A primal dual semismooth Newton algorithms framework for convex composite optimization

We develop a semismooth Newton based algorithm to solve a class of convex composite optimization problem. Different from many SSN based solvers which are designed case by case or only able to solve two blocks problems. The solver proposed in this paper is able to solve multi-block problems such as conic programming, Lasso type problems and quadratic programming. By dealing internally with the intrinsic nonsmooth property of $p(\boldsymbol{x})$, all the different problems are solved in a unified saddle point framework induced from augment Lagrangian strong duality which lowers the entrance barrier and hence user-friendly. Such structured constraints appear pervasively in image processing and machine learning such as robust PCA clustering problems.

Additional Key Words and Phrases: convex composite optimization, semismooth Newton method, Matlab software package.

ACM Reference Format:
. 2018. A primal dual semismooth Newton algorithms framework for convex composite optimization. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation email (Conference acronym 'XX)*. ACM, New York, NY, USA, 20 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

In this paper, we aim to develop a algorithm framework for the following convex composite optimization problem:

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \langle \boldsymbol{c}, \boldsymbol{x} \rangle + \frac{1}{2} \langle \boldsymbol{x}, \mathcal{Q}(\boldsymbol{x}) \rangle + f(\mathcal{B}(\boldsymbol{x})) + p(\boldsymbol{x}), \\ \text{s.t.} \quad & \boldsymbol{x} \in \mathcal{P}_1, \ \mathcal{A}(\boldsymbol{x}) \in \mathcal{P}_2. \end{aligned} \tag{1.1}$$ {general}

where $\boldsymbol{c} \in \mathbb{R}^n$, $\mathcal{A} : \mathbb{R}^n \to \mathbb{R}^m$, $\mathcal{B} : \mathbb{R}^n \to \mathbb{R}^p$ are linear operators, $f : \mathbb{R}^p \to \mathbb{R}$ is a convex function, $\mathcal{P}_1 = \{\boldsymbol{x} \in \mathbb{R}^n | \mathsf{l} \leq \boldsymbol{x} \leq \mathsf{u}\}$ and $\mathcal{P}_2 = \{\boldsymbol{x} \in \mathbb{R}^m | \mathsf{lb} \leq \boldsymbol{x} \leq \mathsf{ub}\}$, $\mathcal{Q} \in \mathbb{S}^n_+$ is a positive semidefinite matrix or operator, $p(x)$ is a convex and nonsmooth function. The choices of $p(\boldsymbol{x})$ provide flexibility to handle many kinds of problems.

While the proposed model (1.1) focus on one single variables $\boldsymbol{x}$, our solver is capable of solving the following more general problem with N block of of variables with shift term:

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \sum_{i=1}^{N} \langle \boldsymbol{c}_i, \boldsymbol{x}_i \rangle + \sum_{i=1}^{N} \frac{1}{2} \langle \boldsymbol{x}_i, \mathcal{Q}_i(\boldsymbol{x}_i) \rangle + \sum_{i=1}^{N} f_i(\mathcal{B}_i(\boldsymbol{x}) - \boldsymbol{b}_{2,i}) + \sum_{i=1}^{N} p_i(\boldsymbol{x}_i - \boldsymbol{b}_{1,i}), \\ \text{s.t.} \quad & \boldsymbol{x}_i \in \mathcal{P}_{1,i}, \ \sum_{i=1}^{N} \mathcal{A}_i(\boldsymbol{x}_i) \in \mathcal{P}_{2,i}, \quad i = 1, \cdots, N. \end{aligned} \tag{1.2}$$ {general-block}

Author's Contact Information:

Manuscript submitted to ACM

In model (1.1), $p(\boldsymbol{x})$ usually serves as a regularizer or constraint, which covers a wide range of nonsmooth problems and applications [5, 6]. Compared to the regularized least squares form, the least squares constrained formulation:

$$\min_{\boldsymbol{x}} \quad \{p(\boldsymbol{x}) \mid \|\mathcal{A}\boldsymbol{x} - b\|_\infty \leq \lambda\} \tag{1.3}$$ {comp2}

is widely believed to be computationally more challenging because of the complicated geometry of the feasible region. Yet, formulation (1.3) is sometimes preferred in real-data modeling since one can always control the noise level of the model by tuning the acceptable tolerance-the parameter. A potentially feasible approach for solving problem (1.3) is the recently developed level-set method while the level-set method is based on iteratively solving the regularized least squares problem with different μ. Therefore, it is of great significance to design efficient algorithms for composite optimization problem (1.1).

However,the existing algorithms can roughly be divided into two categories. The first category consists of algorithms that use only the gradient information. For example, proximal gradient, ADMM, primal-dual, etc. The advantages of first order methods are they are easy to implement, converge fast to a solution with moderate accuracy. However, most of them has slow tail convergence and may fail on slightly more challenging problems. More importantly, the sparsity and low-rank property of the algorithms are not implemented in the iteration precess. The other category is the second order methods such as Newton type methods. The advantages of these methods are they usually converge fast to a solution with high accuracy but they are not easy to implement and may encounter numerical difficulties. Moreover, nearly all of these second order information based solvers rely on certain nondegeneracy assumptions to guarantee the nonsingularity of the corresponding inner linear systems.

When $p(\boldsymbol{x}) = \|\boldsymbol{x}\|_1$, the standard way to solve (1.3) is via linear programming techniques [16] since it is well known that it can be recast as a linear program [17]. Some typical modern solvers rely on interior-point methods (IPM) which are somewhat problematic for large scale problems, since they do not scale well with size. There are some solvers that can handle (1.3) using proximal operator. However, they are designed case by case, it is not user-friendly if the users want to change the model such as adding linear, quadratic term, or shift term $p_i(\boldsymbol{x}_i - \boldsymbol{b}_i)$.

1.1 contribution

The contribution of this paper is listed as follows:

- Generalization: The software packages designed by [2, 8] cannot handle problems such as quadratic programming (QP) and semidefinite programming (SDP) with nonnegative constraints.
- Unification: Although the algorithms proposed in [2, 8] can handle many types of problems, their implementations are designed on a case-by-case basis.
- Second-order algorithms: To the best of our knowledge, we are the first to develop a MATLAB software package incorporating second-order algorithms to solve a wide range of optimization problems.

2 Examples

{2}

The classical models that (1.1) includes are classified as follows. Consider the following convex optimization problems with constraint:

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \|\boldsymbol{x}\|_1, \\ \text{s.t.} \quad & \|\mathcal{B}\boldsymbol{x} - \boldsymbol{b}\|_\infty < \lambda_1. \end{aligned} \tag{2.1}$$ {prob:linf}

For the ℓ_1 constrainted problem:

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \|\mathcal{A}\boldsymbol{x} - \boldsymbol{b}\|_2, \\ \text{s.t.} \quad & \|\boldsymbol{x}\|_1 \leq \lambda_1, \end{aligned} \tag{2.2}$$

{pro-l1con}

For constrained Lasso type problems such as (2.2), we note that our algorithm can solve it directly instead of solving a series of subproblems of level set method [6].

We note that we focus on the conic programing with the following structure:

$$\min_{\boldsymbol{x}} \langle \boldsymbol{c}, \boldsymbol{x} \rangle, \quad \text{s.t. } \boldsymbol{x} \in \mathbb{S}_+^n, \ \mathtt{bl} \leq \mathcal{A}\boldsymbol{x} \leq \mathtt{bu}, \ \mathtt{l} \leq \boldsymbol{x} \leq \mathtt{u}. \tag{2.3}$$

{conic-general}

Model (2.3) also includes SDP with affian constraints such as the relaxation SDP problem arising from BIQ and clustering problems (RCP).

When $p(\boldsymbol{x}) = \delta_{\mathcal{K}}(\boldsymbol{x})$ denotes the nonnegative cone, semidefinite cone, or second-order cone, the corresponding problem is linear programming (LP), semidefinite programming (SDP), or second-order cone programming (SOCP) respectively, i.e. the following classical conic programming:

$$\min_{\boldsymbol{x}} \langle \boldsymbol{c}, \boldsymbol{x} \rangle \quad \text{s.t. } \mathcal{A}\boldsymbol{x} = \boldsymbol{b}, \ \boldsymbol{x} \in \mathbb{S}_+^n. \tag{2.4}$$

{pro:conic}

If h, p are two singleton sets and $\mathcal{P}_2 = \{\boldsymbol{b}\}$, then it corresponds to the classical quadratic programming (QP):

$$\min_{\boldsymbol{x}} \frac{1}{2} \langle \boldsymbol{x}, \boldsymbol{Q}(\boldsymbol{x}) \rangle + \langle \boldsymbol{c}, \boldsymbol{x} \rangle \ \text{ s.t. } \mathcal{A}\boldsymbol{x} = \boldsymbol{b}, \ \mathtt{l} \leq \boldsymbol{x} \leq \mathtt{u}. \tag{2.5}$$

{QP}

It is noted in [1] that it is more welcomed to transform QP into second order conic programming. This transformation is numerically more robust than the one for quadratic problems. However, in this paper, we consider solving the dual of (2.5). This approach can make use of the sparsity of the problem. Furthermore, it can be extended to solve the ℓ_1-QP problem directly.

$$\min_{\boldsymbol{x}} \frac{1}{2} \langle \boldsymbol{x}, Q\boldsymbol{x} \rangle + \|\boldsymbol{x}\|_1 \text{ s.t. } \mathcal{A}\boldsymbol{x} = \boldsymbol{b}, \ \mathtt{l} \leq \boldsymbol{x} \leq \mathtt{u}. \tag{2.6}$$

{l1-QP}

When $Q = 0$ and $\boldsymbol{c} = 0$, the corresponding two-block composite optimization is:

$$\min_{\boldsymbol{x}} \quad \frac{1}{2}\|\mathcal{A}\boldsymbol{x} - \boldsymbol{b}\|^2 + p(\boldsymbol{x}), \tag{2.7}$$

{comp-lasso}

Specially, when $f(\boldsymbol{x}) = \frac{1}{2}\|\mathcal{A}\boldsymbol{x} - \boldsymbol{b}\|^2$ and $p(\boldsymbol{x})$ is a nonnegative positively homogeneous convex function such that $p(0) = 0$, i.e. a gauge function. Model (2.7) covers typical problems that arise in statistical learning. We list a few of them as follows:

- When $p(\boldsymbol{x}) = \lambda\|\boldsymbol{x}\|_1$, the problem corresponds to the classical Lasso problem.
- When $p(\boldsymbol{x}) = \lambda_1\|B\boldsymbol{x}\|_1 + \lambda_2\|\boldsymbol{x}\|_1$, where $B\boldsymbol{x} = [\boldsymbol{x}_2 - \boldsymbol{x}_1, \cdots, \boldsymbol{x}_n - \boldsymbol{x}_{n-1}]^\top \in \mathbb{R}^{n-1}$, then the problem corresponds to the classical Fused Lasso problem.
- When $p(\boldsymbol{x}) = \lambda_1\|\boldsymbol{x}\|_1 + \lambda_2 \sum_{l=1}^{G} w_l \|\boldsymbol{x}_{G_l}\|_2$, then the problem corresponds to the classical Group Lasso problem.

In addition to Lasso type problems, the robust PCA problem for video segmentation problem can be represented by:

$$\min_{X} \|\boldsymbol{X}\|_* + \|\boldsymbol{D} - \boldsymbol{X}\|_1, \tag{2.8}$$

{RPCA}

where $\boldsymbol{X}, \boldsymbol{D} \in \mathbb{R}^{m\times n}$, Model (1.1) also covers problems arising in machine learning. The convex clustering model is presented as:

$$\min_{X \in \mathbb{R}^{d\times n}} \frac{1}{2} \sum_{i=1}^{n} \|\boldsymbol{x}_i - \boldsymbol{a}_i\|^2 + \gamma \sum_{(i,j)\in\mathcal{E}} w_{ij}\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_q, \tag{2.9}$$

{cluster}

where $\boldsymbol{X} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n]$, denotes the calssification feature, $\gamma > 0$ is a tuning parameter, $\mathcal{E} = \cup_{i=1}^{n}\{(i,j)|j \text{ is } i\text{'s } k\text{-nearest neighbors}, i < j \leq n\}$ is the edge set. Typically, p is chosen to be $1, 2$ or ∞. After solving (2.9) and obtaining the optimal solution $\boldsymbol{X}^* = [\boldsymbol{x}_1^*, \cdots, \boldsymbol{x}_n^*]$, we assign the data vector $\boldsymbol{a}_i$ and $\boldsymbol{a}_j$ to the same cluster if and only if $\boldsymbol{x}_i^* = \boldsymbol{x}_j^*$. In other words, $\boldsymbol{x}_i^*$ is the centroid for observation $\boldsymbol{a}_i$. Sparse PCA has the following form:

{pro:sparse}

$$\min_{\boldsymbol{x}} -\langle \boldsymbol{L}, \boldsymbol{x}\rangle + \lambda\|\boldsymbol{x}\|_1, \text{ s.t. } \mathrm{Tr}(\boldsymbol{x}) = 1, \boldsymbol{x} \succeq 0. \tag{2.10}$$

The problems examples are summarized in Table 2.

f	atom	f^*	$\text{prox}_{\lambda p}(\boldsymbol{x})[\boldsymbol{u}]$	$\partial\text{prox}_{\lambda p}(\boldsymbol{x})$ or $\nabla p^*(\boldsymbol{x})$	Assumptions
$\lambda\|\boldsymbol{x}\|^2$	square	$\frac{1}{4\lambda}\|\boldsymbol{y}\|^2$	-	$2\lambda\boldsymbol{x}$	-
$\lambda\sum_{i=1}^n e^{\boldsymbol{x}_i}$	exp	$\sum_{i=1}^n \boldsymbol{y}_i\log\boldsymbol{y}_i - \boldsymbol{y}_i$	-	$\lambda\log(\boldsymbol{x}/\lambda)$	$(\text{dom}(f^*) = \mathbb{R}^n_+)$
$-\lambda\sum_{i=1}^n \log\boldsymbol{x}_i$	log	$-n - \sum_{i=1}^n \log(-\boldsymbol{y}_i)$	-	$[\frac{-\lambda}{\log(\boldsymbol{x})_1}, \cdots, \frac{-\lambda}{\log(\boldsymbol{x})_n}]$	$(\text{dom}(f^*) = \mathbb{R}^n_+)$
$\lambda\log(\sum_{i=1}^n e^{\boldsymbol{x}_i})$	exp	$\sum_{i=1}^n \boldsymbol{y}_i\log(\boldsymbol{y}_i)$, dom = Δ_n.	-	$[e^{\boldsymbol{x}_1}, \cdots, e^{\boldsymbol{x}_n}]/\sum_{i=1}^n e^{\boldsymbol{x}_i}$	$(\text{dom}(f^*) = \mathbb{R}^n_+)$
$\lambda\|\boldsymbol{x}\|_1$	l1	$\delta_{B_{\|\cdot\|_\infty}[0,\lambda]}(\boldsymbol{y})$	$(\|\boldsymbol{x}\| - \lambda\boldsymbol{e})_+ \odot \text{sgn}(\boldsymbol{x})$	$\text{Diag}(\boldsymbol{u})$, $\boldsymbol{u}_i = \begin{cases} 0, & \text{if } \|(\boldsymbol{x})_i\| < \lambda, \\ 1, & \text{otherwise.} \end{cases}$	-
$\lambda\|\boldsymbol{x}\|_2$	l2	$\delta_{B_{\|\cdot\|_2}[0,\lambda]}(\boldsymbol{y})$	$\begin{cases} \boldsymbol{x} - \lambda\boldsymbol{x}/\|\boldsymbol{x}\|, & \text{if } \|\boldsymbol{x}\| > \lambda, \\ 0, & \text{otherwise.} \end{cases}$	$\begin{cases} I - \lambda(I - \boldsymbol{x}\boldsymbol{x}^{\mathrm{T}}/\|\boldsymbol{x}\|^2)/\|\boldsymbol{x}\|, & \text{if } \|\boldsymbol{x}\| > \lambda, \\ 0, & \text{otherwise.} \end{cases}$	-
$\lambda\|\boldsymbol{x}\|_\infty$	linfty	$\delta_{B_{\|\cdot\|_1}[0,\lambda]}(\boldsymbol{y})$	$\boldsymbol{x} - \lambda P_{B_{\|\cdot\|_1}[0,1]}(\boldsymbol{x}/\lambda)$	$\text{Diag}(u)$, $u_i = \begin{cases} 0, & \text{if } \|\boldsymbol{x}/\lambda\| > \mu_*, \text{where } \mu_* \\ & \text{satisfy } \mathbf{1}^{\mathrm{T}}[\boldsymbol{x} - \mu_*\mathbf{1}]_+ = \mathbf{1} \\ 1, & \text{otherwise.} \end{cases}$	-
$\delta_{\mathsf{l}\le\boldsymbol{x}\le\mathsf{u}}(\boldsymbol{x})$	box	$\langle \mathsf{u}, \max\{\boldsymbol{x}, 0\}\rangle + \langle \mathsf{l}, \min\{\boldsymbol{x}, 0\}\rangle$	$P_{\mathsf{l}\le\boldsymbol{x}\le\mathsf{u}}(\boldsymbol{x})$	$\text{Diag}(u), u_i = \begin{cases} 1, & \text{if } x_i/\lambda \in C, \\ 0, & \text{otherwise.} \end{cases}$	
$\delta_{\boldsymbol{x}\in\mathcal{K}}(\boldsymbol{x})$	cone	$\delta_{\boldsymbol{y}\in\mathcal{K}}(-\boldsymbol{y})$	$P_{\mathcal{K}}(\boldsymbol{x})$	Depend on $\mathcal{K}$	
$\lambda\max\{\boldsymbol{x}_i\}$	max	$\delta_{\Delta_n}(\boldsymbol{y})$	$\boldsymbol{x} - \lambda P_{\Delta_n}(\boldsymbol{x}/\lambda)$	$\text{Diag}(\boldsymbol{u}), \boldsymbol{u}_i = \begin{cases} 0, & \text{if } \boldsymbol{x}_i/\lambda \in C, \\ 1, & \text{otherwise.} \end{cases}$	-
$\lambda\sum_{i=1}^k \boldsymbol{x}_{[i]}$	topk	-	$x - \lambda P_{C_{e,k}}(\boldsymbol{x}/\lambda)$, $C = H_{e,k} \cap \text{Box}[\boldsymbol{0}, \boldsymbol{e}]$	$\text{Diag}(\boldsymbol{u}), \boldsymbol{u}_i = \begin{cases} 0, & \text{if } \boldsymbol{x}_i/\lambda \in C, \\ 1, & \text{otherwise.} \end{cases}$	-
$\lambda\sum_{k=1}^n \|\boldsymbol{x}_{[i]}\|$	topkabs	-	$\boldsymbol{x} - \lambda P_C(\boldsymbol{x}/\lambda)$ $C = B_{\|\cdot\|_1,[0,k]} \cap \text{Box}[-e, e]$	$\text{Diag}(\boldsymbol{u}), \boldsymbol{u}_i = \begin{cases} 0, & \text{if } \boldsymbol{x}_i/\lambda \in C, \\ 1, & \text{otherwise.} \end{cases}$	-
$\lambda H_\mu(\boldsymbol{x})$	huber	-	-	$\lambda\boldsymbol{x}$	-
$\lambda_1\|\boldsymbol{x}\|_1 + \lambda_2\|B\boldsymbol{x}\|_1$	fused	$\delta_{\|\boldsymbol{y}\|_\infty<\lambda_1}(\boldsymbol{y}) + \delta_{\|B^{\mathrm{T}}\boldsymbol{y}\|_\infty<\lambda_2}(\boldsymbol{y})$	(4.6)	(4.7)	-

f	atom	f^*	$\text{prox}_{\lambda p}(\boldsymbol{x})[\boldsymbol{u}]$	$\partial\text{prox}_{\lambda p}(\boldsymbol{x})$ or $\nabla p^*(\boldsymbol{x})$	Assumptions
$\delta_{\|\boldsymbol{x}\|_1\le\lambda}(\boldsymbol{x})$	l1con	$\lambda\|\boldsymbol{y}\|_\infty$	$\lambda P_{B_{\|\cdot\|_1}[0,1]}(\boldsymbol{x}/\lambda)$	$\text{diag}(u), \boldsymbol{u}_i = \begin{cases} 0, & \text{if } \|\boldsymbol{x}/\lambda\| > \mu_*, \text{where } \mu_* \\ & \text{satisfy } \mathbf{1}^{\mathrm{T}}[\boldsymbol{x}-\mu_*\mathbf{1}]_+ = \mathbf{1} \\ 1, & \text{otherwise.} \end{cases}$	-
$\delta_{\|\boldsymbol{x}\|_2\le\lambda}(\boldsymbol{x})$	l2com	$\lambda\|\boldsymbol{y}\|_2$	$\begin{cases} \lambda\boldsymbol{x}/\|\boldsymbol{x}\|, & \text{if } \|\boldsymbol{x}\| > \lambda, \\ \boldsymbol{x}, & \text{otherwise.} \end{cases}$	$\lambda\log(\boldsymbol{x}/\lambda)$	$(\text{dom}(f^*) = \mathbb{R}^n_+)$
$\delta_{\|\boldsymbol{x}\|_\infty\le\lambda}(\boldsymbol{x})$	linftycon	$\lambda\|\boldsymbol{y}\|_1$	$P_{\|\boldsymbol{x}\|_\infty\le\lambda}(\boldsymbol{x})$	$[e^{\boldsymbol{x}_1},\cdots,e^{\boldsymbol{x}_n}]/\sum_{i=1}^n e^{\boldsymbol{x}_i}$	$(\text{dom}(f^*) = \mathbb{R}^n_+)$
$\lambda\|\boldsymbol{X}\|_1$	l1	$\delta_{B_{\|\cdot\|_\infty}[0,\lambda]}(\boldsymbol{Y})$	$(\|\boldsymbol{X}\| - \lambda\boldsymbol{E})_+ \odot \text{sgn}(\boldsymbol{X})$	$\boldsymbol{U}_{i,j} = \begin{cases} 0, & \text{if } \|(\boldsymbol{X})_{i,j}\| < \lambda, \\ 1, & \text{otherwise.} \end{cases}$	-
$\lambda\|\boldsymbol{X}\|_F^2$	frobenius	$\frac{1}{4\lambda}\|\boldsymbol{Y}\|_{\mathrm{F}}^2$	-	$2\lambda\boldsymbol{X}$	-
$\lambda\|\boldsymbol{X}\|_F$	frobenius	$\delta_{B_{\|\cdot\|_{\mathrm{F}}}[0,\lambda]}(\boldsymbol{Y})$	$\left(1 - \frac{\lambda}{\max\{\|\boldsymbol{X}\|_F,\lambda\}}\right)\boldsymbol{X}$	$\begin{cases} I - \lambda(I - \frac{\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}}}{\|\boldsymbol{X}\|_{\mathrm{F}}^2})/\|\boldsymbol{X}\|_{\mathrm{F}}, & \text{if } \|(\boldsymbol{X})_i\| < \lambda, \\ 0, & \text{otherwise.} \end{cases}$	-
$\lambda\|\boldsymbol{X}\|_*$	nuclear	$\delta_{B_{\|\cdot\|_{S_\infty}}[0,\lambda]}(\boldsymbol{Y})$	(4.2)	(4.3)	-
$\lambda\|\boldsymbol{X}\|_{1,2}$	l1l2	$\delta_{\|Y\|_{\infty,2}<\lambda}(\boldsymbol{Y})$	$[\max\left(0, 1 - \frac{\lambda}{\|X_1\|_2}\right)X_1,\cdots,\max\left(0, 1 - \frac{\lambda}{\|X_n\|_2}\right)X_n]$	$\begin{cases} I - \lambda(I - \boldsymbol{X_i}\boldsymbol{X_i}^{\mathrm{T}}/\|\boldsymbol{X}_i\|^2)/\|\boldsymbol{X}_i\|, & \text{if } \|\boldsymbol{X}_i\| > \lambda, \\ 0, & \text{otherwise.} \end{cases}$	-
$\lambda\|\boldsymbol{X}\|_{1,\infty}$	l1linfty	$\delta_{\|Y\|_{\infty,1}<\lambda}(\boldsymbol{Y})$	$\boldsymbol{X}_i - \lambda P_{B_{\|\cdot\|_1}[0,1]}(\boldsymbol{X}_i/\lambda)$	$u_{i,j} = \begin{cases} 0, & \text{if } \|\boldsymbol{X}_j/\lambda\| > \mu_{*,j}, \text{where } \mu_* \\ & \text{satisfy } \mathbf{1}^{\mathrm{T}}[\boldsymbol{x}-\mu_{*,j}\mathbf{1}]_+ = \mathbf{1} \\ 1, & \text{otherwise.} \end{cases}$	-
$-\lambda\log\det(\boldsymbol{X})$	logdet	$-n - \log\det(-\boldsymbol{Y})$	$U\text{diag}\left(\frac{\lambda_j(X)+\sqrt{\lambda_j(X)^2+4\lambda}}{2}\right)\boldsymbol{U}^T$	-	$\mathbb{S}^n_{--}$
$\lambda\sigma_1(\boldsymbol{X})$	mmax	$\delta_{B_{\|\cdot\|_*}[0,\lambda]}(\boldsymbol{Y})$	$\boldsymbol{U}\text{diag}\left(\lambda(\boldsymbol{X}) - \lambda P_{\Delta_n}(\lambda(\boldsymbol{X})/\lambda)\right)\boldsymbol{V}^T$	-	γ_n

Table 1. Combined table of functions, their duals, proximal operators, and subdifferentials

{combined-table}

Problem Type	Objective Function	Constraints	Function block	Remarks
General Problem	$\underbrace{\langle \boldsymbol{c}, \boldsymbol{x}\rangle}_{(1)} + \underbrace{\frac{1}{2}\langle \boldsymbol{x}, \mathcal{Q}(\boldsymbol{x})\rangle}_{(2)} + \underbrace{f(\mathcal{B}(\boldsymbol{x}))}_{(3)} + p(\boldsymbol{x})$	$\underbrace{\boldsymbol{x} \in \mathcal{P}_1}_{(4)}, \underbrace{\mathcal{A}(\boldsymbol{x}) \in \mathcal{P}_2}_{(5)}.$	(1) (2) (3) (4) (5)	Handles various optimization problems
Conic programming	$\langle \boldsymbol{c}, \boldsymbol{x}\rangle$	$\mathcal{A}(\boldsymbol{x}) = \boldsymbol{b}, \boldsymbol{x} \geq 0.$	(1)(5)	Linear programming
	$\langle \boldsymbol{C}, \boldsymbol{X}\rangle$	$\mathcal{A}(\boldsymbol{X}) = \boldsymbol{b}, \boldsymbol{X} \in \mathbb{S}^n_+.$	(1)(5)	semidefinite programming
	$\langle \boldsymbol{c}, \boldsymbol{x}\rangle$	$\mathcal{A}(\boldsymbol{x}) = \boldsymbol{b}, \boldsymbol{x} \in \mathcal{Q}^n.$	(1)(5)	quadratic cone programming
SDP with box constraints	$\langle \boldsymbol{C}, \boldsymbol{X}\rangle$	$\mathcal{A}(\boldsymbol{X}) = \boldsymbol{b}, \boldsymbol{x} \in \mathcal{P}_1, \boldsymbol{X} \in \mathbb{S}^n_+.$	(1)(4)(5)	SDP with box constraints
Lasso type Problems	$\frac{1}{2}\|\mathcal{B}(\boldsymbol{x}) - \boldsymbol{b}\|^2 + \lambda\|\boldsymbol{x}\|_1$	-	(3)	Lasso problem
	$\frac{1}{2}\|\mathcal{B}(\boldsymbol{x}) - \boldsymbol{b}\|^2 + \lambda_1\|\boldsymbol{x}\|_1 + \lambda_2\|D\boldsymbol{x}\|_1$	-	(3)	Fused lasso problem
	$\frac{1}{2}\|\mathcal{B}(\boldsymbol{x}) - \boldsymbol{b}\|^2 + \lambda\|\boldsymbol{x}\|_2$	-	(3)	Group Lasso
	$\frac{1}{2}\|\mathcal{B}(\boldsymbol{x}) - \boldsymbol{b}\|^2 + \lambda\sum_{i=1}^k \boldsymbol{x}_{[i]}$	-	(3)	Top-k regression
Matrix Completion	$\|\mathcal{B}(\boldsymbol{X}) - \boldsymbol{B}\|_{\mathrm{F}}^2 + \lambda\|\boldsymbol{X}\|_*$	-	(3)	Low-rank matrix recovery
QP	$\langle \boldsymbol{x}, \mathcal{Q}(\boldsymbol{x})\rangle + \langle \boldsymbol{x}, \boldsymbol{c}\rangle$	$\mathrm{l} \leq \boldsymbol{x} \leq \mathrm{u}, \mathcal{A}(\boldsymbol{x}) = \boldsymbol{b}.$	(1)(2)(4)(5)	Quadratic Programming
QP with regularizer	$\langle \boldsymbol{x}, \mathcal{Q}(\boldsymbol{x})\rangle + \lambda\|\boldsymbol{x}\|_1$	$\mathrm{l} \leq \boldsymbol{x} \leq \mathrm{u}, \mathcal{A}(\boldsymbol{x}) = \boldsymbol{b}.$	(1)(2)(3)(5)	QP with ℓ_1 norm
convex constraint problems	$\|\boldsymbol{x}\|_1$	$\|\mathcal{B}\boldsymbol{x} - \boldsymbol{b}\|_\infty < \lambda$	(3)	ℓ_∞ constraint problem
	$\|\mathcal{B}(\boldsymbol{x}) - \boldsymbol{b}\|_1$	$\|\boldsymbol{x}\|_1 < \lambda$	(3)	ℓ_1 constraint problem
Statistical learning	$-\langle \boldsymbol{L}, \boldsymbol{x}\rangle + \lambda\|\boldsymbol{x}\|_1,$	$\mathrm{Tr}(\boldsymbol{x}) = 1, \boldsymbol{x} \succeq 0$	(3)	Sparse PCA
	$\|\boldsymbol{x}\|_1$	$\mathcal{A}(\boldsymbol{x}) = \boldsymbol{b},$	(5)	Base pursuit
	$\|\boldsymbol{x}_1\|_* + \mu\|\boldsymbol{x}_2\|_1$	$\boldsymbol{x}_1 + \boldsymbol{x}_2 = \boldsymbol{D}$	(3)(5)	Roubst PCA
	$-\log(\det(X)) + \mathrm{Tr}(XS) + \lambda\|X\|_1$	-	(1)(3)	Sparse covariance matrix estimation
	$\frac{1}{2}\sum_{i=1}^n \|\boldsymbol{x}_i - \boldsymbol{a}_i\|^2 + \gamma\sum_{(i,j)\in\mathcal{E}} w_{ij}\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_q$	-	(3)	convex clustering problem

Table 2. The examples of problems Model (1.1) able to solve.

{tabel-problem-summari

3 A primal dual semismooth Newton method for (1.1)

In this section, using the AL duality, we first transform the original problem (1.1) into a saddle point problem. Subsequently, a monotone nonlinear system induced derived from the saddle point problem is presented. Furthermore, such nonlinear system is semismooth and equivalent to the Karush–Kuhn–Tucker (KKT) optimality condition of problem (1.1).

3.1 An equivalent saddle point problem

The dual of model (1.1) can be represented by

$$\begin{aligned} \min_{\boldsymbol{y},\boldsymbol{z},\boldsymbol{s},\boldsymbol{r},\boldsymbol{v}} \quad & \delta^*_{\mathcal{P}_2}(-\boldsymbol{y}) + f^*(-\boldsymbol{z}) + p^*(-\boldsymbol{s}) + \frac{1}{2}\langle \mathcal{Q}\boldsymbol{v}, \boldsymbol{v}\rangle + \delta^*_{\mathcal{P}_1}(-\boldsymbol{r}), \\ \text{s.t.} \quad & \mathcal{A}^*\boldsymbol{y} + \mathcal{B}^*\boldsymbol{z} + \boldsymbol{s} - \mathcal{Q}\boldsymbol{v} + \boldsymbol{r} = \boldsymbol{c}. \end{aligned} \tag{3.1}$$

Introducing the slack variable, the corresponding optimization problem is

$$\begin{aligned} \min_{\boldsymbol{y},\boldsymbol{z},\boldsymbol{s},\boldsymbol{r},\boldsymbol{v}} \quad & \delta^*_{\mathcal{P}_2}(-\boldsymbol{p}) + f^*(-\boldsymbol{q}) + p^*(-\boldsymbol{s}) + \frac{1}{2}\langle \mathcal{Q}\boldsymbol{v}, \boldsymbol{v}\rangle + \delta^*_{\mathcal{P}_1}(-\boldsymbol{t}), \\ \text{s.t.} \quad & \mathcal{A}^*\boldsymbol{y} + \mathcal{B}^*\boldsymbol{z} + \boldsymbol{s} - \mathcal{Q}\boldsymbol{v} + \boldsymbol{r} = \boldsymbol{c},\ \boldsymbol{y} = \boldsymbol{p},\ \boldsymbol{z} = \boldsymbol{q},\ \boldsymbol{r} = \boldsymbol{t}. \end{aligned} \tag{3.2}$$

The corresponding augment Lagrangian function of (3.1) is

$$\begin{aligned} \mathcal{L}_\sigma(\boldsymbol{y},\boldsymbol{s},\boldsymbol{z},\boldsymbol{r},\boldsymbol{v},\boldsymbol{x}_1,\boldsymbol{x}_2,\boldsymbol{x}_3,\boldsymbol{x}_4) &= \delta^*_{\mathcal{P}_2}(-\boldsymbol{p}) + f^*(-\boldsymbol{q}) + p^*(-\boldsymbol{s}) + \frac{1}{2}\langle \mathcal{Q}(\boldsymbol{v}), \boldsymbol{v}\rangle \\ &+ \delta^*_{\mathcal{P}_1}(-\boldsymbol{t}) + \frac{\sigma}{2}\left(\|\boldsymbol{p} - \boldsymbol{y} + \boldsymbol{x}_1/\sigma\|_{\mathrm{F}}^2 + \|\boldsymbol{q} - \boldsymbol{z} + \frac{1}{\sigma}\boldsymbol{x}_2\|_{\mathrm{F}}^2 + \|\boldsymbol{t} - \boldsymbol{r} + \frac{1}{\sigma}\boldsymbol{x}_3\|^2\right) \\ &+ \frac{\sigma}{2}(\|\mathcal{A}^*\boldsymbol{y} + \mathcal{B}^*\boldsymbol{z} + \boldsymbol{s} - \mathcal{Q}\boldsymbol{v} + \boldsymbol{r} - \boldsymbol{c} + \frac{1}{\sigma}\boldsymbol{x}_4\|_{\mathrm{F}}^2) - \frac{1}{2\sigma}\sum_{i=1}^4 \|\boldsymbol{x}_i\|^2. \end{aligned}$$

Eliminating the variables $\boldsymbol{p}, \boldsymbol{q}, \boldsymbol{s}, \boldsymbol{t}$:

$$\begin{aligned} &\boldsymbol{p} = \mathrm{prox}_{\delta^*_{\mathcal{P}_2}/\sigma}(\boldsymbol{x}_1/\sigma - \boldsymbol{y}), \quad \boldsymbol{q} = \mathrm{prox}_{f^*/\sigma}(\boldsymbol{x}_2/\sigma - \boldsymbol{z}), \\ &\boldsymbol{s} = \mathrm{prox}_{p^*/\sigma}(\mathcal{A}^*\boldsymbol{y} + \mathcal{B}^*\boldsymbol{z} - \mathcal{Q}\boldsymbol{v} + \boldsymbol{r} - \boldsymbol{c} + \frac{1}{\sigma}\boldsymbol{x}_4), \quad \boldsymbol{t} = \mathrm{prox}_{\delta^*_{\mathcal{P}_1}/\sigma}(\boldsymbol{x}_3/\sigma - \boldsymbol{r}). \end{aligned} \tag{3.3}$$

Let $\boldsymbol{w} = (\boldsymbol{y}, \boldsymbol{z}, \boldsymbol{r}, \boldsymbol{v}, \boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, \boldsymbol{x}_4)$ and then we obtain that the modified augment Lagrangian function:

$$\begin{aligned} \Phi_\sigma(\boldsymbol{w}) &= \delta^*_{\mathcal{P}_2}(\mathrm{prox}_{\delta^*_{\mathcal{P}_2}/\sigma}(\boldsymbol{x}_1/\sigma - \boldsymbol{y})) + f^*(\mathrm{prox}_{f^*/\sigma}(\boldsymbol{x}_2/\sigma - \boldsymbol{z})) \\ &+ \frac{1}{2}\langle \mathcal{Q}\boldsymbol{v}, \boldsymbol{v}\rangle + \delta^*_{\mathcal{P}_1}(\mathrm{prox}_{\delta^*_{\mathcal{P}_1}}(\boldsymbol{x}_3/\sigma - \boldsymbol{t})) \\ &+ \frac{1}{2\sigma}\|\Pi_{\mathcal{P}_2}(\boldsymbol{x}_1 - \sigma\boldsymbol{y})\|^2 + p^*(\mathrm{prox}_{p^*/\sigma}(\boldsymbol{x}_4/\sigma - \mathcal{A}^*\boldsymbol{y} - \mathcal{B}^*\boldsymbol{z} - \mathcal{Q}\boldsymbol{v} - \boldsymbol{r})) \\ &+ \frac{1}{2\sigma}\|\mathrm{prox}_{\sigma p}(\boldsymbol{x}_4 + \sigma(\mathcal{A}^*\boldsymbol{y} + \mathcal{B}^*\boldsymbol{z} - \mathcal{Q}\boldsymbol{v} + \boldsymbol{r} - \boldsymbol{c}))\|^2 + \frac{1}{2\sigma}\|\Pi_{\mathcal{P}_1}(\boldsymbol{x}_3 - \sigma\boldsymbol{r})\|^2 \\ &+ \frac{1}{2\sigma}\|\mathrm{prox}_{\sigma f}(\boldsymbol{x}_2 - \sigma\boldsymbol{z})\|^2 - \frac{1}{2\sigma}\sum_{i=1}^4 \|\boldsymbol{x}_i\|^2. \end{aligned} \tag{3.4}$$

Then the corresponding saddle point problem is presented as

$$\min_{\boldsymbol{y},\boldsymbol{z},\boldsymbol{r},\boldsymbol{v}} \max_{\boldsymbol{x}_1,\boldsymbol{x}_2,\boldsymbol{x}_3,\boldsymbol{x}_4} \Phi(\boldsymbol{y},\boldsymbol{z},\boldsymbol{r},\boldsymbol{v};\boldsymbol{x}_1,\boldsymbol{x}_2,\boldsymbol{x}_3,\boldsymbol{x}_4). \tag{3.5}$$

Assumption 1. *The dual problem (1.1) has an optimal solution $\boldsymbol{y}_*, \boldsymbol{z}_*, \boldsymbol{s}_*$. Furthermore, Slater's condition holds for the dual problem (3.1), i.e., there exists $-\boldsymbol{y} \in \mathrm{ri}(\mathrm{dom}(\delta^*_{\mathcal{Q}}))$, $-\boldsymbol{s} \in \mathrm{ri}(\mathrm{dom}(\delta^*_{\mathcal{K}}))$, and $-\boldsymbol{z} \in \mathrm{ri}(\mathrm{dom}(h^*))$ such that $\mathcal{A}^*(\boldsymbol{y}) + \boldsymbol{s} + \boldsymbol{z} = \boldsymbol{c}$.*

Lemma 3.1 (Stong duality). *Suppose that Assumption 1 holds. Given $\sigma > 0$, the strong duality holds for (??), i.e.*

$$\min_{\boldsymbol{y},\boldsymbol{z}} \max_{\boldsymbol{x},\boldsymbol{u},\boldsymbol{q}} \Phi(\boldsymbol{y},\boldsymbol{z},\boldsymbol{x},\boldsymbol{u},\boldsymbol{q}) = \max_{\boldsymbol{x},\boldsymbol{u},\boldsymbol{q}} \min_{\boldsymbol{y},\boldsymbol{z}} \Phi(\boldsymbol{y},\boldsymbol{z},\boldsymbol{x},\boldsymbol{u},\boldsymbol{q}), \tag{3.6}$$ {lemma:strong}

where both sides of (3.6) are equivalent to problem (1.1).

The gradient of the saddle point problem can be represented by

$$\begin{aligned}
\nabla_{\boldsymbol{y}}\Phi_\sigma(\boldsymbol{w}) &= \mathcal{A}\mathrm{prox}_{\sigma p}(\boldsymbol{x}_4 + \sigma(\mathcal{A}^*\boldsymbol{y} + \mathcal{B}^*\boldsymbol{z} - \mathcal{Q}\boldsymbol{v} + \boldsymbol{r} - \boldsymbol{c})) - \Pi_{\mathcal{P}_2}(\boldsymbol{x}_1 - \sigma\boldsymbol{y}),\\
\nabla_{\boldsymbol{z}}\Phi_\sigma(\boldsymbol{w}) &= \mathcal{B}\mathrm{prox}_{\sigma p}(\boldsymbol{x}_4 + \sigma(\mathcal{A}^*\boldsymbol{y} + \mathcal{B}^*\boldsymbol{z} - \mathcal{Q}\boldsymbol{v} + \boldsymbol{r} - \boldsymbol{c})) - \mathrm{prox}_{\sigma f}(\boldsymbol{x}_2 - \sigma\boldsymbol{z}),\\
\nabla_{\boldsymbol{r}}\Phi_\sigma(\boldsymbol{w}) &= \mathrm{prox}_{\sigma p}(\boldsymbol{x}_4 + \sigma(\mathcal{A}^*\boldsymbol{y} + \mathcal{B}^*\boldsymbol{z} - \mathcal{Q}\boldsymbol{v} + \boldsymbol{r} - \boldsymbol{c})) - \Pi_{\mathcal{P}_1}(\boldsymbol{x}_3 - \sigma\boldsymbol{r}),\\
\nabla_{\boldsymbol{v}}\Phi_\sigma(\boldsymbol{w}) &= -\mathcal{Q}\mathrm{prox}_{\sigma p}(\boldsymbol{x}_4 + \sigma(\mathcal{A}^*\boldsymbol{y} + \mathcal{B}^*\boldsymbol{z} - \mathcal{Q}\boldsymbol{v} + \boldsymbol{r} - \boldsymbol{c})) + \mathcal{Q}\boldsymbol{v},\\
\nabla_{\boldsymbol{x}_1}\Phi_\sigma(\boldsymbol{w}) &= \frac{1}{\sigma}\Pi_{\mathcal{P}_2}(\boldsymbol{x}_1 - \sigma\boldsymbol{y}) - \frac{1}{\sigma}\boldsymbol{x}_1,\\
\nabla_{\boldsymbol{x}_2}\Phi_\sigma(\boldsymbol{w}) &= \frac{1}{\sigma}\mathrm{prox}_{\sigma f}(\boldsymbol{x}_2 - \sigma\boldsymbol{z}) - \frac{1}{\sigma}\boldsymbol{x}_2,\\
\nabla_{\boldsymbol{x}_3}\Phi_\sigma(\boldsymbol{w}) &= \frac{1}{\sigma}\Pi_{\mathcal{P}_1}(\boldsymbol{x}_3 - \sigma\boldsymbol{r}) - \frac{1}{\sigma}\boldsymbol{x}_3\\
\nabla_{\boldsymbol{x}_4}\Phi_\sigma(\boldsymbol{w}) &= \frac{1}{\sigma}\mathrm{prox}_{\sigma p}(\boldsymbol{x}_4 + \sigma(\mathcal{A}^*\boldsymbol{y} + \mathcal{B}^*\boldsymbol{z} - \mathcal{Q}\boldsymbol{v} + \boldsymbol{r} - \boldsymbol{c})) - \frac{1}{\sigma}\boldsymbol{x}_4.
\end{aligned} \tag{3.7}$$ {eqn:gradient}

We note that if f^* is differentable, then $\boldsymbol{x}_2$ is not exist and the corresponding gradient is $\nabla_{\boldsymbol{z}}\Phi_\sigma(\boldsymbol{w}) = \mathcal{B}\mathrm{prox}_{\sigma p}(\boldsymbol{x}_4 + \sigma(\mathcal{A}^*\boldsymbol{y} + \mathcal{B}^*\boldsymbol{z} - \mathcal{Q}\boldsymbol{v} + \boldsymbol{r} - \boldsymbol{c})) - \nabla f^*(-\boldsymbol{z})$. We make the following existence condition of $\boldsymbol{x}_1, \boldsymbol{x}_2$, and $\boldsymbol{x}_3$ that will be used in the subsequent analysis.

- $\boldsymbol{y}$ exist if and only if $\mathcal{P}_2$ is nontrival, $\boldsymbol{x}_1$ exist if and only if $\mathcal{P}_2$ is not a singleton set.
- $\boldsymbol{r}$ and $\boldsymbol{x}_3$ exist if and only if $\mathcal{P}_1$ is nontrival.
- $\boldsymbol{x}_2$ exist if and only if f is nonsmooth i.e. needs proximal operator.
- $\boldsymbol{v}$ exist if and only if $\mathcal{Q}$ is nontrival.

The nonlinear operator $F(\boldsymbol{w})$ is defined as

$$F(\boldsymbol{w}) = \left(\nabla_{\boldsymbol{y}}\Phi(\boldsymbol{w}); \nabla_{\boldsymbol{z}}\Phi(\boldsymbol{w}); \nabla_{\boldsymbol{r}}\Phi(\boldsymbol{w}); \nabla_{\boldsymbol{v}}\Phi(\boldsymbol{w}); -\nabla_{\boldsymbol{x}_1}\Phi(\boldsymbol{w}); -\nabla_{\boldsymbol{x}_2}\Phi(\boldsymbol{w}); -\nabla_{\boldsymbol{x}_3}\Phi(\boldsymbol{w}); -\nabla_{\boldsymbol{x}_4}\Phi(\boldsymbol{w})\right). \tag{3.8}$$ {eq:def:F}

{def:Jacobian}

Definition 3.2. [9] Let F be a locally Lipschitz continuous mapping. F is called semismooth at $\boldsymbol{x}$ if F is directional differentiable at $\boldsymbol{x}$ and for any $\boldsymbol{d}$ and $J \in \partial F(\boldsymbol{x} + \boldsymbol{d})$, it holds that

$$\|F(\boldsymbol{x} + \boldsymbol{d}) - F(\boldsymbol{x}) - J\boldsymbol{d}\| = o(\|\boldsymbol{d}\|), \quad \boldsymbol{d} \to 0.$$

F is said to be strongly semismooth at $\boldsymbol{x}$ if F is semismooth at $\boldsymbol{x}$ and

$$\|F(\boldsymbol{x} + \boldsymbol{d}) - F(\boldsymbol{x}) - J\boldsymbol{d}\| = O(\|\boldsymbol{d}\|^2), \quad \boldsymbol{d} \to 0.$$

We say F is semismooth (respectively, strongly semismooth) if F is semismooth (respectively, strongly semismooth) for any $\boldsymbol{x}$.

We first compute the generalized Jacobian of $F(\boldsymbol{w})$. Note that for a convex function h, its proximal operator prox_{th} is Lipschitz continuous. Then, by Definition 3.2, the following sets:

$$\begin{aligned} &D_{\Pi_1} := \partial\text{prox}(\boldsymbol{x}_3 - \sigma\boldsymbol{r}),\ D_{\Pi_2} = \partial\text{prox}(\boldsymbol{x}_1 - \sigma\boldsymbol{y}),\ D_f := \partial\text{prox}_{\sigma f}(\boldsymbol{x}_2 - \sigma\boldsymbol{z}), \\ &D_p := \partial\text{prox}_{\sigma p}(\boldsymbol{x}_4 + \sigma(\mathcal{A}^*\boldsymbol{y} + \mathcal{B}^*\boldsymbol{z} - \mathcal{Q}\boldsymbol{v} + \boldsymbol{r} - \boldsymbol{c})) \end{aligned} \tag{3.9}$$

Consequently, the corresponding generalized Jacboian can be represented by

$$\hat{\partial}F(\boldsymbol{w}) := \left\{ \begin{pmatrix} \mathcal{H}_{11} & \mathcal{H}_{12} \\ -\mathcal{H}_{12}^{\top} & \mathcal{H}_{22} \end{pmatrix} \right\}. \tag{3.10}$$

where

$$\begin{aligned} \mathcal{H}_{11} &= \sigma\left(\mathcal{A}, \mathcal{B}, \mathcal{I}, -\mathcal{Q}\right)^{\mathrm{T}} D_p\left(\mathcal{A}, \mathcal{B}, \mathcal{I}, -\mathcal{Q}\right) + \sigma\text{blkdiag}(D_{\Pi_1}, D_f, D_{\Pi_2}, Q), \\ \mathcal{H}_{12} &= \left[\left(-\text{blkdiag}\left(\left[D_{\Pi_1}, D_f, D_{\Pi_2}\right]\right); \mathbf{0}\right), \left(\mathcal{A}, \mathcal{B}, \mathcal{I}, -\mathcal{Q}\right)^{\mathrm{T}} D_{\boldsymbol{x}_4}\right], \\ \mathcal{H}_{22} &= \text{blkdiag}\left\{\frac{1}{\sigma}(\mathcal{I} - D_{\Pi_1}), \frac{1}{\sigma}(\mathcal{I} - D_h), \frac{1}{\sigma}(\mathcal{I} - D_{\Pi_2}), \frac{1}{\sigma}(\mathcal{I} - D_p)\right\}. \end{aligned} \tag{3.11}$$

The linear system can be represented by:

$$\begin{pmatrix} \mathcal{H}_{11} + \tau\mathcal{I} & \mathcal{H}_{12} \\ -\mathcal{H}_{12}^{T} & \mathcal{H}_{22} + \tau\mathcal{I} \end{pmatrix} \begin{pmatrix} \boldsymbol{d}_1 \\ \boldsymbol{d}_2 \end{pmatrix} = \begin{pmatrix} -F_1 \\ -F_2 \end{pmatrix}, \tag{3.12}$$

For a given $\boldsymbol{d}_1$, the direction $\boldsymbol{d}_2$ can be calculated by

$$\boldsymbol{d}_2 = (\mathcal{H}_{22} + \tau\mathcal{I})^{-1}(\mathcal{H}_{12}^{\top}\boldsymbol{d}_1 - F_2). \tag{3.13}$$

Hence, (3.12) can be reduced to a linear system with respect to $\boldsymbol{d}_1$:

$$\widetilde{\mathcal{H}}_{11}\boldsymbol{d}_1 = -\widetilde{F}_1, \tag{3.14}$$

where $\widetilde{F}_1 := \mathcal{H}_{12}(\mathcal{H}_{22} + \tau\mathcal{I})^{-1}F_2 - F_1$ and $\widetilde{\mathcal{H}}_{11} := (\mathcal{H}_{11} + \mathcal{H}_{12}(\mathcal{H}_{22} + \tau\mathcal{I})^{-1}\mathcal{H}_{12}^{\top} + \tau\mathcal{I})$. The definition of $\mathcal{H}_{12}$ in (3.10) yields

$$\widetilde{\mathcal{H}}_{11} = \sigma\left(\mathcal{A}, \mathcal{B}, \mathcal{I}, \mathcal{Q}\right)^{\mathrm{T}} \overline{D}_p\left(\mathcal{A}, \mathcal{B}, \mathcal{I}, \mathcal{Q}\right) + \sigma\text{blkdiag}(\overline{D}_{\Pi_1}, \overline{D}_f, \overline{D}_{\Pi_2}, Q) \tag{3.15}$$

where $D_p = \sigma\hat{D}_p + \tilde{D}_p$, $\tilde{\mathcal{D}}_p = \hat{D}_p(\frac{1}{\sigma}\mathcal{I} - \frac{1}{\sigma}\hat{D}_p + \tau\mathcal{I})^{-1}\hat{D}_p$, $\overline{D}_{\Pi_1}, \overline{D}_f$ and $\overline{D}_\pi$ are defined analogously.

4 Proximal operator

According to (3.13), we need the explicit calculation process of $(\frac{1}{\sigma}(\mathcal{I} - D_p) + \tau_4\mathcal{I})^{-1}$ and. Furthermore, if $\boldsymbol{x}$ and $\mathcal{B}(\boldsymbol{x})$ are replaced by $\boldsymbol{x} - \boldsymbol{b}_1$ or $\mathcal{B}(\boldsymbol{x}) - \boldsymbol{b}_2$ respectively, the corresponding needs to be correct by a shift term. We demonstrate the computation details of some proximal operators in this section.

4.1 Handling `shift` term

For problems that have terms such as $p(\boldsymbol{x} - \boldsymbol{b}_1)$ or $f(\mathcal{B}(\boldsymbol{x}) - \boldsymbol{b}_2)$, the corresponding dual problem is

$$\begin{aligned} \min_{\boldsymbol{y},\boldsymbol{z},\boldsymbol{s},\boldsymbol{r},\boldsymbol{v}} \quad & \delta^*_{\mathcal{P}_2}(-\boldsymbol{p}) + f^*(-\boldsymbol{q}) - \langle \boldsymbol{b}_2, \boldsymbol{q}\rangle - \langle \boldsymbol{b}_1, \boldsymbol{p}\rangle + p^*(-\boldsymbol{s}) + \frac{1}{2}\langle \mathcal{Q}\boldsymbol{v}, \boldsymbol{v}\rangle + \delta^*_{\mathcal{P}_1}(-\boldsymbol{t}), \\ \text{s.t.} \quad & \mathcal{A}^*\boldsymbol{y} + \mathcal{B}^*\boldsymbol{z} + \boldsymbol{s} - \mathcal{Q}\boldsymbol{v} + \boldsymbol{r} = \boldsymbol{c},\ \boldsymbol{y} = \boldsymbol{p},\ \boldsymbol{z} = \boldsymbol{q},\ \boldsymbol{r} = \boldsymbol{t}. \end{aligned} \tag{4.1}$$

If f^* is differentiable, then the corresponding gradient is $-\nabla f^*(-\boldsymbol{q}) - \boldsymbol{b}_2$. If f is nonsmooth, it follows from the property of the proximal operator that $\boldsymbol{q} = \text{prox}_{f^*/\sigma}(\boldsymbol{x}_2/\sigma - \boldsymbol{z} - \boldsymbol{b}_2/\sigma)$. Consequently, we only need to replace the

$\text{prox}_{\sigma f}$ in (3.7) of $\text{prox}_{\sigma f} - \boldsymbol{b}_2$. For $p^*(-\boldsymbol{s})$, the corresponding $\text{prox}_{\sigma p}(\boldsymbol{x}_4 + \sigma(\mathcal{A}^*\boldsymbol{y} + \mathcal{B}^*\boldsymbol{z} - \boldsymbol{Q}\boldsymbol{v} + \boldsymbol{r} - \boldsymbol{c}))$ is replaced by $\text{prox}_{\sigma p}(\boldsymbol{x}_4 + \sigma(\mathcal{A}^*\boldsymbol{y} + \mathcal{B}^*\boldsymbol{z} - \boldsymbol{Q}\boldsymbol{v} + \boldsymbol{r} - \boldsymbol{c})) - \boldsymbol{b}_1$. Consequently, we do not need to introduce slack variables when adding `shift` term in $p(\boldsymbol{x})$ or $f(\mathcal{B}(\boldsymbol{x}))$.

4.2 ℓ_2 norm

For ℓ_2 norm, i.e. $p(\boldsymbol{x}) = \lambda\|\boldsymbol{x}\|_2$, where $\boldsymbol{x} \in \mathbb{R}^n$. The corresponding proximal operator is $\text{prox}_{\lambda\|\cdot\|_2} = \begin{cases} \boldsymbol{x} - \lambda\boldsymbol{x}/\|\boldsymbol{x}\|, & \text{if } \|\boldsymbol{x}\| > \lambda, \\ 0, & \text{otherwise.} \end{cases}$

According to the SMW formular: $(A - \boldsymbol{u}\boldsymbol{u}^{\mathrm{T}})^{-1} = A^{-1} + \frac{A^{-1}\boldsymbol{u}\boldsymbol{u}^{\mathrm{T}}A^{-1}}{1-\boldsymbol{u}^{\mathrm{T}}A^{-1}\boldsymbol{u}}$. Consequently, for $D \in \partial\text{prox}_{\lambda\|\cdot\|_2}\|\cdot\|_2$ we have

$$\begin{aligned}
&\left(\tau I + \frac{1}{\sigma}(I - D)\right)^{-1} = \left(\tau I + \lambda(I - \boldsymbol{x}\boldsymbol{x}^{\mathrm{T}}/\|\boldsymbol{x}\|^2)/\|\boldsymbol{x}\|\right)^{-1} \\
&= \left((\tau + \frac{\lambda}{\|\boldsymbol{x}\|})I - \frac{\lambda}{\|\boldsymbol{x}\|}\boldsymbol{x}\boldsymbol{x}^{\mathrm{T}}/\|\boldsymbol{x}\|^2\right)^{-1} \\
&= \frac{1}{(\tau + \frac{\lambda}{\|\boldsymbol{x}\|})}(I + \frac{\lambda}{\tau\|\boldsymbol{x}\|}\boldsymbol{x}^{\mathrm{T}}\boldsymbol{x}/\|\boldsymbol{x}\|^2).
\end{aligned}$$

4.3 Second order cone

Let $Q \subset \mathbb{R}^n$ denote the second-order cone (SOC), where a vector $\boldsymbol{x} \in Q$ is structured as $\boldsymbol{x} = [x_0; \bar{\boldsymbol{x}}]$ with $x_0 \in \mathbb{R}$ and $\bar{\boldsymbol{x}} \in \mathbb{R}^{n-1}$. The SOC is characterized by the condition $Q = \{\boldsymbol{x} \in \mathbb{R}^n : \|\bar{\boldsymbol{x}}\| \le x_0\}$. For $\boldsymbol{x} \in \text{int}(Q)$, the determinant is defined as $\det(\boldsymbol{x}) = x_0^2 - \|\bar{\boldsymbol{x}}\|^2$, and the inverse of $\boldsymbol{x}$, when $\det(\boldsymbol{x}) \neq 0$, is given by $\boldsymbol{x}^{-1} = \frac{1}{\det(\boldsymbol{x})}[x_0 \quad -\bar{\boldsymbol{x}}^{\top}]^{\top}$. For $p(\boldsymbol{x}) = \delta_Q(\boldsymbol{x})$, we utilize a barrier function $p(\boldsymbol{x}) = \mu g(\boldsymbol{x})$ to replace it, where $\mu > 0$. Define the logarithmic barrier function for any $x \in K$ by $g : \mathbb{R}^n \mapsto \mathbb{R}$ with

$$g(x) = \begin{cases} -\frac{1}{2}\log(\det(x)), & \text{if } \|\bar{x}\| < x_0, \\ +\infty, & \text{otherwise.} \end{cases}$$

We note that $\lim_{\mu\to 0}\mu g(x) = \delta_Q(x)$. For the smoothing function $\mu g(x)$, we have the following lemma.

Lemma 4.1. *(i) The proximal mapping of $\mu \cdot g(x)$ is given by $prox_{\mu g} : \mathbb{R}^n \mapsto int(K)$ with*

$$\text{prox}_{\mu g}(z) = \begin{bmatrix} \frac{1}{2}\left(z_0 + \sqrt{\frac{1}{2}(\|z\|^2 + 4\mu + \Delta)}\right) \\ \frac{z}{2}\left(1 + \frac{\sqrt{2}z_0}{\sqrt{\|z\|^2+4\mu+\Delta}}\right) \end{bmatrix}, \quad z \in \mathbb{R}^n$$

where $\Delta = \sqrt{\det(z)^2 + 8\mu\|z\|^2 + 16\mu^2}$. Furthermore, the inverse function of the proximal mapping is given by $prox_{\mu g}^{-1} : int(K) \to \mathbb{R}$ with

$$\text{prox}_{\mu g}^{-1}(x) = x - \mu x^{-1}, \quad x \in int(K).$$

(ii) The projection function is the limit of the proximal mapping as taking μ to 0. That is,

$$\lim_{\mu\to 0} prox_{\mu g}(z) = \Pi_K(z), \quad z \in \mathbb{R}^n.$$

(iii) For $z \in \mathbb{R}^n$, let $x = prox_{\mu g}(z)$. The inverse matrix of the derivative of the proximal mapping at the point z is given by

$$(\partial_z \text{prox}_{\mu g}(z))^{-1} = I - \mu \partial_x (x^{-1}),$$

where $\partial_x(x^{-1}) = \frac{1}{\det(x)} \begin{bmatrix} 1 & \\ & -I_{n-1} \end{bmatrix} - 2(x^{-1})(x^{-1})^{\top}$.

(iv) The derivative of the proximal mapping at the point z is given by

$$\begin{aligned} \partial_z \text{prox}_{\mu g}(z) &= \begin{bmatrix} \frac{\det(x)}{\det(x)-\mu} & \\ & \frac{\det(x)}{\det(x)+\mu} I_{n-1} \end{bmatrix} - \frac{2\mu \det(x) \begin{bmatrix} \frac{x_0}{\det(x)-\mu} \\ \frac{-x}{\det(x)+\mu} \end{bmatrix} \begin{bmatrix} \frac{x_0}{\det(x)+\mu} \\ \frac{-x}{\det(x)+\mu} \end{bmatrix}^{\top}}{\det(x) + 2\mu \left(\frac{x_0^2}{\det(x)-\mu} + \frac{\|\tilde{x}\|^2}{\det(x)+\mu} \right)} \\ &:= \Lambda + auu^{\top}. \end{aligned}$$

The corresponding linear system is $\sigma D + D\left(\frac{1}{\sigma}(I-D) + \tau_2 I\right)^{-1} D$, where $\Lambda = \begin{bmatrix} a_0 & \\ & a_1 I_{n-1} \end{bmatrix}$. According to the SMW formula

$$(\Lambda - uu^{\top})^{-1} = \Lambda^{-1} + \frac{\Lambda^{-1} uu^{\top} \Lambda^{-1}}{1 - u^{\top}\Lambda^{-1}u},$$

we have that $\frac{1}{\sigma}(I-D) + \tau_2 I = \frac{1}{\sigma}\left(\begin{bmatrix} b_0 & \\ & b_1 I \end{bmatrix} - auu^{\top}\right) := \frac{1}{\sigma}\left(\Lambda_1 - auu^{\top}\right), \Lambda_1 = (1+\sigma\tau_2)I - \Lambda$. Consequently, it follows that $(\frac{1}{\sigma}(I-D) + \tau_2 I)^{-1} = \sigma(\Lambda_1^{-1} + c\Lambda_1^{-1}uu^{\top}\Lambda_1^{-1})$, $c = \frac{a}{1-au^{\top}\Lambda_1^{-1}u}$ is a constant. Hence we have

$$\begin{aligned} &\sigma D + D(\frac{1}{\sigma}(I-D) + \tau_2 I)^{-1} D \\ =&\sigma\left(\Lambda + auu^{\top}\right) + \sigma(\Lambda + auu^{\top})(\Lambda_1^{-1} + c\Lambda_1^{-1}uu^{\top}\Lambda_1^{-1})(\Lambda + auu^{\top}) \\ =&\sigma\left(\Lambda + auu^{\top}\right) + \sigma(\Lambda\Lambda_1^{-1} + c\Lambda\Lambda_1^{-1}uu^{\top}\Lambda_1^{-1} + auu^{\top}\Lambda_1^{-1} + ac\gamma uu^{\top}\Lambda_1^{-1})(\Lambda + auu^{\top}), \\ =&\sigma\left(\Lambda\Lambda_1^{-1}\Lambda + \Lambda + c\Lambda\Lambda_1^{-1}uu^{\top}\Lambda_1^{-1}\Lambda + a(1+c\gamma)\Lambda\Lambda_1^{-1}uu^{\top} + a(1+c\gamma)uu^{\top}\Lambda_1^{-1}\Lambda + \left(a + a^2\gamma + a^2c\gamma^2\right)uu^{\top}\right) \\ =&\sigma\left(\tilde{\Lambda} + \begin{bmatrix} b_0u_0^2 & b_1u_0u_1^{\top} \\ b_1u_0u_1 & b_2u_1u_1^{\top} \end{bmatrix}\right). \end{aligned}$$

where $\tilde{\Lambda} = \Lambda\Lambda_1^{-1}\Lambda + \Lambda, \gamma = u^{\top}\Lambda_1^{-1}u$, and b_0, b_1, b_2 are constants. Denote $\Lambda_1^{-1}\Lambda = \begin{bmatrix} c_0 & \\ & c_1 I \end{bmatrix}$, we have $b_0 = cc_0^2 + 2a(1 + c\gamma)c_0 + a + a^2\gamma + a^2c\gamma^2, b_1 = cc_0c_1 + a(1+c\gamma)(c_0+c_1) + a + a^2\gamma + a^2c\gamma^2, b_2 = cc_1^2 + 2a(1+c\gamma)c_1 + a + a^2\gamma + a^2c\gamma^2$. Consequently, if we set $\tilde{u} = [b_1/b_2u_0; u_1]$, we have $b_2\tilde{u}\tilde{u}^{\top} = \begin{bmatrix} b_1^2/b_2u_0^2 & b_1u_0u_1^{\top} \\ b_1u_0u_1 & b_2u_1u_1^{\top} \end{bmatrix}$. It follows that

$$\sigma D + D\left(\frac{1}{\sigma}(I-D) + \tau_2 I\right)^{-1} D = \sigma\left(\tilde{\Lambda} + \begin{bmatrix} (b_0 - b_1^2/b_2)u_0^2 & 0 \\ 0 & 0 \end{bmatrix}\right) + \sigma b_2 \tilde{u}\tilde{u}^{\top}.$$

Consequently, can be represented as a diagonal matrix plus a rank 1 matrix, which is significant in constructing the Schur matrix when using .

4.4 Nuclear norm

For nuclear norm $\|\boldsymbol{X}\|_*$, let the signular value decomptation of $\boldsymbol{X}$ denoted by $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\mathrm{T}}$, where the its proximal operator can be presented by:

{prox:nuclear}
$$\mathrm{prox}_{\lambda\|\cdot\|_*}(\boldsymbol{X}) = U\,\mathrm{diag}\left(T_\alpha(\boldsymbol{\lambda}(\boldsymbol{X}))\right)V^T. \tag{4.2}$$

Denote

{genJacobian-nuclear}
$$\hat{D}_2(G) = U\left[\frac{\Omega^\mu_{\sigma,\sigma}+\Omega^\mu_{\sigma,-\sigma}}{2}\odot G_1 + \frac{\Omega^\mu_{\sigma,\sigma}-\Omega^\mu_{\sigma,-\sigma}}{2}\odot G_1^\top, (\Omega^\mu_{\sigma,0}\odot(G_2))\right]V^\top, \tag{4.3}$$

where $\sigma = [\sigma^{(1)},\cdots,\sigma^{(m)}]$ is the tensor singular value of $\boldsymbol{X}$, $G_1 = U^\top G V_1 \in \mathbb{R}^{n_1\times n_1}$, $G_2 = U^\top G V_2 \in \mathbb{R}^{n_1\times(n_2-n_1)}$ and $(\Omega^\mu_{\sigma,\sigma})$ is defined by:

$$(\Omega^\mu_{\sigma,\sigma})_{ij} := \begin{cases} \partial_B \mathrm{prox}_{\mu\|\cdot\|_1}(\sigma_i), & \text{if } \sigma_i = \sigma_j, \\ \left\{\frac{\mathrm{prox}_{\mu\|\cdot\|_1}(\sigma_i)-\mathrm{prox}_{\mu\|\cdot\|_1}(\sigma_j)}{\sigma_i-\sigma_j}\right\}, & \text{otherwise.} \end{cases} \tag{4.4}$$

Then for any $G\in\mathbb{R}^{m\times n}$, we have $\hat{D}[G] = D[G]$.

Lemma 4.2. *For operator $\mathcal{T}:\mathbb{R}^{n\times n}\to\mathbb{R}^{n\times n}: \mathcal{T}(\mathcal{G}) = \Omega_1\odot\mathcal{G} + \Omega_2\odot\mathcal{G}^{\mathrm{T}}$. The inverse of $\mathcal{T}$ is*

$$\mathcal{T}^{-1}(\mathcal{G}) = (\Omega_s+\Omega_a)\odot\mathcal{G} + (\Omega_s-\Omega_a)\odot\mathcal{G}^{\mathrm{T}},$$

where $\Omega_s = 1./[2(\Omega_1+\Omega_2)]$, $\Omega_a = 1./[2(\Omega_1-\Omega_2)]$ and $./$ means element wise division.

From the above lemma, $(D_2^{\tau_4})^{-1}$ can be represented as

$$\begin{aligned}(D_2^{\tau_4})^{-1}(\mathcal{G}) = \mathcal{U}\,&\left[(\Omega_s^{\tau_4}+\Omega_a^{\tau_4})\odot\mathcal{G}_1 + (\Omega_s^{\tau_4}-\Omega_a^{\tau_4})\odot\mathcal{G}_1^{\mathrm{T}}, (1./\Omega_3^{\tau_4}\odot\mathcal{G}_2)\right]\mathcal{V}^{\mathrm{T}} \\ &+ \sigma/(1+\sigma\tau_4)\mathcal{G}.\end{aligned} \tag{4.5}$$

where $\Omega_s^{\tau_4} = 1./[2(\Omega_1^{\tau_4}+\Omega_2^{\tau_4})] - \sigma/(1+\sigma\tau_4)E$, $\Omega_a^{\tau_4} = 1./[2(\Omega_1^{\tau_4}-\Omega_2^{\tau_4})]$, E is the matrix of ones with correct size. The detail of the compuation process is summarized in Algorithm 1. Then the low-rank structure can be fully exploited and the total computational cost for each inner iteration is reduced to $O(n_1 r^2)$

4.5 Fused reuglarizer

In this case, $p(x) = \lambda_1\|x\|_1 + \lambda_2\|Bx\|_1$, where $B(x) = [x_2-x_1,\cdots,x_n-x_{n-1}]$. According to lemma [5], the corresponding proximal operator is

{prox-fused}
$$\mathrm{prox}_p(v) = \mathrm{prox}_{\lambda_1\|\cdot\|_1}(x_{\lambda_2}(v)) = \mathrm{prox}_{\lambda_1\|\cdot\|_1}(v - B^{\mathrm{T}}z_{\lambda_2}(Bv)), \tag{4.6}$$

where $z_{\lambda_2}(u) := \mathrm{argmin}\left\{\frac{1}{2}\|B^Tz\|^2 - \langle z,u\rangle \;\middle|\; \|z\|_\infty\le\lambda_2\right\}, \forall u\in\mathbb{R}^{n-1}$. Define the multifunction $\mathcal{M}:\mathbb{R}^n\to\mathbb{R}^{n\times n}$ by:

{genJacobian:fused}
$$\mathcal{M}(v) := \{M\in\mathbb{R}^{n\times n} | M = \Theta P, \Theta\in\partial_B\mathrm{prox}_{\lambda_1\|\cdot\|_1}(x_{\lambda_2}), P\in\mathcal{P}_x(v)\}, \tag{4.7}$$

and $\mathcal{P}_x(v) := \{\hat{P}\in\mathbb{R}^{n-1\times n-1} | \hat{P} = I - B^{\mathrm{T}}(\Sigma_K BB^{\mathrm{T}}\Sigma_K)^\dagger B, K\in\mathcal{K}_z(v)\}$, where $\Sigma_K = \mathrm{Diag}(\sigma_K)\in\mathbb{R}^{n-1\times n-1}$ with

$$(\sigma_K)_i = \begin{cases} 0, & \text{if } i\in K \\ 1, & \text{otherwise, } i = 1,\cdots,n-1. \end{cases}$$

Algorithm 1 The procedure of computing $(D_2^{\tau_4})^{-1}(\mathcal{G})$. {alg:computeD2}

Require: $\mathcal{G}, (\Omega_1^{\tau_4})_{\alpha\alpha}, (\Omega_1^{\tau_4})_{\alpha\bar{\alpha}}, (\Omega_2^{\tau_4})_{\alpha\alpha}, (\Omega_2^{\tau_4})_{\alpha\bar{\alpha}}, U = [U_\alpha, U_{\bar{\alpha}}], V = [V_\alpha, V_{\bar{\alpha}}, V_\beta], (\Omega_1^{\tau_4})_{\alpha\beta}$,penalty parameter σ and τ_4.
Ensure: $(D_2^{\tau_4})^{-1}(\mathcal{G})$

1: Compute $\overline{\mathcal{G}} = \text{fft}(\mathcal{G}, [\,], 3)$,
2: Compute $(\mathcal{G}_1)_{\alpha\alpha}, (\mathcal{G}_1)_{\alpha\bar{\alpha}}, (\mathcal{G}_1)_{\bar{\alpha}\alpha}, (\mathcal{G}_1)_{\alpha\beta}$ where

$$(\mathcal{G}_1)_{\alpha\alpha} = U_\alpha^{\mathrm{T}}\ \overline{\mathcal{G}}\ (V_\alpha), \qquad (\mathcal{G}_1)_{\alpha\bar{\alpha}} = U_\alpha^{\mathrm{T}}\ \overline{\mathcal{G}}\ (V_{\bar{\alpha}}),$$
$$(\mathcal{G}_1)_{\bar{\alpha}\alpha} = U_{\bar{\alpha}}^{\mathrm{T}}\ \overline{\mathcal{G}}\ (V_{\bar{\alpha}}), \qquad (\mathcal{G}_1)_{\alpha\beta} = U_\alpha^{\mathrm{T}}\ \overline{\mathcal{G}}\ (V_\beta).$$

3: Compute

$$\mathcal{G}_2 = \begin{bmatrix} (\mathcal{G}_2)_{\alpha\alpha} & (\mathcal{G}_2)_{\alpha\bar{\alpha}} & (\mathcal{G}_2)_{\alpha\beta} \\ (\mathcal{G}_2)_{\bar{\alpha}\alpha} & 0 & 0 \end{bmatrix},$$

where

$$(\mathcal{G}_2)_{\alpha\alpha} = (\Omega_1^{\tau_4})_{\alpha\alpha} \odot (\mathcal{G}_1)_{\alpha\alpha} + (\Omega_2^{\tau_4})_{\alpha\alpha} \odot ((\mathcal{G}_1)_{\alpha\alpha})^{\mathrm{T}},$$
$$(\mathcal{G}_2)_{\alpha\bar{\alpha}} = (\Omega_1^{\tau_4})_{\alpha\bar{\alpha}} \odot (\mathcal{G}_1)_{\alpha\bar{\alpha}} + (\Omega_2^{\tau_4})_{\alpha\bar{\alpha}} \odot ((\mathcal{G}_1)_{\bar{\alpha}\alpha})^{\mathrm{T}},$$
$$(\mathcal{G}_2)_{\bar{\alpha}\alpha} = ((\Omega_1^{\tau_4})_{\alpha\bar{\alpha}})^{\mathrm{T}} \odot (\mathcal{G}_1)_{\bar{\alpha}\alpha} + ((\Omega_2^{\tau_4})_{\alpha\bar{\alpha}})^{\mathrm{T}} \odot ((\mathcal{G}_1)_{\alpha\bar{\alpha}})^{\mathrm{T}},$$
$$(\mathcal{G}_2)_{\alpha\beta} = (\Omega_1^{\tau_4})_{\alpha\beta} \odot (\mathcal{G}_1)_{\alpha\beta}.$$

4: Compute $\mathcal{G}_3 = \mathcal{G}_{12} + \mathcal{G}_{11} + \mathcal{G}_{21} + \mathcal{G}_{13}$ where

$$\mathcal{G}_{11} = U_\alpha\ (\mathcal{G}_2)_{\alpha\alpha}\ V_\alpha^{\mathrm{T}}, \qquad \mathcal{G}_{12} = U_\alpha\ (\mathcal{G}_2)_{\alpha\bar{\alpha}}\ V_{\bar{\alpha}}^{\mathrm{T}},$$
$$\mathcal{G}_{21} = U_{\bar{\alpha}}\ (\mathcal{G}_2)_{\bar{\alpha}\alpha}\ V_\alpha^{\mathrm{T}}, \qquad \mathcal{G}_{13} = U_\alpha\ (\mathcal{G}_2)_{\alpha\alpha}\ V_\beta^{\mathrm{T}}.$$

5: Compute $(D_2^{\tau_4})^{-1}(\mathcal{G}) = \text{ifft}(\mathcal{G}, [\,], 3) + \sigma/(1+\sigma\tau_4)\mathcal{G}$.

In fact, Define $\Gamma := I_n - B^{\mathrm{T}}(\Sigma BB^{\mathrm{T}}\Sigma)^{\dagger}B = \text{Diag}(\Gamma_1, \cdots, \Gamma_N)$, where for $i = 1, \cdots, N$,

$$\Gamma_i = \begin{cases} \frac{1}{n_i+1}\mathbf{E}_{n_i+1}, & \text{if } i \in J \\ I_{n_i}, & \text{if } i \in \{1, N\} \\ I_{n_i-1}, & \text{otherwise.} \end{cases}$$

Moreover, $\Gamma = H + UU^{\mathrm{T}} = H + U_J U_J^{\mathrm{T}}$, where $H \in \mathbb{R}^{n\times n}$ is a N-block diagonal matrix given by $H =$. Then $D = \Theta P$, with $P = I - B^{\mathrm{T}}(\Sigma BB^{\mathrm{T}}\Sigma)^{\dagger}B$, where

$$\theta_i = \begin{cases} 0, & \text{if } |(x_{\lambda_2}(v))_i| \le \lambda_1 \\ 1, & \text{otherwise,} \quad i = 1, \cdots, n, \end{cases}$$

and let $I_z(v) := \{i | |(z_{\lambda_2}(Bv))_i| = \lambda_2, i = 1, \cdots, n-1\}$. Then $\sigma = \text{Diag}(\sigma) \in \mathbb{R}^{(n-1)\times(n-1)}$ with

$$\sigma_i = \begin{cases} 0, & \text{if } i \in I_z(v) \\ 1, & \text{otherwise, } i = 1, \cdots, n-1. \end{cases}$$

It follows that $\Theta \in \partial_B \text{prox}_{\lambda_1\|\cdot\|_1}(x_{\lambda_2}(v))$ and $P \in \mathcal{P}_x(v)$. We first take a Clarke's generalized Jacobian: $J^k \in \hat{\partial}F(\boldsymbol{w}^k)$. The problem is transformed into solving the following problem by Gaussian elimination:

$$((\tau_1+1)I + \mathcal{A}\tilde{D}\mathcal{A}^{\mathrm{T}})\boldsymbol{d}_{\boldsymbol{y}}^k = \mathcal{A}D\hat{D}^{-1}F_{\boldsymbol{x}} - F_{\boldsymbol{y}},$$

where $\tilde{D} = D\hat{D}^{-1}D + \sigma D$, $\hat{D} = \frac{1}{\sigma}(I - D) + \tau_2 I$ and $D \in \hat{\partial}\text{prox}_{\sigma p}(\boldsymbol{x} - \sigma\mathcal{A}^*\boldsymbol{y})$ is an element of the generalized Jacobian of $\text{prox}_{\sigma p}$. Then

$$d_{\boldsymbol{x}}^k = (\hat{D})^{-1}(-F_{\boldsymbol{x}} + D\mathcal{A}^{\mathrm{T}} d_{\boldsymbol{y}}^k).$$

Therefore, $M = \Theta(H + U_J U_J^{\mathrm{T}}) = \Theta(H + U_J U_J^{\mathrm{T}})\Theta$, $\Theta^2 = \Theta$, $H^2 = H$, $\Theta H = \Theta H \Theta$. Define the following index sets:

$$\alpha := \{i | \theta_i = 1, i \in \{1, \cdots, n\}\}, \quad \beta := \{i \,| h_i = 1, i \in \alpha\},$$

where θ_i and h_i are the i-th diagonal entries of matrices Θ and H respectively. Then we have

$$A\Theta H A^{\mathrm{T}} = A\Theta H \Theta A^{\mathrm{T}} = A_\alpha H A_\alpha^{\mathrm{T}} = A_\beta A_\beta^{\mathrm{T}},$$

where $A_\alpha \in \mathbb{R}^{m\times|\alpha|}$ and $A_\beta \in \mathbb{R}^{m\times|\beta|}$ are two submatrix obtained from A by extracting those columns with indices in α and β. Meanwhile, we have

$$A\Theta(U_j U_j^{\mathrm{T}})A^{\mathrm{T}} = A\Theta(U_j U_j^{\mathrm{T}})\Theta A^{\mathrm{T}} = A_\alpha \tilde{U}\tilde{U}^{\mathrm{T}} A_\alpha^{\mathrm{T}},$$

where $\tilde{U} \in \mathbb{R}^{|\alpha|\times r}$ is a submatrix obtained from ΘU_J by extracting those rows with indices in α and the zero columns in ΘU_J are removed. Therefore, by exploiting the structure in D, we can express ADA^{T} in the following form:

$$ADA^{\mathrm{T}} = A_\beta A_\beta^{\mathrm{T}} + A_\alpha \tilde{U}\tilde{U}^{\mathrm{T}} A_\alpha^{\mathrm{T}}.$$

For $D\hat{D}^{-1}D$ case, where $\hat{D} = \frac{1}{\sigma}(I - D) + \tau_2 I$ $D = \Theta H \Theta$, we should note that $D = \Theta H \Theta = \Theta H$ holds since $\Theta = \text{Diag}(\Theta_1, \cdots, \Theta_N)$. Thus $M = \text{Diag}(\Theta_1\gamma_1, \cdots, \Theta_N\gamma_N)$. Define $J := \{j | \, \Gamma_j \text{ is not an identity matrix}, 1 \le j \le N\}$. Since $\text{supp}(Bx_{\lambda_2}(v)) \subset K$, we have that

$$\Theta_j = \mathbf{O}_{n_j+1} \text{ or } I_{n_j+1} \forall j \in J,$$

which implies $\Theta_j\Gamma_j \in \mathbb{S}_+^{n_j+1} \forall j \in J$. Therefore, $D \in \mathbb{S}_+^n$.As a consequence,

$$D = \text{Diag}(D_1, \cdots, D_n),$$

where

$$D_i = \begin{cases} \frac{1}{n_i+1}\mathbf{E}_{n_i+1}, & \text{if } i \in J \text{ and } \Theta_i = I_{n_i} \\ I_{n_i}, & \text{if } i \notin J \text{ and } i \in \{i, N\}, \\ 0, & \text{if } \Theta_i = 0, \\ I_{n_i-1}, & \text{otherwise.} \end{cases}$$

According to the SMW formular: $(A - vv^{\mathrm{T}})^{-1} = A^{-1} + \frac{A^{-1}vv^{\mathrm{T}}A^{-1}}{1-v^{\mathrm{T}}A^{-1}v}$, the inverse of $\hat{D}^{-1}$ has explicit solution:

$$\begin{aligned}((\frac{1}{\sigma} + \tau_2)I - \frac{1}{\sigma}D)^{-1} &= ((\frac{1}{\sigma} + \tau_2)I - \frac{1}{\sigma}\Theta(H + U_J U_J^{\mathrm{T}})\Theta)^{-1} \\ &= \begin{cases} \frac{\sigma}{1+\tau_2\sigma} I_{n_i+1} + \frac{1}{(1+\tau_2\sigma)(\tau_2)(n_i+1)}\mathbf{E}_{n_1+1}, & \text{if } 1 \\ \frac{1}{\tau_2} I_{n_i}, & \text{if if } i \notin J \text{ and } i \in \{i, N\}, \\ \frac{\sigma}{1+\sigma\tau_2} I_{n_i}, & \text{if } \Theta_i = 0, \\ \frac{1}{\tau_2} I_{n_i-1}, & \text{otherwise.} \end{cases}\end{aligned}$$

As a consequence, $\tilde{D}$ has the followimg formula:

$$\tilde{D} = \begin{cases} (\frac{1}{\tau_2} + \sigma)\frac{1}{n_i+1}\mathbf{E}_{n_1+1}, & \text{if } i \in J \text{ and } \Theta_i = I_{n_i} \\ \frac{1}{\tau_2} + \sigma I_{n_i}, & \text{if if } i \notin J \text{ and } i \in \{i, N\}, \text{and } \Theta_i = I_{n_i} \\ 0, & \text{if } \Theta_i = 0, \\ \frac{1}{\tau_2} + \sigma I_{n_i-1}, & \text{otherwise.} \end{cases}$$

and $D\hat{D}^{-1}$ has the following formula:

$$D\hat{D}^{-1} = \begin{cases} (\frac{1}{(\tau_2)(n_i+1)})\mathbf{E}_{n_1+1}, & \text{if } i \in J \text{ and } \Theta_i = I_{n_i} \\ \frac{1}{\tau_2} I_{n_i}, & \text{if if } i \notin J, i \in \{i, N\}, \text{and } \Theta_i = I_{n_i} \\ 0, & \text{if } \Theta_i = 0, \\ \frac{1}{\tau_2} I_{n_i-1}, & \text{otherwise.} \end{cases}$$

Define $\tilde{D} = \Theta\tilde{H}$ and the following index sets:

$$\alpha := \{i | \theta_i = 1, i \in \{1, \cdots, n\}\}, \beta := \{i | h_i = 1, i \in \alpha\},$$

Then we have $A\Theta\tilde{H}A^{\mathrm{T}} = A\Theta\tilde{H}\Theta A^{\mathrm{T}} = A_\alpha\tilde{U}\tilde{U}A_\alpha^{\mathrm{T}} + \tilde{A}_\beta\tilde{A}_\beta^{\mathrm{T}}$, where $\tilde{U}$ and $\tilde{A}$ are the scaling matrix of U and A. Then we have the following decomposition:

$$A\tilde{D}A^{\mathrm{T}} = W_1 W_2^{\mathrm{T}},$$

where $W_1 := [\tilde{A}_\beta, A_\alpha\tilde{U}\tilde{U}^{\mathrm{T}}] \in \mathbb{R}^{m\times(|\alpha|+|\beta|)}, W_2 = [\tilde{A}_\beta, A_\alpha]$. Using the above decomposition and we obtain

$$((\tau_1+1)I + A\tilde{D}A^{\mathrm{T}})^{-1} = \frac{1}{\tau_1+1}I_m - \frac{1}{\tau_1+1}\tilde{W}_1((\tau_1+1)I_{|\alpha|+|\beta|} + \tilde{W}_2^{\mathrm{T}}\tilde{W}_1)^{-1}\tilde{W}_2^{\mathrm{T}}.$$

Thus, we only need to factorize an $(|\alpha| + |\beta|) \times (|\alpha| + |\beta|)$ matrix and the total computational cost is merely $O(|\alpha| + |\beta|)^3 + O(m(|\alpha| + |\beta|)^2)$.

5 Numerical experiments

The criteria to mearsue the accuracy of $(\boldsymbol{y}, \boldsymbol{z}, \boldsymbol{v}, \boldsymbol{r}, \boldsymbol{x})$ is based on KKT optimality: conditions

$$\eta = \max\{\eta_P, \eta_D, \eta_K, \eta_{\mathcal{P}}\},$$

where

$$\eta_P := \frac{\|\mathcal{A}\boldsymbol{x} - \Pi_{\mathcal{P}_2}(\mathcal{A}\boldsymbol{x} - \boldsymbol{y})\|}{1 + \|\boldsymbol{x}\|}, \eta_D := \frac{\|\mathcal{A}^*(\boldsymbol{y}) + \mathcal{B}^*(\boldsymbol{z}) + \boldsymbol{s} - \mathcal{Q}(\boldsymbol{v}) - \boldsymbol{c}\|}{1 + \|\boldsymbol{c}\|},$$

$$\eta_K := \min\left\{\frac{\|\boldsymbol{x} - \mathrm{prox}_p(\boldsymbol{x} - \boldsymbol{s})\|}{1 + \|\boldsymbol{s}\| + \|\boldsymbol{x}\|}, \frac{\|\mathcal{Q}(\boldsymbol{v}) - \mathcal{Q}(\boldsymbol{x})\|_{\mathrm{F}}}{1 + \|\mathcal{Q}(\boldsymbol{v})\| + \|\mathcal{Q}(\boldsymbol{x})\|}\right\},$$

$$\eta_{\mathcal{P}} = \min\left\{\frac{\|\mathrm{prox}_f(\mathcal{B}\boldsymbol{x} - \boldsymbol{z}) - \mathcal{B}(\boldsymbol{x})\|}{1 + \|\mathcal{B}(\boldsymbol{x})\| + \|\boldsymbol{z}\|} \text{ or } \frac{\|-\nabla f^*(-\boldsymbol{z}) - \mathcal{B}(\boldsymbol{x})\|}{1 + \|\mathcal{B}(\boldsymbol{x})\| + \|\boldsymbol{z}\|}, \frac{\|\Pi_{\mathcal{P}_1}(\boldsymbol{x} - \boldsymbol{r}) - \boldsymbol{x}\|}{1 + \|\boldsymbol{x}\| + \|\boldsymbol{r}\|}\right\}.$$

We also compute the relative gap by

$$\eta_g = \frac{\texttt{|pobj - dobj|}}{1 + \texttt{|pobj|} + \texttt{|dobj|}}.$$

For given accuracy η, we terminate SSNCVX when $\eta <$ `opts.tol`.

5.1 Dataset Source

We are going to test the problem in data from Random Gaussian ,UCI data [1] and LIBSVM dataset [2] These data sets are collected from 10-K Corpus [4] and the UCI data repository [7]. As suggested in [3], for the data sets **pyrim**, **triazines**, **abalone**, **bodyfat**, **housing**, **mpg**, and **space_ga**, we expand their original features by using polynomial basis functions over those features. For example, the last digit in **pyrim5** indicates that an order 5 polynomial is used to generate the basis functions. This naming convention is also used in the rest of the expanded data sets. These test instances, shown in Table 5, can be quite difficult in terms of the problem dimensions and the largest eigenvalue of $\mathcal{A}\mathcal{A}^*$, which is denoted as $\lambda_{\max}(\mathcal{A}\mathcal{A}^*)$.

5.2 Lasso case

In the following table , m denotes the number of number of samples, n denotes the number of features, and "nnz" denotes the number of nonzeros in the solution x obtained by the optimal solution using the following estimation:

$$\text{nnz} := \min\{k | \sum_{i=1}^{k} |\hat{x}_i| \geq 0.999\|x\|_1\},$$

where $\hat{x}$ is obtained by sorting x such that $|\hat{x}_1| \geq |\hat{x}_2| \geq \cdots \geq |\hat{x}_n|$. The stop criterion is set by the following relative KKT residual:

$$\eta = \frac{\tilde{x} - \text{prox}_{\lambda\|\cdot\|_1}(\tilde{x} - \mathcal{A}^*(\mathcal{A}\tilde{x} - b))}{1 + \|\tilde{x}\| + \|\mathcal{A}\tilde{x} - b\|}.$$

We stop the tested algorithms when $\eta < 1e-6$. The algorithms we want to compare is SSNAL: SSN based on ALM (SOTA, Best Paper Prize for Young Researchers in Continuous Optimization 2019). [3] FPC_AS. [4] A accelerated proximal gradient based method, ADMM algorithm, APG algorithm. The test results for different lambda choice , i.e. $\lambda = 10^{-3}\|\mathcal{A}^{\mathrm{T}}\boldsymbol{b}\|_\infty$ and $\lambda = 10^{-4}\|\mathcal{A}^{\mathrm{T}}\boldsymbol{b}\|_\infty$ and different algorithms are given in Table 3 and 4. We can see that, both SSNCP and SSNAL has successfully solved all problems while other first-order methods can not. Furthermore,

id	nnz	SSNCP		SSNAL		FPC_AS		ADMM	
		it	time	it	time	η	time	η	time
uci_CT	13	25	**0.57**	31	0.64	1.4-4	58:25	9.1-7	10:31
log1p.E2006.train	5	35	**11.57**	48	26.62	2.6-1	7:00:01	9.9-7	33:47
E2006.test	1	17	**0.19**	18	0.28	3.7-4	19:25	6.7-7	18
log1p.E2006.test	8	33	**3.27**	45	5.12	6.9-1	4:56:49	9.9-7	3:10
pyrim5	72	57	**1.37**	97	2.16	8.8-1	1:12:55	3.2-5	21:14
triazines4	519	64	**9.91**	92	11.23	9.6-1	7:00:05	3.2-4	2:13:00
abalone7	24	28	**0.72**	57	1.06	**Error**	**Error**	9.9-7	9:52
bodyfat7	2	29	**0.79**	40	1.08	3.6-1	1:12:02	9.9-7	1:49
housing7	158	57	**1.65**	57	1.74	8.4-1	1:41:01	9.9-7	22:12
mpg7	47	40	**0.07**	50	0.11	**Error**	**Error**	9.9-7	07
spacega9	14	26	**0.26**	38	1.01	1.5-2	30:55	9.9-7	37
E2006.train	1	24	**0.82**	18	0.87	1.4-3	1:24:18	9.1-7	10:41

Table 3. The results of our algorithm on Lasso problem with real data($\lambda = 10^{-3}\|\mathcal{A}^{\mathrm{T}}\boldsymbol{b}\|_\infty$), the

{tab:1e3}

[1] https://archive.ics.uci.edu/
[2] https://www.csie.ntu.edu.tw/ cjlin/libsvmtools/datasets/
[3] https://github.com/MatOpt/SuiteLasso
[4] http://www.caam.rice.edu/optimization/L1/FPC_AS/

id	nnz	SSNCP		SSNAL		FPC_AS		ADMM	
		it	time	it	time	η	time	η	time
uci_CT	44	44	**1.26**	53	1.75	2.6-1	0.31	9.9-7	0.31
log1p.E2006.train	599	55	**48.92**	123	78.49	1.3-1	7:00:01	1.7-1	35:21
E2006.test	1	18	**0.20**	18	0.26	2.7-4	20:21	6.3-7	21
log1p.E2006.test	1081	114	**25.24**	176	34.33	8.7-1	4:48:29	9.9-7	2:28
pyrim5	78	77	**2.01**	113	2.53	9.8-1	54:20	1.6-3	21:34
triazines4	260	105	**18.48**	147	28.06	9.8-1	7:47:03	1.1-3	1:55:16
abalone7	59	52	**1.63**	63	2.00	**Error**	**Error**	9.9-7	9:36
bodyfat7	3	41	**1.14**	56	1.51	1.9-1	1:13:08	9.8-7	4:05
housing7	281	70	**2.51**	68	2.52	4.2-1	1:39:36	6.6-6	42:36
mpg7	128	49	**0.11**	56	0.18	**Error**	**Error**	9.9-7	11
spacega9	38	48	**0.53**	54	0.72	4.0-2	30:26	9.7-7	0.31
E2006.train	1	21	**0.75**	49	0.88	9.9-4	1:27:29	9.9-7	10:53

Table 4. The results of our algorithm on Lasso problem with real data($\lambda = 10^{-4}\|\mathcal{A}^{\mathrm{T}}\boldsymbol{b}\|_\infty$), the

{tab:1e4}

Table 5. Statistics of the UCI test instances .

{tab:tensor-image

Probname	(m, n)	$\lambda_{\max}(\mathcal{A}\mathcal{A}^*)$
E2006.train	(3308, 72812)	1.912+05
log1p.E2006.train	(16087,4265669)	5.86e+07
E2006.test	(3308,72812)	4.79e+04
log1p.E2006.test	(3308,1771946)	1.46e+07
pyrim5	(74,169911)	1.22e+06
triazines4	(186,557845)	2.07e+07
abalone7	(4177,6435)	5.21e+05
bodyfat7	(252,116280)	5.29e+04
housing7	(506,77520)	3.28e+05
mpg7	(392,3432)	1.28e+04
spacega9	(3107,5005)	4.01e+03

5.3 Fused Lasso case

For fused lasso case, part of the numerical results are listed in table 6.

id	nnz(x)	nnz(Bx)	SSN		SSNAL		iADMM		ADMM	
			it	time	it	time	η	time	η	time
uci_CT	8	1	44	**0.25**	53	0.42	7.6-7	3:29	2.5-7	2:55
log1p.E2006.train	32	5	25	**12.6**	11	19.6	7.6-7	3:29	2.5-7	2:55
E2006.test	1	1	9	**0.17**	5	0.33	5.3-7	2:21	6.3-7	4:56
log1p.E2006.test	32	5	25	**5.75**	13	6.37	5.3-7	2:21	6.3-7	4:56
pyrim5	1135	74	62	**2.34**	22	3.06	6.0-7	7:59	9.8-7	2:22
triazines4	2670	207	41	**7.2**	56	9.45	6.0-7	7:59	9.8-7	2:22
bodyfat7	63	8	44	**0.81**	14	0.89	6.0-7	7:59	9.8-7	2:22
abalone7	1	1	33	**1.02**	14	1.17	6.0-7	7:59	9.8-7	2:22
housing7	213	25	70	**1.8**	68	2.4	9.2-7	17:24	9.9-7	4:20
mpg7	42	11	48	**0.11**	54	0.15	7.6-7	30:26	9.7-7	7:33
spacega9	24	11	48	**0.37**	54	0.44	7.6-7	30:26	9.7-7	7:33
E2006.train	1	1	9	**0.58**	11	1.12	5.3-7	36:42	9.9-7	10:53

Table 6. The results of our algorithm on Lasso problem with real data($\lambda_1 = 10^{-3}\|A^*b\|_\infty$ and $\lambda_2 = 5\lambda_1$.)

{tab:fuse3}

5.4 QP case

5.5 SDP

5.6 LP case

5.7 SOCP case

References

[1] MOSEK ApS. 2019. *The MOSEK optimization toolbox for MATLAB manual. Version 10.1.0.* http://docs.mosek.com/10.1/toolbox/index.html

[2] Stephen R Becker, Emmanuel J Candès, and Michael C Grant. 2011. Templates for convex cone problems with applications to sparse signal recovery. *Mathematical programming computation* 3 (2011), 165–218.

[3] Ling Huang, Jinzhu Jia, Bin Yu, Byung-Gon Chun, Petros Maniatis, and Mayur Naik. 2010. Predicting execution time of computer programs using sparse polynomial regression. *Advances in neural information processing systems* 23 (2010).

[4] Shimon Kogan, Dimitry Levin, Bryan R Routledge, Jacob S Sagi, and Noah A Smith. 2009. Predicting risk from financial reports with regression. In *Proceedings of human language technologies: the 2009 annual conference of the North American Chapter of the Association for Computational Linguistics*. 272–280.

[5] Xudong Li, Defeng Sun, and Kim-Chuan Toh. 2018. A highly efficient semismooth Newton augmented Lagrangian method for solving Lasso problems. *SIAM Journal on Optimization* 28, 1 (2018), 433–458.

[6] Xudong Li, Defeng Sun, and Kim-Chuan Toh. 2018. On efficiently solving the subproblems of a level-set method for fused Lasso problems. *SIAM Journal on Optimization* 28, 2 (2018), 1842–1866.

[7] Moshe Lichman et al. 2013. UCI machine learning repository.

[8] Jun Liu, Shuiwang Ji, Jieping Ye, et al. 2009. SLEP: Sparse learning with efficient projections. *Arizona State University* 6, 491 (2009), 7.

[9] Robert Mifflin. 1977. Semismooth and semiconvex functions in constrained optimization. *SIAM Journal on Control and Optimization* 15, 6 (1977), 959–972.