Package: slxr (via r-universe)

May 27, 2026

Type Package

Title Spatial-X (SLX) Models for Applied Researchers

Version 0.1.1

Description Tools for estimating, interpreting, and visualizing Spatial-X (SLX) regression models. Provides a formula-based interface with first-class support for variable-specific weights matrices, higher-order spatial lags, temporally-lagged spatial variables (TSLS), and tidy effects decomposition (direct, indirect, total). Designed to lower the barrier to SLX modeling for applied researchers who already work with 'sf' and 'lm'-style formulas. Methods follow Wimpy, Whitten, and Williams (2021) <doi:10.1086/710089>.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Depends R (>= 4.1.0)

Imports spdep, sf, Matrix, tibble, stats, generics, rlang

Suggests ggplot2, modelsummary, broom, knitr, rmarkdown, testthat (>= 3.0.0), dplyr, tidyr, haven

Config/testthat/edition 3

VignetteBuilder knitr

URL https://github.com/cwimpy/slxr

BugReports https://github.com/cwimpy/slxr/issues

Config/pak/sysreqs libabsl-dev cmake libgdal-dev gdal-bin libgeos-dev libssl-dev libproj-dev libsqlite3-dev libudunits2-dev

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

Date/Publication 2026-04-27 18:30:33 UTC

RemoteUrl https://github.com/cwimpy/slxr

RemoteRef HEAD

RemoteSha 53f5f17acf70d96b07bc6cceac4f87489afc43e3

Contents

`defense_burden` *Defense burden data, 1995 cross-section*

Description

A 179-country cross-section of defense spending and conflict indicators for 1995, drawn from the replication archive of Wimpy, Whitten, and Williams (2021). Three row-standardized spatial weights matrices connect the units through different channels: contiguity, alliance, and defense pact.

Usage

```
defense_burden
```

Format

A named list with four elements:

`data` A tibble with 179 rows and 12 variables, arranged by Correlates of War country code (`ccode`). Variables:

- `ccode` - Correlates of War country code.
- `year` - Calendar year (1995).
- `ch_milex` - Change in military expenditures (outcome).
- `milex_tm1` - Lagged military expenditures.
- `log_pop_tm1` - Lagged log population.
- `civilwar_tm1` - Lagged civil war indicator.
- `total_wars_tm1` - Lagged count of interstate wars.

- `alliance_us` - Alliance with the United States.
- `ch_milex_us` - Change in U.S. military expenditures.
- `ch_milex_ussr` - Change in Soviet/Russian military expenditures.
- `region` - Integer region code (1-5).
- `region_name` - Factor: Europe, N Africa/Middle East, Africa, Asia/Oceania, Americas.

`W_contig` A 179 x 179 row-standardized sparse `dgCMatrix` encoding geographic contiguity.

`W_alliance` A 179 x 179 row-standardized sparse `dgCMatrix` encoding alliance ties.

`W_defense` A 179 x 179 row-standardized sparse `dgCMatrix` encoding mutual defense pacts.

Details

This cross-section is intended for pedagogical demonstration of variable-specific SLX specifications. The original paper estimates a pooled panel SLX across 1950-2008 with year-specific weights matrices; that full-panel replication requires block-diagonal W support, which is planned for `slxr` v0.2.

Source

Wimpy, Whitten, and Williams (2021) replication archive, Journal of Politics Dataverse. doi:10.1086/710089

References

Wimpy, C., Whitten, G. D., & Williams, L. K. (2021). X Marks the Spot: Unlocking the Treasure of Spatial-X Models. *Journal of Politics*, 83(2), 722-739.

Examples

```
data(defense_burden)
dim(defense_burden$data)
dim(defense_burden$W_contig)
```

`defense_burden_panel` *Defense burden panel, 1951-2008*

Description

The full country-year panel underlying Wimpy, Whitten, and Williams (2021) Table 3, Model 3. Includes a tibble of 7,661 country-year observations and three named lists of row-standardized sparse weights matrices, one matrix per year, encoding contiguity, alliance, and defense-pact connections.

Usage

```
defense_burden_panel
```

Format

A named list:

`data` A tibble with 7,661 rows and the same 12 columns as defense_burden`$data`, but spanning 1951-2008.

`W_contig` Named list of 58 sparse matrices, one per year, keyed by year as a string. Each matrix is row-standardized and contains the countries observed in that year.

`W_alliance` As above, for alliance ties.

`W_defense` As above, for mutual defense pacts.

Details

Sample excludes observations with missing covariates. Panel is unbalanced: between 63 and 187 countries per year.

Source

Wimpy, Whitten, and Williams (2021) replication archive, Journal of Politics Dataverse. doi:10.1086/710089

See Also

defense_burden for the 1995 cross-section.

Examples

```
data(defense_burden_panel)
names(defense_burden_panel$W_contig)[1:5]
dim(defense_burden_panel$W_contig[["1990"]])
```

`slx` *Fit a Spatial-X (SLX) model*

Description

Estimates a Spatial-X regression of the form

$$y = X\beta + WX\theta + \varepsilon,$$

where `W` is a spatial weights matrix and `X` includes both the directly-entering regressors and a user-specified subset that is spatially lagged. SLX estimation is OLS on the augmented design matrix, which makes estimation, interpretation, and effects decomposition much simpler than for SAR-family models (Wimpy, Whitten, and Williams 2021).

Usage

```
slx(
  formula,
  data,
  W = NULL,
  lag = NULL,
  order = 1L,
  spatial = NULL,
  time_lag = 0L,
  id = NULL,
  time = NULL,
  na.action = stats::na.omit
)
```

Arguments

formula	A standard model formula, e.g. y ~ x1 + x2 + x3.
data	A data frame (or sf object) whose rows correspond, in order, to the rows/columns of W (cross-sectional mode) or contain id and time columns (panel mode).
W	An slx_W object produced by slx_weights(), a numeric or sparse matrix, or - in panel mode - a named list of such objects keyed by time value. Required when spatial is not supplied.
lag	Character vector of variable names from formula that should be spatially lagged. If NULL (default) and W is supplied, all right-hand-side variables are lagged.
order	Integer vector giving the orders of W to include for each lagged variable (e.g. 1:2 adds both Wx and W^2 x). Default 1.
spatial	Optional named list for variable-specific weights matrices. Names must match variables in formula. Each element is either an slx_W, a list of slx_Ws (multiple channels for one variable), a time-keyed list of slx_Ws (single channel, time-varying), or a named list of channels whose values are themselves single or time-keyed slx_Ws.
time_lag	Integer, number of time periods to lag the spatial terms (TSLS). Requires id and time. Default 0.
id, time	Column names identifying the panel unit and period. Required for panel mode or TSLS. time must be numeric.
na.action	How to handle missing values. Defaults to stats::na.omit in cross-sectional mode. In panel mode missing values on formula variables are dropped; id and time must never be missing.

Value

An object of class slx with elements:

fit The underlying lm object.

formula The expanded model formula.

call The original call.

`W` The weights matrix (or list thereof) used.

`lag_terms` A data frame mapping spatial-lag terms to their source variable, W matrix, order, and time lag.

`data` The (possibly augmented) model frame.

`panel` Logical flag.

Cross-sectional and panel modes

When `id` and `time` are `NULL`, `slx()` treats the data as a single cross-section: rows of `data` must correspond, in order, to the rows and columns of `W`. When both `id` and `time` are supplied, `slx()` enters panel mode and performs block-wise spatial lagging period by period. In panel mode the weights matrices must have `dimnames` matching the unit identifiers in `data[[id]]`; the same units need not appear in every period (unbalanced panels are supported).

Time-varying weights

In panel mode, a `W` argument can be either (a) a single `slx_W` object applied to every period or (b) a named list of `slx_W` objects keyed by the stringified `time` value (e.g. `list("1990" = W_90, "1991" = W_91)`). The same options apply to any entry inside `spatial`.

Temporally-lagged spatial lag (TSLS)

The `time_lag` argument implements equation 7 of Wimpy, Whitten, and Williams (2021): it lags every constructed spatial term `W_t x` by `time_lag` periods within each unit so that the regressor becomes `W_t x_{t - time_lag}`. The first `time_lag` periods per unit drop out, mirroring any temporally-lagged variable.

References

Wimpy, C., Whitten, G. D., & Williams, L. K. (2021). X Marks the Spot: Unlocking the Treasure of Spatial-X Models. *Journal of Politics*, 83(2), 722-739.

Vega, S. H., & Elhorst, J. P. (2015). The SLX Model. *Journal of Regional Science*, 55(3), 339-363.

Examples

```
data(defense_burden)
W_c <- slx_weights(style = "custom", matrix = defense_burden$W_contig,
                   row_standardize = FALSE)
fit <- slx(ch_milex ~ milex_tm1 + log_pop_tm1 + civilwar_tm1,
           data = defense_burden$data, W = W_c, lag = "civilwar_tm1")
slx_effects(fit)
```

slx-tidiers *Tidy and glance methods for SLX models*

Description

These methods make `slx` objects compatible with the `broom` and `modelsummary` ecosystems.

Usage

```
## S3 method for class 'slx'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)

## S3 method for class 'slx'
glance(x, ...)
```

Arguments

`x`	An `slx` object.
`conf.int`	Logical; include confidence intervals?
`conf.level`	Confidence level.
`...`	Unused.

Value

`tidy.slx()` returns a `tibble::tibble()` with one row per model coefficient (both direct and spatial-lag terms) and columns `term`, `estimate`, `std.error`, `statistic`, and `p.value`. When `conf.int = TRUE`, `conf.low` and `conf.high` columns are added.

`glance.slx()` returns a one-row `tibble::tibble()` summarizing the overall fit, with columns `r.squared`, `adj.r.squared`, `sigma`, `statistic` (F statistic), `df`, `df.residual`, `nobs`, and `n_lag_terms` (the number of spatial-lag regressors in the model).

Examples

```
data(defense_burden)
W <- slx_weights(style = "custom", matrix = defense_burden$W_contig,
                 row_standardize = FALSE)
fit <- slx(ch_milex ~ milex_tm1 + civilwar_tm1,
           data = defense_burden$data, W = W, lag = "civilwar_tm1")
tidy(fit)
glance(fit)
```

slx_compare *Compare OLS, SLX, and other lm-like models side by side*

Description

Builds a tidy tibble of fit statistics (observations, coefficient count, R-squared, adjusted R-squared, residual standard error, AIC, BIC) for any combination of `lm` and `slx` objects. If a weights matrix is supplied, also reports Moran's I on each model's residuals - the standard diagnostic for spatial autocorrelation left unexplained by the fitted model.

Usage

```
slx_compare(..., W = NULL)
```

Arguments

`...`	Named `lm` or `slx` models. Names are used as the model labels in the output.
`W`	Optional weights matrix to use for Moran's I on residuals. Accepts an `slx_W`, a `listw` object from `spdep`, or `NULL` to skip the test. In panel mode, supply a list keyed by time value.

Value

A tibble with one row per model.

Examples

```
data(defense_burden)
W <- slx_weights(style = "custom", matrix = defense_burden$W_contig,
                 row_standardize = FALSE)
ols <- lm(ch_milex ~ milex_tm1 + civilwar_tm1,
          data = defense_burden$data)
slx_fit <- slx(ch_milex ~ milex_tm1 + civilwar_tm1,
               data = defense_burden$data, W = W,
               lag = "civilwar_tm1")
slx_compare(OLS = ols, SLX = slx_fit, W = W)
```

slx_effects *Direct, indirect, and total effects from an SLX model*

Description

For an SLX model the effects decomposition is trivial — no matrix inversion, no simulation. For each variable x that enters both directly and as a spatial lag, the direct effect is its OLS coefficient $\hat{\beta}$ and the indirect effect at order k is $\hat{\theta}_k$. Standard errors come straight from `vcov()`.

Usage

```
slx_effects(object, by_order = FALSE, conf.level = 0.95)
```

Arguments

object	An slx object returned by slx().
by_order	Logical; if TRUE, report indirect effects separately by order of W. Default FALSE sums across orders for a single indirect effect per variable–W combination.
conf.level	Confidence level for reported intervals. Default 0.95.

Value

A tibble with columns variable, w_name, order (when by_order = TRUE), type (direct/indirect/total), estimate, std.error, conf.low, conf.high, p.value.

Examples

```
data(defense_burden)
W <- slx_weights(style = "custom", matrix = defense_burden$W_contig,
                 row_standardize = FALSE)
fit <- slx(ch_milex ~ milex_tm1 + civilwar_tm1,
           data = defense_burden$data, W = W, lag = "civilwar_tm1")
slx_effects(fit)
```

---

slx_plot_decay *Plot how indirect effects decay across orders of W*

Description

For SLX models fit with higher-order lags (order = 1:k), shows the size of the indirect effect at each order, with confidence intervals.

Usage

```
slx_plot_decay(fit, variables = NULL, conf.level = 0.95)
```

Arguments

fit	An slx model fit with higher-order W terms.
variables	Optional character vector restricting to specific lagged variables. Default plots all.
conf.level	Confidence level.

Value

A ggplot object.

`slx_plot_effects` *Coefficient plot of direct, indirect, and total effects*

Description

Produces a ggplot of the direct, indirect, and total effects from an SLX model, with 95% confidence intervals. For models with variable-specific weights matrices, effects are faceted by `w_name` so spillover patterns from each matrix are visible side-by-side.

Usage

```
slx_plot_effects(
  fit,
  types = c("direct", "indirect", "total"),
  conf.level = 0.95,
  by_order = FALSE,
  ...
)
```

Arguments

`fit`	An `slx` object returned by `slx()`.
`types`	Character vector of effect types to include. Any subset of `c("direct", "indirect", "total")`. Default shows all three.
`conf.level`	Confidence level for the intervals. Default 0.95.
`by_order`	Logical; if `TRUE`, break indirect effects out by order of W. Default `FALSE`.
`...`	Passed to `slx_effects()`.

Value

A `ggplot` object.

Examples

```
data(defense_burden)
W <- slx_weights(style = "custom", matrix = defense_burden$W_contig,
                 row_standardize = FALSE)
fit <- slx(ch_milex ~ milex_tm1 + civilwar_tm1,
           data = defense_burden$data, W = W, lag = "civilwar_tm1")
slx_plot_effects(fit)
```

`slx_plot_shock` *Counterfactual shock plot*

Description

Given a fitted SLX model, a choice of variable, and a target unit, returns a plot of the predicted change in the outcome across every unit in the sample under a unit shock to that variable in the target unit.

Usage

```
slx_plot_shock(fit, variable, unit, magnitude = 1, geom = NULL, top_n = 15)
```

Arguments

`fit`	A cross-sectional `slx` model. Panel shocks are planned for a future release.
`variable`	Character, the name of a spatially-lagged regressor in `fit` to shock.
`unit`	Integer row index (or character id, if the weights matrix has dimnames matching something in `fit$data`) of the unit receiving the shock.
`magnitude`	Numeric, shock size. Default `1`.
`geom`	Optional `sf` object with `nrow(geom) == fit$n` aligned to `fit$data`. If supplied, the function returns a map.
`top_n`	For the non-map plot, how many non-zero indirect effects to show. Default `15`.

Details

For an SLX model at first order with channels indexed by `c`, the predicted change at unit `j` from a shock of size `magnitude` to variable `x` in unit `i` is

$$\text{magnitude} \left(\beta \mathbb{1}\{j = i\} + \sum_c \theta_c \, W_c[j, i] \right).$$

Higher-order lags add additional `theta_{c,k} (W_c^k)[j, i]` terms. No simulation is required: the shock effect is a single column of the spatial multiplier.

If an `sf` object with matching row count is supplied via `geom`, the result is drawn as a choropleth. Otherwise a horizontal bar of the largest effects is returned.

Value

A `ggplot` object.

Examples

```
data(defense_burden)
W_c <- slx_weights(style = "custom", matrix = defense_burden$W_contig,
                   row_standardize = FALSE)
fit <- slx(ch_milex ~ milex_tm1 + civilwar_tm1,
           data = defense_burden$data, W = W_c,
           lag = "civilwar_tm1")
slx_plot_shock(fit, variable = "civilwar_tm1", unit = 1)
```

---

slx_plot_W *Heatmap of a spatial weights matrix*

Description

A quick visual check on the structure of a weights matrix. For large n the heatmap becomes noisy; use the max_n argument to subsample.

Usage

```
slx_plot_W(W, max_n = NULL, labels = NULL)
```

Arguments

W	An slx_W object.
max_n	Optional integer; if nrow(W) > max_n, a random sample of rows/columns is plotted. Default NULL (plot everything).
labels	Optional character vector of row/column labels.

Value

A ggplot object.

---

slx_weights *Construct a spatial weights matrix*

Description

A thin, opinionated wrapper around common spdep weights constructors. Returns a standardized slx_W object that carries both the sparse matrix and the listw form used by downstream routines.

Usage

```
slx_weights(
  x = NULL,
  style = c("contiguity", "rook", "knn", "distance", "custom"),
  k = 5,
  threshold = NULL,
  row_standardize = TRUE,
  matrix = NULL,
  ...
)
```

Arguments

x	An sf object (for "contiguity", "knn", "distance"), a matrix of coordinates, or a raw numeric/sparse matrix (for "custom").
style	Weights style. One of "contiguity" (queen), "rook", "knn", "distance", or "custom".
k	Number of neighbors for style = "knn".
threshold	Distance threshold for style = "distance" (units of the coordinate system).
row_standardize	Logical; row-standardize the matrix? Default TRUE. The paper notes row-standardization is a theoretical choice; set to FALSE when connection count should itself carry weight.
matrix	A numeric or sparse Matrix for style = "custom".
...	Passed to the underlying spdep constructor.

Value

An object of class slx_W with elements W (sparse matrix), listw (spdep listw object), style, and row_standardized.

Examples

```
# Custom weights matrix from bundled data
data(defense_burden)
W <- slx_weights(style = "custom", matrix = defense_burden$W_contig,
                 row_standardize = FALSE)
W


# Contiguity weights from an sf polygon layer
if (requireNamespace("sf", quietly = TRUE) &&
    requireNamespace("spdep", quietly = TRUE)) {
  nc <- sf::st_read(system.file("shape/nc.shp", package = "sf"),
                    quiet = TRUE)
  W_nc <- slx_weights(nc, style = "contiguity")
  W_nc
}
```

Index