Package 'fastconley'

September 26, 2026

Type Package

Title Fast Conley Standard Errors for 'lfe' and 'fixest' Models

Version 0.11.1

Maintainer Richard Bluhm <richard.bluhm@gmail.com>

Description Conley (1999) <doi:10.1016/S0304-4076(98)00084-0> spatial heteroscedasticity and autocorrelation consistent (HAC) standard errors for fixed effects panel and cross-sectional models estimated with felm() from the 'lfe' package (ordinary least squares and instrumental variables) or with feols(), feglm(), and fepois() from the 'fixest' package. Instrumental-variable support is limited to ordinary two-stage least squares. Generalized linear model fits use the M-estimation sandwich built from the stored scores and inverse Hessian. The spatial path uses score accumulation, a three-dimensional cell-grid neighbour search, and compressed sparse row neighbour lists instead of dense distance matrices, yielding large speedups over the original 'conley' package <https://github.com/rbluhm/conley> on big cross-sections and high-dimensional regressions.

License MIT + file LICENSE

URL https://github.com/rbluhm/fastconley, https://rbluhm.github.io/fastconley/

BugReports https://github.com/rbluhm/fastconley/issues

Encoding UTF-8

Depends R (>= 4.0)

Imports data.table, Rcpp, stats

Suggests fixest, knitr, lfe, rmarkdown, testthat (>= 3.0.0)

LinkingTo Rcpp, RcppArmadillo

VignetteBuilder knitr

Config/roxygen2/version 8.0.0

NeedsCompilation yes

Author Richard Bluhm [aut, cre, cph]

Repository CRAN

Date/Publication 2026-09-26 17:00:02 UTC

Contents

`vcovSpHAC` *Spatial HAC variance-covariance matrix*

Description

Computes Conley (1999) spatial HAC variance-covariance matrices for models estimated with `lfe::felm()` (OLS and IV/2SLS) or with `fixest`'s `feols()` (OLS and IV), `feglm()`, and `fepois()`. For GLM fits the variance is the M-estimation sandwich built from the stored score matrix and inverse Hessian. The spatial meat uses a fast CSR/cumulative-score implementation in C++; see `vcovSpHAC.felm` and `vcovSpHAC.fixest` for the per-method argument lists.

Usage

```
vcovSpHAC(reg, ...)
```

Arguments

`reg`	A fitted model object.
`...`	Method-specific arguments.

Value

A numeric matrix (base `"matrix"`) of dimension `k x k`, where `k` is the number of estimated coefficients (absorbed fixed effects excluded): the Conley spatial HAC estimate of the variance-covariance matrix of the coefficient estimates, i.e. the sandwich `bread %*% meat %*% bread` with the kernel-weighted spatial (and, with `lag_cutoff > 0`, serial) cross products in the meat. Row and column names are the coefficient names of the fit, so the matrix can be passed wherever a `vcov` is expected, e.g. `lmtest::coeftest(reg, vcov = V)` or `summary(reg, vcov = V)` for fixest fits, and `sqrt(diag(V))` gives the standard errors. The matrix is symmetric; with the default `ssc = TRUE` it is scaled by `n / (n - K)`, and with the default `psd_fix = TRUE` it is positive semi-definite.

`vcovSpHAC.felm` *Spatial HAC variance-covariance matrix for felm() models*

Description

The fit must have been called with `felm(..., keepCX = TRUE)` so the centered design matrix is stored on the object. IV/2SLS fits (the `felm` multi-part formula with `(endog ~ instruments)`) work out of the box: `lfe` stores the projected (second-stage) design in `cX` and the structural residuals in `residuals`, which is exactly the 2SLS sandwich. Weighted fits are supported (the scores carry the weights and the bread uses $X'WX$). Fits made through lfe's k-class path (any `kclass =` argument, including `kclass = 1`) are rejected: lfe then stores the raw endogenous regressors instead of the projected design, so the sandwich would be silently wrong. Refit without `kclass` for ordinary 2SLS.

Usage

```
## S3 method for class 'felm'
vcovSpHAC(
  reg,
  unit = NULL,
  time = NULL,
  lat = NULL,
  lon = NULL,
  kernel = c("bartlett", "uniform"),
  dist_fn = c("haversine", "spherical", "chord"),
  dist_cutoff = NULL,
  lag_cutoff = 0,
  verbose = FALSE,
  balanced_pnl = FALSE,
  ncores = NULL,
  pixel = 0,
  neighbor = c("grid", "band"),
  csr_weight = c("double", "float"),
  method = c("auto", "pairwise", "grid"),
  ssc = TRUE,
  psd_fix = TRUE,
  maxobsmem = 50000L,
  data = NULL,
  ...
)
```

Arguments

`reg`	A fitted object of class "felm", including IV fits.
`unit`	Optional name of the panel unit variable: an absorbed fixed effect or a column of the model data. When `unit` and `time` are both omitted the data are treated as one cross-section (each row its own unit, every pair within the cutoff enters);

	nothing is inferred from the absorbed fixed effects. Required for `lag_cutoff > 0`.
`time`	Optional name of the time variable. A supplied `time` is honoured even when `unit` is omitted (each row is then its own unit), but `lag_cutoff > 0` requires `unit`. Character/factor times are parsed as their numeric labels when possible; serial HAC rejects labels that are not numeric-parsable because time gaps determine lag weights.
`lat`	Name of the latitude variable. If `NULL` (default), it is auto-detected from the data's column names (`"lat"`, `"latitude"`, case-insensitive); a message reports the pick.
`lon`	Name of the longitude variable. If `NULL` (default), it is auto-detected (`"lon"`, `"long"`, `"longitude"`, `"lng"`, case-insensitive).
`kernel`	Spatial kernel, either "bartlett" or "uniform".
`dist_fn`	Distance function, one of "haversine", "spherical", "chord".
`dist_cutoff`	Spatial cutoff in km.
`lag_cutoff`	Serial HAC lag cutoff.
`verbose`	Print progress messages.
`balanced_pnl`	Whether the panel is balanced and unit locations are time-invariant.
`ncores`	Number of threads for the C++ spatial and serial routines (results do not depend on it). The default uses `getOption("fastconley.ncores")` when set, otherwise all detected logical cores. Under a true-ish `_R_CHECK_LIMIT_CORES_`, the resolved value is capped at two. Values are rounded to a positive integer.
`pixel`	Score-pre-aggregation cell size, in kilometres. Default 0 (exact-coordinate dedupe only). If 'pixel > 0', points are snapped to a uniform 'pixel'-km grid before the dedupe — a speed/accuracy trade-off that can move a point by up to roughly `pixel / sqrt(2)` and can change a pair distance by up to roughly `sqrt(2) * pixel`.
`neighbor`	Neighbor-search strategy for the spatial meat: "grid" (default; 3D cell grid, output-sensitive candidate enumeration) or "band" (deprecated; latitude band scan, the pre-0.5.0 behavior). Both are exact and use identical per-pair accept tests; results agree to floating-point summation order.
`csr_weight`	Storage precision for the balanced-path bartlett kernel weights: "double" (default, exact) or "float" (halves the per-pair weight memory; introduces at most ~6e-8 relative error per weight). Ignored for `kernel = "uniform"`, which stores no weights.
`method`	Spatial meat engine. "pairwise" enumerates neighbor pairs and works for any data. "grid" uses the exact grid-native meat — requires observations on a regular lat/lon lattice (e.g. raster data); cost is independent of the pair count, so it is dramatically faster on dense grids with large cutoffs. The uniform kernel uses sliding-window prefix sums; the bartlett kernel uses per-ring-pair FFT convolutions. Lattices spanning the full longitude circle wrap correctly across the dateline. "auto" (default) picks "grid" when it detects a lattice and a flop-balance estimate says it wins; both engines are exact, so the choice only affects speed (results agree to FP summation order, plus ~1e-12 acos conditioning for the

bartlett spherical/chord weights). Pass `verbose = TRUE` to see which engine ran; if you expect gridded data to use the grid engine and it does not, run once with `method = "grid"` — it errors with the specific reason instead of falling back.

`ssc` Small-sample correction. If `TRUE` (default), the variance matrix is scaled by `n / (n - K)` where `K` counts all estimated parameters including absorbed fixed-effect levels (taken from the regression's parameter count, independent of any clustering used when fitting). This matches `fixest`'s default Conley correction (its cluster adjustment is a no-op for Conley vcovs). Note that `K` is the fitting package's own count: with three or more absorbed fixed effects `lfe` and `fixest` can count the estimable levels differently (e.g. 30 versus 33 for the same model), so the two methods then differ by exactly that factor while their `ssc = FALSE` results are identical. Pass `FALSE` for no correction — that reproduces `rbluhm/conley`, fastconley versions before 0.9.0, and `fixest` with `ssc(adj = FALSE, cluster.adj = FALSE)`.

`psd_fix` The spatial kernels do not guarantee a positive semi-definite variance matrix. If `TRUE` (default), negative eigenvalues are clamped (to 1e-16, as `fixest`'s `vcov_fix` does) and a warning reports when the fix noticeably changed the matrix. If `FALSE`, the matrix is returned as computed, with a warning when it is not positive semi-definite.

`maxobsmem` Deprecated and ignored by the fast spatial path. Supplying it produces a warning; the argument remains for backward compatibility.

`data` Optional. The data frame to draw `lat`/`lon` from. If `NULL` (default), the data is recovered from the fit's call and aligned via a model-frame re-evaluation. Pass it explicitly if the original data has gone out of scope — and when no rows were dropped at fit time (no NAs, no `subset`), the coordinates are then taken by direct column access with no model-frame rebuild at all.

`...` Must be empty; unknown arguments are rejected.

Value

A numeric `k` x `k` matrix, the Conley spatial HAC variance-covariance estimate of the coefficients, with the coefficient names as dimnames; see vcovSpHAC for the structure and how to use it.

Examples

```
if (requireNamespace("lfe", quietly = TRUE)) {
  ## Cross-section on a regular 0.5-degree raster with holes. method =
  ## "grid" forces the exact grid engine; the default method = "auto"
  ## picks it automatically when the raster is large enough to win
  ## (on a toy example this small, pairwise is just as fast).
  set.seed(1)
  cells <- expand.grid(lat = seq(40, 49.5, by = 0.5),
                       lon = seq(-10, 9.5, by = 0.5))
  cells <- cells[sample(nrow(cells), 600), ]   # irregular occupancy
  cells$x <- rnorm(nrow(cells))
  cells$y <- 0.5 * cells$x + rnorm(nrow(cells))

  fit <- lfe::felm(y ~ x, data = cells, keepCX = TRUE)
  V <- vcovSpHAC(fit, lat = "lat", lon = "lon",
```

```
                kernel = "bartlett", dist_fn = "spherical",
                dist_cutoff = 200, ncores = 2, method = "grid",
                data = cells)
  sqrt(diag(V))

  ## Panel with spatial + serial HAC (scattered points: pairwise engine)
  pnl <- data.frame(unit = rep(1:200, each = 5),
                    time = rep(1:5, times = 200),
                    lat = rep(runif(200, 40, 50), each = 5),
                    lon = rep(runif(200, -10, 10), each = 5))
  pnl$x <- rnorm(nrow(pnl))
  pnl$y <- 0.5 * pnl$x + rnorm(nrow(pnl))
  fit2 <- lfe::felm(y ~ x | unit + time, data = pnl, keepCX = TRUE)
  V2 <- vcovSpHAC(fit2, unit = "unit", time = "time",
                  lat = "lat", lon = "lon", kernel = "bartlett",
                  dist_fn = "haversine", dist_cutoff = 300,
                  lag_cutoff = 2, balanced_pnl = TRUE, ncores = 2,
                  data = pnl)
  sqrt(diag(V2))
}
```

`vcovSpHAC.fixest` *Spatial HAC variance-covariance matrix for fixest models*

Description

Supports `fixest::feols()` (including IV/2SLS) and `fixest::feglm()` / `fixest::fepois()` fits.

Usage

```
## S3 method for class 'fixest'
vcovSpHAC(
  reg,
  unit = NULL,
  time = NULL,
  lat = NULL,
  lon = NULL,
  kernel = c("bartlett", "uniform"),
  dist_fn = c("haversine", "spherical", "chord"),
  dist_cutoff = NULL,
  lag_cutoff = 0,
  verbose = FALSE,
  balanced_pnl = FALSE,
  ncores = NULL,
  pixel = 0,
  neighbor = c("grid", "band"),
  csr_weight = c("double", "float"),
  method = c("auto", "pairwise", "grid"),
  ssc = TRUE,
```

```
  psd_fix = TRUE,
  data = NULL,
  ...
)
```

Arguments

reg	A fitted object of class "fixest": a feols() fit (including IV) with demeaned = TRUE, or a feglm() / fepois() fit (any family; lean = TRUE fits are rejected because they carry no score matrix).
unit	Optional name of the panel unit variable. If NULL, each row is its own unit. A positive lag_cutoff requires unit.
time	Optional name of the time variable. It is honoured even when unit is NULL, so it still defines the spatial time blocks. Character/factor times are parsed as their numeric labels when possible; serial HAC rejects labels that are not numeric-parsable because time gaps determine lag weights.
lat	Name of the latitude variable. If NULL (default), auto-detected from the data's column names. See vcovSpHAC.felm.
lon	Name of the longitude variable. If NULL (default), auto-detected. See vcovSpHAC.felm.
kernel	Spatial kernel, either "bartlett" or "uniform".
dist_fn	Distance function, one of "haversine", "spherical", "chord".
dist_cutoff	Spatial cutoff in km.
lag_cutoff	Serial HAC lag cutoff.
verbose	Print progress messages.
balanced_pnl	Whether the panel is balanced and unit locations are time-invariant.
ncores	Number of threads for the C++ spatial and serial routines (results do not depend on it). The default uses getOption("fastconley.ncores") when set, otherwise all detected logical cores, capped at two under a true-ish _R_CHECK_LIMIT_CORES_. Values are rounded to a positive integer.
pixel	Score-pre-aggregation cell size, in kilometres.
neighbor	Neighbor-search strategy: "grid" (default) or the deprecated "band" compatibility path. See vcovSpHAC.felm.
csr_weight	Balanced-path bartlett weight storage: "double" (default) or "float". See vcovSpHAC.felm.
method	Spatial meat engine: "auto" (default), "pairwise", or "grid". See vcovSpHAC.felm.
ssc	Small-sample correction (n / (n - K) when TRUE, the default). See vcovSpHAC.felm.
psd_fix	Clamp negative eigenvalues when TRUE (the default). See vcovSpHAC.felm.
data	Optional. The model frame to draw lat/lon/ unit/time from. If NULL (default), the data is recovered from the fit's call. Pass it explicitly if the original data has gone out of scope, or if you want to override.
...	Must be empty; unknown arguments are rejected.

Details

For `feols`, the fit must have been called with `feols(..., demeaned = TRUE)` so that the centered design matrix `X_demeaned` is stored on the fit object. Weighted fits are supported (the scores carry the weights and the bread uses $X'WX$, matching `fixest`'s own weighted Conley vcov). IV fits work out of the box: `X_demeaned` holds the projected (second-stage) design and `residuals` the structural residuals, which is exactly the 2SLS sandwich.

For `feglm` / `fepois`, no estimation flag is needed: the variance is the M-estimation sandwich $H^{-1}BH^{-1}$, built from the maximum-likelihood score matrix and inverse Hessian that `fixest` stores on every (non-`lean`) fit. Weights, offsets, and the fixed-effect profiling are already folded into the stored scores. This is the same construction `fixest`'s own `vcov_conley()` uses for GLMs — but with exact supported distance calculations, and with the serial-HAC panel extension available via `lag_cutoff` (which `fixest` does not offer for Conley vcovs).

The returned matrix can be passed to `fixest`'s `vcov` argument. For the usual `fixest` workflow, define a one-argument wrapper such as `function(x) vcovSpHAC(x, ...)` and pass that function to `summary()`, `etable()`, or `feols(vcov = )`. The wrapper keeps the coordinate names, cutoffs, panel variables, and optional `data =` argument together.

Value

A numeric k x k matrix, the Conley spatial HAC variance-covariance estimate of the coefficients, with the coefficient names as dimnames; see `vcovSpHAC` for the structure and how to use it.

Examples

```
if (requireNamespace("fixest", quietly = TRUE)) {
  ## feols must be fit with demeaned = TRUE (the keepCX analogue).
  set.seed(1)
  cells <- expand.grid(lat = seq(40, 49.5, by = 0.5),
                       lon = seq(-10, 9.5, by = 0.5))
  cells$x <- rnorm(nrow(cells))
  cells$y <- 0.5 * cells$x + rnorm(nrow(cells))

  fit <- fixest::feols(y ~ x, data = cells, demeaned = TRUE)
  vcov_fc <- function(x) {
    vcovSpHAC(x, lat = "lat", lon = "lon",
              kernel = "uniform", dist_fn = "spherical",
              dist_cutoff = 200, ncores = 2, data = cells)
  }
  V <- vcov_fc(fit)
  sqrt(diag(V))

  ## The same wrapper can be used directly in fixest's vcov argument.
  fit_sum <- summary(fit, vcov = vcov_fc)
  sqrt(diag(fit_sum$cov.scaled))

  ## Poisson (fepois / feglm): no demeaned = TRUE needed — the stored
  ## ML scores and inverse Hessian are used directly.
  cells$cnt <- rpois(nrow(cells), exp(0.4 * cells$x))
  fit_pois <- fixest::fepois(cnt ~ x, data = cells)
  V_pois <- vcovSpHAC(fit_pois, lat = "lat", lon = "lon",
```

```
                        kernel = "uniform", dist_fn = "spherical",
                        dist_cutoff = 200, ncores = 2, data = cells)
  sqrt(diag(V_pois))
}
```

Index