Optimization for Machine Learning (Homework #4)

Assignment date: Nov 2
Due date: Nov 16

Theoretical Problems (10 points)

1. (4 points) Write down the ADMM algorithms for the following problems

   - (2 points) Let $\hat{\Sigma}$ be a $d \times d$ symmetric positive semidefinite matrix. We want to solve the following problem with $d \times d$ symmetric positive definite matrices X and Z:

   $$\min_{X,Z \succ 0} \left[-\ln\det(X) + \operatorname{trace}(\hat{\Sigma}X) + \lambda \|Z\|_1 \right], \qquad X - Z = 0.$$

   Write down the ADMM algorithm, and derive the closed form solutions for the sub-optimization problems. (**Input**: ρ, $\hat{\Sigma}$, X_0, T; **Output**: X_T)

   Solution. The ADMM algorithm is as follows with $f(X) = -\ln\det(X) + \operatorname{trace}(\hat{\Sigma}X)$.

   Algorithm 1: ADMM

   Input: ρ, $\hat{\Sigma}$, X_0, T
   Output: X_T
   Let $\Lambda_0 = 0$
   Let $Z_0 = X_0$
   for $t = 1, 2, \ldots, T$ **do**
      Let $Z_t = \operatorname{trunc}(X_{t-1} + \rho^{-1}\Lambda_{t-1})$ (element-wise operation)
         where $\operatorname{trunc}(u) = \max(0, |u| - \lambda/\rho)\operatorname{sign}(u)$
      Solving: $X_t = \arg\min_x \left[\Lambda_{t-1}^\top X + f(X) + \frac{\rho}{2}\|X - Z_t\|_2^2 \right]$ as follows
         Let $U\Sigma_t U^\top$ be the eigenvalue decomposition of $\hat{\Sigma} + \Lambda_{t-1} - \rho Z_t$, where $\Sigma_t = \operatorname{diag}([\sigma_i])$
         Let $\xi_i = 0.5\rho^{-1}(-\sigma_i + \sqrt{\sigma_i^2 + 4\rho})$
         Let $X_t = U\operatorname{diag}([\xi_i])U^\top$
      Let $\Lambda_t = \Lambda_{t-1} + \rho[X_t - Z_t]$
   Return: X_T

   □

   - (2 points) Let $x \in \mathbb{R}^d$, $b \in \mathbb{R}^d$, and Σ is a $d \times d$ symmetric positive definite matrix, A is $m \times d$ matrix. Assume that we want to solve the problem

   $$\min_x \left[\frac{1}{2} x^\top \Sigma x - b^\top x \right] \qquad \text{subject to } \|Ax\|_\infty \le 1$$

   using ADMM by rewriting it as follows:

   $$\min_{x,z} \left[\frac{1}{2} x^\top \Sigma x - b^\top x + \mathbb{1}(\|z\|_\infty \le 1) \right], \quad Ax - z = 0.$$

Write down the ADMM algorithm for this decomposition with closed form solutions for the sub problems. (**Input**: ρ, Σ, A, b, x_0, T, m; **Output**: x_T)

Solution. The ADMM algorithm is presented as follows.

Algorithm 2: ADMM

Input: ρ, Σ, A, b, x_0, T, m
Output: x_T
Let $\alpha_0 = 0$
Let $z_0 = Ax_0$
for $t = 1, 2, \ldots, T$ **do**
  Let $z_t = \text{trunc}(Ax_{t-1} + \rho^{-1}\alpha_{t-1})$ (element-wise operation)
    where $\text{trunc}(u) = \max(-1, \min(1, u))$
  Let
$$x_t = (\Sigma + \rho A^\top A)^{-1}(A^\top(\alpha_{t-1} - \rho z_t) - b).$$
  Let $\alpha_t = \alpha_{t-1} + \rho[Ax_t - z_t]$
Return: x_T

□

2. (4 points) Write down the SGD algorithms for the following problems. Consider the k-class linear structured SVM problem, which has the following loss function:

$$\frac{1}{n}\sum_{i=1}^{n} \max_{y'} u_{i,y'} + \frac{\lambda}{2}\|w\|_2^2,$$

with the constraints

$$\forall i \in \{1, \ldots, n\}, y' \in \{1, \ldots, k\} : u_{i,y'} = \left[\delta(y', y_i) - w^\top \psi(x_i, y_i) + w^\top \psi(x_i, y')\right].$$

- (2 points) Write down the SGD procedure with batchsize 1 (single step update rule).
  Solution. Randomly sample i, then let $\hat{y} = \arg\max_{y'} u_{i,y'}$.
$$w \leftarrow (1 - \eta\lambda)w + \eta(\psi(x_i, y_i) - \psi(x_i, \hat{y})).$$
  □
- (2 points) Assume that $\sup_y \|\psi(x_i, y)\|_2 \le A$ and you have a budget of T total gradient evaluations. How do you set learning rate in your SGD procedure, and what is the convergence rate you expect based on the lecture?
  Solution. This is a nonsmooth strongly convex optimization problem, and $G = \sup \|\psi(x_i, y_i) - \psi(x_i, \hat{y})\|_2 \le 2A$. We can set $\eta_t = 2/(\lambda(t+1))$ to obtain convergence of
$$\frac{8A^2}{\lambda(T+3)}.$$
  □

3. (6 points) Consider the L_1-L_2 regularized loss minimization problem:

$$\min_w \left[\underbrace{\frac{1}{n}\sum_{i=1}^{n} \ln(1 + \exp(-w^\top x_i y_i)) + \frac{\lambda}{2}\|w\|_2^2}_{f(w)} + \underbrace{\mu\|w\|_1}_{g(w)} \right],$$

where $x_i, w \in \mathbb{R}^d$, $\|x_i\|_2 \le 1$, $y_i \in \{\pm 1\}$,$\lambda < 1$, and $(u)_+ = \max(0, u)$.

- (2 point) Estimate a simple upper bound of smoothness parameter and a simple lower bound of strong convexity parameter. Estimate a simple bound for the SGD variance V.

  Solution. The smoothness parameter: $L = 1/4 + \lambda$; strong convexity parameter: λ; $V \leq 1$. □

- (2 points) Write down the minibatch Accelerated Proximal SGD update rule with batch size m. (**Input**: w_0, λ, $\{\eta_t, \theta_t\}$, T, m; **Output**: w_T; Assume $\lambda > 0$) For $\tilde{T} = mT$ total gradient computations, what is the largest batch size you can choose? How do you want to set constant learning rate and momentum parameters for this batch size, and what's the convergence rate? You may use $O(\cdot)$ notation to hide constants.

  Solution. We have the following method.

Algorithm 3: Stochastic Accelerated Proximal Gradient Descent

Input: w_0, λ, $\{\eta_t, \theta_t\}$, T, m
Output: w_T

for $t = 1, 2, \ldots, T$ **do**
- Choose $\theta_t \in (0, 1]$
- Let $\beta_t = (\theta_{t-1}^{-1} - 1)\theta_t$
- Let $u_t = w_{t-1} + \beta_t(w_{t-1} - w_{t-2})$
- Randomly select a minibatch $\mathcal{B}_t$ of m independent samples from $\mathcal{D}$
- Let $\tilde{w}_t = (1 - \eta_t\lambda)u_t + \eta_t \frac{1}{m} \sum_{i \in \mathcal{B}_t} x_i y_i / (1 + \exp(u_t^\top x_i y_i))$
- Let $w_t = \max(0, |\tilde{w}_t| - \eta_t\mu)\text{sign}(\tilde{w}_t)$ (element-wise operation)

Return: x_T

The batch size can be set as large as $O(\sqrt{\lambda}\tilde{T}/\ln(\tilde{T}))$, with learning rate 0.5 and momentum $\theta_t = O(\sqrt{\lambda})$. This leads to a convergence rate of $O(\ln \tilde{T})/(\lambda\tilde{T}))$. □

- (2 points) Assume that $\lambda = 0$, write down minibatch stochastic RDA update rule for this problem with minibatch size m for $\tilde{T} = mT$ total gradients, by setting constant θ and η_0, and calculate the corresponding setting for η_t. (**Input**: w_0, $\eta_0 > 0$, $\theta \in (0, 1)$, T, m; **Output**: w_T)

  What is the largest batch size m, and what's the corresponding θ and η_0 and what is the convergence rate in terms of $\tilde{T}$? You may use $\tilde{O}(\cdot)$ which is up to a $\ln \tilde{T}$ factor.

  By comparing the solution of the proximal operator, can you explain why dual averaging can achieve more sparsity when the weights are near zero?

  Solution. We have the following method.

Algorithm 4: Stochastic Accelerated Dual Averaging

Input: w_0, $\eta_0 > 0$, $\theta \in (0, 1)$, T, m
Output: w_T

Let $z_0 = x_0$
Let $p_0 \in \partial h(x_0)$
Let $\tilde{\eta}_0 = \eta_0$
for $t = 1, 2, \ldots, T$ **do**
- Let $\eta_t = \tilde{\eta}_{t-1}\theta/(1 - \theta)$
- Let $\tilde{\eta}_t = \tilde{\eta}_{t-1} + \eta_t$
- Let $u_t = (1 - \theta)w_{t-1} + \theta z_{t-1}$
- Randomly select a minibatch $\mathcal{B}_t$ of m independent samples from $\mathcal{D}$
- Let $p_t = p_{t-1} + \eta_t \frac{1}{m} \sum_{i \in \mathcal{B}_t} x_i y_i / (1 + \exp(u_t^\top x_i y_i))$
- Let $z_t = \max(0, |p_t| - \tilde{\eta}_t\mu)\text{sign}(p_t)$ (element-wise operation)
- Let $w_t = (1 - \theta)w_{t-1} + \theta z_t$

Return: w_T

The largest $\eta_0 = O(1)$, corresponding to $\theta = \tilde{O}(\tilde{T}^{-1/4})$ and $m = \tilde{O}(\tilde{T}^{3/4})$. This implies a convergence rate of $\tilde{O}(1/\sqrt{\tilde{T}})$.

In Proximal SGD, proximal coefficients $\eta_t\mu$ matches SGD learning rate of η_t. This means sparsity can be obtain only when gradient is of scale μ, which may be difficult to achieve because μ is small and SGD gradient contains noise, so the minibatch gradient may not be close to zero even at the optimal solution. In dual averaging, the proximal coefficients $\tilde{\eta}_t\mu$ can be significantly larger than individual learning rate η_t, and thus larger shrinkage effect will lead to insensitivity to individual SGD gradient noise. □

Programming Problem (6 points)

- Using the mnist class 1 (positive) versus 7 (negative) data.
- Use the python template "progtemplate.py", and implement functions marked with '# implement'.
  - (4 pts) Implement RDA, RDA-ACCL, ProxACCL, SDCA-perm, ADMM-ACCL-linear and compare the RDA-ACCL algorithm with different θ schedulings. Submit your code and outputs.
  - (2 pts) Compare your plots to the theoretical convergence rates in class, and discuss your experimental results.