Problem Set 1

CS 7301

Due: 9/29/2021 by 11:59pm

Note: all answers should be accompanied by explanations and code for full credit. Late homeworks will not be accepted.

Problem 1: Numerical Gradients (30 pts)

Sometimes a function may be differentiable, but its gradients may be difficult to compute or the gradients are unavailable. In such cases, numerical methods can be used to approximate the derivative. One such strategy is to use the the so-called symmetric difference quotient,

$$\frac{f(x+\epsilon v) - f(x-\epsilon v)}{2\epsilon},$$

with a small $\epsilon > 0$ to approximate the directional derivative in the direction v.
Consider the following optimization problem: Given $A \in \mathbb{R}^{n\times n}$ and $b \in \mathbb{R}^n$,

$$\min_{x\in\mathbb{R}^n} ||Ax - b||_2^2$$

1. Is this a convex optimization problem? Explain.
2. Use gradient descent with a fixed step size to find the global minimum.
3. Use gradient descent with a diminishing step size to find the global minimum.
4. Use gradient descent with a fixed step size and numerical approximations of the derivatives to find the global minimum.
5. How would you compare the performance of the different approaches for this optimization problem?

Problem 2: Projections onto Convex Hulls (30 pts)

For this problem, you will implement the Frank-Wolfe algorithm for projection onto the convex hull of a finite point set. Recall that, given $x^{(1)}, \ldots, x^{(M)} \in \mathbb{R}^n$, their convex hull is the set of all x that can be written as a convex combination of these points.

1. Use the Frank-Wolfe algorithm to compute the projection of a query point q onto the convex hull of the given $x^{(1)}, \ldots, x^{(M)} \in \mathbb{R}^n$. That is, you should solve the following optimization problem.

$$\min_{\lambda \in \mathbb{R}^M} \frac{1}{2} ||q - \sum_{m=1}^{M} \lambda_m x^{(m)}||_2^2$$

   such that

$$\sum_{m=1}^{M} \lambda_m = 1$$

$$\lambda \geq 0$$

   Your Python function should take as input the x's, the query point q, an initial value for λ that satisfies the constraints, and the tolerance ϵ for the Frank-Wolfe convergence condition and return the best function value found during the iterative procedure.

2. Use the method of Lagrange multipliers to construct a dual of this optimization problem.

Problem 3: Convex Envelopes (40 pts)

Consider a collection of points $x^{(1)}, \ldots, x^{(M)} \in \mathbb{R}^n$ with corresponding function values $y^{(1)}, \ldots, y^{(M)} \in \mathbb{R}$. The convex envelope of these points is the convex function $f_{env} : \mathbb{R}^n \to \mathbb{R}$ such that $f_{env}(x^{(m)}) \leq y^{(m)}$ for all m and for any other convex function $g : \mathbb{R}^n \to \mathbb{R}$ such that $g(x^{(m)}) \leq y^{(m)}$ for all m, $f_{env}(x) \geq g(x)$ for all $x \in \mathbb{R}^n$.

1. For a finite point set, is the convex envelope differentiable? Explain.

2. Recall that, for any convex function $f : \mathbb{R}^n \to \mathbb{R}$, and any $x, x' \in \mathbb{R}$, $f(x) \geq f(x') + w^T(x - x')$, where w is a subgradient of f at x'. Using this, we can formulate the problem of evaluating the convex envelope of our collection of points at a query point $x \in \mathbb{R}^n$ as a convex optimization problem:

$$f_{env}(x) = \sup_{w \in \mathbb{R}^n, y \in \mathbb{R}} y$$

   such that

$$y^{(m)} \geq y + w^T(x^{(m)} - x), \text{ for all } m \in \{1, \ldots, M\}.$$

   (a) Explain why this optimization problem is unbounded for certain choices of $x \in \mathbb{R}^n$.

   (b) To fix the unboundedness, we can add an additional constraint that $||w||_2^2 \leq \gamma^2$ for some given $\gamma \geq 0$. In Python, implement projected gradient descent to solve the optimization problem under this additional constraint. Your Python function should take as input the x's, the y's, γ, the query point x, and the number of iterations of projected gradient ascent to perform and return the best function value found during the iterative procedure stating from $w = 0$ and $y = \min_m y^{(m)}$. Hint: you can do the projection analytically if you reformulate the optimization problem to eliminate the linear constraints.

   (c) Construct a dual of the optimization problem in (b) using the method of Lagrange multipliers.

(d) In Python, implement the Frank-Wolfe algorithm to maximize your dual in (c). Your Python function should take as input the x's, the y's, γ, the query point x, a feasible initial point for the Lagrange multipliers, a tolerance ϵ that terminates the Frank-Wolfe algorithm whenever the convergence criteria from class is met, and an upper bound `max_it` on the number of iterations, and returns the best function value found during the iterative procedure. Hint: you can analytically eliminate the Lagrange multiplier corresponding to the constraint $||w||_2^2 \leq \gamma^2$.