Efficient statistical learning of complex data

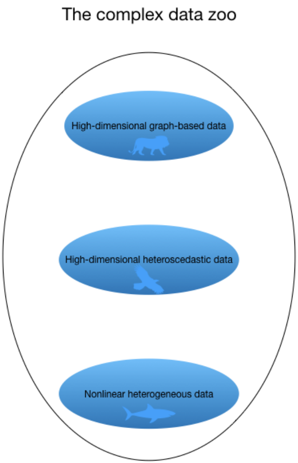
Duzhe Wang

University of Wisconsin-Madison
Department of Statistics

Ph.D. Thesis Defense
November 30, 2020

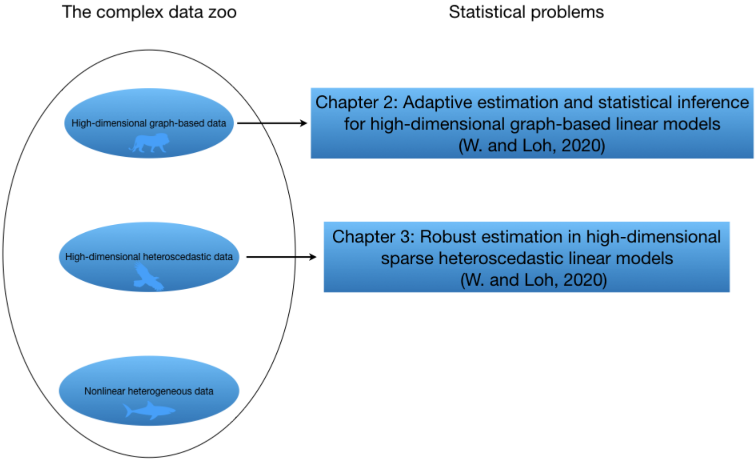
Dissertation committee: Prof. Po-Ling Loh, Prof. Varun Jog, Prof. Hyunseung Kang, Prof. Vivak Patel, Prof. Anru Zhang

Thesis overview



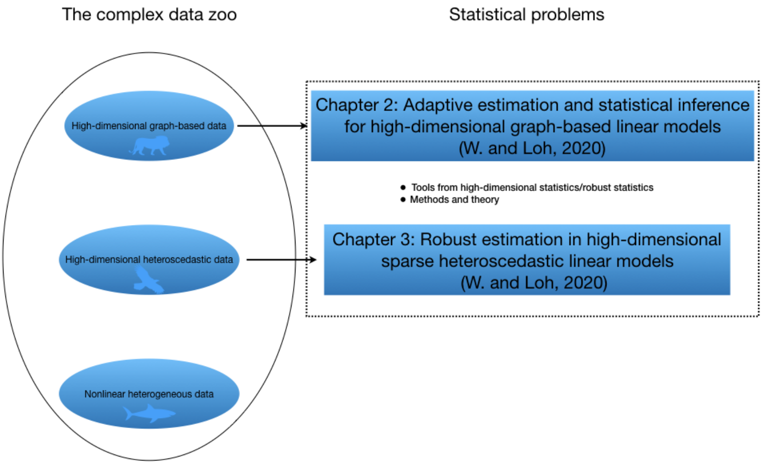
Thanks for showing me around. It's a fun and memorable trip!

Thesis overview



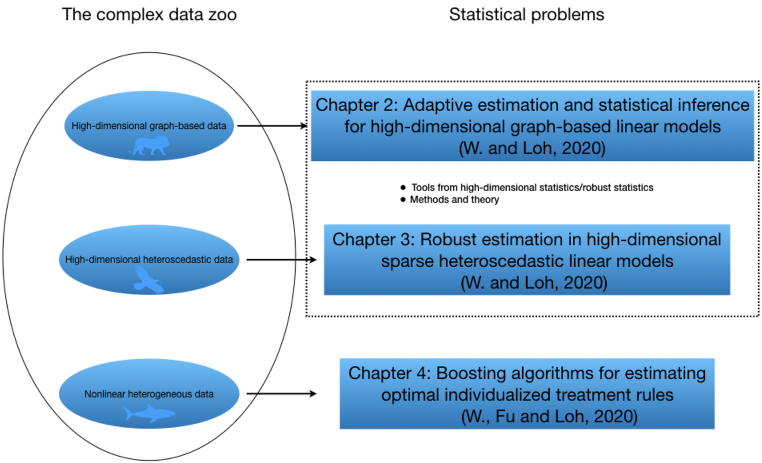
Thanks for showing me around. It's a fun and memorable trip!

Thesis overview



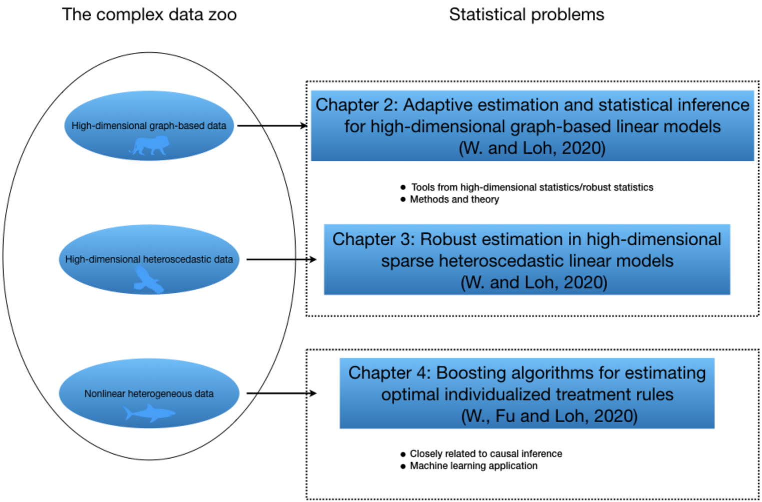
Thanks for showing me around. It's a fun and memorable trip!

Thesis overview



Thanks for showing me around. It's a fun and memorable trip!

Thesis overview



Thanks for showing me around. It's a fun and memorable trip!

Outline of today's talk

- High-dimensional linear models

- Beyond linear models

Outline of today's talk

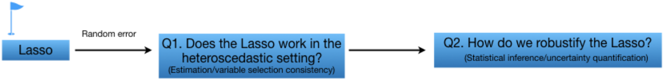
- High-dimensional linear models

Lasso

- Beyond linear models

Outline of today's talk

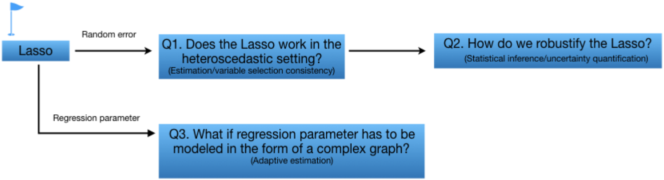
- High-dimensional linear models



- Beyond linear models

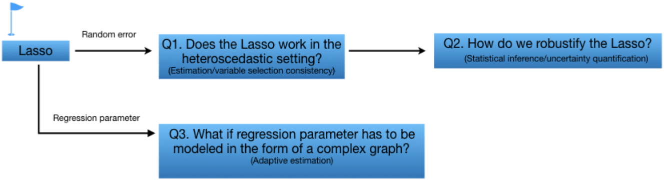
Outline of today's talk

- High-dimensional linear models

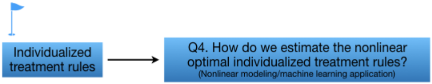


- Beyond linear models

- High-dimensional linear models

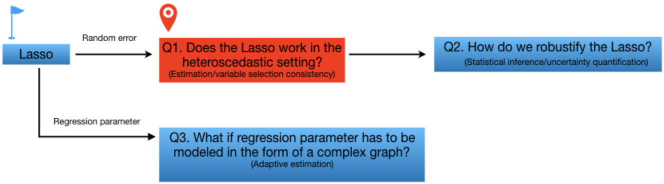


- Beyond linear models

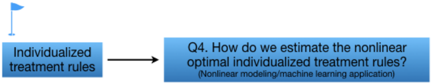


Outline of today's talk

- High-dimensional linear models



- Beyond linear models



High-dimensional sparse heteroscedastic linear models

$$y = X\beta^* + \varepsilon$$

- $X = (X_1, \ldots, X_N)^T \in \mathbb{R}^{N \times n}$: sub-Gaussian design matrix with $X_i \in \mathbb{R}^n$
- $y = (y_1, \ldots, y_N)^T \in \mathbb{R}^N$: response vector
- $\beta^* \in \mathbb{R}^n$: true s-sparse regression coefficients
- $\varepsilon = (\varepsilon_1, \ldots, \varepsilon_N)^T \in \mathbb{R}^N$: ε_i's are independent, conditionally normal random variables with

$$\mathbb{E}(\varepsilon_i \mid X_i) = 0, \quad \mathbb{E}\left(\varepsilon_i^2 \mid X_i\right) = W_i$$

- High-dimensional setting: $N \ll n$
- Conditional heteroscedasticity: $W_i = g(X_i, \beta^*)$
  - Parametric function form of g is known
  - An example in PET imaging: $W_i = c|X_i^T \beta^*|$ (Jia, Rohe and Yu, 2013)
- Lasso:

$$\hat{\beta} = \underset{\beta \in \mathbb{R}^n}{\operatorname{argmin}} \quad \frac{1}{2N} \|y - X\beta\|_2^2 + \lambda \|\beta\|_1.$$

Estimation consistency

Theorem

Assume that $0 < L_1 \le W_i \le L_2 < \infty$ for $1 \le i \le N$. If $\lambda \gtrsim \sqrt{\frac{L_2 \log n}{N}}$, then with high probability, we have $\|\hat{\beta} - \beta^*\|_2 \lesssim \lambda\sqrt{s}$ and $\|\hat{\beta} - \beta^*\|_1 \lesssim \lambda s$.

Estimation consistency

Theorem

Assume that $0 < L_1 \le W_i \le L_2 < \infty$ for $1 \le i \le N$. If $\lambda \gtrsim \sqrt{\frac{L_2 \log n}{N}}$, then with high probability, we have $\|\hat{\beta} - \beta^*\|_2 \lesssim \lambda\sqrt{s}$ and $\|\hat{\beta} - \beta^*\|_1 \lesssim \lambda s$.

- If $\lambda \asymp \sqrt{\frac{L_2 \log n}{N}}$, then

$$\left\|\hat{\beta} - \beta^*\right\|_2 \le \mathcal{O}_{\mathbb{P}}\left(\sqrt{\frac{sL_2 \log n}{N}}\right) \quad \text{and} \quad \left\|\hat{\beta} - \beta^*\right\|_1 \le \mathcal{O}_{\mathbb{P}}\left(s\sqrt{\frac{L_2 \log n}{N}}\right)$$

- The Lasso is ℓ_2-consistent if $\frac{sL_2 \log n}{N} = o(1)$
- If $W_i = \sigma_\varepsilon^2$ for $i = 1, \ldots, N$ (homoscedastic case), then

$$\left\|\hat{\beta} - \beta^*\right\|_2 \le \mathcal{O}_{\mathbb{P}}\left(\sigma_\varepsilon\sqrt{\frac{s \log n}{N}}\right) \quad \text{and} \quad \left\|\hat{\beta} - \beta^*\right\|_1 \le \mathcal{O}_{\mathbb{P}}\left(\sigma_\varepsilon s\sqrt{\frac{\log n}{N}}\right)$$

Variable selection consistency

Mutual incoherence condition

Let S denote the support set of β^*. For covariance matrix Σ_x, there exists a constant $\alpha \in (0,1)$, such that $\left\|(\Sigma_x)_{S^cS}(\Sigma_x)_{SS}^{-1}\right\|_\infty \le \frac{\alpha}{2}$.

Theorem

Assume that mutual incoherence condition holds. Let $\widehat{S}$ be the support set of $\hat{\beta}$. Let $\lambda \gtrsim \frac{4}{1-\alpha}\sqrt{\frac{L_2 \log(n-s)}{N}}$.

(a) With high probability, we have $\widehat{S} \subseteq S$.

(b) If $\min_{i\in S} |\beta_i^*| > \sqrt{\frac{8}{\lambda_{\min}(\Sigma_x)}}\sqrt{\frac{L_2 \log s}{N}} + \lambda\left(\left\|(\Sigma_x)_{SS}^{-1}\right\|_\infty + cs\sqrt{\frac{s}{N}}\right)$, then with high probability, we have $S = \widehat{S}$.

$\|M\|_\infty = \max_{i=1,\dots,m} \sum_{j=1}^n |M_{ij}|$

Hurdles

(1) Characterizing (asymptotic) distribution of the Lasso is difficult

- KKT condition:

$$M\sqrt{N}(\hat{\beta} - \beta^*) + \sqrt{N}\lambda\hat{k} = \frac{1}{\sqrt{N}}X^T\varepsilon,$$

  where $M = \frac{1}{N}X^TX$ and $\hat{k}$ be the subgradient of $\|\cdot\|_1$ at $\hat{\beta}$

- Low-dimensional setting:

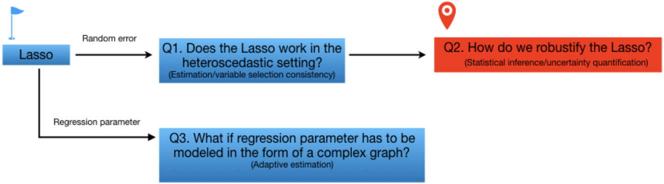
$$\sqrt{N}(\hat{\beta} - \beta^*) = \frac{1}{\sqrt{N}}M^{-1}X^T\varepsilon - \sqrt{N}\lambda M^{-1}\hat{k}$$

- High-dimensional setting: hard to characterize $\sqrt{N}(\hat{\beta} - \beta^*)$

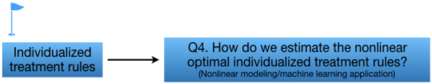
(2) No side/prior information on heteroscedasticity is used in the Lasso

Outline of today’s talk

- High-dimensional linear models



- Beyond linear models



Robustifying the Lasso under heteroscedasticity

Algorithm

Input: dataset $\{(X_i, y_i)\}_{i=1}^N$, formula of g, tuning parameters λ and μ.

(1) Solve the Lasso to obtain a preliminary estimator $\hat{\beta}$.

(2) Set $\widehat{W}_i = g(X_i, \hat{\beta})$, $\widehat{W} = \mathrm{diag}(\widehat{W}_1, \dots, \widehat{W}_N)$, and $\widehat{\Sigma}_N = \frac{1}{N} X^T \widehat{W}^{-1} X$.

(3) Solve the following optimization problem to obtain $\widehat{\Theta}$:

$$\boxed{\begin{aligned} \widehat{\Theta} = \underset{\Theta \in \mathbb{R}^{n\times n}}{\operatorname{argmin}} \quad & \|\Theta\|_{1,1} = \sum_{i=1}^{n}\sum_{j=1}^{n} |\Theta_{ij}| \\ \text{subject to} \quad & \|\Theta \widehat{\Sigma}_N - I_n\|_\infty \le \mu. \end{aligned}}$$

(4) Output the final estimator:

$$\tilde{\beta} = \hat{\beta} + \frac{1}{N} \widehat{\Theta} X^T \widehat{W}^{-1} (y - X\hat{\beta}).$$

Robustifying the Lasso under heteroscedasticity

Algorithm

Input: dataset $\{(X_i, y_i)\}_{i=1}^N$, formula of g, tuning parameters λ and μ.

(1) Solve the Lasso to obtain a preliminary estimator $\hat{\beta}$.

(2) Set $\widehat{W}_i = g(X_i, \hat{\beta})$, $\widehat{W} = \mathrm{diag}(\widehat{W}_1, \ldots, \widehat{W}_N)$, and $\widehat{\Sigma}_N = \frac{1}{N} X^T \widehat{W}^{-1} X$.

(3) Solve the following optimization problem to obtain $\widehat{\Theta}$:

$$\widehat{\Theta} = \underset{\Theta \in \mathbb{R}^{n \times n}}{\mathrm{argmin}} \quad \|\Theta\|_{1,1} = \sum_{i=1}^n \sum_{j=1}^n |\Theta_{ij}|$$
$$\text{subject to} \quad \|\Theta \widehat{\Sigma}_N - I_n\|_\infty \le \mu.$$

$$\widehat{\Theta} = (\hat{\theta}_1, \ldots, \hat{\theta}_n)^T$$
$$\hat{\theta}_i = \underset{\theta \in \mathbb{R}^n}{\mathrm{argmin}} \quad \|\theta\|_1$$
$$\text{subject to} \quad \left\|\widehat{\Sigma}_N \theta - e_i\right\|_\infty \le \mu.$$

(4) Output the final estimator:

$$\tilde{\beta} = \hat{\beta} + \frac{1}{N} \widehat{\Theta} X^T \widehat{W}^{-1} (y - X\hat{\beta}).$$

Ideas

- One-step MLE in the low-dimensional setting:

$$\hat{\beta}_1 = \hat{\beta}_0 - [\nabla S(\hat{\beta}_0)]^{-1} S(\hat{\beta}_0),$$

  where $S(\beta)$ is the score function and $\hat{\beta}_0$ is an initial estimator of β^*
- If W is known, then in the low-dimensional setting, one-step MLE is

$$\hat{\beta}_1 = \hat{\beta}_0 + \left(\frac{1}{N} X^T W^{-1} X\right)^{-1} \frac{1}{N} X^T W^{-1} (y - X\hat{\beta}_0)$$

Ideas

- One-step MLE in the low-dimensional setting:

$$\hat{\beta}_1 = \hat{\beta}_0 - [\nabla S(\hat{\beta}_0)]^{-1} S(\hat{\beta}_0),$$

  where $S(\beta)$ is the score function and $\hat{\beta}_0$ is an initial estimator of β^*
- If W is known, then in the low-dimensional setting, one-step MLE is

$$\hat{\beta}_1 = \hat{\beta}_0 + \left(\frac{1}{N} X^T W^{-1} X\right)^{-1} \frac{1}{N} X^T W^{-1} (y - X\hat{\beta}_0)$$

- Challenges:
  (1) In the high-dimensional setting, $\frac{1}{N} X^T W^{-1} X$ is singular
  (2) In the heteroscedastic setting, W is unknown
- Solutions:
  (1) Find a sparse approximate inverse
  (2) Utilize the side information to estimate W

Uncertainty quantification

Theorem

Assume that $\Sigma^* = \mathbb{E}\left(\frac{X_i X_i^T}{W_i}\right)$ is positive definite. Let $\Theta^* \in \mathbb{R}^{n\times n}$ denote the inverse of Σ^*. Let

$$\lambda \asymp \sqrt{\frac{L_2 \log n}{N}} \quad \text{and} \quad \mu \asymp \frac{1}{L_1}\sqrt{\frac{\log n}{N}} + s\sqrt{\frac{L_2 \log n}{N}}.$$

Under a set of assumptions, we have for $1 \le j \le n$,

$$\sqrt{N}\left(\tilde{\beta}_j - \beta_j^*\right) \xrightarrow{d} N\left(0, e_j^T \Theta^* e_j\right).$$

Statistical inference

- Similar arguments yield

$$\frac{\sqrt{N}\left(\tilde{\beta}_j - \beta_j^*\right)}{\sqrt{e_j^T \widehat{\Theta} \widehat{\Sigma}_N \widehat{\Theta}^T e_j}} \to N(0,1)$$

- Confidence interval: let $\Phi(x)$ be the cumulative distribution function of $N(0,1)$. Then

$$\left[\tilde{\beta}_j - \Phi^{-1}\left(1-\frac{\alpha}{2}\right)\sqrt{\frac{e_j^T \widehat{\Theta} \widehat{\Sigma}_N \widehat{\Theta}^T e_j}{N}}, \tilde{\beta}_j + \Phi^{-1}\left(1-\frac{\alpha}{2}\right)\sqrt{\frac{e_j^T \widehat{\Theta} \widehat{\Sigma}_N \widehat{\Theta}^T e_j}{N}}\right]$$

  provides an asymptotically valid $(1-\alpha)$-confidence interval for β_j^*
- Hypothesis testing: test statistic for testing whether β_j^* is equal to 0

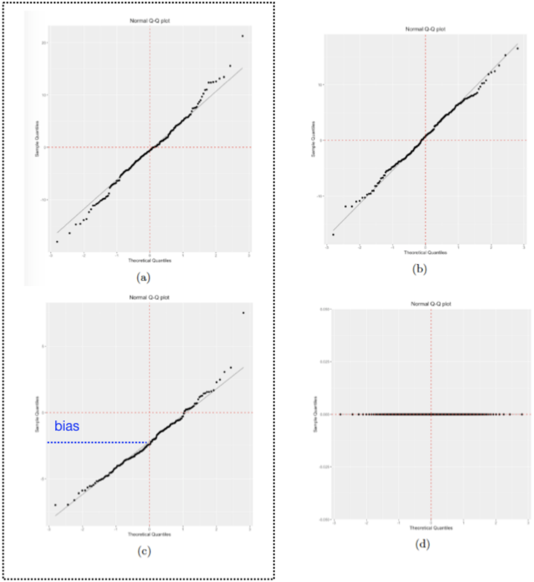
Simulation settings

- $(N, n) = (120, 150)$
- $\beta^* = (3, 4, 3, 1.5, 2, 1.5, 0, ..., 0)^T$
- $X_i \sim N(0, \Sigma_x)$, where $(\Sigma_x)_{S^c S} = 0$, and $(\Sigma_x)_{ij} = 0.5^{|i-j|}$ if both i and j in are S or both i and j are in S^c
- $\varepsilon_i \mid X_i \sim N(0, W_i)$, where

$$W_i = \min\left(\frac{1}{25}\exp\left(\frac{1}{2}\left|X_i^T\beta^*\right|\right), 5\right)$$

- $y_i = X_i^T\beta^* + \varepsilon_i$

Simulation results



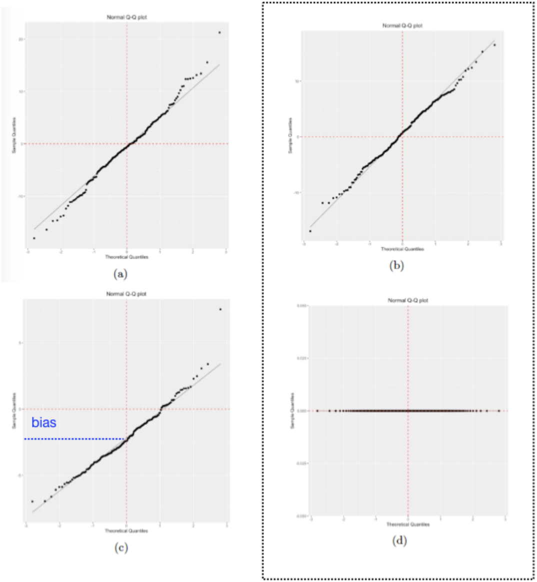
(a) Q-Q plot of $\sqrt{N}\left(\tilde{\beta}_5 - \beta_5^*\right)$

(b) Q-Q plot of $\sqrt{N}\left(\tilde{\beta}_7 - \beta_7^*\right)$

(c) Q-Q plot of $\sqrt{N}\left(\hat{\beta}_5 - \beta_5^*\right)$

(d) Q-Q plot of $\sqrt{N}\left(\hat{\beta}_7 - \beta_7^*\right)$

Simulation results



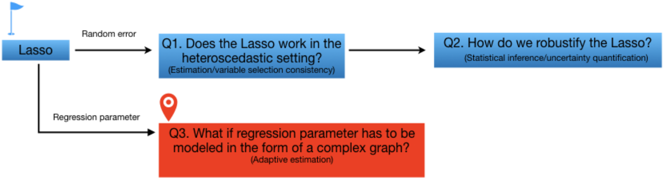
(a) Q-Q plot of $\sqrt{N}\left(\tilde{\beta}_5 - \beta_5^*\right)$

(b) Q-Q plot of $\sqrt{N}\left(\tilde{\beta}_7 - \beta_7^*\right)$

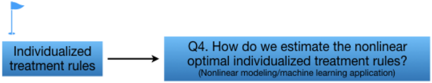
(c) Q-Q plot of $\sqrt{N}\left(\hat{\beta}_5 - \beta_5^*\right)$

(d) Q-Q plot of $\sqrt{N}\left(\hat{\beta}_7 - \beta_7^*\right)$

- High-dimensional linear models

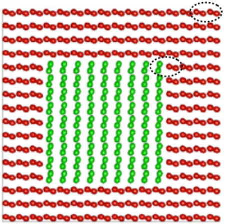


- Beyond linear models



Motivating examples

- Diffusion tensors



Synthetic DTI field (cited from Liu et al., 2013)

- Each pixel corresponds to one diffusion tensor
- Most adjacent diffusion tensors are same or vary smoothly
- Diffusion tensors vary sharply in some tissue boundary
- Stejskal-Tanner model: $y = X\beta + \varepsilon$
  - y: diffusion signal intensities across all pixels with a log scale
  - X: design matrix including b-values and directions of diffusion gradients
  - β: diffusion tensors across all pixels

Motivating examples

- Gene expression

Metabolic pathways (cited from Wikipedia)

- Identify genes which are associated with a target gene among hundreds of candidates from a large number of metabolic pathways
- Genes within a same cluster (pathway) have similar patterns
- Model: $y = X\beta + \varepsilon$
  - y: expression levels of the target gene
  - X: design matrix including expression levels of candidate genes
  - β: association levels between genes

Motivating examples

- Gene expression

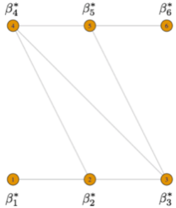
Metabolic pathways (cited from Wikipedia)

- Identify genes which are associated with a target gene among hundreds of candidates from a large number of metabolic pathways
- Genes within a same cluster (pathway) have similar patterns
- Model: $y = X\beta + \varepsilon$
    - y: expression levels of the target gene
    - X: design matrix including expression levels of candidate genes
    - β: association levels between genes

- Conclusion: these high-dimensional datasets have structures that can be captured in the form of complex graphs

High-dimensional graph-based linear models

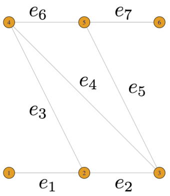
$$y = X\beta^* + \varepsilon$$



- $X = (X_1, \ldots, X_N)^T \in \mathbb{R}^{N\times n}$: design matrix with $X_i \in \mathbb{R}^n$
- $y = (y_1, \ldots, y_N)^T \in \mathbb{R}^N$: response vector
- $\beta^* \in \mathbb{R}^n$: unknown true regression coefficients
- $\varepsilon = (\varepsilon_1, \ldots, \varepsilon_N)^T \in \mathbb{R}^N$: i.i.d. random error
- High-dimensional setting: $N \ll n$
- Graph setting: coordinates of β^* correspond to nodes of some known underlying undirected graph
- Goal: estimate sparse regression coefficients with certain graph-based structure

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be an undirected graph with $|\mathcal{V}| = n$ and $|\mathcal{E}| = p$.

- Oriented incidence matrix $F \in \{-1, 0, 1\}^{p \times n}$: if the kth edge is $(i, j) \in \mathcal{E}$ with $i < j$, then the kth row of F is $(0, \dots, -1, \dots., +1, \dots, 0)$



$$F = \begin{bmatrix} -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 \end{bmatrix}$$

- Laplacian matrix: $L = F^T F \in \mathbb{R}^{n \times n}$
- Higher-order graph difference operator: graph difference operator of order $k + 1$ is

$$\Delta^{(k+1)} = \begin{cases} F^T \Delta^{(k)} = L^{\frac{k+1}{2}} \in \mathbb{R}^{n \times n} & \text{for odd k} \\ F \Delta^{(k)} = F L^{\frac{k}{2}} \in \mathbb{R}^{p \times n} & \text{for even k.} \end{cases}$$

Graph-based structure

Graph-based piecewise polynomial structure (Wang et al., 2016)

for $k \geq 0$ and $s > 0$, β^* is (k, s)-piecewise polynomial over the graph $\mathcal{G}$ if

$$\|\Delta^{(k+1)}\beta^*\|_0 \leq s$$

- $k = 0$: piecewise constant
- $k = 1$: piecewise linear
- $k = 2$: piecewise quadratic
- Local smoothness
- Focus on simultaneously sparse and piecewise polynomial regression coefficients:

$$\beta^* \in \mathcal{S}(k, s_1, s_2) = \left\{\beta \in \mathbb{R}^n : \|\Delta^{(k+1)}\beta\|_0 \leq s_1, \|\beta\|_0 \leq s_2\right\}$$

Graph-based adaptive estimation

k-th order Graph-Piecewise-Polynomial-Lasso

$$\hat{\beta}_g = \underset{\beta \in \mathbb{R}^n}{\operatorname{argmin}} \quad \frac{1}{2N} \|y - X\beta\|_2^2 + \lambda \|\beta\|_1 + \lambda_g \|\Delta^{(k+1)} \beta\|_1$$

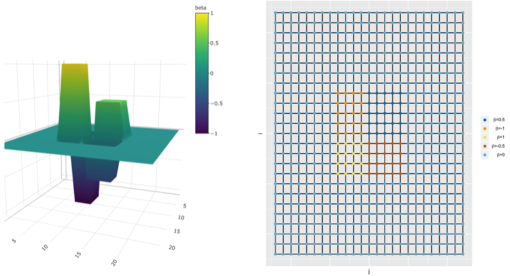
- $k = 0$: Graph-Fused Lasso (Kim, Sohn and Xing, 2019)
- Assume k is known for theory, but tune k in practice
- Equivalent formulation:

$$\hat{\beta}_g = \underset{\beta \in \mathbb{R}^n}{\operatorname{argmin}} \quad \frac{1}{2N} \|y - X\beta\|_2^2 + \lambda \|D\beta\|_1 ,$$

where $D = \begin{bmatrix} \frac{\lambda_g}{\lambda} \Delta^{(k+1)} \\ I_n \end{bmatrix}$

- Graph-Piecewise-Polynomial-Lasso can be solved efficiently via the ADMM algorithm

Simulation settings



$$B^*_{ij} = \begin{cases} 0.5 & (i,j) \in [9,13] \times [13,17] \\ -1 & (i,j) \in [9,13] \times [9,12] \\ +1 & (i,j) \in [14,17] \times [9,12] \\ -0.5 & (i,j) \in [14,17] \times [13,17] \\ 0 & \text{otherwise.} \end{cases}$$

- Coordinates of β^* correspond to a 2d grid graph with 25 rows and 25 columns ($n = 625, p = 1200$)
- $\beta^* \in \mathbb{R}^{625}$ is sparse and piecewise constant ($s_1 = 54, s_2 = 81$)
- Construction of β^*: first construct $B^* \in \mathbb{R}^{25\times 25}$, then stack columns of B^* on top of one another
- $X_i \sim N(0, I_{n\times n})$; $\varepsilon_i \sim N(0, 0.1)$; $y_i = X_i^T \beta^* + \varepsilon_i$
- $N = 250$

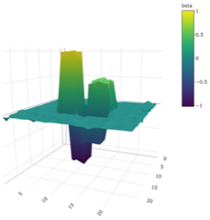
Simulation results

- Graph-Smooth-Lasso:

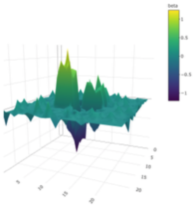
$$\hat{\beta}^{\text{gsmooth}} = \underset{\beta \in \mathbb{R}^n}{\operatorname{argmin}} \quad \frac{1}{2N} \| y - X\beta \|_2^2 + \lambda_1 \| \beta \|_1 + \lambda_2 \| \Delta^{(1)} \beta \|_2^2$$

- Graph-Spline-Lasso:

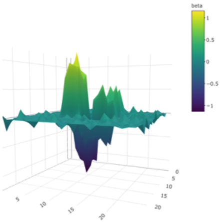
$$\hat{\beta}^{\text{gspline}} = \underset{\beta \in \mathbb{R}^n}{\operatorname{argmin}} \quad \frac{1}{2N} \| y - X\beta \|_2^2 + \lambda_1 \| \beta \|_1 + \lambda_2 \| \Delta^{(2)} \beta \|_2^2$$



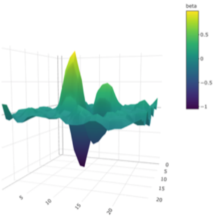
Our approach



Lasso



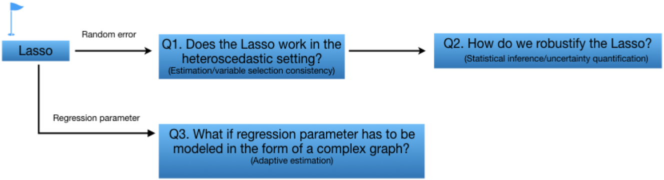
Graph-Smooth-Lasso



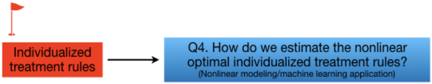
Graph-Spline-Lasso

Outline of today's talk

- High-dimensional linear models

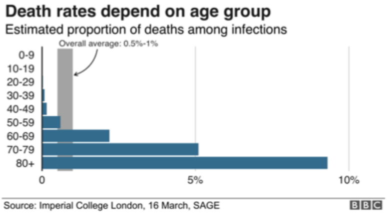


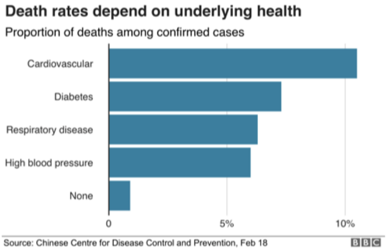
- Beyond linear models



A motivating example

- COVID-19 patients are a very heterogeneous population





A motivating example

- Heterogeneous treatment effects:

"Primary efficacy analysis demonstrates BNT162b2 to be 95% *effective against COVID-19 beginning 28 days after the first dose;170 confirmed cases of COVID-19 were evaluated, with 162 observed in the placebo group versus 8 in the vaccine group"*

—Phase 3 study results of Pfizer and BioNTech's vaccine

"Today's primary analysis was based on 196 cases, of which 185 cases of COVID-19 were observed in the placebo group versus 11 cases observed in the mRNA-1273 group, resulting in a point estimate of vaccine efficacy of 94.1%*."*

—Phase 3 study results of Moderna's vaccine

Hope of precision medicine

Hope of precision medicine

Individualized treatment rules

- $\{(X_i, A_i, Y_i), 1 \le i \le n\}$: i.i.d. observations of (X, A, Y)
  - $X \subset \mathcal{X} \subset \mathbb{R}^p$: prognostic variables
  - $A \subset \mathcal{A} = \{-1, +1\}$: the given treatment
  - $Y \subset \mathbb{R}$: the patient clinical outcome (with larger being better)
- Individualized treatment rule (ITR):

$$\mathcal{D} : \mathcal{X} \to \{-1, +1\}$$

  - e.g., $\mathcal{D}(x) = 1$, $\mathcal{D}(x) = \text{sign}(x^T 1)$

Individualized treatment rules

- $\{(X_i, A_i, Y_i), 1 \leq i \leq n\}$: i.i.d. observations of (X, A, Y)
  - $X \subset \mathcal{X} \subset \mathbb{R}^p$: prognostic variables
  - $A \subset \mathcal{A} = \{-1, +1\}$: the given treatment
  - $Y \subset \mathbb{R}$: the patient clinical outcome (with larger being better)
- Individualized treatment rule (ITR):

$$\mathcal{D} : \mathcal{X} \to \{-1, +1\}$$

  - e.g., $\mathcal{D}(x) = 1$, $\mathcal{D}(x) = \text{sign}(x^T 1)$
- The optimal ITR $\mathcal{D}^*(x)$:

$$\mathcal{D}^*(x) = \underset{a \in \mathcal{A}}{\text{argmax}} \quad \underbrace{Q(x, a) := \mathbb{E}(Y | x, a)}_{\text{Quality}}$$

Two learning frameworks

Generic method of indirect learning

(1) Assume $Q(x,1)$ and $Q(x,-1)$ are in some specified functional space $\mathcal{F}$

(2) Estimate $Q(x,1)$ and $Q(x,-1)$: this is a regression problem

(3) Estimated optimal ITR:

$$\widehat{\mathcal{D}}(x) = \text{sign}\left(\widehat{Q}(x,1) - \widehat{Q}(x,-1)\right)$$

Generic method of direct learning

(1) Note $\mathcal{D}^*(x) = \text{sign}(f^*(x))$. Assume $f^*(x) \in \mathcal{F}$

(2) Estimate $f^*(x)$: this is a classification problem

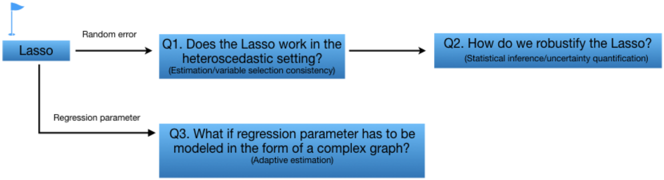
(3) Estimated optimal ITR:

$$\widehat{\mathcal{D}}(x) = \text{sign}(\hat{f}(x))$$

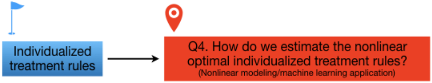
- Goal: estimate the nonlinear and complex optimal ITR $\mathcal{D}^*(x)$ with the observed dataset

Outline of today's talk

- High-dimensional linear models



- Beyond linear models



Our first proposed method

- One instance of indirect learning
- Key ideas:
  - Additive regression trees: assume

$$Q(x,1) = \sum_{t=1}^{K} b_1^{(t)}(x),$$

    and

$$Q(x,-1) = \sum_{t=1}^{K} b_{-1}^{(t)}(x),$$

    where $b_1^{(t)}(x)$ and $b_{-1}^{(t)}(x)$ are regression trees
  - Use boosting algorithm to estimate regression trees sequentially

XGBoost algorithm

- 1st iteration:

Estimation of $b_1^{(1)}$

(1) Fit a tree to the training data (X_i, Y_i):

$$\hat{f}^{(1)} = \underset{f}{\operatorname{argmin}} \sum_{i:A_i=1} (Y_i - f(X_i))^2 + J(f)$$

- $f(x)$ is a regression tree: $f(x) = w_{q(x)}(q : \mathbb{R}^p \to T, w \in \mathbb{R}^{|T|})$, where q represents the tree structure and T represents the leaves
- $J(f)$ is the cost complexity of a regression tree: $J(f) = \gamma|T| + \frac{1}{2}\lambda\|w\|_2^2$

(2) Shrinkage: $\hat{b}_1^{(1)} = \eta\hat{f}^{(1)}$, where $0 < \eta < 1$

XGBoost algorithm

- tth iteration ($t > 1$):

Estimation of $b_1^{(t)}$

(1) After $(t-1)$th iteration: $\hat{Y}_i^{(t-1)} = \sum_{k=1}^{t-1} \hat{b}_1^{(k)}(X_i)$ is the estimated outcome value of X_i

(2) Fit a tree to the training data (X_i, Y_i):

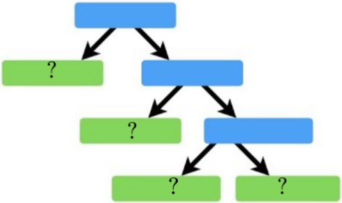
$$\hat{f}^{(t)} = \underset{f}{\operatorname{argmin}} \sum_{i:A_i=1} [Y_i - (\hat{Y}_i^{(t-1)} + f(X_i))]^2 + J(f),$$

(3) Shrinkage: $\hat{b}_1^{(t)} = \eta \hat{f}^{(t)}$

- Output the boosted model:

$$\widehat{Q}(x, 1) = \sum_{t=1}^{K} \hat{b}_1^{(t)}(x)$$

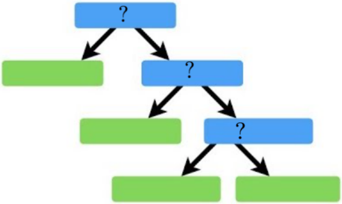
How do we fit a regression tree?



- Decide optimal leaf weights: for a fixed tree structure T, let $I_j = \{i | q(X_i) = j\}$ be the instance set of leaf j. Then

$$w_j^* = \frac{2 \sum_{i \in I_j} (Y_i - \hat{Y}_i^{(t-1)})}{2|I_j| + \lambda}$$

How do we fit a regression tree?



- Decide optimal leaf weights: for a fixed tree structure T, let $I_j = \{i | q(X_i) = j\}$ be the instance set of leaf j. Then

$$w_j^* = \frac{2 \sum_{i \in I_j} (Y_i - \hat{Y}_i^{(t-1)})}{2|I_j| + \lambda}$$

- Split finding algorithm for estimating tree structure T: Chen and Guestrin, 2016

Summary

Algorithm

Input: dataset $\{(X_i, Y_i, A_i)\}_{i=1}^n$, number of iterations K, learning rate η, maximum of tree depth d

(1) Train bst.plus1 = XGBoost($\{(X_i, Y_i); A_i = 1\}, K, \eta, d$)

(2) Train bst.minus1 = XGBoost($\{(X_i, Y_i); A_i = -1\}, K, \eta, d$)

(3) Output the estimated optimal ITR:

$$\widehat{\mathcal{D}}(x) = \mathrm{sign}(\text{bst.plus1}(x) - \text{bst.minus1}(x))$$

Our second proposed method

- One instance of direct learning
- Key ideas:
  - Assume $f^*(x) = \sum_{t=1}^{K} b^{(t)}(x)$ where $b^{(t)}$ are regression trees
  - Use boosting algorithm to estimate $b^{(t)}$ sequentially

Our second proposed method

- One instance of direct learning
- Key ideas:
  - Assume $f^*(x) = \sum_{t=1}^{K} b^{(t)}(x)$ where $b^{(t)}$ are regression trees
  - Use boosting algorithm to estimate $b^{(t)}$ sequentially

Fisher consistency theorem

Assume $Y = \mu(X) + \delta(X) \times A + \varepsilon$. Let $\pi_A(X) = P(A|X)$. Then we have

$$\mu = \underset{g}{\operatorname{argmin}} \; \mathbb{E}\left\{\frac{1}{\pi_A(X)}(Y - g(X))^2\right\}.$$

Furthermore,

$$f^* = \underset{f}{\operatorname{argmin}} \; \mathbb{E}\left\{\underbrace{\frac{|Y-\mu(X)|}{\pi_A(X)}}_{\text{weight}} \phi\left(\overbrace{\underbrace{A \times \operatorname{sign}(Y-\mu(X))}_{\text{adjusted label}} f(X)}^{\text{functional margin}}\right)\right\},$$

where $\phi(x) = \log(1 + e^{-2x})$.

Before XGBoost

Estimation of $\mu(x)$

(1) Assume $\mu(x) = \alpha_0 + \alpha^T x$

(2) Estimate α_0 and α:

$$\mu = \underset{g}{\operatorname{argmin}} \quad E\left\{\frac{1}{\pi_A(X)}(Y - g(X))^2\right\}$$

$$\hat{\alpha}_0, \hat{\alpha} = \underset{\alpha_0,\alpha}{\operatorname{argmin}} \sum_{i=1}^{n} \frac{1}{\pi_{A_i}(X_i)} \left(Y_i - \alpha_0 - \alpha^T X_i\right)^2$$

(3) Estimate $\mu(x)$:

$$\hat{\mu}(x) = \hat{\alpha}_0 + \hat{\alpha}^T x$$

XGBoost algorithm

- 1st iteration:

Estimation of $b^{(1)}$

(1) Fit a tree to the training data (X_i, A_i, Y_i):

$$f^* = \underset{f}{\operatorname{argmin}} \quad \mathbb{E}\left\{\frac{|Y - \mu(X)|}{\pi_A(X)}\phi\left(A \times \mathrm{sign}(Y - \mu(X))f(X)\right)\right\}$$

$$\hat{f}^{(1)} = \underset{f}{\operatorname{argmin}} \sum_{i=1}^{n} \frac{|Y_i - \hat{\mu}(X_i)|}{\pi_{A_i}(X_i)} \underbrace{\phi\left(A_i f(X_i) \times \mathrm{sign}\left(Y_i - \hat{\mu}(X_i)\right)\right)}_{\text{Use second-order approximation}} + J(f)$$

(2) Shrinkage: $\hat{b}_1^{(1)} = \eta \hat{f}^{(1)}$, where $0 < \eta < 1$

XGBoost algorithm

- tth iteration:

Estimation of $b^{(t)}$

(1) Fit a tree to the training data (X_i, A_i, Y_i):

$$\hat{f}^{(t)} = \underset{f}{\operatorname{argmin}} \sum_{i=1}^{n} \frac{|Y_i - \hat{\mu}(X_i)|}{\pi_{A_i}(X_i)} \underbrace{\phi\Big(A_i\Big(\hat{Y}_i^{(t-1)} + f(X_i)\Big) \times \operatorname{sign}(Y_i - \hat{\mu}(X_i))\Big)}_{\text{Use second-order approximation}} + J(f)$$

(2) Shrinkage: $\hat{b}^{(t)} = \eta \hat{f}^{(t)}$

Summary

Algorithm

Input: dataset $\{(X_i, A_i, Y_i)\}_{i=1}^n$, number of iterations K, shrinkage parameter η and maximum tree depth d.

(1) Estimate the common effect μ.

(2) Train bst = XGBoost($\{X_i, A_i \mathrm{sign}(Y_i - \hat{\mu}(X_i))\}, K, \eta, d$) with weighted deviance loss

(3) Output the estimated optimal ITR:

$$\widehat{\mathcal{D}}(x) = \mathrm{sign}\left(\mathrm{bst}(x)\right)$$

Simulation settings

- $X_i \in \mathbb{R}^{10}$: each component is i.i.d. $U(-1,1)$
- A_i: $P(A_i = -1) = P(A_i = 1) = 0.5$
- $\varepsilon_i \sim N(0,1)$
- $Y_i = 1 + 2X_{1i} + X_{2i} + 0.5X_{3i} + \delta(X_i) \times A_i + \varepsilon_i$, where X_{1i}, X_{2i} and X_{3i} are the first, second and third components of X_i, and

$$\delta(X_i) = 0.2 + X_{1i}^2 + X_{2i}^2 - X_{3i}^2 - X_{4i}^2$$

- Polynomial-type optimal ITR:

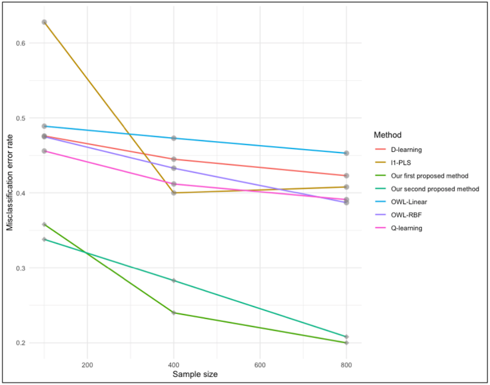
$$\mathcal{D}^*(X) = \begin{cases} 1 & 0.2 + X_1^2 + X_2^2 - X_3^2 - X_4^2 > 0 \\ -1 & 0.2 + X_1^2 + X_2^2 - X_3^2 - X_4^2 < 0 \end{cases}$$

Simulation results

- Misclassification rate of an ITR: $\frac{1}{n}\sum_{i=1}^{n} I(\mathcal{D}^*(X_i) \neq \mathcal{D}(X_i))$ for a testing dataset $\{(X_i, A_i, Y_i), 1 \leq i \leq n\}$

Simulation results

- Misclassification rate of an ITR: $\frac{1}{n}\sum_{i=1}^{n} I(\mathcal{D}^*(X_i) \neq \mathcal{D}(X_i))$ for a testing dataset $\{(X_i, A_i, Y_i), 1 \leq i \leq n\}$



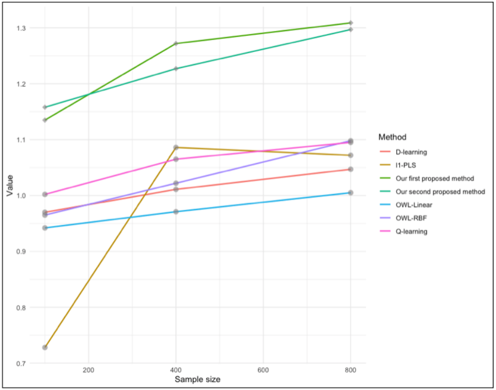
Linear competitors: D-learning (Qi et al., 2019), ℓ_1-PLS (Qian and Murphy, 2011), OWL-Linear (Zhao et al., 2012), Q-learning (benchmark); Nonlinear competitor: OWL-RBF (Zhao et al., 2012)

Simulation results

- Value function of an ITR: $V(\mathcal{D}) = \mathbb{E}^{\mathcal{D}}(Y) = \mathbb{E}\left\{ Y \frac{I(A=\mathcal{D}(X))}{\pi_A(X)} \right\}$
- Estimated value function of an ITR: $\widehat{V}(\mathcal{D}) = \frac{\frac{1}{n}\sum_{i=1}^{n} \frac{Y_i}{\pi_{A_i}(X_i)} I(\mathcal{D}(X_i)=A_i)}{\frac{1}{n}\sum_{i=1}^{n} \frac{I(\mathcal{D}(X_i)=A_i)}{\pi_{A_i}(X_i)}}$

Simulation results

- Value function of an ITR: $V(\mathcal{D}) = \mathbb{E}^{\mathcal{D}}(Y) = \mathbb{E}\left\{Y \frac{I(A=\mathcal{D}(X))}{\pi_A(X)}\right\}$
- Estimated value function of an ITR: $\widehat{V}(\mathcal{D}) = \frac{\frac{1}{n}\sum_{i=1}^{n} \frac{Y_i}{\pi_{A_i}(X_i)} I(\mathcal{D}(X_i)=A_i)}{\frac{1}{n}\sum_{i=1}^{n} \frac{I(\mathcal{D}(X_i)=A_i)}{\pi_{A_i}(X_i)}}$



Diabetes data analysis

- The dataset was collected from a randomized, double-blind, parallel-group Phase III trial (Charbonnel and Matthews et al., 2005)
- $\mathcal{A} = \{\text{gliclazide}, \text{pioglitazone}\}$
- Among 1247 patients, 624 patients received gliclazide and 623 received pioglitazone
- X: 21 pretreatment covariates, e.g., BMI and blood pressure
- Y: primary efficacy endpoint, i.e., change of HbA1c level during 52 weeks

Diabetes data analysis

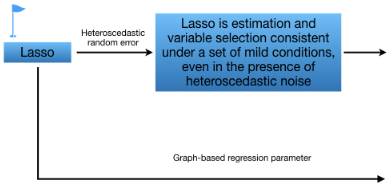
- The dataset was collected from a randomized, double-blind, parallel-group Phase III trial (Charbonnel and Matthews et al., 2005)
- $\mathcal{A} = \{\text{gliclazide}, \text{pioglitazone}\}$
- Among 1247 patients, 624 patients received gliclazide and 623 received pioglitazone
- X: 21 pretreatment covariates, e.g., BMI and blood pressure
- Y: primary efficacy endpoint, i.e., change of HbA1c level during 52 weeks
- Results:

Method	Our first proposed method	Our second proposed method	Q-learning	l1-PLS	D-learning	OWL-Linear	OWL-RBF
Estimated value	1.447	1.448	1.369	1.428	1.416	1.360	1.363

Summary

Efficient statistical learning of complex data

- High-dimensional linear models



One-step estimator

Input: dataset $\{(X_i, y_i)\}_{i=1}^N$, formula of g, tuning parameters λ and μ.

(1) Solve the Lasso to obtain a preliminary estimator $\hat{\beta}$.

(2) Set $\widehat{W}_i = g(X_i, \hat{\beta})$, $\widehat{W} = \mathrm{diag}(\widehat{W}_1, \ldots, \widehat{W}_N)$, and $\widehat{\Sigma}_N = \frac{1}{N} X^T \widehat{W}^{-1} X$.

(3) Solve the following optimization problem to obtain $\widehat{\Theta}$:

$$\widehat{\Theta} = \operatorname*{argmin}_{\Theta \in \mathbb{R}^{n \times n}} \quad |\Theta|_{1,1} = \sum_{i=1}^n \sum_{j=1}^n |\Theta_{ij}|$$
$$\text{subject to} \quad |\Theta \widehat{\Sigma}_N - I_n|_\infty \le \mu.$$

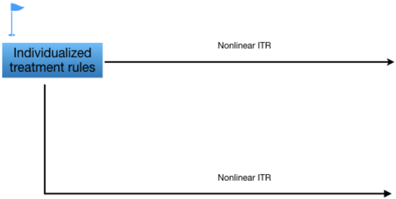
(4) Output the final estimator:

$$\tilde{\beta} = \hat{\beta} + \frac{1}{N} \widehat{\Theta} X^T \widehat{W}^{-1} (y - X\hat{\beta}).$$

k-th order Graph-Piecewise-Polynomial-Lasso

$$\hat{\beta}_g = \operatorname*{argmin}_{\beta \in \mathbb{R}^n} \frac{1}{2N} \|y - X\beta\|_2^2 + \lambda \|\beta\|_1 + \lambda_g \|\Delta^{(k+1)} \beta\|_1$$

- Beyond linear models



Boosting algorithm 1

Input: dataset $\{(X_i, Y_i, A_i)\}_{i=1}^n$, number of iterations K, learning rate η, maximum of tree depth d

(1) Train bst.plus1 = XGBoost($\{(X_i, Y_i); A_i = 1\}, K, \eta, d$)

(2) Train bst.minus1 = XGBoost($\{(X_i, Y_i); A_i = -1\}, K, \eta, d$)

(3) Output the estimated optimal ITR:

$$\widehat{\mathcal{D}}(x) = \mathrm{sign}(\mathrm{bst.plus1}(x) - \mathrm{bst.minus1}(x))$$

Boosting algorithm 2

Input: dataset $\{(X_i, A_i, Y_i)\}_{i=1}^n$, number of iterations K, shrinkage parameter η and maximum tree depth d.

(1) Estimate the common effect μ.

(2) Train bst = XGBoost($\{X_i, A_i \mathrm{sign}(Y_i - \hat{\mu}(X_i))\}, K, \eta, d$) with weighted deviance loss

(3) Output the estimated optimal ITR:

$$\widehat{\mathcal{D}}(x) = \mathrm{sign}(\mathrm{bst}(x))$$

Takeaway

- Our work has revealed the importance of designing efficient statistical learning methods which adapt to the unique features of complex data
- This thesis leaves the door open for a more extensive investigation of efficient statistical learning for complex data
  - Other interesting types of complex data, e.g., high-frequency financial data, large network data
  - Unsupervised learning
- Our hope: develop more approaches which not only ensure statistical and computational soundness, but also provide easy-to-use, accessible software

Takeaway

- Our work has revealed the importance of designing efficient statistical learning methods which adapt to the unique features of complex data
- This thesis leaves the door open for a more extensive investigation of efficient statistical learning for complex data
  - Other interesting types of complex data, e.g., high-frequency financial data, large network data
  - Unsupervised learning
- Our hope: develop more approaches which not only ensure statistical and computational soundness, but also provide easy-to-use, accessible software

Thank you!