EE364b Spring 2023 Homework 5

Due Sunday 5/14 at 11:59pm via Gradescope

5.1 *(11 points) So you think ADMM is fast, huh?*

Consider a regression problem with a data matrix $X \in \mathbb{R}^{n\times(p+1)})$, where each column represents a predictor. Suppose that the matrix X is split into J groups over its columns:

$$X = [\vec{1}\ X_{(1)}\ X_{(2)} \ldots\ X_{(J)}]$$

where $\vec{1} = [1\ 1 \ldots 1] \in \mathbb{R}^n$ is a vector of all ones. The groups are typically determined by the types of predictors. To achieve sparsity over the groups rather than individual predictors, we may write $\beta = (\beta_0, \beta_{(1)}, \ldots, \beta_{(J)})$, where β_0 is an intercept term and each $\beta_{(j)}$ is an appropriate coefficient block of β corresponding to $X_{(j)}$, and solve the regularized optimization problem:

$$\min_{\beta\in\mathbb{R}^{p+1}} f(\beta) + h(\beta).$$

Here $h(\beta)$ is a convex regularization term to promote the sparsity over groups. In this problem, we will use group Lasso to predict the Parkinson's disease (PD) symptom score on the Parkinsons dataset[1]. The PD symptom score is measured on the unified Parkinson's disease rating scale (UPDRS). This data contains 5, 785 observations, 18 predictors (provided in **X_train.csv**), and an outcome - the total UPDRS (provided in **y_train.csv**)

The 18 columns in the predictor matrix have the following groupings (in column ordering):

- age: Subject age in years
- sex: Subject gender, 0–male, 1–female
- Jitter(%), Jitter(Abs), Jitter:RAP, Jitter:PPQ5, Jitter:DDP: Several measures of variation in fundamental frequency of voice
- Shimmer, Shimmer(dB), Shimmer:APQ3, Shimmer:APQ5, Shimmer:APQ11, Shimmer:DDA: Several measures of variation in amplitude of voice
- NHR, HNR: Two measures of ratio of noise to tonal components in the voice
- RPDE: A nonlinear dynamical complexity measure
- DFA: Signal fractal scaling exponent
- PPE: A nonlinear measure of fundamental frequency variation

[1] 'Exploiting Nonlinear Recurrence and Fractal Scaling Properties for Voice Disorder Detection', Little MA, McSharry PE, Roberts SJ, Costello DAE, Moroz IM. BioMedical Engineering Online 2007, 6:23 (26 June 2007)

We consider the group LASSO problem, where $h(\beta) = \lambda \sum_j w_j \|\beta_{(j)}\|_2$:

$$\min_{\beta \in \mathbb{R}^{p+1}} \frac{1}{2n} \|X\beta - y\|_2^2 + \lambda \sum_j w_j \|\beta_{(j)}\|_2 \tag{1}$$

A typical choice for weights on groups w_j is $\sqrt{p_j}$, where p_j is number of predictors that belong to the jth group, to account for the group sizes. We will solve the problem using both ADMM and proximal gradient descent method.

(a) (1 point) Derive the proximal operator for the convex function $h(z) = \lambda \sum_j w_j \|z_{(j)}\|_2$.

(b) (1 point) Derive the ADMM updates for Eq. (1) and Eq. (2) as described in the lecture slides (page 17, link).
Hint: You can let $h(\alpha) = \lambda \sum_j w_j \|\alpha_{(j)}\|_2$, and rewrite the original objective as $f(z) + h(\alpha)$ with the consensus constraint $\alpha = z$.

(c) (1 point) Implement ADMM to solve the least squares group lasso problem on the Parkinsons dataset. You may use the implementation template we have provided you in `hw5_q1_template.py`. Set $\lambda = 0.02$.

(d) (1 point) Derive the proximal gradient method updates for Eq. (1) as described in the lecture slides (page 20, link).

(e) (1 point) Implement proximal gradient descent to solve least squares group lasso problem on the Parkinsons dataset. You may start from the code template we have provided. Do not implement line-search, acceleration, or restarts as part of this question. Set $\lambda = 0.02$, use a fixed step-size $t = 0.5$, and initialize at $\beta^0 = 0$.

(f) (1 point) Plot $f^k - f^*$ versus k for the first 10000 iterations on a semi-log scale for both methods for the training data, where f^k denotes the objective value at step k, and the optimal objective value is $f^* = 49.9649$. Print the components of the solutions numerically. Which groups are selected, i.e., non-zero at the solution?

(g) (1 point) ADMM is much faster than naive proximal-gradient descent, partially because a fixed step-size is sub-optimal. Implement the following line-search condition [BT09]:

$$f(\beta^{k+1}) \leq f(\beta^k) + \langle \nabla f(\beta^k), \beta^{k+1} - \beta^k \rangle + \frac{1}{2t^k} \left\| \beta^{k+1} - \beta^k \right\|_2^2 .$$

If this condition holds, we accept step-size t^k. If it fails, reduce the step-size as $t^k \leftarrow t^k \cdot \gamma$, where $\gamma = 0.8$ is a backtracking parameter, and try the proximal-gradient update again.

We also want a way to increase the step-size across iterations. This allows the method to adapt to local smoothness of the objective. After every successful iteration, initialize the step-size for the next iteration $t^{k+1} \leftarrow t^k/\gamma$. Initialize the first step-size at $t^0 = 10$ and add PGD with line-search (PGD-LS) to the comparison figure.

(h) (1 point) Augment PGD-LS with acceleration to obtain the FISTA algorithm [BT09]. In particular, modify the update to the following:

$$\beta^{k+1} = \text{Prox}_h(v^k - t^k \nabla f(v^k))$$
$$\xi^{k+1} = 1 + \frac{1}{2}(1 + 4(\xi^k)^2)^{1/2}$$
$$v^{k+1} = \beta^{k+1} + \frac{\xi^k - 1}{\xi^{k+1}}(\beta^{k+1} - \beta^k),$$

where $v^0 = \beta^0$ and $\xi^0 = 1$. Note that setting $v^{k+1} = \beta^{k+1}$ reduces to regular proximal-gradient descent.

Hint: The line-search condition should be evaluated at β^{k+1} and v^k, since it applies only to the proximal-gradient step and cannot take into account the extrapolation. We will handle the extrapolation in the next part.

Hint 2: You should observe a periodic change in the training objective for FISTA. This is characteristic of the method and not a bug.

(i) (1 point) We can do even better by adding restarts to FISTA. Restarts reset $v^{k+1} = \beta^{k+1}$ and $\xi^{k+1} = 1$ to adapt to local curvature of the objective. Perform a restart whenever

$$\langle \beta^{k+1} - \beta^k, v^k - \beta^{k+1} \rangle \geq 0.$$

The vector $v^k - \beta^{k+1}$ is called the proximal gradient mapping and has similar properties to the negative gradient; this rule checks whether the accelerated update is a descent direction with respect to the proximal gradient mapping. Implement restarts and add restarted FISTA (R-FISTA) to your comparison.

(j) (1 point) The PCA whitening of $X_{(i)}$ is given by

$$\tilde{X}_{(i)} = X_{(i)} U \Lambda^{-1/2},$$

where $U \Lambda U^\top = X_{(i)}^\top X_{(i)}$ is the eigendecomposition of $X_{(i)}^\top X_{(i)}$. Show $\tilde{X}_{(i)}^\top \tilde{X}_{(i)} = I$ and that Eq. (1) with PCA whitened data is equivalent to the following problem:

$$\min_{z \in \mathbb{R}^{J \cdot n}} \frac{1}{2n} \Big\| \sum_i z - y \Big\|_2^2 + \lambda \sum_j w_j \|z_{(j)}\|_2 \quad \text{s.t.} \quad z_{(i)} \in \text{Range}(X_{(i)}), i \in [J]. \qquad (2)$$

Describe how to recover the optimal weights $\beta^*_{(i)}$ from the optimal group predictions $z^*_{(i)}$. When is Eq. (2) a more desirable objective to minimize than Eq. (1) and when is it less desirable?

(k) (1 point) Implement PCA whitening and solve the whitened version of Eq. (1) (do **not** to solve Eq. (2)). Use the same regularization parameter as before ($\lambda = 0.02$). How many blocks are active in the final solution? What does this imply about the effects of whitening on regularization? Should we use whitening for the Parkinsons dataset?

5.2 (5 points) *Proximal operators of activation functions.* Consider the convex function $f(x) = \max(x, 0)$ for $x \in \mathbf{R}$. f is the ReLU function, which is a popular activation function for neural networks.

(a) Describe the subdifferential operator $F(x) = \{(x, \partial f(x)) \; : \; x \in \mathbf{R}\}$ and plot its graph.

(b) Find the resolvent operator $(I + \lambda F)^{-1}(x)$ using the graphical method using the graphical approach outlined in the lecture slides and plot its graph.

(c) Find the proximal operator of f directly using its definition and verify that it matches with the resolvent operator in part (b).

(d) Let $\phi(x)$ be a convex function with proximal operator $\mathbf{prox}_{\lambda\phi}(v)$, where $x, v \in \mathbf{R}$, and define $h(x) = \phi(x) + c_1 x + c_2$. Show that $\mathbf{prox}_{\lambda h}(v) = \mathbf{prox}_{\lambda\phi}(v - \lambda c_1)$.

(e) Another popular activation function for neural networks is the Leaky ReLU function, which can be defined as $g_a(x) = \max(ax, x)$ when $0 < a < 1$. Derive the proximal operator of g_a when $0 < a < 1$. *Hint: You may find the results of parts (c) and (d) useful.*

5.3 (7 points) *Solving LPs via alternating projections.* Consider an LP in standard form,

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax = b \\ & x \succeq 0, \end{array}$$

with variable $x \in \mathbf{R}^n$, and where $A \in \mathbf{R}^{m \times n}$. A tuple $(x, \nu, \lambda) \in \mathbf{R}^{2n+m}$ is primal-dual optimal if and only if

$$Ax = b, \quad x \succeq 0, \quad -A^T \nu + \lambda = c, \quad \lambda \succeq 0, \quad c^T x + b^T \nu = 0.$$

These are the KKT optimality conditions of the LP. The last constraint, which states that the duality gap is zero, can be replaced with an equivalent condition, $\lambda^T x = 0$, which is complementary slackness.

(a) (1 point) Let $z = (x, \nu, \lambda)$ denote the primal-dual variable. Express the optimality conditions as $z \in \mathcal{A} \cap \mathcal{C}$, where $\mathcal{A}$ is an affine set, and $\mathcal{C}$ is a simple cone. Give $\mathcal{A}$ as $\mathcal{A} = \{z \mid Fz = g\}$, for appropriate F and g.

(b) (1 point) Explain how to compute the Euclidean projections onto $\mathcal{A}$ and also onto $\mathcal{C}$.

(c) (2 points) Implement alternating projections to solve the standard form LP. Use $z^{k+1/2}$ to denote the iterate after projection onto $\mathcal{A}$, and z^{k+1} to denote the iterate after projection onto $\mathcal{C}$. Your implementation should exploit factorization caching in the projection onto $\mathcal{A}$, but you don't need to worry about exploiting structure in the matrix F.

Test your solver on a problem instance with $m = 100$, $n = 500$. Plot the residual $\|z^{k+1} - z^{k+1/2}\|_2$ over 1000 iterations. (This should converge to zero, although perhaps slowly.)

Here is a simple method to generate LP instances that are feasible. First, generate a random vector $\omega \in \mathbf{R}^n$. Let $x^\star = \max\{\omega, 0\}$ and $\lambda^\star = \max\{-\omega, 0\}$, where the maximum is taken elementwise. Choose $A \in \mathbf{R}^{m \times n}$ and $\nu^\star \in \mathbf{R}^m$ with random entries, and set $b = Ax^\star$, $c = -A^T\nu^\star + \lambda^\star$. This gives you an LP instance with optimal value $c^T x^\star$.

(d) (3 points) Implement Dykstra's alternating projection method as shown in the lecture slides and try it on the same problem instances from part (c). Verify that you obtain a speedup, and plot the same residual as in part (c).

References

[BT09] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Imaging Sci.*, 2(1):183–202, 2009.