Package 'MeshDataSimplification'

May 3, 2017

Version 0.1

Date 2017-04-01

Title Simplification Strategy for Surface Grids Augmented with Distributed Data

Author Franco Dassi [aut],
Bree Ettinger [aut],
Marco Martinolli [aut],
Simona Perotto [aut],
Laura M. Sangalli [aut],
Stefano Ubbiali [aut, cre]

Maintainer Stefano Ubbiali <stefano.ubbiali@mail.polimi.it>

Depends R (>= 3.0.2), Rcpp (>= 0.9.11), plot3D

LinkingTo Rcpp (>= 0.9.11), RcppEigen

Description
Iterative simplification strategy for surface triangular grids augmented with distributed data.
Each iteration corresponds to an edge collapse where the selection of the edge to contract derives from
a cost functional based on both geometric and statistical-based considerations.
The library can handle both zero and higher genus surfaces.
For a detailed description of the algorithm, please refer to:
Dassi, F., Ettinger, B., Perotto, S., Sangalli, L.M. (2015),
A mesh simplification strategy for a spatial regression analysis over the cortical surface of the brain,
Applied Numerical Mathematics, Vol. 90, pp. 111-131.

NeedsCompilation yes

SystemRequirements C++11

RoxygenNote 6.0.1

R topics documented:

`get.data.locations` *Get the list of data points locations*

Description

Get the list of data points locations

Usage

```
get.data.locations(x)
```

Arguments

x	An object of class simplification, created through `setup.simplification` or `setup.simplification.from.file`.

Value

A #data-by-3 matrix storing the coordinates of data points locations.

`get.edges` *Get the list of edges*

Description

Get the list of edges

Usage

```
get.edges(x)
```

Arguments

x	An object of class simplification, created through setup.simplification or setup.simplification.from.file.

Value

A #edges-by-2 matrix, where the i-th row stores the Id's of the end-points of the i-th edge.

---

get.nodes *Get the list of nodes*

Description

Get the list of nodes

Usage

```
get.nodes(x)
```

Arguments

x	An object of class simplification, created through setup.simplification or setup.simplification.from.file.

Value

A #nodes-by-3 matrix, where the i-th row stores the coordinates of the i-th node.

---

get.observations *Get the list of observations*

Description

Get the list of observations

Usage

```
get.observations(x)
```

Arguments

x	An object of class simplification, created through setup.simplification or setup.simplification.from.file.

Value

A #data-by-1 vector storing the observations.

get.quantity.of.information *Get the quantity of information for each element*

Description

Get the quantity of information for each element

Usage

```
get.quantity.of.information(x)
```

Arguments

x	An object of class simplification, created through setup.simplification or setup.simplification.from.file.

Value

A #elements-by-1 vector, storing the quantity of information for each element.

get.surface.mesh *Get surface mesh*

Description

Get surface mesh

Usage

```
get.surface.mesh(x)
```

Arguments

x	An object of class simplification, created through setup.simplification or setup.simplification.from.file.

Value

A SURFACE_MESH object, whose order is compliant with the mesh used to initialize the simplification framework.

get.triangles *Get the list of elements*

Description

Get the list of elements

Usage

```
get.triangles(x)
```

Arguments

x	An object of class simplification, created through setup.simplification or setup.simplification.from.file.

Value

A #elements-by-3 matrix, where the i-th row stores the Id's of the vertices of the i-th triangle.

plot.surface.mesh *Plot a mesh in a 3D perspective*

Description

Plot the three-dimensional surface mesh held by an object of class simplification. The package plot3D is used.

Usage

```
plot.surface.mesh(x, phi = 40, theta = 40, ...)
```

Arguments

x	An object of class simplification, created through setup.simplification or setup.simplification.from.file.
phi	Colatitude (in degrees) characterizing the viewing direction; default is 40.
theta	Longitude (in degrees) characterizing the viewing direction; default is 40.
...	Additional arguments passed to the plotting methods; these include: xlim, ylim, zlim, xlab, ylab, zlab, main, sub, r, d, scale, expand, box, axes, nticks, ticktype.

Examples

```
## Instantiate an object of class simplification for the simplification of a pawn geometry;
## suppose that the components of the edge cost function are equally weighted
data(pawn)
obj <- setup.simplification(mesh)
## Plot the original mesh
plot.surface.mesh(obj, main = "Original mesh, 2522 nodes")
## Run the simplification strategy, reducing the mesh down to n = 1500 nodes
run.simplification(obj, 1500)
## Plot the simplified mesh
plot.surface.mesh(obj, main = "Simplified mesh, 1500 nodes")
## Resume the simplification procedure, reducing the mesh down to n = 1000 nodes
run.simplification(obj, 1000)
## Plot the simplified mesh
plot.surface.mesh(obj, main = "Simplified mesh, 1000 nodes")
```

---

run.simplification *Run the simplification process*

Description

Function running the simplification process by invoking the method `simplify` of the `RcppSimplification` class through the attribute `simplifier` of the `simplification` object given as argument. Note that the procedure is completely carried out at the C++ level by the `meshsimplification` library.

Usage

```
run.simplification(x, numNodesMax, file = '')
```

Arguments

`x`	An object of class `simplification`, created through `setup.simplification` or `setup.simplification.from.file`.
`numNodesMax`	Final number of nodes.
`file`	String specifying the path to the location where the decimated mesh will be stored; .inp file format supported. If the path is not provided, the mesh will no printed.

Value

A list with the following fields:

`mesh`	a SURFACE_MESH object, storing the simplified grid;
`locations`	a #data-by-3 matrix with the locations of data points onto the simplified grid;
`qoi`	a #elements-by-1 vector with the quantity of information (QOI) for each triangle.

Examples

```
## Instantiate an object of class simplification for the simplification of a pawn geometry;
## suppose that the components of the edge cost function are equally weighted
data(pawn)
obj <- setup.simplification(mesh)
## Plot the original mesh
plot.surface.mesh(obj, main = "Original mesh, 2522 nodes")
## Run the simplification strategy, reducing the mesh down to n = 1500 nodes
run.simplification(obj, 1500)
## Plot the simplified mesh
plot.surface.mesh(obj, main = "Simplified mesh, 1500 nodes")
## Resume the simplification procedure, reducing the mesh down to n = 1000 nodes
run.simplification(obj, 1000)
## Plot the simplified mesh
plot.surface.mesh(obj, main = "Simplified mesh, 1000 nodes")
```

---

setup.simplification *Initialize the simplification framework from an object of class SURFACE_MESH*

Description

This function instantiates an object of class `simplification`, holding all informations regarding a surface two dimensional grid, and useful to run the simplification process. This function heavily relies on `RcppSimplification` - a wrapper for the template class `simplification<Triangle, MeshType::DATA, DataGeo>` provided by the C++ library `meshsimplification`.

Usage

```
setup.simplification(mesh, loc = NULL, val = NULL, wgeom = 1/3, wdisp = 1/3, wequi = 1/3)
```

Arguments

`mesh`	A SURFACE_MESH object.
`loc`	#data-by-3 vector with data locations; default is NULL, i.e. locations are supposed to coincide with grid nodes.
`val`	#data-by-1 vector with the observations; default is NULL, i.e. observations set to zero.
`wgeom`	Weight for the geometric component of the edge cost function; default is 1/3. Note that the all weights should be positive and sum up to one.
`wdisp`	Weight for the data displacement component of the edge cost function; default is 1/3. Note that the all weights should be positive and sum up to one.
`wequi`	Weight for the data equidistribution component of the edge cost function; default is 1/3. Note that the all weights should be positive and sum up to one.

Value

An object of class simplification, provided with the following fields:

`simplifier`	An object of class `RcppSimplification`.
`order`	Either '1' or '2', saying whether each triangle should be represented through three points (the vertices) or six points (the vertices plus the midpoints of the edges). These representations respectively allow to build a linear or quadratic Finite Element basis over the mesh. The Finite Element order is determined from the input SURFACE_MESH object.

See Also

`plot.surface.mesh`, `run.simplification`

Examples

```
## Instantiate an object of class simplification for the simplification of a pawn geometry;
## suppose that the components of the edge cost function are equally weighted
data(pawn)
obj <- setup.simplification(mesh)
## Plot the original mesh
plot.surface.mesh(obj, main = "Original mesh, 2522 nodes")
## Run the simplification strategy, reducing the mesh down to n = 1500 nodes
run.simplification(obj, 1500)
## Plot the simplified mesh
plot.surface.mesh(obj, main = "Simplified mesh, 1500 nodes")
## Resume the simplification procedure, reducing the mesh down to n = 1000 nodes
run.simplification(obj, 1000)
## Plot the simplified mesh
plot.surface.mesh(obj, main = "Simplified mesh, 1000 nodes")
```

`setup.simplification.from.file`

Initialize the simplification framework from file

Description

This function instantiates an object of class `simplification`, holding all informations regarding a surface two dimensional grid, and useful to run the simplification process. This function heavily relies on `RcppSimplification` - a wrapper for the template class `simplification<Triangle, MeshType::DATA, DataGeo>` provided by the C++ library `meshsimplification`.

Usage

```
setup.simplification.from.file(file, index = 1, wgeom = 1/3, wdisp = 1/3, wequi = 1/3)
```

Arguments

file	Absolute or relative path to the input mesh; .inp and .vtk file formats are supported.
index	Either '0' or '1', saying whether vertices Id's are enumerated according to a 0-based or 1-based indexing, respectively. Default is '1'.
wgeom	Weight for the geometric component of the edge cost function; default is 1/3. Note that the all weights should be positive and sum up to one.
wdisp	Weight for the data displacement component of the edge cost function; default is 1/3. Note that the all weights should be positive and sum up to one.
wequi	Weight for the data equidistribution component of the edge cost function; default is 1/3. Note that the all weights should be positive and sum up to one.

Value

An object of class simplification, provided with the following fields:

simplifier	An object of class RcppSimplification.
order	Either '1' or '2', saying whether each triangle should be represented through three points (the vertices) or six points (the vertices plus the midpoints of the edges). These representations respectively allow to build a linear or quadratic Finite Element basis over the mesh. In case of the simplification framework initialized from file, the order is supposed to be '1'.

See Also

plot.surface.mesh, run.simplification

Index