MATH-UA 253/MA-UY 3204 - Fall 2022 - Homework #1

Problem 1 (computing the gradient and Hessian of the linear least squares cost function using two different methods). A *quadratic form* is a scalar-valued function $f : \mathbb{R}^n \to \mathbb{R}$ of the form:

$$f(x) = x^\top A x + b^\top x + c, \tag{1}$$

where $x, b \in \mathbb{R}^n$, $A \in \mathbb{R}^{n\times n}$, and $c \in \mathbb{R}$.

(a) Rewrite the squared ℓ_2 norm $\|Cy + d\|_2^2$, where $C \in \mathbb{R}^{m\times n}, y \in \mathbb{R}^n$, and $d \in \mathbb{R}^m$, as a quadratic form $g(y)$. What is the dimension of the domain of g?

(b) Write $g(y)$ out using summations instead of matrix notation. Using this form of g, compute the first and second partials of g: $\partial g/\partial y_i$ and $\partial^2 g/\partial y_i \partial y_j$ for each i and j.

(c) Now, what are the gradient and Hessian of g? Simplify these expressions using matrix notation.

(d) Recall that—in general—the Taylor expansion of a function g about y with base point h is given by:

$$g(y + h) = g(y) + \nabla g(y)^\top h + \frac{1}{2} h^\top \nabla^2 g(y) h + O(\|h\|_2^3), \tag{2}$$

where $\nabla g(y)$ denotes the gradient of g at y and $\nabla^2 g(y)$ denotes its Hessian. Evaluate the quadratic form from part (a) at $y + h$. Simplify the resulting expression by collecting terms according to their power of h. You should get a constant term (of the form "$D(y)$"), a linear term (of the form "$E(y)h$"), and a quadratic term (of the form "$h^\top F(y)h$"). Once you have determined $E(y)$ and $F(y)$, match these expressions with the linear and quadratic terms of (2) in order to find $\nabla g(y)$ and $\nabla^2 g(y)$ indirectly.

Problem 2 (experimenting with gradient descent). You now should know that the gradient is the direction of *steepest ascent* at a point. That is, for a scalar-valued function $f : \mathbb{R}^n \to \mathbb{R}$, $\nabla f(x)$ is the vector which points in the direction of fastest increase at x, along with the rate of change in f. This gives us an idea for a simple iterative method for minimizing functions. With an iterative method for optimization (really, the main focus of this course!), the idea is to generate a sequence of vectors $x_0, x_1, x_2, \ldots$ such that $f(x_n) \to f(x^*) = \min_x f(x)$ as $n \to \infty$. The idea of *gradient descent* is simple: let x_0 be an initial guess for x^* (it could be completely arbitrary). For each n, we set:

$$x_{n+1} = x_n - \nabla f(x_n). \tag{3}$$

That is, we take a *step* in the direction of steepest *decrease* with magnitude $|\nabla f(x_n)|$. We learn about more sophisticated variations on this idea during the course. In the mean time, we will begin to experiment with gradient descent in this problem.

Consider the cost function:

$$f(x, y) = (1 - \cos(\pi x/2))y^2, \qquad (x, y) \in B = [-1, 1] \times [-1, 1]. \tag{4}$$

(a) This function does *not* have a unique minimizer over B. What is its set of minimizers over B, and what value of f is obtained there?

(b) Use Python with numpy and matplotlib to make a `contourf` plot of f over B. You may find the `np.meshgrid` and `np.linspace` functions helpful.

(c) Generate random initial iterates $r_0 = (x_0, y_0)$ in B using `np.random.uniform` and take 10 gradient descent steps. Use `plot` to plot each of these *trajectories* as piecewise linear paths *on top of your `contourf` plot*. Make sure set `marker='.'` and `linewidth=1` (or smaller) to make it easy to see where each iterate r_n lies on the trajectory.

(d) Take a look at several of these trajectories—give an explanation for any patterns you might observe.

(e) Generate a large number of these trajectories (1000 of them, for example), and plot them all simultaneously. What do you notice?