**Package** ‘Name of the Package”

July 16, 2014

**Title**

**Version**

**Date**

**Dependencies**

**License**

**URL**

**System Requirements**

**Author**

**Maintainer**

**Needs Compilation**

**Repository**

**Date/Publication**

ProcTraj

**Author**

This function was developed by adapting the code provided on the OpenAir documentation.

**Description**

The ProcTraj function is responsible for all setup, execution, and data gathering of the HySplit model.

**Usage**

procTraj(lat = 51.4, lon = -45.8,

year = 2010, name = “london”,

start.month = 02, end.month = 12 ,

start.day = 10, end.day = 12,

hour.interval = "1 hour",

met = “path/to/metereological/files”,

out = “path/to/output/directory”,

hours = 3, height = 100, hy.path = “path/to/hysplit/”, ID=1 )

**Arguments**

lat, lon: numeric: Initial trajectory starting location.

year: Integer: It specifies the year of the trajectory calculation e.g. 1978. This year must agree with the year of the meteorological data file.

start.day, end.day: Integer: It specifies the day’s interval within a month for trajectory calculation. For example, for start.day = 10 and end.day = 28, it will generate trajectories between those days for the months specified by ***start.month*** and ***end.month*** with a time interval specified by ***hour.interval***.

start.month, end.month: Integer: It specifies the month’s interval of the trajectories. For example, for start.month = 1 and end.month = 3, it will generate trajectories between the months of January and March, for the days specified by ***start.month*** and ***start.day*** with a time interval specified by ***hour.interval***.

hour.interval: String value specifying the hour interval for the trajectories. E.g. “1 hour”

name: String: Name of the trajectory endpoints file.

met: String: Directory location of the meteorological file. More information concerning to meteorological files can be found in *http://www.meteozone.com/home/tutorial/html/meteo\_ftp.html*.

out: String: Directory location to which the .RData trajectory end-points file will be written. Always terminate with the appropriate slash (\ or /). An example would be out=”/home/user/Desktop/hysplit/”

hours: Integer: Total run time. It specifies the duration of the calculation in hours. Backward calculations are entered as **negative** hours. A backward trajectory starts from the trajectory termination point and proceeds upwind. Meteorological data are processed in reverse-time order. **Because only two additional meteorological files are loaded, one for the last and another for the next month, it is recommended a maximum trajectory length of 24 hours**.

height: numeric: The initial trajectories height. Height is entered as meters above ground-level.

hy.path: String: The local path where HySplit is located. Example for linux/OS X Operating Systems "/home/thalles/Desktop/hysplit/trunk/"

ID: Integer: Process ID. When called in Parallel, the ID argument ensures that each process will deal with separate set of files preventing race conditions among different processes.

**Details**

In order to make the input files for HySplit consistent, the ProcTraj function will always load 3 meteorological files for a specific month. For example, for the month of January of 2014, it will load the meteorological files from December of 2013, January of 2014, and February of 2014.

**Values**

Returns a data frame with pre-calculated HySplit forward or back trajectories.

AddMetFiles

**Author**

This function was developed by adapting the code provided on the OpenAir documentation.

**Description**

This function is called by the function ProcTraj and it is responsible for writing the location of the meteorological files in the script file that is being generated by ProcTraj. This function must not be called directly by users.

**Usage**

**Arguments**

month: Integer: It specifies the current month of the trajectory calculation.

year: Integer: It specifies the current year of the trajectory calculation e.g. 1978.

met: String: : String: Directory location of the meteorological file. More information concerning to meteorological files can be found in http://www.meteozone.com/home/tutorial/html/meteo\_ftp.html.

bat.file: String: The full directory location where the script is being written.

control.file: String: A constant string that indicates the name “CONTROL”.

**Details**

**Values**

Nothing.

ReadFiles

**Author**

This function was developed by adapting the code provided on the OpenAir documentation.

**Description**

The function ReadFiles is called by the function ProcTraj and it is responsible for reading all the output files generated by HySplit and put all that data together in an .R file with some additional data. Given the path directory of the files generated by HySplit, which is usually the directory called “working”, this function will read all the output files that concern to Forward or Backward trajectories. This function must not be called by users.

**Usage**

**Arguments**

hours: The trajectories length. **POSSIBLE IT IS NOT BEING USED**

hy.path: String: The local path where HySplit is located.

ID: Integer: Since this function is designed to parallel execution, this ID refers to the process ID and it ensures that each process will only read the files that was generated by itself.

time.zome: String: Specify the desired time zone for displaying the trajectories. “GMT” is the default time zone value. **NOT IMPLEMENTED**

**Details**

This function adds 2 new columns to the HySplit output. The first one “date” represents the trajectory initial/departing time, and the second one “date2” represents the trajectory development in time based on the trajectory hour increment previously defined.

**Value**

A object of type data frame with all the HySplit output data together.

Df2SpLinesDF **old DF2SLDF**

**Author**

**Description**

This function converts an object of type data frame, calculated by the function ProcTraj, into an Object of class SpatialLinesDataFrame.

**Usage**

crs <- CRS("+proj=longlat +datum=NAD27")

spLinesDF <- Df2SpLinesDF( spLines, data.frame, crs )

**Arguments**

spLines: Object of class SpatialLines calculated by the function Df2SpLines.

df: Data Frame Object created by the function ProcTraj.

crs: String: Valid projection string. An example would be crs="+proj=longlat +datum=NAD27".

... : Arguments passed to or from methods.

**Details**

The additional information which each line of the SpatialLinesDataFrame will have concerns to the first row of an individual trajectory in the data frame.

**Value**

Returns an object of class SpatialLinesDataFrame.

Df2SpLines

**Author**

**Description**

This function converts an object of type data frame, calculated by the function ProcTraj, into an object of type Spatial Lines.

**Usage**

crs <- CRS("+proj=longlat +datum=NAD27")

spLines <- Df2SpLines(data.frame, crs)

**Arguments**

df: Data Frame Object created by the function ProcTraj.

crs: String: Valid projection string. An example would be crs= "+proj=longlat +datum=NAD27"

... : Arguments passed to or from methods.

**Details**

An individual line consists of a set of lines in the data frame that contains the same ID. This function identifies individual trajectories based on their length. It is assumed that all trajectories calculated by HySplit using the ProcTraj function have the same length. Thus, once known the length of the trajectories, this function splits the data frame in X different data frames where each data frame contains L rows, L being the trajectory’s length and X being the number of rows in the initial data frame divided by the trajectory’s length. Each of the X different data frames will be transformed into a specific line.

**Value**

Returns an object of class SpatialLines.

PlotBgMap

PlotTraj

PlotTrajFreq

SplitSpLines

**Author**

**Description**

This function divides an object of class SpatialLines defined by the argument **sp.lines** into a number of groups defined by the argument **into**.

**Usage**

list.splines <- SplitSpLines(spLines, 8)

**Arguments**

sp.lines: Object of class SpatialLines calculated by the function Df2SpLines.

into: Integer: Number of times that the sp.lines object must be divided

**Details**

If the number provided by the argument **into** is not multiple of the number of rows in the SpatialLines object, the last element of the list will contain a SpatialLines object with more lines than the first ones. Thus, the original SpatialLines object will not be equally divided.

**Value**

Returns a list of SpatialLines Object.

Df2SpPoints **old hySplit2SPDF**

**Author**

**Description**

This function converts an object of class data frame, calculated by the function ProcTraj, into an object of class SpatialPoints.

**Usage**

crs <- CRS("+proj=longlat +datum=NAD27")

spPoints <- Df2SpPoints(data.frame, crs)

**Arguments**

df: Data Frame Object created by the function ProcTraj.

crs: Object of class "CRS"; holding a valid proj4 string.

... : Arguments passed to or from methods.

**Details**

An individual Point is represented by a row of the data frame.

**Value**

Returns an object of class SpatialPoints.