
rdp Documentation

Release 0.7

Fabian Hirschmann

December 19, 2016

1	Installation	3
2	Usage	5

The Ramer–Douglas–Peucker algorithm (RDP) is an algorithm for reducing the number of points in a curve that is approximated by a series of points.

An interactive version of this algorithm can be found in [this blog post](#).

This implementation works on 2D and 3D data.

Installation

The rdp package is available via pip:

```
pip install rdp
```

The code of this package is hosted at [GitHub](#).

Usage

`rdp.rdp` (*M*, *epsilon*=0, *dist*=<function *pldist*>, *algo*='iter', *return_mask*=False)
Simplifies a given array of points using the Ramer-Douglas-Peucker algorithm.

Example:

```
>>> from rdp import rdp
>>> rdp([[1, 1], [2, 2], [3, 3], [4, 4]])
[[1, 1], [4, 4]]
```

This is a convenience wrapper around both `rdp.rdp_iter()` and `rdp.rdp_rec()` that detects if the input is a numpy array in order to adapt the output accordingly. This means that when it is called using a Python list as argument, a Python list is returned, and in case of an invocation using a numpy array, a NumPy array is returned.

The parameter `return_mask=True` can be used in conjunction with `algo="iter"` to return only the mask of points to keep. Example:

```
>>> from rdp import rdp
>>> import numpy as np
>>> arr = np.array([1, 1, 2, 2, 3, 3, 4, 4]).reshape(4, 2)
>>> arr
array([[1, 1],
       [2, 2],
       [3, 3],
       [4, 4]])
>>> mask = rdp(arr, algo="iter", return_mask=True)
>>> mask
array([ True, False, False,  True], dtype=bool)
>>> arr[mask]
array([[1, 1],
       [4, 4]])
```

Parameters

- **M** (numpy array with shape (n, d) where n is the number of points and d their dimension) – a series of points
- **epsilon** (*float*) – epsilon in the rdp algorithm
- **dist** (function with signature $f(\text{point}, \text{start}, \text{end})$ – see `rdp.pldist()`) – distance function
- **algo** (*string*) – either `iter` for an iterative algorithm or `rec` for a recursive algorithm
- **return_mask** (*bool*) – return mask instead of simplified array

`rdp.rdp_rec` (*M*, *epsilon*, *dist*=<function *pldist*>)
Simplifies a given array of points.

Recursive version.

Parameters

- **M** (*numpy array*) – an array
- **epsilon** (*float*) – epsilon in the rdp algorithm
- **dist** (function with signature `f(point, start, end)` – see `rdp.pldist()`) – distance function

`rdp.rdp_iter` (*M*, *epsilon*, *dist*=<function *pldist*>, *return_mask*=*False*)
Simplifies a given array of points.

Iterative version.

Parameters

- **M** (*numpy array*) – an array
- **epsilon** (*float*) – epsilon in the rdp algorithm
- **dist** (function with signature `f(point, start, end)` – see `rdp.pldist()`) – distance function
- **return_mask** (*bool*) – return the mask of points to keep instead

`rdp.pldist` (*point*, *start*, *end*)
Calculates the distance from *point* to the line given by the points *start* and *end*.

Parameters

- **point** (*numpy array*) – a point
- **start** (*numpy array*) – a point of the line
- **end** (*numpy array*) – another point of the line

P

`pldist()` (in module `rdp`), 6

R

`rdp()` (in module `rdp`), 5

`rdp_iter()` (in module `rdp`), 6

`rdp_rec()` (in module `rdp`), 5