Package ‘coastMDT’

June 15, 2017

Title Estimate a Coastal MDT Based on Various Input Files

Version 0.0.1

Date 2017-03-01

Author Karina Nielsen, Ole B. Andersen, Per Knudsen, Christopher W. Hughes, Mederic Gravelle and Luciana Fenoglio-Marc.

Maintainer Karina Nielsen `<karni@space.dtu.dk>`

Description Estimate the coastal MDT. Altimetry data and tidegauge data is combined to form an improved coastal MDT. The software package coastMDT was developed as part of the deliverable D4 in the ESA STSE project ``GOCE++ Dynamic Topography at the coast and tide gauge unification (DYCOT)''. More information regarding the projects are found in the project deliverables D1-D12 which are (or will be) available from the project web page http://gocehsu.eu/. Installation instructions are found in the tutorial ``coastMDT_tutorial.pdf'' available from https://github.com/cavios/coastMDT/tree/master/doc.

License GPL-2

Imports ncdf4,
fields,
FNN,
raster,
sp,
akima,
parallel

URL `https://github.com/cavios/coastMDT`

LazyData TRUE

BugReports `https://github.com/cavios/coastMDT/issues`

RoxygenNote 6.0.1

R topics documented:

compareWithTG 2
ellipsoidTF2MT 3

getBoxMean . . . 4
getCoastLine . . . 4
getCount . . . 5
getData . . . 6
getError . . . 6
getIdMat . . . 7
getLandComb . . . 8
getLandInfo . . . 9
getLandVal . . . 9
getSubGrid . . . 10
getSubTG . . . 11
getTGCoast . . . 12
getTGid . . . 12
getTGVal . . . 13
grid2file . . . 14
iterativeAveSmoother . . . 15
iterativeAveSmootherBoot . . . 16
plotMDT . . . 17
polygonizeCoast . . . 17
readncdf1var . . . 18
readRegGridBin . . . 18
tideConvert . . . 19
wgs2topCorr . . . 20

Index **21**

`compareWithTG` *Extract MDT values at TG positions*

Description

The function extracts MDT values at the position of the tide gauges, and compares the MDT values of the field with MDT values based on the tide gauges.

Usage

```
compareWithTG(TG, dat, lonlim, latlim, boxlon = 3, boxlat = 3,
  export = FALSE, tgfile = "TGcompare.csv")
```

Arguments

`TG`	Data frame or matrix with tide gauge information. TG should contain at least the columns with the names 'Longitude', 'Latitude', and 'TGMDT'. 'TGMDT' should contain MDT values at the tide gauge positions.
`dat`	An object as returned by the function 'getSubGrid' or 'iterativeAveSmoother', which includes a list containing a matrix g[lon,lat], a vector lon (longitudes) and a vector lat (latitudes).

`lonlim`	Vector of length 2 with the longitude data grid limits, c(lonlim[1],lonlim[2]). The limits must be given in whole degrees.
`latlim`	Vector of length 2 with the longitude data grid limits, c(lonlim[1],lonlim[2]). The limits must be given in whole degrees.
`boxlon`	The number ((2 x boxlon) +1) of grid cells in the longitude direction, that is used to estimate the altimetry based MDT value at the coast.
`boxlat`	The number ((2 x boxlat) +1) of grid cells in the latitude direction, that is used to estimate the altimetry based MDT value at the coast.
`export`	If true the information estimated in the function is saved in a csv file. The default name is "TGcompare.csv". The file contains; All columns in the data frame TG,Alt_mean MDT,MDT_Alt_sd,bias corrected difference (alt-TG_bias_corr).
`tgfile`	a character string giving the name of the file.

Details

Besides the list, that is returned. A plot of the difference between the altimetry and the tide gauges MDT values is automatically generated

Value

A list that includes: mean: The mean values of the field in the box, defined by boxlon and boxlat at each tide gauge position. sd: The standard deviation of the field in the box, defined by boxlon and boxlat at each tide gauge position. bias: The bias between the mean field values and the tide gauge MDT values. diff: The difference between the mean field values and the bias corrected tide gauge MDT values RMS: The RMS of the mean field values and the tide gauge MDT values

`ellipsoidTF2MT` *Function that estimates the correction to go from a tide free ellipsoid to a mean tide ellipsoid.*

Description

Function that estimates the correction to go from a tide free ellipsoid to a mean tide ellipsoid.

Usage

```
ellipsoidTF2MT(phi, h2 = 0.62)
```

Arguments

`phi`	The latitude in degrees.
`h2`	Love number, the default value is h2=0.62

Details

...

Value

An array with the height differences in meter.

`getBoxMean` *Helper function: Find mean value in a box*

Description

Helper function: Find mean value in a box

Usage

```
getBoxMean(dat, id, boxlon = 4, boxlat = 4)
```

Arguments

`dat`	Matrix[lon,lat] with MDT values
`id`	Matrix[N,2] with row and column id, reprecenting the center of the box
`boxlon`	Integer. The number ((2 x boxlon) +1) of grid cells in the longitude direction, that is used to estimate the altimetry based MDT value at the coast.
`boxlat`	Integer. The number ((2 x boxlat) +1) of grid cells in the latitude direction, that is used to estimate the altimetry based MDT value at the coast.

Details

...

Value

list with mean and sd values

`getCoastLine` *Helper function to getLandVal: Extract the coast line from the land mask*

Description

Helper function to getLandVal: Extract the coast line from the land mask

Usage

```
getCoastLine(mask, land = 0, water = 1)
```

Arguments

`mask`	Matrix[lon,lat] with land mask values.
`land`	Integer value for land. Default is 0.
`water`	Integer value for land. Default is 1.

Details

...

Value

List with the elements; matrix g[lon,lat], which contains the location of the land value which has a water neighbor. Matrix id which contains two columns; row no and col no of the matrix g where a land value is identified as coast.

`getCount` *Helper function for the bootstrap function getError*

Description

This function returns a matrix with the number of times each element must be counted.

Usage

```
getCount(IMat, nr, nc)
```

Arguments

`IMat`	a matrix that divides the area into sub areas. It is the output of the function getIdMat.
`nr`	the number of rows in the data matrix (the raw MDT).
`nc`	the number of columns in the data matrix (the raw MDT)

Details

...

Value

matrix[lon,lat] where each element represents the number of times each elements should be counted in the iterative filter, when the sd is estimated for the filtered MDT, by the use of bootstrap.

getData — *Function to download data from the web*

Description

Function to download data from the web

Usage

```
getData(localdir = tempdir(), files = NULL,
  url = "https://raw.githubusercontent.com/cavios/coastMDT/master/data/files/")
```

Arguments

`localdir`	A character string with the name of the directory where the data will be stored. If no name is given the data is automatically stored in a temporal directory.
`files`	A character vector with the names of the files to be downloaded. If not specified all data for the coastMDT package is downloaded.
`url`	A character string with the url, that specifies where the data is located. If not specified, the url is where the data for the coastMDT package is located.

Details

...

getError — *Estimate a MDT error field via bootstrap*

Description

With the function getError it is possible to estimate an MDT error field via the bootstrap approach.

Usage

```
getError(dat, land, mask, bootNr = 100, nnx = 3, nny = 3,
  ncores = detectCores())
```

Arguments

`dat`	An object as returned by the function 'getSubGrid' or 'iterativeAveSmoother', which includes a list containing a matrix g[lon,lat], a vector lon (longitudes) and a vector lat (latitudes).
`land`	Matrix[lon,lat] containing land values

`mask`	An object as returned by the function 'getSubGrid', which includes a list containing a matrix g[lon,lat], a vector lon (longitudes) and a vector lat (latitudes). mask$g is a Matrix[lon,lat] representing the land mask, where land=0 and water=1.
`bootNr`	Number of bootstap data sets
`nnx`	number of grid cells in the east-west direction
`nny`	number of grid cells in the north-south direction
`ncores`	Number of available cores
`dat`	An object as returned by the function 'getSubGrid', which includes a list containing a matrix g[lon,lat], a vector lon (longitudes) and a vector lat (latitudes). The matrix dat$g[lon,lat] containes the values to be filtered.

Details

The data is sampled in blocks, which size is specified by the user in the arguments nnx and nny. nnx and nny should be chosen carefully. If nnx and nny are too small the data will not be independent and the error will be underestimated. If too large, there will be data gabs and the errors will not be representative.

`getIdMat` *Helper function for getError*

Description

The function getIdMat divides a matrix into sub matrices specified by the arguments nnx and nny .

Usage

```
getIdMat(mask, nnx, nny, nr, nc)
```

Arguments

`mask`	An object as returned by the function 'getSubGrid', which includes a list containing a matrix g[lon,lat], a vector lon (longitudes) and a vector lat (latitudes). mask$g is a Matrix[lon,lat] representing the land mask, where land=0 and water=1.
`nnx`	number of grid cells in the east-west direction
`nny`	number of grid cells in the north-south direction
`nr`	number of rows in the data matrix
`nc`	number of columns in the data matrix

Value

List with the elements; matrix[lon,lat] Mat (ids of the submatrices), vector mysamples (the values of the ids), and nrSam (the length of mysamples).

`getLandComb` *Helper funtion to getLandVal: Finds land values based on tide gauges and altimetry*

Description

Helper funtion to getLandVal: Finds land values based on tide gauges and altimetry

Usage

```
getLandComb(polyCoast, TG, TGcorr, dat, lonlim, latlim, boxlon = 4,
  boxlat = 4)
```

Arguments

`polyCoast`	Matrix[lon,lat], the out put of the function polygonizeCoast. The matrix contains the coastlines of the region defined by lonlim and latlim, where the integer values represents the coast id.
`TG`	Data frame or matrix with tide gauge information. TG should contain at least the columns with the names 'Longitude', 'Latitude', and 'TGMDT'. 'TGMDT' should contain MDT values at the tide gauge positions.
`TGcorr`	Vector with bias corrected tide gauge values. Obtained from the helper function getTGVal.
`dat`	Matrix[lon,lat] with MDT values
`lonlim`	Vector of length 2 with the longitude data grid limits, c(lonlim[1],lonlim[2]). The limits must be given in whole degrees.
`latlim`	Vector of length 2 with the longitude data grid limits, c(lonlim[1],lonlim[2]). The limits must be given in whole degrees.
`boxlon`	Integer. The number ((2 x boxlon) +1) of grid cells in the longitude direction, that is used to estimate the altimetry based MDT value at the coast.
`boxlat`	Integer. The number ((2 x boxlat) +1) of grid cells in the latitude direction, that is used to estimate the altimetry based MDT value at the coast.

Details

...

Value

Matrix[lon,lat] with MDT land values at the coast line, defined by polyCoast

getLandInfo — *Helper function to getLandVal: Estimate land MDT values at the coastline based on altimetry*

Description

Helper function to getLandVal: Estimate land MDT values at the coastline based on altimetry

Usage

```
getLandInfo(mycoast, mask, dat, boxlon = 4, boxlat = 4)
```

Arguments

`mycoast`	Matrix[N,2] with row and column values of the coast line. mycoast is the out put of helper function getCoastLine
`mask`	Matrix[lon,lat] representing the land mask, where land=0 and water=1
`dat`	Matrix[lon,lat] with MDT values
`boxlon`	Integer. The number ((2 x boxlon) +1) of grid cells in the longitude direction, that is used to estimate the altimetry based MDT value at the coast.
`boxlat`	Integer. The number ((2 x boxlat) +1) of grid cells in the latitude direction, that is used to estimate the altimetry based MDT value at the coast.

Details

...

Value

Matrix[N,4]; row id, col id, mean MDT value, sd of MDT

getLandVal — *Estimates MDT land values based based on altimetry, tide gauges of both*

Description

Estimates MDT land values based based on altimetry, tide gauges of both

Usage

```
getLandVal(dat, mask, lonlim, latlim, TG = NULL, type = "alt",
  intMethod = "lin", boxlon = 4, boxlat = 4)
```

Arguments

`dat`	An object as returned by the function 'getSubGrid', which includes a list containing a matrix g[lon,lat], a vector lon (longitudes) and a vector lat (latitudes).
`mask`	An object as returned by the function 'getSubGrid', which includes a list containing a matrix g[lon,lat], a vector lon (longitudes) and a vector lat (latitudes). mask$g is a Matrix[lon,lat] representing the land mask, where land=0 and water=1.
`lonlim`	Vector of length 2 with the longitude data grid limits, c(lonlim[1],lonlim[2]). The limits must be given in whole degrees
`latlim`	Vector of length 2 with the longitude data grid limits, c(lonlim[1],lonlim[2]). The limits must be given in whole degrees
`TG`	Data frame or matrix with tide gauge information. TG should contain at least the columns with the names 'Longitude', 'Latitude', and 'TGMDT'. 'TGMDT' should contain MDT values at the tide gauge positions.
`type`	Character string representing the data to be used when the land values are estimated. type="tg": tide gauge data is used to estimate the MDT land values, type="alt": altimetry data is used, and type="both": both altimetry and tide gauge data is used.
`intMethod`	Character string describing the interpolation method used. "lin": linear interpolation and "nn": nearest neighbor interpolation
`boxlon`	Integer. The number ((2 x boxlon) +1) of grid cells in the longitude direction, that is used to estimate the altimetry based MDT value at the coast.
`boxlat`	Integer. The number ((2 x boxlat) +1) of grid cells in the latitude direction, that is used to estimate the altimetry based MDT value at the coast.

Details

...

Value

Matrix[lon,lat] with land values

`getSubGrid` *Extract sub grid*

Description

Extract sub grid

Usage

```
getSubGrid(grid, lonlim, latlim, res = 0.125, glonlim = c(0 + (res/2), 360 -
  (res/2)), glatlim = c(-90 + (res/2), 90 - (res/2)))
```

Arguments

`grid`	Input grid[longitude,latitude] of type matrix.
`lonlim`	Vector of length two containing the longitude limits of the sub grid. The limits must be given with the smallest longitude first for example c(270,300), except when the 0 longitude id crossed. In this case for example c(355,10). Acceptable values are between 0 and 360.
`latlim`	Vector of length two containing the latitude limits of the sub grid.
`res`	The resolution of the input grid in decimal degrees.
`glonlim`	Vector of length two containing the longitude limits of the input grid. The default is c(0,360).
`glatlim`	Vector of length two containing the latitude limits of the input grid. The default is c(-90,90).

Value

List with the elements; matrix g (sub grid), vector lon (longitudes), vector lat (latitudes) ##' @details ...

Examples

```
## Not run: data(landmask8)
out<-getSubGrid(landmask8,c(280,300),c(30,60))
image(out$lon,out$lat,out$g)
## End(Not run)
```

`getSubTG` *Find subset of tide gauges*

Description

Find subset of tide gauges

Usage

```
getSubTG(TG, lonlim, latlim)
```

Arguments

`TG`	Data frame or matrix with tide gauge information. TG should contain at least the columns with the names 'Longitude' and 'Latitude'.
`lonlim`	Vector of length 2 with the longitude data grid limits, c(lonlim[1],lonlim[2]). The limits must be given in whole degrees.
`latlim`	Vector of length 2 with the longitude data grid limits, c(lonlim[1],lonlim[2]). The limits must be given in whole degrees.

Value

subset of TG

getTGCoast *Helper function to getLandVal: Identify Coast lines with tide gauges*

Description

Helper function to getLandVal: Identify Coast lines with tide gauges

Usage

```
getTGCoast(polyCoast, TG, lonlim, latlim)
```

Arguments

`polyCoast`	Matrix[lon,lat], the out put of the function polygonizeCoast. The matrix contains the coastlines of the region defined by lonlim and latlim, where the integer values represents the coast id.
`TG`	Data frame or matrix with tide gauge information. TG should contain at least the columns with the names 'Longitude', 'Latitude', and 'TGMDT'. 'TGMDT' should contain MDT values at the tide gauge positions.
`lonlim`	Vector of length 2 with the longitude data grid limits, c(lonlim[1],lonlim[2]). The limits must be given in whole degrees.
`latlim`	Vector of length 2 with the longitude data grid limits, c(lonlim[1],lonlim[2]). The limits must be given in whole degrees

Details

...

Value

Vector with coast id for the tide gauges.

getTGid *Helper function: Finds row and col id of Tide gauges*

Description

Helper function: Finds row and col id of Tide gauges

Usage

```
getTGid(TG, lonlim, latlim)
```

Arguments

`TG`	Data frame or matrix with tide gauge information. TG should contain at least the columns with the names 'Longitude', 'Latitude', and 'TGMDT'. 'TGMDT' should contain MDT values at the tide gauge positions.
`lonlim`	Vector of length 2 with the longitude data grid limits, c(lonlim[1],lonlim[2]). The limits must be given in whole degrees.
`latlim`	Vector of length 2 with the longitude data grid limits, c(lonlim[1],lonlim[2]). The limits must be given in whole degrees.

Value

Matrix[N,2] with row and coulmn id for the tide gauges

`getTGVal` *Helper function to getLandVal:*

Description

Extract MDT values at TG positions and estimates a potential bias between the tide gauges and the model based MDT.

Usage

```
getTGVal(TG, dat, mask, lonlim, latlim, boxlon = 4, boxlat = 4)
```

Arguments

`TG`	Data frame or matrix with tide gauge information. TG should contain at least the columns with the names 'Longitude', 'Latitude', and 'TGMDT'. 'TGMDT' should contain MDT values at the tide gauge positions.
`dat`	Matrix[lon,lat] with MDT values
`mask`	An object as returned by the function 'getSubGrid', which includes a list containing a matrix g[lon,lat], a vector lon (longitudes) and a vector lat (latitudes). mask$g is a Matrix[lon,lat] representing the land mask, where land=0 and water=1.
`lonlim`	Vector of length 2 with the longitude data grid limits, c(lonlim[1],lonlim[2]). The limits must be given in whole degrees.
`latlim`	Vector of length 2 with the longitude data grid limits, c(lonlim[1],lonlim[2]). The limits must be given in whole degrees.
`boxlon`	Integer. The number ((2 x boxlon) +1) of grid cells in the longitude direction, that is used to estimate the altimetry based MDT value at the coast.
`boxlat`	Integer. The number ((2 x boxlat) +1) of grid cells in the latitude direction, that is used to estimate the altimetry based MDT value at the coast.

Details

...

Value

list(TGland=out,bias=bias). TGland is data frame with 4 columns; row id, col id, corrected tide gauge value, sd of boxmean value of modeled MDT

`grid2file` *Write a grid[lon,lat] to a netcdf file*

Description

This function `grid2file` saves a grid[lon,lat] to a netcdf file.

Usage

```
grid2file(grid, varname = "MDT", filename = "grid.nc")
```

Arguments

`grid`	An object as returned by the function 'iterativeAveSmoother', which includes a list containing a matrix g[lon,lat], a vector lon (longitudes) and a vector lat (latitudes).
`varname`	A string containing the name of the variable, the default is 'MDT'
`filename`	A string containing the file name, the default is 'grid.nc'

Details

...

Examples

```
## Not run:
grid2file(boxTG_ALT,filename='MDT_filtered.nc')

## End(Not run)
```

`iterativeAveSmoother` *Iterative box filter*

Description

The function `iterativeAveSmoother` is a simple average filter applied nit number of times. The size of the filter in the E-W direction is scaled according to the latitude.

Usage

```
iterativeAveSmoother(dat, mask, land, radius = 0.15/0.83, nit = 10,
  res = 0.125)
```

Arguments

`dat`	An object as returned by the function 'getSubGrid', which includes a list containing a matrix g[lon,lat], a vector lon (longitudes) and a vector lat (latitudes). The matrix dat$g[lon,lat] containes the values to be filtered.
`mask`	An object as returned by the function 'getSubGrid', which includes a list containing a matrix g[lon,lat], a vector lon (longitudes) and a vector lat (latitudes). mask$g is a Matrix[lon,lat] representing the land mask, where land=0 and water=1.
`land`	Matrix[lon,lat] containing land values
`radius`	Filter radius. Default is radius=0.15/0.83
`nit`	Number of iterations of the box filter. Default is nit=10
`res`	Grid spacing of the matrix dat. Default is dlat=0.125

Details

...

Value

List with the elements; matrix[lon,lat] g (grid), vector lon (longitudes), vector lat (latitudes).

iterativeAveSmootherBoot

Iterative box filter

Description

The function `iterativeAveSmoother` is a simple average filter applied nit number of times. The size of the filter in the E-W direction is scaled according to the latitude.

Usage

```
iterativeAveSmootherBoot(dat, mask, land, radius = 0.15/0.83, nit = 10,
  res = 0.125, countMat = NULL)
```

Arguments

`dat`	An object as returned by the function 'getSubGrid', which includes a list containing a matrix g[lon,lat], a vector lon (longitudes) and a vector lat (latitudes). The matrix dat$g[lon,lat] containes the values to be filtered.
`mask`	An object as returned by the function 'getSubGrid', which includes a list containing a matrix g[lon,lat], a vector lon (longitudes) and a vector lat (latitudes). mask$g is a Matrix[lon,lat] representing the land mask, where land=0 and water=1.
`land`	Matrix[lon,lat] containing land values
`radius`	Filter radius. Default is radius=0.15/0.83
`nit`	Number of iterations of the box filter. Default is nit=10
`res`	Grid spacing of the matrix dat. Default is dlat=0.125
`countMat`	A matrix where each element represent the number of times each element i the data matrix "dat" should be counted.

Details

...

Value

List with the elements; matrix[lon,lat] g (grid), vector lon (longitudes), vector lat (latitudes).

`plotMDT` *Plot MDT grid*

Description

Plot MDT grid

Usage

```
plotMDT(dat, zlim, addContour = TRUE, conlev = 0.05, TGdat = NULL,
  legendUnit = "m", ...)
```

Arguments

`dat`	An object as returned by the function 'getSubGrid' or 'iterativeAveSmoother', which includes a list containing a matrix g[lon,lat], a vector lon (longitudes) and a vector lat (latitudes).
`zlim`	Range of the MDT values given as a Vector of length 2.
`addContour`	Bolean; To add a contour plot. Default is TRUE
`conlev`	The spacing between contour lines given in meters. The default is 0.05.
`TGdat`	MDT tide gauge data file. The file TGdat must contain at least the columns; Longitude, Latitude, TGMDT. The default is NULL.
`...`	Additional arguments to image.plot from fields

Details

...

`polygonizeCoast` *Helper function to getLandVal: Turns land mask matrix into polygons*

Description

Helper function to getLandVal: Turns land mask matrix into polygons

Usage

```
polygonizeCoast(mask, landVal = 0)
```

Arguments

`mask`	Matrix[lon,lat] with land mask. Land=0 (default) and water=1.
`landVal`	integeer representing the land value in the mask

Value

Matrix[lon,lat] whith coast line ids ##' @details ...

`readncdf1var` *Read net cdf file with one variable*

Description

This function `readncdf1var` ....

Usage

```
readncdf1var(filename)
```

Arguments

`filename`	String containing the filename

Value

Matrix var[longitude,latitude] ##' @details ...

Examples

```
## Not run:
mydat<-readncdf1var('landmask8.nc')

## End(Not run)
```

`readRegGridBin` *Function that reads a regular global binary grid.*

Description

Function that reads a regular global binary grid.

Usage

```
readRegGridBin(filename, nx = 2880, ny = 1440, res = 0.125, ...)
```

Arguments

`filename`	A character string with the file name.
`nx`	Number of rows in the grid (longitude). Deafault is nx=2880.
`ny`	Number of columns in the grid (latitude). Deafault is ny=1440.
`res`	The grid spacing in degrees. Default is res=0.125.
`...`	Additional arguments to readBin.

Details

The default is grid is a 1/8 degree grid which is 2880 longitudes by 1440 latitudes: longitudes (0.5,1.5,2.5 .... 2879.5)/8 degrees, latitudes -90 + (0.5,1.5,2.5 .... 1439.5)/8 degrees. The type is real*4, hence the length of the file i 4*nx*ny. The unit is meter.

Value

An object of the type coastMDT; a list containing a matrix g[lon,lat], a vector lon (longitudes) and a vector lat (latitudes).

`tideConvert` *Function for converting between different permanent tide systems.*

Description

Function for converting between different permanent tide systems.

Usage

```
tideConvert(phi, convtype, k = 0.3)
```

Arguments

`phi`	The latitude in degrees.
`convtype`	A character string giving the type of conversion. The legal strings are; 'MT2ZT', 'ZT2MT', 'ZT2TF', 'TF2ZT', 'MT2TF', and 'TF2MT'. Here, MT is mean tide, ZT is zero tide and TF is tide free (or nontidal)
`k`	is a love number, the default value is k=0.3

Details

The conversion expressions are based on Ekman, 1989. The correction must be added.

Value

The conversion correction; An array of height differences in meters.

`wgs2topCorr` *Function that estiamtes the height difference between WGS84 and Topex ellipsoids*

Description

Function that estiamtes the height difference between WGS84 and Topex ellipsoids

Usage

```
wgs2topCorr(phi)
```

Arguments

`phi`	The latitude in degrees.

Details

...

Value

An array with the height differences.

Index

compareWithTG, 2

ellipsoidTF2MT, 3

getBoxMean, 4
getCoastLine, 4
getCount, 5
getData, 6
getError, 6
getIdMat, 7
getLandComb, 8
getLandInfo, 9
getLandVal, 9
getSubGrid, 10
getSubTG, 11
getTGCoast, 12
getTGid, 12
getTGVal, 13
grid2file, 14

iterativeAveSmoother, 15
iterativeAveSmootherBoot, 16

plotMDT, 17
polygonizeCoast, 17

readncdf1var, 18
readRegGridBin, 18

tideConvert, 19

wgs2topCorr, 20