

# Visualization

Ruth Anderson

UW CSE 160

Autumn 2020

# matplotlib

- Strives to emulate MATLAB
  - Pro: familiar to MATLAB users
  - Pro: powerful
  - Con: not the best design for a plotting library
- One important function for HW6:

```
plot(xvalues, yvalues)
```

# Plot

```
import matplotlib.pyplot as plt
```

```
xs = [1, 2, 3, 4, 5]  
# ys = [x**2 for x in xs]  
ys = []  
for x in xs:  
    ys.append(x**2)
```

```
plt.plot(xs, ys)
```

```
plt.show()
```

no return value?

Has a side effect on the figure (like “print” statement)

```
import matplotlib.pyplot as plt
```

```
xs = range(-100, 110, 10)
```

```
x2 = [x**2 for x in xs]
```

```
negx2 = [-x**2 for x in xs]
```

```
plt.plot(xs, x2)
```

```
plt.plot(xs, negx2)
```

```
plt.xlabel("x")
```

```
plt.ylabel("y")
```

```
plt.ylim(-2000, 2000)
```

```
plt.axhline(0) # horiz line
```

```
plt.axvline(0) # vert line
```

```
plt.savefig("quad.png")
```

```
plt.show() # resets state
```

Incrementally  
modify the figure.

Save your figure to a file

Display plot

Call `savefig` before `show`, `show` clears the state

```

def myplot(xs, ys, description):
    plt.plot(xs, ys, linewidth=2, color='green', linestyle='-', marker='s', label=description)

def setup_plot():
    plt.xlabel("x")
    plt.ylabel("y")
    plt.axhline(0, linestyle=':', color='red')
    plt.axvline(0, linestyle=':', color='red')

def finish_plot():
    plt.legend()
    plt.show()

setup_plot()
myplot(xs, x2, "x**2")
finish_plot()

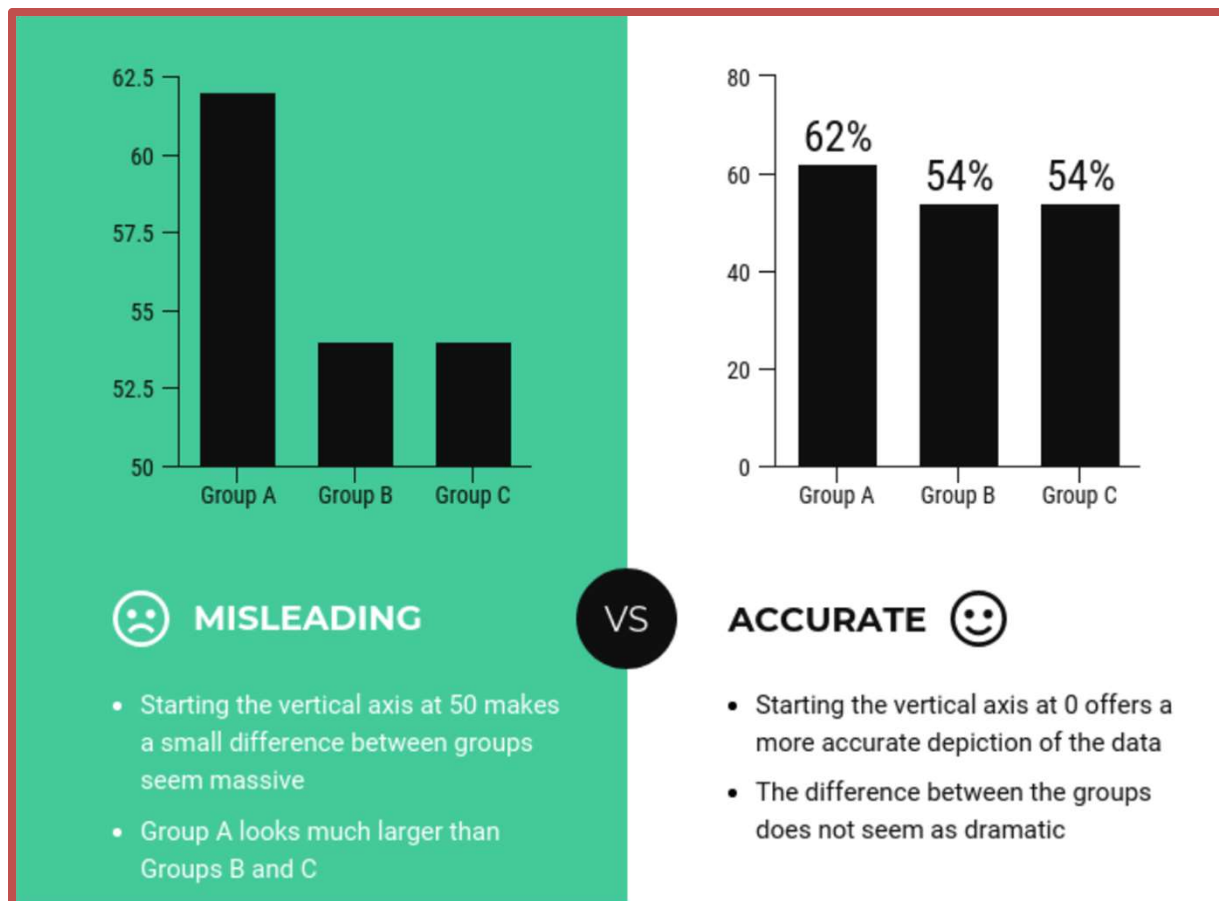
setup_plot()
myplot(xs, negx2, "-x**2")
finish_plot()

```

We can group these options into functions as usual, but remember that they are operating on a global, hidden variable (the figure)

# Visualization: Pros and Cons

- Visualizations can Mislead ([link](#)):



- Visualizations can be powerful:

London cholera epidemic  
Map by Dr. John Snow  
1854

