Package 'supercells'

November 12, 2026

Title Superpixels of Spatial Data

Version 2.0.1

Description Creates superpixels based on input spatial data.
This package works on spatial data with one variable (e.g., continuous raster), many variables (e.g., RGB rasters), and spatial patterns (e.g., areas in categorical rasters).
It is based on the SLIC algorithm (Achanta et al. (2012) <doi:10.1109/TPAMI.2012.120>), and readapts it to work with arbitrary dissimilarity measures.

License GPL (>= 3)

Encoding UTF-8

Depends R (>= 4.0.0)

Imports sf, terra (>= 1.4-21), philentropy (>= 0.6.0), units

LinkingTo cpp11

URL https://jakubnowosad.com/supercells/

BugReports https://github.com/Nowosad/supercells/issues

Suggests knitr, covr, testthat (>= 3.0.0), rmarkdown, stars

Config/testthat/edition 3

Config/roxygen2/version 8.1.0

NeedsCompilation yes

Author Jakub Nowosad [aut, cre] (ORCID:
<https://orcid.org/0000-0002-1057-3721>),
Pascal Mettes [ctb] (Author of the initial C++ implementation of the
SLIC Superpixel algorithm for image data),
Lucas De Oto [ctb]

Maintainer Jakub Nowosad <nowosad.jakub@gmail.com>

Repository CRAN

Date/Publication 2026-11-12 18:30:02 UTC

Contents

sc_list_distances . . . 2
sc_metrics_global . . . 3
sc_metrics_pixels . . . 5
sc_metrics_supercells . . . 6
sc_slic . . . 8
sc_slic_convergence . . . 10
sc_slic_get_params . . . 11
sc_slic_points . . . 12
sc_slic_raster . . . 14
sc_slic_set_params . . . 15
sc_tune_compactness . . . 16
supercells . . . 17
use_adaptive . . . 20
use_meters . . . 20

Index **22**

`sc_list_distances` *List supported distance measures*

Description

Returns information about the distance measures that can be provided by name to the `dist_fun` argument. The table describes their input requirements, what they compare, the effect of rescaling individual raster layers, and practical guidance.

Usage

```
sc_list_distances()
```

Details

Other measures can be supplied explicitly as a function to `dist_fun`. In that case, the function must take two numeric vectors and return one finite, nonnegative value, with smaller values meaning greater similarity.

For `"cosine_distance"`, two zero profiles have distance zero. A zero profile and a nonzero profile have distance one.

Value

A data frame with one row per supported measure and columns:

`name` Value accepted by `dist_fun`.

`label` Measure name.

`input` Type of the input vector.

`compares` What aspect of the vectors the measure compares.

`effect_of_layer_rescaling` Effect of multiplying individual raster layers by positive constants.

`guidance` Practical usage guidance.

Examples

```
sc_list_distances()
```

`sc_metrics_global` *Global supercells metrics*

Description

Computes global distance diagnostics for supercells

Usage

```
sc_metrics_global(
  x,
  sc,
  metrics = c("spatial", "value", "combined", "balance"),
  scale = TRUE,
  step,
  compactness,
  dist_fun = NULL
)
```

Arguments

`x`	The input SpatRaster used to create `sc`.
`sc`	An sf object returned by `sc_slic()`.
`metrics`	Character vector of metric ideas to return. Options: `"spatial"`, `"value"`, `"combined"`, `"balance"`. Default: `c("spatial", "value", "combined", "balance")`.
`scale`	Logical. If `TRUE`, scales spatial and value distances; output columns are named with the `_scaled` suffix.
`step`	A step value used for the supercells If missing, uses `attr(sc, "step")` when available
`compactness`	A compactness value used for the supercells If missing, uses `attr(sc, "compactness")` when available. Compactness mode is read from `attr(sc, "compactness_method")` when available.
`dist_fun`	A distance function name or function, as in `sc_slic()`. If missing or `NULL`, uses `attr(sc, "dist_fun")` when available.

Details

Requires sc with metadata columns (supercells, x, y). If they are missing, they are derived from geometry and row order. Use outcomes = c("supercells", "coordinates", "values") when calling sc_slic() or supercells() to preserve original centers and IDs. Metrics are averaged across supercells (each supercell has equal weight). With adaptive compactness (set compactness = use_adaptive() in sc_slic()), value distances are scaled separately for each center using a local maximum calculated from its final position and search window. The combined and balance metrics use these scaled distances.

Value

A data.frame with a single row and columns:

step Step size used to generate supercells. Returned in meters when the input used step = use_meters(...), otherwise in cells.

compactness Compactness value used to generate supercells; NA for adaptive compactness.

compactness_method Compactness method: "constant" for fixed compactness, "local_max" for adaptive compactness.

n_supercells Number of supercells with at least one non-missing pixel.

mean_value_dist / mean_value_dist_scaled Mean per-supercell value distance from cells to their supercell centers, averaged across supercells. Returned as mean_value_dist (or mean_value_dist_scaled when scale = TRUE). Lower values indicate more homogeneous supercells.

mean_spatial_dist / mean_spatial_dist_scaled Mean per-supercell spatial distance from cells to their supercell centers, averaged across supercells; units are grid cells (row/column index distance). If the input supercells were created with step = use_meters(...), distances are reported in meters. Returned as mean_spatial_dist (or mean_spatial_dist_scaled when scale = TRUE). Lower values indicate more compact supercells.

mean_combined_dist Mean per-supercell combined distance, computed from value and spatial distances using compactness and step, averaged across supercells. Returned as mean_combined_dist. Lower values indicate lower overall distance and are mainly useful for ranking.

balance Mean signed log ratio of scaled value distance to scaled spatial distance (averaged across supercells); 0 indicates balance between value and spatial terms, negative values indicate spatial dominance, and positive values indicate value dominance. Supercells with a zero scaled component are excluded; NA is returned if none remain.

See Also

sc_slic(), sc_metrics_pixels(), sc_metrics_supercells()

Examples

```
library(supercells)
vol = terra::rast(system.file("raster/volcano.tif", package = "supercells"))
vol_sc = sc_slic(vol, step = 8, compactness = 7)
sc_metrics_global(vol, vol_sc)
```

`sc_metrics_pixels` *Pixel-level supercells metrics*

Description

Computes per-pixel distance diagnostics from each pixel to its supercell center

Usage

```
sc_metrics_pixels(
  x,
  sc,
  metrics = c("spatial", "value", "combined", "balance"),
  scale = TRUE,
  step,
  compactness,
  dist_fun = NULL
)
```

Arguments

`x`	The input SpatRaster used to create `sc`.
`sc`	An sf object returned by `sc_slic()`.
`metrics`	Character vector of metrics to return. Options: `"spatial"`, `"value"`, `"combined"`, `"balance"`. Default: `c("spatial", "value", "combined", "balance")`.
`scale`	Logical. If `TRUE`, returns `spatial` and `value` as scaled distances (`spatial_scaled`, `value_scaled`).
`step`	A step value used for the supercells If missing, uses `attr(sc, "step")` when available
`compactness`	A compactness value used for the supercells If missing, uses `attr(sc, "compactness")` when available. Compactness mode is read from `attr(sc, "compactness_method")` when available.
`dist_fun`	A distance function name or function, as in `sc_slic()`. If missing or `NULL`, uses `attr(sc, "dist_fun")` when available.

Details

If `sc` lacks `supercells`, `x`, or `y` columns, they are derived from geometry and row order, which may differ from the original centers. With adaptive compactness (set `compactness = use_adaptive()` in `sc_slic()`), value distances are scaled separately for each center using a local maximum calculated from its final position and search window. The combined and balance metrics use these scaled distances.

Value

A SpatRaster with one or more layers depending on `metrics` and `scale`.

spatial Spatial distance from each pixel to its supercell center in grid-cell units (row/column index distance). If the input supercells were created with `step = use_meters(...)`, distances are reported in meters. Lower values indicate more compact supercells. Returned as `spatial_scaled` when `scale = TRUE`.

value Value distance from each pixel to its supercell center in the raster value space. Lower values indicate more homogeneous supercells. Returned as `value_scaled` when `scale = TRUE`.

combined Combined distance using `compactness` and `step`. Overall distance; mainly useful for ranking.

balance Signed log ratio of scaled value distance to scaled spatial distance. `0` indicates balance; negative values indicate spatial dominance; positive values indicate value dominance. Always computed from scaled components. Returns `NA` when either component is zero.

See Also

`sc_slic()`, `sc_metrics_supercells()`, `sc_metrics_global()`

Examples

```
library(supercells)
vol = terra::rast(system.file("raster/volcano.tif", package = "supercells"))
vol_sc = sc_slic(vol, step = 8, compactness = 7)
metrics = sc_metrics_pixels(vol, vol_sc, scale = TRUE)
terra::panel(metrics, nr = 1)
```

---

sc_metrics_supercells *Supercell-level metrics*

Description

Computes per-supercell distance diagnostics

Usage

```
sc_metrics_supercells(
  x,
  sc,
  metrics = c("spatial", "value", "combined", "balance"),
  scale = TRUE,
  step,
  compactness,
  dist_fun = NULL
)
```

Arguments

x	The input SpatRaster used to create sc.
sc	An sf object returned by sc_slic().
metrics	Character vector of metric ideas to return. Options: "spatial", "value", "combined", "balance". Default: c("spatial", "value", "combined", "balance").
scale	Logical. If TRUE, scales spatial and value distances; output columns are named with the _scaled suffix.
step	A step value used for the supercells If missing, uses attr(sc, "step") when available
compactness	A compactness value used for the supercells If missing, uses attr(sc, "compactness") when available. Compactness mode is read from attr(sc, "compactness_method") when available.
dist_fun	A distance function name or function, as in sc_slic(). If missing or NULL, uses attr(sc, "dist_fun") when available.

Details

If sc lacks supercells, x, or y columns, they are derived from geometry and row order, which may differ from the original centers. With adaptive compactness (set compactness = use_adaptive() in sc_slic()), value distances are scaled separately for each center using a local maximum calculated from its final position and search window. The combined and balance metrics use these scaled distances.

Value

An sf object with one row per supercell and columns:

supercells Supercell ID.

mean_spatial_dist Mean spatial distance from cells to the supercell center in grid-cell units (row/column index distance). If the input supercells were created with step = use_meters(...), distances are reported in meters. Lower values indicate more compact supercells. Returned as mean_spatial_dist_scaled when scale = TRUE.

mean_value_dist Mean value distance from cells to the supercell center in value space. Lower values indicate more homogeneous supercells. Returned as mean_value_dist_scaled when scale = TRUE.

mean_combined_dist Mean combined distance using compactness and step to scale value and spatial distances. Overall distance; mainly useful for ranking.

balance Signed log ratio of scaled value distance to scaled spatial distance. 0 indicates balance; negative values indicate spatial dominance; positive values indicate value dominance. Returns NA when either scaled component is zero.

See Also

sc_slic(), sc_metrics_pixels(), sc_metrics_global()

Examples

```
library(supercells)
vol = terra::rast(system.file("raster/volcano.tif", package = "supercells"))
vol_sc = sc_slic(vol, step = 8, compactness = 7)
cl = sc_metrics_supercells(vol, vol_sc)
head(cl)
```

`sc_slic` *Create supercells*

Description

Creates supercells from single- or multi-band rasters using an extended SLIC algorithm. The function supports either a target number of supercells (`k`) or a fixed grid spacing (`step`), as well as optional custom centers and chunking.

Usage

```
sc_slic(
  x,
  step = NULL,
  compactness,
  dist_fun = "euclidean",
  avg_fun = "mean",
  clean = TRUE,
  minarea,
  iter = 10,
  k = NULL,
  centers = NULL,
  outcomes = c("supercells", "coordinates", "values"),
  chunks = FALSE,
  verbose = 0
)
```

Arguments

`x`	An object of class SpatRaster (terra) or class stars (stars).
`step`	Initial center spacing (alternative is `k`). Provide a plain numeric value for cell units, or use `use_meters()` for map-distance steps in meters (automatically converted to cells using raster resolution).
`compactness`	A compactness value. Use `sc_tune_compactness()` to estimate it. Use `use_adaptive()` to enable adaptive compactness based on local feature-space distances.
`dist_fun`	A supported distance name from `sc_list_distances()` or a custom function that takes two numeric vectors and returns one finite, nonnegative value. Smaller values must mean greater similarity.

avg_fun	An averaging function name or custom function used to summarize values within each supercell. Supported names: "mean" and "median". A custom function must accept a numeric vector and return a single numeric value.
clean	Should connectivity of the supercells be enforced?
minarea	When clean = TRUE, connected components containing minarea cells or fewer are merged into an adjacent component. If minarea is omitted or set to 0, the threshold is selected automatically as one quarter of the raster cell count divided by the number of initialized centers: floor(ncell / (4 * n_centers)).
iter	Number of iterations.
k	The number of supercells desired (alternative to step).
centers	Optional sf object of custom centers. Requires step.
outcomes	Character vector controlling which fields are returned. Allowed values are "supercells", "coordinates", and "values". Default is full output (c("supercells", "coordinates", "values")). Use outcomes = "values" for value summaries only.
chunks	Chunking option. Use FALSE for no chunking, TRUE for automatic chunking based on size, or a numeric value for a fixed chunk size (in number of cells per side).
verbose	Verbosity level.

Details

Use sc_slic() for polygon outputs. For raster or point centers outputs, see sc_slic_raster() and sc_slic_points(). Evaluation and diagnostic options:

- Iteration convergence: use sc_slic_convergence() and plot its output.
- Pixel diagnostics: sc_metrics_pixels() for per-pixel spatial, value, and combined distances.
- Cluster diagnostics: sc_metrics_supercells() for per-supercell summaries.
- Global diagnostics: sc_metrics_global() for a single-row summary.

Value

An sf object with the supercell polygons and summary statistics. Information on step, compactness, and compactness_method are attached to the result as attributes (compactness is NA for adaptive compactness).

References

Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., & Süsstrunk, S. (2012). SLIC Superpixels Compared to State-of-the-Art Superpixel Methods. IEEE Transactions on Pattern Analysis and Machine Intelligence, 34(11), 2274–2282. https://doi.org/10.1109/tpami.2012.120

Nowosad, J., Stepinski, T. (2022). Extended SLIC superpixels algorithm for applications to non-imagery geospatial rasters. International Journal of Applied Earth Observation and Geoinformation, https://doi.org/10.1016/j.jag.2022.102935

See Also

sc_tune_compactness(), use_meters(), use_adaptive(), sc_slic_raster(), sc_slic_points(), sc_slic_convergence(), sc_metrics_pixels(), sc_metrics_supercells(), sc_metrics_global()

Examples

```
library(supercells)
# One variable

vol = terra::rast(system.file("raster/volcano.tif", package = "supercells"))
vol_slic1 = sc_slic(vol, step = 8, compactness = 1)
terra::plot(vol)
plot(sf::st_geometry(vol_slic1), add = TRUE, lwd = 0.2)
```

---

sc_slic_convergence *SLIC convergence diagnostics*

Description

Runs SLIC and returns per-iteration mean combined distance. The output can be plotted directly with plot().

Usage

```
sc_slic_convergence(
  x,
  step = NULL,
  compactness,
  dist_fun = "euclidean",
  avg_fun = "mean",
  clean = FALSE,
  minarea,
  iter = 10,
  k = NULL,
  centers = NULL,
  verbose = 0
)
```

Arguments

x	An object of class SpatRaster (terra) or class stars (stars).
step	Initial center spacing (alternative is k). Provide a plain numeric value for cell units, or use use_meters() for map-distance steps in meters (automatically converted to cells using raster resolution).
compactness	A compactness value. Use sc_tune_compactness() to estimate it. Use use_adaptive() to enable adaptive compactness based on local feature-space distances.

dist_fun	A supported distance name from sc_list_distances() or a custom function that takes two numeric vectors and returns one finite, nonnegative value. Smaller values must mean greater similarity.
avg_fun	An averaging function name or custom function used to summarize values within each supercell. Supported names: "mean" and "median". A custom function must accept a numeric vector and return a single numeric value.
clean	Connectivity cleaning is not applied to convergence diagnostics. Setting this to TRUE produces a warning and is otherwise ignored.
minarea	Ignored because connectivity cleaning is not applied.
iter	Number of iterations.
k	The number of supercells desired (alternative to step).
centers	Optional sf object of custom centers. Requires step.
verbose	Verbosity level.

Value

A data frame with class sc_slic_convergence and columns:

iter Iteration number.

mean_distance Mean combined distance across assigned cells at each iteration.

See Also

sc_slic(), plot()

Examples

```
library(supercells)
vol = terra::rast(system.file("raster/volcano.tif", package = "supercells"))
conv = sc_slic_convergence(vol, step = 8, compactness = 5, iter = 10)
plot(conv)
```

---

sc_slic_get_params *Get stored* sc_slic() *parameters*

Description

Returns key sc_slic() parameters stored as attributes on a supercells object.

Usage

```
sc_slic_get_params(sc)
```

Arguments

sc	An sf object returned by sc_slic().

Value

A one-row data.frame with columns: `step`, `compactness`, `compactness_method`, and `dist_fun`. The `dist_fun` column is character; custom distance functions are stored as `NA`.

See Also

`sc_slic()`, `sc_slic_set_params()`

`sc_slic_points` *Create supercells as points*

Description

Runs the SLIC workflow and returns supercell centers as points. Use `iter = 0` to return the initial centers before iterations. For polygon outputs, use `sc_slic()`; for raster output, use `sc_slic_raster()` By default, returns the same fields as `sc_slic()`: `outcomes = c("supercells", "coordinates", "values")`. Use `outcomes = "values"` for value summaries only.

Usage

```
sc_slic_points(
  x,
  step = NULL,
  compactness,
  dist_fun = "euclidean",
  avg_fun = "mean",
  clean = TRUE,
  minarea,
  iter = 10,
  k = NULL,
  centers = NULL,
  outcomes = c("supercells", "coordinates", "values"),
  chunks = FALSE,
  verbose = 0
)
```

Arguments

`x`	An object of class SpatRaster (terra) or class stars (stars).
`step`	Initial center spacing (alternative is `k`). Provide a plain numeric value for cell units, or use `use_meters()` for map-distance steps in meters (automatically converted to cells using raster resolution).
`compactness`	A compactness value. Use `sc_tune_compactness()` to estimate it. Use `use_adaptive()` to enable adaptive compactness based on local feature-space distances.
`dist_fun`	A supported distance name from `sc_list_distances()` or a custom function that takes two numeric vectors and returns one finite, nonnegative value. Smaller values must mean greater similarity.

avg_fun	An averaging function name or custom function used to summarize values within each supercell. Supported names: "mean" and "median". A custom function must accept a numeric vector and return a single numeric value.
clean	Should connectivity of the supercells be enforced?
minarea	When clean = TRUE, connected components containing minarea cells or fewer are merged into an adjacent component. If minarea is omitted or set to 0, the threshold is selected automatically as one quarter of the raster cell count divided by the number of initialized centers: floor(ncell / (4 * n_centers)).
iter	Number of iterations.
k	The number of supercells desired (alternative to step).
centers	Optional sf object of custom centers. Requires step.
outcomes	Character vector controlling which fields are returned. Allowed values are "supercells", "coordinates", and "values". Default is full output (c("supercells", "coordinates", "values")). Use outcomes = "values" for value summaries only.
chunks	Chunking option. Use FALSE for no chunking, TRUE for automatic chunking based on size, or a numeric value for a fixed chunk size (in number of cells per side).
verbose	Verbosity level.

Value

An sf object with supercell center points and summary statistics

See Also

sc_slic(), sc_slic_raster()

Examples

```
library(supercells)
vol = terra::rast(system.file("raster/volcano.tif", package = "supercells"))

# initial centers (only after local minima placement, no iterations)
init_pts = sc_slic_points(vol, step = 12, compactness = 1, iter = 0)
terra::plot(vol)
plot(sf::st_geometry(init_pts), add = TRUE, pch = 3, col = "red")

# final supercell centers
vol_pts = sc_slic_points(vol, step = 12, compactness = 1)
terra::plot(vol)
plot(sf::st_geometry(vol_pts), add = TRUE, pch = 16, col = "red")
```

sc_slic_raster *Create supercells as a raster*

Description

Runs the SLIC workflow and returns a raster of supercell IDs IDs are 1-based and are unique across chunks when chunking is used For polygon outputs, use `sc_slic()`; for point centers, use `sc_slic_points()`

Usage

```
sc_slic_raster(
  x,
  step = NULL,
  compactness,
  dist_fun = "euclidean",
  avg_fun = "mean",
  clean = TRUE,
  minarea,
  iter = 10,
  k = NULL,
  centers = NULL,
  outcomes = "supercells",
  chunks = FALSE,
  verbose = 0
)
```

Arguments

x	An object of class SpatRaster (terra) or class stars (stars).
step	Initial center spacing (alternative is k). Provide a plain numeric value for cell units, or use `use_meters()` for map-distance steps in meters (automatically converted to cells using raster resolution).
compactness	A compactness value. Use `sc_tune_compactness()` to estimate it. Use `use_adaptive()` to enable adaptive compactness based on local feature-space distances.
dist_fun	A supported distance name from `sc_list_distances()` or a custom function that takes two numeric vectors and returns one finite, nonnegative value. Smaller values must mean greater similarity.
avg_fun	An averaging function name or custom function used to summarize values within each supercell. Supported names: "mean" and "median". A custom function must accept a numeric vector and return a single numeric value.
clean	Should connectivity of the supercells be enforced?
minarea	When `clean = TRUE`, connected components containing `minarea` cells or fewer are merged into an adjacent component. If `minarea` is omitted or set to `0`, the threshold is selected automatically as one quarter of the raster cell count divided by the number of initialized centers: `floor(ncell / (4 * n_centers))`.

iter	Number of iterations.
k	The number of supercells desired (alternative to step).
centers	Optional sf object of custom centers. Requires step.
outcomes	Character vector controlling which fields are returned. Only "supercells" is supported in sc_slic_raster().
chunks	Chunking option. Use FALSE for no chunking, TRUE for automatic chunking based on size, or a numeric value for a fixed chunk size (in number of cells per side).
verbose	Verbosity level.

Value

A SpatRaster with supercell IDs.

See Also

sc_slic(), sc_slic_points()

Examples

```
library(supercells)
vol = terra::rast(system.file("raster/volcano.tif", package = "supercells"))
vol_ids = sc_slic_raster(vol, step = 8, compactness = 1)
terra::plot(vol_ids)
```

---

sc_slic_set_params *Set stored* sc_slic() *parameters*

Description

Writes key sc_slic() parameters to attributes on a supercells object.

Usage

```
sc_slic_set_params(sc, params)
```

Arguments

sc	An sf object.
params	A data.frame, typically from sc_slic_get_params(). Only the first row is used.

Value

The input object with updated attributes.

See Also

sc_slic(), sc_slic_get_params()

sc_tune_compactness *Estimate a compactness value*

Description

Estimates a compactness value for a chosen raster scale. The current implementation supports one tuning metric, "local_variability", which estimates compactness directly from local value variability.

Usage

```
sc_tune_compactness(
  x,
  step = NULL,
  dist_fun = "euclidean",
  metric = "local_variability",
  stat = stats::median,
  k = NULL,
  centers = NULL,
  ...
)
```

Arguments

x	A SpatRaster (terra) or stars object.
step	Initial center spacing (alternative is k). Provide a plain numeric value for cell units, or use use_meters() for map-distance steps in meters (automatically converted to cells using raster resolution).
dist_fun	A supported distance name from sc_list_distances() or a custom function that takes two numeric vectors and returns one finite, nonnegative value. Smaller values must mean greater similarity.
metric	Which compactness metric to return. Currently only "local_variability" is supported. The argument is kept for future additions. For "local_variability", compactness = median(local_mean(d_value)) / dim_scale, where local means are computed in windows around initial centers and dim_scale is inferred from the number of raster layers and dist_fun. Root-sum-square measures use sqrt(n_layers), additive and path-sum measures use n_layers, and other named or custom measures use 1. This keeps compactness adjustable to dimensionality without requiring a separate user-facing scaling argument.
stat	The summary statistic to use for the local variability metric. Default is median. If it is NULL, all local variability values will be returned without summarization.
k	The number of supercells desired (alternative to step).
centers	Optional sf object of custom initial centers. Requires step.
...	Additional arguments for the compactness metric. For example, dim_scale for "local_variability" if you want to override the default scaling.

Value

A data frame with columns `step`, `metric`, `dist_fun`, and `compactness`. With the default `stat = median`, it contains one row. With `stat = NULL`, it contains one row per initial center, with `step`, `metric`, and `dist_fun` repeated for each row.

See Also

`sc_slic()`, `use_meters()`, `use_adaptive()`

Examples

```
library(terra)
vol = rast(system.file("raster/volcano.tif", package = "supercells"))
tune = sc_tune_compactness(vol, step = 8)
tune$compactness
```

---

supercells *Creates supercells*

Description

Creates supercells by bringing the idea of superpixels into spatial analysis and grouping raster cells into coherent regions. It uses a modified version of the SLIC algorithm by Achanta et al. (2012), allowing specification of a distance function.

Usage

```
supercells(
  x,
  k,
  compactness,
  dist_fun = "euclidean",
  avg_fun = "mean",
  clean = TRUE,
  iter = 10,
  transform = NULL,
  step,
  minarea,
  metadata = TRUE,
  chunks = FALSE,
  verbose = 0
)
```

Arguments

x	An object of class SpatRaster (terra) or class stars (stars)
k	The number of supercells desired by the user (the output number can be slightly different!). You can use either k or step. It is also possible to provide a set of points (an sf object) as k together with the step value to create custom cluster centers.
compactness	A compactness value. Larger values cause supercells to be more compact/even (square). Use use_adaptive() to enable adaptive compactness based on local feature-space distances. A compactness value depends on the range of input cell values and selected distance measure.
dist_fun	A supported distance name from sc_list_distances() or a custom function that takes two numeric vectors and returns one finite, nonnegative value. Smaller values must mean greater similarity. Default: "euclidean".
avg_fun	An averaging function - how the values of the supercells' centers are calculated? The algorithm internally implements common functions "mean" and "median" (provided with quotation marks), but also accepts any fitting R function (e.g., base::mean() or stats::median(), provided as plain function name: mean). Default: "mean". See details for more information.
clean	Should connectivity of the supercells be enforced?
iter	The number of iterations performed to create the output.
transform	Transformation to be performed on the input. By default, no transformation is performed. Currently available transformation is "to_LAB": first, the conversion from RGB to the LAB color space is applied, then the supercells algorithm is run, and afterward, a reverse transformation is performed on the obtained results. (This argument is experimental and may be removed in the future).
step	Initial center spacing. You can use either k or step. Provide a plain numeric value for cell units, or use use_meters() for map-distance steps in meters (automatically converted to cells using raster resolution).
minarea	When clean = TRUE, connected components containing minarea cells or fewer are merged into an adjacent component. If minarea is omitted or set to 0, the threshold is selected automatically as one quarter of the raster cell count divided by the number of initialized centers: floor(ncell / (4 * n_centers)).
metadata	Logical. Controls whether metadata columns ("supercells", "x", "y") are included.
chunks	Should the input (x) be processed with chunking before deriving supercells? Use FALSE (default), TRUE (automatic chunking based on size), or a numeric value (chunk side length in number of cells). When TRUE, the memory limit can be set with options(supercells.chunk_mem_gb = 4).
verbose	An integer specifying the level of text messages printed during calculations. 0 means no messages (default), 1 provides basic messages (e.g., calculation stage).

Details

If you want to use additional arguments for the averaging function (avg_fun), you can create a custom function. For example, if you want to calculate the mean by removing missing values,

you can use the following code: `my_mean = function(x) mean(x, na.rm = TRUE)` and then provide `avg_fun = my_mean`. For raster IDs or point centers outputs, see `sc_slic_raster()` and `sc_slic_points()`. For evaluation and diagnostics, see `sc_metrics_pixels()`, `sc_metrics_supercells()`, and `sc_metrics_global()`.

Value

An sf object with several columns: (1) supercells - an id of each supercell, (2) y and x coordinates, (3) one or more columns with average values of given variables in each supercell.

References

Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., & Süsstrunk, S. (2012). SLIC Superpixels Compared to State-of-the-Art Superpixel Methods. IEEE Transactions on Pattern Analysis and Machine Intelligence, 34(11), 2274–2282. https://doi.org/10.1109/tpami.2012.120

Nowosad, J. Motif: an open-source R tool for pattern-based spatial analysis. Landscape Ecol (2021). https://doi.org/10.1007/s10980-020-01135-0

See Also

`sc_slic()`

Examples

```
library(supercells)
# One variable

vol = terra::rast(system.file("raster/volcano.tif", package = "supercells"))
vol_slic1 = supercells(vol, k = 50, compactness = 1)
terra::plot(vol)
plot(sf::st_geometry(vol_slic1), add = TRUE, lwd = 0.2)

# RGB variables
# ortho = terra::rast(system.file("raster/ortho.tif", package = "supercells"))
# ortho_slic1 = supercells(ortho, k = 1000, compactness = 10, transform = "to_LAB")
# terra::plot(ortho)
# plot(sf::st_geometry(ortho_slic1), add = TRUE)
#
# ### RGB variables - colored output
#
# rgb_to_hex = function(x){
#   apply(t(x), 2, function(x) rgb(x[1], x[2], x[3], maxColorValue = 255))
# }
# avg_colors = rgb_to_hex(sf::st_drop_geometry(ortho_slic1[4:6]))
#
# terra::plot(ortho)
# plot(sf::st_geometry(ortho_slic1), add = TRUE, col = avg_colors)
```

use_adaptive *Use adaptive compactness mode*

Description

Creates a compactness mode object for adaptive compactness. The `"local_max"` method scales value distances separately for each center using the largest feature-space distance within its local search window. The scale is updated before each assignment iteration, while spatial distances remain scaled by `step`.

Usage

```
use_adaptive(method = "local_max")
```

Arguments

`method`	Adaptive compactness method. Currently only `"local_max"` is supported.

Value

An adaptive compactness mode object for `compactness` arguments.

Examples

```
use_adaptive()
```

use_meters *Mark step values as meters*

Description

Creates a units value in meters for use in `step` arguments. Use plain numerics for cell units, and `use_meters()` for map-distance steps.

Usage

```
use_meters(x)
```

Arguments

`x`	A single positive numeric value.

Value

A units::units object in meters (`m`).

Examples

```
use_meters(100)
```

Index

plot(), *10, 11*

sc_list_distances, 2
sc_list_distances(), *8*, *11*, *12*, *14*, *16*, *18*
sc_metrics_global, 3
sc_metrics_global(), *6*, *7*, *9*, *10*, *19*
sc_metrics_pixels, 5
sc_metrics_pixels(), *4*, *7*, *9*, *10*, *19*
sc_metrics_supercells, 6
sc_metrics_supercells(), *4*, *6*, *9*, *10*, *19*
sc_slic, 8
sc_slic(), *3–7*, *9*, *11–15*, *17*, *19*
sc_slic_convergence, 10
sc_slic_convergence(), *9*, *10*
sc_slic_get_params, 11
sc_slic_get_params(), *15*
sc_slic_points, 12
sc_slic_points(), *9*, *10*, *14*, *15*, *19*
sc_slic_raster, 14
sc_slic_raster(), *9*, *10*, *12*, *13*, *19*
sc_slic_set_params, 15
sc_slic_set_params(), *12*
sc_tune_compactness, 16
sc_tune_compactness(), *8*, *10*, *12*, *14*
supercells, 17

units::units, *20*
use_adaptive, 20
use_adaptive(), *8*, *10*, *12*, *14*, *17*, *18*
use_meters, 20
use_meters(), *8*, *10*, *12*, *14*, *16–18*