Assignment #6

Due: 12:00 noon November 2, 2020

Upload at: https://www.gradescope.com/courses/135869/assignments/774900

Assignments in COS 302 should be done individually. See the course syllabus for the collaboration policy.

Remember to append your Colab PDF as explained in the first homework, with all outputs visible.
When you print to PDF it may be helpful to scale at 95% or so to get everything on the page.

Problem 1 (10pts)

Consider the following scalar-valued function

$$f(x, y, z) = x^2 y + \sin(z + 6y),$$

where $x, y, z \in \mathbb{R}$.

(A) Compute partial derivatives with respect to x, y, and z.

(B) We can consider f to take a vector $\boldsymbol{\theta} \in \mathbb{R}^3$ as input where $\boldsymbol{\theta} = [x, y, z]^T$. Show the gradient $\nabla_{\boldsymbol{\theta}} f$ as a vector and evaluate it at $\boldsymbol{\theta} = [3, \frac{\pi}{2}, 0]^T$.

Problem 2 (10pts)

The purpose of this problem is to demonstrate Clairaut's Theorem, which states that in general, the order in which you partial differentiate does not matter. Consider the following scalar-valued function,

$$f(x, y, z) = x \sin(xy),$$

where $x, y \in \mathbb{R}$.

(A) Compute $\frac{\partial}{\partial x}\frac{\partial}{\partial y} f(x, y)$. This means we first compute the partial derivative of f with respect to y, then compute the partial derivative of the resulting function with respect to x. This is sometimes denoted $\partial_{xy} f$.

(B) Compute $\frac{\partial}{\partial y}\frac{\partial}{\partial x} f(x, y)$.

You should have gotten the same answer for parts (A) and (B), which demonstrates Clairaut's Theorem. This holds more generally for n variables as well, that is, if $f(x_1, x_2, \ldots, x_n)$ is a function of n variables. This result can be useful when differentiating functions, since it's possible that it's more convenient computationally to differentiate in a particular order.

Problem 3 (24pts)

Consider the following vector function from $\mathbb{R}^3$ to $\mathbb{R}^3$:

$$f(\boldsymbol{x}) = \begin{bmatrix} \sin(x_1 x_2 x_3) \\ \cos(x_2 + x_3) \\ \exp\{-\frac{1}{2}(x_3^2)\} \end{bmatrix}$$

(A) What is the Jacobian matrix of $f(\boldsymbol{x})$?

(B) Write the determinant of this Jacobian matrix as a function of $\boldsymbol{x}$.

(C) Is the Jacobian a full rank matrix for all $\boldsymbol{x} \in \mathbb{R}^3$? Explain your reasoning.

Problem 4 (10pts)

Compute the gradients for the following expressions. (You can use identities, but show your work.)

(A) $\nabla_{\boldsymbol{x}} \frac{1}{2}(\boldsymbol{x}-\mu)^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\mu)$ Assume $\boldsymbol{x}, \mu \in \mathbb{R}^n$ and invertible symmetric $\boldsymbol{\Sigma} \in \mathbb{R}^{n \times n}$.

(B) $\nabla_{\boldsymbol{x}}(\boldsymbol{c}+\boldsymbol{A}\boldsymbol{x})^T(\boldsymbol{c}-\boldsymbol{B}\boldsymbol{x})$ Assume $\boldsymbol{x} \in \mathbb{R}^n$, $\boldsymbol{c} \in \mathbb{R}^m$ and $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{m \times n}$.

Problem 5 (12pts)

(A) The sigmoid function $f : \mathbb{R} \to \mathbb{R}$ (also called the *logistic function*) is defined to be:

$$f(z) = \frac{1}{1 + e^{-z}} \tag{1}$$

Compute the derivative of the sigmoid function, i.e., $f'(z)$. And verify that $f'(z) = f(z)\,(1 - f(z))$

(B) The cost function of a very popular machine learning model logistic regression has the following form:

$$c(\boldsymbol{\theta}, \boldsymbol{x}, y) = -y \log\left(\frac{1}{1 + e^{-\boldsymbol{\theta}^\top \boldsymbol{x}}}\right) - (1 - y) \log\left(1 - \frac{1}{1 + e^{-\boldsymbol{\theta}^\top \boldsymbol{x}}}\right) \tag{2}$$

with $\boldsymbol{\theta} \in \mathbb{R}^d, \boldsymbol{x} \in \mathbb{R}^d, y \in \mathbb{R}$. Compute the partial derivative with regards to θ, i.e. $\frac{\partial c(\theta, x, y)}{\partial \theta}$. And verify that $\frac{\partial c(\theta, x, y)}{\partial \theta} = (f(\boldsymbol{\theta}^\top \boldsymbol{x}) - y)\,\boldsymbol{x}^\top$.

Problem 6 (32pts)

In gradient descent, we attempt to minimize some function $f(\boldsymbol{x})$ by altering parameters $\boldsymbol{x} \in \mathbb{R}^n$ according to the following formula:

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t - \lambda(\nabla_{\boldsymbol{x}} f(\boldsymbol{x}_t))^T$$

for some small $\lambda \geq 0$ known as the *learning rate* or *step size*. We adjust $\boldsymbol{x}$ so as to move in a direction proportional to the negative gradient. We will discuss gradient descent in more detail later in the course.

Consider the simple function $f(\boldsymbol{x}) = \boldsymbol{x}^T A \boldsymbol{x}$ for a constant matrix $A \in \mathbb{R}^{n \times n}$.

(A) Implement a function `f(A, x)` that takes as input an $n \times n$ numpy array `A` and a 1D array `x` of length n and returns the value of f as defined above.

(B) Implement a function `grad_f(A, x)` that takes the same two arguments as above but returns $\nabla_{\boldsymbol{x}} f(\boldsymbol{x})$ evaluated at `x`.

(C) Now implement a third and final function `grad_descent(A, x0, lr, num_iters)` that takes the additional arguments `lr`, representing the learning rate λ above, and `num_iters` indicating the total number of iterations of gradient descent to perform. The function should print the values of $\boldsymbol{x}_t$ and $f(\boldsymbol{x}_t)$ after each iteration of gradient descent.

(D) We will perform gradient descent on f with $A = \begin{bmatrix} 1 & 0 \\ 0 & 4 \end{bmatrix}$. Set the initial $\boldsymbol{x}_0$ to be `np.array([10 10])`. Run gradient descent for 50 iterations with learning rates of 1, 0.25, 0.1, and 0.01. What do you notice? Does $\boldsymbol{x}_t$ always converge to the same value? Does our gradient descent algorithm work every time?

Problem 7 (2pts)

Approximately how many hours did this assignment take you to complete?

My notebook URL: `https://colab.research.google.com/XXXXXXXXXXXXXXXXXXXXXXX`

Changelog

- 19 October 2020 – Updated for Fall, 2020.
- 23 March 2020 – Initial version.
- 24 March 2020 – Fixed typo in problem 3.
- 24 March 2020 – Fixed second typo in problem 3.
- 26 March 2020 – Clarifications for problem 3.