Introduction to the Course

INTRODUCTION TO LINEAR MODELING IN PYTHON

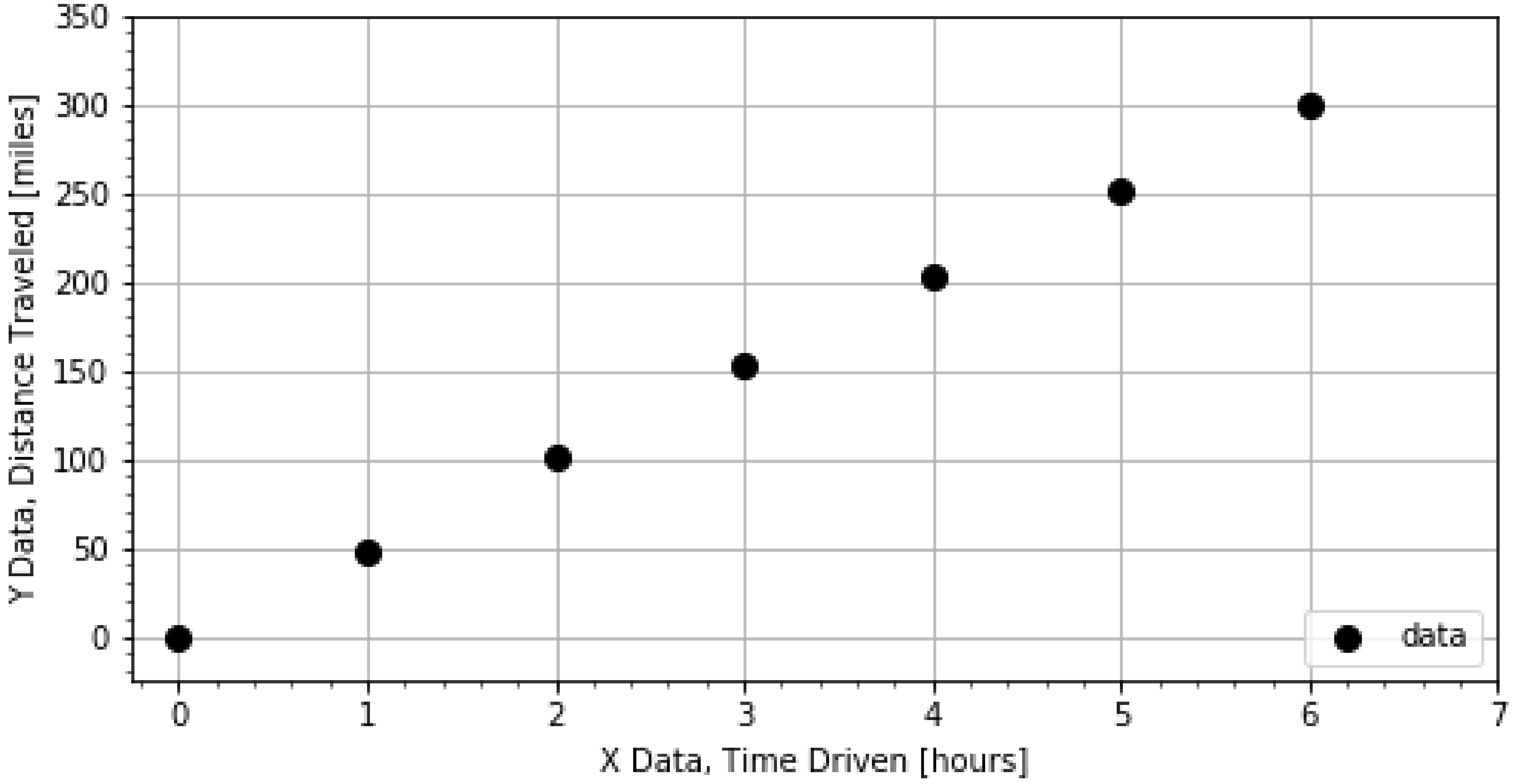
Jason Vestuto
Data Scientist

Introduction to Chapter 1

Chapter Roadmap:

- Motivating Examples
- Data Visualization
- Descriptive Statistics

Example Trip Data



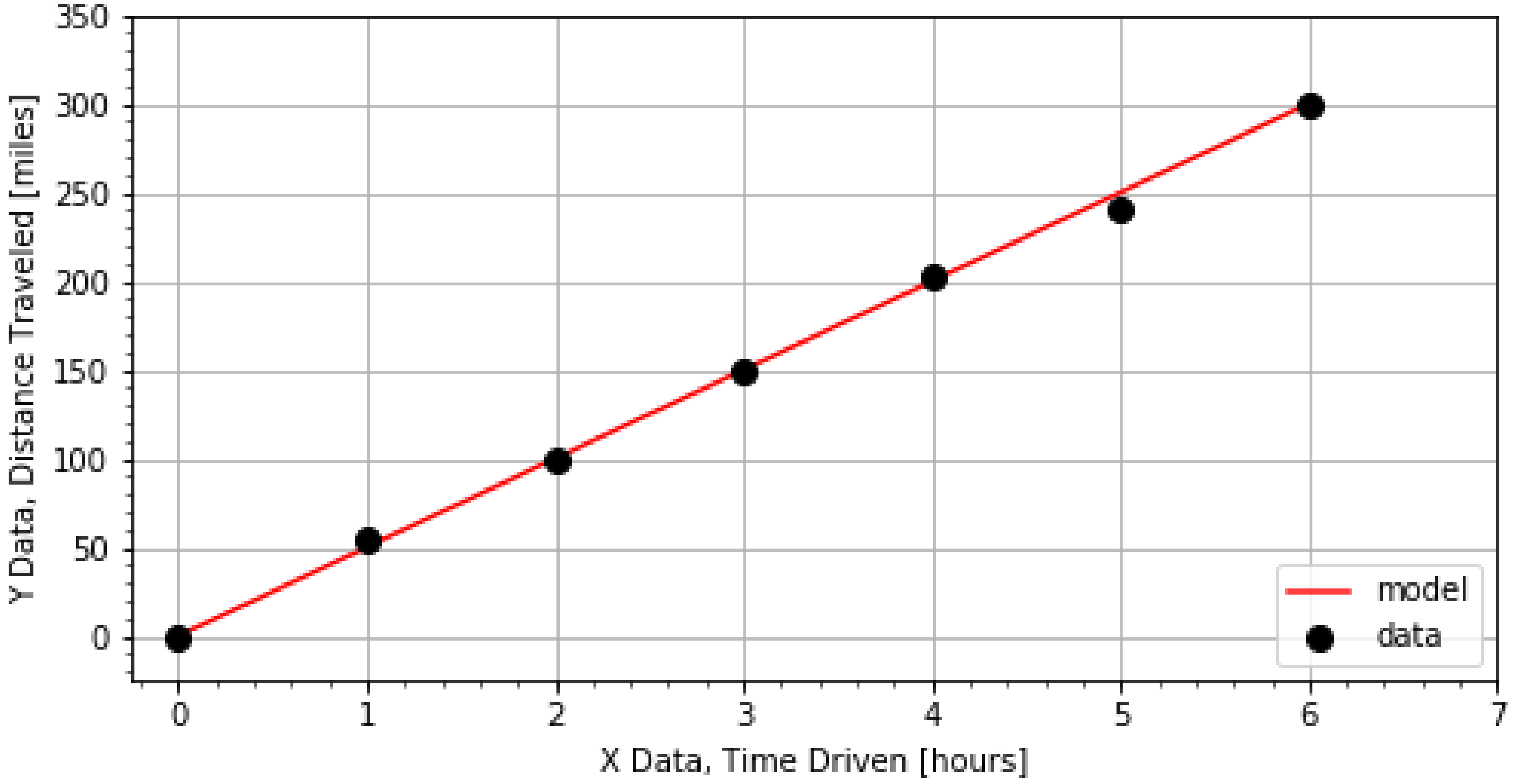
Models as Descriptions

```
# Range of y data, in miles
y_range = np.max(y) - np.min(y) = 300 - 0 = 300
```

```
# Range of x data, in hours
x_range = np.max(x) - np.min(x) = 6 - 0 = 6
```

```
# Estimating the speed
mph = y_range / x_range = 300 / 6 = 50
```

Visualizing a Model



Model Predictions

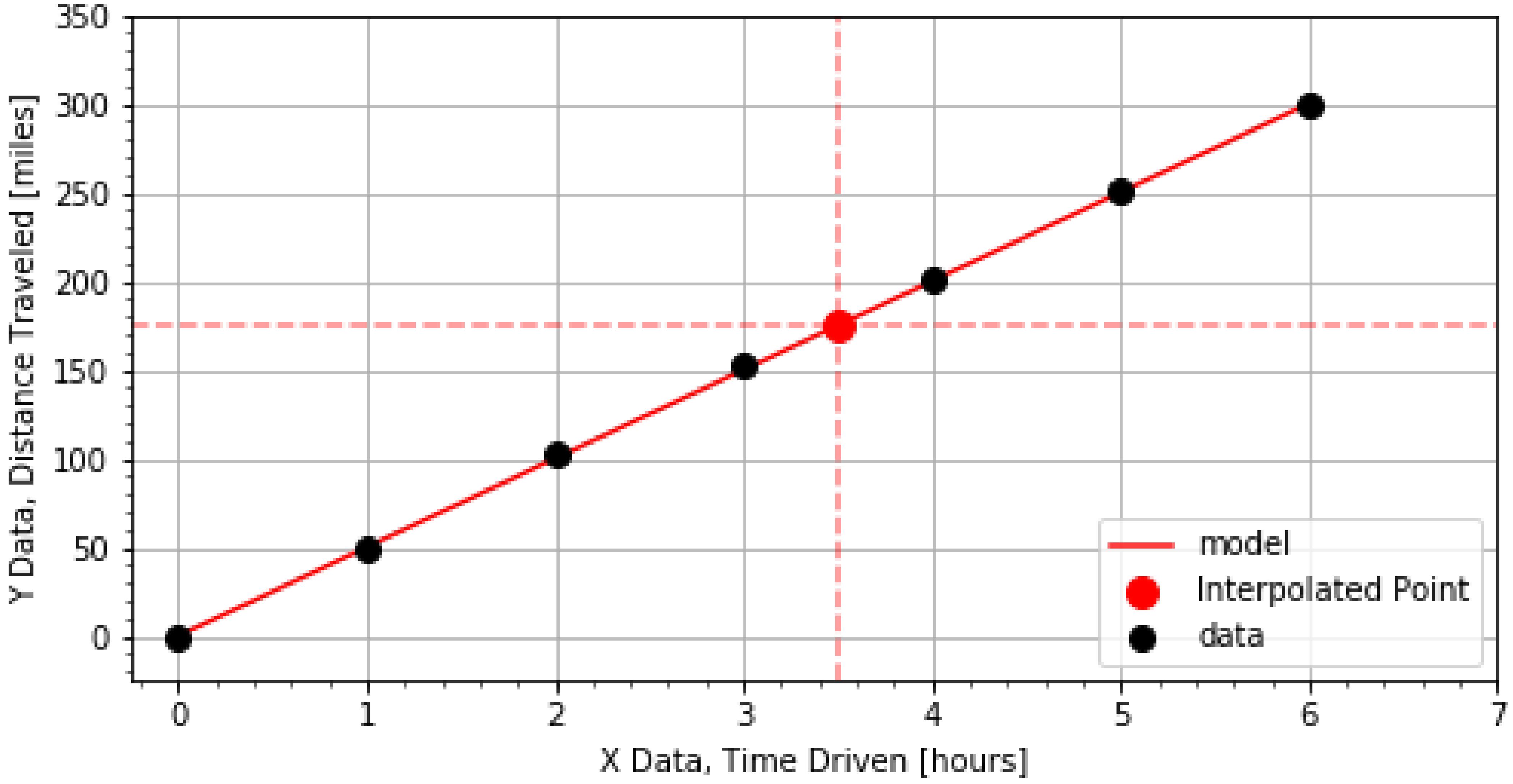
```
# Model as python expression
miles = 50*hours
```

```
# Model predicts distance is 300 miles at 6 hours
time = 6
distance = 50 * time = 50 * 6 = 300
```

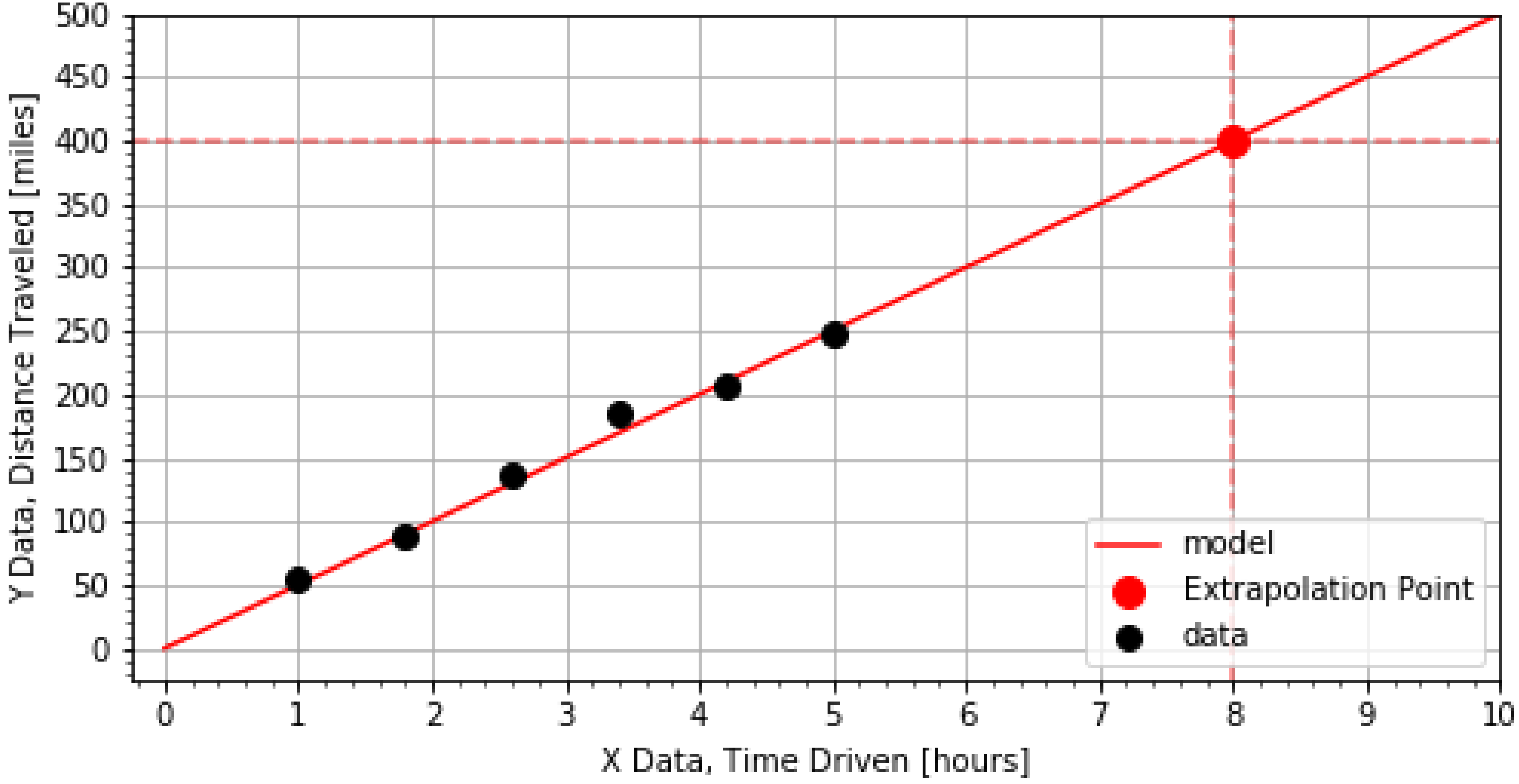
```
def model(time):
    return 50*time


predicted_distance = model(time=10)
```

Interpolation



Extrapolation



Let's practice!

Visualizing Linear Relationships

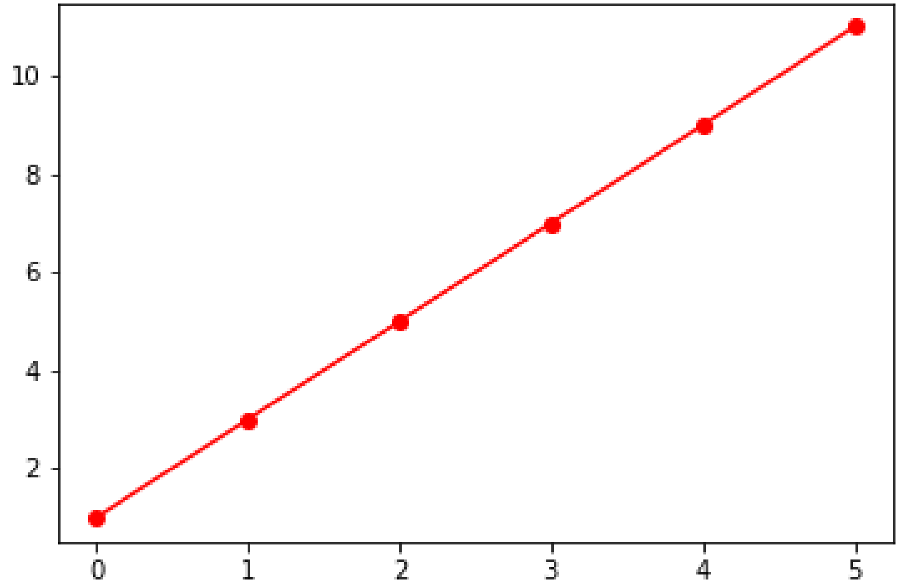
INTRODUCTION TO LINEAR MODELING IN PYTHON

Jason Vestuto
Data Scientist

Quick Plots

```
import matplotlib.pyplot as plt
plt.plot(x, y, 'r-o')
```

```
plt.show()
```



Object Interface

```
# Import the pyplot module
import matplotlib.pyplot as plt
```

```
# Create figure and axis objects
fig, axis = plt.subplots()
```

```
# Prepare initial style options
options = dict(marker='o', color='blue')
```

Object Interface

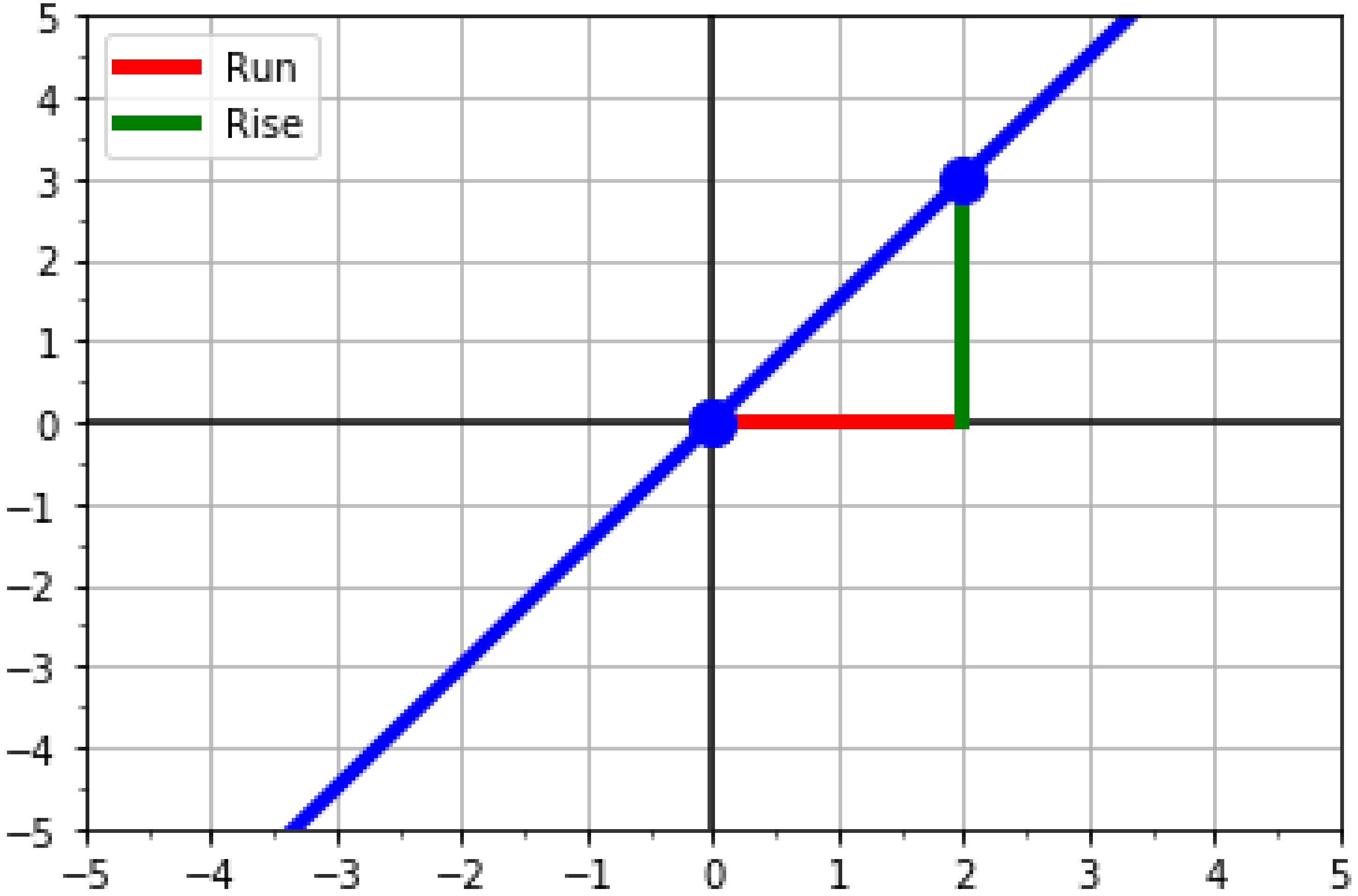
```
# Call the plot method on the axis object
line = axis.plot(x, y, **options)
```

```
# Modify the axis object with set methods
_ = axis.set_ylabel('Times')
_ = axis.set_xlabel('Distances')
```

```
# Display figure
plt.show()
```

Visualizing Linear Data

- two points:
  - `(x1,y1) = (0,0)`
  - `(x2,y2) = (2,3)`
- change in x and y:
  - `dy = (y2 - y1) = 3 - 0`
  - `dx = (x2 - x1) = 2 - 0`
- slope = rise-over-run
  - `slope = dy/dx = 3/2`
- intercept:
  - when `x=0` : `y1 = 0`



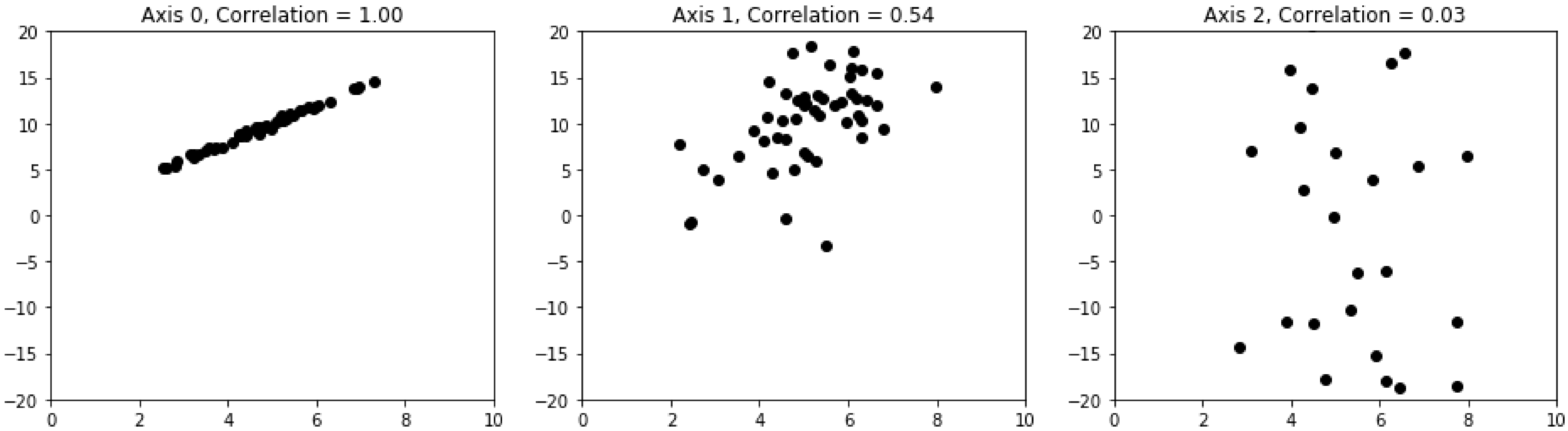
Let's practice!

Quantifying Linear Relationships

INTRODUCTION TO LINEAR MODELING IN PYTHON

Jason Vestuto
Data Scientist

Pre-Visualization



Review of Single Variable Statistics

```
# Mean
mean = sum(x)/len(x)
```

```
# Deviation, sometimes called "centering"
dx = x - np.mean(x)
```

```
# Variance
variance = np.mean(dx*dx)
```

```
# Standard Deviation
stdev = np.sqrt(variance)
```

Covariance

```
# Deviations of two variables
dx = x - np.mean(x)
dy = y - np.mean(y)
```

```
# Co-vary means to vary together
deviation_products = dx*dy
```

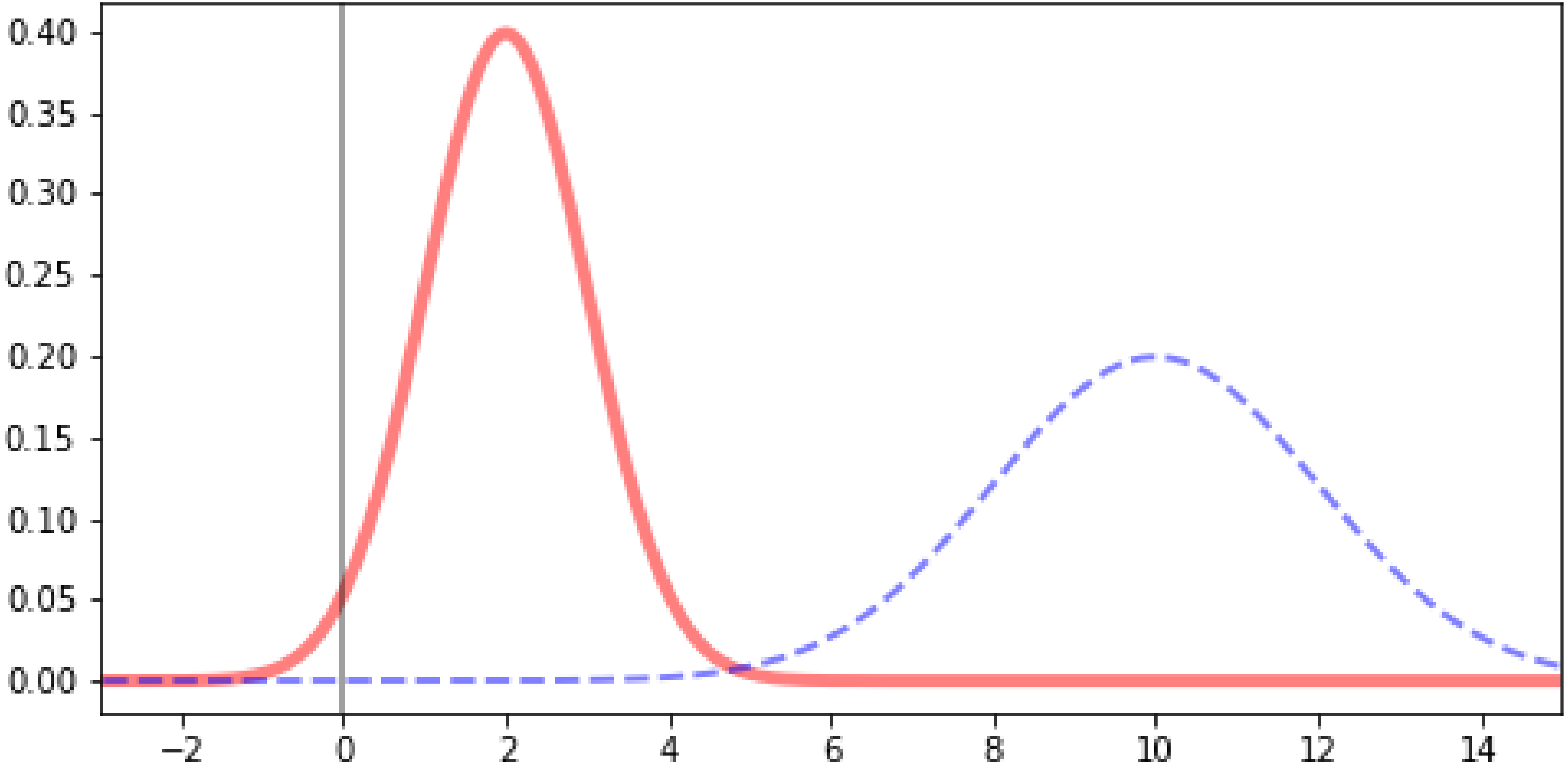
```
# Covariance as the mean
covariance = np.mean(dx*dy)
```

Correlation

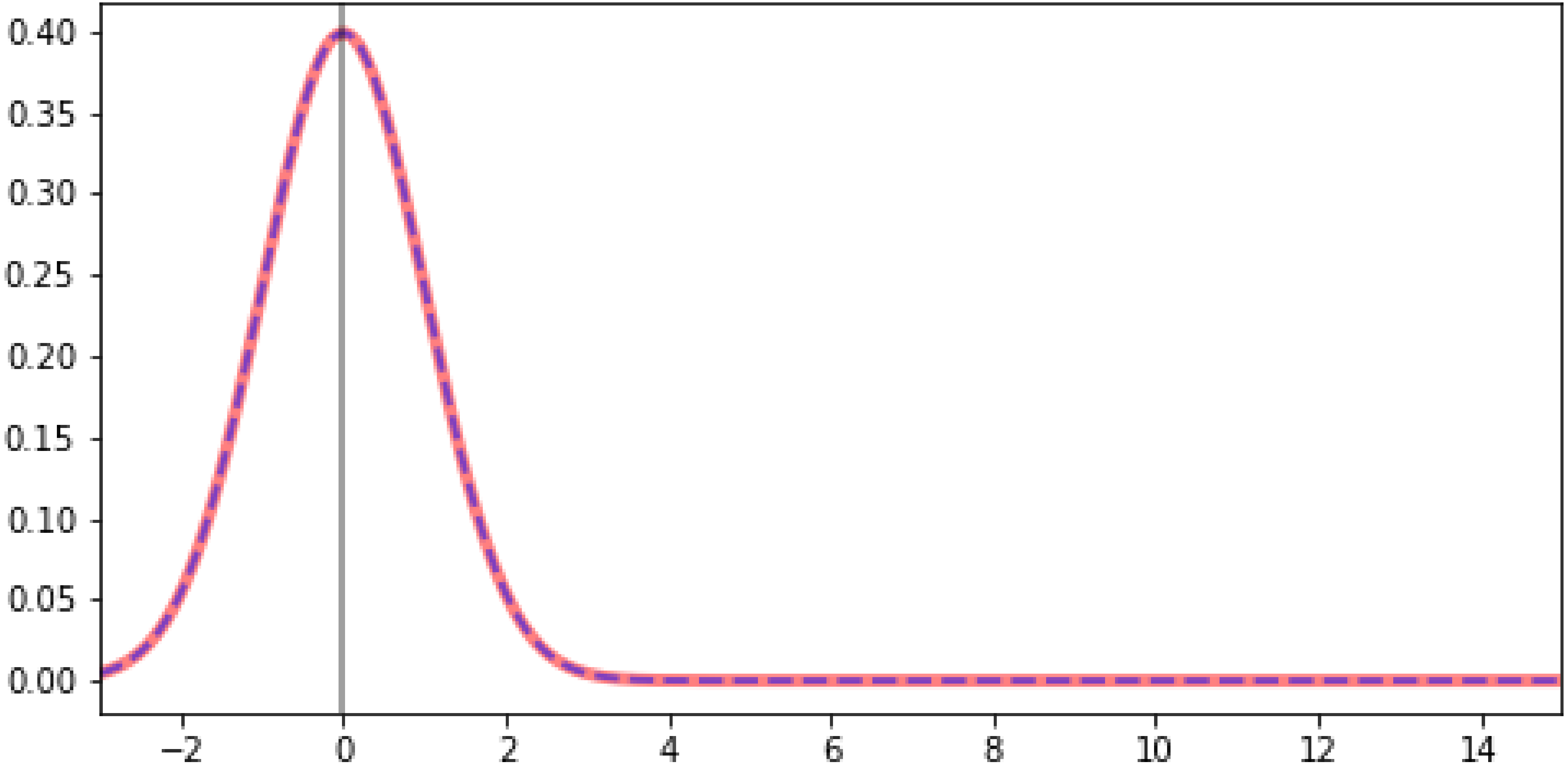
```
# Divide deviations by standard deviation
zx = dx/np.std(x)
zy = dy/np.std(y)
```

```
# Mean of the normalize deviations
correlation = np.mean(zx*zy)
```

Normalization: Before

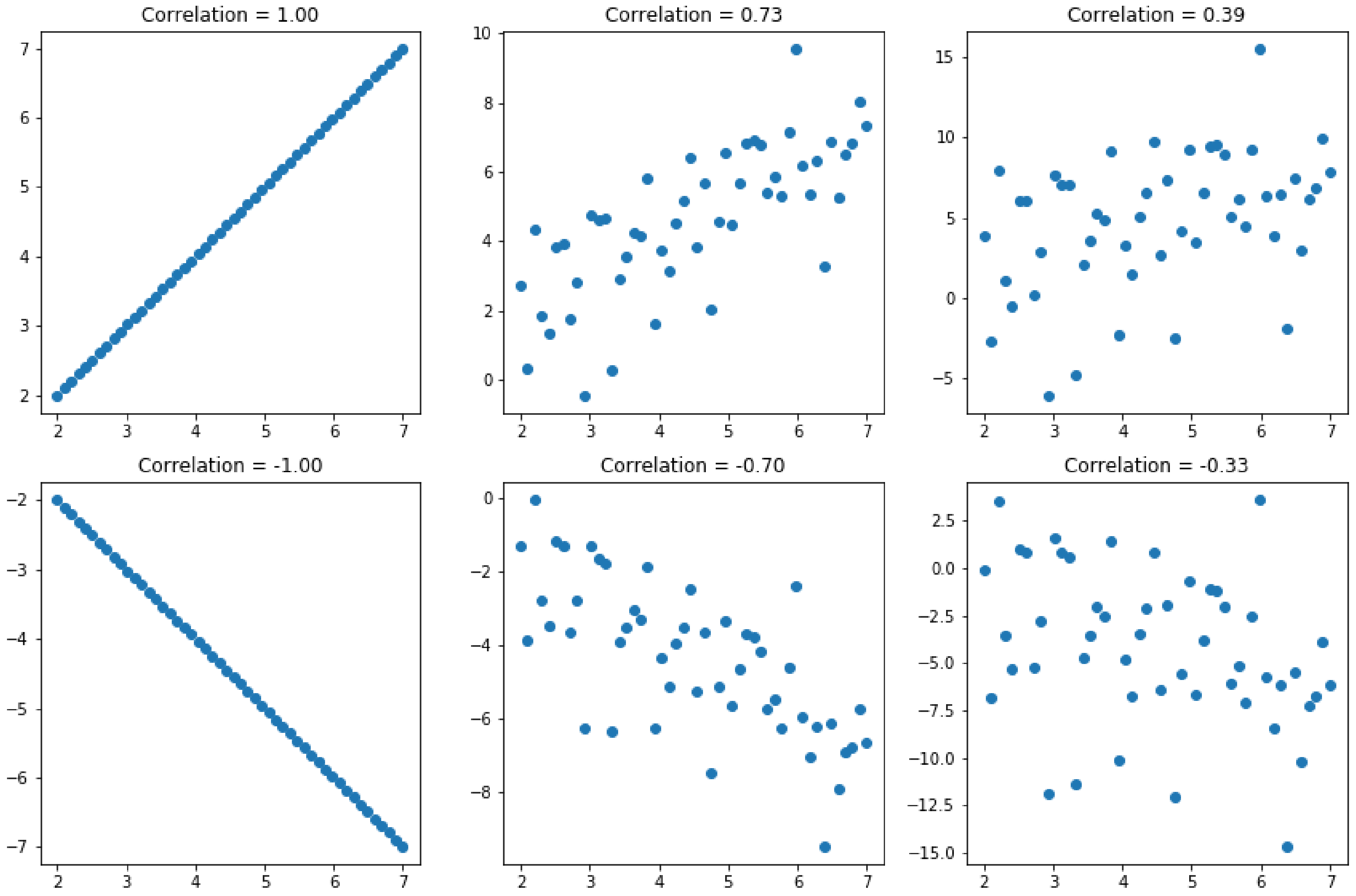


Normalization: After



Magnitude versus Direction

- Correlation values: -1 to +1



Let's practice!