Package ‘rmapshaper’

May 26, 2021

Type  Package
Title  Client for 'mapshaper' for 'Geospatial' Operations
Version  0.4.5
Description  Edit and simplify 'geojson', 'Spatial', and 'sf' objects. This is wrapper around the 'mapshaper' 'JavaScript' library by Matthew Bloch <https://github.com/mbloch/mapshaper/> to perform topologically-aware polygon simplification, as well as other operations such as clipping, erasing, dissolving, and converting 'multi-part' to 'single-part' geometries. It relies on the 'geojsonio' package for working with 'geojson' objects, the 'sf' package for working with 'sf' objects, and the 'sp' and 'rgdal' packages for working with 'Spatial' objects.
License  MIT + file LICENSE
URL  https://github.com/ateucher/rmapshaper
BugReports  https://github.com/ateucher/rmapshaper/issues
Imports  geojsonio (>= 0.9.4), geojsonlint (>= 0.4.0), jsonlite (>= 1.7.0), methods, readr (>= 1.4.0), sf (>= 0.9-0), sp (>= 1.4-0), V8 (>= 3.4.2)
Suggests  knitr, magrittr, rgdal, rgeos, rmarkdown, testthat (>= 2.1.0), covr
VignetteBuilder  knitr
Encoding  UTF-8
RoxygenNote  7.1.1
NeedsCompilation  no
Author  Andy Teucher [aut, cre],
        Kenton Russell [aut] (JavaScript support),
        Matthew Bloch [cph] (mapshaper Javascript library)
Maintainer  Andy Teucher <andy.teucher@gmail.com>
Repository  CRAN
Date/Publication  2021-05-26 18:50:02 UTC
**R topics documented:**

- apply_mapshaper_commands .................................................. 2
- check_sys_mapshaper ...................................................... 3
- drop_null_geometries ....................................................... 3
- ms_clip ................................................................. 4
- ms_dissolve ........................................................... 5
- ms_erase ................................................................. 7
- ms_explode ............................................................ 9
- ms_filter_fields ......................................................... 11
- ms_filter_islands ....................................................... 12
- ms_innerlines .......................................................... 13
- ms_lines ............................................................... 15
- ms_points ............................................................. 16
- ms_simplify .......................................................... 18
- rmapshaper .......................................................... 20

---

**apply_mapshaper_commands**

*Apply a mapshaper command string to a geojson object*

**Description**

Apply a mapshaper command string to a geojson object

**Usage**

```r
apply_mapshaper_commands(data, command, force_FC, sys = FALSE, sys_mem = 8)
```

**Arguments**

- `data` geojson object or path to geojson file. If a file path, sys must be true
- `command` valid mapshaper command string
- `force_FC` should the output be forced to be a FeatureCollection (or SpatialDataFrame) even if there are no attributes? Default TRUE. FeatureCollections are more compatible with rgdal::readOGR and geojsonio::geojson_sp. If FALSE and there are no attributes associated with the geometries, a GeometryCollection (or Spatial object with no dataframe) will be output.
- `sys` Should the system mapshaper be used instead of the bundled mapshaper? Gives better performance on large files. Requires the mapshaper node package to be installed and on the PATH.
- `sys_mem` How much memory (in GB) should be allocated if using the system mapshaper (sys = TRUE)? Default 8. Ignored if sys = FALSE.
check_sys_mapshaper

Value
geojson

check_sys_mapshaper  Check the system mapshaper

Description
Check the system mapshaper

Usage
check_sys_mapshaper(command = "mapshaper-xl", verbose = TRUE)

Arguments
command  either "mapshaper-xl" (default) or "mapshaper"
verbose  Print a message stating mapshaper's current version? Default TRUE

Value
TRUE (with a message) if appropriate version is installed, otherwise throws an error

drop_null_geometries  Drop features from a geo_list or geo_json FeatureCollection with null geometries

Description
Drop features from a geo_list or geo_json FeatureCollection with null geometries

Usage
drop_null_geometries(x)

Arguments
x  a geo_list or geo_json FeatureCollection

Value
a geo_list or geo_json FeatureCollection with Features with null geometries removed
ms_clip

Remove features or portions of features that fall outside a clipping area.

Description

Removes portions of the target layer that fall outside the clipping layer or bounding box.

Usage

ms_clip(
    target,
    clip = NULL,
    bbox = NULL,
    remove_slivers = FALSE,
    force_FC = TRUE,
    sys = FALSE,
    sys_mem = 8
)

Arguments

target the target layer from which to remove portions. One of:
    • geo_json or character points, lines, or polygons;
    • geo_list points, lines, or polygons;
    • SpatialPolygons, SpatialLines, SpatialPoints;
    • sf or sfc points, lines, or polygons object

clip the clipping layer (polygon). One of:
    • geo_json or character polygons;
    • geo_list polygons;
    • SpatialPolygons*;
    • sf or sfc polygons object

bbox supply a bounding box instead of a clipping layer to extract from the target layer. Supply as a numeric vector: c(minX,minY,maxX,maxY).

remove_slivers Remove tiny sliver polygons created by clipping. (Default FALSE)

force_FC should the output be forced to be a FeatureCollection (or Spatial*DataFrame) even if there are no attributes? Default TRUE. FeatureCollections are more compatible with rgdal::readOGR and geojsonio::geojson_sp. If FALSE and there are no attributes associated with the geometries, a GeometryCollection (or Spatial object with no dataframe) will be output.

sys Should the system mapshaper be used instead of the bundled mapshaper? Gives better performance on large files. Requires the mapshaper node package to be installed and on the PATH.

sys_mem How much memory (in GB) should be allocated if using the system mapshaper (sys = TRUE)? Default 8. Ignored if sys = FALSE.
Value
clipped target in the same class as the input target

Examples

```r
if (rmapshaper::check_v8_major_version() >= 6L) {
  library(geojsonio, quietly = TRUE)
  library(sp)

  poly <- structure("{
    "type": "FeatureCollection",
    "features": [{"type": "Feature",
      "properties": {},
      "geometry": {"type": "Polygon",
        "coordinates": 
        [[[52.8658, -44.7219], [53.7702, -40.4873], [55.3204, -37.5579],
          [56.2757, -37.917], [56.184, -40.6443], [61.0835, -40.7529],
          [58.0202, -43.634], [61.6699, -45.0578], [62.737, -46.3841],
          [55.7763, -46.2637], [54.9742, -49.1184], [52.799, -45.9386],
          [52.0329, -49.5677], [50.1747, -52.1014], [49.0698, -52.3641],
          [52.7068, -45.7639], [43.2278, -47.1908], [48.4755, -45.1388],
          [50.327, -43.5287], [48.0804, -41.2784], [49.6307, -40.6159],
          [52.8658, -44.7219]]]}
    }, class = c("json", "geo_json"))
  poly <- geojson_sp(poly)
  plot(poly)

  clip_poly <- structure(
    "{
      "type": "Feature",
      "properties": {},
      "geometry": {
        "type": "Polygon",
        "coordinates": [
          [51, -40],
          [55, -40],
          [55, -45],
          [51, -45],
          [51, -40]
        ]
      }
    }
  , class = c("json", "geo_json"))
  clip_poly <- geojson_sp(clip_poly)
  plot(clip_poly)

  out <- ms_clip(poly, clip_poly)
  plot(out, add = TRUE)
}
```

ms_dissolve

Aggregate shapes in a polygon or point layer.
Description

Aggregates using specified field, or all shapes if no field is given. For point layers, replaces a group of points with their centroid.

Usage

ms_dissolve(
input,  
field = NULL,  
sum_fields = NULL,  
copy_fields = NULL,  
weight = NULL,  
snap = TRUE,  
force_FC = TRUE,  
sys = FALSE,  
sys_mem = 8
)

Arguments

input spatial object to dissolve. One of:
  • geo_json or character points or polygons;
  • geo_list points or polygons;
  • SpatialPolygons, or SpatialPoints

field the field to dissolve on

sum_fields fields to sum

copy_fields fields to copy. The first instance of each field will be copied to the aggregated feature.

weight Name of an attribute field for generating weighted centroids (points only).

snap Snap together vertices within a small distance threshold to fix small coordinate misalignment in adjacent polygons. Default TRUE.

force_FC should the output be forced to be a FeatureCollection (or Spatial*DataFrame) even if there are no attributes? Default TRUE. FeatureCollections are more compatible with rgdal::readOGR and geojsonio::geojson_sp. If FALSE and there are no attributes associated with the geometries, a GeometryCollection (or Spatial object with no dataframe) will be output.

sys Should the system mapshaper be used instead of the bundled mapshaper? Gives better performance on large files. Requires the mapshaper node package to be installed and on the PATH.

sys_mem How much memory (in GB) should be allocated if using the system mapshaper (sys = TRUE)? Default 8. Ignored if sys = FALSE.

Value

the same class as the input
Examples

library(geojsonio)
library(sp)

poly <- structure('
  {"type":"FeatureCollection",
   "features":[
     {"type":"Feature",
      "properties":{"a": 1, "b": 2},
      "geometry":{"type":"Polygon","coordinates":[
        [102,2],[102,3],[103,3],[103,2],[102,2]
      ]}
    },
    {"type":"Feature",
     "properties":{"a": 5, "b": 3},
     "geometry":{"type":"Polygon","coordinates":[
       [100,0],[100,1],[101,1],[101,0],[100,0]
     ]}
    ]}',
  class = c("json", "geo_json"))

poly <- geojson_sp(poly)
plot(poly)
length(poly)
poly@data

# Dissolve the polygon
out <- ms_dissolve(poly)
plot(out)
length(out)
out@data

# Dissolve and summing columns
out <- ms_dissolve(poly, sum_fields = c("a", "b"))
plot(out)
out@data

ms_erase

Remove features or portions of features that fall inside a specified area

Description

Removes portions of the target layer that fall inside the erasing layer or bounding box.

Usage

ms_erase(
  target,
  erase = NULL,
  bbox = NULL,
  remove_slivers = FALSE,
  force_FC = TRUE,
  sys = FALSE,
Arguments

target the target layer from which to remove portions. One of:
  • geo_json or character points, lines, or polygons;
  • geo_list points, lines, or polygons;
  • SpatialPolygons, SpatialLines, SpatialPoints

erase the erase layer (polygon). One of:
  • geo_json or character polygons;
  • geo_list polygons;
  • SpatialPolygons*

bbox supply a bounding box instead of an erasing layer to remove from the target layer. Supply as a numeric vector: c(minX, minY, maxX, maxY).

remove_slivers Remove tiny sliver polygons created by erasing. (Default FALSE)

force_FC should the output be forced to be a FeatureCollection (or Spatial*DataFrame) even if there are no attributes? Default TRUE. FeatureCollections are more compatible with rgdal::readOGR and geojsonio::geojson_sp. If FALSE and there are no attributes associated with the geometries, a GeometryCollection (or Spatial object with no dataframe) will be output.

sys Should the system mapshaper be used instead of the bundled mapshaper? Gives better performance on large files. Requires the mapshaper node package to be installed and on the PATH.

sys_mem How much memory (in GB) should be allocated if using the system mapshaper (sys = TRUE)? Default 8. Ignored if sys = FALSE.

Value

erased target in the same format as the input target

Examples

if (rmapshaper:::check_v8_major_version() >= 6L) {
  library(geojsonio, quietly = TRUE)
  library(sp)

  points <- structure("{
    "type": "FeatureCollection",
    "features": [
      {"type": "Feature",
       "properties": {}
      },
      {"type": "Feature",
       "properties": {}
      },
      {"type": "Feature",
       "properties": {}
      },
      {"type": "Feature",
       "properties": {}
      },
      {"type": "Feature",
       "properties": {}
      }
    ],
    "bbox": c(52.8658, -44.7219, 55.3204, -37.5579),
    "force_FC": TRUE
  },
  CLASS = "FeatureCollection")
  print(points)
}

ms_explode

Convert multipart lines or polygons to singlepart

Description
For objects of class Spatial (e.g., SpatialPolygonsDataFrame), you may find it faster to use sp::disaggregate.

Usage
ms_explode(input, force_FC = TRUE, sys = FALSE, sys_mem = 8)

Arguments
input One of:
- geo_json or character multipart lines, or polygons;
- geo_list multipart lines, or polygons;
ms_explode

- multipart SpatialPolygons, SpatialLines;
- sf or sfc multipart lines, or polygons object

force_FC

 should the output be forced to be a FeatureCollection (or Spatial*DataFrame) even if there are no attributes? Default TRUE. FeatureCollections are more compatible with rgdal::readOGR and geojsonio::geojson_sp. If FALSE and there are no attributes associated with the geometries, a GeometryCollection (or Spatial object with no dataframe) will be output.

sys

 Should the system mapshaper be used instead of the bundled mapshaper? Gives better performance on large files. Requires the mapshaper node package to be installed and on the PATH.

sys_mem

 How much memory (in GB) should be allocated if using the system mapshaper (sys = TRUE)? Default 8. Ignored if sys = FALSE.

Details

There is currently no method for SpatialMultiPoints

Value

 same class as input

Examples

```r
library(geojsonio)
library(sp)

poly <- structure("{"type":"FeatureCollection","crs":{
"type":"name","properties":{
"name": _"urn:ogc:def:crs:OGC:1.3:CRS84"}}},"features":
[{
"type":"Feature","geometry":{"type":"MultiPolygon","coordinates":
[([[102,2],[102,3],[103,3],[103,2],[102,2]],[[100,0],[100,1],[101,1],[101,0],[100,0]]))],"properties":{
"rmapshaperid":0}}],

class = c("json", "geo_json")
"

poly <- geojson_sp(poly)
plot(poly)
length(poly)
poly@data

# Explode the polygon
out <- ms_explode(poly)
plot(out)
length(out)
out@data
```
ms_filter_fields  
Delete fields in the attribute table

Description

Removes all fields except those listed in the fields parameter

Usage

ms_filter_fields(input, fields, sys = FALSE, sys_mem = 8)

Arguments

input  
spatial object to filter fields on. One of:
• geo_json or character points, lines, or polygons;
• geo_list points, lines, or polygons;
• SpatialPolygonsDataFrame, SpatialLinesDataFrame, SpatialPointsDataFrame;
• sf object

fields  
character vector of fields to retain.

sys  
Should the system mapshaper be used instead of the bundled mapshaper? Gives better performance on large files. Requires the mapshaper node package to be installed and on the PATH.

sys_mem  
How much memory (in GB) should be allocated if using the system mapshaper (sys = TRUE)? Default 8. Ignored if sys = FALSE.

Value

object with only specified attributes retained, in the same class as the input

Examples

library(geojsonio)
library(sp)

poly <- structure("{"type":"FeatureCollection","features": [{"type":"Feature","properties":{"a": 1, "b":2, "c": 3}, "geometry":{"type":"Polygon","coordinates": [[102,2],[102,4],[104,4],[104,2],[102,2]]}}], class = c("json", "geo_json"))
poly <- geojson_sp(poly)
poly@data

# Filter (keep) fields a and b, drop c
out <- ms_filter_fields(poly, c("a", "b"))
out@data
ms_filter_islands  
Remove small detached polygons (islands)

Description
Remove small detached polygons, keeping those with a minimum area and/or a minimum number of vertices. Optionally remove null geometries.

Usage
```r
ms_filter_islands(
  input,
  min_area = NULL,
  min_vertices = NULL,
  drop_null_geometries = TRUE,
  force_FC = TRUE,
  sys = FALSE,
  sys_mem = 8
)
```

Arguments
- **input**: spatial object to filter. One of:
  - `geo_json` or `character` polygons;
  - `geo_list` polygons;
  - `SpatialPolygons*`;
  - `sf` or `sfc` polygons object
- **min_area**: minimum area of polygons to retain. Area is calculated using planar geometry, except for the area of unprojected polygons, which is calculated using spherical geometry in units of square meters.
- **min_vertices**: minimum number of vertices in polygons to retain.
- **drop_null_geometries**: should features with empty geometries be dropped? Default `TRUE`. Ignored for `SpatialPolygons*`, as it is always `TRUE`.
- **force_FC**: should the output be forced to be a FeatureCollection (or Spatial*DataFrame) even if there are no attributes? Default `TRUE`. FeatureCollections are more compatible with `rgdal::readOGR` and `geojsonio::geojson_sp`. If `FALSE` and there are no attributes associated with the geometries, a GeometryCollection (or Spatial object with no dataframe) will be output.
- **sys**: Should the system mapshaper be used instead of the bundled mapshaper? Gives better performance on large files. Requires the mapshaper node package to be installed and on the PATH.
- **sys_mem**: How much memory (in GB) should be allocated if using the system mapshaper (`sys = TRUE`)? Default 8. Ignored if `sys = FALSE`. 
Value

object with only specified features retained, in the same class as the input

Examples

```r
library(geojsonio)
library(sp)

poly <- structure("{"type":"FeatureCollection",
"features": [{"type":"Feature","properties":{},
"geometry": {"type":"Polygon",
"coordinates": [[[102,2],[102,4],[104,4],[104,2],[102,2]]],
{"type":""Feature","properties":{},
"geometry": {"type":""Polygon",
"coordinates": [[[100,2],[98,4],[101.5,4],[100,2]]],
{"type":""Feature","properties":{},
"geometry": {"type":""Polygon",
"coordinates": [[[100,0],[100,1],[101,1],[101,0],[100,0]]]}},
class = c("json", "geo_json")"
},

poly <- geojson_sp(poly)
plot(poly)

out <- ms_filter_islands(poly, min_area = 12391399903)
plot(out)
```

`ms_innerlines` Create a line layer consisting of shared boundaries with no attribute data

Description

Create a line layer consisting of shared boundaries with no attribute data

Usage

`ms_innerlines(input, force_FC = TRUE, sys = FALSE, sys_mem = 8)`

Arguments

- `input` input polygons object to convert to inner lines. One of:
  - `geo_json` or character polygons;
  - `geo_list` polygons;
  - `SpatialPolygons*`;
  - `sf` or `sfc` polygons object
force_FC should the output be forced to be a FeatureCollection (or Spatial*DataFrame) even if there are no attributes? Default TRUE. FeatureCollections are more compatible with rgdal::readOGR and geojsonio::geojson_sp. If FALSE and there are no attributes associated with the geometries, a GeometryCollection (or Spatial object with no dataframe) will be output.

sys Should the system mapshaper be used instead of the bundled mapshaper? Gives better performance on large files. Requires the mapshaper node package to be installed and on the PATH.

sys_mem How much memory (in GB) should be allocated if using the system mapshaper (sys = TRUE)? Default 8. Ignored if sys = FALSE.

Value

lines in the same class as the input layer, but without attributes

Examples

```r
library(geojsonio)
library(sp)

poly <- structure(list("type":"FeatureCollection",
  "features":[
    "type":"Feature",
    "properties":{"foo": "a"},
    "geometry":{"type":"Polygon","coordinates":[[
  [102,2],[102,3],[103,3],[103,2],[102,2]]]}
  ],"type":"Feature",
  "properties":{"foo": "a"},
  "geometry":{"type":"Polygon","coordinates":[[
  [103,3],[104,3],[104,2],[103,2],[103,3]]]}
  ),"type":"Feature",
  "properties":{"foo": "b"},
  "geometry":{"type":"Polygon","coordinates":[[
  [102,1],[102,2],[103,2],[103,1],[102,1]]]}
  ),"type":"Feature",
  "properties":{"foo": "b"},
  "geometry":{"type":"Polygon","coordinates":[[
  [103,1],[103,2],[104,2],[104,1],[103,1]]]}
  ), class = c("json", "geo_json")

poly <- geojson_sp(poly)
plot(poly)

out <- ms_innerlines(poly)
plot(out)
```
ms_lines

Convert polygons to topological boundaries (lines)

Description

Convert polygons to topological boundaries (lines)

Usage

ms_lines(input, fields = NULL, force_FC = TRUE, sys = FALSE, sys_mem = 8)

Arguments

input input polygons object to convert to inner lines. One of:
  • geo_json or character polygons;
  • geo_list polygons;
  • SpatialPolygons*;
  • sf or sfc polygons object

fields character vector of field names. If left as NULL (default), external (unshared)
  boundaries are attributed as TYPE 0 and internal (shared) boundaries are TYPE 1. Giving a field name adds an intermediate level of hierarchy at TYPE 1, with
  the lowest-level internal boundaries set to TYPE 2. Supplying a character vector
  of field names adds additional levels of hierarchy.

force_FC should the output be forced to be a FeatureCollection (or Spatial*DataFrame)
  even if there are no attributes? Default TRUE. FeatureCollections are more com-
  patible with rgdal::readOGR and geojsonio::geojson_sp. If FALSE and there are
  no attributes associated with the geometries, a GeometryCollection (or Spatial
  object with no dataframe) will be output.

sys Should the system mapshaper be used instead of the bundled mapshaper? Gives
  better performance on large files. Requires the mapshaper node package to be
  installed and on the PATH.

sys_mem How much memory (in GB) should be allocated if using the system mapshaper
  (sys = TRUE)? Default 8. Ignored if sys = FALSE.

Value

topological boundaries as lines, in the same class as the input

Examples

library(geojsonio)
library(sp)

poly <- structure("type":"FeatureCollection",
  "features":[
ms_points

Create points from a polygon layer

Description

Can be generated from the polygons by specifying location to be "centroid" or "inner", OR by specifying fields in the attributes of the layer containing x and y coordinates.

Usage

```r
ms_points(
  input,
  location = NULL,
  x = NULL,
  y = NULL,
  force_FC = TRUE,
  sys = FALSE,
  sys_mem = 8
)
```

Arguments

- `input` input polygons object to convert to points. One of:
  - `geo_json` or character polygons;
  - `matrix`
**ms_points**

- geo_list polygons;
- SpatialPolygons;
- sf or sfc polygons object

**location**
either "centroid" or "inner". If "centroid", creates points at the centroid of the largest ring of each polygon feature. If "inner", creates points in the interior of the largest ring of each polygon feature. Inner points are located away from polygon boundaries. Must be NULL if x and y are specified. If left as NULL (default), will use centroids.

**x**
name of field containing x coordinate values. Must be NULL if location is specified.

**y**
name of field containing y coordinate values. Must be NULL if location is specified.

**force_FC**
should the output be forced to be a FeatureCollection (or Spatial*DataFrame) even if there are no attributes? Default TRUE. FeatureCollections are more compatible with rgdal::readOGR and geojsonio::geojson_sp. If FALSE and there are no attributes associated with the geometries, a GeometryCollection (or Spatial object with no dataframe) will be output.

**sys**
Should the system mapshaper be used instead of the bundled mapshaper? Gives better performance on large files. Requires the mapshaper node package to be installed and on the PATH.

**sys_mem**
How much memory (in GB) should be allocated if using the system mapshaper (sys = TRUE)? Default 8. Ignored if sys = FALSE.

**Value**
points in the same class as the input.

**Examples**

```r
library(geojsonio)
library(sp)

poly <- structure("{
  "type": "FeatureCollection",
  "features": [
    {"type": "Feature",
      "properties": {
        "x_pos": 1,
        "y_pos": 2
      },
      "geometry": {
        "type": "Polygon",
        "coordinates": [[[102,2],[102,4],[104,4],[104,2],[102,2]]]
      }
    },
    {"type": "Feature",
      "properties": {
        "x_pos": 3,
        "y_pos": 4
      },
      "geometry": {
        "type": "Polygon",
        "coordinates": [[[100,2],[98,4],[101.5,4],[100,2]]]
      }
    },
    {"type": "Feature",
      "properties": {
        "x_pos": 5,
        "y_pos": 6
      },
      "geometry": {
        "type": "Polygon",
        "coordinates": [[[100,0],[100,1],[101,1],[101,0],[100,0]]]
      }
    }
  ],
  class = c("json", "geo_json")
}

poly <- geojson_sp(poly)
summary(poly)
plot(poly)
```
# Convert to points using centroids
out <- ms_points(poly, location = "centroid")
summary(out)
plot(out)

# Can also specify locations using attributes in the data
out <- ms_points(poly, x = "x_pos", y = "y_pos")
summary(out)
plot(out)

---

**ms_simplify**  
*Topologically-aware geometry simplification.*

**Description**

Uses mapshaper to simplify polygons.

**Usage**

```r
ms_simplify(
  input,
  keep = 0.05,
  method = NULL,
  weighting = 0.7,
  keep_shapes = FALSE,
  no_repair = FALSE,
  snap = TRUE,
  explode = FALSE,
  force_FC = TRUE,
  drop_null_geometries = TRUE,
  snap_interval = NULL,
  sys = FALSE,
  sys_mem = 8
)
```

**Arguments**

- **input**: spatial object to simplify. One of:
  - geo_json or character polygons or lines;
  - geo_list polygons or lines;
  - SpatialPolygons* or SpatialLines*;
  - sf or sfc polygons or lines object
- **keep**: proportion of points to retain (0-1; default 0.05)
**ms_simplify**

A simplified version of the input geometry.

**method**

Simplification method to use: "vis" for Visvalingam algorithm, or "dp" for Douglas-Peuker algorithm. If left as NULL (default), uses Visvalingam simplification but modifies the area metric by underweighting the effective area of points at the vertex of more acute angles, resulting in a smoother appearance. See this [link](https://github.com/mbloch/mapshaper/wiki/Simplification-Tips) for more information.

**weighting**

Coefficient for weighting Visvalingam simplification (default is 0.7). Higher values produce smoother output. weighting=0 is equivalent to unweighted Visvalingam simplification.

**keep_shapes**

Prevent small polygon features from disappearing at high simplification (default FALSE).

**no_repair**

disable intersection repair after simplification (default FALSE).

**snap**

Snap together vertices within a small distance threshold to fix small coordinate misalignment in adjacent polygons. Default TRUE.

**explode**

Should multipart polygons be converted to singlepart polygons? This prevents small shapes from disappearing during simplification if keep_shapes = TRUE. Default FALSE.

**force_FC**

Should the output be forced to be a FeatureCollection (or Spatial*DataFrame) even if there are no attributes? Default TRUE. FeatureCollections are more compatible with rgdal::readOGR and geojsonio::geojson_sp. If FALSE and there are no attributes associated with the geometries, a GeometryCollection (or Spatial object with no dataframe) will be output.

**drop_null_geometries**

Should Features with null geometries be dropped? Ignored for Spatial* objects, as it is always TRUE.

**snap_interval**

Specify snapping distance in source units, must be a numeric. Default NULL.

**sys**

Should the system mapshaper be used instead of the bundled mapshaper? Gives better performance on large files. Requires the mapshaper node package to be installed and on the PATH.

**sys_mem**

How much memory (in GB) should be allocated if using the system mapshaper (sys = TRUE)? Default 8. Ignored if sys = FALSE.

**Value**

A simplified representation of the geometry in the same class as the input.

**Examples**

```r
# With a simple geojson object
poly <- structure(
  list(
    type = "Feature",
    properties = list(),
    geometry = list(
      type = "Polygon",
      coordinates = list(
        c(-70.603637, -33.399918),
        c(-70.614624, -33.395332),
        c(-70.615337, -33.395218),
        c(-70.603637, -33.399918)
      )
    )
  ),
  class = "FeatureList"
)
```
[[-70.639343, -33.392466],
[-70.659942, -33.391962],
[-70.639792, -33.404504],
[-70.697021, -33.419406],
[-70.701141, -33.434306],
[-70.700454, -33.446339],
[-70.694274, -33.458369],
[-70.682601, -33.465816],
[-70.668869, -33.472117],
[-70.646209, -33.473835],
[-70.624923, -33.472117],
[-70.609817, -33.468107],
[-70.595397, -33.458369],
[-70.587158, -33.442901],
[-70.587158, -33.426283],
[-70.590591, -33.414248],
[-70.594711, -33.406224],
[-70.603637, -33.399918]]

`, class = c("json", "geo_json"))

ms_simplify(poly, keep = 0.1)

# With a SpatialPolygonsDataFrame:

poly_sp <- geojsonio::geojson_sp(poly)
ms_simplify(poly_sp, keep = 0.5)

---

**rmapshaper**

**rmapshaper: Client for 'mapshaper' for 'Geospatial' Operations**

**Description**

Edit and simplify 'geojson', 'Spatial', and 'sf' objects. This is wrapper around the 'mapshaper' ‘javascript’ library by Matthew Bloch [https://github.com/mbloch/mapshaper/](https://github.com/mbloch/mapshaper/) to perform topologically-aware polygon simplification, as well as other operations such as clipping, erasing, dissolving, and converting 'multi-part' to 'single-part' geometries. It relies on the 'geojsonio' package for working with 'geojson' objects, the 'sf' package for working with 'sf' objects, and the 'sp' and 'rgdal' packages for working with 'Spatial' objects.

**rmapshaper functions**

All functions

- **ms_simplify** - simplify polygons or lines
- **ms_clip** - clip an area out of a layer using a polygon layer or a bounding box. Works on polygons, lines, and points
- **ms_erase** - erase an area from a layer using a polygon layer or a bounding box. Works on polygons, lines, and points
- **ms_dissolve** - aggregate polygon features, optionally specifying a field to aggregate on. If no field is specified, will merge all polygons into one.
- **ms_explode** - convert multipart shapes to single part. Works with polygons, lines, and points in geojson format, but currently only with polygons and lines in the Spatial classes (not SpatialMultiPoints and SpatialMultiPointsDataFrame).
- **ms_lines** - convert polygons to topological boundaries (lines)
- **ms_innerlines** - convert polygons to shared inner boundaries (lines)
- **ms_points** - create points from a polygon layer
- **ms_filter_fields** - Remove fields from the attributes
- **ms_filter_islands** - Remove small detached polygons

**Author(s)**

Andy Teucher <andy.teucher@gmail.com>
Index

apply_mapshaper_commands, 2
check_sys_mapshaper, 3
drop_null_geometries, 3
ms_clip, 4, 20
ms_dissolve, 5, 21
ms_erase, 7, 21
ms_explode, 9, 21
ms_filter_fields, 11, 21
ms_filter_islands, 12, 21
ms_innerlines, 13, 21
ms_lines, 15, 21
ms_points, 16, 21
ms_simplify, 18, 20
rmapshaper, 20