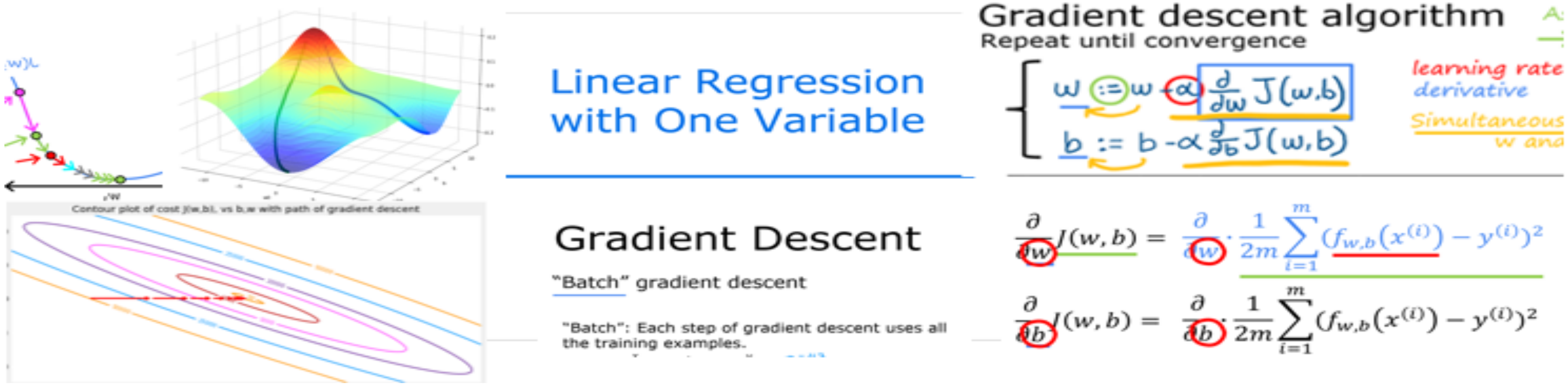
Optional Lab: Gradient Descent for Linear Regression


J(w)
w
Contour plot of cost J(w,b), vs b,w with path of gradient descent
learning rate
derivative
Simultaneous
w and

Linear Regression with One Variable

Gradient descent algorithm

Repeat until convergence

$$\begin{cases} w := w - \alpha \frac{\partial}{\partial w} J(w,b) \\ b := b - \alpha \frac{\partial}{\partial b} J(w,b) \end{cases}$$

Gradient Descent

"Batch" gradient descent

"Batch": Each step of gradient descent uses all the training examples.

$$\frac{\partial}{\partial w} J(w,b) = \frac{\partial}{\partial w} \frac{1}{2m} \sum_{i=1}^{m} (f_{w,b}(x^{(i)}) - y^{(i)})^2$$

$$\frac{\partial}{\partial b} J(w,b) = \frac{\partial}{\partial b} \frac{1}{2m} \sum_{i=1}^{m} (f_{w,b}(x^{(i)}) - y^{(i)})^2$$

Goals

In this lab, you will:

- automate the process of optimizing w and b using gradient descent.

Tools

In this lab, we will make use of:

- NumPy, a popular library for scientific computing
- Matplotlib, a popular library for plotting data
- plotting routines in the lab_utils.py file in the local directory

In [ ]:

```
import math, copy
import numpy as np
import matplotlib.pyplot as plt
```