Talk 2: Distributed Optimization and Statistical Learning via ADMM (II)

WEN Hao

2021-5-13

Main Resource: Chapter 8 of [1]

1 Distributed Model Fitting Overview

Consider a general convex (linear) model fitting problem

$$\text{minimize} \quad \ell(Ax - b) + r(x)$$

where

$$\begin{aligned}
&x \in \mathbb{R}^n : \text{parameter vector} \\
&A \in \text{Mat}_{m \times n}(\mathbb{R}) : \text{feature matrix} \\
&b \in \mathbb{R}^m : \text{output (response, etc) vector} \\
&\ell : \mathbb{R}^m \to \mathbb{R} : \text{convex loss function} \\
&r : \mathbb{R}^n \to \mathbb{R} : \text{convex regularization function}
\end{aligned}$$

Recall that ℓ is generally expressed as $\mathop{\mathbb{E}}_{z \sim \mathcal{D}} \text{loss}(x; z)$.

Question 1.1 *$\ell(Ax, b)$ could be better? ref. classification.*

For linear models with bias term, one can always add the bias term as the first (or last) element of x, and add a column with values 1 to the feature matrix A. In this way, the model can be written in a uniform and simple way Ax.

ℓ is usually additive w.r.t. samples, i.e.

$$\ell(Ax - b) = \sum_{i=1}^{m} \ell_i(a_i^T x - b_i)$$

where each ℓ_i is the loss function for sample i. For example one can assign (different) weights to each sample, thus different loss function yields from a common base loss function. For concrete examples, ref. a scikit-learn example.

Important examples of r:

$$\begin{aligned}
&r(x) = \lambda\|x\|_2^2 \text{ : ridge penalty}\\
&r(x) = \lambda\|x\|_1 \text{ : lasso penalty}\\
&r(x) = \lambda_2\|x\|_2^2 + \lambda_1\|x\|_1 \text{ : elastic net}\\
&etc.
\end{aligned}$$

2 Examples of Model Fitting

2.1 (Linear) Regression

Consider a linear model

$$b = a^T x$$

One models each sample (measurement) as

$$b_i = a_i^T x + \varepsilon_i$$

with ε_i being measurement error or noise, which are independent with log-concave density p_i (sometimes simpler, IID with density p). The likelihood function of the parameters x w.r.t. the observations $\{(a_i, b_i)\}_{i=1}^m$ is

$$\text{LH}(x) = \prod_{i=1}^{m} p_i(\varepsilon_i) = \prod_{i=1}^{m} p_i(b_i - a_i^T x)$$

If $r = 0$ (no regularization), then the model fitting problem can be interpreted as maximum likelihood estimation (MLE) of x under noise model p_i. For example, if we assume that $\varepsilon_i \sim N(0, \sigma^2)$ (IID), then the likelihood function of x is

$$\text{LH}(x) = \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(b_i - a_i^T x)^2}{2\sigma^2}\right)$$

Therefore,

$$\begin{aligned}
\text{MLE}(x) &= \arg\max_x\{\text{LH}(x)\} = \arg\min_x\{\text{NLL}(x)\} \\
&= \arg\min_x \left\{ \frac{1}{2\sigma^2} \sum_{i=1}^{n} (b_i - a_i^T x)^2 \right\} \\
&= \arg\min_x \left\{ \sum_{i=1}^{m} (b_i - a_i^T x)^2 \right\}
\end{aligned}$$

a least square problem.

If r_i is taken to be the negative log prior density of x_i, then the model fitting problem can be interpreted as max a posteriori estimates (MAP) (= arg max $\{\text{LH} \cdot \text{prior}\}$) estimation. Again, we model each sample (measurement) as $b_i = a_i^T x + \varepsilon_i$ with $\varepsilon_i \sim N(0, \sigma^2)$. Then

- if the parameters x are endowed with Laplacian prior, then MAP of x is equivalent to lasso,
- if the parameters x are endowed with normal prior, then MAP of x is equivalent to ridge regression.

For example, let x be endowed with Laplacian prior

$$p(x_j) = \frac{1}{2\tau} \exp\left(-\frac{|x_j|}{\tau}\right)$$

Then

$$\begin{aligned}
\text{MAP}(x) &= \arg\max_x\{p(x) \cdot \text{LH}(x)\} \\
&= \arg\max_x \left\{ \prod_{j=1}^{n} \frac{1}{2\tau} \exp\left(-\frac{|x_j|}{\tau}\right) \cdot \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(b_i - a_i^T x)^2}{2\sigma^2}\right) \right\} \\
&= \arg\min_x \left\{ \sum_{i=1}^{m} (b_i - a_i^T x)^2 + \lambda \|x\|_1 \right\}
\end{aligned}$$

2.2 Classification

Consider a binary classification problem (multi-class or multi-label problems can be generalized as vector or sum or mean of this kind of problems). Suppose we have samples $\{p_i, q_i\}_{i=1}^m$, with $q_i \in \{-1, 1\}$. The goal is to find a weight vector w and bias v s.t. $\text{sign}(p_i^T w + v) = q_i$ holds "for as many samples as possible". The function

$$f(p_i) = p_i^T w + v$$

is called a discriminant function ("decision function" in scikit-learn), telling on which side of the classifying hyperplane we are and how far we are away from it. The (margin-based) loss functions is usually given by

$$\ell_i(p_i^T w + v) = \ell_i(q_i(p_i^T w + v)) \quad \text{(by abuse of notation)}$$

where the quantity $\mu_i := q_i(p_i^T w + v)$ is called the margin of sample i.

As a function of the margin μ_i, ℓ_i should be (positive) decreasing. Common loss functions are

$$\begin{aligned} &\text{hinge loss}: \quad (1 - \mu_i)_+ \\ &\text{exponential loss}: \quad \exp(-\mu_i) \\ &\text{logistic loss}: \quad \log(1 + \exp(-\mu_i)) \end{aligned}$$

Recall that SVM (SVC) is to solve

$$\text{minimize} \quad \sum_{i=1}^m (1 - q_i(\, p_i^T x \, + v))_+ + \lambda \|x\|_2^2$$

where hinge loss and ℓ_2 regularizer are used. $p_i^T x$ is the SVM kernel, which can be generalized to non-linear ones $k(p_i, x)$. (for more kernel functions, ref. scikit-learn docs)

Let $f(\mu) = \dfrac{1}{1 + \exp(-\mu)}$, then $f(\mu_i) = f(q_i(p_i^T w + v))$ can be given as the probability of predicting the ground truth. In this case, the (binary) cross entropy loss is given as

$$\text{CE}_i(x) = -(1 \cdot \log(f(\mu_i)) + 0 \cdot \log(1 - f(\mu_i))) = \log(1 + \exp(-\mu_i))$$

For more loss functions and deeper insights for classification, ref. Wikipedia and references listed therein.

3 Splitting across Examples (Horizontal splitting)

In the model fitting problem

$$\text{minimize} \quad \ell(Ax - b) + r(x)$$

we partition the feature matrix A and labels b by rows, i.e.

$$A = \begin{pmatrix} A_1 \\ \vdots \\ A_N \end{pmatrix}, \quad b = \begin{pmatrix} b_1 \\ \vdots \\ b_N \end{pmatrix},$$

where $A_i \in \text{Mat}_{m_i \times n}, b_i \in \mathbb{R}^{m_i}$ are from samples of "client" i. The model fitting problem thus is formulated as follows

$$\begin{aligned} &\text{minimize} && \sum_{i=1}^{N} \ell_i(A_i x_i - b_i) + r(z) \\ &\text{subject to} && x_i = z \end{aligned}$$

as a **consensus problem (with regularization)**.

The scaled ADMM iterations of the above optimization problem are

$$\begin{aligned} x_i^{k+1} &= \arg\min_{x_i} \left\{ \ell_i(A_i x_i - b_i) + \frac{\rho}{2} \|x_i - z^k + u_i^k\|_2^2 \right\} = \text{prox}_{\tilde{\ell}_i, \rho}(z^k - u_i^k) \\ z^{k+1} &= \arg\min_{z} \left\{ r(z) + \frac{N\rho}{2} \|z - \overline{x}^{k+1} - \overline{u}^k\|_2^2 \right\} = \text{prox}_{r, N\rho}(\overline{x}^{k+1} + \overline{u}^k) \\ u_i^{k+1} &= u_i^k + (x_i^{k+1} - z^{k+1}) \end{aligned}$$

where $\tilde{\ell}_i(x_i) := \ell_i(A_i x_i - b_i)$. It can be seen that

$$\begin{aligned} &x\text{-update} \leftarrow \text{parallel } \ell_2\text{-regularized model fitting problems} \\ &z\text{-update} \leftarrow \text{averaging } x, z, \text{ and minimization problem} \end{aligned}$$

3.1 Example: Lasso

Recall that Lasso is the following optimization problem

$$\text{minimize} \quad \frac{1}{2}\|Ax - b\|_2^2 + \lambda \|x\|_1$$

The corresponding distributed (consensus) version of ADMM algorithm is

$$
\begin{aligned}
x_i^{k+1} &= \arg\min_{x_i} \left\{ \frac{1}{2}\|A_i x_i - b_i\|_2^2 + \frac{\rho}{2}\|x_i - z^k + u_i^k\|_2^2 \right\} \\
z^{k+1} &= \arg\min_{z} \left\{ \lambda\|z\|_1 + \frac{N\rho}{2}\|z - \overline{x}^{k+1} - \overline{u}^k\|_2^2 \right\} = \mathrm{S}_{\lambda/N\rho}(\overline{x}^{k+1} + \overline{u}^k) \\
u_i^{k+1} &= u_i^k + (x_i^{k+1} - z^{k+1})
\end{aligned}
$$

Each x_i-update is a ridge regression problem, which is equivalent to the least square problem

$$
\text{minimize} \left\| \begin{pmatrix} A_i \\ \sqrt{\rho} I \end{pmatrix} x_i - \begin{pmatrix} b_i \\ \sqrt{\rho}(z^k - u_i^k) \end{pmatrix} \right\|_2^2
$$

thus having analytic solution (and numerically solved by the so-called direct method)

$$
\begin{aligned}
x_i^{k+1} &= \left(\begin{pmatrix} A_i \\ \sqrt{\rho} I \end{pmatrix}^T \cdot \begin{pmatrix} A_i \\ \sqrt{\rho} I \end{pmatrix} \right)^{-1} \cdot \begin{pmatrix} A_i \\ \sqrt{\rho} I \end{pmatrix}^T \cdot \begin{pmatrix} b_i \\ \sqrt{\rho}(z^k - u_i^k) \end{pmatrix} \\
&= (A_i^T A_i + \rho I)^{-1} (A_i^T b_i + \rho(z^k - u_i^k))
\end{aligned}
$$

Accelerations on x_i-updates:

(1) $(A_i^T A_i + \rho I)^{-1}$ is independent of k, hence (its factorizations) can be pre-computed and used for each x_i update.

(2) If further, $m_i < n$ (# samples < # features), by Woodbury matrix identity (or matrix inverse lemma),

$$
(A_i^T A_i + \rho I)^{-1} = \frac{1}{\rho} - \frac{1}{\rho} A_i^T (A_i A_i^T + \rho I)^{-1} A_i
$$

The size $A_i A_i^T + \rho I$ is smaller, hence requires less computation.

3.2 Example: SVM (SVC)

Recall again that the SVM (SVC) is the following optimization problem

$$
\text{minimize} \quad \sum_{i=1}^{m} (1 - q_i(p_i^T x + v))_+ + \lambda \|x\|_2^2
$$

Ignore the bias term v for convenience, otherwise one can replace x by $\begin{pmatrix} x \\ v \end{pmatrix}$, and replace p_i^T by $(p_i^T, 1)$. Write

$$A = \begin{pmatrix} -q_1 p_1^T \\ \vdots \\ -q_m p_m^T \end{pmatrix},$$

then the problem rewrites

$$\text{minimize} \quad \mathbf{1}^T(\mathbf{1} + Ax)_+ + \lambda\|x\|_2^2$$

and in the horizontal splitting consensus form as

$$\begin{array}{ll} \text{minimize} & \mathbf{1}^T(\mathbf{1} + A_i x_i)_+ + \lambda\|z\|_2^2 \\ \text{subject to} & x_i = z \end{array}$$

with ADMM iterations

$$\begin{aligned} x_i^{k+1} &= \arg\min_{x_i} \left\{ \mathbf{1}^T(\mathbf{1} + A_i x_i)_+ + \frac{\rho}{2}\|x_i - z^k + u_i^k\|_2^2 \right\} \\ z^{k+1} &= \arg\min_{z} \left\{ \lambda\|z\|_2^2 + \frac{N\rho}{2}\|z - \overline{x}^{k+1} - \overline{u}^k\|_2^2 \right\} = \frac{N\rho}{2\lambda + N\rho}(\overline{x}^{k+1} + \overline{u}^k) \\ u_i^{k+1} &= u_i^k + (x_i^{k+1} - z^{k+1}) \end{aligned}$$

4 Splitting across Features (Vertical splitting)

Let the feature matrix A and parameter vector x be partitioned vertically as

$$A = (A_1, \cdots, A_N), \quad x = (x_1, \cdots, x_N)$$

with $A_i \in \mathrm{Mat}_{m \times n_i}(\mathbb{R}), x_i \in \mathbb{R}^{n_i}$. Each A_i can be considered as "partial" feature matrix, and $A_i x_i$ "partial" predictions. The "full" prediction is given as

$$Ax = \sum_{i=1}^{N} A_i x_i$$

The model fitting problem hence is formulated as follows

$$\text{minimize} \quad \ell(\sum_{i=1}^{N} A_i x_i - b) + \sum_{i=1}^{N} r_i(x_i)$$

or better to be written

$$\text{minimize} \quad \sum_{i=1}^{N} r_i(x_i) + \ell(\sum_{i=1}^{N} A_i x_i - b)$$

which can be further formulated as a sharing problem

$$\begin{array}{ll}\text{minimize} & \sum_{i=1}^{N} r_i(x_i) + \ell(\boxed{\sum_{i=1}^{N} z_i} - b) \\ \text{subject to} & A_i x_i = z_i\end{array}$$

The scaled ADMM iterations (slightly different from a standard sharing problem) are

$$\begin{aligned}
x_i^{k+1} &= \arg\min_{x_i} \left\{ r_i(x_i) + \frac{\rho}{2} \|A_i x_i - A_i x_i^k - \overline{z}^k + \overline{Ax}^k + u^k\|_2^2 \right\} \\
\overline{z}^{k+1} &= \arg\min_{\overline{z}} \left\{ \ell(N\overline{z} - b) + \frac{N\rho}{2} \|\overline{z} - \overline{Ax}^{k+1} - u^k\|_2^2 \right\} \\
u^{k+1} &= u^k + (\overline{Ax}^{k+1} - \overline{z}^{k+1})
\end{aligned}$$

which can be interpreted as

$$\begin{aligned}
x\text{-update} &\leftarrow \text{parallel regularized } (r_i) \text{ least square problems} \\
\overline{z}\text{-update} &\leftarrow \ell_2 \text{ regularized loss } (\ell) \text{ minimization problem}
\end{aligned}$$

Here $\overline{Ax} := \frac{1}{N} \sum_{i=1}^{N} A_i x_i$

4.1 Example: Lasso

We fit the Lasso optimization problem

$$\text{minimize} \quad \frac{1}{2}\|Ax - b\|_2^2 + \lambda \|x\|_1$$

into the form of the vertical splitting sharing problem as

$$\text{minimize} \quad \frac{1}{2} \left\| \sum_{i=1}^{N} z_i - b \right\|_2^2 + \lambda \sum_{i=1}^{N} \|x_i\|_1$$

$$\text{subject to} \quad A_i x_i = z_i$$

with ADMM iterations

$$\begin{aligned}
x_i^{k+1} &= \arg\min_{x_i} \left\{ \lambda \|x_i\|_1 + \frac{\rho}{2} \|A_i x_i - A_i x_i^k - \overline{z}^k + \overline{Ax}^k + u^k\|_2^2 \right\} \\
&= \arg\min_{x_i} \left\{ \frac{1}{2} \|A_i x_i - \underbrace{(A_i x_i^k - \overline{Ax}^k + \overline{z}^k - u^k)}_{v_i}\|_2^2 + \frac{\lambda}{\rho} \|x_i\|_1 \right\} \\
&\leftarrow N \text{ parallel smaller Lasso problem} \\
\overline{z}^{k+1} &= \arg\min_{\overline{z}} \left\{ \frac{1}{2} \|N\overline{z} - b\|_2^2 + \frac{N\rho}{2} \|\overline{z} - \overline{Ax}^{k+1} - u^k\|_2^2 \right\} \\
&= \frac{1}{N+\rho} \left(b + \overline{Ax}^{k+1} + \overline{u}^k \right) \\
u^{k+1} &= u^k + (\overline{Ax}^{k+1} - \overline{z}^{k+1})
\end{aligned}$$

For the x_i-update, $x_i^{k+1} := \arg\min_{x_i} \left\{ \frac{1}{2} \|v_i - A_i x_i\|_2^2 + \frac{\lambda}{\rho} \|x_i\|_1 \right\}$ has to satisfy the subgradient conditions

$$A_i^T (v_i - A_i x_i^{k+1}) = \frac{\lambda}{\rho} \partial \|x_i^{k+1}\|_1 = \frac{\lambda}{\rho} \begin{pmatrix} s_1 \\ \vdots \\ s_n \end{pmatrix}$$

$$\text{where} \quad s_j \begin{cases} = \text{sign}((x_i^{k+1})_j) & \text{if } (x_i^{k+1})_j \neq 0 \\ \in [-1, 1] & \text{if } (x_i^{k+1})_j = 0 \end{cases}$$

It is claimed that

$$x_i^{k+1} = 0 \iff \|A_i^T v_i\|_\infty \leqslant \frac{\lambda}{\rho}$$

Indeed, consider

$$\mathcal{L}(x_i) := \frac{1}{2} \|v_i - A_i x_i\|_2^2 + \frac{\lambda}{\rho} \|x_i\|_1,$$

then

$$\begin{aligned}
0 \text{ is solution to } \arg\min_{x_i} \mathcal{L}(x_i) &\iff \nabla_s \mathcal{L}(0) \geqslant 0, \ \forall s \\
&\iff \langle -A_i^T (v_i - 0), s \rangle + \frac{\lambda}{\rho} \|s\|_1 \geqslant 0, \ \forall s
\end{aligned}$$

$$
\begin{aligned}
&\iff \frac{\lambda}{\rho} \geqslant \max_{\|s\|_1=1} \langle A_i^T v_i, s\rangle \\
&\iff \frac{\lambda}{\rho} \geqslant \|A_i^T v_i\|_\infty
\end{aligned}
$$

For more, ref. [2] exercise 2.1.

4.2 Example: Group Lasso

Group Lasso is the following generalization, where features are (rearranged if needed) grouped and corr. to a vertical splitting, of the standard Lasso:

$$
\text{minimize} \quad \left\{ \frac{1}{2} \left\| \sum_{i=1}^{N} A_i x_i - b \right\|_2^2 + \lambda \sum_{i=1}^{N} \|x_i\|_2 \right\}
$$

ADMM iterations are

$$
\begin{aligned}
x_i^{k+1} &= \operatorname*{arg\,min}_{x_i} \left\{ \frac{1}{2} \|A_i x_i - \underbrace{(A_i x_i^k - \overline{Ax}^k + \overline{z}^k - u^k)}_{v_i}\|_2^2 + \frac{\lambda}{\rho}\|x_i\|_2 \right\} \\
\overline{z}^{k+1} &= \operatorname*{arg\,min}_{\overline{z}} \left\{ \frac{1}{2}\|N\overline{z} - b\|_2^2 + \frac{N\rho}{2}\|\overline{z} - \overline{Ax}^{k+1} - u^k\|_2^2 \right\} \\
&= \frac{1}{N+\rho}\left(b + \overline{Ax}^{k+1} + \overline{u}^k \right) \\
u^{k+1} &= u^k + (\overline{Ax}^{k+1} - \overline{z}^{k+1})
\end{aligned}
$$

For the x_i-update, one similarly has

$$
A_i^T (v_i - A_i x_i^{k+1}) = \frac{\lambda}{\rho} \partial \|x_i^{k+1}\|_2 \begin{cases} = \dfrac{\lambda}{\rho} \cdot \dfrac{x_i^{k+1}}{\|x_i^{k+1}\|_2} & \text{if } x_i^{k+1} \neq 0 \\ \in \dfrac{\lambda}{\rho} \cdot \mathbb{B}(0,1) & \text{if } x_i^{k+1} = 0 \end{cases}
$$

i.e.

$$
x_i^{k+1} = (A_i^T A_i + \tilde{\lambda})^{-1} A_i^T v_i \quad \text{with } \tilde{\lambda} \text{ satisfying } \tilde{\lambda}\rho\|x_i^{k+1}\|_2 = \lambda \text{ if } x_i^{k+1} \neq 0.
$$

Again, it's claimed that (note the difference with ordinary Lasso on the penalty term)

$$
x_i^{k+1} = 0 \iff \|A_i^T v_i\|_2 \leqslant \frac{\lambda}{\rho}
$$

4.3 Example: SVM

The vertical splitting version of SVM is

$$\text{minimize} \quad \mathbf{1}^T(\mathbf{1} + \sum_{i=1}^{N} A_i x_i)_+ + \lambda \sum_{i=1}^{N} \|x_i\|_2^2$$

ADMM iterations are

$$\begin{aligned}
x_i^{k+1} &= \operatorname*{arg\,min}_{x_i} \left\{ \frac{1}{2} \|A_i x_i - \underbrace{(A_i x_i^k - \overline{Ax}^k + \overline{z}^k - u^k)}_{v_i}\|_2^2 + \frac{\lambda}{\rho} \|x_i\|_2^2 \right\} \\
&\leftarrow \text{ parallel ridge regression} \\
&= \left(A_i^T A_i + \frac{2\lambda}{\rho} I \right)^{-1} A_i^T v_i \\
\overline{z}^{k+1} &= \operatorname*{arg\,min}_{\overline{z}} \left\{ \mathbf{1}^T(\mathbf{1} + N\overline{z})_+ + \frac{N\rho}{2} \|\overline{z} - \underbrace{(\overline{Ax}^{k+1} + u^k)}_{s}\|_2^2 \right\} \\
&= \operatorname*{arg\,min}_{\overline{z}} \left\{ \sum_{j=1}^{n} \left((1 + N\overline{z}_j)_+ + \frac{N\rho}{2} (\overline{z}_j - s_j)^2 \right) \right\} \\
u^{k+1} &= u^k + (\overline{Ax}^{k+1} - \overline{z}^{k+1})
\end{aligned}$$

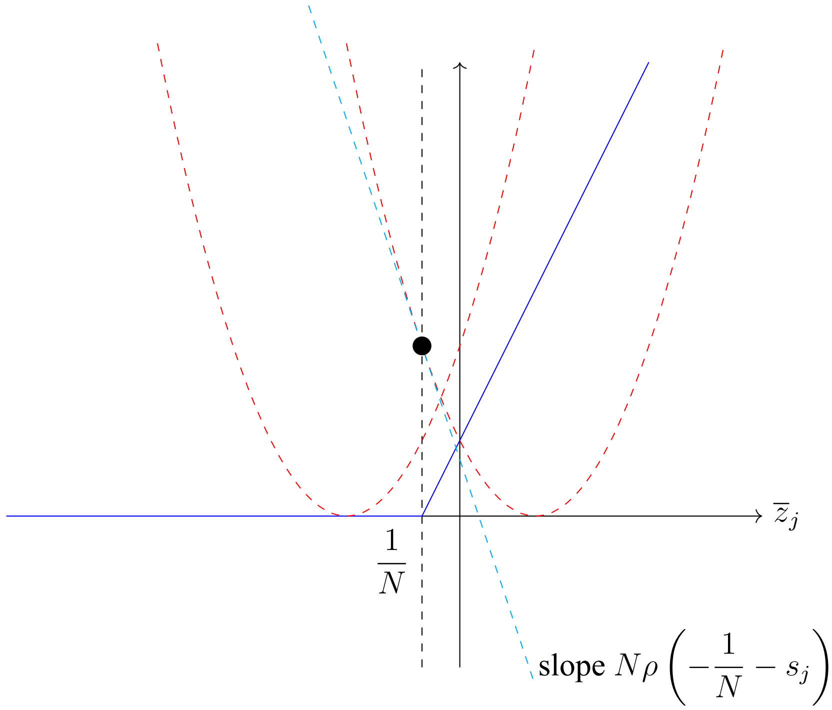

Figure 1: sketch of $\overline{z}$-update of vertical splitting SVM

$\overline{z}$-update splits to the component level, i.e.

$$(1 + N\overline{z}_j)_+ + \frac{N\rho}{2}(\overline{z}_j - s_j)^2$$

and are easily computed

$$\overline{z}_j = \begin{cases} s_j - \frac{1}{\rho} & \text{if } s_j > -\frac{1}{N} + \frac{1}{\rho} \\ -\frac{1}{N} & \text{if } s_j \in [-\frac{1}{N}, -\frac{1}{N} + \frac{1}{\rho}] \\ s_j & \text{if } s_j < -\frac{1}{N} \end{cases}$$

References

[1] S. Boyd, N. Parikh, and E. Chu, *Distributed optimization and statistical learning via the alternating direction method of multipliers*. Now Publishers Inc., 2011.

[2] T. Hastie, R. Tibshirani, and M. Wainwright, *Statistical Learning with Sparsity: the LASSO and Generalizations*. Chapman and Hall/CRC, 2019.