Homework 3

Convex Optimization 10-725/36-725

Due Thursday March 5 at 4:00pm
submitted to Mallory Deptola in GHC 8001
(make sure to submit each problem separately)

1 Conjugate Fundamentals [Veeru]

Recall that for any function f, convex or not, we can define its conjugate f^* as

$$f^*(y) = \max_x \; y^T x - f(x)$$

An immediate property of the conjugate is that $f(x) + f^*(y) \geq x^T y$ for all x, y. This will be helpful for the coming parts. Assume that f is closed; or for simplicity, just replace this by the assumption that $\mathrm{dom}(f) = \mathbb{R}^n$.

(a) Show that f^{**}, the conjugate of f^*, satisfies $f^{**} \leq f$.

(b) Prove that f^{**} is the pointwise maximum of all affine functions that underestimate f, i.e.,

$$f^{**}(x) = \max\{g(x) : g \text{ is affine}, \; g \leq f\}.$$

(c) Assuming f is convex, show that $f^{**} = f$.

Hint: note that from part (b) it suffices to find, at each x, an affine underestimator g of f such that $g(x) = f(x)$. To find such a function, use the fact that f is convex, and so its epigraph $\mathrm{epi}(f)$ is a convex set. Therefore it has a supporting hyperplane at $(x, f(x))$: there is some $(a, b) \neq 0$ such that

$$a^T x + bf(x) \leq a^T z + bt,$$

for all $(z, t) \in \mathrm{epi}(f)$. Use this to find the desired affine underestimator with $g(x) = f(x)$.

(d) Again assuming that f is convex, show that

$$x \in \partial f^*(y) \iff y \in \partial f(x).$$

Hint: one direction is straightforward, recalling the max rule for subgradients. For the other direction, apply part (c).

2 Conjugates, Duality, and Proximal Mappings [Veeru]

Let f and g be closed, convex functions.

(a) For a matrix $A \in \mathbb{R}^{m \times n}$, prove that the dual problem of

$$\min_x \; f(x) + g(Ax) \tag{1}$$

is

$$\max_y \ -f^*(-A^T y) - g^*(y). \tag{2}$$

Hint: introduce the auxiliary variable $z = Ax$ in the primal problem.

(b) Assume that f is strictly convex. Prove that this implies f^* is differentiable, and that

$$\nabla f^*(y) = \operatorname*{argmin}_z \ f(z) - y^T z.$$

Hint: use Question 1 part (d).

From now on, assume that f is strictly convex and smooth, and g is not smooth, but we know its proximal operator

$$\text{prox}_{g,t}(x) = \operatorname*{argmin}_z \ \frac{1}{2t}\|x - z\|_2^2 + g(z).$$

Note that this *does not* necessarily mean that we know the proximal operator for $h(x) = g(Ax)$. Therefore we cannot easily apply proximal gradient descent to the primal problem (1). However, as you will show in the next few parts, knowing the the proximal mapping of g *does* lead to the proximal mapping of g^*, which leads to an algorithm on the dual problem (2).

(c) Prove first that

$$\text{prox}_{g,1}(x) + \text{prox}_{g^*,1}(x) = x,$$

for all x. (This is sometimes called the Moreau decomposition.) Note the specification $t = 1$ in the above.

Hint: again, use Question 1 part (d).

(d) Verify that for $t > 0$, we have $(tg)^*(x) = tg^*(x/t)$. Use this, and part (c), to prove that for any $t > 0$,

$$\text{prox}_{g,t}(x) + t \cdot \text{prox}_{g^*,1/t}(x/t) = x,$$

for all x.

Hint: apply part (c) to the function tg. Then note $\text{prox}_{g,t}(x) = \text{prox}_{tg,1}(x)$, and the same for g^*.

(e) Now write down a proximal gradient descent algorithm for the dual problem (2). Use parts (b) and (d) of this question to express all quantities in terms of f and g. That is, your proximal gradient descent updates should not have any appearences of ∇f^* and $\text{prox}_{g^*,t}(\cdot)$.

3 Practice with Conjugates [Junier]

Below we go through an example of the use of conjugates for deriving to the dual of an optimization problem. Please specify the domain of the conjugate function, where appropriate.

(a) Derive the conjugate function of $f(x) = \log(1 + e^{-x})$.

(b) Derive the conjugate function of $f(\beta) = \sum_{i=1}^n \log(1 + e^{-y_i \beta_i})$, where $y_i \in \{-1, 1\}$.

Note: if $z_i \in \{0, 1\}$, and we let $y_i = 2z_i - 1 \in \{-1, 1\}$, for each $i = 1, \ldots n$ (this is just a relabeling of the classes), then minimizing $f(\beta)$ is the same as minimizing $\ell(\beta) = \sum_{i=1}^n (-z_i \beta_i + \log(1 + e^{\beta_i}))$. The latter may look more familiar, i.e., it is the logistic loss with binary outcomes.

(c) Prove that the dual of

$$\min_{\beta} \sum_{i=1}^{n} \log(1 + e^{-y_i \beta_i}) + \lambda \|D\beta\|_1,$$

where $D \in \mathbb{R}^{m \times n}$ is an arbitrary matrix and $y_i \in \{-1, 1\}$, is

$$\begin{aligned} \min_{u} \quad & \sum_{i=1}^{n} y_i (D^T u)_i \log(y_i (D^T u)_i) + (1 - y_i (D^T u)_i) \log(1 - y_i (D^T u)_i) \\ \text{subject to} \quad & 0 \le y_i (D^T u)_i \le 1, \quad i = 1, \dots n, \quad \|u\|_\infty \le \lambda. \end{aligned}$$

Hint: use Q2 part (a).

(d) For the pair of primal and dual problems in part (c), write down the primal solution β in terms of dual solution u. Write down u in terms of β, or explain why an analytical solution is unavailable.

4 Binary Image Denoising [Nicole]

Often when an image is compressed (e.g. via JPEG compression), noise can be introduced by the method of compression. This is particularly problematic in the case of binary images.

One method for denoising the images is to minimize the following objective, as in Q3 part (c):

$$\min_{\beta} \sum_{i=1}^{n} \log(1 + e^{-y_i \beta_i}) + \lambda \|D\beta\|_1.$$

Here y is the image to be denoised, with each component corresponding to a pixel, i.e., $y \in \mathbb{R}^{n_1 n_2}$ is an unraveled version of an $n_1 \times n_2$ image, and each component of β corresponds to the probability that a given pixel in the image is equal to 1. The matrix D chosen so that

$$\|D\beta\|_1 = \sum_{i \sim j} |\beta_i - \beta_j|$$

where $i \sim j$ means that pixels i and j are either horizontally or vertically, adjacent.

The rationale behind this convex program is that we observe pixel values y_i on an image that are either +1 with probability p_i or -1 with probability $1 - p_i$. If we believe that the underlying probabilities are piecewise constant across the image, i.e., a pixel will resemble its neighbors, then the above program tries to "denoise" the observe binary values by fitting estimated probabilities β_i that are piecewise continuous.

Unfortunately, it is not easy to solve the primal problem directly. We can almost solve the dual formulation, that you derived in Q3 part (c):

$$\begin{aligned} \min_{u} \quad & \sum_{i=1}^{n} y_i (D^T u)_i \log(y_i (D^T u)_i) + (1 - y_i (D^T u)_i) \log(1 - y_i (D^T u)_i) \\ \text{subject to} \quad & 0 \le y_i (D^T u)_i \le 1, \quad i = 1, \dots n, \quad \|u\|_\infty \le \lambda, \end{aligned}$$

but the trouble is that the constraints are not easy to deal with. E.g., in projected gradient descent, we would have to project onto $\{u : 0 \le y_i (D^T u)_i \le 1,\ i = 1, \dots n,\ \|u\|_\infty \le \lambda\}$ at each iteration, which is hard.

To get around this, we will "lift" the constraints into the criterion, by instead solving

$$\min_u \quad \sum_{i=1}^n y_i (D^T u)_i \log(y_i (D^T u)_i) + (1 - y_i (D^T u)_i) \log(1 - y_i (D^T u)_i) \\ - t \cdot \sum_{i=1}^n \Big(\log(y_i (D^T u)_i) + \log(1 - y_i (D^T u)_i) \Big) - t \cdot \sum_{i=1}^n \Big(\log(\lambda - u_i) + \log(u_i + \lambda)) \Big)$$

Here t is a small but positive constant, and so when the constraints are close to being violated, the second half of the criterion function (the part we added) approaches infinity. Choosing a small t may seem like a hack, but we will learn a more principled way of doing this later when we talk about the log barrier method.

(a) What makes it difficult to solve the primal formulation with the techniques we have learned so far?

(b) If y is an unraveled version of an $n_1 \times n_2$ image in column-major order, then what dimensions should D have? Describe its entries.

(c) Derive the gradient of the dual objective. How may flops does it take to compute this gradient? (Hint: what is the key property of D that you are using?)

(d) Implement gradient descent to solve the modified dual problem given above on the image y in `img.mat` and `img.RData`, found on the class website. The file also contains the constants t and λ, as well as a feasible initial starting point, u_0. You will have to construct D yourself, as in part (c) above. You can choose the step sizes in whatever manner you find appropriate (e.g., backtracking), and you can choose to use acceleration, or not. Plot the value of the objective function versus the number of iterations.

Are the iterations here cheap? (Hint: did you store D in an appropriate format?)

(e) Reconstruct the primal solution β from the dual solution u, using Q3 part (d). Now compute the estimated probabilities across pixels, according to

$$p_i = 1/(1 + e^{-\beta_i}), \quad i = 1, \ldots n_1 n_2.$$

Produce a binary image, according to whether the estimated probabilities are greater than or less than 0.5. Plot this image.

(f) Bonus: who is this a picture of?