Introduction

This repository is intended for replication of the numerical results appearing in the main text and online appendices of the paper "Selecting Penalty Parameters of High-Dimensional M-Estimators using Bootstrapping after Cross-Validation," authored by (Denis) [Chetverikov] and (Jesper Riis-Vestergaard) [Sørensen], and accepted for publication in the [*Journal of Political Economy*]. [arXiv version]

Reproducibility Workflow

All simulations were carried out in `R` (version 4.2.2) with cross-validation done using `glmnet::cv.glmnet` (`glmnet` version 4.1.6) and refitting following variable selection using `stats::glm`.

Note: To access the main script and functions (`bcvBinary.R`), for both the simulations and the empirical illustration the working directory is taken as the root directory (`hdme-jpe`).

Main Scripts and Functions

The file `bcvBinary.R`, placed in the root directory, contains the main functions for calculating the ℓ_1-penalized M-estimates for a given dataset with penalty parameter selected by bootstrapping-after-cross-validation (BCV-ℓ_1-ME), possibly with refitting following variable selection (post-BCV-ℓ_1-ME). Specifically, the functions `bcv_binary` and `post_bcv_binary` implement the BCV-ℓ_1-ME and post-BCV-ℓ_1-ME, respectively, in the context of the binary response model, taking a link function as input (the default link being `link = "logit"`). These two functions correspond to respectively Steps 1.a and 1.b of Algorithm 5.1 of the main text, when the penalty rule (λ_1) is BCV. The functions `bcv_linear` and `post_bcv_linear` implement the BCV-ℓ_1-ME and post-BCV-ℓ_1-ME, respectively, in the context of the linear (mean) regression models. These two functions are used for Steps 2.a and 2.b of Algorithm 5.1 of the main text, when the penalty rule (λ_2) is BCV. Finally, the functions `debias_bcv` and `debias_post_bcv` carry out three-step debiasing as in Step 3 of Algorithm 5.1 of the main text following BCV in both Steps 1.a and 2.a without any refitting (`debias_bcv`) or with refitting (`debias_post_bcv`) in both steps.

Simulations

The scripts `simBinary.R` and `simLinear.R` in the `simulations` folder contain helper functions for simulating from the models appearing in Section 6 of the main text as well as Online Appendices H and I.

When running the scripts mentioned below, all workspaces are output to the `simulations` folder, and all figures are output to `simulations/img` subfolder.

Section 6 of the Main Text and Online Appendix H.2

For the simulations in Section 6 of the main text and the additional simulations in Online Appendix H, run the script `hd_binary.v02.R`, which produces the workspace (`simulations_results_probit_A2_C3_N3_Rho5_R2000_B1000_K3_interceptTRUE_standardizeT`

Based on this workspace, the script `create_figures_v05.R` then produces the (non-existence) numbers reported in Section 6.3.1, Figures 1-5 in Sections 6.2 and 6.3, and Figures H.2.1-H.2.5 in Online Appendix H.2.

Online Appendix H.1

The script `mu_by_sim.R` approximates the debiasing coefficient vector μ_0 (associated with the simulations in Section 6 of the main text) via simulation and saves the approximation in the workspace `mu_sims_S1e+05_p100.Rdata`.

Based on this workspace, the script `mu_plots.R` then produces Figures H.1.1 and H.1.2 in Online Appendix H.1.

Online Appendix I

For the numerical comparisons with existing penalty selection methods in Online Appendix I, run the scripts `BCCH12ECTAcomp.R` and `BCW16JBEScomp.R` for Online Appendices I.1 and I.2, respectively. These scripts create the workspaces `BCCH12ECTA_comparison_linear_N3_Rho5_R2000_B1000_K3_interceptFALSE_standardizeFALSE.Rdata` and `BCW16JBES_comparison_logit_N3_Rho5_R2000_B1000_K3_interceptFALSE_standardizeFALSE.Rdata`, respectively.

Based on these workspaces, the scripts `BCCH12ECTAcomp_figures.R` and `BCW16JBEScomp_figures.R` then produce Figures I.1.1 and I.2.1, respectively.

Scope and Runtimes

All simulation-related scripts were run on a Linux server with 88 CPUs and 125 GB of RAM. Within this environment, the main text simulation script `hd_binary.v02.R` can be run in less than 48 hours. The comparison scripts `BCCH12ECTAcomp.R` and `BCW16JBEScomp.R` can be run in less than an hour each.

While a full-scale replication is not feasible on a standard desktop machine, the user can reduce the number of Monte Carlo replications (assigned to `nummc` within each script) from the 2,000 used in the paper to 10-100 and run the scripts in reasonable time. The resulting workspaces can then be used to reproduce less smooth versions of the figures appearing in the paper.

Empirical Illustration

The `application` folder contains the compressed dataset `ppcs_full.zip` used in the empirical illustration in Section 7 of the main text.

The script `application_v05.R` in the `application` folder unzips the dataset and uses it to produce the numbers reported in or around Tables 1-3 in Section 7 of the main text. The relevant numbers are output to the terminal.

The script `application_v05.R` can be run on a standard desktop machine. For example, with an an Intel(R) Core(TM) i7-8700 CPU (3.20GHz, 6 Cores, 12 Logical Processors) and 32 GB RAM, the script takes less than 10 minutes.