Package: explodemap (via r-universe)

June 6, 2026

Title Hierarchical Exploded-View Cartography

Version 0.2.0

Description Tools for generating hierarchical exploded-view maps from dense administrative boundary data. The package applies rigid-body translations to polygon geometries using a centroid-driven vector field, preserving the internal geometry of each feature while separating units within and across regions. Parameters can be derived analytically from dataset geometry using closed-form models for regional separation and local expansion. The package also includes grouped layouts, optional bounded collision refinement, and an interactive focus-map widget for selected-area inspection in 'htmlwidgets' and 'Shiny'. It implements the methodology described in George Arthur (2026) <https://github.com/PrigasG/explodemap> ``A Hierarchical Vector-Based Framework for Multi-Scale Exploded-View Cartography''.

License MIT + file LICENSE

URL https://prigasg.github.io/explodemap/, https://github.com/PrigasG/explodemap

BugReports https://github.com/PrigasG/explodemap/issues

Depends R (>= 4.1.0)

Imports dplyr, ggplot2, htmlwidgets, purrr, rlang, sf, utils

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

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Config/pak/sysreqs libabsl-dev cmake libgdal-dev gdal-bin libgeos-dev make libuv1-dev libssl-dev libproj-dev libsqlite3-dev libudunits2-dev

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

Date/Publication 2026-06-06 10:51:20 UTC

RemoteUrl https://github.com/prigasg/explodemap

RemoteRef HEAD

RemoteSha ceeec224e93268ffa99fe2cd9cd8d4ab3f15281e

Contents

`apply_region_offsets` *Apply documented display offsets by region*

Description

Applies rigid display offsets to all features in each named region. This is intended as a reproducible finishing step for publication or dashboard layouts after an exploded map has already been computed.

Usage

```
apply_region_offsets(
  x,
  offsets,
  region_col = NULL,
  offset_region_col = "region",
  dx_col = "dx_m",
  dy_col = "dy_m",
  update_plots = TRUE
)
```

Arguments

x	An sf, exploded_map, or grouped_exploded_map object.
offsets	A data frame, or a path to a CSV file, containing one row per region and numeric offset columns.
region_col	Name of the region column in x. For exploded_map and grouped_exploded_map objects, the value recorded in x$diagnostics is used when this is NULL.
offset_region_col	Name of the region column in offsets.
dx_col	Name of the horizontal offset column in offsets.
dy_col	Name of the vertical offset column in offsets.
update_plots	Logical; when TRUE, rebuild stored ggplot objects for exploded map objects.

Details

Offsets are interpreted in the coordinate units of the projected geometry being adjusted. The default column names dx_m and dy_m assume a metric projected CRS, matching the package's exploded-map workflow.

Value

For an sf input, an sf object with the same attributes and CRS as x; geometries belonging to each listed region are translated by the supplied offsets, and the normalized offset table is stored in the display_offsets attribute. For an exploded_map, returns the same S3 object structure with sf_exp, sf_exp_wgs, stored plots, and display_offsets updated. For a grouped_exploded_map, returns the same S3 object structure with sf_grouped, sf_grouped_wgs, stored plots, and display_offsets updated. The offsets are rigid translations, so polygon shape, area, and within-region relative geometry are preserved.

Examples

```
library(sf)
x <- st_sf(
  region = c("A", "B"),
  geometry = st_sfc(
    st_point(c(0, 0)),
    st_point(c(10, 0)),
```

```
    crs = 3857
  )
)
offsets <- data.frame(region = "B", dx_m = 5, dy_m = 2)
apply_region_offsets(x, offsets, region_col = "region")
```

`cache_clear` *Clear explodemap download cache*

Description

Clear explodemap download cache

Usage

```
cache_clear(key = NULL)
```

Arguments

`key`	Specific cache key to clear, or NULL to clear all

Value

Invisibly returns `NULL`. The function is called for its side effect of deleting cached `.rds` files and reporting what was removed.

`cache_list` *List cached datasets*

Description

List cached datasets

Usage

```
cache_list()
```

Value

Invisibly returns a character vector of cached `.rds` file names. The visible output is a message listing the cache directory and file sizes, or a message that the cache is empty.

calibration_row *Extract calibration row from an exploded_map*

Description

Returns a one-row data.frame suitable for binding across datasets to build a calibration table.

Usage

```
calibration_row(x)
```

Arguments

x	An exploded_map object

Value

A one-row data.frame

compute_stats *Compute geometry statistics for parameter derivation*

Description

Extracts w_bar, R_local, n_bar, n_regions, and the tightness ratio from a projected sf object with a grouping column.

Usage

```
compute_stats(
  sf_obj,
  region_col,
  centroid_fun = c("centroid", "point_on_surface")
)
```

Arguments

sf_obj	Projected sf object
region_col	Name of the grouping column
centroid_fun	"centroid" (default) or "point_on_surface"

Value

A named list of geometry statistics

derive_params *Derive displacement parameters from geometry statistics*

Description

Implements Analytical Results 1 and 2 from the paper: alpha_r = gamma_r * w_bar / (2 * sin(pi / n_regions)) alpha_l = gamma_l * 2 * R_local / sqrt(n_bar)

Usage

```
derive_params(stats, gamma_r = 3, gamma_l = 1.136, p = 1.25)
```

Arguments

stats	Output of compute_stats()
gamma_r	Regional clearance coefficient (default 3.0)
gamma_l	Local clearance coefficient (default 1.136)
p	Distance scaling exponent (default 1.25)

Value

Named list with alpha_r, alpha_l, p, gamma_r, gamma_l

estimate_block_radii *Estimate block radius for each region*

Description

Computes the 85th percentile of distances from child-unit centroids to their region centroid. The 85th percentile is preferred over the maximum because the maximum is sensitive to outlier units and produces over-conservative block radii.

Usage

```
estimate_block_radii(
  sf_obj,
  region_col,
  quantile_p = 0.85,
  centroid_fun = c("centroid", "point_on_surface")
)
```

Arguments

sf_obj	Projected sf object with region column
region_col	Grouping column name
quantile_p	Quantile for radius estimation (default 0.85)
centroid_fun	"centroid" or "point_on_surface"

Value

data.frame with columns: region, block_radius, cx, cy, n_units

---

explode_grouped *Create a three-level grouped exploded map*

Description

Combines Level 1 (local explosion within regions) with Level 2/3 (anchor-based region block placement). This is the full three-level extension from Section 12 of the paper.

Usage

```
explode_grouped(
  sf_obj,
  region_col,
  mode = c("auto", "auto_collision", "manual"),
  anchors = NULL,
  alpha_l = NULL,
  p = 1.25,
  gamma_l = 1.136,
  kappa = 1.8,
  padding = 50000,
  delta = 15000,
  lambda = 0.18,
  eta = 0.18,
  padding_sep = 20000,
  anchor_expand = NULL,
  anchor_buffer = NULL,
  density_scale = NULL,
  block_sep = NULL,
  max_iter = 60,
  fix_invalid = TRUE,
  centroid_fun = c("centroid", "point_on_surface"),
  plot = TRUE,
  export = NULL,
  label = "Grouped Layout",
  quiet = FALSE
)
```

Arguments

sf_obj	Projected sf object with region column
region_col	Grouping column name
mode	"auto", "auto_collision", or "manual"
anchors	For mode = "manual": data.frame with anchor positions
alpha_l	Local expansion parameter for Level 1 (metres)
p	Distance scaling exponent (default 1.25)
gamma_l	Local clearance coefficient (default 1.136); used if alpha_l is NULL
kappa	Radial expansion factor (default 1.8)
padding	Base padding (default 50000)
delta	Log-density scaling (default 15000)
lambda	Spring coefficient (default 0.18)
eta	Repulsion step (default 0.18)
padding_sep	Minimum block separation (default 20000)
anchor_expand, anchor_buffer, density_scale, block_sep	Optional aliases for kappa, padding, delta, and padding_sep, respectively. These names are convenient in Shiny dashboards where the controls describe the visual effect rather than the solver term.
max_iter	Max collision iterations (default 60)
fix_invalid	Auto-repair invalid geometries (default TRUE)
centroid_fun	"centroid" or "point_on_surface"
plot	Print plots (default TRUE). Automatically suppressed inside a live Shiny session; use plot.grouped_exploded_map() inside renderPlot().
export	NULL, TRUE, or file path
label	Title for plots
quiet	If TRUE, suppress message() output. Default FALSE.

Details

The guarantees of Propositions 1-3 apply strictly at Level 1. Higher levels preserve structural grouping and directional correspondence rather than topological coverage.

Value

A grouped_exploded_map S3 object (inherits from exploded_map)

`explode_section` *Explode one selected section and keep the rest as context*

Description

`explode_section()` is a dashboard-oriented helper for exploratory maps. It applies an exploded layout only to the requested section, then recombines the untouched remainder of the layer as geographic context. The result can be passed directly to `focus_map()` with `context_col` to fade or hide the non-selected features.

Usage

```
explode_section(
  sf_obj,
  section_col,
  section,
  region_col = section_col,
  layout = c("explode", "grouped"),
  context = c("fade", "hide", "none"),
  role_col = ".explodemap_role",
  all_values = "all",
  ...
)
```

Arguments

`sf_obj`	Projected `sf` object.
`section_col`	Column containing the high-level sections users choose from, such as `"North"`, `"Central"`, and `"South"`.
`section`	Selected section value. Values in `all_values` explode the full layer.
`region_col`	Column used for the explosion inside the selected section. Defaults to `section_col`. For municipality drill-downs, this is often a county column.
`layout`	`"explode"` for `explode_sf()` or `"grouped"` for `explode_grouped()`.
`context`	`"fade"` or `"hide"` keeps non-selected features in the returned object as context; `"none"` drops them.
`role_col`	Name of the role column added to the output. Focus features are marked `"focus"` and context features are marked `"context"`.
`all_values`	Values that mean "all sections". Default `"all"`.
`...`	Passed to `explode_sf()` or `explode_grouped()`.

Value

An `exploded_map` or `grouped_exploded_map` object with recombined focus/context geometry and extra diagnostics.

Examples

```
poly <- function(xmin, ymin, xmax, ymax) {
  sf::st_polygon(list(rbind(
    c(xmin, ymin), c(xmax, ymin), c(xmax, ymax),
    c(xmin, ymax), c(xmin, ymin)
  )))
}

municipalities <- sf::st_sf(
  NAME = c("A", "B", "C", "D"),
  nj_region = c("South", "South", "North", "North"),
  county_name = c("Atlantic", "Cape May", "Bergen", "Hudson"),
  geometry = sf::st_sfc(
    poly(0, 0, 1000, 1000),
    poly(2000, 0, 3000, 1000),
    poly(0, 2000, 1000, 3000),
    poly(2000, 2000, 3000, 3000),
    crs = 3857
  )
)

focused <- explode_section(
  municipalities,
  section_col = "nj_region",
  section = "South",
  region_col = "county_name",
  alpha_r = 1800,
  alpha_l = 1200,
  plot = FALSE,
  quiet = TRUE
)

focus_map(
  focused,
  label_col = "NAME",
  context_col = ".explodemap_role",
  context_mode = "fade"
)
```

`explode_sf` *Explode any sf object with an existing grouping column*

Description

Explode any sf object with an existing grouping column

Usage

```
explode_sf(
  sf_obj,
  region_col = "region",
  gamma_r = 3,
  gamma_l = 1.136,
  p = 1.25,
  alpha_r = NULL,
  alpha_l = NULL,
  refine = FALSE,
  refine_min_gap = NULL,
  refine_max_shift = NULL,
  refine_max_iter = 20,
  refine_step = 0.5,
  refine_within = c("region", "all"),
  allow_other = FALSE,
  fix_invalid = TRUE,
  centroid_fun = c("centroid", "point_on_surface"),
  plot = TRUE,
  export = NULL,
  label = "Custom Dataset",
  quiet = FALSE
)
```

Arguments

sf_obj	Projected sf object (metric CRS)
region_col	Name of the column defining groups
gamma_r	Regional clearance coefficient (default 3.0)
gamma_l	Local clearance coefficient (default 1.136)
p	Distance scaling exponent (default 1.25)
alpha_r	Optional manual override for regional separation (metres). May be supplied independently of alpha_l.
alpha_l	Optional manual override for local expansion (metres). May be supplied independently of alpha_r.
refine	If TRUE, apply a bounded collision-refinement pass after the analytical displacement. Default is FALSE.
refine_min_gap	Optional minimum boundary-to-boundary gap in map units. If NULL and refine = TRUE, uses 2% of the characteristic diameter.
refine_max_shift	Optional maximum correction per feature in map units. If NULL and refine = TRUE, uses 10% of alpha_r + alpha_l.
refine_max_iter	Maximum refinement iterations.
refine_step	Fraction of each gap deficit corrected per iteration.

refine_within	Refine pairs within each "region" (default) or across "all" features.
allow_other	If TRUE, permits "Other" units
fix_invalid	If TRUE, auto-repairs invalid geometries
centroid_fun	"centroid" (default) or "point_on_surface"
plot	Print plots on return. Automatically suppressed when called inside a live Shiny session; use plot.exploded_map() inside renderPlot() instead.
export	NULL, TRUE, or file path
label	Title for plots
quiet	If TRUE, suppress all message() output. Default FALSE.

Value

An exploded_map S3 object

`explode_sf_with_lookup` *Explode any sf object using an external lookup table*

Description

Joins the lookup to sf_obj before exploding. Unmatched units are labelled "Other".

Usage

```
explode_sf_with_lookup(
  sf_obj,
  join_col,
  lookup,
  lookup_key = join_col,
  region_col = "region",
  quiet = FALSE,
  ...
)
```

Arguments

sf_obj	Projected sf object
join_col	Column in sf_obj to join on
lookup	data.frame with join key and region column
lookup_key	Column name in lookup matching join_col
region_col	Column name in lookup containing region labels
quiet	If TRUE, suppress message() output. Default FALSE. Passed through to explode_sf().
...	Passed to explode_sf()

Value

An exploded_map S3 object

`explode_state` *Explode a US state from TIGER/Line data*

Description

Downloads municipal boundaries automatically, groups by county-to-region mapping, derives parameters via Analytical Results 1-2, and returns an exploded_map S3 object.

Usage

```
explode_state(
  state_fips,
  crs,
  region_map,
  gamma_r = 3,
  gamma_l = 1.136,
  p = 1.25,
  alpha_r = NULL,
  alpha_l = NULL,
  refine = FALSE,
  refine_min_gap = NULL,
  refine_max_shift = NULL,
  refine_max_iter = 20,
  refine_step = 0.5,
  refine_within = c("region", "all"),
  allow_other = FALSE,
  fix_invalid = TRUE,
  centroid_fun = c("centroid", "point_on_surface"),
  plot = TRUE,
  export = NULL,
  label = paste0("FIPS ", state_fips),
  quiet = FALSE
)
```

Arguments

state_fips	2-digit FIPS code (e.g. "34" for NJ)
crs	Projected CRS EPSG code (metric units)
region_map	Named list: region_name -> character vector of county names
gamma_r	Regional clearance coefficient (default 3.0)
gamma_l	Local clearance coefficient (default 1.136)
p	Distance scaling exponent (default 1.25)

alpha_r	Optional manual override for regional separation (metres). May be supplied independently of alpha_l.
alpha_l	Optional manual override for local expansion (metres). May be supplied independently of alpha_r.
refine	If TRUE, apply a bounded collision-refinement pass after the analytical displacement. Default is FALSE.
refine_min_gap	Optional minimum boundary-to-boundary gap in map units. If NULL and refine = TRUE, uses 2% of the characteristic diameter.
refine_max_shift	Optional maximum correction per feature in map units. If NULL and refine = TRUE, uses 10% of alpha_r + alpha_l.
refine_max_iter	Maximum refinement iterations.
refine_step	Fraction of each gap deficit corrected per iteration.
refine_within	Refine pairs within each "region" (default) or across "all" features.
allow_other	If TRUE, permits units mapped to "Other"
fix_invalid	If TRUE, auto-repairs invalid geometries
centroid_fun	"centroid" (default) or "point_on_surface"
plot	Print plots on return (default TRUE). Automatically suppressed when called inside a live Shiny session; use plot.exploded_map() inside renderPlot() instead.
export	NULL (no export), TRUE (auto-named GeoJSON), or a file path
label	Title for plots and print output
quiet	If TRUE, suppress all message() output (useful inside Shiny reactive() and observe() where messages are invisible to users). Default FALSE.

Value

An exploded_map S3 object

---

export_topojson *Export an exploded map as TopoJSON*

Description

Converts an sf, exploded_map, or grouped_exploded_map object to TopoJSON using the external mapshaper command-line tool. The input is first written as a temporary GeoJSON file, then converted by mapshaper.

Usage

```
export_topojson(x, file, simplify = NULL, overwrite = FALSE)
```

Arguments

x	An sf, exploded_map, or grouped_exploded_map object. For exploded_map objects, the WGS84 exploded geometry (sf_exp_wgs) is exported. For grouped_exploded_map objects, the WGS84 grouped geometry (sf_grouped_wgs) is exported.
file	Output file path. Should end in .topojson or .json.
simplify	Optional simplification proportion passed to mapshaper -simplify. Must be a single number between 0 and 1 (exclusive). For example, simplify = 0.5 retains 50% of vertices. Default is NULL (no simplification). Note that simplification modifies polygon geometry and therefore breaks the exact geometry preservation guarantee of Proposition 1.
overwrite	Logical; if TRUE, overwrite file if it already exists. Default is FALSE.

Details

This is intended as a convenience helper for downstream tools such as Power BI, D3.js, and Observable that prefer or require TopoJSON input.

This function requires the external mapshaper command-line tool to be installed and available on the system path. It can be installed with:

```
npm install -g mapshaper
```

The mapshaper tool is not an R package dependency; it is invoked via system2(). If mapshaper is not found, the function errors with an informative message.

Value

Invisibly returns the output file path.

Examples

```
## Not run:
# Two-level export
result <- explode_sf(my_sf, region_col = "region", plot = FALSE)
export_topojson(result, "exploded.topojson")

# Three-level grouped export
grp <- explode_grouped(my_sf, region_col = "region", plot = FALSE)
export_topojson(grp, "grouped.topojson")

# With simplification (breaks Proposition 1 guarantee)
export_topojson(result, "simplified.topojson", simplify = 0.5)

# Raw sf object
export_topojson(my_sf, "raw.topojson", overwrite = TRUE)

## End(Not run)
```

focus_map *Interactive focus-map viewer*

Description

Renders spatial features as a smooth, interactive SVG map. Click any polygon to zoom in and lift it from the map with a "toast" effect; right-click or press Escape to reset. Camera transitions use D3's optimal zoom interpolation for fluid 60 fps motion with zero server round-trips.

Usage

```
focus_map(
  x,
  label_col = NULL,
  id_col = NULL,
  group_col = NULL,
  group_palette = NULL,
  context_col = NULL,
  context_values = "context",
  context_mode = c("fade", "hide", "show"),
  context_fill = "#cfd9df",
  context_opacity = 0.18,
  context_clickable = FALSE,
  focus_preset = c("none", "municipal", "drilldown", "municipal_drilldown"),
  simplify = TRUE,
  fill = "#2d6ea3",
  fill_opacity = 0.58,
  stroke = "#ffffff",
  lift_scale = 1.16,
  focus_padding = 40,
  focus_size = 0.76,
  min_focus_width = 0,
  min_focus_height = 0,
  tiny_feature_threshold = 48,
  tiny_feature_boost = 1,
  max_zoom = NULL,
  origin_context = c("none", "socket", "inset", "both"),
 origin_context_position = c("bottom-left", "bottom-right", "top-left", "top-right"),
  focus_context_opacity = 0.3,
  show_drag_zoom = FALSE,
  font_size = 14,
  show_labels = TRUE,
  show_sidebar = TRUE,
  performance_mode = NULL,
  info_cols = NULL,
  info_labels = NULL,
  info_title = NULL,
```

```
  info_position = c("top-right", "top-left", "bottom-right", "bottom-left"),
  info_card_scale = 1,
  area_min = 5000,
  width_min = 95,
  height_min = 28,
  width = "100%",
  height = "600px",
  elementId = NULL
)

focusmapOutput(outputId, width = "100%", height = "600px")

renderFocusmap(expr, env = parent.frame(), quoted = FALSE)
```

Arguments

x	An sf, exploded_map, or grouped_exploded_map object.
label_col	Character. Column name for polygon labels. Auto-detected if NULL.
id_col	Optional stable feature ID column for Shiny selection events. Defaults to row order if NULL.
group_col	Character. Optional column for region/group colouring. Polygons sharing a group value share a hue.
group_palette	Optional named character vector of colours for group_col values. Names should match group values; unmatched groups fall back to the widget palette.
context_col	Optional column identifying features that should remain as geographic context rather than active focus features.
context_values	Character vector of values in context_col that mark context features. Default "context".
context_mode	How context features are drawn: "fade" keeps them visible but muted, "hide" makes them invisible, and "show" draws them normally.
context_fill	Fill colour for context features when context_mode = "fade".
context_opacity	Fill opacity for faded context features.
context_clickable	Should context features remain clickable? Default FALSE.
focus_preset	Optional named preset for common interactive workflows. "municipal" tunes small-area focus, source cues, drag zoom, and dense layer performance. "drilldown" tunes context fading and source cues for selected-section maps. "municipal_drilldown" combines both. Explicit arguments supplied by the user override preset defaults.
simplify	Controls geometry simplification for rendering performance. TRUE (default) applies a sensible tolerance (dTolerance = 0.001 in WGS 84 degrees, $\approx$ 100 m). A positive number sets a custom tolerance. FALSE disables simplification. Only affects the widget copy — the original data is never modified.
fill	Fill colour (used when group_col is NULL). Default "#2d6ea3".
fill_opacity	Fill opacity. Default 0.58.

`stroke` Stroke colour. Default `"#ffffff"`.

`lift_scale` Initial toast lift scale. Default `1.16`. Increase this to make the lifted feature larger.

`focus_padding` Extra screen-space padding in pixels around the lifted feature during focus. Increase this if large lifted features feel too close to the map edge.

`focus_size` Target fraction of the map viewport the lifted feature may occupy. Increase this to make selected areas appear larger while preserving `focus_padding`.

`min_focus_width`, `min_focus_height`
Minimum focused feature width and height in screen pixels. When a selected feature is very small, the widget may zoom past the usual density-aware default until the lifted feature reaches these dimensions. Set to `0` to disable either constraint.

`tiny_feature_threshold`
Screen-pixel size below which a selected feature receives an adaptive lift-scale boost. Set to `0` to disable.

`tiny_feature_boost`
Maximum multiplier applied to `lift_scale` for the smallest features. Values below `1` are not allowed.

`max_zoom` Optional maximum camera zoom. If `NULL`, a density-aware default is used.

`origin_context` How the selected feature's source location should be shown while focused. The default `"none"` keeps focus maps visually unchanged unless this feature is explicitly enabled. `"socket"` keeps the source outline in the main map, `"inset"` shows a small overview map, `"both"` uses both, and `"none"` disables the cue.

`origin_context_position`
Position for the overview inset: `"bottom-left"`, `"bottom-right"`, `"top-left"`, or `"top-right"`.

`focus_context_opacity`
Fill opacity for non-selected features while a feature is focused. Lower values make tiny selected areas easier to read.

`show_drag_zoom` Show a widget-level drag-zoom toggle. When enabled, users can draw a marquee rectangle to zoom into dense clusters while ordinary feature clicks continue to focus the map. Shift-drag works as a shortcut even when the button is hidden.

`font_size` Label font size in px. Default `14`.

`show_labels` Show labels on lifted shapes? Default `TRUE`.

`show_sidebar` Deprecated and has no effect. Will be removed in a future version.

`performance_mode`
Logical or `NULL`. If `NULL`, dense layers automatically use shorter camera transitions and lighter in-flight rendering. Set `TRUE` to force it or `FALSE` to disable it.

`info_cols` Optional character vector of columns to show in a non-blocking focus card when a feature is selected.

`info_labels` Optional named character vector or list for display labels in the focus card. Names should match `info_cols`.

`info_title` Optional column to use as the focus card title. Defaults to `label_col`.

info_position	Position for the focus card: "top-right", "top-left", "bottom-right", or "bottom-left".
info_card_scale	Relative size for the focus card. Values above 1 make the card larger; values below 1 make it more compact.
area_min	Min screen area (px^2) for label visibility.
width_min	Min screen width (px) for label visibility.
height_min	Min screen height (px) for label visibility.
width	Widget width. Default "100%".
height	Widget height. Default "600px".
elementId	Optional element ID.
outputId	Shiny output ID.
expr	Expression that returns a focus_map() widget.
env	Environment in which to evaluate expr.
quoted	Logical. Is expr already quoted?

Details

Accepts raw sf objects, exploded_map results from explode_sf, or grouped_exploded_map results from explode_grouped. For exploded objects, the displaced (WGS 84) geometry is used automatically.

Value

An htmlwidgets object.

Examples

```
poly <- function(xmin, ymin, xmax, ymax) {
  sf::st_polygon(list(rbind(
    c(xmin, ymin), c(xmax, ymin), c(xmax, ymax),
    c(xmin, ymax), c(xmin, ymin)
  )))
}

counties <- sf::st_sf(
  NAME = c("A", "B"),
  region = c("North", "South"),
  geometry = sf::st_sfc(
    poly(-74.2, 40.0, -74.0, 40.2),
    poly(-73.9, 40.0, -73.7, 40.2),
    crs = 4326
  )
)

focus_map(counties, label_col = "NAME", group_col = "region")
```

focus_map_preset *Focus-map option presets*

Description

Returns a named list of `focus_map()` arguments for common interactive mapping workflows. Presets are intentionally plain lists so they can be inspected, modified, or passed through `do.call()`.

Usage

```
focus_map_preset(
  name = c("municipal", "drilldown", "municipal_drilldown", "none")
)
```

Arguments

`name`	Preset name. `"municipal"` tunes small-area focus maps with adaptive sizing, source cues, drag zoom, and dense-layer performance. `"drilldown"` tunes selected-section maps with faded context and source cues. `"municipal_drilldown"` combines both.

Value

A named list of `focus_map()` arguments.

Examples

```
focus_map_preset("municipal")
```

layout_regions *Compute region anchor positions for grouped layouts*

Description

Implements the anchor layout procedure from Section 12. Supports three modes: automatic radial placement, automatic with collision resolution, and manual user-specified anchors.

Usage

```
layout_regions(
  sf_obj,
  region_col,
  mode = c("auto", "auto_collision", "manual"),
  anchors = NULL,
  kappa = 1.8,
  padding = 50000,
```

```
  delta = 15000,
  lambda = 0.18,
  eta = 0.18,
  padding_sep = 20000,
  max_iter = 60,
  quantile_p = 0.85,
  centroid_fun = c("centroid", "point_on_surface"),
  quiet = FALSE
)
```

Arguments

sf_obj	Projected sf object with region column
region_col	Grouping column name
mode	"auto" (radial only), "auto_collision" (radial + solver), or "manual"
anchors	For mode = "manual": data.frame with columns (region_col, anchor_x, anchor_y)
kappa	Radial expansion factor (default 1.8)
padding	Base padding in map units (default 50000)
delta	Log-density scaling factor (default 15000)
lambda	Spring coefficient for collision solver (default 0.18)
eta	Repulsion step size for collision solver (default 0.18)
padding_sep	Minimum separation between blocks (default 20000)
max_iter	Max iterations for collision solver (default 60)
quantile_p	Quantile for block radius estimation (default 0.85)
centroid_fun	"centroid" or "point_on_surface"
quiet	If TRUE, suppress message() output. Default FALSE.

Value

data.frame with region, anchor_x, anchor_y, block_radius, n_units

---

plot.exploded_map *Plot an exploded_map object*

Description

Plot an exploded_map object

Usage

```
## S3 method for class 'exploded_map'
plot(x, which = c("exploded", "original", "both"), ...)
```

Arguments

x	An exploded_map object
which	"exploded" (default), "original", or "both"
...	Ignored

Value

Invisibly returns x, the original exploded_map object. The method is called for its side effect of drawing the stored ggplot2 original and/or exploded map objects.

---

plot.grouped_exploded_map

Plot a grouped_exploded_map object

Description

Plot a grouped_exploded_map object

Usage

```
## S3 method for class 'grouped_exploded_map'
plot(x, which = c("grouped", "original", "local", "all"), ...)
```

Arguments

x	A grouped_exploded_map object
which	"grouped" (default), "original", "local", or "all"
...	Ignored

Value

Invisibly returns x, the original grouped_exploded_map object. The method is called for its side effect of drawing the stored ggplot2 grouped, original, and/or local map objects.

print.exploded_map *Print an exploded_map object*

Description

Print an exploded_map object

Usage

```
## S3 method for class 'exploded_map'
print(x, ...)
```

Arguments

x	An exploded_map object
...	Ignored

Value

Invisibly returns x, the original exploded_map object. The method is called for its side effect of printing a compact diagnostic overview of units, regions, derived parameters, and optional refinement information.

print.grouped_exploded_map *Print a grouped_exploded_map object*

Description

Print a grouped_exploded_map object

Usage

```
## S3 method for class 'grouped_exploded_map'
print(x, ...)
```

Arguments

x	A grouped_exploded_map object
...	Ignored

Value

Invisibly returns x, the original grouped_exploded_map object. The method is called for its side effect of printing a compact diagnostic overview of units, regions, grouping mode, and grouped-layout parameters.

summary.exploded_map *Summary of an exploded_map object*

Description

Summary of an exploded_map object

Usage

```
## S3 method for class 'exploded_map'
summary(object, ...)
```

Arguments

object	An exploded_map object
...	Ignored

Value

Invisibly returns object, the original exploded_map object. The method is called for its side effect of printing a human-readable summary of dataset size, geometry statistics, displacement parameters, refinement diagnostics, and implied calibration coefficients.

summary.grouped_exploded_map *Summary of a grouped_exploded_map object*

Description

Summary of a grouped_exploded_map object

Usage

```
## S3 method for class 'grouped_exploded_map'
summary(object, ...)
```

Arguments

object	A grouped_exploded_map object
...	Ignored

Value

Invisibly returns object, the original grouped_exploded_map object. The method is called for its side effect of printing dataset size, grouping diagnostics, anchor parameters, and anchor radii.

Index