Package 'segmetric'

October 14, 2022

Title Metrics for Assessing Segmentation Accuracy for Geospatial Data

Version 0.2.0

Description A system that computes metrics to assess the segmentation accuracy of geospatial data. These metrics calculate the discrepancy between segmented and reference objects, and indicate the segmentation accuracy. For more details on choosing evaluation metrics, we suggest seeing Costa et al. (2018) <doi:10.1016/j.rse.2017.11.024> and Jozdani et al. (2020) <doi:10.1016/j.isprsjprs.2020.01.002>.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.1.2

Imports magrittr, graphics, sf, units

Suggests classInt, dplyr, testthat (>= 3.0.0)

Depends R (>= 2.10)

NeedsCompilation no

Author Rolf Simoes [aut] (<https://orcid.org/0000-0003-0953-4132>), Alber Sanchez [aut] (<https://orcid.org/0000-0001-7966-2880>), Michelle Picoli [aut, cre] (<https://orcid.org/0000-0001-9855-2046>)

Maintainer Michelle Picoli <mipicoli@gmail.com>

Repository CRAN

Date/Publication 2022-03-30 16:20:02 UTC

R topics documented:

segmetric-package 2
general_functions 3
metric_functions 5
plot 7
ref_sf 9
seg_sf 10

set_functions 11
subset_handling_functions 13
%>% 15

Index **17**

`segmetric-package` *segmetric*

Description

Metrics for assessing segmentation accuracy for geospatial data.

Purpose

The `segmetric` package provides a set of metrics for the segmentation accuracy assessment (or evaluation) of geospatial data. It includes more than 20 metrics used in the literature for spatial segmentation assessment (Van Rijsbergen, 1979; Levine and Nazif, 1982; Janssen and Molenaar, 1995; Lucieer and Stein, 2002; Carleer et al., 2005; Moller et al., 2007; van Coillie et al., 2008; Costa et al., 2008; Weidner, 2008; Feitosa et al., 2010; Clinton et al. 2010; Persello and Bruzzone, 2010; Yang et al., 2014; and Zhang et al., 2015).

Extensions

The `segmetric` package is extensible and provides a set of functions to ease the implementation of new metrics. See `?sm_reg_metric()` to find how new metrics are implemented.

Contributions

Contribution to this package could be done at `segmetric`'s page on GitHub: https://github.com/michellepicoli/segmetric.

Author(s)

Maintainer: Michelle Picoli <mipicoli@gmail.com> (ORCID)

Authors:

- Rolf Simoes <rolfsimoes@gmail.com> (ORCID)
- Alber Sanchez <albhasan@gmail.com> (ORCID)

References

- Carleer, A.P., Debeir, O., Wolff, E., 2005. Assessment of very high spatial resolution satellite image segmentations. Photogramm. Eng. Remote. Sens. 71, 1285-1294. doi: 10.14358/PERS.71.11.1285.
- Clinton, N., Holt, A., Scarborough, J., Yan, L., Gong, P., 2010. Accuracy assessment measures for object-based image segmentation goodness. Photogramm. Eng. Remote. Sens. 76, pp. 289-299.

- Costa, G.A.O.P., Feitosa, R.Q., Cazes, T.B., Feijo, B., 2008. Genetic adaptation of segmentation parameters. In: Blaschke, T., Lang, S., Hay, G.J. (Eds.), Object-based Image Analysis. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 679-695. doi: 10.1007/978354077058-9_37.
- Feitosa, R.Q., Ferreira, R.S., Almeida, C.M., Camargo, F.F., Costa, G.A.O.P., 2010. Similarity metrics for genetic adaptation of segmentation parameters. In: 3rd International Conference on Geographic Object-Based Image Analysis (GEOBIA 2010). The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Ghent.
- Janssen, L.L.F., Molenaar, M., 1995. Terrain objects, their dynamics and their monitoring by the integration of GIS and remote sensing. IEEE Trans. Geosci. Remote Sens. 33, pp. 749-758. doi: 10.1109/36.387590.
- Levine, M.D., Nazif, A.M., 1982. An experimental rule based system for testing low level segmentation strategies. In: Preston, K., Uhr, L. (Eds.), Multicomputers and Image Processing: Algorithms and Programs. Academic Press, New York, pp. 149-160.
- Lucieer, A., Stein, A., 2002. Existential uncertainty of spatial objects segmented from satellite sensor imagery. Geosci. Remote. Sens. IEEE Trans. 40, pp. 2518-2521. doi: 10.1109/TGRS.2002.805072.
- Moller, M., Lymburner, L., Volk, M., 2007. The comparison index: a tool for assessing the accuracy of image segmentation. Int. J. Appl. Earth Obs. Geoinf. 9, pp. 311-321. doi: 10.1016/j.jag.2006.10.002.
- Persello, C., Bruzzone, L., 2010. A novel protocol for accuracy assessment in classification of very high resolution images. IEEE Trans. Geosci. Remote Sens. 48, pp. 1232-1244. doi: 10.1109/TGRS.2009.2029570.
- Van Coillie, F.M.B., Verbeke, L.P.C., De Wulf, R.R., 2008. Semi-automated forest stand delineation using wavelet based segmentation of very high resolution optical imagery. In: Object-Based Image Analysis: Spatial Concepts for Knowledge-Driven Remote Sensing Applications, pp. 237-256. doi: 10.1007/9783540770589_13.
- Van Rijsbergen, C.J., 1979. Information Retrieval. Butterworth-Heinemann, London.
- Weidner, U., 2008. Contribution to the assessment of segmentation quality for remote sensing applications. Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci. 37, pp. 479-484.
- Yang Yang, J., Li, P., He, Y., 2014. A multi-band approach to unsupervised scale parameter selection for multi-scale image segmentation. ISPRS J. Photogramm. Remote Sens. 94, pp. 13-24. doi: 10.1016/j.isprsjprs.2014.04.008.
- Zhang, X., Feng, X., Xiao, P., He, G., Zhu, L., 2015a. Segmentation quality evaluation using region-based precision and recall measures for remote sensing images. ISPRS J. Photogramm. Remote Sens. 102, pp. 73-84. doi: 10.1016/j.isprsjprs.2015.01.009.

`general_functions` *General functions*

Description

These functions manipulate segmetric objects.

- `sm_area()`: Return a vector of areas, one for each polygon.
- `sm_centroid()`: Return the centroids of the given polygons.
- `sm_intersection()`: Return the intersection of the given simple features.
- `sm_subset_union()`: Return the union of the given simple features.
- `sm_rbind()`: Return the merge of unique simple features.

Usage

```
sm_area(s, order = NULL)

sm_centroid(s)

sm_intersection(s1, s2, touches = TRUE)

sm_subset_union(s)

sm_rbind(...)
```

Arguments

`s, s1, s2`	Either a `ref_sf`, a `seg_sf`, or a `subset_sf` object (inherited from `sf`).
`order`	A `subset_sf`. This argument arranges the returned values according to the object passed here.
`touches`	A `logical`. Is the border part of the intersection?
`...`	For `sm_rbind()`, a set of `subset_sf` objects to be merged.

Value

- `sm_area()`: Return a `numeric` vector with polygons' area.
- `sm_centroid()`: Return a `subset_sf` object with polygons' centroid.
- `sm_intersection()`: Return a `subset_sf` object with intersection between polygons.
- `sm_subset_union()`: Return a `subset_sf` object with union between intersecting polygons.
- `sm_rbind()`: Return a `subset_sf` object with unique features.

`metric_functions` *Metric functions*

Description

The `sm_compute()` computes a given metric (`metric_id` parameter) from segmentation objects. It compares the reference to the segmentation polygons using a metric.

A list with all supported metrics can be obtained by `sm_list_metrics()` (see Details for more information).

The `sm_metric_subset()` returns the subset used to compute the metrics in segmetric object.

Usage

```
sm_compute(m, metric_id, ...)

sm_metric_subset(m, metric_id = NULL)
```

Arguments

`m`	A `segmetric` object.
`metric_id`	A `character` vector with metrics id to be computed.
`...`	Any additional argument to compute a metric (see Details).

Details

- "`OS1`" refers to Oversegmentation. Its values range from 0 (optimal) to 1 (Clinton et al., 2010).
- "`US1`" refers to Undersegmentation. Its values range from 0 (optimal) to 1 (Clinton et al., 2010).
- "`OS2`" refers to Oversegmentation. Its values range from 0 (optimal) to 1 (Persello and Bruzzone, 2010).
- "`US2`" refers to Undersegmentation. Its values range from 0 (optimal) to 1 (Persello and Bruzzone, 2010).
- "`OS3`" refers to Oversegmentation. Its values range from 0 (optimal) to 1 (Yang et al., 2014).
- "`US3`" refers to Undersegmentation. Its values range from 0 (optimal) to 1 (Yang et al., 2014).
- "`AFI`" refers to Area Fit Index. Its optimal value is 0 (Lucieer and Stein, 2002; Clinton et al., 2010).
- "`QR`" refers to Quality Rate. Its values range from 0 (optimal) to 1 (Weidner, 2008; Clinton et al., 2010).
- "`D_index`" refers to Index D. Its values range from 0 (optimal) to 1 (Levine and Nazif, 1982; Clinton et al., 2010).
- "`precision`" refers to Precision. Its values range from 0 to 1 (optimal) (Van Rijsbergen, 1979; Zhang et al., 2015).

- "recall" refers to Recall. Its values range from 0 to 1 (optimal) (Van Rijsbergen, 1979; Zhang et al., 2015).
- "UMerging" refers to Undermerging. Its values range from 0 (optimal) to 1 (Levine and Nazif, 1982; Clinton et al., 2010).
- "OMerging" refers to Overmerging. Its optimal value is 0 (Levine and Nazif, 1982; Clinton et al., 2010).
- "M" refers to Match. Its values range from 0 to 1 (optimal) (Janssen and Molenaar, 1995; Feitosa et al., 2010).
- "E" refers to Evaluation Measure. Its values range from 0 (optimal) to 100 (Carleer et al., 2005).
- "RAsub" refers to Relative Area. Its values range from 0 to 1 (optimal) (Müller et al., 2007; Clinton et al., 2010).
- "RAsuper" refers to Relative area. Its values range from 0 to 1 (optimal) (Müller et al., 2007; Clinton et al., 2010).
- "PI" refers to Purity Index. Its values range from 0 to 1 (optimal) (van Coillie et al., 2008).
- "Fitness" refers to Fitness Function. Its optimal value is 0 (Costa et al., 2008).
- "ED3" refers to Euclidean Distance. Its values range from 0 (optimal) to 1 (Yang et al., 2014).
- "F_measure" refers to F-measure metric. Its values range from 0 to 1 (optimal) (Van Rijsbergen, 1979; Zhang et al., 2015). It takes the optional weight argument alpha, ranging from 0.0 to 1.0 (the default is 0.5).

Value

Return a numeric vector with computed metric.

References

A complete list of cited references is available in ?segmetric.

See Also

sm_list_metrics()

Examples

```
# load sample datasets
data("sample_ref_sf", package = "segmetric")
data("sample_seg_sf", package = "segmetric")

# create segmetric object
m <- sm_read(ref_sf = sample_ref_sf, seg_sf = sample_seg_sf)

# compute AFI metric and summarize it
sm_compute(m, "AFI") %>% summary()

# compute three metrics and summarize them
sm_compute(m, c("AFI", "OS1", "US2")) %>% summary()
```

```
# compute OS1, F_measure, and US2 metrics using pipe
m <- sm_compute(m, "OS1") %>%
  sm_compute("F_measure") %>%
  sm_compute("US2")

# summarize them
summary(m)
```

---

plot *Plot function*

Description

Plot a segmetric map according to the parameter `type`:

- `"base"`: simple plot of the reference or segmentation polygons;
- `"subset"`: plot polygons from a subset over the base plot;
- `"choropleth"`: plot a choropleth map from polygons of a subset using metric values.

Usage

```
## S3 method for class 'segmetric'
plot(
  x,
  type = "base",
  ...,
  title = NULL,
  background = "#FAFAFA",
  layers = c("ref_sf", "seg_sf"),
  ref_color = "#FF00009F",
  ref_fill = "#FFFFFF00",
  ref_label = "reference",
  ref_size = 2,
  ref_symbol = 2,
  seg_color = "#0000009F",
  seg_fill = "#FFFFFF00",
  seg_label = "segment",
  seg_size = 1,
  seg_symbol = 3,
  selected_fill = "#9A9AFF50",
  plot_centroids = TRUE,
  centroids_color = "#000000FF",
  centroids_label = "centroid",
  subset_id = NULL,
  subset_color = "#FFFFFF00",
```

```
  subset_fill = "#F0E4167F",
  metric_id = NULL,
  break_style = "jenks",
  plot_extent = NULL,
  plot_legend = TRUE,
  plot_axes = TRUE
)
```

Arguments

x	A segmetric object.
type	A character. Either "base", "subset", or "choropleth".
...	Ignored.
title	A character with plot title
background	A character with a valid hexadecimal color in rgb or rgba format. Set the map background color.
layers	A character. One or both of "ref_sf" and "seg_sf" (works only for type = "base" and type = "subset").
ref_color, seg_color, ref_fill, seg_fill	A character with a valid hexadecimal color in rgb or rgba format. Set the border and fill colors for reference and segmentation polygons.
ref_label, seg_label, centroids_label	A character with legend labels for reference polygons, segmentation polygons, and centroids.
ref_size, seg_size	A numeric. Set symbol's size for centroids.
ref_symbol, seg_symbol	An integer. Symbol to represent polygons' centroids (see pch param in points).
selected_fill	A character with a valid hexadecimal color in rgb or rgba format. Set the fill color of selected reference or segmentation polygons depending on subset_id.
plot_centroids	A logical. Plot centroids or not.
centroids_color	A character with a valid hexadecimal color in rgb or rgba format. Set the border colors for centroids.
subset_id	A character with subset name (required for type = "subset")
subset_color, subset_fill	A character with a valid hexadecimal color in rgb or rgba format. Set the border and fill colors for subset polygons (works only with type = "subset").
metric_id	A character with metric to be plotted in choropleth maps (required for type = "choropleth")
break_style	A character with the name of a method to compute the intervals for choropleth maps. Can be one of "sd", "equal", "pretty", "quantile", "kmeans", "hclust", "bclust", "fisher", "jenks", "dpih", and "headtails" (see style parameter in classIntervals).

plot_extent	A sf object. Set the map extent for a plot.
plot_legend	A logical. Plot legend or not.
plot_axes	A logical. Plot coordinates axis or not.

---

ref_sf *LEM+ dataset*

Description

ref_sf: a dataset containing field boundaries from Luiz Eduardo Magalhaes municipality, Brazil.

The data covers the following extent: xmin: -46.37683 ymin: -12.34579 xmax: -46.15776 ymax: -12.13663 CRS: EPSG:4326

sample_ref_sf: a subset of ref_sf dataset.

Usage

```
ref_sf

sample_ref_sf
```

Format

These datasets are objects of class sf (inherited from tbl_df, tbl, data.frame) with 2 variables:

- id: identification
- geometry: polygons

ref_sf: a dataset with 195 features.

sample_ref_sf: a dataset with 5 features.

Source

Oldoni et al. (2020) doi: 10.1016/j.dib.2020.106553.

References

- Oldoni, L.V., Sanches, I.D.A., Picoli, M.C.A., Covre, R.M. and Fronza, J.G., 2020. LEM+ dataset: For agricultural remote sensing applications. Data in Brief, 33, p.106553.

Examples

```
data("ref_sf", package = "segmetric")
data("sample_ref_sf", package = "segmetric")
```

seg_sf *Segmentation dataset*

Description

seg200_sf,seg500_sf,seg800_sf,seg1000_sf: a dataset containing segments generated from PlanetScope image, level 3B, acquired on Feb 18, 2020, with 3.7-meter resolution (Planet Team, 2017), using the multiresolution segmentation method (Baatz and Schape, 2000).

The data covers the approximately the same area of LEM+ dataset (see ref_sf).

The data was post-processed using the spectral difference algorithm on band 3.

The polygons were simplified using the Douglas-Peucker algorithm in QGIS.

Self-intersections were removed using SAGA's Polygon Self-Intersection.

Segmentation parameters:

- scale parameter: 200 (seg200_sf), 500 (seg500_sf), 800 (seg800_sf), and 1000 (seg1000_sf)
- shape: 0.9
- compactness: 0.1

Spectral difference parameters:

- spectral difference: 20

Simplification parameter:

- distance: 10-meters

Only those polygons intersecting reference data with an area-perimeter ratio above 25 were selected.

sample_seg_sf: a subset of seg_sf dataset.

Usage

```
seg200_sf

seg500_sf

seg800_sf

seg1000_sf

sample_seg_sf
```

Format

These datasets are objects of class `sf` (inherited from `tbl_df`, `tbl`, `data.frame`) with 2 variables:

- `id`: identification
- `geometry`: polygons

`seg200_sf`: a dataset with 547 features. `seg500_sf`: a dataset with 215 features. `seg800_sf`: a dataset with 169 features. `seg1000_sf`: a dataset with 158 features.

An object of class `sf` (inherits from `tbl_df`, `tbl`, `data.frame`) with 215 rows and 2 columns.

An object of class `sf` (inherits from `tbl_df`, `tbl`, `data.frame`) with 169 rows and 2 columns.

An object of class `sf` (inherits from `tbl_df`, `tbl`, `data.frame`) with 158 rows and 2 columns.

`sample_seg_sf`: a dataset with 6 features extracted from `seg500_sf` dataset.

References

- Planet Team, 2017. Planet Application Program Interface: In Space for Life on Earth. San Francisco, CA. https://api.planet.com
- Baatz, M., Schape, A., 2000. Multiresolution segmentation - an optimization approach for high quality multi-scale image segmentation. In: Strobl, J., Blaschke, T., Griesebner, G. (Eds.), Angewandte Geographische Informations-Verarbeitung XII. Wichmann Verlag, Karlsruhe, Germany, pp. 12-23. <>

Examples

```
data("seg200_sf", package = "segmetric")
data("seg500_sf", package = "segmetric")
data("seg800_sf", package = "segmetric")
data("seg1000_sf", package = "segmetric")
data("sample_seg_sf", package = "segmetric")
```

`set_functions` *Set functions*

Description

These functions compute subsets required to calculate segmentation metrics as described in Clinton et al. (2010) and Costa et al. (2017).

- `sm_ref()` returns the set of n polygons of reference, represented by $X = \{x_i : i = 1, ..., n\}$
- `sm_seg()` returns the set of m segmentation polygons, represented by $Y = \{y_j : j = 1, ..., m\}$
- `sm_ytilde()` returns $\tilde{Y}_i$, a subset of Y, where $\tilde{Y}_i = \{y_j : \text{area}(\text{x}_\text{i} \cap \text{y}_\text{j}) \neq 0\}$
- `sm_xtilde()` returns $\tilde{X}_j$, a subset of X, where $\tilde{X}_j = \{x_i : \text{area}(\text{y}_\text{j} \cap \text{x}_\text{i}) \neq 0\}$
- `sm_yprime()` returns Y'_i, a subset of Y, where $Y'_i = \{y_j : max(\text{area}(\text{x}_\text{i} \cap \text{y}_\text{j}))\}$
- `sm_xprime()` returns X'_j, a subset of X, where $X'_j = \{x_i : max(\text{area}(\text{y}_\text{j} \cap \text{x}_\text{i}))\}$

- `sm_ya()` returns Ya_i, a subset of $\tilde{Y}_i$, where $Ya_i = \{y_j : \text{centroid}(\text{x}_\text{i}) \text{ in } \text{y}_\text{j}\}$
- `sm_yb()` returns Yb_i, a subset of $\tilde{Y}_i$, where $Yb_i = \{y_j : \text{centroid}(\text{y}_\text{j}) \text{ in } \text{x}_\text{i}\}$
- `sm_yc()` returns Yc_i, a subset of $\tilde{Y}_i$, where $Yc_i = \{y_j : \text{area}(\text{x}_\text{i} \cap \text{y}_\text{j})/\text{area}(\text{y}_\text{j}) > 0.5\}$
- `sm_yd()` returns Yd_i, a subset of $\tilde{Y}_i$, where $Yd_i = \{y_j : \text{area}(\text{x}_\text{i} \cap \text{y}_\text{j})/\text{area}(\text{x}_\text{i}) > 0.5\}$
- `sm_ystar()` returns Y_i^*, where $Y_i^* = Ya_i \cup Yb_i \cup Yc_i \cup Yc_i$
- `sm_ycd()` returns Ycd_i, where $Ycd_i = Yc_i \cup Yd_i$
- `sm_ye()` returns Ye_i, a subset of $\tilde{Y}_i$, where $Ye_i = \{y_j : \text{area}(\text{x}_\text{i} \cap \text{y}_\text{j})/\text{area}(\text{y}_\text{j}) = 1\}$
- `sm_yf()` returns Yf_i, a subset of $\tilde{Y}_i$, where $Yf_i = \{y_j : \text{area}(\text{x}_\text{i} \cap \text{y}_\text{j})/\text{area}(\text{y}_\text{j}) > 0.55\}$
- `sm_yg()` returns Yg_i, a subset of $\tilde{Y}_i$, where $Yg_i = \{y_j : \text{area}(\text{x}_\text{i} \cap \text{y}_\text{j})/\text{area}(\text{y}_\text{j}) > 0.75\}$

Usage

```
sm_ytilde(m)

sm_xtilde(m)

sm_yprime(m)

sm_xprime(m)

sm_ya(m)

sm_yb(m)

sm_yc(m)

sm_yd(m)

sm_ystar(m)

sm_ycd(m)

sm_ye(m)

sm_yf(m)

sm_yg(m)
```

Arguments

`m`	A `segmetric` object.

Value

- `sm_ref()`: Return an object of class `ref_sf` (inherited from `sf`) containing identification (`ref_id`) and geometry (`geometry`) columns.

- sm_seg(): Return an object of class seg_sf (inherited from sf) containing identification (seg_id) and geometry (geometry) columns.
- sm_ytilde(), sm_xtilde(), sm_yprime(), sm_xprime(), sm_ya(), sm_yb(), sm_yc(), sm_yd(), sm_ystar(), sm_ycd(), sm_ye(), sm_yf(), and sm_yg(): Return an object of class subset_sf (inherited from sf) containing identification (ref_id and seg_id), and geometry (geometry) columns.

References

Clinton, N., Holt, A., Scarborough, J., Yan, L., & Gong, P. (2010). Accuracy Assessment Measures for Object-based Image Segmentation Goodness. Photogrammetric Engineering & Remote Sensing, 76(3), 289–299. doi: 10.14358/PERS.76.3.289.

Costa, H., Foody, G. M., & Boyd, D. S. (2018). Supervised methods of image segmentation accuracy assessment in land cover mapping. Remote Sensing of Environment, 205(December 2017), 338–351. doi: 10.1016/j.rse.2017.11.024.

---

subset_handling_functions

Subset handling functions

Description

These functions are intended to be used in new metric extensions. They handle subset_sf data (inherited from sf class) stored in segmetric objects.

- sm_list() lists subsets already computed and stored in a segmetric object.
- sm_exists() verifies if a subset_id exists in a segmetric object.
- sm_subset() evaluates and stores a subset_sf object.
- sm_indirect() finds the subset_id of a given subset_sf object stored in a segmetric object.
- sm_segmetric() returns the segmetric object that stores a given subset object (either a ref_sf, a seg_sf, or a subset_sf).
- sm_get() retrieves a subset_sf object stored in a segmetric object.
- sm_inset() operator equivalent to inner join but returns only objects from s1, or its corresponding row in s2 if parameter return_index is TRUE.
- sm_group_by(): Apply a function to groups of subset_sf.

Usage

```
sm_list(m)

sm_exists(m, subset_id)

sm_subset(m, subset_id, expr = NULL)
```

```
sm_indirect(s)

sm_segmetric(s)

sm_get(m, subset_id)

sm_ref(m)

sm_seg(m)

sm_inset(s1, s2, return_index = FALSE)

## S3 method for class 'ref_sf'
sm_inset(s1, s2, return_index = FALSE)

## S3 method for class 'seg_sf'
sm_inset(s1, s2, return_index = FALSE)

## S3 method for class 'subset_sf'
sm_inset(s1, s2, return_index = FALSE)

sm_group_by(s, by, fn, ...)
```

Arguments

`m`	A `segmetric` object.
`subset_id`	A `character` value informing a subset name.
`expr`	A valid piece of code in R inside curly braces. This code is evaluated to generate a subset.
`s`, `s1`, `s2`	Either a `ref_sf`, a `seg_sf`, or a `subset_sf` object.
`return_index`	A `logical` value indicating if the corresponding rows in `s1` should be returned instead of the actual corresponding values of `s2`.
`by`	A `character` value with the column to group.
`fn`	A `function` to apply on each group.
`...`	For `sm_group_by()`, extra parameter to pass to `fn` function.

Value

- `sm_list()`: Return a `character` vector with all names of subsets stored in the `segmetric` object.
- `sm_exists()`: Return a `logical` value indicating if a given subset name is stored in the `segmetric` object.
- `sm_subset()`: Return a `subset_sf` object.
- `sm_indirect()`: Return the subset name of a given `subset_sf` object stored in a `segmetric` object.
- `sm_segmetric()`: Return a `segmetric` object that stores a given `subset_sf` object.

- sm_get(): Return a subset_sf object stored in a segmetric object.
- sm_inset(): Return either a subset_sf object or an integer vector with the index of corresponding rows of s2 object.
- sm_group_by(): Return a subset_sf object.

Examples

```
# load sample datasets
data("sample_ref_sf", package = "segmetric")
data("sample_seg_sf", package = "segmetric")

# create segmetric object
m <- sm_read(ref_sf = sample_ref_sf, seg_sf = sample_seg_sf)

# lists only 'ref_sf' and 'seg_sf'
sm_list(m)

# computes 'Y_tilde' subset and stores it as 'test_subset' subset id
# sm_ytilde(m) also stores a subset under 'Y_tilde' id
s <- sm_subset(m, "test_subset", sm_ytilde(m))

# lists 'ref_sf', 'seg_sf', 'test_subset', and 'Y_tilde'
sm_list(m)

# which segmetric object stores `s` subset?
m2 <- sm_segmetric(s)

# m is identical to m2
identical(m, m2)

# which name `s` subset is stored in `m` segmetric object?
sm_indirect(s)

# retrieve 'test_subset' data from `m` object
s2 <- sm_get(m, 'test_subset')

# s is identical to s2
identical(s, s2)
```

%>% *Pipe*

Description

Imported function from magrittr.

Arguments

lhs	left hand side parameter.
rhs	right hand side parameter, usually a function.

Value

Return any object evaluated by function call `rhs(lhs, ...)`.

Index

∗ **datasets**
    ref_sf, 9
    seg_sf, 10
%>%, 15
_PACKAGE *(*segmetric-package*)*, 2

classIntervals, *8*

general_functions, 3

metric_functions, 5

plot, 7
points, *8*

ref_sf, 9, *10*

sample_ref_sf *(*ref_sf*)*, 9
sample_seg_sf *(*seg_sf*)*, 10
seg1000_sf *(*seg_sf*)*, 10
seg200_sf *(*seg_sf*)*, 10
seg500_sf *(*seg_sf*)*, 10
seg800_sf *(*seg_sf*)*, 10
seg_sf, 10
segmetric *(*segmetric-package*)*, 2
segmetric-package, 2
set_functions, 11
sm_area *(*general_functions*)*, 3
sm_centroid *(*general_functions*)*, 3
sm_compute *(*metric_functions*)*, 5
sm_exists *(*subset_handling_functions*)*, 13
sm_get *(*subset_handling_functions*)*, 13
sm_group_by *(*subset_handling_functions*)*, 13
sm_indirect *(*subset_handling_functions*)*, 13
sm_inset *(*subset_handling_functions*)*, 13
sm_intersection *(*general_functions*)*, 3
sm_list *(*subset_handling_functions*)*, 13
sm_metric_subset *(*metric_functions*)*, 5
sm_rbind *(*general_functions*)*, 3
sm_ref *(*subset_handling_functions*)*, 13
sm_seg *(*subset_handling_functions*)*, 13
sm_segmetric *(*subset_handling_functions*)*, 13
sm_subset *(*subset_handling_functions*)*, 13
sm_subset_union *(*general_functions*)*, 3
sm_xprime *(*set_functions*)*, 11
sm_xtilde *(*set_functions*)*, 11
sm_ya *(*set_functions*)*, 11
sm_yb *(*set_functions*)*, 11
sm_yc *(*set_functions*)*, 11
sm_ycd *(*set_functions*)*, 11
sm_yd *(*set_functions*)*, 11
sm_ye *(*set_functions*)*, 11
sm_yf *(*set_functions*)*, 11
sm_yg *(*set_functions*)*, 11
sm_yprime *(*set_functions*)*, 11
sm_ystar *(*set_functions*)*, 11
sm_ytilde *(*set_functions*)*, 11
subset_handling_functions, 13