Semismooth Newton Algorithm for Efficient Projections onto $l_{1,\infty}$-norm Ball

Dejun Chu (储德军)[1], Changshui Zhang[2], Shiliang Sun[3], Qing Tao[4]

[1] 合肥工业大学软件学院, cdj@hfut.edu.cn

[2] 清华大学自动化系, zcs@mail.tsinghua.edu.cn

[3] 华东师范大学计算机科学与技术学院, slsun@cs.ecnu.edu.cn

[4] 陆军炮兵防空兵学院信息工程系, taoqing@gmail.com

Introduction

- The structured sparsity-inducing $l_{1,\infty}$-norm plays an important role in jointly sparse models which select or remove simultaneously all the variables forming a group. However, its resulting problem is more difficult to solve than the conventional l_1-norm constrained problem.
- The main contribution is that we develop an efficient semismooth Newton method for computing the $l_{1,\infty}$-norm ball projection at most $O(dm)$ iterations.

Problem Statement

- Define the $l_{1,\infty}$-norm: $\|\boldsymbol{W}\|_{1,\infty} = \sum_{i=1}^{d} \max_j |W_{i,j}|$
- The Euclidean projection with respect to $\boldsymbol{A}$ onto the $l_{1,\infty}$-norm ball can be cast as

$$\min_{\boldsymbol{W}} \frac{1}{2}\|\boldsymbol{W}-\boldsymbol{A}\|_F^2, \quad s.t. \sum_{i=1}^{d} \max_j |W_{i,j}| \le C$$

- Reformulate the projection problem as

$$\min_{\boldsymbol{W},\boldsymbol{\mu}} \frac{1}{2}\|\boldsymbol{W}-\boldsymbol{A}\|_F^2$$

$$s.t. \quad 0 \le W_{i,j} \le \mu_i, \ \forall i,j \quad \sum_{i=1}^{d} \mu_i = C.$$

- Obtain the primal optimal

$$W_{i,j} = \min\left(\max\left(A_{i,j},0\right),\mu_i\right) = \min\left(A_{i,j},\mu_i\right).$$

- Rewrite the problem as

$$\min_{\boldsymbol{\mu}} \frac{1}{2}\sum_{i=1}^{d}\sum_{j=1}^{m} \max\left(A_{i,j}-\mu_i,0\right)^2$$

$$s.t. \sum_{i=1}^{d}\mu_i = C, \quad \mu_i \ge 0, \ i=1,\dots,d.$$

- Derive the KKT conditions

$$\sum_{j=1}^{m} \max\left(A_{i,j}-\mu_i,0\right) = \theta, \ \mu_i > 0, \ \text{for all } i.$$

- Semismooth equations with $\boldsymbol{x} = (\mu_1, \mu_1, \cdots \mu_1, \theta)^{\mathrm{T}}$

$$F(\boldsymbol{x}) = \begin{pmatrix} \sum_j \max\{A_{1,j}-\mu_1,0\}-\theta \\ \vdots \\ \sum_j \max\{A_{d,j}-\mu_d,0\}-\theta \\ -\mu_1-\mu_2-\ldots-\mu_d+C \end{pmatrix} = \boldsymbol{0}$$

Our Semismooth Newton Method

- To find the Newton direction, we solve the equation

$$J(\boldsymbol{x}^{(t)})\boldsymbol{v} = -F(\boldsymbol{x}^{(t)}),$$

where $J(x^{(t)})$ is an arbitrary element of the generalized Jacobian matrix of $F(x^{(t)})$,

$$J(\boldsymbol{x}) = \begin{pmatrix} -|\mathcal{I}(\mu_1)| & 0 & \ldots & 0 & -1 \\ 0 & -|\mathcal{I}(\mu_2)| & \ddots & \vdots & -1 \\ \vdots & \ddots & \ddots & 0 & \vdots \\ 0 & \ldots & 0 & -|\mathcal{I}(\mu_d)| & -1 \\ -1 & -1 & \ldots & -1 & 0 \end{pmatrix}.$$

- The LU factorization $J=LU$, where

$$L = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 & 0 \\ \frac{1}{|\mathcal{I}(\mu_1)|} & \frac{1}{|\mathcal{I}(\mu_2)|} & \cdots & \frac{1}{|\mathcal{I}(\mu_d)|} & 1 \end{pmatrix}, \quad U = \begin{pmatrix} -|\mathcal{I}(\mu_1)| & \cdots & 0 & -1 \\ 0 & \ddots & 0 & \vdots \\ \vdots & \cdots & -|\mathcal{I}(\mu_d)| & -1 \\ 0 & \cdots & 0 & \sum_{i=1}^{d}\frac{1}{|\mathcal{I}(\mu_i)|} \end{pmatrix}$$

- We take the linear time complexity $O(dm)$ to compute the Newton step $\boldsymbol{v}$ via forward substitution and back substitution, i.e.,

$$L\boldsymbol{z} = -F, \quad U\boldsymbol{v} = \boldsymbol{z}.$$

Our Proposed Algorithm

Algorithm 1 Semismooth Newton Algorithm for $\ell_{1,\infty}$-norm Ball Projection

1: **Input:** $\boldsymbol{A} \in \mathbb{R}^{d\times m}$, $C>0, \epsilon>0, t=0$.
2: **if** $\|\boldsymbol{A}\|_{1,\infty} \le C$ **then**
3: $\quad \boldsymbol{W} = \boldsymbol{A}$ and return.
4: **end if**
5: Initialize $\theta^{(0)} \in [0, \max_i \sum_{j=1}^{m} A_{i,j}]$ and compute $\mu_i^{(0)}$ to satisfy the equation (5).
6: **repeat**
7: $\quad$ Compute $\mathcal{G} = \{i \mid \theta^{(t)} \le \sum_{j=1}^{m} A_{i,j}\}$.
8: $\quad$ Update $\mathcal{I}(\mu_i^{(t)}) = \left\{j \mid A_{i,j} \ge \mu_i^{(t)}, i \in \mathcal{G}\right\}$.
9: $\quad$ Compute $F(\boldsymbol{x}^{(t)})$ via (6).
10: $\quad$ Compute $\boldsymbol{v}^{(t)}$ via (12)-(15).
11: $\quad$ Update $\boldsymbol{x}^{(t+1)} = \boldsymbol{x}^{(t)} + \boldsymbol{v}^{(t)}$.
12: $\quad t = t+1$.
13: **until** stopping criterion is satisfied, i.e., $\|\boldsymbol{v}^{(t)}\| \le \epsilon$.
14: $\boldsymbol{\mu} = \max(\boldsymbol{\mu}, 0)$.
15: $W_{i,j} = \min(A_{i,j}, \mu_i)$ for all i,j.
16: **Output:** $\boldsymbol{W}$

Convergence Analysis

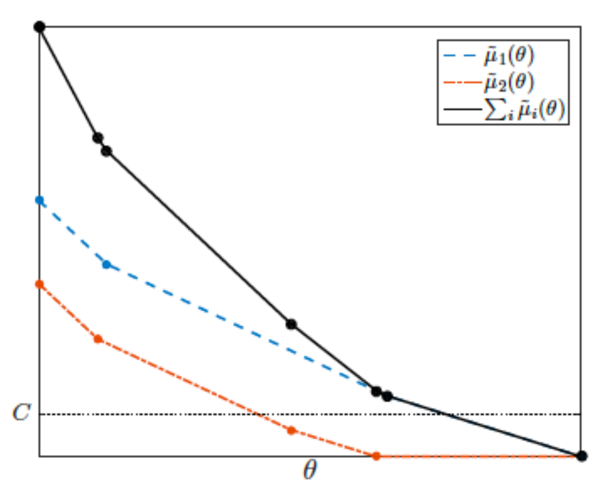
- Build a univariate equation from the equality constraint

$$s(\theta) = \sum_{i=1}^{d} \tilde{\mu}_i(\theta) - C = 0$$

where

$$\tilde{\mu}_i(\theta) = \begin{cases} \frac{\sum_{j\in\mathcal{I}(\mu_i)} A_{i,j}-\theta}{|\mathcal{I}(\mu_i)|}, & \text{if } 0 \le \theta < \sum_{j=1}^{m} A_{i,j} \\ 0, & \text{if } \theta \ge \sum_{j=1}^{m} A_{i,j} \end{cases}$$

$$\mathcal{I}(\mu_i) = \{j \mid A_{i,j} \ge \mu_i, j=1,\ldots,m\}$$



- Equivalent variant of Algorithm 1

$$\text{Compute } \tilde{\mu}_i(\theta^{(t)}) = \frac{\sum_{j\in\mathcal{I}(\mu_i^{(t)})} A_{i,j}-\theta^{(t)}}{|\mathcal{I}(\mu_i^{(t)})|}.$$

$$\text{Update } \theta^{(t+1)} = \theta^{(t)} - \frac{s(\theta^{(t)})}{s'(\theta^{(t)})} \text{ and } \mu_i^{(t+1)} = \frac{\sum_{j\in\mathcal{I}(\mu_i^{(t)})} A_{i,j}-\theta^{(t+1)}}{|\mathcal{I}(\mu_i^{(t)})|}$$

- Convergence theorem in finite steps

Proposition

Suppose $\theta^{(t)}$ lies between two breakpoints, i.e., $\theta^{(t)} \in (\Theta_{[j-1]}, \Theta_{[j]}]$. Assume $s(\Theta_{[j]}) > 0$. There holds

$$\theta^{(t)} \le \Theta_{[j]} < \theta^{(t+1)}.$$

Theorem

Algorithm 2 converges to the root of equation $s(\theta)=0$ in $O(dm)$ steps at most.

Experimental Results

- Projection onto the $l_{1,\infty}$-norm ball

Table: Running times (s) and accuracy on a $10,000 \times 10,000$ matrix $\boldsymbol{A}$.

$\frac{C}{\|\boldsymbol{A}\|_{1,\infty}}$	SBM Error	SBM Time	NRFM Error	NRFM Time	NRFM Speedup	Proposed SNA Error	Proposed SNA Time	Proposed SNA Speedup
0.01	1.05×10^{-5}	35.43	2.06×10^{-11}	10.33	3.42	1.47×10^{-12}	2.85	12.43
0.05	8.55×10^{-6}	35.06	2.06×10^{-11}	10.71	3.30	3.18×10^{-12}	3.53	9.91
0.1	6.40×10^{-6}	34.85	4.54×10^{-12}	10.99	3.16	1.18×10^{-11}	4.83	7.21
0.2	3.32×10^{-6}	33.87	1.27×10^{-11}	11.78	2.87	5.45×10^{-11}	5.48	6.17
0.3	1.54×10^{-6}	33.36	5.63×10^{-11}	12.31	2.71	3.82×10^{-11}	4.60	7.25
0.4	6.32×10^{-7}	32.95	1.81×10^{-11}	12.98	2.53	9.45×10^{-11}	3.75	8.78
0.5	2.26×10^{-7}	32.61	2.54×10^{-11}	13.70	2.38	6.54×10^{-11}	3.25	10.03

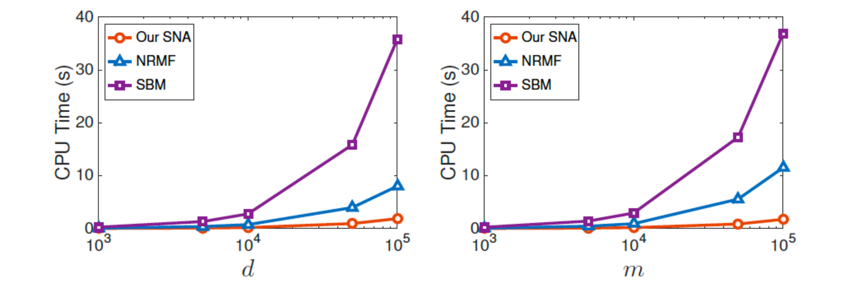


Figure: Time versus data dimensions and tasks. The left figure displays the result for varying dimensions and fixed tasks $m=1000$. The right figure displays the result for varying tasks and fixed dimensions $d=1000$.

- Application to multitask Lasso problem

$$\min_{\boldsymbol{W}} \mathcal{L}(\boldsymbol{W}) = \frac{1}{2}\sum_{i=1}^{n}\|\boldsymbol{y}_i - \boldsymbol{X}_i\tilde{\boldsymbol{w}}_i\|_2^2$$

$$s.t. \quad \|\boldsymbol{W}\|_{1,\infty} \le C$$

with projected gradient method

$$\boldsymbol{W}_{k+1} = \mathcal{P}_{\ell_{1,\infty}}\left(\boldsymbol{W}_k - \lambda_k \nabla\mathcal{L}(\boldsymbol{W}_k)\right)$$

Table: Projection times (s) on dataset School.

C/d	SBM Time	NRFM Time	NRFM Speedup	Our SNA Time	Our SNA Speedup
0.01	0.017	0.002	8.755	0.002	7.688
0.05	0.459	0.116	3.944	0.053	8.717
0.1	0.442	0.137	3.224	0.057	7.724
0.5	0.434	0.172	2.528	0.065	6.712
1.0	0.389	0.172	2.199	0.067	5.617

- **Source code is available at *https://github.com/djchu/projontol1inf***