

Package: BorderEffect (via r-universe)

October 5, 2026

Title Detection of Edge Effects in Field Trials via Besag-Kempton Competition

Version 0.1.0

Description Estimates and evaluates the intraspecific competition coefficient associated with the edge (border) effect in agricultural field trials, using the Besag-Kempton autoregressive model and a least-squares estimator following Darghan, Rivera, Gonzalez and Castellanos (2022) [doi:10.47280/RevFacAgron\(LUZ\).v39.n1.18](https://doi.org/10.47280/RevFacAgron(LUZ).v39.n1.18). Provides field-layout generation, spatial weight matrices, a formula interface for arbitrary two-way designs, a simulation-based decision procedure for presence or absence of the edge effect, Moran's I diagnostics, and a 'shiny' application.

License MIT + file LICENSE

Encoding UTF-8

Language en-US

Depends R (>= 4.1.0)

Imports stats, graphics, grDevices

Suggests shiny, ape, testthat (>= 3.0.0), knitr, rmarkdown

Config/testthat/edition 3

VignetteBuilder knitr

Roxygen list(markdown = TRUE)

LazyData true

Config/roxygen2/version 8.0.0

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

Date/Publication 2026-08-05 12:52:37 UTC

RemoteUrl <https://github.com/segamboam/bordereffect>

RemoteRef HEAD

RemoteSha 5809392055bd6c624bead7cfc00216be778c6c19

Contents

border_effect	2
border_indicators	3
border_test	4
border_weights	5
field_layout	5
field_trial	7
kappa_hat	7
moran_border	8
run_border_app	9

Index	10
-------	----

border_effect	<i>Estimate the edge-effect competition coefficient</i>
---------------	---

Description

Fits the least-squares Besag-Kempton competition coefficient κ for a two-way (treatment + block) field trial. The projection onto the design’s column space is obtained from `model.matrix(formula, data)` via `qr.resid()`, so any design expressible as an R formula is supported.

Usage

```
border_effect(formula, data, coords, weights = NULL)
```

Arguments

formula	A model formula, e.g. <code>response ~ trt + blk</code> .
data	A <code>data.frame</code> containing the response and the design factors.
coords	A <code>field_layout</code> , or a <code>data.frame</code> /matrix of x/y coordinates aligned row-wise with data.
weights	Optional precomputed weight matrix (see <code>border_weights()</code>); built from <code>coords</code> when <code>NULL</code> .

Value

An object of class `border_effect`: a list with `kappa`, `call`, `formula`, `terms`, `weights`, `residuals`, `fitted`, `coords`, `data`.

Note

`field_layout()` reserves the column names `x` and `y` for plot coordinates. When your data is a `field_layout` (or shares its coordinates via `coords`), give the response a different name (e.g. `resp`, `yield`, `response`) so it does not overwrite the `y` coordinate.

References

Darghan, Rivera, Gonzalez & Castellanos (2022). [doi:10.47280/RevFacAgron\(LUZ\).v39.n1.18](https://doi.org/10.47280/RevFacAgron(LUZ).v39.n1.18)

Examples

```
f1 <- field_layout(nx = 12, ny = 8, ntrt = 3, nblk = 2, seed = 3000)
set.seed(1); f1$resp <- rnorm(96, 2.21, 0.06)
be <- border_effect(resp ~ trt + blk, data = f1, coords = f1)
be
```

border_indicators	<i>Edge (border) indicator vector from plot coordinates</i>
-------------------	---

Description

Given the coordinates of the units in a rectangular field trial, returns a 0/1 indicator of the units belonging to a chosen border pattern. The outer ring ("outer1") and the second ring ("outer2") correspond to the sets S_1 and S_2 of the paper; the side and multi-side patterns reproduce the partial-border scenarios.

Usage

```
border_indicators(
  coords,
  pattern = c("outer1", "outer2", "lower", "left", "upper", "right", "lower_left",
    "lower_upper", "lower_right", "left_upper", "left_right", "upper_right",
    "lower_left_upper", "lower_left_right", "lower_upper_right", "left_upper_right",
    "central_h")
)
```

Arguments

coords	A field_layout, a data.frame with x/y, or a numeric matrix.
pattern	One of "outer1", "outer2", "lower", "left", "upper", "right", "lower_left", "lower_upper", "lower_right", "left_upper", "left_right", "upper_right", "lower_left_upper", "lower_left_right", "lower_upper_right", "left_upper_right", "central_h".

Value

An integer vector of 0s and 1s, length nrow(coords).

Note

The side and ring patterns are defined from the global minimum/maximum x and y coordinates (matching the original method), so they assume an aligned rectangular grid. With arrangement = "triangular" in `field_layout()`, the half-shifted even rows are not fully captured by the side patterns (e.g. "left"/"right" only flag the exact-min/max-x column, which excludes the shifted units in even rows). Users of triangular layouts should interpret side/ring patterns accordingly.

References

Darghan, Rivera, Gonzalez & Castellanos (2022). [doi:10.47280/RevFacAgron\(LUZ\).v39.n1.18](https://doi.org/10.47280/RevFacAgron(LUZ).v39.n1.18)

Examples

```
f1 <- field_layout(nx = 5, ny = 4, ntrt = 2, nblk = 2)
table(border_indicators(f1, "outer1"))
```

border_test

Simulation-based decision for the presence of the edge effect

Description

Implements the paper's decision procedure: an indicator model response $\sim S1$ (or response $\sim S1 + S2$ for the double outer edge) is fitted, new responses are generated with `simulate()`, and κ is recomputed for each simulated response using the two-way design. The resulting distribution yields a confidence interval; if it contains zero the edge effect is judged absent, otherwise present.

Usage

```
border_test(object, pattern = "outer1", nsim = 500, level = 0.95, seed = NULL)
```

Arguments

object	A border_effect object.
pattern	Border pattern to test (see <code>border_indicators()</code>). "outer2" uses the double-edge model $\sim S1 + S2$; all others use $\sim S1$.
nsim	Number of simulated responses (default 500).
level	Confidence level for the interval (default 0.95).
seed	Optional seed; RNG state is restored on exit.

Value

An object of class border_test: a list with kappa_obs, kappa_sim, ci, level, contains_zero, decision, pattern, nsim.

References

Darghan, Rivera, Gonzalez & Castellanos (2022). [doi:10.47280/RevFacAgron\(LUZ\).v39.n1.18](https://doi.org/10.47280/RevFacAgron(LUZ).v39.n1.18)

Examples

```
f1 <- field_layout(nx = 12, ny = 8, ntrt = 3, nblk = 2, seed = 3000)
set.seed(1); f1$resp <- rnorm(96, 2.21, 0.06)
be <- border_effect(resp ~ trt + blk, data = f1, coords = f1)
border_test(be, pattern = "outer1", nsim = 200, seed = 42)
```

border_weights	<i>Spatial weight matrix from plot coordinates</i>
----------------	--

Description

Builds an inverse-distance weight matrix W for the units of a field trial, as used by the Besag-Kempton competition model. The diagonal is zero (a unit does not compete with itself) and, by default, the matrix is standardized to sum to one.

Usage

```
border_weights(coords, method = c("idw"), power = 1, standardize = TRUE)
```

Arguments

coords	A field_layout, a data.frame with columns x and y, or a numeric matrix whose first two columns are the coordinates.
method	Weighting method. Currently only "idw" (inverse distance).
power	Positive exponent applied to the distance before inverting.
standardize	If TRUE (default) divide by the total sum so $\text{sum}(W) == 1$.

Value

A symmetric numeric matrix of class border_weights with zero diagonal.

References

Darghan, Rivera, Gonzalez & Castellanos (2022). doi:[10.47280/RevFacAgron\(LUZ\).v39.n1.18](https://doi.org/10.47280/RevFacAgron(LUZ).v39.n1.18)

Examples

```
f1 <- field_layout(nx = 4, ny = 4, ntrt = 2, nblk = 2)
W <- border_weights(f1)
sum(W)
```

field_layout	<i>Generate a rectangular field-trial layout with border indicators</i>
--------------	---

Description

Places $n_x * n_y$ experimental units on a rectangular grid (square or triangular / "tres bolillos" arrangement), randomly assigns treatments, blocks and replicates, and appends 0/1 indicator columns for every border pattern (see [border_indicators\(\)](#)).

Usage

```
field_layout(
  nx,
  ny,
  ntrt = 3,
  nblk = 2,
  arrangement = c("square", "triangular"),
  width = NULL,
  height = NULL,
  randomize = TRUE,
  seed = NULL
)

## S3 method for class 'field_layout'
plot(x, color_by = "trt", ...)
```

Arguments

<code>nx, ny</code>	Number of columns and rows of the grid.
<code>ntrt, nblk</code>	Number of treatments and blocks.
<code>arrangement</code>	"square" or "triangular" (even rows shifted by half a column spacing).
<code>width, height</code>	Physical extent of the grid. Defaults to <code>nx - 1</code> , <code>ny - 1</code> .
<code>randomize</code>	If TRUE (default) randomize the treatment/block assignment.
<code>seed</code>	Optional integer seed; RNG state is restored on exit.
<code>x</code>	A <code>field_layout</code> object.
<code>color_by</code>	Column used to color the points (default "trt").
<code>...</code>	Passed to <code>plot()</code> .

Value

A data.frame of class `field_layout` with columns `x`, `y`, `trt`, `blk`, `rep`, and one integer column per border pattern.

Note

With `arrangement = "triangular"`, the appended border-pattern columns (see `border_indicators()`) are still computed from the global min/max `x/y`, which assumes an aligned rectangular grid; the half-shifted even rows mean side patterns like "left"/"right" only capture the exact-min/max-`x` column. Interpret side/ring indicators with this in mind for triangular layouts.

References

Darghan, Rivera, Gonzalez & Castellanos (2022). [doi:10.47280/RevFacAgron\(LUZ\).v39.n1.18](https://doi.org/10.47280/RevFacAgron(LUZ).v39.n1.18)

Examples

```
f1 <- field_layout(nx = 12, ny = 8, ntrt = 3, nblk = 2, seed = 3000)
head(f1)
```

field_trial	<i>Example field trial with a single outer edge effect</i>
-------------	--

Description

A simulated staggered-planting trial (12 x 8 units, 3 treatments, 2 blocks) where the outer ring has a higher mean response than the interior, used to demonstrate edge-effect detection.

Usage

```
field_trial
```

Format

A data.frame with 96 rows and 6 columns:

x, y plot coordinates

trt treatment factor (T1-T3)

blk block factor (B1-B2)

rep replicate factor

response simulated yield

References

Darghan, Rivera, Gonzalez & Castellanos (2022). [doi:10.47280/RevFacAgron\(LUZ\).v39.n1.18](https://doi.org/10.47280/RevFacAgron(LUZ).v39.n1.18)

kappa_hat	<i>Least-squares estimator of the Besag-Kempton competition coefficient</i>
-----------	---

Description

Computes the closed-form least-squares estimator of the competition coefficient κ associated with the edge effect, given a response vector, a spatial weight matrix, and a design. The perpendicular projection onto the column space of the design is applied implicitly via `qr.resid()`, so no explicit $n \times n$ projection matrix is formed and the estimator works for any (possibly rank-deficient) design.

Usage

```
kappa_hat(y, W, design)
```

Arguments

y Numeric response vector of length n .

W Symmetric $n \times n$ spatial weight matrix (see `border_weights()`).

design An $n \times p$ model matrix (e.g. from `model.matrix()`).

Value

A single numeric value, $\hat{\kappa}$.

References

Darghan, Rivera, Gonzalez & Castellanos (2022). doi:[10.47280/RevFacAgron\(LUZ\).v39.n1.18](https://doi.org/10.47280/RevFacAgron(LUZ).v39.n1.18)

Examples

```
n <- 12
xy <- expand.grid(x = 1:4, y = 1:3)
W <- border_weights(xy)
X <- model.matrix(~ rep(0:1, length.out = n))
kappa_hat(rnorm(n), W, X)
```

moran_border

Moran's I spatial autocorrelation of model residuals

Description

Computes Moran's I for the residuals of the fitted two-way model held in a `border_effect` object, using the same weight matrix used to estimate κ . A small p-value indicates spatial dependence of the residuals, a signature of the edge effect. The statistic and its normal-approximation variance follow the standard definition (as in `ape::Moran.I`).

Usage

```
moran_border(object)
```

Arguments

`object` A `border_effect` object.

Value

A list with observed, expected, sd, and (two-sided) p.value.

References

Darghan, Rivera, Gonzalez & Castellanos (2022). doi:[10.47280/RevFacAgron\(LUZ\).v39.n1.18](https://doi.org/10.47280/RevFacAgron(LUZ).v39.n1.18)

Examples

```
f1 <- field_layout(nx = 6, ny = 4, ntrt = 4, nblk = 3, seed = 5)
set.seed(2); f1$resp <- rnorm(24)
be <- border_effect(resp ~ trt + blk, data = f1, coords = f1)
moran_border(be)
```

run_border_app*Launch the BorderEffect Shiny application*

Description

Opens an interactive application with two modes: **Analyze** a real trial (upload a CSV with x , y , trt , $block$, $response$, or pick one of the bundled example scenarios – no border, single/double outer border, and unilateral/bilateral/trilateral partial borders – then read the κ density, confidence interval and verdict) and **Explore** the method (adjust layout and effect size and watch the κ distribution respond). All statistics are computed by the package's exported functions.

Usage

```
run_border_app()
```

Value

A Shiny app object (invisibly, when run interactively).

Examples

```
if (interactive() && requireNamespace("shiny", quietly = TRUE)) {  
  run_border_app()  
}
```

Index

* datasets

- field_trial, 7
- border_effect, 2
- border_indicators, 3
- border_indicators(), 4–6
- border_test, 4
- border_weights, 5
- border_weights(), 2, 7
- field_layout, 5
- field_layout(), 3
- field_trial, 7
- kappa_hat, 7
- model.matrix(), 7
- moran_border, 8
- plot(), 6
- plot.field_layout(field_layout), 5
- qr.resid(), 2, 7
- run_border_app, 9
- simulate(), 4