!Optimizer Panda team

Ali Fathi - Mohammad Rashid - Shayan Ranjbarzadeh

27/8/2021

Table of Contents

Round 1

Ali Fathi - Mohammad Rashid - Shayan Ranjbarzadeh

$$
\begin{aligned}
\text{minimize } & 0^T v \\
\text{s.t.: } & Sv = 0 \\
& l \preceq v \preceq u
\end{aligned}
$$

Or simply use the code for round 2 :))

Round 2

Original problem

$$
\begin{aligned}
\text{minimize} \quad & \|v\|_0 \\
\text{s.t.:} \quad & Sv = 0 \\
& l \preceq v \preceq u
\end{aligned}
$$

Weighted Algorithm (Linear!)

$$
\begin{aligned}
\text{minimize } & \Sigma_{i=1}^{n} w_i |v_i| \\
\text{s.t.: } & Sv = 0 \\
& l \preceq v \preceq u
\end{aligned}
$$

If $w = \vec{1}_n$, the wighted problem would be norm 1 minimization. If $w_i \approx \infty$ for each $i \in I_z$ and 0 otherwise, it would find some sparse solution in which $v_{I_z} = 0$.

!Optimizer Panda team

Ali Fathi - Mohammad Rashid - Shayan Ranjbarzadeh

Basic updating:

$$\vec{w^{(0)}} = \vec{1}_n$$
$$w_i^{(t+1)} = \frac{1}{|v_i^{(t)}| + \epsilon}$$

(ϵ prevents division by 0)
Trying to push small elements of v to zero

Weighted Algorithm

Updating Weights

NW4 updating (used in contest):

$$\vec{w^{(0)}} = \vec{1}_n$$

$$w_i^{(t+1)} = \frac{1 + (|v_i^{(t)}| + \epsilon)^p}{(|v_i^{(t)}| + \epsilon)^{p+1}}$$

In which $0 < p < 1$ is some modifiable parameter (0.8 was used in contest).
Other updating methods could be found at [1].

Multiplying NW4 weight with some random number:

$$\vec{w^{(0)}} = \vec{1}_n$$

$$w_i^{(t+1)} = \frac{1 + (|v_i^{(t)}| + \epsilon)^p}{(|v_i^{(t)}| + \epsilon)^{p+1}} \times r_i^3$$

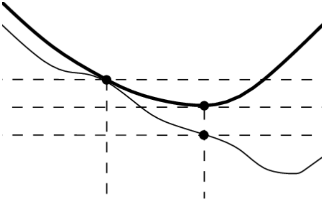
$$r_i \sim Unif[0, 1]$$

Distribution and power (3) are adjusted experimentally.

The Theory Behind Wighted Algorithm

merit function

- A convex approximation of l_0-norm: $\Phi_\epsilon(v)$ such that $\lim_{\epsilon\to 0}\Phi_\epsilon(v) = \|v\|_0$
  (ϵ does more than preventing division by 0!)



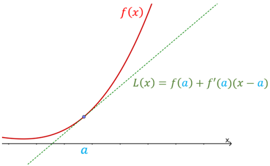
- e.g.:

$$\Phi_\epsilon(v) = \sum_{i=1}^{n} \log\left(|v_i| + \epsilon\right)$$

(the merit function for NW4 could be found at [1])

- Using linear approximation of $\Phi_\epsilon(v)$:

$$\Phi_\epsilon(v) \approx \Phi_\epsilon(v^{(t)}) + \nabla\Phi_\epsilon(v^{(t)})^T.(v - v^{(t)})$$

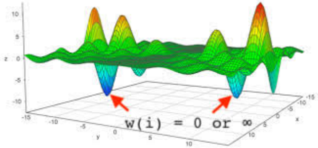


- for example for logarithmic Φ_ϵ we have:

$$w^{(t+1)} := \nabla\Phi_\epsilon(v^{(t)}) = \left(\frac{1}{|v_1^{(t)}| + \epsilon}, \cdots, \frac{1}{|v_n^{(t)}| + \epsilon}\right)^T$$

Advantages

- It is linear (LP) and fast
- Local optima are sparse



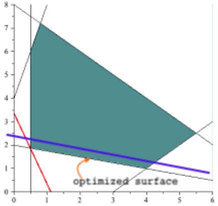
Other tested algorithms

- Dual Density method:

$$\begin{aligned} \text{maximize } & \alpha\Phi_\epsilon(s) + b^T y \\ \text{s.t.: } & S^T y - u + v = 0 \\ & s = w - u + v \\ & (s, u, v, w) \succeq 0 \\ & b^T y := \gamma = min\{\|Wv\|_1 : Sv = 0\} \end{aligned}$$

Other tested algorithms

- Greedy fixing minimum elements (with minimum absolute value) to zero.
- Fixing optimal norm 1 surface by adding the constraint $1^T|\vec{v}| \leq 1.1z_0$, in which z_0 is minimum norm 1 objective value, and then optimize with random weights.



- Greedy and random search among sparse neighbors
- Projecting candidate edges on $null(S)$ plate

Ideas to work better with data

precision modification

- Ignore unnecessary precision in S, L, U and V:

$$|x| < 2E - 5 \Rightarrow x \leftarrow 0$$

Ideas to work better with data

obliged non-zero elements

- Ignoring definitely-nonzero elements of V from objective function:

$$L \preceq V \preceq U$$

$$L_{i,j} > 0 \text{ or } U_{i,j} < 0 \Rightarrow V_{i,j} \neq 0$$
$$\forall i \; (\exists j \; L_{i,j} > 0 \text{ or } U_{i,j} < 0$$
$$\Rightarrow \text{ i'th row of } V \text{ could not be knocked out})$$

and w_i would be zero in these indices (i) and the solver wouldn't try to make those rows zero.

Ideas to work better with data

putting zero elements aside

- Ignoring definitely-zero elements of V from problem:

$$L \preceq V \preceq U$$

$$L_{i,j} = 0 = U_{i,j} \Rightarrow V_{i,j} = 0$$

and we totally exclude those elements, and consequently the problem size shrinks dramatically!

Lower bound analysis

power of linearity

- Most of non-zero elements of the best v are those which are obliged to be non-zero by l and u
- We knocked out (i.e. putting zero) the other non-zero elements and checked the feasibility (whether $Sv = 0$ and $l \leq v \leq u$ and $v_i = 0$ is feasible or not) and if it got infeasible, we count it in lower bound.
- Result: for our best answer, l_0-norm is highly near this lower bound.

Round 3

Original problem

$$
\begin{aligned}
\text{minimize} \quad & \|V\|_{2,0} \\
\text{s.t.:} \quad & SV = 0 \\
& L \preceq V \preceq U
\end{aligned}
$$

!Optimizer Panda team

Ali Fathi - Mohammad Rashid - Shayan Ranjbarzadeh

$$
\begin{aligned}
\text{minimize } & \|V\|_{1,1} = \Sigma_{j=1}^{c} \|v_j\|_1 \\
\text{s.t.: } & Sv_j = 0 \quad \forall j \\
& l_j \preceq v_j \preceq u_j \quad \forall j
\end{aligned}
$$

(c is the number of columns in V and v_j denotes j'th column of V)
In $\|V\|_{p,0}$, p-norm of rows beside norm 0, just distinguishes whether a row is all-zero or not. Both $p = 1$ and $p = 2$ fulfill this job (and even $\|V\|_{1,0} = \|V\|_{2,0}$!). Afterward, norm 0 in $\|V\|_{1,0}$ is replaced with norm 1.

Separation

Transforming round 3 problem to many round 2 problems

The last problem is equivalent to solve c distinct LPs ($1 \leq j \leq c$):

$$\begin{aligned} \text{minimize} \quad & \|v_j\|_1 \\ \text{s.t.:} \quad & Sv_j = 0 \\ & l_j \preceq v_j \preceq u_j \end{aligned}$$

If the solver is super-linear (e.g. $O(n^{1+\delta})$ for arbitrary δ), having c problems of size n would be solved faster than a problem of size $c * n$.

Sharing weights in separated problems

$$
\begin{aligned}
\text{minimize } & \Sigma_{i=1}^{n} w_i \left\| v_i' \right\|_1 \\
& = \Sigma_{j=1}^{c} \left(\Sigma_{i=1}^{n} w_i |(v_j)_i| \right) \\
\text{s.t.: } & S v_j = 0 \quad \forall j \\
& l_j \preceq v_j \preceq u_j \quad \forall j
\end{aligned}
$$

In which v_i' denotes i'th row of V.
The weights are updated just the same as round 2, but by using $\left\| v_i'^{(t)} \right\|_2$ (or norm-1) instead of $|v_i^{(t)}|$ (weights would be projected on each separated problem).

Round 4

Original problem

$$
\begin{aligned}
&\text{minimize} \;\; \|V\|_{2,0} + \lambda \left\| (SV)^T \right\|_{2,0} \\
&\text{s.t.: } L \preceq V \preceq U
\end{aligned}
$$

Which means freeing each column of V from constraint $SV = 0$, has a penalty of λ.

Round 4

Norm 1,1 approximation, but freeing some columns

$$\begin{aligned} \text{minimize } & \Sigma_{i=1}^{n} w_i \left\| v_i' \right\|_1 \\ \text{s.t.: } & S v_j = 0 \quad \forall j \in J \\ & L \preceq V \preceq U \end{aligned}$$

In which:

$$J = \{ j \in \{1, \cdots, c\} \mid c_j < \lambda \}$$

$$\begin{aligned} c_j = & (\text{minimize } \left\| v_j \right\|_1 \text{ s.t. } S v_j = 0 \text{ and } l_j \preceq v_j \preceq u_j) \\ & -(\text{minimize } \left\| v_j \right\|_1 \text{ s.t. } l_j \preceq v_j \preceq u_j) \end{aligned}$$

(It is a heuristic of the advantage gained by freeing column j)

- Simultaneously optimize the terms in objective function, using norm 1,1 approximation and separation:

$$\text{minimize } \|V\|_{1,1} + \lambda \left\|(SV)^T\right\|_{1,1}$$
$$= \Sigma_{i=1}^{n} w_i \|v_i'\|_1 + \lambda \Sigma_{j=1}^{c} \hat{w}_j \|Sv_j\|_1$$
$$\text{s.t.: } L \preceq V \preceq U$$

and the separated problem j would be like:

$$\text{minimize } \Sigma_{i=1}^{n} w_i |(v_j)_i| + \lambda \hat{w}_j \|Sv_j\|_1$$
$$\text{s.t.: } l_j \preceq v_j \preceq u_j$$

with the same updating method (according to the l_2-norm of corresponding vector) and also normalized to be summed 1, to keep their proportion to be λ.

Round 5

Original problem

$$
\begin{aligned}
\text{minimize} \quad & \|V\|_{2,0} \\
\text{s.t.:} \quad & \left\|(SV)^T\right\|_{2,0} \leq K \\
& L \preceq V \preceq U
\end{aligned}
$$

Round 5

Norm 1,1 approximation, but freeing some columns

$$\begin{aligned} \text{minimize } & \Sigma_{i=1}^{n} w_i \left\| v_i' \right\|_1 \\ \text{s.t.: } & S v_j = 0 \quad \forall j \in J \\ & L \preceq V \preceq U \end{aligned}$$

In which:

$$J = \{1, \cdots, c\} \backslash \mathrm{K} - \underset{\{1,\cdots,c\}}{\operatorname{argmax}}(c_j)$$

$$\begin{aligned} c_j = & (\text{minimize } \ \left\| v_j \right\|_1 \ \text{ s.t. } S v_j = 0 \text{ and } l_j \preceq v_j \preceq u_j) \\ & -(\text{minimize } \ \left\| v_j \right\|_1 \ \text{ s.t. } l_j \preceq v_j \preceq u_j) \end{aligned}$$

(freeing K most advantageous columns)

Round 5

Other ideas

- Using round 4 code by binary searching on λ (start with some λ, solve the problem, then increase or decrease it if more or less than K columns of answer V are freed, respectively)

References

- Yun-Bin Zhao Sparse Optimization Theory and Methods, 2018
- Jialiang Xu & Yun-Bin Zhao Dual-density-based reweighted ℓ_1-algorithms for a class of ℓ_0-minimization problems, J Glob Optim (2021)
- https://jump.dev/MathOptInterface.jl/v0.8.1/apimanual/

OPTIMIZER

Sharif University of Technology

?SOAL Optimization and Applications Laboratory