

Package: griddy (via r-universe)

May 31, 2026

Title Geospatial Distribution Dynamics for Simple Features

Version 0.1.0

Description Tools for exploratory geospatial distribution dynamics with 'sf' objects and tidy data. Provides pooled and time-specific classification of longitudinal spatial values, classic discrete Markov transition matrices, spatial Markov matrices conditioned on spatial-lag classes, endpoint and adjacent rank mobility summaries, and 'ggplot2' visualizations. Methods follow Rey (2001) [doi:10.1111/j.1538-4632.2001.tb00444.x](https://doi.org/10.1111/j.1538-4632.2001.tb00444.x) and Rey et al. (2016) [doi:10.1007/s10109-016-0234-x](https://doi.org/10.1007/s10109-016-0234-x); design is inspired by the Python 'PySAL' 'gidddy' package <https://pysal.org/gidddy/>.

URL <https://github.com/dshkol/griddy>, <https://dshkol.github.io/griddy/>

BugReports <https://github.com/dshkol/griddy/issues>

License MIT + file LICENSE

Encoding UTF-8

Language en-US

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Depends R (>= 4.2)

Imports dplyr, ggplot2, rlang, scales, sf, spdep, tibble, tidyr

Suggests cancensus, knitr, microbenchmark, pkgdown, rmarkdown, sfdep, spData, testthat (>= 3.0.0), tidycensus

VignetteBuilder knitr

Config/testthat/edition 3

LazyData true

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

Repository <https://dshkol.r-universe.dev>

Date/Publication 2026-05-27 16:32:44 UTC

RemoteUrl <https://github.com/dshkol/griddy>

RemoteRef HEAD

RemoteSha e9a36d2930eeffda04abd8f41adf33737ad29845

Contents

class_intervals	2
classify_dynamics	3
lag_intervals	4
markov_dynamics	5
plot_rank_mobility	5
plot_spatial_markov	6
plot_transition_matrix	7
rank_mobility	7
spatial_markov	8
steady_state	10
transition_matrix	10
usjoin	11

Index **13**

class_intervals	<i>Class Intervals Used By A griddy Object</i>
-----------------	--

Description

Class Intervals Used By A griddy Object

Usage

```
class_intervals(x, ...)
```

Arguments

x	A griddy result object.
...	Reserved for future methods.

Value

A tibble of class intervals when available.

Examples

```
panel <- data.frame(id = rep(1:3, each = 2), year = rep(2020:2021, 3), value = 1:6)
classes <- classify_dynamics(panel, id, year, value, k = 3)
class_intervals(classes)
```

classify_dynamics *Classify Longitudinal Values For Distribution Dynamics*

Description

Converts numeric values in a long panel into ordered classes suitable for transition analysis.

Usage

```
classify_dynamics(  
  data,  
  id,  
  time,  
  value,  
  method = c("pooled_quantile", "time_quantile", "fixed", "existing"),  
  k = 5,  
  breaks = NULL,  
  labels = NULL  
)
```

Arguments

data	A data frame or sf object in long format.
id, time, value	Columns identifying spatial unit, time, and value.
method	Classification method. "pooled_quantile" uses one set of quantile breaks across all periods. "time_quantile" uses period-specific ranks. "fixed" uses user-supplied breaks. "existing" treats value as an already classified state.
k	Number of quantile classes.
breaks	Breaks for method = "fixed".
labels	Optional class labels.

Value

A data frame or sf object with a class column and class `grd_classes`.

Examples

```
panel <- data.frame(  
  id = rep(letters[1:4], each = 3),  
  year = rep(2020:2022, times = 4),  
  value = c(8, 9, 11, 10, 12, 13, 15, 14, 16, 20, 22, 25)  
)  
  
classes <- classify_dynamics(panel, id, year, value, k = 3)  
classes  
class_intervals(classes)
```

lag_intervals	<i>Spatial Lag Intervals Used By A Spatial Markov Object</i>
---------------	--

Description

Spatial Lag Intervals Used By A Spatial Markov Object

Usage

```
lag_intervals(x)
```

Arguments

x A `grd_spatial_markov` object.

Value

A tibble of spatial-lag class intervals.

Examples

```
panel <- data.frame(
  id = rep(1:4, each = 2),
  year = rep(2020:2021, times = 4),
  value = c(1, 2, 2, 3, 4, 3, 5, 6)
)
grid <- sf::st_sf(
  id = 1:4,
  geometry = sf::st_make_grid(
    sf::st_bbox(c(xmin = 0, ymin = 0, xmax = 2, ymax = 2)),
    n = c(2, 2)
  )
) |>
dplyr::mutate(
  nb = sfdep::st_contiguity(geometry),
  wt = sfdep::st_weights(nb)
)
spatial <- spatial_markov(panel, id, year, value, geometry = grid, k = 2)
lag_intervals(spatial)
```

markov_dynamics	<i>Estimate Classic Markov Transition Dynamics</i>
-----------------	--

Description

Estimates transition counts and probabilities between adjacent periods for a classified long panel.

Usage

```
markov_dynamics(classes, id, time, state = class)
```

Arguments

`classes` A classified data frame from `classify_dynamics()` or any data frame with ID, time, and state columns.

`id, time, state` Columns identifying spatial unit, time, and state.

Value

A `grd_markov` object.

Examples

```
panel <- data.frame(
  id = rep(letters[1:4], each = 3),
  year = rep(2020:2022, times = 4),
  value = c(8, 9, 11, 10, 12, 13, 15, 14, 16, 20, 22, 25)
)

classes <- classify_dynamics(panel, id, year, value, k = 3)
markov <- markov_dynamics(classes, id, year, class)

markov
transition_matrix(markov)
steady_state(markov)
```

plot_rank_mobility	<i>Plot Rank Mobility</i>
--------------------	---------------------------

Description

Plot Rank Mobility

Usage

```
plot_rank_mobility(x)
```

Arguments

x A `grd_rank_mobility` object returned by `rank_mobility()`.

Value

A ggplot object.

plot_spatial_markov *Plot Spatial Markov Matrices*

Description

Plot Spatial Markov Matrices

Usage

```
plot_spatial_markov(x)
```

Arguments

x A `grd_spatial_markov` object.

Value

A ggplot object.

Examples

```
panel <- data.frame(
  id = rep(1:4, each = 2),
  year = rep(2020:2021, times = 4),
  value = c(1, 2, 2, 3, 4, 3, 5, 6)
)
grid <- sf::st_sf(
  id = 1:4,
  geometry = sf::st_make_grid(
    sf::st_bbox(c(xmin = 0, ymin = 0, xmax = 2, ymax = 2)),
    n = c(2, 2)
  )
) |>
dplyr::mutate(
  nb = sfdep::st_contiguity(geometry),
  wt = sfdep::st_weights(nb)
)
spatial <- spatial_markov(panel, id, year, value, geometry = grid, k = 2)
plot_spatial_markov(spatial)
```

plot_transition_matrix
Plot A Transition Matrix

Description

Plot A Transition Matrix

Usage

```
plot_transition_matrix(x)
```

Arguments

x A `grd_markov` object.

Value

A `ggplot` object.

Examples

```
panel <- data.frame(id = rep(1:4, each = 2), year = rep(2020:2021, 4), value = 1:8)
markov <- panel |>
  classify_dynamics(id, year, value, k = 2) |>
  markov_dynamics(id, year, class)

plot_transition_matrix(markov)
```

rank_mobility *Compute Rank Mobility*

Description

Computes simple rank changes for map-ready exploratory analysis.

Usage

```
rank_mobility(
  data,
  id,
  time,
  value,
  compare = c("endpoint", "adjacent"),
  descending = TRUE
)
```

Arguments

data	A long data frame or sf object.
id, time, value	Columns identifying spatial unit, time, and value.
compare	"endpoint" compares first and last observed periods per unit. "adjacent" compares adjacent periods.
descending	If TRUE, larger values receive better ranks.

Value

A `grd_rank_mobility` data frame or sf object.

Examples

```
panel <- data.frame(
  id = rep(letters[1:4], each = 3),
  year = rep(2020:2022, times = 4),
  value = c(8, 9, 11, 10, 12, 13, 15, 14, 16, 20, 22, 25)
)

rank_mobility(panel, id, year, value)
```

spatial_markov

Estimate Spatial Markov Transition Dynamics

Description

Estimates transition matrices conditioned on the class of each unit's spatial lag at the start of the transition period.

Usage

```
spatial_markov(
  data,
  id,
  time,
  value,
  geometry = NULL,
  listw = NULL,
  nb = NULL,
  k = 5,
  lag_k = k,
  class_method = c("pooled_quantile", "time_quantile", "fixed"),
  zero.policy = TRUE
)
```

Arguments

<code>data</code>	A long data frame or sf object.
<code>id, time, value</code>	Columns identifying spatial unit, time, and value.
<code>geometry</code>	An sf tibble with one row per spatial unit (a single time slice's geography), carrying <code>nb</code> and <code>wt</code> list-columns produced by <code>sfdep::st_contiguity()</code> and <code>sfdep::st_weights()</code> . The preferred input.
<code>listw</code>	A row-standardized spdep listw object. Accepted for compatibility with prior workflows; use <code>geometry</code> for new code.
<code>nb</code>	A spdep neighbor list, used only when both <code>geometry</code> and <code>listw</code> are NULL. Converted with <code>spdep::nb2listw(style = "W")</code> .
<code>k</code>	Number of value classes.
<code>lag_k</code>	Number of spatial-lag classes.
<code>class_method</code>	Value classification method passed to <code>classify_dynamics()</code> .
<code>zero.policy</code>	Passed to spdep lag/weight helpers.

Value

A `grd_spatial_markov` object.

Examples

```

panel <- data.frame(
  id = rep(1:4, each = 3),
  year = rep(2020:2022, times = 4),
  value = c(1, 2, 3, 2, 3, 4, 4, 3, 5, 5, 6, 7)
)

grid <- sf::st_sf(
  id = 1:4,
  geometry = sf::st_make_grid(
    sf::st_bbox(c(xmin = 0, ymin = 0, xmax = 2, ymax = 2)),
    n = c(2, 2)
  )
) |>
dplyr::mutate(
  nb = sfdep::st_contiguity(geometry),
  wt = sfdep::st_weights(nb)
)

spatial <- spatial_markov(panel, id, year, value, geometry = grid, k = 2)

spatial
lag_intervals(spatial)
transition_matrix(spatial, "count", lag_class = "Q1")

```

steady_state	<i>Estimate Stationary Distribution</i>
--------------	---

Description

Estimate Stationary Distribution

Usage

```
steady_state(x)
```

Arguments

x A `grd_markov` object or transition probability matrix.

Value

A numeric vector.

Examples

```
prob <- matrix(c(0.8, 0.2, 0.3, 0.7), nrow = 2, byrow = TRUE)
steady_state(prob)
```

transition_matrix	<i>Extract A Transition Matrix</i>
-------------------	------------------------------------

Description

Extract A Transition Matrix

Usage

```
transition_matrix(x, type = c("probability", "count"), lag_class = NULL)
```

Arguments

x A `grd_markov` or `grd_spatial_markov` object.
 type "probability" or "count".
 lag_class Optional lag class for `grd_spatial_markov`.

Value

A matrix.

Examples

```
panel <- data.frame(
  id = rep(letters[1:4], each = 3),
  year = rep(2020:2022, times = 4),
  value = c(8, 9, 11, 10, 12, 13, 15, 14, 16, 20, 22, 25)
)
markov <- panel |>
  classify_dynamics(id, year, value, k = 3) |>
  markov_dynamics(id, year, class)

transition_matrix(markov, "count")
```

usjoin

US State Per-Capita Personal Income, 1929-2009

Description

Per-capita personal income for the 48 contiguous US states in nominal dollars. Mirrors the canonical usjoin panel used in PySAL giddy and the spatial Markov literature, so examples and validation in this package are directly comparable to that reference work.

Usage

```
usjoin
```

Format

A tibble with 3,888 rows (48 states x 81 years) and 4 columns:

name State name.

state_fips State FIPS code, integer.

year Year, integer 1929 to 2009.

income Nominal per-capita personal income, integer USD.

Source

PySAL libpysal examples/us_income/usjoin.csv. The original series is constructed from US Bureau of Economic Analysis state personal income tables. See <https://github.com/pysal/libpysal>.

References

Rey, S. J. (2001). Spatial empirics for economic growth and convergence. *Geographical Analysis*, 33(3), 195-214.

Rey, S. J., Kang, W., & Wolf, L. (2016). The properties of tests for spatial effects in discrete Markov chain models of regional income distribution dynamics. *Journal of Geographical Systems*, 18(4), 377-398.

Examples

```
data(usjoin)
head(usjoin)
range(usjoin$year)
```

Index

* datasets

usjoin, 11

class_intervals, 2

classify_dynamics, 3

classify_dynamics(), 5, 9

lag_intervals, 4

markov_dynamics, 5

plot_rank_mobility, 5

plot_spatial_markov, 6

plot_transition_matrix, 7

rank_mobility, 7

rank_mobility(), 6

spatial_markov, 8

steady_state, 10

transition_matrix, 10

usjoin, 11