An Attention Algorithm for Solving Large Scale Structured l_0-norm Penalty Estimation Problems

Tso-Jung Yen

Institute of Statistical Science
Academia Sinica

tjyen@stat.sinica.edu.tw

Department of Economics

Waseda University-Academia Sinica Joint Workshop

December 12, 2020

Outline

- Problem setting
- Algorithm
- Simulation experiments
- Discussion

Problem Setting

- Linear regression model:

$$\mathbf{y} = \sum_{j=1}^{m} \mathbf{X}_{G_j} \boldsymbol{\beta}_{\boldsymbol{G_j}} + \boldsymbol{\epsilon},$$

  where $\mathbf{y}$ is an n-dimensional response vector, $\boldsymbol{\epsilon}$ is an n-dimensional vector of i.i.d. errors, $\mathbf{X}_{G_j} = [\mathbf{x}_{[k]}]_{k \in G_j}$ is an $n \times p_j$ matrix with $p_j = |G_j|$, $G_j \subseteq \{1, 2, \cdots, p\}$ is an index set, and $\boldsymbol{\beta}_{\boldsymbol{G_j}} = \{\beta_k\}_{k \in G_j}$ is a p_j-dimensional vector of regression coefficients.

- Grouped variable selection problem:
  - Select $\mathbf{X}_{G_j}$'s for prediction; Estimate $\boldsymbol{\beta}_{\boldsymbol{G_j}} = \{\beta_k\}_{k \in G_j}$ jointly.
  - Estimated $\boldsymbol{\beta}_{\boldsymbol{G_j}}$ should either be zero-valued or contains non-zero entries.
  - Such a requirement can be done by using penalized estimation procedure with a structured penalty function.

Problem Setting

- Group variable selection problem (contd):
  - Penalty function:

$$\text{pen}(\boldsymbol{\beta}) = \sum_{j=1}^{m} \left(\lambda ||\boldsymbol{\beta}_{\boldsymbol{G_j}}||_0 + \tau p_j \mathbb{I}\{||\boldsymbol{\beta}_{\boldsymbol{G_j}}||_2 \neq 0\} \right), \tag{1}$$

    where $||\boldsymbol{\beta}_{\boldsymbol{G_j}}||_0 = \sum_{k \in G_j} \mathbb{I}\{\beta_k \neq 0\}$, and $\mathbb{I}\{||\boldsymbol{\beta}_{\boldsymbol{G_j}}||_2 \neq 0\}$ is an indicator function such that $\mathbb{I}\{||\boldsymbol{\beta}_{\boldsymbol{G_j}}||_2 \neq 0\} = 1$ if $||\boldsymbol{\beta}_{\boldsymbol{G_j}}||_2 \neq 0$ and $\mathbb{I}\{||\boldsymbol{\beta}_{\boldsymbol{G_j}}||_2 \neq 0\} = 0$ otherwise.

  - Structured l_0-norm penalized estimation:

$$\begin{aligned} \widehat{\boldsymbol{\beta}}^{\lambda,\tau} &= \arg\min_{\boldsymbol{\beta}} \left\{ \frac{1}{2} \left|\left| \mathbf{y} - \sum_{j=1}^{m} \mathbf{X}_{G_j} \boldsymbol{\beta}_{\boldsymbol{G_j}} \right|\right|_2^2 \right. \\ &\quad \left. + \sum_{j=1}^{m} \left(\lambda ||\boldsymbol{\beta}_{\boldsymbol{G_j}}||_0 + \tau p_j \mathbb{I}\{||\boldsymbol{\beta}_{\boldsymbol{G_j}}||_2 \neq 0\} \right) \right\}, \end{aligned}$$

    where $\boldsymbol{\beta} = (\boldsymbol{\beta}_{G_1}, \boldsymbol{\beta}_{G_2}, \cdots, \boldsymbol{\beta}_{G_m})$ is a $\sum_{j=1}^{m} p_j = p$-dimensional vector, $\lambda \geq 0$ and $\tau \geq 0$ are tuning parameters.

Problem Setting

- Group variable selection problem (contd):
  - Difficulties:
    - The estimation problem is NP-hard. An exact solution is not practical when p is large.
    - Approximate solutions may be more desirable.
  - Conventional approaches:
    - Gradient descent algorithm $+$ proximal operator $=$ Proximal gradient algorithm.
    - A customized proximal operator of the structured l_0-norm penalty function (1) is needed for carrying out iteration.
    - When p is very large, the gradient descent algorithm can be slow at each iteration.

Problem Setting

- Group variable selection problem (contd):
  - Our approach:
    - Blockwise coordinate descent algorithm;
    - Customized proximal operator of the structured l_0-norm penalty function (1);
    - Randomized attention mechanism focusing on blocks that are needed to be updated.

Algorithm

- The **StructZero** algorithm:

  1. **Pre-iteration stage:** Compute the initial values.
  2. **Iteration stage:** Run the iteration scheme under a given value of tuning parameters (τ, λ).
  3. **Post-iteration stage:** Compute model selection criteria (e.g. AIC, BIC or mean squared prediction error (MSPE)).

- At **2.Iteration stage**, the algorithm
  - Runs an iterative scheme in a *stochastic* and *non-cyclic* way;
  - Updates multiple blocks of parameters simultaneously at each iteration.

Algorithm

- Two iterative schemes:
  - **StructZero_Simpl**:

$$\boldsymbol{\beta_{G_j}}^r = \arg\min_{\boldsymbol{\beta_{G_j}}} \left\{ \frac{1}{2} \left|\left| \boldsymbol{\beta_{G_j}} - \left[\boldsymbol{\beta_{G_j}}^{r-1} - c_j \nabla_{\boldsymbol{\beta_{G_j}}} l(\boldsymbol{\beta}^{r-1}) \right] \right|\right|_2^2 + g(\boldsymbol{\beta_{G_j}}) \right\}, \quad (2)$$

  - **StructZero_Accel**:

$$\begin{aligned} \boldsymbol{\beta_{G_j}}^r &= \arg\min_{\boldsymbol{\beta_{G_j}}} \left\{ \frac{1}{2} \left|\left| \boldsymbol{\beta_{G_j}} - \left[\boldsymbol{\alpha_{G_j}}^{r-1} - c_j \nabla_{\boldsymbol{\beta_{G_j}}} l(\boldsymbol{\alpha}^{r-1}) \right] \right|\right|_2^2 + g(\boldsymbol{\beta_{G_j}}) \right\}, \\ z^r &= \frac{2}{r+2}, \\ \boldsymbol{\alpha_{G_j}}^r &= \boldsymbol{\beta_{G_j}}^r + \frac{z^r(1-z^{r-1})}{z^{r-1}} (\boldsymbol{\beta_{G_j}}^r - \boldsymbol{\beta_{G_j}}^{r-1}), \end{aligned} \quad (3)$$

where

$$g(\boldsymbol{\beta_{G_j}}) = \lambda ||\boldsymbol{\beta_{G_j}}||_0 + \tau p_j \mathbb{I}\{||\boldsymbol{\beta_{G_j}}||_2 \neq 0\}, \quad (4)$$

r is the number of iteration, and c_j is the stepsize.

Algorithm

- Two iterative schemes (contd):
  - **StructZero_Simpl** is an example of the generic iterative scheme commonly used in running the proximal gradient algorithm.
  - **StructZero_Accel** equips the momentum acceleration mechanism.
  - (2) and the first line of (3) are examples of the proximal operator on the function (4).
  - Both **StructZero_Simpl** and **StructZero_Accel** iterative schemes can be seen as examples of the blockwise coordinate descent algorithm if
    - They are carried out *deterministically*;
    - They are carried out in a *cyclic* way;
    - Only one block of parameters are updated at each iteration.

Algorithm

2. Iteration stage:

2.1. For $r = 1, 2, \cdots$, **max_iter**:

2.1.1. Compute the probability distribution Q^r for sampling the group indices $\{1, 2, \cdots, m\}$.

2.1.2. Select v group indices from $\{1, 2, \cdots, m\}$ according to Q^r. Let $\mathcal{H}_r$ denote the set of the selected indices.

2.1.3. For j in $\mathcal{H}_r$, run either **StructZero_Simpl** or **StructZero_Accel**.

2.1.4. For $j = 1, 2, \cdots, m$, compute the following quantity:

$$\xi_j^r = \begin{cases} \frac{||\boldsymbol{\beta}_{\boldsymbol{G_j}}{}^r - \boldsymbol{\beta}_{\boldsymbol{G_j}}{}^{r-1}||_2}{\sqrt{p_j}} \text{ if } j \in \mathcal{H}_r \\ \xi_j^{r-1} \text{otherwise} \end{cases} . \tag{5}$$

If the stopping criterion $m^{-1} \sum_{j=1}^{m} \xi_j^r \leq$ **tol_err**, then set $r =$ **max_iter** and terminate the iterative scheme.

2.2. Return to **2.1.** and run the iterative scheme with another value of (τ, λ).

Computational Details

- The **proximal operator** of the structured l_0-norm function (4):

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \left\{ \frac{1}{2} ||\boldsymbol{\theta} - \mathbf{b}||_2^2 + g(\boldsymbol{\theta}) \right\}, \tag{6}$$

where

$$g(\boldsymbol{\theta}) = \lambda ||\boldsymbol{\theta}||_0 + \tau p_j \mathbb{I}\{||\boldsymbol{\theta}||_2 \neq 0\},$$

- **Case 1:** When $\tau > 0$ and $\lambda = 0$, the kth element of the proximal operator (6) can be expressed as

$$\theta_k^* = b_k \mathbb{I}\{||\mathbf{b}||_2 \geq \sqrt{2\tau p_j}\}.$$

where b_k is the kth element of $\mathbf{b}$.

- **Case 2:** When $\tau = 0$ and $\lambda > 0$, the kth element of the proximal operator (6) can be expressed as

$$\theta_k^* = b_k \mathbb{I}\{|b_k| \geq \sqrt{2\lambda}\}. \tag{7}$$

Formula (7) is called the hard thresholding operator.

Computational Details

- The **attention distribution** $Q^r = \{Q_1^r, Q_2^r, \cdots, Q_m^r\}$ for sampling block indices $\{1, 2, \cdots, m\}$:

$$Q_j^r = \frac{\xi_j^{r-1} L_j + 10^{-8}}{\sum_{j'=1}^m \xi_{j'}^{r-1} L_{j'} + 10^{-8}},$$

  where $L_j = \Lambda_{\max}(\mathbf{X}_{G_j}^T \mathbf{X}_{G_j})$, i.e. the maximum eigenvalue of the Gram matrix $\mathbf{X}_{G_j}^T \mathbf{X}_{G_j}$, ξ_j^{r-1} is defined in (5), and 10^{-8} is a constant for the baseline probability.

- Why Q^r?
  - The iteration error at $(r-1)$th iteration (distance between the update at the $(r-1)$th iteration and the update at the $(r-2)$th iteration) is a signal indicating whether $\boldsymbol{\beta}_{\boldsymbol{G_j}}$ needs to be updated further or not at the rth iteration.
  - If the distance is small, i.e. the current update and the previous update of $\boldsymbol{\beta}_{\boldsymbol{G_j}}$ is close, then $\boldsymbol{\beta}_{\boldsymbol{G_j}}$ may not need a further update.
  - The algorithm can then turn its attention on those with larger distances between their current updates and previous updates.

Computational Details

- The **gradient** computation at each iteration:

$$\begin{aligned}\nabla_{\boldsymbol{\beta}_{\boldsymbol{G_j}}} l(\boldsymbol{\beta}^r) &= -[\mathbf{X}^T(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}^r)]_{G_j} \\ &= -\mathbf{a}_{G_j} + \mathbf{B}_{G_j}(\boldsymbol{\beta}^r_{G_j} - \boldsymbol{\beta}^{r-1}_{G_j}) + \mathbf{X}^T_{G_j}\mathbf{d}^{r-1}, \qquad (8)\end{aligned}$$

where

$$\begin{aligned}\mathbf{a}_{G_j} &= \mathbf{X}^T_{G_j}\mathbf{y}, \\ \mathbf{B}_{G_j} &= \mathbf{X}^T_{G_j}\mathbf{X}_{G_j}, \\ \mathbf{d}^{r-1} &= \mathbf{X}\boldsymbol{\beta}^{r-1}.\end{aligned}$$

- $\mathbf{a}_{G_j}$ is a p_j-dimensional vector, $\mathbf{B}_{G_j}$ is a $p_j \times p_j$ matrix; and $\mathbf{d}^{r-1}$ is an n-dimensional vector.

- $\mathbf{a}_{G_j}$ and $\mathbf{B}_{G_j}$ are computed at the pre-iteration stage, while $\mathbf{d}^{r-1}$ can be calculated incrementally at each iteration with computational cost $O(np_j)$.

- The computation cost of (8) at the rth iteration is $O(p_j^2) + O(np_j)$.

Simulation Experiments

- The ground truth model:

$$\mathbf{y} = \sum_{j \in \mathcal{H}^{\text{true}}} \mathbf{X}_{G_j} \boldsymbol{\beta}_{G_j}^{\text{true}} + \boldsymbol{\epsilon},$$

where $\mathbf{y}$ is an n-dimensional vector, $\mathbf{X}_{G_j}$ is an $n \times p_j$ matrix, $\boldsymbol{\epsilon} \sim \text{MVN}(0, \sigma_\epsilon^2 \mathbf{I}_{n \times n})$, and

$$\beta_k^{\text{true}} \sim \frac{1}{2}\text{Normal}(2, 0.25) + \frac{1}{2}\text{Normal}(-2, 0.25).$$

- Rows of $\mathbf{X}_{G_j}$ are i.i.d. $\text{MVN}(0, \boldsymbol{\Sigma}_{G_j})$.
- The residual variance σ_ϵ^2 is defined in terms of the signal-to-noise ratio (SNR) in a way such that

$$\text{SNR} = \frac{\sum_{j \in \mathcal{H}^{\text{true}}} (\boldsymbol{\beta}_{G_j}^{\text{true}})^T \boldsymbol{\Sigma}_{G_j} (\boldsymbol{\beta}_{G_j}^{\text{true}})}{\sigma_\epsilon^2}.$$

Simulation Experiments

- Programming language and computational environment:
    - The **StructZero** algorithm was coded in C++ using package “Rcpp” under the R programming environment.
    - When involving a large matrix, the corresponding computation was coded using “SparseMatrix” module in package “RcppEigen”.
    - The simulation experiments were carried out under a 64-bit Linux machine built on Dell’s PowerEdge server with 2 Intel Xeon E5-2650v4 (2.2GHz/12c) CPUs and 448 GB memory.

Simulation Experiments

- **Convergence of the algorithm:**
  - The number of *true* groups $|\mathcal{H}^{\text{true}}| = 3$, with sizes equal to 50, 60 and 40, respectively. All covariates in the three true groups have non-zero valued regression coefficients.
  - SNR (signal-to-noise ratio) $= 0.5$.
  - Two cases: $(n, m) = (2000, 1500)$ and $(n, m) = (50000, 5000)$, where n is the sample size, and m is the number of groups in the regression model.
  - The size of groups: $p_{\min} = 5$ and $p_{\max} = 40 \Rightarrow 32000 < p < 35000$ for the first case, and $p = 113,681$ for the second case.
  - In both cases $p > n$.
  - Other hyperparameters specified by researchers: v is the number of groups updated in each iteration; λ and τ are tuning parameters in the estimation.

Simulation Experiments

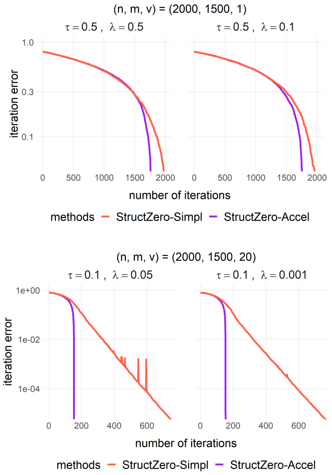
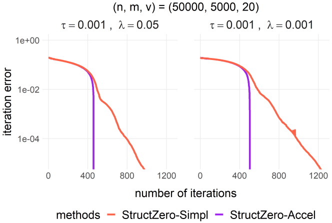


Figure: Plots of iteration error vs iteration number with $(n, m) = (2000, 1500)$. Top left: $(v, \tau, \lambda) = (1, 0.5, 0.5)$; Top right: $(v, \tau, \lambda) = (1, 0.5, 0.1)$; Bottom left: $(v, \tau, \lambda) = (20, 0.1, 0.05)$; Bottom right: $(v, \tau, \lambda) = (20, 0.1, 0.001)$.

Simulation Experiments



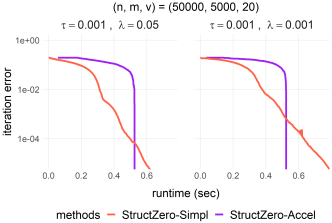


Figure: Plots of iteration error vs iteration number with $(n, m) = (50000, 5000)$. The two plots are based on results from the two experiments with $(v, \tau, \lambda) = (20, 0.001, 0.05)$ and $(v, \tau, \lambda) = (20, 0.001, 0.001)$, respectively; Top: Iteration error against the number of iterations; Bottom Iteration error against the runtime measured by second.

Simulation Experiments

- **Runtime of the algorithm:**
  - The number of *true* groups $|\mathcal{H}^{\text{true}}| = 3$, with sizes equal to 15, 37 and 22, respectively.
  - Each group has the number of zero valued regression coefficients equal to 5, 10 and 2, respectively $\Rightarrow$ The number of non-zero valued regression coefficients $= 57$.
  - SNR $= 0.5$; $n = 10,000$; $p_{\min} = 5$ and $p_{\max} = 40$.
  - 30 cases: The number of groups $m = 500, 1000, \cdots, 14500, 15000 \Rightarrow$ $11,739 \leq p \leq 339,608$. In all cases $p > n$.
  - **StructZero algorithm:** Tolerance error $= 10^{-6}$; $v = 200$; $\tau = 0$ (No groupwise variable selection); The number of $\lambda = 100$.
  - The number of CPU cores: 1, 5, 10, 20. Parallel computing over *tuning parameters*.
  - **Lasso estimation**: "glmnet" (version 2.0-13); 100 tuning parameter values; The *Strong Rule* is used for screening covariates at the pre-iteration stage.

Simulation Experiments

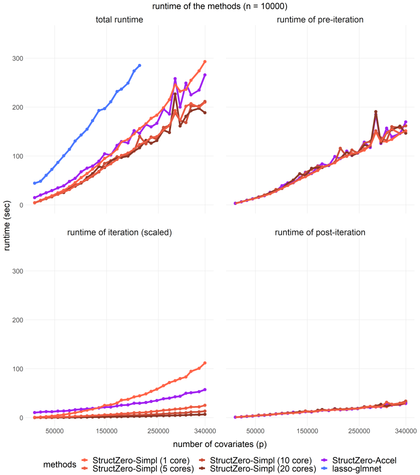


Figure: Runtime of the algorithms corresponding to the structured l_0-norm estimation and the lasso of the glmnet. Top left: Total runtime; Top right: Runtime of the pre-iteration stage; Bottom Left: Sum of 100 runtimes measured at the iteration stage scaled by the number of CPU cores; (d) Runtime of the post-iteration stage.

Simulation Experiments

- **Accuracy of the algorithm:**
  - Ground truth model: The same as the one in the previous section.
  - The number of groups $m = 1500 \Rightarrow$ The number of variables $32000 \leq p \leq 35000$.
  - 80 cases: Sample size $n = 500$, 2000, 5000 and 10000; 20 values of SNR ranging from 0.05 to 10.
  - **StructZero algorithm**: Tolerance error $= 10^{-6}$; $v = 200$; $\tau = 0$; The number of $\lambda = 100$.
  - **Lasso estimation**: "glmnet" (version 2.0-13); 100 tuning parameter values.
  - Tuning parameter selection criteria: Mean squared prediction error (MSPE). New data were used for evaluating the MSPE.
  - Performance criteria: (a) mean squared error (MSE), (b) True positive rate (TPE), and (c) false discovery rate (FDR).

Simulation Experiments

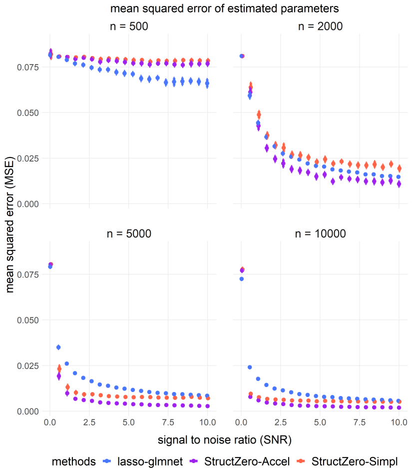


Figure: MSE of the algorithms corresponding to the structured l_0-norm estimation and the lasso of the glmnet. Top left: $n = 500$; Top right: $n = 2000$; Bottom left: $n = 5000$; Bottom right $n = 10000$.

Simulation Experiments

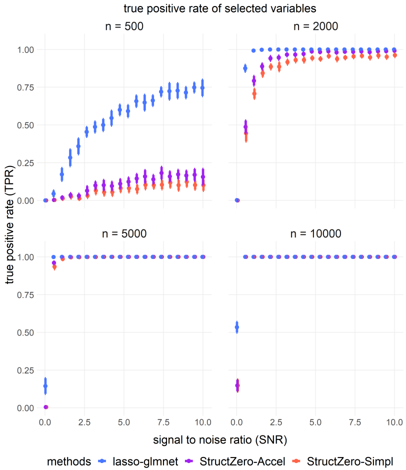


Figure: True positive rate of the algorithms corresponding to the structured l_0-norm estimation and the lasso of the glmnet. Top left: $n = 500$; Top right: $n = 2000$; Bottom left: $n = 5000$; Bottom right $n = 10000$.

Simulation Experiments

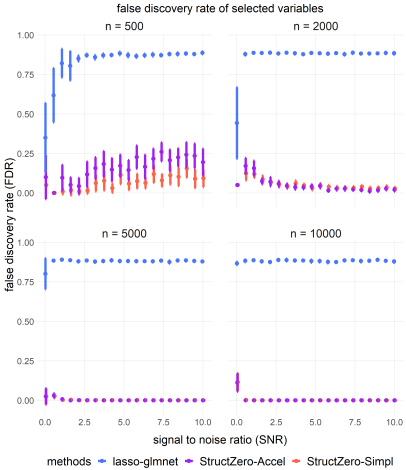


Figure: False discovery rate of the algorithms corresponding to the structured l_0-norm estimation and the lasso of the glmnet. Top left: $n = 500$; Top right: $n = 2000$; Bottom left: $n = 5000$; Bottom right $n = 10000$.

Simulation Experiments

- Sample complexity and signal-to-noise ratio:
  - **StructZero algorithm** vs **Lasso estimation**.
  - The ground truth model is the same as the one in the previous section.
  - 2500 cases: 50 values of sample size n from 200 to 10000, and 50 values of SNR from 0.1 to 5.0. 20 replicates for each case.
  - Performance criterion:

$$t\text{-statistics} = \frac{\overline{\mathsf{MSPE}}^{\mathsf{StructZero}} - \overline{\mathsf{MSPE}}^{\mathsf{lasso}}}{\sqrt{\frac{\widehat{\mathsf{Var}}(\mathsf{MSPE}^{\mathsf{StructZero}})}{20} + \frac{\widehat{\mathsf{Var}}(\mathsf{MSPE}^{\mathsf{lasso}})}{20}}},$$

    where $\overline{\mathsf{MSPE}}^{\mathsf{StructZero}}$ and $\overline{\mathsf{MSPE}}^{\mathsf{lasso}}$ are the sample mean of the corresponding MSPE values, while $\widehat{\mathsf{Var}}(\mathsf{MSPE}^{\mathsf{StructZero}})$ and $\widehat{\mathsf{Var}}(\mathsf{MSPE}^{\mathsf{lasso}})$ are estimated sample variances, and 20 is the number of replicates.

Simulation Experiments

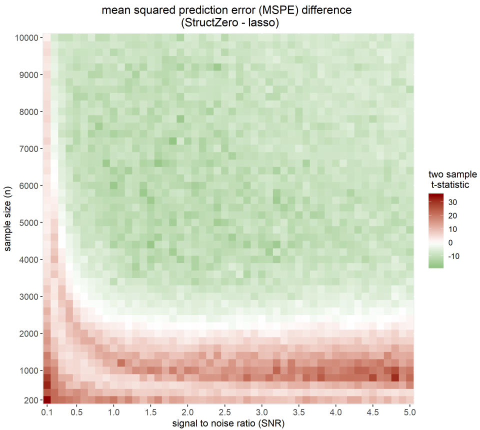


Figure: Heatmap of the two sample t-statistic for mean squared prediction error (MSPE) between the structured l_0-norm estimation and the lasso estimation. In each plot the x-axis is the signal-to-noise ratio (SNR), and the y-axis is the sample size n. Colored points refer to values of the two sample t-statistic.

Discussion

- In this talk, we have
  - Developed algorithms for carrying out structured l_0-norm estimation with large data;
  - Derived closed form representations for the proximal operator of the structured $l_0$norm penalty function;
  - Developed an attention mechanism for accelerating the iteration procedure.
- Future research directions:
  - **Hyperparameter selection:** The hyperparameter v, the number of groups updated at each iteration, plays an important role in running our algorithm.
  - **Convergence analysis:** Available mathematical tools such as the "expected separable overapproximation" and the "Kurdyka-Łojasiewicz inequality" may be helpful here.