Optimization for Learning - FRTN50
Assignment 1

Written by: Martin Morin

Latest update: October 18, 2019

Introduction

The goal of this hand-in is to become familiar with some of the steps involved in solving an optimization problem. This include forming Fenchel-dual problems and finding gradients and/or proximal operators. Most of these steps will not need to be repeated every time a new problem is encountered and one can often rely on previous work by you or others. However, it is still important to be aware of and understand them.

Problem The problem we will solve is the following constrained problem

$$\min_{x\in S} f(x) = \min_{x} f(x) + \iota_S(x)$$

where ι_S is the indicator function for the set S. In this case we let f be a quadratic, $f(x) = \frac{1}{2}x^TQx + q^Tx$ where $Q \succ 0$, and S be the box, $S = \{x : \forall i, a_i \leq x_i \leq b_i\}$ where a_i and b_i are the lower and upper bounds on the i:th coordinate of x.

Solution Method To solve the optimization problem we will use the *proximal gradient* method. Proximal gradient solves problems of the form

$$\min_{x} \phi(x) + h(x) \tag{1}$$

where ϕ is differentiable and h is proximable, i.e. prox_h can be cheaply computed. Proximal gradient is a method that, starting from some arbitrary initial guess x^0, iteratively performs the following update

$$x^{k+1} = \text{prox}_{\gamma h}(x^k - \gamma\nabla\phi(x^k)) \tag{2}$$

until x^k is deemed to have converged. In this hand-in we simply run it a large fixed number of iterations and plot the norm of the step-length/residual, $\|x^{k+1} - x^k\|$, of each step to make sure it converge to zero.

The iteration (2) has a positive step-size parameter γ that will affect the convergence. It should be tuned to the problem or chosen based on the properties of ϕ and h. In the closed convex case with ϕ being L-smooth, i.e. $\nabla\phi$ is L-Lipschitz continuous, the maximal step-size to guarantee convergence is $\gamma < \frac{2}{L}$. The proximal gradient method is one of the fundamental building blocks of modern optimization methods and will be covered in more detail later in the course.

Assignment

Solve the following tasks.

Task 1 Derive f^* and ι_S^* and write down the Fenchel-dual problem.

Task 2 Show that f and f^* are L-, and L^*-smooth respectively. Find L and L^*.

Task 3 Derive expressions for ∇f, ∇f^*, $\text{prox}_{\gamma\iota_S}$ and $\text{prox}_{\gamma\iota_S^*}$.

Task 4 Let $y^\star$ be a solution to the dual problem, derive an expression that gives a solution to the primal problem given $y^\star$.

Task 5 The file `functions.jl` contains empty JULIA-functions for evaluating the functions, gradients, proximal operators, and primal solution from the previous tasks. Use your results to fill in the functions.

Task 6 The file `problem.jl` contains a function for generating Q, q, a, and b that define the quadratic f and the box constraint set S. Use Task 5 to solve the primal problem using the proximal gradient method.

Try a range of different step-sizes. What seems to be the best choice? Does the upper bound $\gamma < \frac{2}{L}$ seem reasonable?

Test different initial points for the algorithm, does this affect the solution the algorithm converge to? Reason about why/why not it affects the solution? Does your solution satisfy the constraint $x^\star \in S$? What about the iterates, do they always satisfy the constraint, $x^k \in S$? Why/why not?

Task 7 Solve the dual problem. Similar to the previous task, find/verify the upper bound on the step-size and find a good step-size choice.

Compare the solutions from the primal and the one extracted from the dual, are they the same? Do they satisfy the constraint $x^\star \in S$? Let y^k be the iterates of the dual method, using the expression from Task 4, extract the primal iterates $\hat{x}^k$ from y^k. Does $\hat{x}^k$ always satisfy the constraint $\hat{x}^k \in S$?

How does the function values, $f(\hat{x}^k)$, develop over the iterations? What about $f(\hat{x}^k) + \iota_S(\hat{x}^k)$?

Submission

See the latest version of the course program for instruction on how to submit the assignment. Your submission should contain the following files.

- Your filled in version of `functions.jl` from Task 5.
- Your code that solves the problems from Task 6 and 7.
- A single pdf containing the following:
  - Your derivations and final expressions from Task 1-4.

– A couple of paragraphs explaining and commenting on your findings from Task 6 and 7. Answer the questions raised in the tasks and motivate them with plots whenever possible.