Getting started with the coastMDT package

Karina Nielsen

2017-12-13

Contents

Preface **2**

Installation **2**
From GitHub . . . 2
Dependencies . . . 2

Getting started with R **2**

Introduction to the package "coastMDT" **3**
Different height systems . . . 3

The geodetic MDT **3**

Using the package "coastMDT", an example **3**
Download and Load data . . . 3
Data for constructing the MDT . . . 4
Constructing the raw MDT in the mean tide system, altimetry . . . 4
Constructing the tide gauge MDT in the mean tide system relative to the TOPEX ellipsoid. . . . 5
Extraction a subsection . . . 6
Defining the land values . . . 6
Filtering . . . 7
Error estimate . . . 7
Plotting . . . 8
Save output . . . 8
Compare with tide gauge data . . . 9
Save plot to a pdf file . . . 9
Summary . . . 9

References **10**

Preface

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

The software is implemented in the open source language "R" which can be found at https://www.r-project.org/. The source code to the "R" software package coastMDT is located at GitHub at https://github.com/cavios/coastMDT

From GitHub

The easiest way to install the package is directly from Github by using the R devtools package. The R-version should be 3.2.2 or higher.

1. Open R
2. Install the devtools library by typing

```
install.packages('devtools')
```

3. Install the coastMDT package

```
devtools::install_github("cavios/coastMDT/coastMDT")
```

4. Without devtools When using this option start by installing the package dependencies (please, see below under Dependencies)

```
install.packages("https://raw.githubusercontent.com/cavios/coastMDT/master/coastMDT_0.0.1.tar.gz")
```

Dependencies

The packages depends on the following R libraries which can be installed from R with the function `install.packages`. Hence, to install the package "ncdf4"use the following command for the R window:

```
install.packages("ncdf4")
```

- ncdf4
- smoothie
- fields
- FNN
- raster
- sp
- akima

Getting started with R

The section gives a short introduction to R, which is useful to new R users.

R tutorials can be found at the r-project web side https://cran.r-project.org/manuals.html

Help pages can be accessed by typing "?" in front of a given function. If we want to access the help for the function `plot` we write

```
?plot
```

To start the web based help interface

```
help.start()
```

To exit R write

```
q()
```

Introduction to the package "coastMDT"

coastMDT is an R package built to improve the estimate of the coastal mean dynamic topography (MDT). The package combine altimetry and tide gauge based MDT values. The tide gauge based MDT values are used to constrain the MDT values at the coastline and on land. An iterative average filter is used to smooth the raw MDT. Under each iteration the land values are reset. In this introduction, examples are given to illustrate how to use the package.

Different height systems

The geodetic MDT

The geodetic MDT ξ is constructed from the MSS and the geoid.

$$\xi = MSS - N, \tag{1}$$

where N is the geoid.

To load the package simply write:

```
library(coastMDT)
```

Using the package "coastMDT", an example

This section gives a step by step guide on estimating the coastal MDT.

Download and Load data

The test data sets are kept separate from the package but can be downloaded via the function getData. The data can be saved to a directory. In the example below the data and save in the folder "MDTdat". In the example below it is assumed that the folder "MDTdat" exists. If not it must be created before. If a data directory is not specified, data is saved in a temporary directory.

```
getData() # Data is stored in a temporary directory
getData(localdir="MDTdat") # Data is stored in the directory "MDTdat"
```

A manual, "data4coastMDT.pdf", describing the available data sets are available from Github https://github.com/cavios/coastMDT/tree/master/data

The available data sets are listed below. All the grid data sets in the package are given on an 1/8 degree grid. The first cell is longitude 0 to 1/8 degree, latitude -90 to -90+1/8 degree and the order is east to west, then south to north. The data set "TG" contains information regarding the tide gauge data among other the MSL.

- **DTU15MSS**: DTU15 mean sea surface on 1/8 degree grid (0-360 degree).
- **eigen6c4r**:Geoid model based on EIGEN-6C4 on 1/8 degree grid (0-360 degree).
- **landmask8**:Land/ocean mask on 1/8 degree grid (0-360 degree).
- **difmss15eig6c4r**: Raw MDT based on the mean sea surface DTU15MSS and the geoid model eigen6c4r.
- **dDTU15MSS_ref2003_2007**: Grid to transform the DTU15 MSS (DTU15MSS) to the MSS of the reference period 2003-2007. The grid is defined on 1/8 degree grid (0-360 degree).
- **ibCor5Y_2003_2007**: Inverse barometer corrections for the 5-year reference period 2003-2007 on 1/8 degree grid (0-360 degree).
- **dacCor5Y_2003_2007**: Dynamic atmosphere correction for the 5-year reference period 2003-2007 on 1/8 degree grid (0-360 degree).
- **mean2TF_AddThis**: Grid to go from the mean tide system to the tide free system. The grid is defined on 1/8 degree grid (0-360 degree).
- **TF2mean_AddThis**: Grid to go from the tide free system to the mean tide system. The grid is defined on 1/8 degree grid (0-360 degree).
- **TG**: Tide gauge data. Please, see the data description manual "data4coastMDT.pdf" (https://github.com/cavios/coastMDT/blob/master/data/data4coastMDT.pdf) for a complete description.

The downloaded data sets can easily be loaded into R as demonstrated in the example below. It is here assumed that R is run from the directory, where the data is located.

```
load('DTU15MSS.rda')
```

The users can also import their own data in which case other appropriate R functions should be used depending on the format of the data. The package has a simple function `readncdf1var` to read one-variable NetCDF files.

Further, there is a function `readRegGridBin` that allows the user to read regular binary grids of type real4. The following example demonstrates how to use the function for the ocean MDT model "livc_5yr_p125grid.dat". Here `nx` and `ny` are the dimension of the grid

```
livc5Y<-readRegGridBin('livc_5yr_p125grid.dat', nx = 2880, ny = 1440, res = 0.125)
```

Data for constructing the MDT

The data needed to construct the coastal MDT are listed below. It is here assumed that R is run from the directory, where the data is located.

```
load('DTU15MSS.rda') # MSS
load('eigen6c4r.rda')# Geoid model
load('landmask8.rda')# land/ocean mask
load('dDTU15MSS_ref2003_2007.rda') # grid to correct the MSS to the reference period.
load('ibCor5Y_2003_2007.rda')# Inverse barometer correction
load('TG.rda')# Tide gauge MDT
```

Constructing the raw MDT in the mean tide system, altimetry

The geodetic MDT ξ is constructed from the MSS and the geoid.

$$\xi = MSS - N, \tag{2}$$

where N is the geoid. In R the two grids are simply subtracted.

```
MDTraw<-DTU15MSS-eigen6c4r
```

The data sets DTU15MSS and eigen6c4r are referenced to the TOPEX ellipsoid and are in the mean tide system.

When working with different data types such as altimetry and tide gauge data it is important to make sure that the data are in the same height reference system, time period and that the same corrections has been applied. In this example we will work with tide gauge data in covering the period 2003-2007. Hence the DTU15 MSS needs to be converted to the same reference period.

The inverse barometer correction is applied to the MSS. Hence it must be re-added to the MSS in case the tide gauge data are not IB corrected. In the example below the MSS is corrected for the reference period and the IB effect is re-added to the MSS.

```
MDTraw<-DTU15MSS+dDTU15MSS_ref2003_2007+ibCor5Y_2003_2007-eigen6c4r
```

Constructing the tide gauge MDT in the mean tide system relative to the TOPEX ellipsoid.

Altimetry products are normally given in the mean tide system, while tide gauge products are given in a tide free system. The height difference between the mean tide and the tide free ellipsoid height is given by (Ekman 1989)

$$h_m - h_f = h_2(0.099 - 0.296 * \sin(\phi)^2). \tag{3}$$

Here ϕ is the latitude and $h_2 = 0.62$. This expression is implemented in the function `ellipsoidTF2MT`

The altimetry MSS is referenced relative to the TOPEX ellipsoid, while the tide gauge data is referenced relative to WGS84. Hence we convert the tide gauge data to the TOPEX reference ellipsoid. The following expression approximates the height difference Δh assuming that the difference in latitude is very small.

$$\Delta h = -((aTop - aWGS)cos(\phi)^2 + (bTop - bWGS)sin(\phi)^2). \tag{4}$$

Here ϕ is the latitude, $aTop$ and $aWGS$ is the equatorial radius of TOPEX and WGS84, respectively, and $bTop$ and $bWGS$ is the polar radius of TOPEX and WGS84, respectively.

Finally, the tide gauge based MDT can be calculated as

```
# TG$RLR_ell_2005.5+TG$MSL_2003_2007, MSL above WGS84
# ellipsoidTF2MT(TG$Latitude), conversion of tide free to mean tide ellipsoid
# wgs2topCorr(TG$Latitude),  conversion of WGS84 to TOPEX ellipsoid
TGMDT<-TG$RLR_ell_2005.5+TG$MSL_2003_2007+ ellipsoidTF2MT(TG$Latitude)+
       wgs2topCorr(TG$Latitude)-TG$eigen6c4rC
TG<-cbind(TG,TGMDT) # Attaching the MDT values to the rest of the TG data set.
```

The altimetry and tide gauge MDTs should now be given in the same height reference system, in the same reference period, and the same correction should be applied.

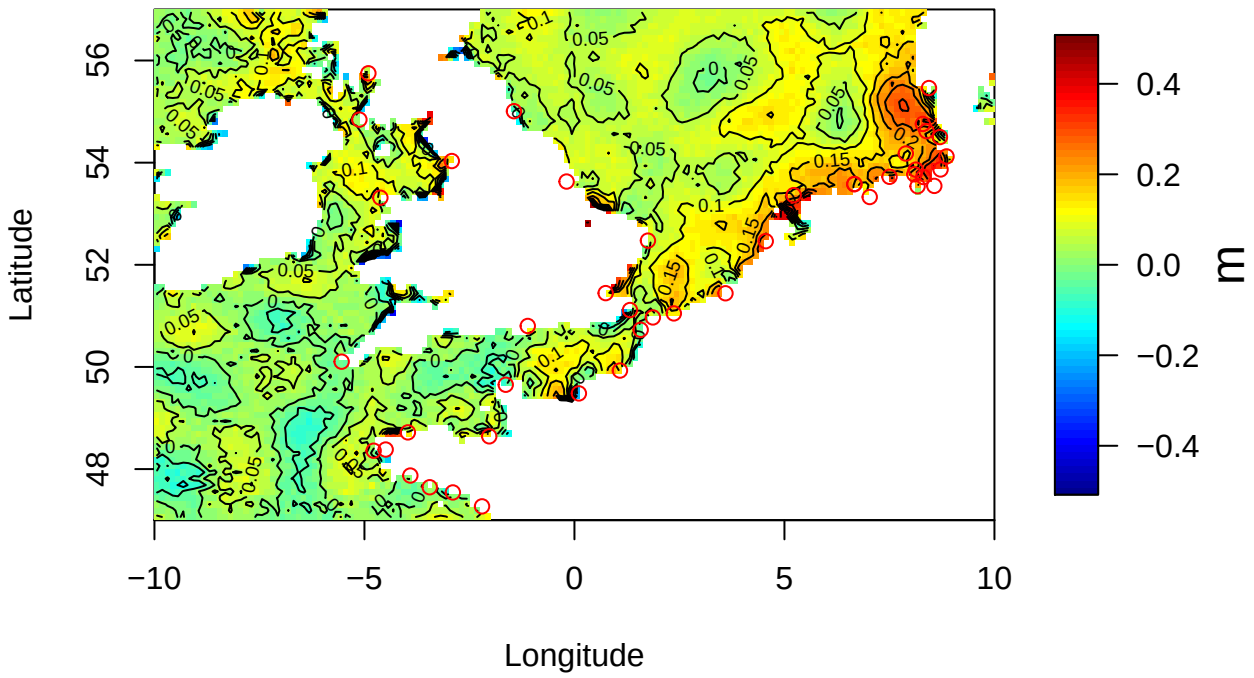

Figure 1: The raw MDT and the location of tide gauges

Extraction a subsection

The available data sets are global,but the user has the possibility to extract regional grids with the function `getSubGrid`. In the example below the coast of Northern Europe is extracted from the raw MDT and the land/ocean mask. The tide gauge data can be extracted with the function `getSubTG`.

```
#Region of interest; here the Northern Europe
lonlim<-c(350,10)
latlim<-c(47,57)
#sub grids and data
rawSub<-getSubGrid(MDTraw,lonlim,latlim)
mask<-getSubGrid(landmask8,lonlim,latlim)
TGsub<-getSubTG(TG,lonlim,latlim)
```

Defining the land values

MDT Land values are estimated with the function `getLandVal`. The land values may be based on tide gauge MDT values, the altimetry alone, or a combination of both. This option is specified with the parameter `type=("tg","alt", or "both")`. In the example below both tige gauge values and altimetry is used to contrain the land MDT values, see figure 2.

```
mylandTG_ALT <- getLandVal(rawSub, mask, lonlim, latlim, TG = TGsub, type = "both",
    intMethod = "lin")
```

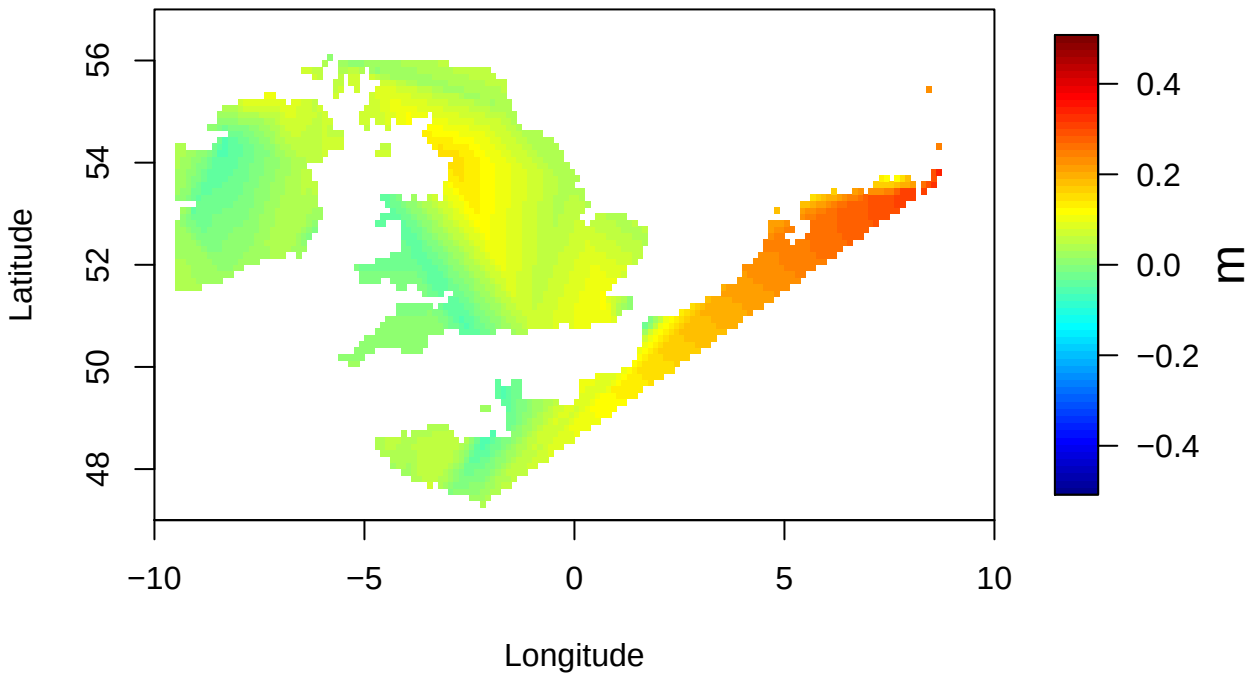

Figure 2: Land values based on tide gauge and altimetry data

```
plotMDT(mylandTG_ALT, c(-0.5, 0.5), addContour = FALSE)
```

Filtering

After the land values are estimated we can start to filter the raw MDT. The package applies an iterative average filter. After each iteration the land values are reset to the original values. It is possible to vary the radius of the filter and the number of iterations that is used. In the example below the default values are used.

```
boxTG_ALT <- iterativeAveSmoother(rawSub, mask, mylandTG_ALT)
```

Error estimate

Due to the complexity of estimating the MDT it is not straight forward to estimate the corresponding error field of the MDT. However, the package allows to user to estimate an error field by using the bootstrap approach. The raw MDT of the selected region is divided into blocks. To create new bootstrap data sets we sample with replacement between the different blocks of data. Each bootstrap data set is filtered. This gives us N different estimates of the MDT in each grid cell from which the standard deviation can be estimated. The size of the data blocks is specified by the user. The block size should be chosen large enough to ensure data independence but not much larger than the size of the filter. If the blocks are to large there will be too

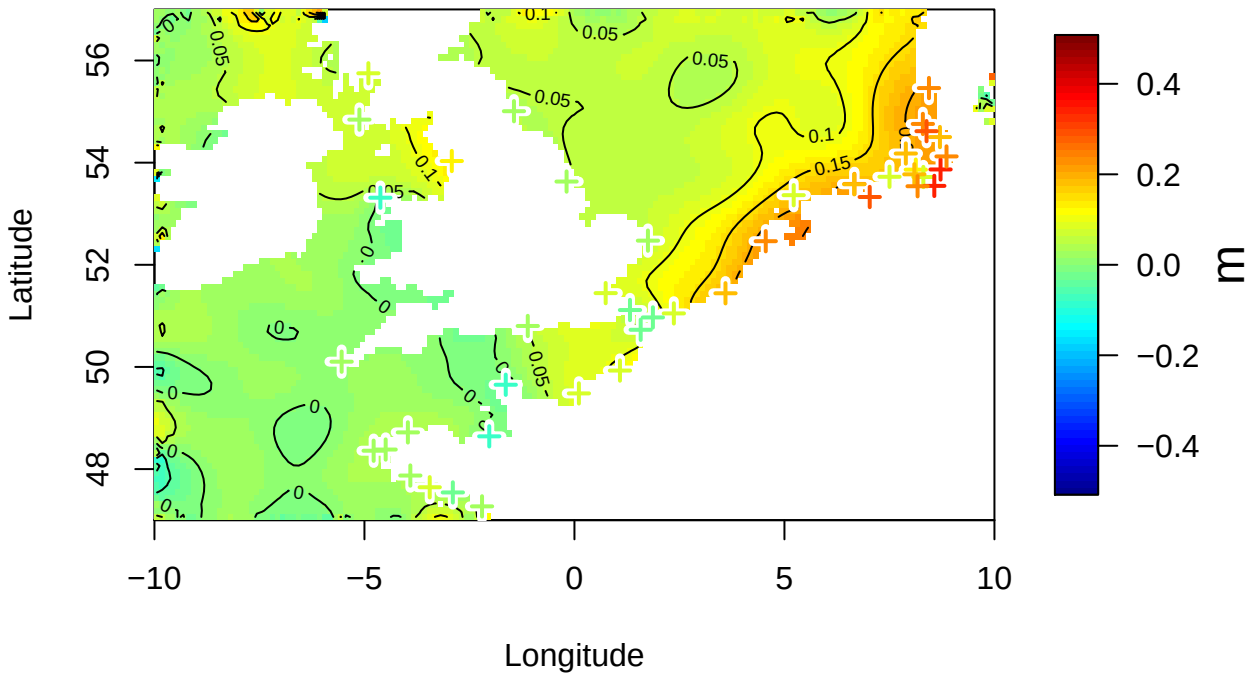

Figure 3: The filtered MDT. The red crosses displays the positions of the tide gauges

large gabs in the filtered MDT and the error will be over estimated. The example below illustrates how to use the function `getError`.

```
sdMDT <- getError(rawSub, mylandTG_ALT, mask, bootNr = 50, nnx = 3, nny = 3)
```

Plotting

The package includes a simple plotting function, so the output MDT can easily be displayed. In this function it is also possible to plot the tide gauge MDT values in the same color scale defined by the option `zlim`, by adding the argument `TGdat`. The example below displays the MDT for the recurring example

```
plotMDT(boxTG_ALT, zlim = c(-0.5, 0.5), conlev = 0.05, TGdat = TGsub)
```

Save output

The filtered MDT can be saved to a NetCDF file with the function `grid2file`

```
grid2file(boxTG_ALT, filename = "MDT_TG_Alt_NEurope.nc")
```

Compare with tide gauge data

The function `compareWithTG` allows to compare the Altimetry based MDT with the tide gauge data. The comparison can be saved to a file with the following command.

```
outTG <- compareWithTG(TGsub, boxTG_ALT, lonlim, latlim, export = TRUE, tgfile = "myfile.csv")
```

Save plot to a pdf file

The MDT plot can be saved to a pdf file with the function `pdf`, see `?pdf` for help

```
pdf("myMDT.pdf")
plotMDT(boxTG_ALT, c(-0.5, 0.5), conlev = 0.05)
points(TGsub$Longitude, TGsub$Latitude, col = "red", pch = 3)
dev.off()
```

Summary

The compleete example is summarized below

```
library(coastMDT)
# download data
getData()
# load data
load("DTU15MSS.rda")
load("dDTU15MSS_ref2003_2007.rda")
load("eigen6c4r.rda")
load("ibCor5Y_2003_2007.rda")
load("landmask8.rda")
load("TG.rda")
# construct the raw altimetry MDT
raw <- DTU15MSS + dDTU15MSS_ref2003_2007 + ibCor5Y_2003_2007 - eigen6c4r
# construct tide gauge MDT
TGMDT <- TG$RLR_ell_2005.5 + TG$MSL_2003_2007 + ellipsoidTF2MT(TG$Latitude) +
    wgs2topCorr(TG$Latitude) - TG$eigen6c4rC
TG <- cbind(TG, TGMDT)  # Attaching the MDT values to the rest of the TG data set.
# define sub region
lonlim <- c(350, 10)
latlim <- c(47, 57)
# extract sub grids
mask <- getSubGrid(landmask8, lonlim, latlim)
rawSub <- getSubGrid(raw, lonlim, latlim)
TGsub <- getSubTG(TG, lonlim, latlim)
# define land values
mylandTG_ALT <- getLandVal(rawSub, mask, lonlim, latlim, TG = TGsub, type = "both",
    intMethod = "lin")
# filter MDT
boxTG_ALT <- iterativeAveSmoother(rawSub, mask, mylandTG_ALT)
# plotting
plotMDT(boxTG_ALT, c(-0.5, 0.5), conlev = 0.05, TGdat = TGsub)
```

References

Ekman, Martin. 1989. “Impacts of geodynamic phenomena on systems for height and gravity.” *Bulletin Géodésique* 63 (3). Springer-Verlag: 281–96. doi:10.1007/BF02520477.