This form documents the artifacts associated with the article (i.e., the data and code supporting the computational findings) and describes how to reproduce the findings.

Part 1: Data

- ☐ This paper does not involve analysis of external data (i.e., no data are used or the only data are generated by the authors via simulation in their code).
- ☒ I certify that the author(s) of the manuscript have legitimate access to and permission to use the data used in this manuscript.

Abstract

The datasets originate from the Voteview project, which tracks every congressional roll call vote in U.S. history. The datasets contain two files: `Sall_members.csv` providing background information on Senators, and `Sall_votes.csv` recording roll call vote outcomes. While Voteview updates continuously as new votes occur, the version used here represents a snapshot at the time of download and may differ slightly from the most current data available.

Availability

☒ Data **are** publicly available.
☐ Data **cannot be made** publicly available.

If the data are publicly available, see the *Publicly available data* section. Otherwise, see the *Non-publicly available data* section, below.

Publicly available data

- ☒ Data are available online at: https://voteview.com/data
- ☒ Data are available as part of the paper's supplementary material.
- ☐ Data are publicly available by request, following the process described here:
- ☐ Data are or will be made available through some other mechanism, described here:

Non-publicly available data

Description

File format(s)

☐ CSV or other plain text.
☐ Software-specific binary format (.Rda, Python pickle, etc.):
☐ Standardized binary format (e.g., netCDF, HDF5, etc.):
☐ Other (please specify):

Data dictionary

☒ Provided by authors in the following file(s): `Sall_votes.csv` and `Sall_members.csv`.
☐ Data file(s) is(are) self-describing (e.g., netCDF files)
☐ Available at the following URL

We provide the key components of the two datasets used in our analysis.

- `Sall_votes.csv`. A table with more than four million rows, each representing a voting record. The variables are:

- **congress**: The number of the congressional session.
- **icpsr**: A unique numeric identifier for each legislator.
- **rollnumber**: A unique numeric identifier for each roll call vote.
- **cast_code**: The recorded vote of the legislator (e.g., `1 = Yea`, `6 = Nay`, `9 = Present/Abstain`).
- **chamber**: A binary variable with values `President` or `Senate`

- `Sall_members.csv`. A table with nearly ten thousand rows, each representing a legislator. The variables are:
  - **congress**: The number of the congressional session.
  - **name**: The legislator's full name.
  - **icpsr**: A unique numeric identifier for the legislator (consistent with `Sall_votes.csv`).
  - **party_code**: A numeric code representing the legislator's political party.
  - **chamber**: A binary variable with values `President` or `Senate`

Additional Information (optional)

Part 2: Code

Abstract

We provide the code in reproducing the empirical sample complexity analysis, high-dimensional cases, and real-world data analysis.

Description

Code format(s)

☒ Script files
  ☒ R
  ☐ Python
  ☐ Matlab
  ☐ Other:
☒ Package
  ☒ R
  ☐ Python
  ☐ MATLAB toolbox
  ☐ Other:
☐ Reproducible report
  ☐ R Markdown
  ☐ Jupyter notebook
  ☐ Other:
☒ Shell script
☐ Other (please specify):

Supporting software requirements

Version of primary software used R version 4.1.2

Libraries and dependencies used by the code The R packages used in our experiments are listed below:

```
Package, Version
numDeriv, 2016.8-1.1
ROI, 1.0-1
CVXR, 1.0-15
ECOSolveR, 0.5.5
Matrix, 1.6-1
foreach, 1.5.2
parallel, 4.1.2
stringr, 1.5.1
here, 1.0.1
pROC, 1.19.0.1
reshape2, 1.4.4
dplyr, 1.1.4
ggnetwork, 0.5.12
ggpubr, 0.6.0
ggpmisc, 0.6.0
network, 1.18.1
sna, 2.7.1
```

Supporting system/hardware requirements (optional)

We conducted our experiments on a Linux platform, which we recommend for reproducibility. The system information is summarized below:

> Ubuntu 22.04.1 LTS, Intel(R) Xeon(R) Platinum 8352V CPU @ 2.10GHz, Memory 251GB

Parallelization used

☐ No parallel code used
☒ Multi-core parallelization on a single machine/node
  - Number of cores used: 50 cores

☐ Multi-machine/multi-node parallelization
  - Number of nodes and cores used:

License

☐ MIT License (default)
☐ BSD
☒ GPL v3.0
☐ Creative Commons
☐ Other: (please specify)

Additional information (optional)

Part 3: Reproducibility workflow

Scope

The provided workflow reproduces:

☐ Any numbers provided in text in the paper
☒ The computational method(s) presented in the paper (i.e., code is provided that implements the method(s))
☒ All tables and figures in the paper

☐ Selected tables and figures in the paper, as explained and justified below:

Workflow

- `simu_degree.R`: conduct experiments for empirical sample complexity analysis on the degree. It is essential for reproducing Figure 1 and Table S1.
- `simu_beta.R`: conduct experiments for empirical sample complexity analysis on the "maximum" signal. It reproduces Figure 2 and Figure S1.
- `simu_high.R`: conduct experiments for high-dimensional cases. It is helpful for reproducing Figure 3 and Figure S2.
- `simu_p.R`: empirical sample complexity analysis on the dimension. It is essential for reproducing Figure S3.
- `simu_ws.R`: empirical sample complexity analysis on the weakest signal. It reproduces Figure S4.
- `DataAnalysis.R`: for real-world data analysis. It can reproduce Figure 4.
- `batch.sh`: the shell script for simulations in the paper

Location

The workflow is available:

☐ As part of the paper's supplementary material.
☒ In this Git repository: to maintain anonymity during the review process, we have kept the code repository private. Our code and workflow will be published on `github.com` once it gets acceptance.
☐ Other (please specify):

Format(s)

☐ Single master code file
☒ Wrapper (shell) script(s)
☐ Self-contained R Markdown file, Jupyter notebook, or other literate programming approach
☒ Text file (e.g., a readme-style file) that documents workflow
☐ Makefile
☐ Other (more detail in *Instructions* below)

Instructions

Conduct the following code to reproduce the results in **simulation studies**:

```
chmod 777 batch.sh
./batch.sh
```

Get results in **real-world data analysis** via conducting the R script `DataAnalysis.R`

Expected run-time

Approximate time needed to reproduce the analyses on a standard desktop machine:

☐ < 1 minute
☐ 1-10 minutes
☐ 10-60 minutes
☐ 1-8 hours
☒ > 8 hours
☐ Not feasible to run on a desktop machine, as described here:

Additional information (optional)

Notes (optional)