## Matplotlib - The Foundation

- Cross-platform data visualization library built on NumPy arrays
- Designed to work with the broader SciPy stack
- Works well with many operating systems and graphics backends
- Highly customizable appearance
- **Two interfaces:**
  - MATLAB-style (pyplot): quick and simple
  - Object-oriented: more control, better for complex plots
- Cons: Verbose syntax, no interactive capability by default 😔
- **Today:** We'll use mostly the OO interface

## MATLAB-style vs Object-Oriented

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

# MATLAB-style (pyplot) - Quick and simple
plt.plot([1, 2, 3], [1, 4, 9])
plt.xlabel('X')
plt.ylabel('Y')
plt.title('Simple Plot')
plt.show()

# Object-Oriented - More control
fig, ax = plt.subplots()
ax.plot([1, 2, 3], [1, 4, 9])
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_title('Simple Plot')
plt.show()
```

## Matplotlib - From Python Script

- From a python file:

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

x = np.linspace(0, 10, 100)
fig, ax = plt.subplots()
ax.plot(x, np.sin(x), label='sin(x)')
ax.plot(x, np.cos(x), label='cos(x)')
ax.set_xlabel('x')
ax.set_ylabel('y')
ax.legend()

plt.show() # Display the plot
```

- A single `plt.show()` command blocks execution, Use `plt.savefig('plot.png')` to

## Matplotlib - From Jupyter Notebook

- From a Jupyter Notebook:

```
%matplotlib inline
# Alternative: %matplotlib notebook (interactive)

import matplotlib.pyplot as plt
import numpy as np

x = np.linspace(0, 10, 100)
fig, ax = plt.subplots(figsize=(10, 6))
ax.plot(x, np.sin(x), '-', label='sin(x)')
ax.plot(x, np.cos(x), '--', label='cos(x)')
ax.set_xlabel('x')
ax.set_ylabel('y')
ax.legend()
ax.grid(True, alpha=0.3)
```

# Matplotlib Figure Anatomy

## Understanding the Components

- **Figure:** The entire plot window
  - Can contain multiple subplots (axes)
  - Set size with `figsize=(width, height)`
- **Axes:** A single plot area
  - Can have multiple axes in one figure
  - Has x-axis, y-axis, title, labels
- **Axis:** The x or y axis
  - Controls ticks, tick labels, limits
- **Key concept:**
  - `fig, ax = plt.subplots()`
  - Then use `ax.method()` to customize

## Multiple Plots in One Figure

```python
# Create 2x2 grid of subplots
fig, axes = plt.subplots(2, 2, figsize=(12, 10))
# Access individual subplots
axes[0, 0].plot(x, np.sin(x))
axes[0, 0].set_title('Sine')

axes[0, 1].plot(x, np.cos(x))
axes[0, 1].set_title('Cosine')

axes[1, 0].plot(x, np.tan(x))
axes[1, 0].set_title('Tangent')

axes[1, 1].plot(x, np.exp(x))
axes[1, 1].set_title('Exponential')

plt.tight_layout()  # Adjust spacing
```

# Seaborn - Statistical Visualization

## Built on Matplotlib, Made for Data

- High-level interface for statistical graphics
- **Advantages:**
  - Beautiful default styles
  - Works directly with Pandas DataFrames
  - Statistical functions built-in
  - Less code for complex plots
- **Best for:**
  - Exploratory data analysis
  - Distribution plots
  - Categorical plots
  - Regression plots
- Still uses matplotlib underneath - can mix both!

# Seaborn Quick Example

## Much Less Code for Statistical Plots

```python
import seaborn as sns
import pandas as pd
# Load example dataset
df = pd.read_csv('sales_data.csv')

# Distribution plot with one line
sns.histplot(data=df, x='Sales', hue='City', kde=True)

# Box plot with one line
sns.boxplot(data=df, x='City', y='Sales')

# Scatter with regression line
sns.regplot(data=df, x='Cost', y='Sales')

# Correlation heatmap
sns.heatmap(df.corr(), annot=True, cmap='coolwarm')
```

# Seaborn Plot Types

## Key Functions for Exploration

- **Distribution plots:**
  - `histplot()`: histograms with KDE
  - `kdeplot()`: density plots
  - `boxplot()`, `violinplot()`: summary + distribution
- **Categorical plots:**
  - `barplot()`: means with confidence intervals
  - `countplot()`: count of observations
  - `stripplot()`, `swarmplot()`: individual points
- **Relationship plots:**
  - `scatterplot()`: x-y relationships
  - `lineplot()`: time series
  - `regplot()`: scatter + regression
  - `heatmap()`: correlation matrices

# Exploratory Data Analysis Workflow

## Using Visualization

- **Step 1: Understand individual variables**
  - Distribution: `histplot()`, `boxplot()`
  - Look for: skewness, outliers, gaps
- **Step 2: Examine relationships**
  - Scatter plots: `scatterplot()`, `pairplot()`
  - Correlation: `heatmap(df.corr())`
- **Step 3: Compare groups**
  - Box plots, violin plots by category
  - Small multiples (faceting)
- **Step 4: Identify patterns and anomalies**
  - Time series plots
  - Highlight outliers
- **Iterate and refine!**

# Practical Tips for Exploration

## Making Your EDA More Effective

- **Start simple, add complexity**
  - Begin with basic plots, enhance as needed
- **Use appropriate figure sizes**
  - `figsize=(10, 6)` for most plots
  - Larger for complex plots with many subplots
- **Always label your axes**
  - Include units!
- **Use color meaningfully**
  - Group categories, show gradients
  - Avoid rainbow colormaps
- **Save your plots**
  - `plt.savefig('plot.png', dpi=300, bbox_inches='tight')`
- **Create a plotting function for repeated analysis**

# Color in Data Visualization

## Using Color Effectively

- **Sequential:** ordered data (low to high)
  - viridis, plasma, Blues, Greens
  - Use for: heatmaps, geographic data
- **Diverging:** data with meaningful center
  - RdBu, coolwarm, PiYG
  - Use for: correlation, deviations from mean
- **Qualitative:** categorical data
  - tab10, Set2, Paired
  - Use for: different categories
- **Best practices:**
  - Limit to 7-8 colors
  - Use colorblind-friendly palettes
  - Test in grayscale

# Common Matplotlib Customizations

## Making Plots Publication-Ready

- **Styling:**
  - `plt.style.use('seaborn-v0_8')`
  - `sns.set_theme(style='whitegrid')`
- **Figure size:**
  - `fig, ax = plt.subplots(figsize=(10, 6))`
- **Fonts:**
  - `plt.rcParams['font.size'] = 12`
- **Grids:**
  - `ax.grid(True, alpha=0.3, linestyle='--')`
- **Legends:**
  - `ax.legend(loc='best', frameon=False)`
- **Tight layout:**
  - `plt.tight_layout()`

# Interactive Visualization

## Beyond Static Plots

- **Why interactive?**
  - Explore large datasets
  - Zoom, pan, hover for details
  - Better for presentations and dashboards
- **Tools:**
  - **Plotly:** interactive plots, works in Jupyter
  - **Bokeh:** interactive web visualizations
  - **Altair:** declarative statistical visualization
  - **Panel/Dash:** interactive dashboards
- **In Jupyter:**
  - `%matplotlib notebook` for basic interactivity
  - Plotly Express for quick interactive plots
- **Trade-off:** More complex, larger file sizes

# Checklist for Good Plots

## Before Sharing Your Visualization

- ☐ Axes labeled with units
- ☐ Title or caption explains what's shown
- ☐ Legend when using multiple series
- ☐ Appropriate chart type for the data
- ☐ Y-axis starts at zero (for bar charts)
- ☐ Colorblind-friendly palette
- ☐ Readable font sizes
- ☐ No chartjunk or unnecessary elements
- ☐ Data represented accurately
- ☐ Clear and unambiguous message

# Summary

- **Always visualize** - don't trust summary statistics alone
- **Choose the right chart** for your question
- **Prioritize clarity** over aesthetics
- **Be honest** with your data representation
- **Common mistakes to avoid:**
  - Pie charts with too many slices
  - Truncated bar chart axes
  - Rainbow colormaps
  - Too much information in one plot
- **For exploration:** Use Seaborn and Pandas plotting
- **For publication:** Use Matplotlib with careful customization
- **Practice makes perfect!**

# Resources for Further Learning

## Continue Your Journey

- **Books:**
  - Fundamentals of Data Visualization (Claus Wilke) - Free online
  - The Visual Display of Quantitative Information (Edward Tufte)
- **Online galleries:**
  - Matplotlib gallery: matplotlib.org/stable/gallery
  - Seaborn gallery: seaborn.pydata.org/examples
  - Python Graph Gallery: python-graph-gallery.com
- **Practice datasets:**
  - Seaborn built-in datasets
  - Kaggle datasets
- **Critique bad visualizations:** /r/dataisugly on Reddit

# Questions?