PyESG Documentation

Release 0.0.13

Feng Zhu

January 05, 2015

CONTENTS

Contents:

CHAPTER ONE

INTRODUCTION

The Python library of Earth Spherical Geometry (PyESG) is aiming to perform spherical geometry on the earth and interpolate data from an unstructured mesh grid to another one, which is the so-called “regridding” or “remapping”.

CHAPTER

TWO

SPHERICAL GEOMETRY

2.1 Point

2.2 Arc

2.3 Triangle

2.4 Quadrangle

2.5 Mesh

CHAPTER

THREE

REGRIDDING

This section will introduce the algorithm for regridding in the PyESG.

3.1 Search

3.2 Interpolation

CHAPTER

FOUR

CLASSES

class `pyesg.`**`Arc`**`(`*p1*, *p2*`)`

An arc on the earth defined by two points p1 and p2. It also can be seen as the relationship between two points.

`distance``()`

Calculate the great-circle distance of the arc.

[reference: http://en.wikipedia.org/wiki/Great-circle_distance]

`rad``()`

Convert great-circle distance on the earth to radians.

`waypoint``(`*k*`)`

Calculate the location of a selected point (lat, lon) according to: + the location of point 1 (lat1, lon1); + the location of point 2 (lat2, lon2); + the coefficient k decides the position between point 1 and point 2,

e.g.: + when k = 0.0, (lat, lon) is point 1; + when k = 0.5, (lat, lon) is the mid-point; + when k = 1.0, (lat, lon) is point 2. [reference: http://en.wikipedia.org/wiki/Great-circle_navigation]

class `pyesg.`**`Interp`**

Interpolation algorithms.

`barycentric``(`*point*, *triangle*`)`

(point, triangle) -> weight1, weight2, weight3

Barycentric Interpolation: interpolation in a triangle.

[reference: https://classes.soe.ucsc.edu/cmps160/Fall10/resources/barycentricInterpolation.pdf]

`regrid``(`*mesh_old*, *mesh_new*, *method='standard'*`)`

(mesh_old, mesh_new) -> matrix of weights and points indices.

Calculate the remapping coefficients (weights) from an old mesh to a new mesh. + When method == standard, the search algorithm can resolve any situation but slow; + When method == quick, the situation is that mesh_old and mesh_new are very similar to each other with some points nudged.

class `pyesg.`**`Mesh`**`(`*lat2d*, *lon2d*`)`

Unstructed mesh grids, which is defined by 2 dimenional arrays lat2d and lon2d.

class `pyesg.`**`Point`**`(`*lat*, *lon*`)`

A point on the earth defined by the latitude and longitude (unit: degree).

`lat_deg``()`

radians -> degree

`lon_deg``()`

radians -> degree

`spherical_coord``()`

p (lat, lon) -> x, y, z

Return the (x, y, z) in a UNIT spherical coordinate system.

[reference: http://en.wikipedia.org/wiki/Spherical_coordinate_system]

vector()

Return the vector from the center of the Earth to the point.

class pyesg.**Quadrangle**(*p1*, *p2*, *p3*, *p4*)

An quadrangle on the earth defined by three points p1, p2, p3, and p4. It also can be seen as the relationship between four points.

Note: p1 -> p2 -> p3 -> p4 should be rotative.

angles()

Treated as two triangles.

area()

[reference: http://mathworld.wolfram.com/SphericalPolygon.html]

class pyesg.**Triangle**(*p1*, *p2*, *p3*)

An triangle on the earth defined by three points p1, p2, and p3. It also can be seen as the relationship between three points.

angles()

Calculate the included angle between two sides on the earth. If we set a is the side p2-p3, b the side p3-p1, and c the side p1-p2. Then the return value A is the included angle betwen sides b and c.

area()

Calculate the area of the triangle bounded by the sides made by the three points p1 (lat1, lon1), p2 (lat2, lon2), and p3 (lat3, lon3) according to the Girard's Theorem: $area = R^2 * E$, where R is the radius of the sphere, and E the angle excess: $E = A + B + C - pi$. Cosine rules are used to calculate the angles A, B, and C.

[references: http://www.princeton.edu/~rvdb/WebGL/GirardThmProof.html http://en.wikipedia.org/wiki/Spherical_trigonometry http://mathforum.org/library/drmath/view/65316.html]

CHAPTER

FIVE

ABOUT

5.1 Author

5.2 License

Unless stated otherwise, all files in the PyESG project, PyESG's webpage (and wiki), all images and all documentation including this User's Guide are licensed using the new BSD license:

```
Copyright (c) 2015, Feng Zhu
All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions are met:

* Redistributions of source code must retain the above copyright notice, this
  list of conditions and the following disclaimer.

* Redistributions in binary form must reproduce the above copyright notice,
  this list of conditions and the following disclaimer in the documentation
  and/or other materials provided with the distribution.

* Neither the name of [project] nor the names of its
  contributors may be used to endorse or promote products derived from
  this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE
FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER
CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY,
OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
```

CHAPTER

SIX

INDICES AND TABLES

- *genindex*
- *modindex*
- *search*

PYTHON MODULE INDEX