A Demo of Reduced Gradient Descent Optimization

Jiyuan Liu

June 14, 2020

Date Performed:	June 13, 2020
Partners:	None
Supervisor:	Xinwang Liu

1 Objective

Defining $\mathbf{M} \in \mathbb{R}^{m\times m}$ a positive-defined symmetric matrix, the optimization problem can be given as

$$\min_{\boldsymbol{\beta}} \boldsymbol{\beta}^\top \mathbf{M}\boldsymbol{\beta} \quad s.t. \sum_{p=1}^{m} \beta_p = 1,\ \beta_p \geq 0,\ \forall p. \tag{1}$$

It can be observed that Eq. (1) is convex. Therefore, the minimum can be found with an unique solution. In fact, This is a classical Quadratic Programming problem which can be easily solved via *quadprog* function in MATLAB. But we are going to show how to solve it via Reduced Gradient Descent algorithm.

2 Optimization One

Directly applying Gradient Descent algorithm is infeasible, for there are constraint on $\boldsymbol{\beta}$. For ease of expression, The optimization problem is firstly transformed into

$$\begin{aligned} &\min_{\boldsymbol{\beta}\in\triangle} \mathcal{J}(\boldsymbol{\beta}) \\ &s.t.\ \mathcal{J}(\boldsymbol{\beta}) = \boldsymbol{\beta}^\top \mathbf{M}\boldsymbol{\beta}, \end{aligned} \tag{2}$$

where $\triangle = \{\boldsymbol{\beta} \in \mathbb{R}^m \mid \sum_{p=1}^{m} \beta_p = 1,\ \beta_p \geq 0,\ \forall p\}$. At the same time, $\mathcal{J}(\boldsymbol{\beta})$ is differentiable, and

$$\frac{\partial \mathcal{J}(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = 2\mathbf{M}\boldsymbol{\beta} \tag{3}$$

To fulfill this goal, we firstly handle the equality constraint by computing the reduced gradient by following Rakotomamonjy et al. (2008). Let β_u be a non-zero component of $\boldsymbol{\beta}$ and $\triangledown\mathcal{J}(\boldsymbol{\beta})$ denote the reduced gradient of $\mathcal{J}(\boldsymbol{\beta})$. The p-th ($1 \leq p \leq m$) element of $\triangledown\mathcal{J}(\boldsymbol{\beta})$ is

$$[\triangledown\mathcal{J}(\boldsymbol{\beta})]_p = \frac{\partial \mathcal{J}(\boldsymbol{\beta})}{\partial \beta_p} - \frac{\partial \mathcal{J}(\boldsymbol{\beta})}{\partial \beta_u},\ \forall p \neq u \tag{4}$$

and

$$[\triangledown\mathcal{J}(\boldsymbol{\beta})]_u = \sum_{p=1, p\neq u}^{m} \left(\frac{\partial \mathcal{J}(\boldsymbol{\beta})}{\partial \beta_u} - \frac{\partial \mathcal{J}(\boldsymbol{\beta})}{\partial \beta_p}\right). \tag{5}$$

We choose u to be the index of the largest component of vector $\boldsymbol{\beta}$ which is considered to provide better numerical stability. The gradients for updating $\boldsymbol{\beta}$ can be given as

$$d_p = \begin{cases} 0 & \text{if } \beta_p = 0 \text{ and } [\triangledown\mathcal{J}(\boldsymbol{\beta})]_p > 0 \\ -[\triangledown\mathcal{J}(\boldsymbol{\beta})]_p & \text{if } \beta_p > 0 \text{ and } p \neq u \\ -[\triangledown\mathcal{J}(\boldsymbol{\beta})]_u & \text{if } p = u \end{cases} \tag{6}$$

This way, $\boldsymbol{\beta}$ can be computed via updating scheme $\boldsymbol{\beta} \leftarrow \boldsymbol{\beta} + \alpha\mathbf{d}$, where α is the step size. The overall algorithm can be designed in Algorithm 1.

Algorithm 1 A Demo of Reduced Gradient Descent Algorithm

Input: $\mathbf{M}$

Output: $\boldsymbol{\beta}$

1: initialize $\boldsymbol{\beta}^{(1)} = \mathbf{1}_m/m$, $t=1$ and $flag = 1$.
2: **while** $flag$ **do**
3: compute $\mathbf{d}^{(t)}$ in Eq. (11).
4: update $\boldsymbol{\beta}^{(t+1)} \leftarrow \boldsymbol{\beta}^{(t)} + \alpha \mathbf{d}^{(t)}$.
5: **if** $\max|\boldsymbol{\beta}^{(t+1)} - \boldsymbol{\beta}^{(t)}| \le 1e^{-4}$ **then**
6: $flag = 0$.
7: **end if**
8: $t \leftarrow t+1$.
9: **end while**

3 Optimization Two

Directly applying Gradient Descent algorithm is infeasible, for there are constraint on $\boldsymbol{\beta}$. For ease of expression, The optimization problem is firstly transformed into

$$\begin{aligned} &\min_{\boldsymbol{\beta}\in\triangle} \mathcal{J}(\boldsymbol{\beta}) \\ &s.t.\ \mathcal{J}(\boldsymbol{\beta}) = \boldsymbol{\beta}^\top \mathbf{M}\boldsymbol{\beta}, \end{aligned} \tag{7}$$

where $\triangle = \{\boldsymbol{\beta} \in \mathbb{R}^m \mid \sum_{p=1}^m \beta_p = 1,\ \beta_p \ge 0,\ \forall p\}$. At the same time, $\mathcal{J}(\boldsymbol{\beta})$ is differentiable, and

$$\frac{\partial \mathcal{J}(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = 2\mathbf{M}\boldsymbol{\beta} \tag{8}$$

We define an index set

$$\Omega' = \{p \mid \text{if } \beta_p = 0 \text{ and } \frac{\partial \mathcal{J}(\boldsymbol{\beta})}{\partial \beta_p} > 0\}, \tag{9}$$

and the complement set

$$\Omega = \{p \mid p \notin \Omega'\}. \tag{10}$$

In order to ensure $\sum_{p=1}^m \beta_p = 1,\ \beta_p \ge 0,\ \forall p$, we define the reduced gradient of $\mathcal{J}(\boldsymbol{\beta})$ by following Rakotomamonjy et al. (2008) as

$$[\triangledown \mathcal{J}(\boldsymbol{\beta})]_p = \begin{cases} 0 & p \in \Omega' \\ \frac{\partial \mathcal{J}(\boldsymbol{\beta})}{\partial \beta_p} - \frac{\partial \mathcal{J}(\boldsymbol{\beta})}{\partial \beta_u} & p \neq u,\ p\in\Omega,\ u\in\Omega \\ \sum_{p\in\Omega, p\neq u}(\frac{\partial \mathcal{J}(\boldsymbol{\beta})}{\partial \beta_u} - \frac{\partial \mathcal{J}(\boldsymbol{\beta})}{\partial \beta_p}) & p = u,\ p\in\Omega,\ u\in\Omega, \end{cases} \tag{11}$$

where u is chosen as the index of the largest component of vector $\boldsymbol{\beta}$ which is considered to provide better numerical stability. The gradients for updating $\boldsymbol{\beta}$ can be obtained as

$$d_p = -[\triangledown \mathcal{J}(\boldsymbol{\beta})]_p,\ \forall p. \tag{12}$$

This way, $\boldsymbol{\beta}$ can be computed via updating scheme $\boldsymbol{\beta} \leftarrow \boldsymbol{\beta} + \alpha \mathbf{d}$, where α is the step size. The overall algorithm can be designed in Algorithm 2.

Algorithm 2 A Demo of Reduced Gradient Descent Algorithm

Input: $\mathbf{M}$

Output: $\boldsymbol{\beta}$

initialize $\boldsymbol{\beta}^{(1)} = \mathbf{1}_m/m$, $t = 1$ and $flag = 1$.
while $flag$ **do**
    compute $\mathbf{d}^{(t)}$ in Eq. (12).
    update $\boldsymbol{\beta}^{(t+1)} \leftarrow \boldsymbol{\beta}^{(t)} + \alpha \mathbf{d}^{(t)}$.
    if $\max |\boldsymbol{\beta}^{(t+1)} - \boldsymbol{\beta}^{(t)}| \leq 1e^{-4}$ **then**
        $flag = 0$.
    end if
    $t \leftarrow t + 1$.
end while

4 Experiment

The used dataset is *Protein Fold*. Fig. (1) compares $\boldsymbol{\beta}$ obtained from the three optimization processes. It can be observed that **Optimization Two** achieves more similar $\boldsymbol{\beta}$ with the baseline *quadprog()*. At the same time, we plot the objective values along with iterations on *Protein Fold* in Fig. (2).

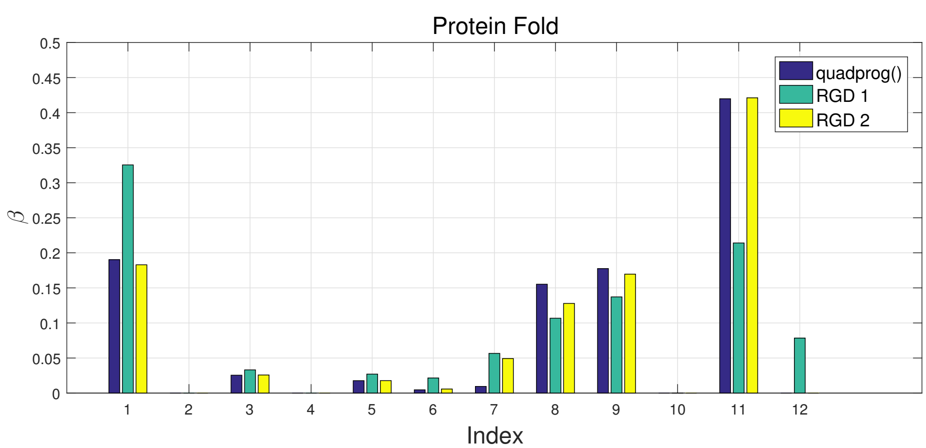

Figure 1: Comparison of $\boldsymbol{\beta}$ obtained from the three optimization processes on *Protein Fold*.

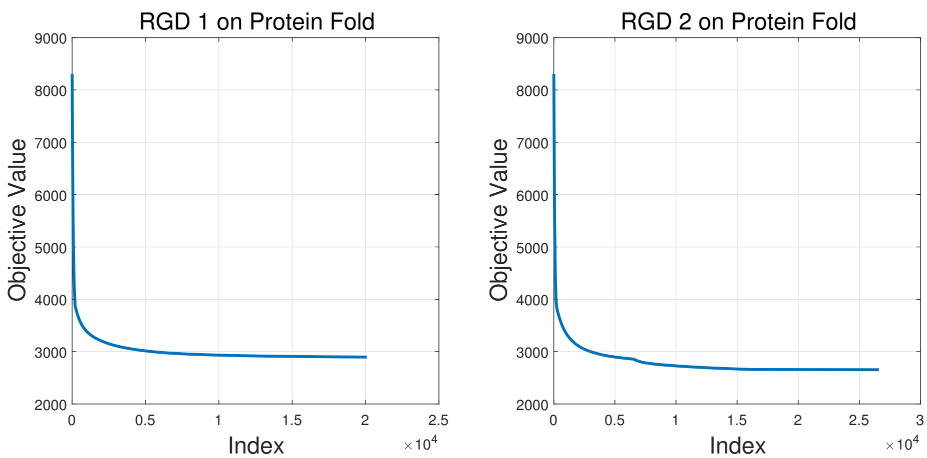

Figure 2: Objective values along with iterations on *Protein Fold*.

References

Rakotomamonjy, A., Bach, F., Canu, S., and Grandvalet, Y. (2008). Simplemkl. *Journal of Machine Learning Research*, 9.