Package: leakyIV (via r-universe)

October 30, 2024

Title Leaky Instrumental Variables

Version 0.0.1

Date 2024-04-09

Maintainer David S. Watson <david.s.watson11@gmail.com>

Description Instrumental variables (IVs) are a popular and powerful tool for estimating causal effects in the presence of unobserved confounding. However, classical methods rely on strong assumptions such as the exclusion criterion, which states that instrumental effects must be entirely mediated by treatments. In the so-called ``leaky'' IV setting, candidate instruments are allowed to have some direct influence on outcomes, rendering the average treatment effect (ATE) unidentifiable. But with limits on the amount of information leakage, we may still recover sharp bounds on the ATE, providing partial identification. This package implements methods for ATE bounding in the leaky IV setting with linear structural equations. For details, see Watson et al. (2024) <doi:10.48550/arXiv.2404.04446>.

License GPL (>= 3)

URL https://github.com/dswatson/leakyIV

BugReports https://github.com/dswatson/leakyIV/issues

Imports data.table, corpcor, glasso, Matrix, mvnfast, foreach

Encoding UTF-8

RoxygenNote 7.3.1

Repository https://dswatson.r-universe.dev

RemoteUrl https://github.com/dswatson/leakyiv

RemoteRef HEAD

RemoteSha bfab1c0e42717950a14693b878ffda25c12b3da9

Contents

`exclusion_test` *Testing Exclusion*

Description

Performs a Monte Carlo test against the null hypothesis that minimum leakage is zero, a necessary but insufficient condition for exclusion.

Usage

```
exclusion_test(
  dat,
  normalize = TRUE,
  method = "mle",
  approx = TRUE,
  n_sim = 1999L,
  parallel = TRUE,
  return_stats = FALSE,
  ...
)
```

Arguments

`dat`	Input data. Either (a) an $n \times d$ data frame or matrix of observations with columns for treatment, outcome, and candidate instruments; or (b) a $d \times d$ covariance matrix over such variables. Note that in either case, the order of variables is presumed to be treatment (X), outcome (Y), leaky instruments (Z). `exclusion_test` requires at least two candidate instruments Z.
`normalize`	Scale candidate instruments to unit variance?
`method`	Estimator for the covariance matrix. Options include (a) `"mle"`, the default; (b) `"shrink"`, an analytic empirical Bayes solution; or (c) `"glasso"`, the graphical lasso. See details.
`approx`	Use nearest positive definite approximation if the estimated covariance matrix is singular? See details.
`n_sim`	Number of Monte Carlo replicates.
`parallel`	Run Monte Carlo simulations in parallel? Must register backend beforehand, e.g. via `doParallel`.
`return_stats`	Return observed statistic and simulated null distribution?
`...`	Extra arguments to be passed to graphical lasso estimator if `method = "glasso"`. Note that the regularization parameter `rho` is required as input, with no default.

Details

The classic linear instrumental variable (IV) model relies on the *exclusion criterion*, which states that instruments Z have no direct effect on the outcome Y, but can only influence it through the treatment X. This implies a series of tetrad constraints that can be directly tested, given a model for sampling data from the covariance matrix of the observable variables (Watson et al., 2024).

We assume that data are multivariate normal and impose the null hypothesis by modifying the estimated covariance matrix to induce a linear dependence between the vectors for $\text{Cov}(Z, X)$ and $\text{Cov}(Z, Y)$. Our test statistic is the determinant of the cross product of these vectors, which equals zero if and only if the null hypothesis is true. We generate a null distribution by simulating from the null covariance matrix and compute a *p*-value by estimating the proportion of statistics that exceed the observed value. Future releases will provide support for a wider range of data generating processes.

Numerous methods exist for estimating covariance matrices. `exclusion_test` provides support for maximum likelihood estimation (the default), as well as empirical Bayes shrinkage via `corpcor::cov.shrink` (Schäfer & Strimmer, 2005) and the graphical lasso via `glasso::glasso` (Friedman et al., 2007). These latter methods are preferable in high-dimensional settings where sample covariance matrices may be unstable or singular. Alternatively, users can pass a pre-computed covariance matrix directly as `dat`.

Estimated covariance matrices may be singular for some datasets or Monte Carlo samples. Behavior in this case is determined by the `approx` argument. If `TRUE`, the test proceeds with the nearest positive definite approximation, computed via Higham's (2002) algorithm (with a warning). If `FALSE`, the sampler will attempt to use the singular covariance matrix (also with a warning), but results may be invalid.

Value

Either a scalar representing the Monte Carlo *p*-value of the exclusion test (default) or, if `return_stats = TRUE`, a named list with three entries: `psi`, the observed statistic; `psi0`, a vector of length `n_sim` with simulated null statistics; and `p_value`, the resulting *p*-value.

References

Watson, D., Penn, J., Gunderson, L., Bravo-Hermsdorff, G., Mastouri, A., and Silva, R. (2024). Bounding causal effects with leaky instruments. *arXiv* preprint, 2404.04446.

Spirtes, P. Calculation of entailed rank constraints in partially non-linear and cyclic models. In *Proceedings of the 29th Conference on Uncertainty in Artificial Intelligence*, 606–615, 2013.

Friedman, J., Hastie, T., and Tibshirani, R. (2007). Sparse inverse covariance estimation with the lasso. *Biostatistics*, 9:432-441.

Schäfer, J., and Strimmer, K. (2005). A shrinkage approach to large-scale covariance estimation and implications for functional genomics. *Statist. Appl. Genet. Mol. Biol.*, 4:32.

Higham, N. (2002). Computing the nearest correlation matrix: A problem from finance. *IMA J. Numer. Anal.*, 22:329–343.

Examples

```
set.seed(123)
```

```
# Hyperparameters
n <- 200
d_z <- 4
beta <- rep(1, d_z)
theta <- 2
rho <- 0.5

# Simulate correlated residuals
S_eps <- matrix(c(1, rho, rho, 1), ncol = 2)
eps <- matrix(rnorm(n * 2), ncol = 2)
eps <- eps %*% chol(S_eps)

# Simulate observables from the linear IV model
z <- matrix(rnorm(n * d_z), ncol = d_z)
x <- z %*% beta + eps[, 1]
y <- x * theta + eps[, 2]
obs <- cbind(x, y, z)

# Compute p-value of the test
exclusion_test(obs, parallel = FALSE)
```

---

leakyIV *Bounding Causal Effects with Leaky Instruments*

Description

Estimates bounds on average treatment effects in linear IV models under limited violations of the exclusion criterion.

Usage

```
leakyIV(
  dat,
  tau,
  p = 2,
  normalize = TRUE,
  method = "mle",
  approx = TRUE,
  n_boot = NULL,
  bayes = FALSE,
  parallel = TRUE,
  ...
)
```

Arguments

`dat`	Input data. Either (a) an $n \times d$ data frame or matrix of observations with columns for treatment, outcome, and candidate instruments; or (b) a $d \times d$ covariance matrix over such variables. The latter is incompatible with bootstrapping. Note that in either case, the order of variables is presumed to be treatment (X), outcome (Y), leaky instruments (Z).
`tau`	Either (a) a scalar representing the upper bound on the p-norm of linear weights on Z in the structural equation for Y; or (b) a vector representing upper bounds on the absolute value of each such coefficient. See details.
`p`	Power of the norm for the `tau` threshold.
`normalize`	Scale candidate instruments to unit variance?
`method`	Estimator for the covariance matrix, if one is not supplied by `dat`. Options include (a) `"mle"`, the default; (b) `"shrink"`, an analytic empirical Bayes solution; or (c) `"glasso"`, the graphical lasso. See details.
`approx`	Use nearest positive definite approximation if the estimated covariance matrix is singular? See details.
`n_boot`	Optional number of bootstrap replicates.
`bayes`	Use Bayesian bootstrap?
`parallel`	Compute bootstrap estimates in parallel? Must register backend beforehand, e.g. via `doParallel`.
`...`	Extra arguments to be passed to graphical lasso estimator if `method = "glasso"`. Note that the regularization parameter `rho` is required as input, with no default.

Details

Instrumental variables are defined by three structural assumptions: they must be (A1) *relevant*, i.e. associated with the treatment; (A2) *unconfounded*, i.e. independent of common causes between treatment and outcome; and (A3) *exclusive*, i.e. only affect outcomes through the treatment. The `leakyIV` algorithm (Watson et al., 2024) relaxes (A3), allowing some information leakage from IVs Z to outcomes Y in linear systems. While the average treatment effect (ATE) is no longer identifiable in this setting, sharp bounds can be computed exactly.

We assume the following structural equation for the treatment: $X := Z\beta + \epsilon_X$, where the final summand is a noise term that correlates with the additive noise in the structural equation for the outcome: $Y := Z\gamma + X\theta + \epsilon_Y$. The ATE is given by the parameter θ. Whereas classical IV models require each γ coefficient to be zero, we permit some direct signal from Z to Y. Specifically, `leakyIV` provides support for two types of information leakage: (a) thresholding the *p*-norm of linear weights γ (scalar `tau`); and (b) thresholding the absolute value of each γ coefficient one by one (vector `tau`).

Numerous methods exist for estimating covariance matrices. `leakyIV` provides support for maximum likelihood estimation (the default), as well as empirical Bayes shrinkage via `corpcor::cov.shrink` (Schäfer & Strimmer, 2005) and the graphical lasso via `glasso::glasso` (Friedman et al., 2007). These latter methods are preferable in high-dimensional settings where sample covariance matrices may be unstable or singular. Alternatively, users can pass a pre-computed covariance matrix directly as `dat`.

Estimated covariance matrices may be singular for some datasets or bootstrap samples. Behavior in this case is determined by the `approx` argument. If `TRUE`, `leakyIV` proceeds with the nearest positive definite approximation, computed via Higham's (2002) algorithm (with a warning). If `FALSE`, bounds are NA (also with a warning).

Uncertainty can be evaluated in leaky IV models using the bootstrap, provided that covariances are estimated internally and not passed directly. Bootstrapping provides a nonparametric sampling distribution for min/max values of the ATE. Set `bayes = TRUE` to replace the classical bootstrap with a Bayesian bootstrap for approximate posterior inference (Rubin, 1981).

Value

A data frame with columns for `ATE_lo` and `ATE_hi`, representing lower and upper bounds of the partial identification interval for the causal effect of X on Y. When bootstrapping, the output data frame contains `n_boot` rows, one for each bootstrap replicate.

References

Watson, D., Penn, J., Gunderson, L., Bravo-Hermsdorff, G., Mastouri, A., and Silva, R. (2024). Bounding causal effects with leaky instruments. *arXiv* preprint, 2404.04446.

Friedman, J., Hastie, T., and Tibshirani, R. (2007). Sparse inverse covariance estimation with the lasso. *Biostatistics*, 9:432-441.

Schäfer, J., and Strimmer, K. (2005). A shrinkage approach to large-scale covariance estimation and implications for functional genomics. *Statist. Appl. Genet. Mol. Biol.*, 4:32.

Higham, N. (2002). Computing the nearest correlation matrix: A problem from finance. *IMA J. Numer. Anal.*, 22:329–343.

Rubin, D.R. (1981). The Bayesian bootstrap. *Ann. Statist.*, *9*(1): 130-134.

Examples

```
set.seed(123)

# Hyperparameters
n <- 200
d_z <- 4
beta <- rep(1, d_z)
gamma <- rep(0.1, d_z)
theta <- 2
rho <- 0.5

# Simulate correlated residuals
S_eps <- matrix(c(1, rho, rho, 1), ncol = 2)
eps <- matrix(rnorm(n * 2), ncol = 2)
eps <- eps %*% chol(S_eps)

# Simulate observables from a leaky IV model
z <- matrix(rnorm(n * d_z), ncol = d_z)
x <- z %*% beta + eps[, 1]
y <- z %*% gamma + x * theta + eps[, 2]
obs <- cbind(x, y, z)
```

```
# Run the algorithm
leakyIV(obs, tau = 1)

# With bootstrapping
leakyIV(obs, tau = 1, n_boot = 10)

# With covariance matrix input
S <- cov(obs)
leakyIV(S, tau = 1)
```

Index