ReHLine: Regularized Composite ReLU-ReHU Loss Minimization with Linear Computation and Linear Convergence

Ben Dai* (CUHK)
Yixuan Qiu* (SUFE)

NEURAL INFORMATION PROCESSING SYSTEMS

Introduction

In this paper, we consider a general regularized ERM based on a convex (but possible **nonsmooth**) PLQ loss with linear **constraints**:

$$\min_{\beta \in \mathbb{R}^d} \sum_{i=1}^n L_i(\mathbf{x}_i^\intercal \beta) + \frac{1}{2}\|\beta\|_2^2, \qquad \text{s.t. } \mathbf{A}\beta + \mathbf{b} \geq \mathbf{0}, \qquad (1)$$

where $\mathbf{x}_i \in \mathbb{R}^d$ is the covariate vector for the i-th observation, $\beta \in \mathbb{R}^d$ is an unknown coefficient vector, $\mathbf{A} \in \mathbb{R}^{K \times d}$ and $\mathbf{b} \in \mathbb{R}^K$ are posed as linear inequality constraints for β, and $L_i(\cdot) \geq 0$ is a convex piecewise linear-quadratic loss (PLQ) loss function.

Table 1. Overview of existing algorithms in solving (1).

Algorithm	Complexity (per iteration)	#Iteration	Complexity (total)
P-GD	$\mathcal{O}(n)$	$\mathcal{O}(\varepsilon^{-1})$ [6]	$\mathcal{O}(n\varepsilon^{-1})$
CD	$\mathcal{O}(n^2)$	$\mathcal{O}(\log(\varepsilon^{-1}))$ [31]	$\mathcal{O}(n^2\log(\varepsilon^{-1}))$
IPM	$\mathcal{O}(n^2)$	$\mathcal{O}(\log(\varepsilon^{-1}))$ [18]	$\mathcal{O}(n^2\log(\varepsilon^{-1}))$
ADMM	$\mathcal{O}(n^2)$	$o(\varepsilon^{-1})$ [9, 20]	$o(n^2\varepsilon^{-1})$
SDCA	$\mathcal{O}(n)$	$\mathcal{O}(\varepsilon^{-1})$ [39]	$\mathcal{O}(n\varepsilon^{-1})$
ReHLine (ours)	$\mathcal{O}(n)$	$\mathcal{O}(\log(\varepsilon^{-1}))$	$\mathcal{O}(n\log(\varepsilon^{-1}))$

Contribution. Compared with existing algorithms, the proposed ReHLine solver has four appealing "*linear properties*":

- It applies to any convex piecewise linear-quadratic loss function (potential for non-smoothness included).
- In addition, it supports linear constraints on the parameter vector.
- The optimization algorithm has a provable linear convergence rate.
- The per-iteration computation is linear in the sample size.

The ReHLine Decomposition

Definition 1. A function $L(z)$ is composite ReLU-ReHU, if there exist $\mathbf{u}, \mathbf{v} \in \mathbb{R}^L$ and $\tau, \mathbf{s}, \mathbf{t} \in \mathbb{R}^H$ such that

$$L(z) = \sum_{l=1}^L \mathrm{ReLU}(u_l z + v_l) + \sum_{h=1}^H \mathrm{ReHU}_{\tau_h}(s_h z + t_h), \qquad (3)$$

where $\mathrm{ReLU}(z) = z_+$, and $\mathrm{ReHU}_{\tau_h}(z)$ defined in (2).

Theorem 1. A loss function L is convex PLQ iff it is composite ReLU-ReHU.

Table 2. Some widely used composite ReHLine losses as in (3).

Problem	Loss ($L_i(z_i)$)	Composite ReLU-ReHU Parameters
SVM	$c_i(1 - y_i z_i)_+$	$u_{1i} = -c_i y_i$, $v_{1i} = c_i$
s-SVM	$c_i \mathrm{ReHU}_1(-(y_i z_i - 1))$	$s_{1i} = -\sqrt{c_i} y_i$, $t_{1i} = \sqrt{c_i}$, $\tau = \sqrt{c_i}$
SVM^2	$c_i((1 - y_i z_i)_+)^2$	$s_{1i} = -\sqrt{2c_i} y_i$, $t_{1i} = \sqrt{2c_i}$, $\tau = \infty$
LAD	$c_i\lvert y_i - z_i\rvert$	$u_{1i} = c_i$, $v_{1i} = -c_i y_i$, $u_{2i} = -c_i$, $v_{2i} = c_i y_i$
SVR	$c_i(\lvert y_i - z_i\rvert - \varepsilon)_+$	$u_{1i} = c_i$, $v_{1i} = -(y_i + \varepsilon)$, $u_{2i} = -c_i$, $v_{2i} = y_i - \varepsilon$
QR	$c_i \rho_\kappa(y_i - z_i)$	$u_{1i} = -c_i\kappa$, $v_{1i} = \kappa c_i y_i$, $u_{2i} = c_i(1-\kappa)$, $v_{2i} = -c_i(1-\kappa)y_i$

Taken together, (1) can be rewritten as:

$$\min_{\boldsymbol{\beta} \in \mathbb{R}^d} \sum_{i=1}^n \sum_{l=1}^L \mathrm{ReLU}(u_{li}\mathbf{x}_i^\intercal \boldsymbol{\beta} + v_{li}) + \sum_{i=1}^n \sum_{h=1}^H \mathrm{ReHU}_{\tau_{hi}}(s_{hi}\mathbf{x}_i^\intercal \boldsymbol{\beta} + t_{hi}) + \frac{1}{2}\|\boldsymbol{\beta}\|_2^2, \quad \text{s.t. } \mathbf{A}\boldsymbol{\beta} + \mathbf{b} \geq \mathbf{0}, \qquad (4)$$

where $\mathbf{U} = (u_{li}), \mathbf{V} = (v_{li}) \in \mathbb{R}^{L \times n}$ and $\mathbf{S} = (s_{hi}), \mathbf{T} = (t_{hi}), \mathrm{T} = (\tau_{hi}) \in \mathbb{R}^{H \times n}$ are the ReLU-ReHU loss parameters, as illustrated in Table 2.

The **Lagrangian dual**, which is a box-QP, is presented in **Theorem 2** of our paper, which is derived using the Karush-Kuhn-Tucker (KKT) condition:

$$\widehat{\boldsymbol{\beta}} = \sum_{k=1}^K \widehat{\xi}_k \mathbf{a}_k - \sum_{i=1}^n \mathbf{x}_i \left(\sum_{l=1}^L \widehat{\lambda}_{li} u_{li} + \sum_{h=1}^H \widehat{\gamma}_{hi} s_{hi} \right) = \mathbf{A}^\intercal \widehat{\boldsymbol{\xi}} - \bar{\mathbf{U}}_{(3)} vec(\widehat{\boldsymbol{\Lambda}}) - \bar{\mathbf{S}}_{(3)} vec(\widehat{\boldsymbol{\Gamma}}).$$

Algorithm and Results

This proposed **ReHLine** is based on the coordinate descent (CD), drawing inspiration from *Liblinear*. Its motivation is to utilize the **linear structure** in the KKT conditions, and simultaneously update primal and dual variables, considerably reducing the computational complexity for CD updates. For illustration, we only demo one dual variable, see the full details in the paper.

Canonical CD updates. By excluding the terms unrelated to λ_{li}:

$$\lambda_{li}^{\text{new}} = \mathcal{P}_{[0,1]}\left(\frac{u_{li}\mathbf{x}_i^\intercal \left(\sum_{k=1}^K \xi_k \mathbf{a}_k - \sum_{(l',i') \neq (l,i)} \lambda_{l'i'} u_{l'i'} \mathbf{x}_{i'} - \sum_{h',i'} \gamma_{h'i'} s_{h'i'} \mathbf{x}_{i'} \right) + v_{li}}{u_{li}^2 \|\mathbf{x}_i\|_2^2} \right), \qquad (10)$$

updating one λ_{li} value requires $\mathcal{O}(K + nd + nL + nH)$ of computation. Adding all variables together, the canonical CD update rule for one full cycle has a computational complexity of $\mathcal{O}\big((K + nd + nL + nH)(K + nL + nH)\big)$.

ReHLine updates significantly reduces the computational complexity of canonical CD by updating β according to the KKT condition (9) after each update of a dual variable.

$$\lambda_{li}^{\text{new}} = \mathcal{P}_{[0,1]}\left(\lambda_{li}^{\text{old}} - \frac{\nabla_{\lambda_{li}} \mathcal{L}(\lambda^{\text{old}})}{u_{li}^2 \|\mathbf{x}_i\|_2^2} \right) = \mathcal{P}_{[0,1]}\left(\lambda_{li}^{\text{old}} + \frac{u_{li}\mathbf{x}_i^\intercal \beta^{\text{old}} + v_{li}}{u_{li}^2 \|\mathbf{x}_i\|_2^2} \right).$$

Accordingly, the primal variable β is updated as

$$\beta^{\text{new}} = \beta^{\text{old}} - (\lambda_{li}^{\text{new}} - \lambda_{li}^{\text{old}}) u_{li} \mathbf{x}_i,$$

updating one λ_{li} value only requires $\mathcal{O}(d)$ of computation. Adding all variables together, the ReHLine update rule for one full cycle has a computational complexity of $\boxed{\mathcal{O}\big((K + nL + nH)d\big)}$.

Theorem 2. Let $\mu^{(q)}$ be a sequence of iterates generated by ReHLine. Then the dual objective converges at least linearly to that of μ^*.

Table 5. The running times of SOTA solvers on ML tasks using the *Benchopt*. "✗" indicates cases where the solver produced an invalid solution or exceeded the allotted time limit ("objective" for failure on objective function, and "both" for both objective and feasibility). **Speed-up** refers to the speed-up in running time achieved by **ReHLine**.

Task	Dataset	ECOS	MOSEK	SCS	DCCP	ReHLine
FairSVM	SPF (×1e-4)	✗	✗	✗	✗	4.25(±0.5)
	Philippine (×1e-2)	1550(±0.6)	87.4(±0.2)	130(±42)	1137(±9.2)	1.03(±0.2)
	Sylva-prior (×1e-2)	✗	✗	✗	✗	0.47(±0.1)
	Creditcard (×1e-1)	175(±0.2)	64.2(±0.1)	161(±405)	✗	0.64(±0.2)
	Fail/Succeed	2/2	2/2	2/2	3/1	0/4
	Speed-up (on Creditcard)	273x	100x	252x	∞	–

Task	Dataset	ECOS	MOSEK	SCS	HQREG	ReHLine
RidgeHuber	Liver-disorders (×1e-4)	✗	✗	✗	4.90(±0.00)	1.40(±0.20)
	Kin8nm (×1e-3)	✗	✗	✗	1.58(±0.21)	2.04(±0.30)
	House-8L (×1e-3)	✗	925(±2)	✗	2.42(±0.34)	0.80(±0.21)
	Topo-2-1 (×1e-2)	2620(±1040)	267(±1)	213(±2)	3.53(±0.67)	1.78(±0.32)
	BT (×1e-1)	✗	2384(±433)	✗	12.5(±1.8)	5.28(±1.31)
	Fail/Succeed	4/1	2/3	4/1	0/5	0/5
	Speed-up (on BT)	∞	452x	∞	2.37x	–

Task	Dataset	SAGA	SAG	SDCA	SVRG	ReHLine
sSVM	SPF (×1e-4)	39.9(±4.6)	28.3(±5.0)	15.0(±2.4)	41.4(±3.9)	4.80(±1.20)
	Philippine (×1e-2)	24.3(±27.8)	5.53(±9.8)	1.47(±0.19)	15.8(±6.8)	0.89(±0.10)
	Sylva-prior (×1e-2)	3.37(±9.81)	3.00(±0.56)	1.57(±0.23)	3.40(±0.84)	0.86(±0.14)
	Creditcard (×1e-2)	10.4(±1.4)	15.0(±2.0)	14.0(±1.9)	11.2(±1.4)	6.36(±1.92)
	Fail/Succeed	0/4	0/4	0/4	0/4	0/4
	Speed-up (on Creditcard)	1.6x	2.3x	2.2x	1.7x	–

Task	Dataset	ECOS	MOSEK	SCS	ReHLine
ElasticQR	LD (×1e-4)	✗	106(±7)	34.9(±25.0)	2.60(±0.30)
	Kin8nm (×1e-3)	✗	92.0(±1.0)	63.1(±58.5)	4.12(±0.95)
	House-8L (×1e-3)	887(±161)	277(±34)	✗	7.21(±1.99)
	Topo-2-1 (×1e-2)	4752(±2015)	✗	✗	3.04(±0.49)
	BT (×1e-0)	7079(±2517)	✗	✗	2.49(±0.56)
	Fail/Succeed	3/2	2/3	3/2	0/5
	Speed-up (on BT)	2843x	∞	∞	–

Task	Dataset	ECOS	MOSEK	SCS	LIBLINEAR	ReHLine
SVM	SPF (×1e-4)	✗	372(±1)	237(±27)	12.7(±0.1)	3.90(±0.10)
	Philippine (×1e-2)	1653(±41)	86.5(±0.2)	153(±146)	1.80(±0.02)	0.82(±0.02)
	Sylva-prior (×1e-3)	✗	731(±2)	843(±1006)	16.0(±0.6)	4.08(±0.84)
	Creditcard (×1e-2)	2111(±804)	✗	1731(±4510)	23.1(±2.5)	5.08(±1.45)
	Fail/Succeed	2/2	1/3	0/4	0/4	0/4
	Speed-up (on Creditcard)	415x	∞	340x	4.5x	–