Package: SpatialInference (via r-universe)

October 24, 2026

Title Tools for Statistical Inference with Geo-Coded Data

Version 0.1.0

Description Fast computation of Conley (1999) <doi:10.1016/S0304-4076(98)00084-0> spatial heteroskedasticity and autocorrelation consistent (HAC) standard errors for linear regression models with geo-coded data, with a fast C++ implementation by Christensen, Hartman, and Samii (2021) <doi:10.1017/S0020818321000187>. Performance-critical distance calculations, kernel weighting, and variance component accumulation are implemented in C++ via 'Rcpp' and 'RcppArmadillo'. Includes tools for estimating the spatial correlation range from covariograms and correlograms following the bandwidth selection method proposed in Lehner (2026) <doi:10.48550/arXiv.2603.03997>, and diagnostic visualizations for bandwidth selection.

Depends R (>= 4.1.0)

License GPL (>= 3)

URL https://github.com/axlehner/SpatialInference

BugReports https://github.com/axlehner/SpatialInference/issues

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

LinkingTo Rcpp, RcppArmadillo

Imports Rcpp, sf, data.table, magrittr, stats, tibble

Suggests lfe, fixest, dplyr, stringr, spdep, ncf, gstat, sandwich, ggplot2, modelsummary, knitr, rmarkdown, geosphere, testthat (>= 3.0.0)

Config/testthat/edition 3

VignetteBuilder knitr

NeedsCompilation yes

Author Alexander Lehner [aut, cre] (ORCID:
<https://orcid.org/0000-0001-5885-5966>)

Maintainer Alexander Lehner <alehner@worldbank.org>

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

Date/Publication 2026-03-25 21:56:54 UTC

RemoteUrl https://github.com/cran/SpatialInference

RemoteRef HEAD

RemoteSha 0c9e7f5f8f6b87f2c73faac62dd04659089a3262

Contents

Bal_XeeXhC . . . 2
compute_conley_lfe . . . 3
conley_SE . . . 4
coords_as_columns . . . 6
covgm_range . . . 7
DistMat . . . 8
extract_corr_range . . . 8
gravity_centroid . . . 9
grid_FE . . . 10
inverseu_plot_conleyrange . . . 11
lm_sac . . . 13
TimeDist . . . 14
US_counties_centroids . . . 15
XeeXhC . . . 16
XeeXhC_Lg . . . 16

Index **18**

`Bal_XeeXhC` *Spatial Variance Component for Balanced Panel*

Description

Spatial Variance Component for Balanced Panel

Usage

```
Bal_XeeXhC(dmat, X, e, n1, k)
```

Arguments

dmat	pre-computed distance matrix (n x n)
X	model matrix (n x k)
e	vector of residuals (length n)
n1	number of observations
k	number of regressors

Value

A k x k matrix representing the spatial X'ee'X component.

---

compute_conley_lfe *Compute Conley Standard Error for a Single Coefficient*

Description

Convenience wrapper around conley_SE() that returns only the spatial Conley standard error for the first regressor. Useful for quick extraction of Conley SEs in scripting contexts.

Usage

```
compute_conley_lfe(lfeobj, cutoff, kernel_choice = "bartlett", ...)
```

Arguments

lfeobj	A regression object of class "felm" from lfe::felm(). Must be estimated with keepCX = TRUE and cluster variables lat + lon.
cutoff	Numeric. The spatial bandwidth (cutoff distance in km).
kernel_choice	Character string specifying the kernel function. Default is "bartlett". See conley_SE() for available kernels.
...	Additional arguments passed to conley_SE().

Value

A single numeric value: the spatial Conley standard error for the first (or only) regressor.

References

Lehner, A. (2026). Bandwidth selection for spatial HAC standard errors. *arXiv preprint* arXiv:2603.03997. doi:10.48550/arXiv.2603.03997

Conley, T. G. (1999). GMM estimation with cross sectional dependence. *Journal of Econometrics*, 92(1), 1–45. doi:10.1016/S03044076(98)000840

Examples

```
data(US_counties_centroids)
if (requireNamespace("lfe", quietly = TRUE)) {
  reg <- lfe::felm(noise1 ~ noise2 | unit + year | 0 | lat + lon,
                   data = US_counties_centroids, keepCX = TRUE)
  compute_conley_lfe(reg, cutoff = 500)
}
```

`conley_SE` *Conley Spatial HAC Variance-Covariance Estimation*

Description

Computes Conley (1999) spatial HAC (Heteroskedasticity and Autocorrelation Consistent) variance-covariance matrices for regression models estimated with `lfe::felm()`. Supports cross-sectional spatial correlation, serial (temporal) correlation, and the combined spatial HAC estimator. Multiple kernel functions and distance metrics are available.

Usage

```
conley_SE(
  reg,
  unit,
  time,
  lat,
  lon,
  kernel = "bartlett",
  dist_fn = "Haversine",
  dist_cutoff = 500,
  lag_cutoff = 5,
  lat_scale = 111,
  verbose = FALSE,
  cores = 1,
  balanced_pnl = FALSE
)
```

Arguments

`reg`	A regression object of class `"felm"` from `lfe::felm()`. Must be estimated with `keepCX = TRUE` and with latitude/longitude variables passed as cluster variables.
`unit`	Character string naming the cross-sectional unit identifier variable in the `felm` cluster variables.
`time`	Character string naming the time period identifier variable in the `felm` cluster variables.
`lat`	Character string naming the latitude variable in the `felm` cluster variables.

`lon`	Character string naming the longitude variable in the `felm` cluster variables.
`kernel`	Character string specifying the kernel function for spatial weighting. One of `"bartlett"` (default), `"epanechnikov"`, `"gaussian"`, `"parzen"`, `"biweight"`, or `"uniform"`.
`dist_fn`	Character string specifying the distance function. `"Haversine"` (default) for great-circle distance in km, or `"SH"` for the 111 km/degree approximation.
`dist_cutoff`	Numeric. The spatial bandwidth (cutoff distance in km) beyond which observations receive zero weight. Default is `500`.
`lag_cutoff`	Numeric. The temporal bandwidth (number of time periods) for serial correlation correction. Default is `5`.
`lat_scale`	Numeric. Scaling factor for latitude (km per degree). Default is `111`.
`verbose`	Logical. If `TRUE`, prints progress messages during computation. Default is `FALSE`.
`cores`	Integer. Number of CPU cores for parallel computation via `parallel::mclapply()`. Default is `1` (no parallelism).
`balanced_pnl`	Logical. If `TRUE`, assumes a balanced panel and pre-computes the distance matrix once for efficiency. Default is `FALSE`.

Value

A named list of three variance-covariance matrices, each of dimension k x k where k is the number of regressors:

OLS The standard OLS variance-covariance matrix from the felm object.

Spatial The spatial-only Conley VCV, correcting for cross-sectional spatial correlation.

Spatial_HAC The full spatial HAC VCV, correcting for both spatial and serial correlation.

References

Christensen, D., Hartman, A. C. and Samii, C. (2021). Legibility and external investment: An institutional natural experiment in Liberia. *International Organization*, 75(4), 1087–1108. doi:10.1017/S0020818321000187

Conley, T. G. (1999). GMM estimation with cross sectional dependence. *Journal of Econometrics*, 92(1), 1–45. doi:10.1016/S03044076(98)000840

Lehner, A. (2026). Bandwidth selection for spatial HAC standard errors. *arXiv preprint* arXiv:2603.03997. doi:10.48550/arXiv.2603.03997

Newey, W. K. and West, K. D. (1987). A simple, positive semi-definite, heteroskedasticity and autocorrelation consistent covariance matrix. *Econometrica*, 55(3), 703–708. doi:10.2307/1913610

Examples

```
data(US_counties_centroids)
if (requireNamespace("lfe", quietly = TRUE)) {
  reg <- lfe::felm(noise1 ~ noise2 | unit + year | 0 | lat + lon,
                   data = US_counties_centroids, keepCX = TRUE)
  vcvs <- conley_SE(reg, unit = "unit", time = "year",
```

```
                  lat = "lat", lon = "lon",
                  kernel = "bartlett", dist_cutoff = 500)
  # Spatial standard errors:
  sqrt(diag(vcvs$Spatial))
}
```

`coords_as_columns` *Extract Coordinates as Columns from an sf Object*

Description

Extracts point coordinates from an `sf` or `sfc` object and returns them as a tibble. This is a lightweight alternative to `sf::st_coordinates()` that returns a tibble directly.

Usage

```
coords_as_columns(x, names = c("x", "y"))
```

Arguments

`x`	An `sf` or `sfc` object with `sfc_POINT` geometry.
`names`	Character vector of length 2 specifying the column names for the x and y coordinates. Default is `c("x", "y")`.

Value

A `tibble::tibble()` with two columns named according to the `names` argument, containing the x and y coordinates.

References

Pebesma, E. (2018). Simple features for R: Standardized support for spatial vector data. *The R Journal*, 10(1), 439–446. doi:10.32614/RJ2018009

Examples

```
data(US_counties_centroids)
coords <- coords_as_columns(US_counties_centroids)
head(coords)
```

covgm_range *Covariogram Range Estimation and Visualization*

Description

Estimates a covariogram from an sf data frame (either from a single variable or regression residuals) using gstat::variogram() with covariogram = TRUE, extracts the zero-crossing as the estimated correlation range via extract_corr_range(), and produces a diagnostic plot.

Usage

```
covgm_range(
  df.input,
  depvar = "noise1",
  indepvar = "noise2",
  maxdist = NA,
  spacing = NA,
  single.variable = FALSE
)
```

Arguments

df.input	An sf data frame with point geometries.
depvar	Character string naming the dependent variable. Default is "noise1".
indepvar	Character string naming the independent variable (used when single.variable = FALSE). Default is "noise2".
maxdist	Numeric. Maximum distance for the covariogram (in metres). Default is NA, which uses 2/3 of the maximum pairwise distance.
spacing	Numeric. Bin width for the covariogram (in metres). Default is NA, which uses maxdist / 150.
single.variable	Logical. If TRUE, computes the covariogram directly from depvar. If FALSE (default), first regresses depvar on indepvar via fixest::feols() and uses the residuals.

Value

A ggplot2::ggplot() object showing the covariogram with the estimated correlation range marked by a vertical red line and annotated text.

References

Lehner, A. (2026). Bandwidth selection for spatial HAC standard errors. *arXiv preprint* arXiv:2603.03997. doi:10.48550/arXiv.2603.03997

Pebesma, E. J. (2004). Multivariable geostatistics in S: the gstat package. *Computers & Geosciences*, 30(7), 683–691. doi:10.1016/j.cageo.2004.03.012

Examples

```
data(US_counties_centroids)
if (requireNamespace("fixest", quietly = TRUE) &&
    requireNamespace("gstat", quietly = TRUE) &&
    requireNamespace("ggplot2", quietly = TRUE)) {
  covgm_range(US_counties_centroids)
}
```

DistMat *Create Distance Matrix*

Description

Create Distance Matrix

Usage

```
DistMat(M, cutoff, kernel = "bartlett", dist_fn = "Haversine")
```

Arguments

M	a matrix of locations (n x 2, latitude and longitude)
cutoff	the distance cutoff (bandwidth) in km
kernel	(string) kernel function (default is bartlett-triangular)
dist_fn	(string) distance function (Haversine)

Value

A symmetric n x n matrix of kernel weights with 1s on the diagonal.

---

extract_corr_range *Extract Correlation Range from a Correlogram or Covariogram*

Description

Identifies the distance at which spatial autocorrelation first crosses zero, providing an estimate of the spatial correlation range. Works with correlograms from `ncf::correlog()` and covariograms from `gstat::variogram()` (with `covariogram = TRUE`).

Usage

```
extract_corr_range(input, returnzeroifNA = FALSE)
```

Arguments

input	A correlogram object from ncf::correlog() (class "correlog") or a covariogram from gstat::variogram() (class "gstatVariogram"). Must be a covariogram (not a variogram) when using gstat.
returnzeroifNA	Logical. If TRUE, returns 1 instead of NA when no zero-crossing is found. Default is FALSE.

Details

For correlograms, the function detects the first sign change in the rounded and floored correlation values. For covariograms, it finds the first index where gamma transitions from positive to non-positive. In both cases, the estimated range is the midpoint between the last positive and first non-positive distance bins.

Value

A numeric value representing the estimated correlation range. For covariograms, the unit is km (distance in metres divided by 1000). For correlograms, the unit matches the input distance unit.

References

Lehner, A. (2026). Bandwidth selection for spatial HAC standard errors. *arXiv preprint* arXiv:2603.03997. doi:10.48550/arXiv.2603.03997

Pebesma, E. J. (2004). Multivariable geostatistics in S: the gstat package. *Computers & Geosciences*, 30(7), 683–691. doi:10.1016/j.cageo.2004.03.012

Examples

```
# With a mock gstatVariogram:
mock_vgm <- data.frame(
  np = rep(100, 10),
  dist = seq(50000, 500000, by = 50000),
  gamma = c(5, 3, 2, 1, 0.5, -0.2, -0.5, -0.3, -0.1, -0.05),
  dir.hor = 0, dir.ver = 0, id = "var1"
)
class(mock_vgm) <- c("gstatVariogram", "data.frame")
extract_corr_range(mock_vgm)
```

---

gravity_centroid *Weighted Mean Centre (Centre of Gravity)*

Description

Computes the mean centre of an sf data frame, optionally weighted by an attribute variable. The function first extracts polygon or point centroids using sf::st_centroid(), then calculates the (weighted) arithmetic mean of the x and y coordinates. This is the two-dimensional analogue of a weighted mean and corresponds to the "centre of gravity" or "mean centre" in spatial statistics. A common use case is computing the population-weighted centroid of a set of administrative units.

Usage

```
gravity_centroid(df.sf, weight = NA)
```

Arguments

df.sf	An sf data frame with polygon or point geometries. Polygon geometries are reduced to their centroids before computation.
weight	Numeric vector of weights with length equal to nrow(df.sf). If NA (the default), all observations receive equal weight and the result is the simple (unweighted) geographic mean centre. Weights do not need to sum to one; the function normalises internally.

Value

An sfc_POINT object (CRS 4326 / WGS84) representing the (weighted) mean centre as a single point.

References

Arlinghaus, S. L. (1994). *Practical Handbook of Spatial Statistics*. CRC Press.

Examples

```
data(US_counties_centroids)

# Unweighted mean centre (geographic centroid of all county centroids)
gravity_centroid(US_counties_centroids)

# Weighted mean centre (shifted toward areas with higher noise1 values)
gravity_centroid(US_counties_centroids,
                 weight = abs(US_counties_centroids$noise1) + 1)
```

---

grid_FE *Create Spatial Grid Fixed Effects*

Description

Overlays a regular rectangular grid on an sf data frame, intersects each observation with the grid, and returns a factor variable identifying the grid cell to which each observation belongs. This is useful for constructing spatial fixed effects in regression models: by including grid_FE() as a factor variable, the regression absorbs location-specific variation at the resolution of the chosen grid. Finer grids absorb more spatial variation but consume more degrees of freedom.

Usage

```
grid_FE(df.sf, size, distance = FALSE)
```

Arguments

df.sf	An sf data frame with geometries (points or polygons).
size	Numeric. When distance = FALSE (default), an integer specifying the number of grid cells along each axis (e.g., size = 10 creates a 10 x 10 grid). When distance = TRUE, a numeric value giving the cell side length in CRS units (e.g., metres for projected data). Passed to sf::st_make_grid() as n or cellsize, respectively.
distance	Logical. If FALSE (default), size is the number of cells per axis. If TRUE, size is the cell dimension in CRS units.

Details

The grid is constructed via sf::st_make_grid() and observations are assigned to cells via sf::st_intersection(). Observations that fall outside the grid (e.g., in coastal regions where cells do not cover the full bounding box) are dropped during intersection.

Value

A factor vector with one element per observation that survived the intersection (see Details), where each level is a grid cell ID.

References

Pebesma, E. (2018). Simple features for R: Standardized support for spatial vector data. *The R Journal*, 10(1), 439–446. doi:10.32614/RJ2018009

Examples

```
data(US_counties_centroids)

# Create a 5 x 5 grid of spatial fixed effects
grid_ids <- grid_FE(US_counties_centroids, size = 5)
table(grid_ids)

# Finer grid (10 x 10)
grid_ids_fine <- grid_FE(US_counties_centroids, size = 10)
length(levels(grid_ids_fine))
```

---

inverseu_plot_conleyrange

Inverse-U Plot of Conley Standard Errors vs. Cutoff Distance

Description

Visualizes the relationship between the spatial bandwidth (cutoff distance) and the resulting Conley standard error. This typically reveals an inverse-U shaped relationship, helping to identify the appropriate bandwidth for spatial HAC estimation.

Usage

```
inverseu_plot_conleyrange(
  df.input,
  cutoffrange = NA,
  kernel_choice_conley = "epanechnikov",
  depvar = "noise1",
  indepvar = "noise2",
  range_add = FALSE,
  ...
)
```

Arguments

df.input	An sf data frame containing the regression variables and columns lat, lon, unit, year.
cutoffrange	Numeric vector of cutoff distances (in km) to evaluate.
kernel_choice_conley	Character string specifying the kernel function. Default is "epanechnikov". See conley_SE() for options.
depvar	Character string naming the dependent variable column. Default is "noise1".
indepvar	Character string naming the independent variable column. Default is "noise2".
range_add	Logical. If TRUE, overlays the covariogram-estimated correlation range as a vertical red line. Default is FALSE.
...	Additional arguments (currently unused).

Value

A ggplot2::ggplot() object showing Conley SE (y-axis) against cutoff distance (x-axis), with the HC1 standard error as a grey dashed horizontal reference line.

References

Lehner, A. (2026). Bandwidth selection for spatial HAC standard errors. *arXiv preprint* arXiv:2603.03997. doi:10.48550/arXiv.2603.03997

Conley, T. G. (1999). GMM estimation with cross sectional dependence. *Journal of Econometrics*, 92(1), 1–45. doi:10.1016/S03044076(98)000840

Examples

```
data(US_counties_centroids)
if (requireNamespace("lfe", quietly = TRUE) &&
    requireNamespace("fixest", quietly = TRUE) &&
    requireNamespace("ggplot2", quietly = TRUE)) {
  inverseu_plot_conleyrange(US_counties_centroids,
                            cutoffrange = seq(100, 1000, by = 200))
}
```

lm_sac *Linear Model with Spatial Autocorrelation Diagnostics*

Description

Estimates a linear regression via `lfe::felm()` and augments the output with Moran's I tests for spatial autocorrelation, optional correlograms for range estimation, and Conley spatial HAC standard errors. The returned object has class `"custom"` prepended, enabling display via `modelsummary::modelsummary()` with custom `tidy` and `glance` methods.

Usage

```
lm_sac(
  formula.chr,
  data.sf,
  knn_number = 20,
  conley_cutoff = 5,
  conley_kernel = "bartlett",
  correlograms = FALSE,
  ...
)
```

Arguments

`formula.chr`	Character string specifying the regression formula in `felm` syntax (e.g., `"y ~ x1 + x2 | fe1 + fe2 | 0 | lat + lon"`).
`data.sf`	An `sf` data frame containing the variables referenced in `formula.chr`, with point geometries and columns `lat` and `lon`.
`knn_number`	Integer. Number of nearest neighbours for the spatial weights matrix used in Moran's I tests. Default is `20`.
`conley_cutoff`	Numeric. Spatial bandwidth (cutoff distance in km) for the Conley standard error. Default is `5`.
`conley_kernel`	Character string specifying the kernel function. Default is `"bartlett"`. See `conley_SE()` for options.
`correlograms`	Logical. If `TRUE`, estimates correlograms via `ncf::correlog()` and uses the extracted correlation range as an additional flexible Conley cutoff. Default is `FALSE`.
`...`	Additional arguments passed to `lfe::felm()` and `stats::lm()`.

Value

An object of class `c("custom", "lm")` with additional components:

spatial_FE Character, the spatial fixed effect variable name.

Moran_lmresid Moran's I test statistic on the OLS residuals, or `NA` if the test failed.

Moran_response Moran's I test statistic on the response variable, or NA if the test failed.

correlog.range_resid Estimated correlation range from the residual correlogram (km), or NA if `correlograms = FALSE`.

correlog.range_response Estimated correlation range from the response correlogram (km), or NA if `correlograms = FALSE`.

conley_SE Numeric vector of Conley spatial standard errors (with 0 for intercept and higher-order FE coefficients).

conley_SE_flex Conley SEs using the correlogram-based cutoff, or NA if `correlograms = FALSE`.

References

Lehner, A. (2026). Bandwidth selection for spatial HAC standard errors. *arXiv preprint* arXiv:2603.03997. doi:10.48550/arXiv.2603.03997

Conley, T. G. (1999). GMM estimation with cross sectional dependence. *Journal of Econometrics*, 92(1), 1–45. doi:10.1016/S03044076(98)000840

Moran, P. A. P. (1950). Notes on continuous stochastic phenomena. *Biometrika*, 37(1/2), 17–23. doi:10.2307/2332142

Bivand, R. S., Pebesma, E. and Gomez-Rubio, V. (2013). *Applied Spatial Data Analysis with R*. 2nd ed. Springer.

Examples

```
data(US_counties_centroids)
if (requireNamespace("lfe", quietly = TRUE) &&
    requireNamespace("spdep", quietly = TRUE) &&
    requireNamespace("stringr", quietly = TRUE) &&
    requireNamespace("dplyr", quietly = TRUE) &&
    requireNamespace("sandwich", quietly = TRUE)) {
  out <- lm_sac("noise1 ~ noise2 | unit + year | 0 | lat + lon",
                US_counties_centroids, conley_cutoff = 500)
  out$conley_SE
}
```

`TimeDist` *Temporal Variance Component (Time Dimension)*

Description

Temporal Variance Component (Time Dimension)

Usage

```
TimeDist(times, cutoff, X, e, n1, k)
```

Arguments

`times`	numeric vector of time period identifiers
`cutoff`	the lag cutoff (number of time periods)
`X`	model matrix (n x k)
`e`	vector of residuals (length n)
`n1`	number of observations
`k`	number of regressors

Value

A k x k matrix representing the temporal X'ee'X component.

`US_counties_centroids` *Centroids of Contiguous US Counties (2017)*

Description

An `sf` data frame containing the centroids of all 3,108 counties of the contiguous United States (2017 geographies), along with synthetic spatially-correlated noise variables for use in examples and vignettes.

Usage

```
US_counties_centroids
```

Format

An `sf` data frame with 3,108 rows and the following columns:

STATE Numeric state FIPS code.

NAME County name.

NAMELSAD County name with legal/statistical area description.

GISJOIN Unique ID for joining with IPUMS data (2017 geographies).

lat Latitude of the county centroid (WGS84).

lon Longitude of the county centroid (WGS84).

unit Cross-sectional unit identifier (constant `1` for cross-sectional use).

year Time period identifier (constant `1` for cross-sectional use).

noise1 Synthetic spatially-correlated variable (noise 1).

noise2 Synthetic spatially-correlated variable (noise 2).

dist Distance variable (spatially non-stationary example).

geometry Point geometry column (NAD83 / EPSG:4269).

Source

IPUMS NHGIS, University of Minnesota, https://www.nhgis.org.

XeeXhC *Spatial Variance Component (X'ee'X)*

Description

Spatial Variance Component (X'ee'X)

Usage

```
XeeXhC(M, cutoff, X, e, n1, k, kernel = "bartlett", dist_fn = "Haversine")
```

Arguments

M	matrix of locations (n x 2, latitude and longitude)
cutoff	the distance cutoff (bandwidth) in km
X	model matrix (n x k)
e	vector of residuals (length n)
n1	number of observations
k	number of regressors
kernel	(string) kernel function (default is bartlett-triangular)
dist_fn	(string) distance function (Haversine)

Value

A k x k matrix representing the spatial X'ee'X component.

XeeXhC_Lg *Spatial Variance Component for Large Datasets*

Description

Memory-efficient variant that avoids constructing the full n x n distance matrix.

Usage

```
XeeXhC_Lg(M, cutoff, X, e, n1, k, kernel = "bartlett", dist_fn = "Haversine")
```

Arguments

M	matrix of locations (n x 2, latitude and longitude)
cutoff	the distance cutoff (bandwidth) in km
X	model matrix (n x k)
e	vector of residuals (length n)
n1	number of observations
k	number of regressors
kernel	(string) kernel function (default is bartlett-triangular)
dist_fn	(string) distance function (Haversine)

Value

A k x k matrix representing the spatial X'ee'X component.

Index

∗ **datasets**
    US_counties_centroids, 15

Bal_XeeXhC, 2

compute_conley_lfe, 3
conley_SE, 4
conley_SE(), *3*, *12, 13*
coords_as_columns, 6
covgm_range, 7

DistMat, 8

extract_corr_range, 8
extract_corr_range(), *7*

fixest::feols(), *7*

ggplot2::ggplot(), *7*, *12*
gravity_centroid, 9
grid_FE, 10
gstat::variogram(), *7–9*

inverseu_plot_conleyrange, 11

lfe::felm(), *3, 4*, *13*
lm_sac, 13

modelsummary::modelsummary(), *13*

ncf::correlog(), *8, 9*, *13*

parallel::mclapply(), *5*

sf::st_centroid(), *9*
sf::st_coordinates(), *6*
sf::st_intersection(), *11*
sf::st_make_grid(), *11*
stats::lm(), *13*

tibble::tibble(), *6*
TimeDist, 14

US_counties_centroids, 15

XeeXhC, 16
XeeXhC_Lg, 16