Gradient Descent

CSC 461: Machine Learning

Fall 2022

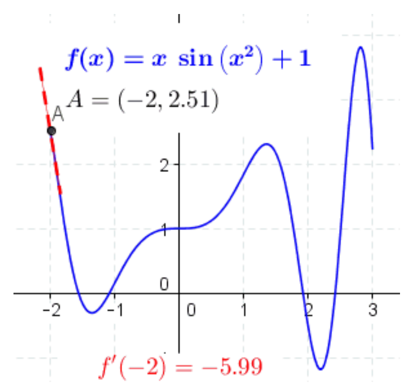
Prof. Marco Alvarez
University of Rhode Island

Derivatives and gradient vectors

From calculus …

- Given a function $f(x)$, its derivative $f'(x)$ tells us how much a very small increment to the argument will change the value of the function
  - ✓ steepness of the function at x
  - ✓ rate of change at x
- Positive slope
  - ✓ very small increase in x **increases** $f(x)$
- Negative slope
  - ✓ very small increase in x **decreases** f(x)

Scalar function of scalar argument



For a function $y = f(x)$ the derivative is given by $f'(x) = \alpha$, such that $\Delta y = \alpha \Delta x$

https://en.wikipedia.org/wiki/Derivative

Scalar function of multivariate argument

$$y = f(\mathbf{x})$$
$$\Delta y = \alpha \Delta \mathbf{x}$$

Input is a vector $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{bmatrix}$

Derivative is a row vector $\alpha = [\alpha_1, \alpha_2, \ldots, \alpha_d]$

$$\nabla_{\mathbf{x}} f(\mathbf{x}) = \left[\frac{\partial f(\mathbf{x})}{\partial x_1}, \frac{\partial f(\mathbf{x})}{\partial x_2}, \ldots, \frac{\partial f(\mathbf{x})}{\partial x_d}\right]$$

partial derivatives

how much a very small increment to α_i will change the output

What is the gradient?

- A **gradient** is the transpose of the **derivative**
  - ✓ a column vector of **partial derivatives**

$$\nabla_{\mathbf{x}} f(\mathbf{x})^T = \begin{bmatrix} \frac{\partial y}{\partial x_1} \\ \frac{\partial y}{\partial x_2} \\ \vdots \\ \frac{\partial y}{\partial x_d} \end{bmatrix}$$

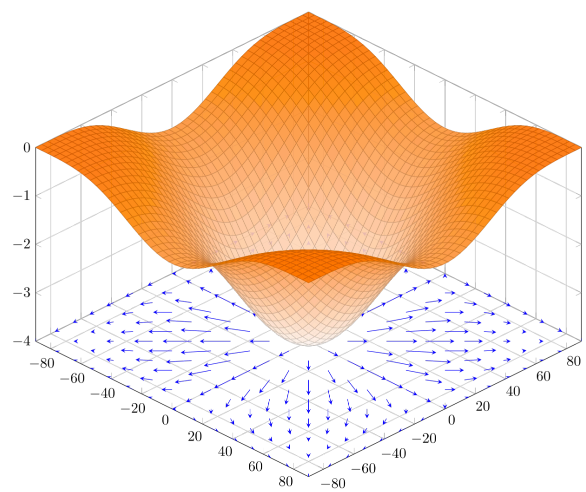
Example

- What is the gradient of:

$$f(\mathbf{x}) = x_1^3 + 2x_2 + 5x_3^4?$$

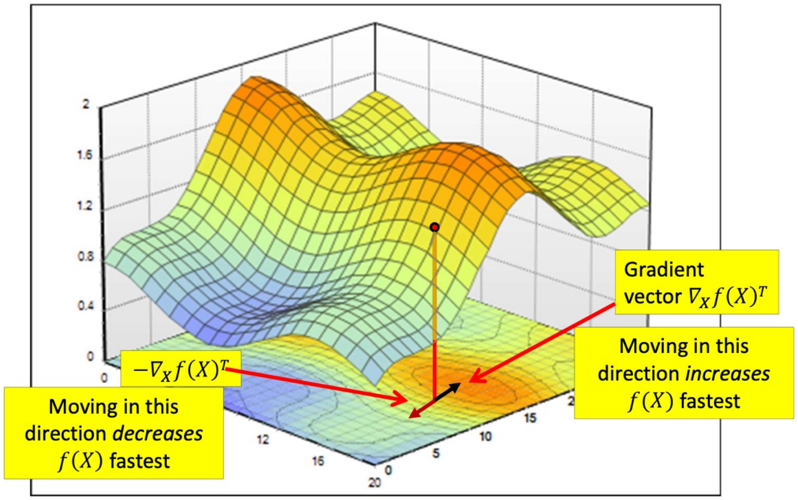
$$f(x, y, z) = x^3 + 2y + 5z^4?$$

$$f(x, y, z) = x^3y^2 + 2xy + 5yz^4?$$

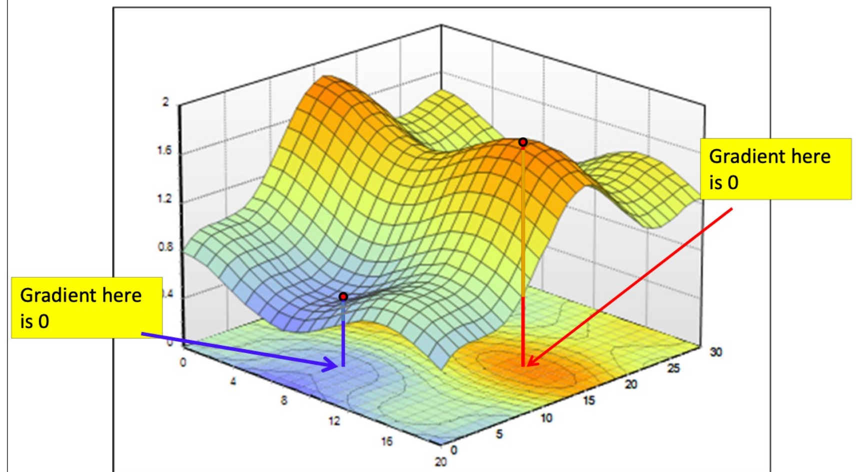


The gradient of the function $f(x, y) = -(\cos^2 x + \cos^2 y)^2$ depicted as a projected vector field on the bottom plane.

https://en.wikipedia.org/wiki/Gradient



From 11-785 Introduction to Deep Learning at CMU



From 11-785 Introduction to Deep Learning at CMU

The Hessian

$$\nabla^2_{\mathbf{x}} f(\mathbf{x}) = \begin{bmatrix} \frac{\partial^2 f(\mathbf{x})}{\partial x_1 \partial x_1} & \frac{\partial^2 f(\mathbf{x})}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f(\mathbf{x})}{\partial x_1 \partial x_d} \\ \frac{\partial^2 f(\mathbf{x})}{\partial x_2 \partial x_1} & \frac{\partial^2 f(\mathbf{x})}{\partial x_2 \partial x_2} & \cdots & \frac{\partial^2 f(\mathbf{x})}{\partial x_2 \partial x_d} \\ \vdots & \vdots & \cdots & \vdots \\ \frac{\partial^2 f(\mathbf{x})}{\partial x_d \partial x_1} & \frac{\partial^2 f(\mathbf{x})}{\partial x_d \partial x_2} & \cdots & \frac{\partial^2 f(\mathbf{x})}{\partial x_d \partial x_d} \end{bmatrix}$$

Gradient descent

Unconstrained minimization

- Given a multivariate function $f(\mathbf{x})$:
  - ✓ calculate the gradient $\nabla_{\mathbf{x}} f(\mathbf{x})$ and solve for $\mathbf{x}$ where $\nabla_{\mathbf{x}} f(\mathbf{x}) = 0$
  - ✓ compute the Hessian at solution $\mathbf{x}^*$
    - ✓ if **positive definite** (positive eigenvalues) then it is a **local minima**
    - ✓ if **negative definite** (negative eigenvalues) then it is a **local maxima**
- Example
  - ✓ $f(x, y, z) = x^3 + 3x^2y - yz^3 + z^2$

From Linear Regression

- Minimize loss using a **closed-form** solution
  - ✓ setting the derivative of the loss to 0, then solving for $\mathbf{w}$

$$\arg\min_{\mathbf{w}} \frac{1}{n}\|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2 \qquad \mathbf{w}^* = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

- Issues?
  - ✓ what happens if we change the loss function?
  - ✓ what if data has high-dimensionality?
  - ✓ what if L1 regularization is used?

Alternative solutions

- Iterative methods
  - ✓ apply an update rule iteratively until finding the solution (or approximating)

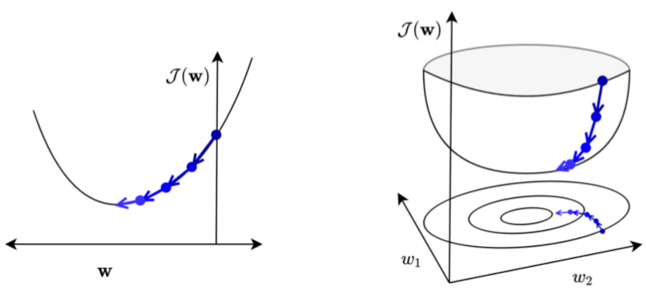
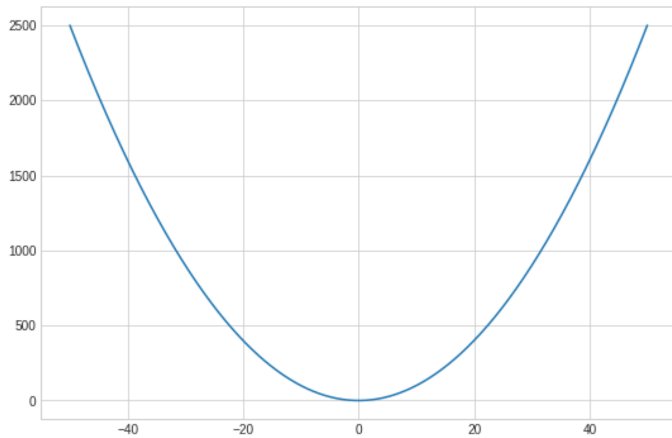


Figure from https://www.cs.toronto.edu/~rgrosse/courses/csc311_f21/lectures/lec03.pdf

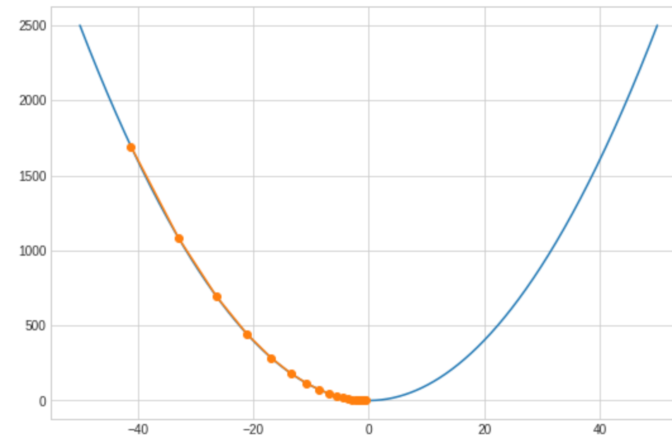
Gradient descent

- Optimization technique used to find the value of x where $f(x)$ is **minimum**
  - ✓ guess a starting point
  - ✓ walk iteratively (taking steps) in the **opposite direction** of the function's **gradient**
- Alternatively, to find the maximum, walk in the direction of the gradient (**gradient ascent**)
- Step size (**learning rate**) is critical
  - ✓ hyperparameter

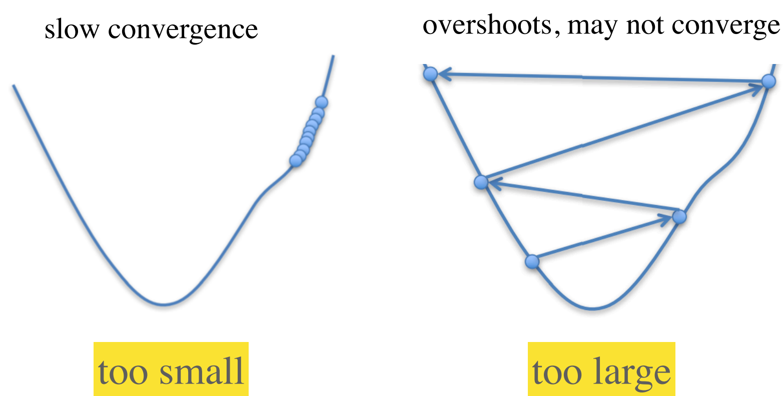
Example with $y = f(x) = x^2$



Example with $y = f(x) = x^2$



The learning rate (step size)



https://www.cc.gatech.edu/~bboots3/CS4641-Fall2018/Lecture3/03_LinearRegression.pdf

Show me the code

```
# define a function
f = lambda x: x ** 2
# define the derivative
df = lambda x: 2 * x

# generate some data
x = np.linspace(-50, 50, 1000)

# apply gradient descent
curr = # initialize with any random value
for i in range(n_steps):
    curr = curr - (l_rate * df(curr))
```

Playground

https://uclaacm.github.io/gradient-descent-visualiser/

Colab

https://colab.research.google.com/drive/1ieijvZI7iK66uzEHbNRTh-p4CmORpGpc