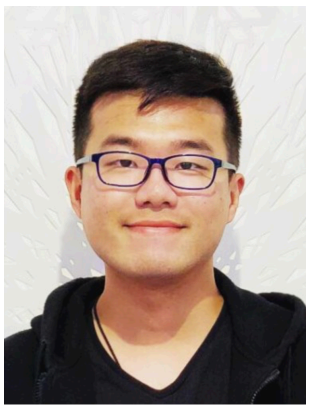
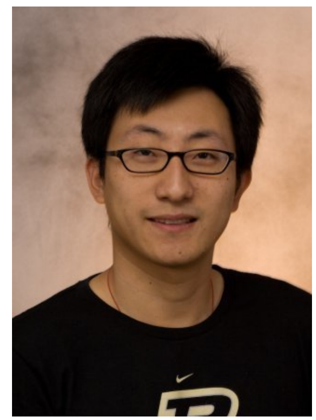
Distributed Bootstrap for Simultaneous Inference Under High Dimensionality





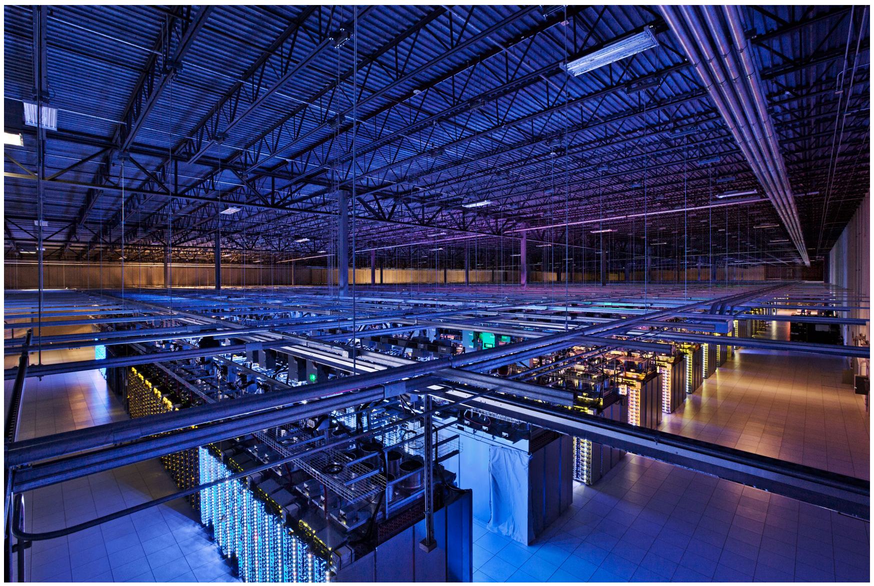
Yang Yu
Purdue University &
Facebook

Shih-Kang Chao
University of Missouri

Guang Cheng
Purdue University

github.com/skchao74/Distributed-bootstrap

- Data growing **faster** than processing speeds
- Only solution is to parallelize on large clusters
- Wide use in both enterprises and web industry

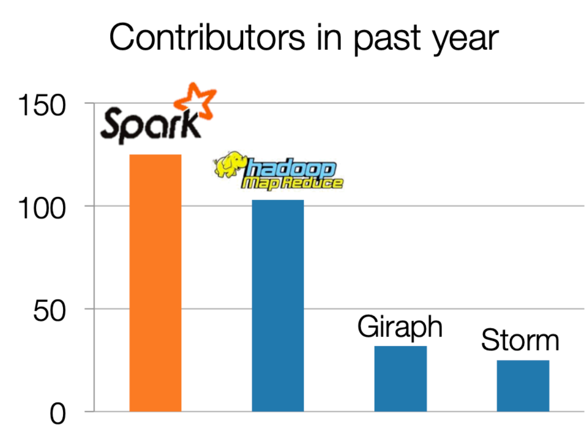


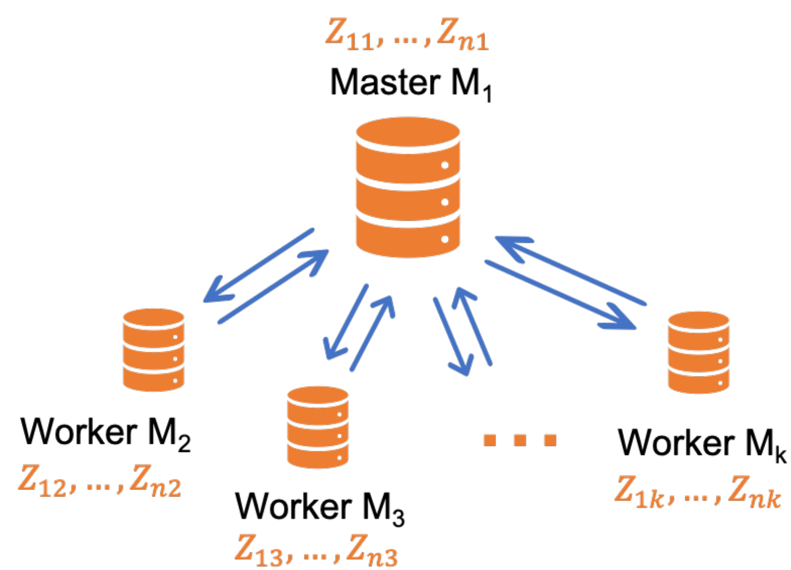
This slide is taken from CME 323 “Distributed Algorithms and Optimization” at Stanford, 2020 Spring

Computational Frameworks

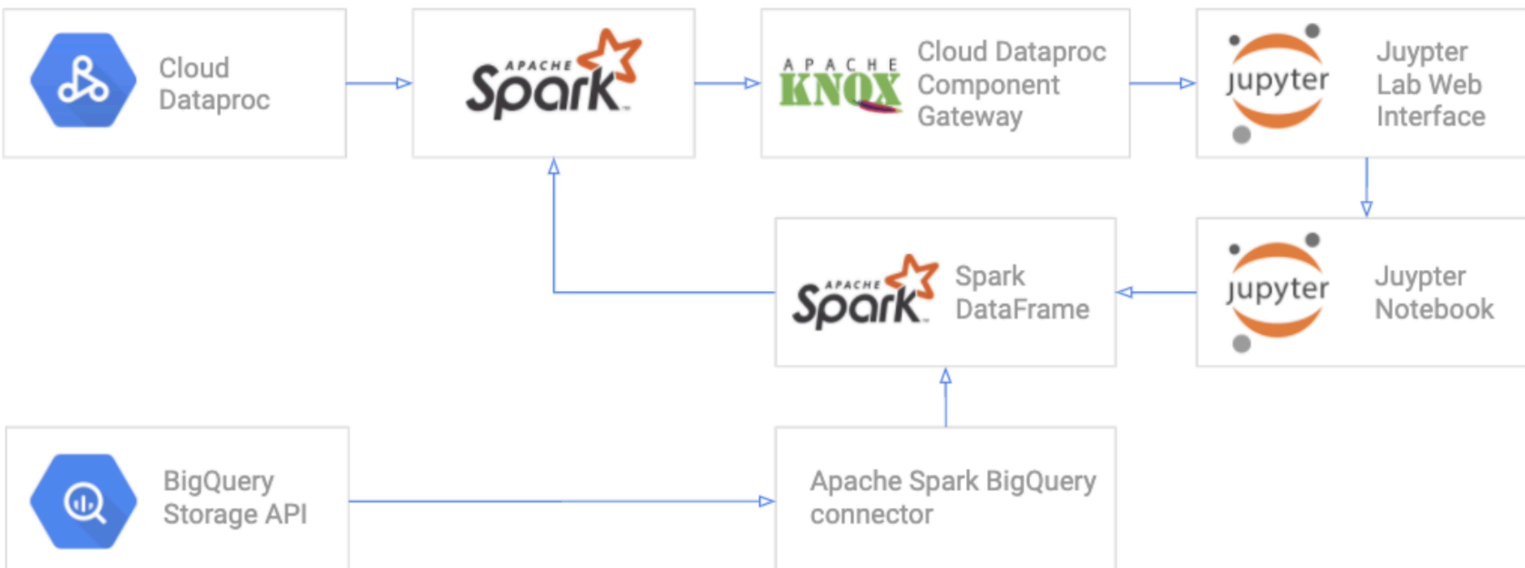
- **Spark** is the most popular, others: Hadoop, Storm, Flink …
- Clean API in JAVA, Scala, Python and R
- Built in cloud by service providers, e.g. Google and Amazon

200+ developers, 50+ companies contributing





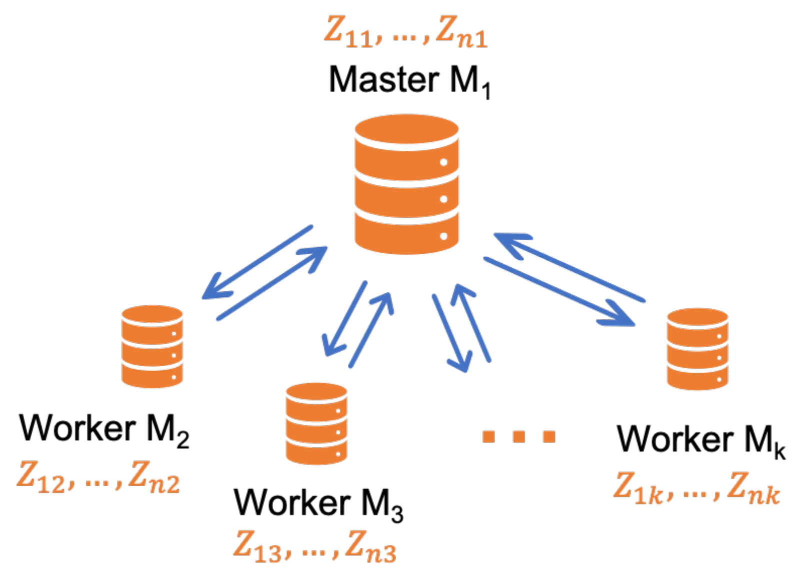
- Google connects it with Bigquery API
- Scalability solution for enterprise users
- Data scientists only need to focus on business insights, not on hardware architecture



Flowchart: “Apache Spark BigQuery Connector — Optimization tips & example Jupyter Notebooks”. 2020 May, Medium

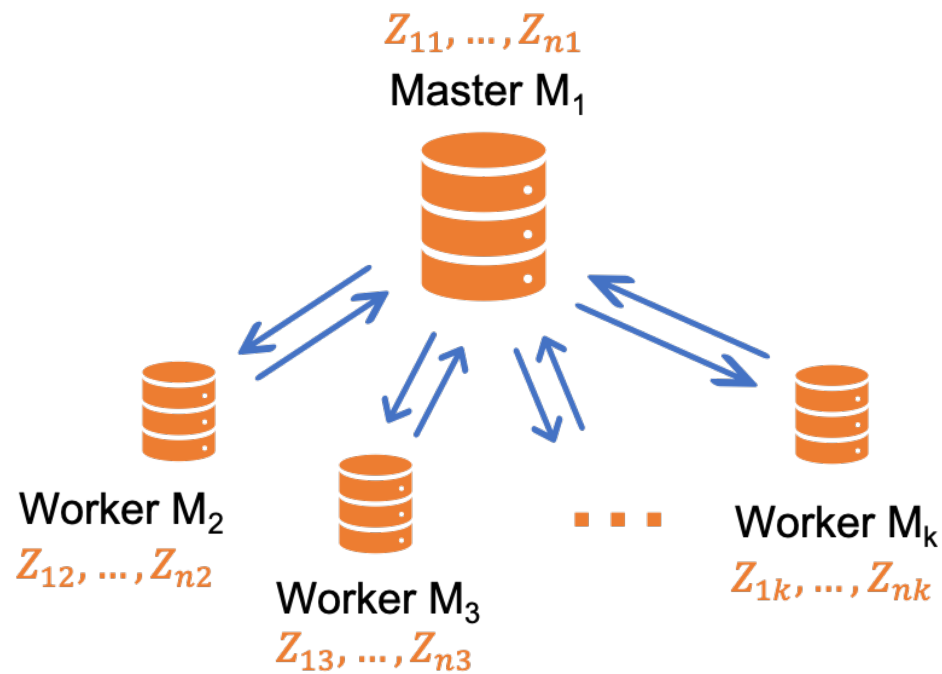
Communication cost: a fundamental issue

- A fixed communication cost is incurred every time the master communicates with workers
- The cost depends on
  - Bandwidth
  - Message size
  - Latency of each machine
    - synchronization barrier
  - Number of workers k



Challenges for Statistical Methodology

- Computation involving entire data typically requires at least one communication
- Inference like MCMC or bootstrap typically requires hundreds or thousands of communications
- How to maximize the parallelism while preserving statistical accuracy?
- **High dimensional statistical inference?**



Contributions

We propose a **distributed bootstrap inference……**

- Theoretically valid for HD sparse GLMs
  - Utilize ℓ_1 penalty to enforce sparsity
  - **Lower bound on the number of the communication rounds between master and workers that warrants statistical accuracy**
- Proposed a new and efficient distributed cross-validation for tuning
- Validated with simulation and real dataset

Distributed loss

Global loss:

$$\mathscr{L}_N(\theta) = \frac{1}{k}\sum_{j=1}^{k}\mathscr{L}_j(\theta)$$

Local loss:

$$\mathscr{L}_j(\theta) = \frac{1}{n}\sum_{i=1}^{n}\mathscr{L}(\theta; Z_{ij}), j = 1, \cdots, k$$

- Z_{ij} : data i in j worker node, i.i.d. in i, j
- $\mathscr{L}$: twice differentiable w.r.t. $\theta \in \mathbb{R}^d$, $d > n$
- True parameter:

$$\theta^* = \arg\min_{\theta} \mathscr{L}^*(\theta), \text{ where } \mathscr{L}^*(\theta) = \mathrm{E}_Z[\mathscr{L}(\theta; Z)]$$

θ^* is a sparse vector, $\|\theta^*\|_0 = s^*$

Simultaneous confidence set

- Testing high dimensional unknown $H_0 : \theta^* = \theta_0$
- Union of individual confidence intervals would lead to too many rejections, i.e. can't control the family-wise error rate (FWER: the probability of falsely rejecting at least one hypothesis under H_0)
- Instead, we consider simultaneous testing, i.e. we reject whenever $\sqrt{N}(\hat{\theta}_l - \theta_{0,l}) > c(\alpha)$, where $c(\alpha)$ is the **quantile** of $\widehat{T}$:

$$c(\alpha) := \inf\{t \in \mathbb{R} : P(\widehat{T} \leq t) \geq \alpha\}$$

$$\widehat{T} = \|\sqrt{N}(\hat{\theta} - \theta_0)\|_\infty$$

Review: Non-distributed, small data setting

$c(\alpha) := \inf\{t \in \mathbb{R} : P(\widehat{T} \leq t) \geq \alpha\}$ $\quad \widehat{T} = \|\sqrt{N}(\hat{\theta} - \theta_0)\|_\infty$

- The de-biased Lasso:

$$\hat{\theta} = \hat{\theta}_{Lasso} - \hat{\Theta} \nabla \mathscr{L}_N(\hat{\theta}_{Lasso})$$

where $\hat{\theta}_{Lasso}$ is the Lasso estimator, $\hat{\Theta}$ is a surrogate for inverse Hessian

- $\hat{\theta}$ is $\sqrt{N}$ asymptotic Gaussian (many papers, e.g. van de Geer et al. 2014, Zhang and Zhang 2014, Javanmard & Montanari 2014…)
- Bootstrap estimator $\hat{c}(\alpha)$ has been studied by many, e.g. Zhang and Cheng (2017)
- How to compute $\hat{\theta}_{Lasso}$, $\hat{\Theta}$ and perform bootstrap distributedly?

Distributed de-biased Lasso

Algorithm 1 `k-grad/n+k-1-grad` with de-biased ℓ_1-CSL estimator

$\widetilde{\theta}^{(0)} \leftarrow \arg\min_\theta \mathcal{L}_1(\theta) + \lambda^{(0)}\|\theta\|_1$ at $\mathcal{M}_1$ # Initial estimator: obtained by data in master only

Compute $\widetilde{\Theta}$ by running $\texttt{Node}(\nabla^2 \mathcal{L}_1(\widetilde{\theta}^{(0)}), \{\lambda_l\}_{l=1}^d)$ at $\mathcal{M}_1$ # surrogate Hessian: only use the master node to compute

for $t = 1, \ldots, \tau$ **do** # τ: number of communication rounds

  $\nabla \mathcal{L}_N(\widetilde{\theta}^{(t-1)}) \leftarrow k^{-1} \sum_{j=1}^k \nabla \mathcal{L}_j(\widetilde{\theta}^{(t-1)})$ at $\mathcal{M}_1$

  if $t < \tau$ **then**

    $\widetilde{\theta}^{(t)} \leftarrow \arg\min_\theta \mathcal{L}_1(\theta) - \theta^\top \left(\nabla \mathcal{L}_1(\widetilde{\theta}^{(t-1)}) - \nabla \mathcal{L}_N(\widetilde{\theta}^{(t-1)})\right) + \lambda^{(t)}\|\theta\|_1$ at $\mathcal{M}_1$

    # Iteratively improving the estimator using Communication-efficient surrogate learning (CSL, Jordan et al. 2019, Wang et al. 2017)

  else

    $\widetilde{\theta}^{(\tau)} \leftarrow \widetilde{\theta}^{(\tau-1)} - \widetilde{\Theta} \nabla \mathcal{L}_N(\widetilde{\theta}^{(\tau-1)})$ at $\mathcal{M}_1$ # de-biased step

  end if

end for

Run $\texttt{DistBoots}$(‘`k-grad`’ or ‘`n+k-1-grad`’, $\widetilde{\theta} = \widetilde{\theta}^{(\tau)}, \{\mathbf{g}_j = \nabla \mathcal{L}_j(\widetilde{\theta}^{(\tau-1)})\}_{j=1}^k$,

$\widetilde{\Theta} = \widetilde{\Theta}$) at $\mathcal{M}_1$

Multiplier bootstrap: classical

- Need to bootstrap the distribution of $\widehat{T} = \|\sqrt{N}(\hat{\theta} - \theta^*)\|_\infty$
- If τ is sufficiently large, $\hat{\theta}^{(\tau-1)} \approx \theta_0$, and the de-biased $\hat{\theta}^{(\tau)}$ satisfies

$$\sqrt{N}(\hat{\theta}^{(\tau)} - \theta^*) \approx -\frac{1}{\sqrt{k}}\tilde{\Theta}\sum_{j=1}^{k}\sqrt{n}\,\nabla\mathscr{L}_j(\theta^*; Z_{ij})$$

where the gradients are centered

- Classical multiplier bootstrap of $\widehat{T}$: $\varepsilon_{ij}^{(b)}$ i.i.d. $\mathcal{N}(0,1)$,

$$\widehat{T}^{(b)} = \left\| \frac{1}{\sqrt{k}}\tilde{\Theta}\sum_{j=1}^{k}\sqrt{n}\sum_{i=1}^{n}\varepsilon_{ij}^{(b)}\left(\hat{g}_{ij} - \bar{g}\right)\right\|_\infty$$

$\hat{g}_{ij} = \nabla\mathscr{L}(\hat{\theta}; Z_{ij})$, $\bar{g}$ = average($\hat{g}_{ij}$)

Distributed bootstrap: k-grad

- Classical multiplier bootstrap requires **many communications** — the same number as the bootstrap samples (in hundreds)
- For remedy, in YCC (2020, ICML), we proposed the k-grad bootstrap:

$$\overline{W}^{(b)} = \left\| \frac{1}{\sqrt{k}} \tilde{\Theta} \sum_{j=1}^{k} \varepsilon_j^{(b)} \sqrt{n} \underbrace{\left(g_j^{(\tau-1)} - \bar{g}^{(\tau-1)}\right)}_{\text{de-mean}} \right\|$$

where $g_j^{(\tau-1)} = \sum_{i=1}^{n} \nabla \mathscr{L}(\theta^{(\tau-1)}; Z_{ij})$, and $\bar{g}^{(\tau-1)} = \mathrm{avg}(g_j^{(\tau-1)})$

- Simulate k-grad samples $\{\overline{W}^{(1)}, \cdots, \overline{W}^{(B)}\}$, set $\hat{c}(\alpha)$ to be its empirical $1 - \alpha$ quantile

Distributed bootstrap: $n + k - 1$ grad

- The k-grad is inaccurate when k is too small due to degenerate variance (like sample variance is inaccurate when n is small!)
- For remedy, we propose the $n + k - 1$-grad bootstrap:

$$\widetilde{W}^{(b)} = \left\| \frac{1}{\sqrt{k}} \tilde{\Theta} \left(\sum_{i=1}^{n} \varepsilon_{1i}^{(b)} \left(g_{i1}^{(\tau-1)} - \bar{g}^{(\tau-1)} \right) + \sum_{j=2}^{k} \varepsilon_{j}^{(b)} \sqrt{n} \left(g_{j}^{(\tau-1)} - \bar{g}^{(\tau-1)} \right) \right) \right\|$$

- Set $\hat{c}(\alpha)$ by the $1 - \alpha$ quantile of samples $\{ \widetilde{W}^{(1)}, \cdots, \widetilde{W}^{(B)} \}$

Cross-validation for model tuning

- Classical CV is very computationally demanding

Algorithm 2 Distributed K-fold cross-validation for t-step CSL

Require: $(t-1)$-step CSL estimate $\widetilde{\theta}^{(t-1)}$, set Λ of candidate values for $\lambda^{(t)}$, partition of master data $\mathcal{Z} = \bigcup_{q=1}^{K} \mathcal{Z}_q$, partition of worker gradients $\mathcal{G} = \bigcup_{q=1}^{K} \mathcal{G}_q$ # partition data into K shares

for $q = 1, \ldots, K$ **do**

  $\mathcal{Z}_{train} \leftarrow \bigcup_{r \neq q} \mathcal{Z}_r$; $\mathcal{Z}_{test} \leftarrow \mathcal{Z}_q$ # K-1 shares as train, 1 share as test

  $\mathcal{G}_{train} \leftarrow \bigcup_{r \neq q} \mathcal{G}_r$; $\mathcal{G}_{test} \leftarrow \mathcal{G}_q$

  $g_{1,train} \leftarrow \mathrm{Avg}_{Z \in \mathcal{Z}_{train}} \left(\nabla \mathcal{L}(\widetilde{\theta}^{(t-1)}; Z) \right)$; $g_{1,test} \leftarrow \mathrm{Avg}_{Z \in \mathcal{Z}_{test}} \left(\nabla \mathcal{L}(\widetilde{\theta}^{(t-1)}; Z) \right)$ # master node gradients

  $\bar{g}_{train} \leftarrow \mathrm{Avg}_{g \in \{g_{1,train}\} \cup \mathcal{G}_{train}} (g)$; $\bar{g}_{test} \leftarrow \mathrm{Avg}_{g \in \{g_{1,test}\} \cup \mathcal{G}_{test}} (g)$ # worder node gradients

  for $\lambda \in \Lambda_t$ **do**

    $\beta \leftarrow \arg\min_\theta \mathrm{Avg}_{Z \in \mathcal{Z}_{train}} (\mathcal{L}(\theta; Z)) - \theta^\top (g_{1,train} - \bar{g}_{train}) + \lambda \|\theta\|_1$ # gradient corrections

    $Loss(\lambda, q) \leftarrow \mathrm{Avg}_{Z \in \mathcal{Z}_{test}} (\mathcal{L}(\beta; Z)) - \beta^\top (g_{1,test} - \bar{g}_{test})$ followed by CSL

  end for

end for

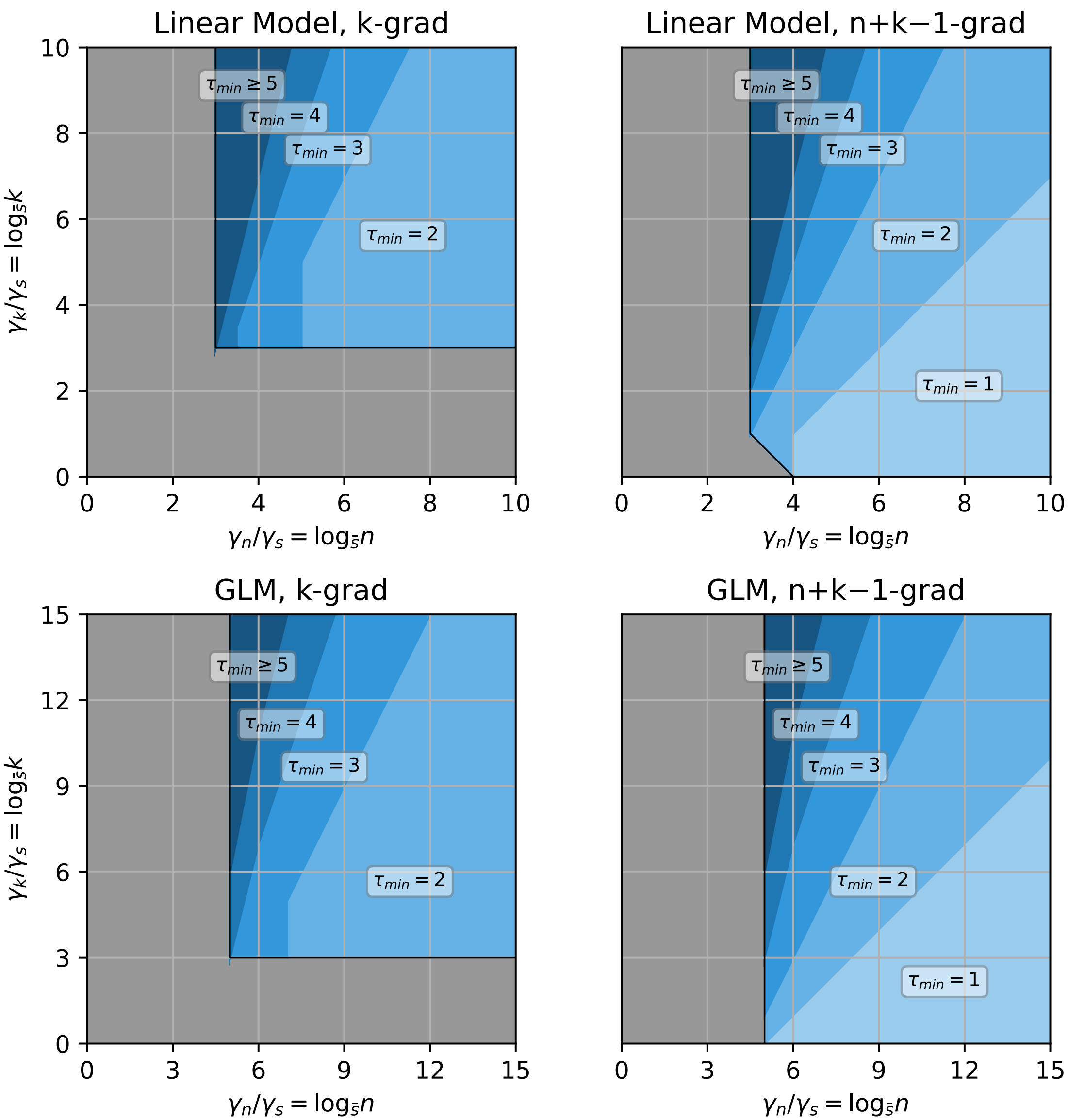
Return $\lambda^{(t)} = \arg\min_{\lambda \in \Lambda} K^{-1} \sum_{q=1}^{K} Loss(\lambda, q)$ # the test loss used for selecting lambda

Theoretical guarantees

- Goal: accurately control the FWER, i.e. under the null

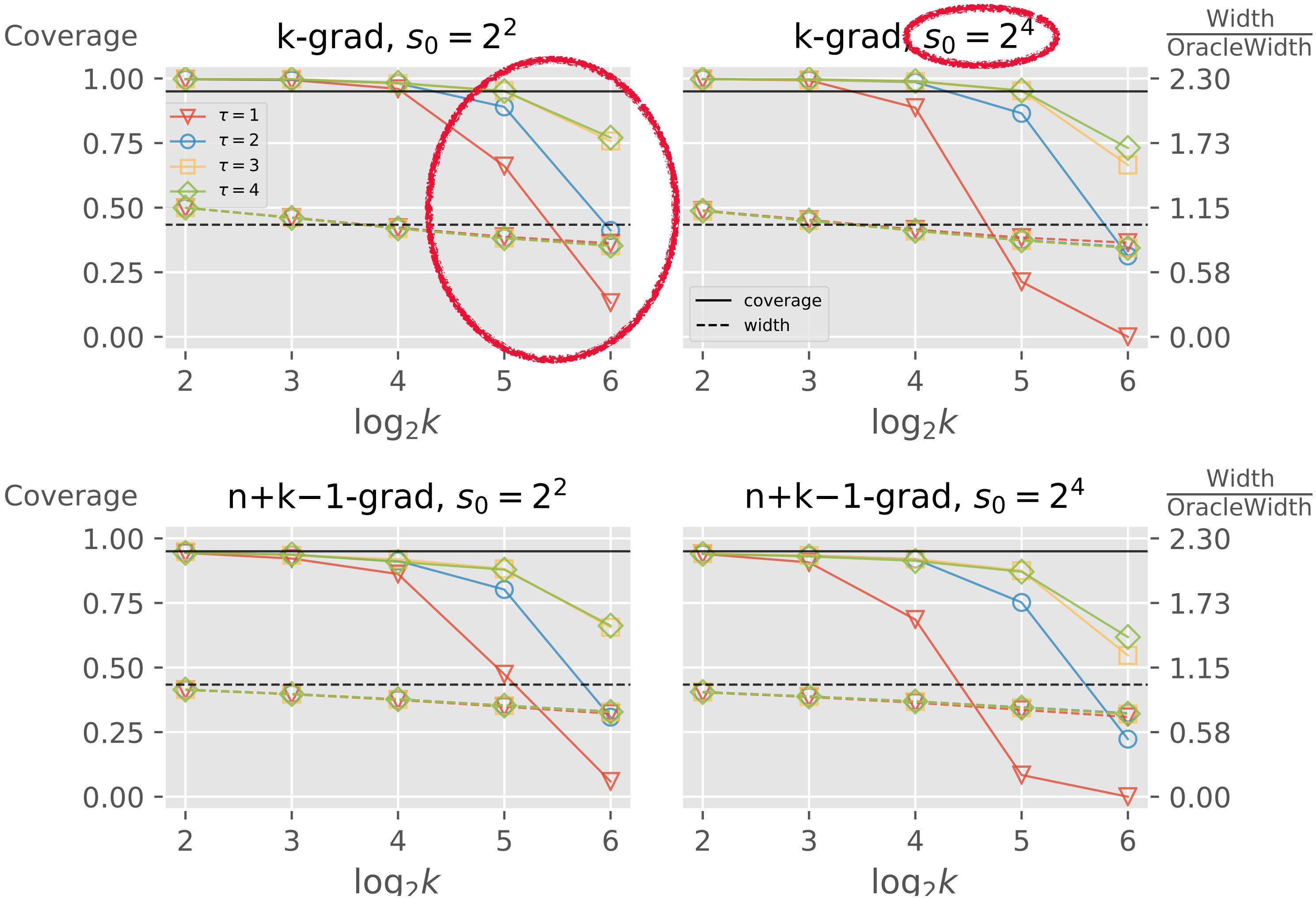
$$\sup_{\alpha \in (0,1)} \left| P(T \geq \hat{c}(\alpha)) - \alpha \right| \to 0, \text{ as } d, N \to \infty$$

- What is the minimal number of communication rounds τ_{min} for this?
- Critically depend on the interplay between
  - number of workers: k
  - max sparsity level $\bar{s}$ of θ and inverse Hessian (but not the nominal dimensionality d !)
- We will obtain guarantees for **least square** and **generalized linear models,** e.g. logistic regression



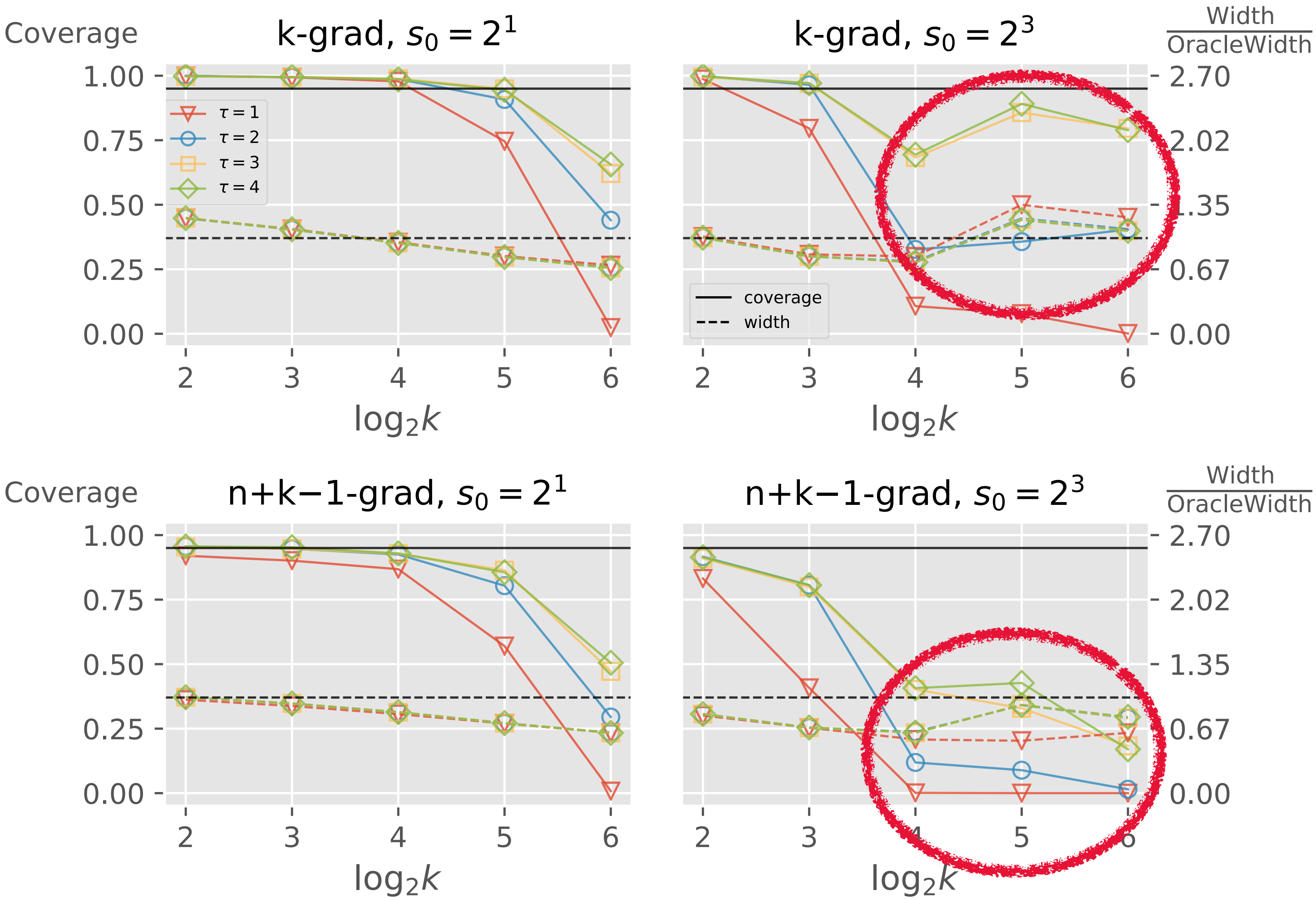
- Greater local sample size n requires less $\tau_{\min}$
- Greater number of workers k needs greater $\tau_{\min}$
- Higher $\bar{s}$ also requires a higher τ_{min}
- More complicated model like GLM requires a greater $\tau_{\min}$
- $n + k - 1$-grad requires a smaller $\tau_{\min}$

Simulation: coverage = 1-FWER, LM, Toeplitz cov



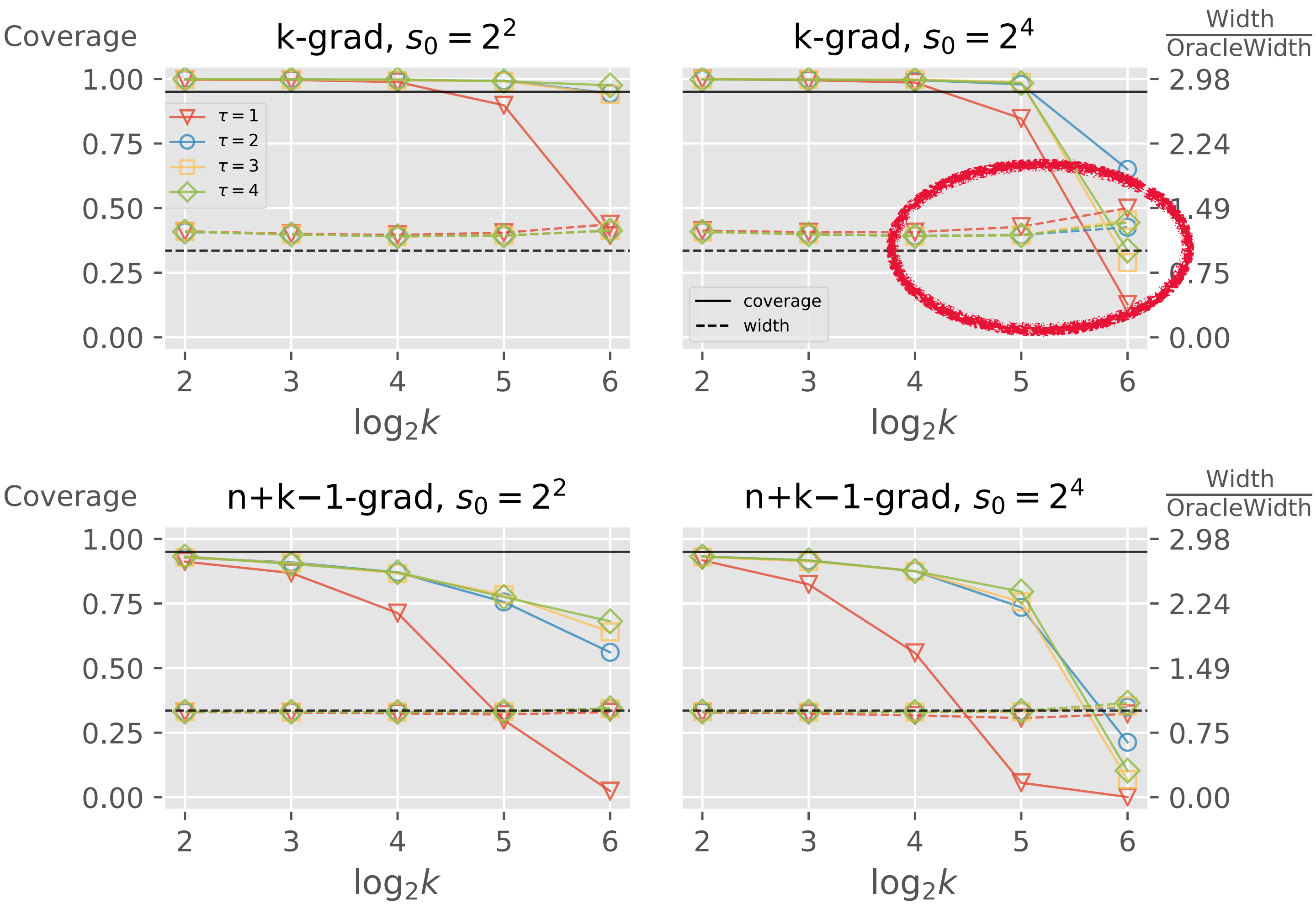
- k-grad slightly over covers; $n + k - 1$ grad is more accurate
- Efficiency: $\hat{c}(\alpha)$ is close to $c(\alpha)$, the true quantile
- larger k reduces performance, but a greater τ helps

Simulation: coverage = 1-FWER, GLM, Toeplitz cov



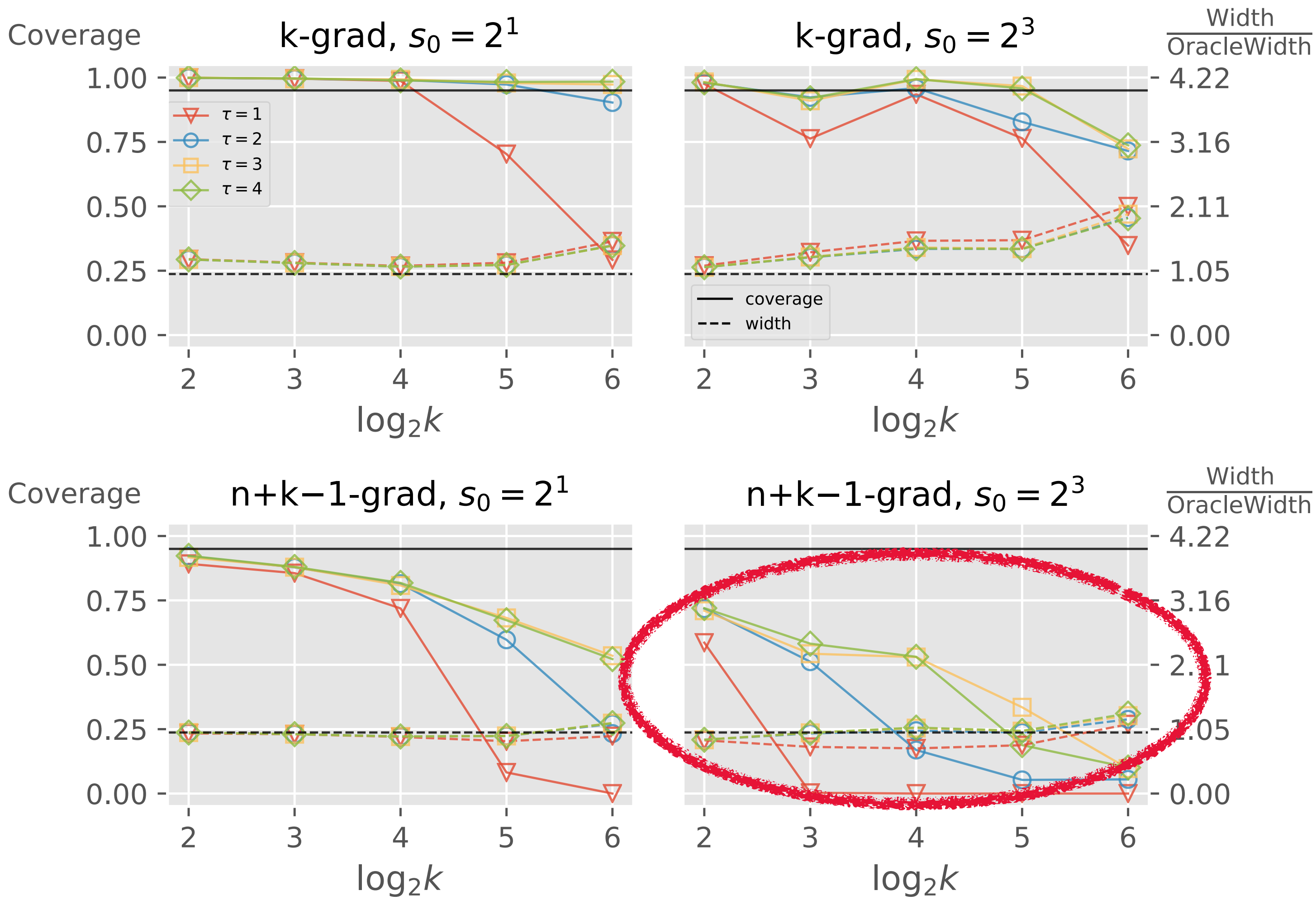
- Similar patterns as the LM, but more variations when model is less sparse

Simulation: coverage = 1-FWER, LM, constant corr



- Similar patterns as the Toeplitz design. The only notable difference is $\hat{c}(\alpha)$ is greater for larger k

Simulation: coverage = 1-FWER, GLM, constant corr

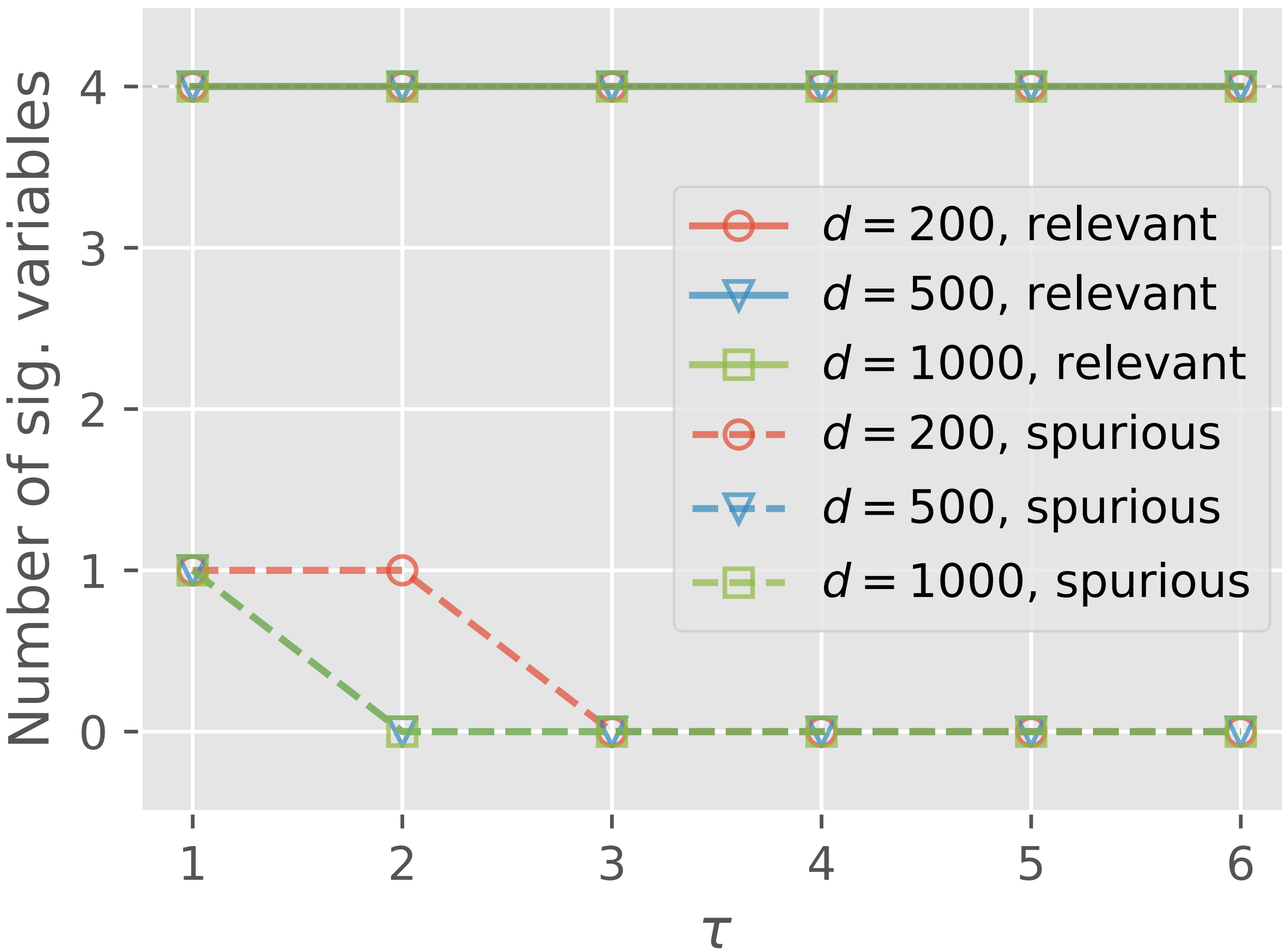


- When the model is not sparse, $n + k - 1$-grad need more iterations, i.e. a higher τ

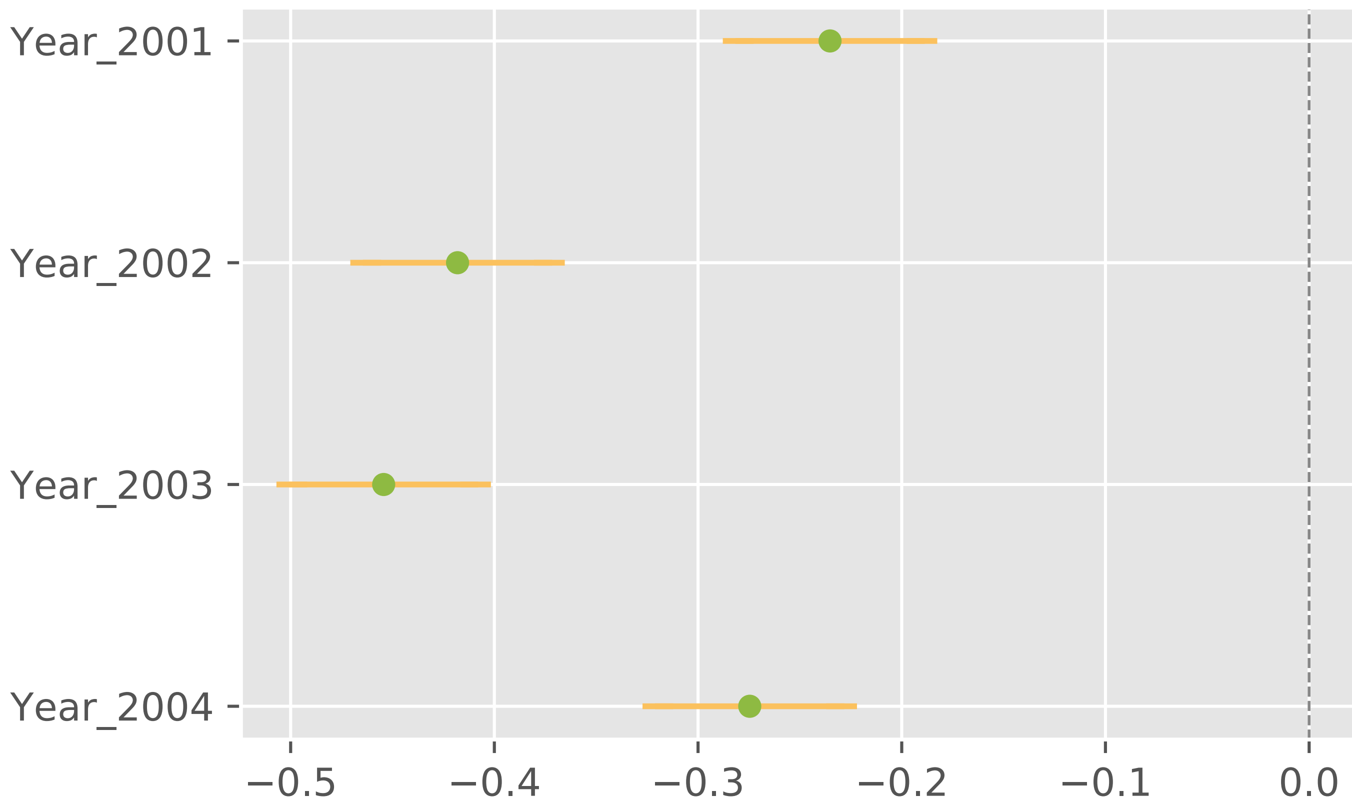
Semi-synthetic data

- US Airline On-time Performance data: public, 1987-2008
- Response: flight delay time (binary)
- Predictors: mostly categorical - year, month, DayOfWeek, DepTime, ArrTime, Carrier, Origin, Destination
- $N =$ 113.9 million, 230 predictors after making dummy variables
- Pre-select 4 predictors of significance by t test, and 1 intercept
- Form a new design matrix: synthesize $d - 5$ fake $\mathcal{N}(0, \text{Toeplize}_{d-5})$ predictors, and combine them with the 5 real predictors

Variable screening: can our method correctly identify the 5 real predictors?

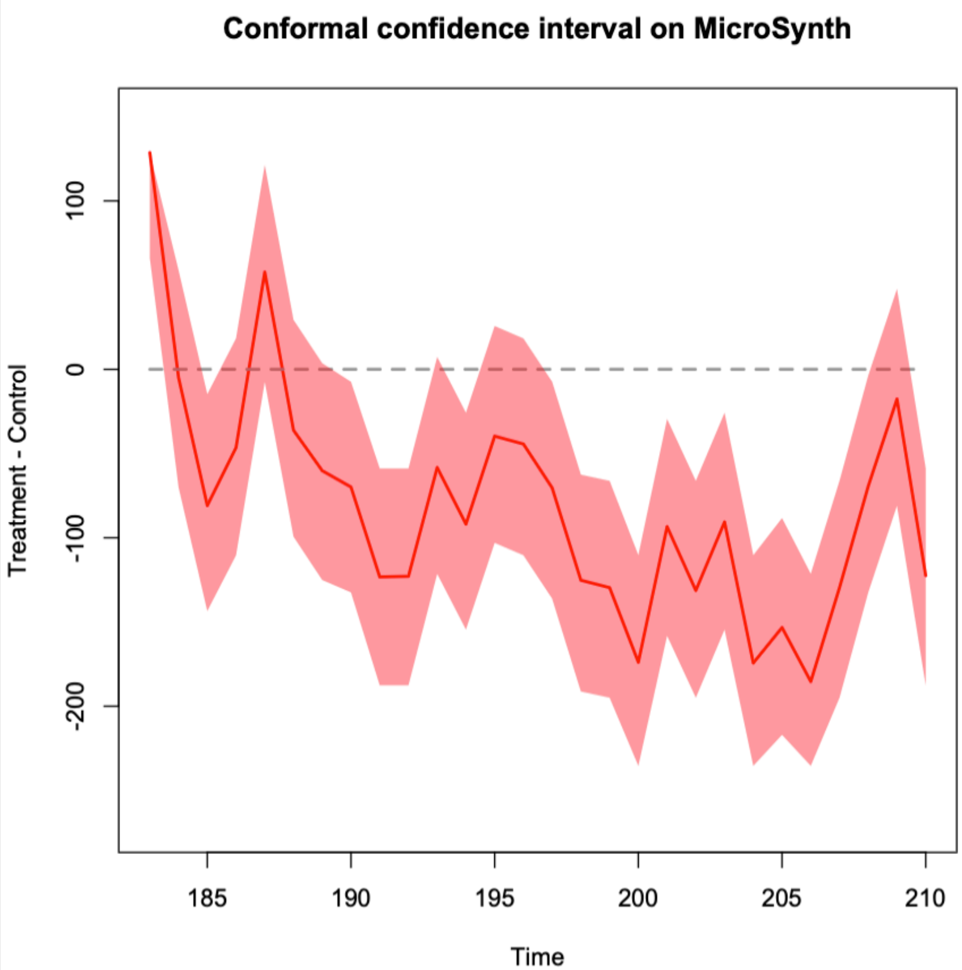


- False positive: only 1 for small τ; no false positive for larger τ
- True positive: all four variables



- The four significant predictors are year 2001—2004
- Coefficient of the four years are significantly negative — after the 911 attack, delays reduced due to reduced air traffic and new regulation to relieve airport congestion and delay

Future research: Scalable causal inference



- Fundamental problem of causal inference (Rubin 1974): can't observe the "untreated outcomes" of treated units
- To measure the treatment effect, one needs to synthesize the untreated outcomes of the treated using, e.g. the control units
- Challenge: in industrial level data, control/treated units are very large (possibly in billions)

Thank you!

https://arxiv.org/abs/2102.10080