Draft

GOAT

(Geophysical Observation Analysis Tool)

version **v2014F**

GUIDE

Written by

Ori Adam

Updated Monday, June 16, 2014

Contents

page

Introduction

My interest in observational analysis began sometime late 2011. My first project was to calculate the difference in precipitation climatology between two commonly used reanalyses: (NCEP-I and ERA-40).

It took me a week to accomplish this seemingly simple and routine task. I was annoyed by the fact that I spent the better part of the week dealing with the technical aspects of the task (such as format conversion and unit standardization), rather than the scientific aspects. Considering I had planned many more such calculations, involving several climate models and reanalyses, it seemed like I was about to waste a considerable amount of time doing similar calculations.

I searched for a tool or a framework that would make such calculations more accessible but could not find anything capable of handling the terrible lack of standardization across datasets. The plethora of available tools, blogs and dedicated sites was of some help, but was also indicative of the gravity of the problem. Nili Harnik, my postdoc advisor at the time, suggested that an attempt to consolidate some analysis code and data archives is worth the effort. A couple of months later I presented her with the first version of GOAT.

GOAT (Geophysical Observation Analysis Tool) is a tool designed with the key objective of standardizing and simplifying the access to various observational datasets. Users with minimal background in geophysical sciences will (I hope) find it easy and self explanatory to explore and visualize geophysical data using the graphic user-interface (GUI). Advanced researchers can make use of the many generic functionalities of the GOAT coding environment.

Work on GOAT is by no means complete. Nonetheless, I do believe it has reached that level where other scientists can benefit from it and further its development.

As with any piece of software, bugs and errors are unavoidable. I urge users to be critical of GOAT output and data. Bug reports, suggestions, and GOAT scripts are welcome.

Please note that GOAT is not a proprietary data source. Relevant data sources should be properly cited.

Finally, any element of GOAT is free to use, modify, copy, or distribute, so long as the original developers are acknowledged.

With best intentions,

Ori Adam

Acknowledgements

The idea of a MATLAB based unified platform for visualization and analysis of geophysical data was prompted by Nili Harnik, who has closely guided the development of GOAT ever since its inception early 2012. Additional contributors are:
Tapio Schneider (scientific guidance)
Gilad Sitton (logo and graphical advice)
Yair Altman (MATLAB, Java and GUI advice)

1.1 Installing GOAT

Download and unzip the latest version from *www.goat-geo.org*.

Automated
The fastest way to install goat is by running *GOAT_Install.m*.

- Place the GOAT folder where you would like GOAT to be installed
- Open MATLAB and set working directory to … *GOAT/Installation*
- Type 'GOAT_Install' at the command window.
- The automated install will set the appropriate paths and add a few shortcuts at a newly created GOAT shortcut tab (for MATLAB versions earlier than 2012 it will ask permission to add shortcuts at the shortcut toolbar).

Custom
On some machines the *savepath* function fails due to write permissions of the *pathdef.m* file. You will need to select a new destination for the *pathdef.m* file or edit permissions. To install GOAT, simply add the following paths:
GOAT/Code
GOAT/Scripts
GOAT/MyScripts
with all subfolders. This can be done by typing the following at the command window

```
>> addpath(genpath(fullfile(GOATroot,'Code')))     ;
>> addpath(genpath(fullfile(GOATroot,'Scripts')))  ;
>> addpath(genpath(fullfile(GOATroot,'MyScripts')))  ;
```

where GOATroot is the address of the 'GOAT' folder.

To launch GOAT, type 'GOAT' at the command window. The default GOAT GUI will appear. The size of the window and data folder preferences can be customized at *GOAT/MyScripts/GOAT_MyDefaults.m.*
Type
```
>> edit GOAT_MyDefaults.m
```
at the command window to edit GOAT preferences

1.2 Updating GOAT

There are 2 ways of updating GOAT without affecting existing preferences, Virtual fields, Data and WORK folders.

a) press the *GOAT_Update* shortcut or run the scripts *GOAT_VersionUpdateCode.m* from the directory …*/GOAT/Code*. This code downloads the latest update code directly from *goat-geo.org* and installs the latest GOAT version.
b) run the script *GOAT_VersionUpdate* from the directory …*/GOAT/Code*. This code downloads and installs the latest GOAT version from *goat-geo.org*.

1.3 Personal Preferences (GOAT/MyScripts/GOAT_MyDefaults.m)

Field	Description
G.pos	A figure size vector for general purposes
G.Saving.PapersizeWide	Page size option for exported pdf figures
G.Saving.PapersizeTall	Page size option for exported pdf figures
G.Saving.PapersizeSquare	Page size option for exported pdf figures
G.Saving.PapersizeGUI	Page size option for exported pdf figures (GUI size)
G.Saving.dpdfResolution	resolution of exported pdfs ('-dpdf' renderer)
G.TextSize_XL	Largest font size shown in GUI
G.TextSize_L	(relatively) Large GUI fonts
G.TextSize_M	(relatively) Medium sized GUI fonts
G.TextSize_S	(relatively) Small GUI fonts
G.GUI.Fonts.gcaFont	axis font size in GUI figures
G.GUI.Fonts.uiButtonFont	font type of text shown on GUI buttons
G.GUI.Fonts.uiHeaderFont	font type of GUI headers
G.GUI.Fonts.MessageFont	font type of GUI message area
G.topo.ShadeColor	Color of shaded land areas
G.topo.alpha	Visibility of land areas
G.topo.Lakealpha	Visibility of Lakes
G.GUI.pos.Main	position vector of main GUI
G.GUI.pos.Export	position vector of Export GUI
G.GUI.pos.Limits	position vector of Limits GUI
G.GUI.pos.Filter	position vector of Filter GUI
G.GUI.pos.Settings	position vector of Settings GUI
G.DataFoldersAddress	Commented by default. A cell array of folders containing several datasets.
G.ModelFoldersAddress	Commented by default. A cell array of dataset addresses
G.Settings.KeyPressFlag	Flag for enabling/disabling keypress (1/0)
G.Settings.webLinks	Optional. Web links accessible via the settings GUI.
G.Settings.webLinksTitle	Web link title.

2.1 GOAT standard archiving formats

GOAT is engineered around the assumption that all data is archived in a consistent standard.
GOAT accepts two data archiving formats:

1. MAT: data is archived as *.mat files on some local drive
2. LINK: data from an OPeNDAP source, *.nc files (Network Common Data Form -NetCDF) available via ftp, http or a local drive, is 'linked' to GOAT.

The MAT archiving standard is described first. The LINK archiving standard is described in section 2.5 in reference to the MAT format.

For optimized disk space usage and performance, MAT format data is preferably saved as single type variables. The units and variable naming standard is adopted from the Climate Model Intercomparison Project, phase 5 (CMIP5) archiving standard (see more at *http://cmip-pcmdi.llnl.gov/cmip5/data_description.html*).

GOAT currently supports only spatially structured (i.e. spatial ordinates are 1-dimensional vectors) data at three temporal archiving 'Calendars': '*Gregorian', 'Julian'* and '*List'* (described below).

2.2 Gregorian calendar (MAT format)

In Gregorian GOAT calendar, filenames are given by:
*[Month*10000 + Day*100 + Hour].mat*
where each file contains the spatial data of a single date (herein also referred to as a 'frame') with dimensions ordered by (*longitude , latitude , vertical*). The data matrix in a frame file of any field is always named 'data', with the exception of the '*BoerBeta*' field, for which the data matrix is named 'bata' (more on the BoerBeta field in section 2.7).
Each frame data file must have the following attributes:
DataFolder: Dataset root folder

Calendar: Currently supported calendar types are
- *'Monthly' :* Monthly averages
- *'Daily' :* Daily averages
- *'DailyX1' :* 1 time daily sampling (usually at 00:00H)
- *'DailyX2' :* 2 times daily sampling (usually at 00:00H and 12:00H)
- *'DailyX4' :* 4 times daily sampling (00, 06, 12 and 18:00H)
- *'DailyX8' :* 8 times daily sampling (00, 03, 06, 09, 12, 15, 18, and 21:00H)
- *'DailyX12' :* 12 times daily sampling (00, 02, 04, ... 22:00H)
- *'DailyX24' :* 24 times daily sampling (00, 01, 02, ... 23:00H)
- *'Daily365' :* Same*'Daily'* but all years are non leap-years.
- *'Daily365XNN' :* Same as all of the *'DailyXNN'* (NN = 1,2,4,8,12,24) temporal resolutions but all years are non leap-years.
- *'Daily360' :* Same*'Daily'* but with 30 days in each month and no leap-years.
- *'Daily360XNN' :* Same as all of the *'DailyXNN'* (NN = 1,2,4,8,12,24) temporal resolutions but with 30 days in each month and no leap-years.

Field: Name of the field, preferably in CMIP5 archiving standard.

Year: A positive integer

Dims file: A File named *Dims.mat*, located under the field folder, containing the following fields:

Units (string):	Field units
FieldDescription (string):	A brief description of the field
P (vector):	Vertical ordinate
lon (vector):	Longitudes
lat (vector):	Latitudes
Plabel (string, optional)	Level ordinate string (default = '*Level*')
lonlabel (string, optional)	Horizontal axis ordinate string (default = '*Longitude*')
latlabel (string, optional)	Vertical axis ordinate string (default = '*Latitude*')
VerticalOrdinate (string):	Currently supported vertical ordinates are

'Pressure (mb)' :	Isobaric coordinates [hPa]
'Height (m)' :	Geometric height in [m]
'Depth (m)' :	Geometric depth in [m]
'Sigma' :	Sigma coordinates [unitless]
'Potential Temperature (K)':	Isentropic coordinates in [K]

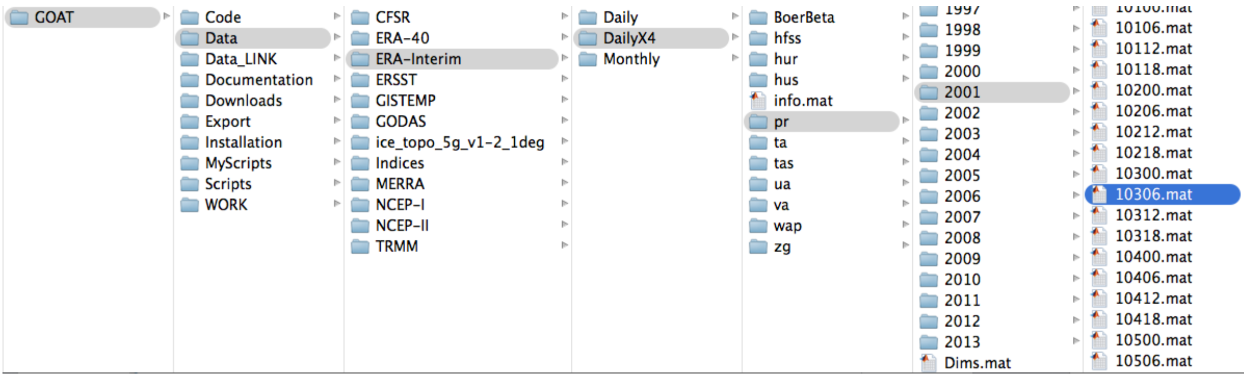
Topography file: A File named *Topography.mat*, located under the field folder name.

Obligatory variables of the *Topography.mat* file are:
latg (vector): longitudes of coasts
long (vector): latitudes of coasts
topo (2D matrix): topography map
topolon (vector): topography longitudes
topolat (vector): topography latitudes
Optional variables of the *Topography.mat* file are:
LakesIdx (vector): indices of lake coasts within the '*latg'* and '*long'* vectors
AntLonAppendix (vector): appended vector to *'long'* required to 'close' the continent of Antarctica in some projections
AntLatAppendix (vector): same as AntLonAppendix
topoType (scalar): default topography display. 1) None, 2) Atmo, 3) Ocean, 4) Ocean+Atmo
topoInterval (scalar): interval between topography lines
topoIntervalMin (scalar): minimal topography elevation
topoIntervalMax(scalar): maximal topography elevation

The default *Topography.mat* file is located at *GOAT/Code/Topography.mat*.

Info file: A file named *info.mat* which can be placed in either the Calendar or Field folders. If *info.mat* files exist in both the Calendar and Field folders, the Field *info.mat* file is preferred. The file *info.mat* contains the the string variable '*infostr*' (can be formatted using sprintf) which is displayed as a 'tooltip' when hovering above the info ([i]) element in the GUI.

The 'Finder' (Mac file browsing utility) screenshot shown bellow, displays an example of the gregorian archiving standard. In this example (left to right), DataFolder is '*ERA-Interim*', Calendar is '*DailyX4*', Field is '*pr*' (precipitation rate), *Year* is 2001, Month is *Jan*, Day is 03 and Hour is 06:00. The matfile '*10306.mat*' contains a 2D matrix (*lon* , *lat*) named '*data*' of the precipitation field on Jan.3.2001 at 06:00.

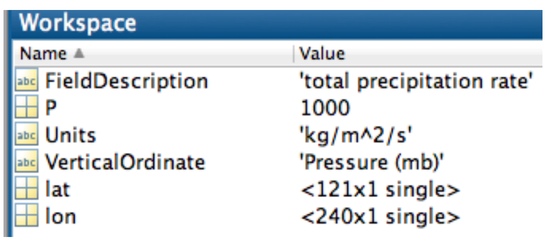


Loading the matfile '*10306.mat*' onto the workspace yields

Workspace

Name ▲	Value
data	<240x121 single>

Similarly, loading *GOAT/Data/ERA-Interim/DailyX4/pr/Dims.mat* yields

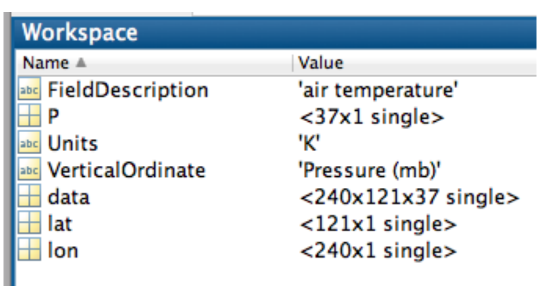


Workspace

Name ▲	Value
FieldDescription	'total precipitation rate'
P	1000
Units	'kg/m^2/s'
VerticalOrdinate	'Pressure (mb)'
lat	<121x1 single>
lon	<240x1 single>

Note the the fields '*lonlabel*', '*latlabel*', and '*Plabel*' are not included in the *Dims* file so that default ordinate labels are used: '*Longitude*', '*Latitude*', and '*Level*' for *lon* (x-axis), *lat* (y-axis) and *P* (vertical axis), respectively.

Loading the equivalent *Dims.mat* file for the 3D field '*ta*' (air temperature) yields



Workspace

Name ▲	Value
FieldDescription	'air temperature'
P	<37x1 single>
Units	'K'
VerticalOrdinate	'Pressure (mb)'
data	<240x121x37 single>
lat	<121x1 single>
lon	<240x1 single>

where we notice the added vertical dimension (37 pressure levels).

the *info.mat* file contains the string field '*infostr*', with a description of the dataset. Note that by creating '*infostr*' using the MATLAB function '*sprintf*', the string '*infostr*' can be displayed as a multi-line string.

typing the command
>> load Topography
at the command window will load *GOAT/Code/Topography.mat* to the workspace.

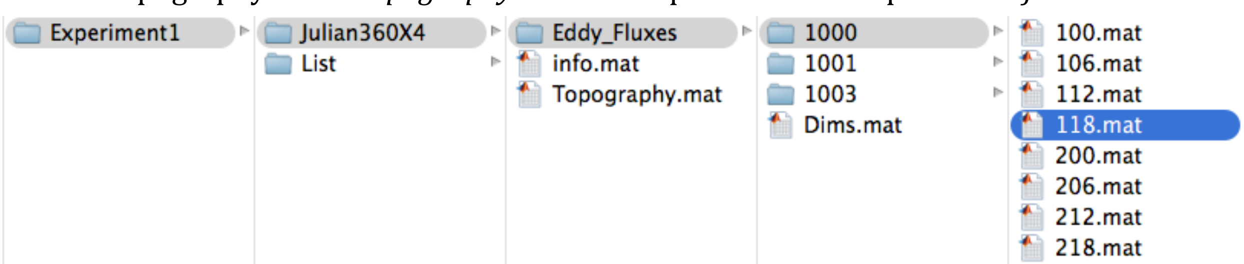
with the default (present day) coast and topography.

2.3 Julian Calendar (MAT format)

The Julian calendar differs from the Gregorian calendar in the following specifications:
filenames are given by:
*[Day*100 + Hour].mat*
Calendar:

'Julian' : 365 Julian days per year
'JulianXNN' : 365 days per year, NN times daily (NN = 1,2,4,8,12,24)
'JulianYYY : YYY days per year, daily averages
'JulianYYYXNN' : YYY days per year, NN times daily (NN = 1,2,4,8,12,24)

Shown below is an example of an experiment output at a '*Julian360X4*' Calendar with custom topography set at *Topography.mat* and experiment description at *info.mat*.



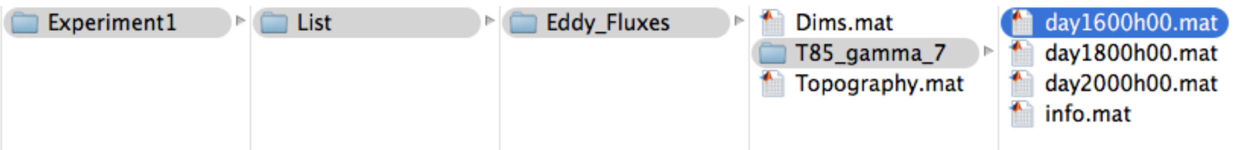
Here the dataset folder is '*Experiment1*', the Calendar *Julian360X4* refers to a Julian calendar with 360 days per year and 4 times daily data, and the field is '*Eddy_Fluxes*'. Only the first two days are shown.

The Julian calendar is the same as the Gregorian calendar in all other aspects (including the *Dims.mat, info.mat*, and *Topography.mat* files).

2.4 List archiving standard (MAT format)

The List calendar differs from the Gregorian calendar in the following specifications:
Calendar = 'List' or 'list'
Year = any string, referred to herein as a 'Particular List Folder'.
Filename = any string

An example of data archived in List calendar is shown below. The data frames refer to statistically steady states after 1600, 1800 and 2000 days, making the List Calendar the obvious choice for archiving the experiment. In this example (left to right), DataFolder is '*Experiment1*', Calendar is '*List*', Field is '*Eddy_Fluxes*', particular List is '*T85_gamma_7*'.



The List calendar is the same as the Gregorian calendar in all other aspects (including the *Dims.mat*, *info.mat* and *Topography.mat* files).

The List calendar is useful for archiving invariants, statistical derivatives such as climatological means, statistically steady states of model simulations, or any data that is inconsistent with the Julian or Gregorian calendars. The List calendar folder can reside alongside the Gregorian or Julian Calendar folders.

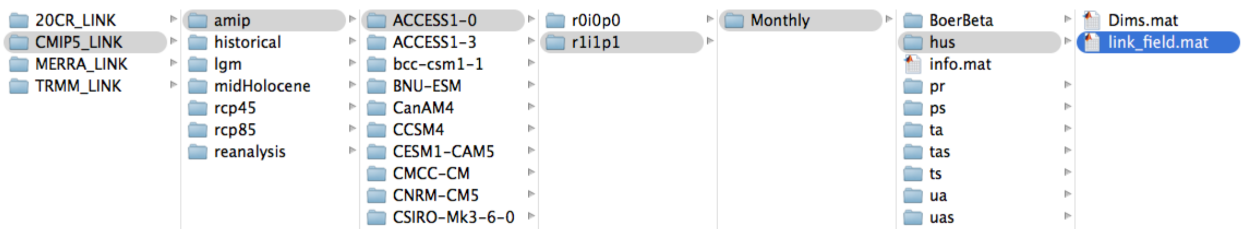
Note that the *info.mat* file is included in the Particular List folder. GOAT ignores *Topography.mat* and *info.mat* that exist in Particular List folders. Moreover, the info.mat can be placed in both the Particular List folder, the Field Folder, or the Calendar folder. GOAT will use the *info.mat* (and *Topography.mat*) files branching farthest from the Calendar folder.

2.5 LINK archiving standard

OPeNDAP sources and NetCDF files readable using the '*ncread*' MATLAB function, can be 'linked' to GOAT without the need to convert them to *.mat format. The protocol for linking OPeNDAP and NetCDF files to GOAT - the Auto Link Protocol (ALP) - is described below.

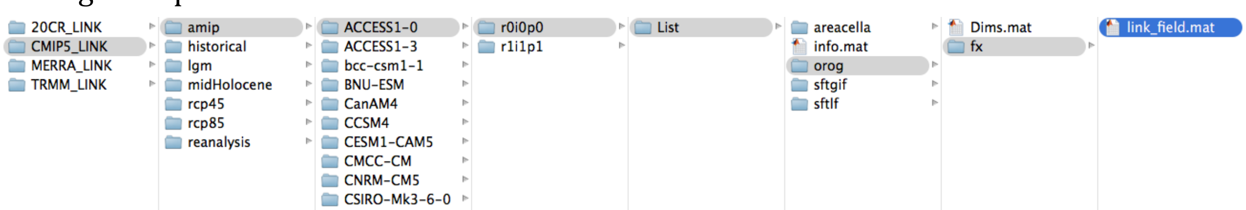
Linked datasets are archived in the same manner as *.mat datasets with the exception that data files (i.e. all data frames) are replaced by the file *link_field.mat*, containing all the necessary information in order to retrieve the data. The actual data may be an OPeNDAP source, ftp source, a file downloadable via http, or a NetCDF file saved on some local drive. An optional *link_info.txt* file may also be added next to the *link_field.mat* containing information regarding the linked file.

Shown below is an example of a linked CMIP5 model dataset. In this example, DataFolder is '*r1i1p1*', Calendar is '*Monthly*', and Field is '*hus*' (specifiv humidity). The Years folders are replaced by the file *link_field.mat* containing links to the actual *.nc files.



Note that the *Dims.mat*, *info.mat*, and *Topography.mat* files are the same as in the MAT format.

List calendars in the LINK format differ from linked Julian and Gregorian calendars in that the *link_field.mat* file is located at the Particular List folder (the equivalent of a Year folder) and not alongside the *Dims.mat* file, as shown below. This is done in order to enable having multiple Particular Lists for the same field.



The *link_field.mat* field contains a 2D cell array named '*Attributes*'. As described further below, the *Attributes* cell array can be 'Compressed' in order to minimize disk usage. The following describes the *Attributes* array in its decompressed mode.

Attributes array

Each row in the *Attributes* array corresponds to a single frame file. For the Gregorian and Julian Calendars, the *Attributes* array always begins at the first date of a Calendar Year and ends with the last date of a Calendar Year. This means that, for example, for a *Daily* dataset which begins at Feb.1, the first 31 rows are null. An additional row is adde at the end of the *Attributes* array in which additional global field attributes are assigned.

Each column in the *Attributes* array contains a specific descriptor

Column 1: File Address (or URL)

Column 2: field name (the name used to read the data from the file)

Column 3: start indices. (see help on '*ncread*' for more information on start indices)

Column 4: count indices. (see help on '*ncread*' for more information on count indices)

Column 5: Factor and Offset. If the variable contained in Column 5 is a scalar, it is interpreted as a Factor multiplying the retrieved data. If Factor is a two element vector, the first value is interpreted as a factor multiplying the retrieved data, and the second as an offset.

Column 6 (Optional, default = empty): Dimension Permute indices (e.g., if the original form of the data is lat×lon×Pressure, Permute = [2 1 3])

Column 7 (Optional, default = empty): Add Dimension (Reshape). (e.g., if the original form of the data is lat×Pressure, Reshape = 1. Similarly, if the original form of the data is lon×Pressure, Reshape = 2).

Column 8 (Optional, default = empty): Valid Range. Minimal and maximal valid values. Values above or below the Valid Range are set to Nan. If only one value is given as the Valid Range, it is taken as the minimal valid value and the maximal valid value is set to inf.

Column 9 (Optional, default = empty): Contains frame specific attribute pairs for ECMWF MARS server 'retrieve' function.

Column 10 (Optional, default = empty): Applicable only in List Calendar. For each row, if a string appears at column 10, it is displayed as the frame name. If column 10 is empty or non-existent, the frame name is taken from Column 1.

The last row of the Attributes array contains the following attributes:

Column 1: *link type*: possible link types are

- 'dap': indicating that Column1 entries are OPeNDAP URLs
- 'ncf': indicating that Column1 entries are NetCDF file addresses
- 'dnc': indicating that Column1 entries are http links to downloadable NetCDF files.
- 'eri': Unique to ECMWF MARS server. Column1 entries are date strings.

Column 2: Year span. For example, {[1979 2014]} indicates that the first and last years of the dataset are 1979 and 2014, respectively. This entry is not necessary for List Calendars.

Column 3: A structure named *Defs* containing the following additional definitions:

- *Defs.ReferenceSettingsField*: A string of a reference field from which to borrow the display settings such as colormap, shading, linewidth, etc.
- *Defs.evalStr_load*: a string to be evaluated using *eval(Defs.evalStr_load)* immediately after the Attributes are loaded. This enables updating the linking GOAT to Rotating or other time varying OPeNDAP sources that require updated links. For example, the Defs.evalStr_load may contain a command that edits the OPeNDAP URL to contain today's date.

Column 4: Calendar (e.g., '*Daily*', '*List*', etc.)

Shown below is an example of the Attributes of the last year (2014) of the Monthly means of the surface pressure field ('*ps*') from the MERRA reanalysis *tavgM_2d_slv_Nx* subset.

	1	2	3	4	5	6	7
421	'http://goldsmr2.sci.gsfc.nasa.gov:80/dods/MATMNXSLV'	'ps'	[1,1,421]	[Inf,Inf,1]	1	[]	[]
422	'http://goldsmr2.sci.gsfc.nasa.gov:80/dods/MATMNXSLV'	'ps'	[1,1,422]	[Inf,Inf,1]	1	[]	[]
423	'http://goldsmr2.sci.gsfc.nasa.gov:80/dods/MATMNXSLV'	'ps'	[1,1,423]	[Inf,Inf,1]	1	[]	[]
424	'http://goldsmr2.sci.gsfc.nasa.gov:80/dods/MATMNXSLV'	'ps'	[1,1,424]	[Inf,Inf,1]	1	[]	[]
425	'http://goldsmr2.sci.gsfc.nasa.gov:80/dods/MATMNXSLV'	'ps'	[1,1,425]	[Inf,Inf,1]	1	[]	[]
426	'http://goldsmr2.sci.gsfc.nasa.gov:80/dods/MATMNXSLV'	'ps'	[1,1,426]	[Inf,Inf,1]	1	[]	[]
427	'http://goldsmr2.sci.gsfc.nasa.gov:80/dods/MATMNXSLV'	'ps'	[1,1,427]	[Inf,Inf,1]	1	[]	[]
428	'http://goldsmr2.sci.gsfc.nasa.gov:80/dods/MATMNXSLV'	'ps'	[1,1,428]	[Inf,Inf,1]	1	[]	[]
429	'http://goldsmr2.sci.gsfc.nasa.gov:80/dods/MATMNXSLV'	'ps'	[1,1,429]	[Inf,Inf,1]	1	[]	[]
430	'http://goldsmr2.sci.gsfc.nasa.gov:80/dods/MATMNXSLV'	'ps'	[1,1,430]	[Inf,Inf,1]	1	[]	[]
431	'http://goldsmr2.sci.gsfc.nasa.gov:80/dods/MATMNXSLV'	'ps'	[1,1,431]	[Inf,Inf,1]	1	[]	[]
432	'http://goldsmr2.sci.gsfc.nasa.gov:80/dods/MATMNXSLV'	'ps'	[1,1,432]	[Inf,Inf,1]	1	[]	[]
433	'dap'	[1979,2014]	<1x1 struct>	'Monthly'	[]	[]	[]

The link type is '*dap*', indicating that the entries in Column 1 are OPeNDAP URLs. The OPeNDAP variable name is '*ps*'. The start indices progress along the time (third) dimension with a count of 1 for each time frame. The Span, indicated in *Attributes*(end,2) is between 1979 and 2014. The *Defs* structure is stored in *Attributes*(end,3), and the Calendar (Monthly) is stored in *Attributes*(end,4).

It is apparent that the *Attributes* array contains many redundancies. Aside for Column 3, all array values are fixed. In addition, Columns 6 and 7 are empty. In order to reduce these redundancies, it is possible to compress the *Attributes* array using the function *GOAT_ALP_CompressAttributes*. Following the command

```
>> Attributes = GOAT_ALP_CompressAttributes(Attributes)
```

the now compressed *Attributes* looks like

	1	2	3	4	5
1	<1x2 cell>	<1x2 cell>	<432x2 cell>	<1x2 cell>	<1x2 cell>
2	'dap'	[1979,2014]	<1x1 struct>	'Monthly'	432

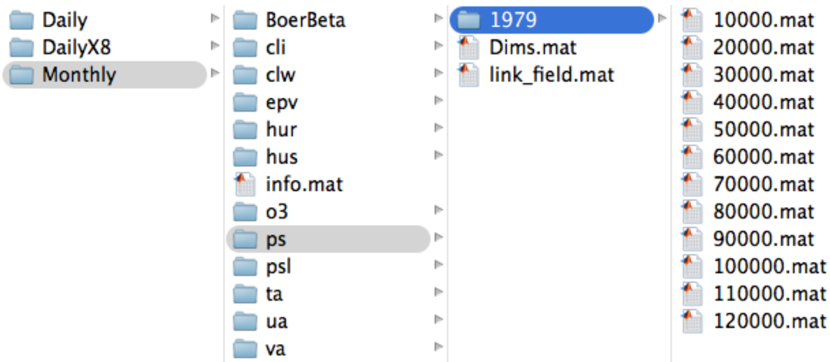
The redundant Columns 6 and 7 are removed. The first row of the *Attributes* array now contains cells with information on the unique entries, and the number of times these unique entries appear. The last row of the compressed *Attributes* is the same as that of the uncompressed *Attributes*, with the exception that Column 5 now contains the number of data frames in the array. This entry is used by GOAT to distinguish between compressed and uncompressed *Attributes* arrays (empty for uncompressed and some scalar for compressed). The compressed *Attributes* array is decompressed using the following syntax

```
>> Attributes = GOAT_ALP_DecompressAttributes(Attributes)
```

both compressed and decompressed *Attributes* can be stored in the *link_field.mat* file. GOAT will automatically identify and decompress compressed *Attributes* arrays.

2.6 Merged LINK and MAT archiving format

It is possible to merge the LINK and MAT archiving formats. In case of an overlap of MAT and LINK frames, MAT frames are used. The example below shows a merged LINK and MAT archive. For 1979, MAT frames will be used instead of the LINKED frames.



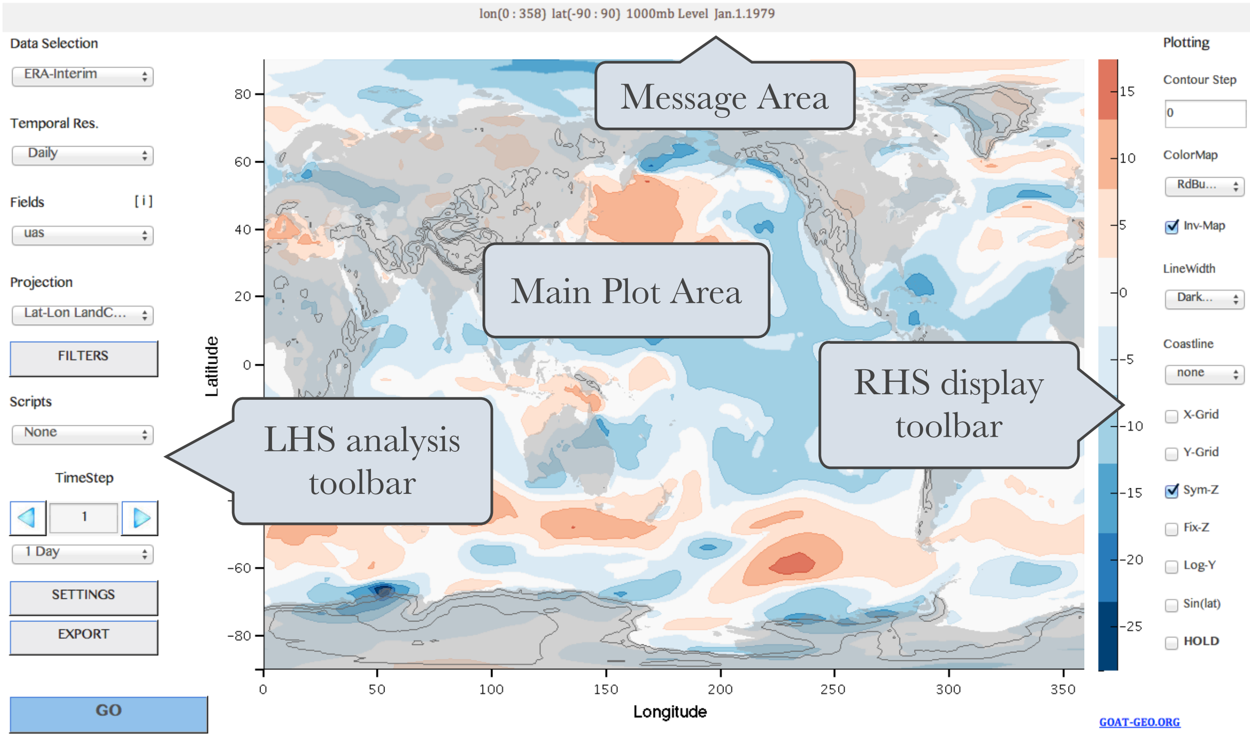
2.7 Boer Beta Function

In 'Isobaric', 'Isentropic' and 'Depth' vertical coordinates, spurious below-surface data is often represented as Nan or interpolated values. For cases where below surface is interpolated (such as for example, in the ERA-Interim reanalysis), GOAT includes a built in 'BoerBeta' function (Boer, 1982) defined as ones above surface and zeros below surface (pressure greater than surface pressure in 'Isobaric' coordinates, depth greater than sea bottom in 'Depth' coordinates and and potential temperature smaller than surface temperature in 'Isentropic' coordinates). The 'BoerBeta' function is automatically used in vertical integrals to exclude spurious contributions.

For linked BoerBeta functions, a descriptor at *Attributes*(end,1) defines how the function is derived:

'*bds*': data loaded from OPeNDAP source. BoerBeta derived from surface pressure.
'*bns*': data loaded from *.nc file. BoerBeta derived from surface pressure.
'*dds*': data loaded from a downloaded *.nc file. BoerBeta derived from surface pressure.
'*bdn*': data loaded from OPeNDAP source. BoerBeta derived from Nan values.
'*bnn*': data loaded from *.nc file. BoerBeta derived from Nan values.
'*dnn*': data loaded from downloaded *.nc file. BoerBeta derived from Nan values.
'*ers*': data loaded from a *.nc file downloaded from the ECMWF MARS server. BoerBeta derived from surface pressure.
'*ern*': data loaded from a *.nc file downloaded from the ECMWF MARS server. BoerBeta derived from Nan values.

3.1 Basic GUI operations



The centre of the GUI displays the main plotting area. A shaded message bar, common to most GOAT GUIs is located at the top part of the GUI. Analysis and display GUI operations are performed via the toolbars to the left and right of the main plotting area. Hovering above GUI elements activates tooltips with dynamic descriptions of each feature.

Keypress

Some GUI features can be activated by pressing keyboard keys. For example, when focus is on the main GUI figure, pressing '*g*' on the keyboard is equivalent to pressing the GO button. The keypress key is also described in the tooltips. When focus is on a particular feature (for example, the Data Selection dropdown menu), the keypress functionality of that particular feature takes over. To place focus on the main GUI figure press the 'control' key (applicable for all operating systems). The key press functionality can be disabled in the *GOAT_MyDefaults.m* file by setting 'G.Settings.KeyPressFlag = 0' ;

3.2 RHS toolbar

The RHS toolbar mostly controls display properties. The default contouring function is *contourf* (contour surfaces). In the descriptions below, the default value is denoted by {}.

Contour Step [{0 = system default} | any value requiring less than 100 contours]
It is possible to manually set the contour step in the '*Contour Step*' edit box. Setting 0 in the contour step edit box enables MATLAB's default contouring step selecting scheme. Values requiring more than 100 contours are not allowed and will revert to the default contouring step selecting scheme.

Color Map
Above 300 colormaps are available via the ColorMaps dropdown menu. The maps are derived from various sources, including popular tools such as cbrewer (see References). Some fields are automatically assigned default customized colormaps (set by the function *GOAT_FieldSettings*).

Inv-Map
A flag for inverting the colormap.

LineWidth [{Darken edgecolor} | None | 0.5:0.5:10]
By default, the edges of the contour surfaces are darkened using the function *GOAT_darken_edge* (applicable to any contourf map). It is also possible to select no edge lines (None) or set the with of the contour lines from 0.5 to 10 (in which case they are shown in black).

Coast LineWidth [{none} | 0.5:0.5:10]
Sets the coast line width in the latitude-longitude and polar stereographic projections.

X/Y -Grid
Show X or Y grid lines as grey (rgb =[.5 .5 .5]) solid lines.

Sym-Z
Make the colormap (or Y-limit for one-dimensional plots) symmetric with respect to zero. This flag assigns the brightest colour to zero values in diverging maps. In one-dimensional plots it sets the Y-limit symmetric with respect to zero.

Fix-Z
Fix the colormap limits (caxis property) or (Y-limit for one-dimensional plots). A buffer of 10% is added to the colormap limits (or Y-limits). This addition can be adjusted in the *Settings GUI* (see below) at the *Fix-Z Buffer* editbox.

Log-Y
Use logarithmic scale for vertical axis.

Sin(Lat)
Use sin(latitude) in plots involving latitude.

HOLD

When this flag is on, the current plot is fixed and consecutive plots are overlaid on the existing plot as grey contour lines. This can be done multiple times, where each new plot replaces the top plot without affecting the fixed plot. Fixed and overlaid plots need have the same projection but can consist of different fields with different spatial dimensions.

HOLD-DIFF

Made visible only when HOLD is checked. When this flag is on, the difference between current analysis and Held data is shown. If the spatial dimensions of the current analysis do not match those of the Held data, the current analysis is interpolated to the spatial dimensions of the Held data. Care should be taken when this functionality is applied to Hovmöller plots.

3.3 LHS toolbar

Data Selection

by default, GOAT looks for dataset folders in GOAT/Data. A dataset folder must be in a standard GOAT archiving format (see Sections 2.2 and 2.3):
DataSet/Calendar/Field/Year/files.mat (e.g., *NCEPI/Daily/pr/2000/10100.mat*).
It is possible to set multiple main dataset folders (in addition to or instead of *GOAT/Data*) in *GOAT_MyDefaults.m* by editing 'G.DataFoldersAddress', and adding a cell array of main dataset folder strings (i.e., the folders containing the dataset folder). When this variable is commented, GOAT will revert to *GOAT/Data* as the default main Data folder. It is also possible to browse to a certain dataset folder by pressing the 'browse' option in the *Data Selection* dropdown menu. If this option is used, the Tooltip of this feature will show the full address of the dataset.

Temporal Resolution

The *Temporal Res.* dropdown menu displays the available time resolutions in the dataset folder. In addition, when the time resolutions DailyX2, DailyX4 or DailyX8 exist, the *Temporal Res.* menu also adds a 'virtual' time resolution DailyX1 which is a subset of any of the above. This relieves the need to save redundant subsets of DailyX1 fields. Note that the Daily temporal resolution refers to daily averages while DailyX1 refers to one time daily data.

Fields

The fields dropdown menu consists of the fields listed under *DataSet/TemporalResolution* and fields that can be derived from the existing fields, termed as *'Virtual Fields'* in GOAT terminology. Hovering over the *Fields* pane will cause a Tooltip with a description and units of the fields to appear (taken from *Dims.mat*). Some virtual fields are built into GOAT. These are *'ta_pot'* (atmospheric potential temperature), *'Sp'* (atmospheric static stability), *'N^2'* (Brunt Väisälä frequency) and *'Psi'* (atmospheric meridional mass stream function).

Virtual Fields

The built-in virtual fields are editable at *GOAT_VirtualFields.m.* An unlimited number of virtual fields can be added to these fields using customized virtual field functions. These functions are located at:
GOAT/MyScripts/VirtualFields/GOAT_VirtualField_FieldName.m
A wide range of not-activated virtual fields is provided in GOAT/MyScripts/VirtualFields, which can be used as templates. The virtual fields function format is simple, allowing adding, removing and editing virtual fields with little effort. See further documentation in section 5.3 and within virtual field functions.

Info [i]

Displays the string contained in the *info.mat* file, if it exists.

Projection

GOAT provides 9 selectable map projection. See section 3.7 for additional information of the preferences of each projection. The projections are:
Lat-Lon LandCover: latitude-longitude cylindrical projection showing shaded land areas. A coastline can be added, using the RHS Coast LineWidth pane (set to 'none' by default).
Lat-Lon: latitude-longitude cylindrical projection (same as above with no shaded land areas).
Lat-Level: Latitude-Level (Level = Pressure, Height, etc) projection.
Lon-Level: Longitude-Level (Level = Pressure, Height, etc) projection.
Hovmöller-Lon: longitude Hovmöller plot.
Hovmöller-Lat: latitude Hovmöller plot.
Hovmöller-Level: Level (Pressure, Height, etc.) Hovmöller plot.
NH Polar Stereographic*: Polar Stereographic projection of the NH with coastlines
SH Polar Stereographic*: Polar Stereographic projection of the SH with coastlines
*The Polar Stereographic projections may not work or appear differently on some systems due to known bugs in *contourfm* function.
An additional (10th) Projection option is shown for LINK archives, shown as
install linked data as *.mat: When chosen, instead performing an analysis, the selected data will be installed in MAT format (see more in section 4).

Scripts

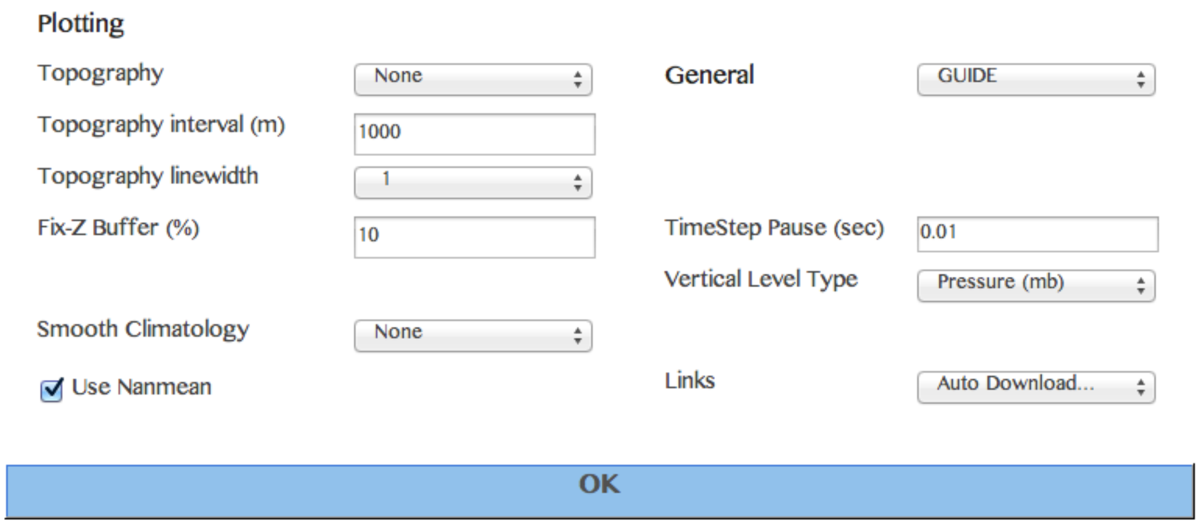
It is possible to add unlimited functionalities to GOAT by simply placing m-files in the *GOAT/Scripts* folder. This feature acts as a built-in 'add-on' functionality. m-files (but not folders) placed in the *GOAT/Scripts* folder are automatically recognized by GOAT and added to the *Scripts* menu. Header text of the m-files will appear as tooltips. GOAT scripts can consist of free scripts or functions in the format G = function_name(G) (G is the main GOAT structure containing all variables and figure handles). It is recommended but not obligatory to name the scripts '*GOAT_Script_ScriptName.m'.* Several scripts are included in the default GOAT version. To run a script, select it in the *Scripts* menu and press 'GO'. The default scripts are:

- *cbrewer_colormaps_legend.m*: display the cbrewer colormap legend.
- *GOAT_Script_Add_Subtruct_Fields.m*: A tool for adding or subtracting up to 5 fields with consistent spatial dimensions.
- *GOAT_Script_Close_All_Figures_But_GUI.m*: Closes all open figures aside for the GUI.
- *GOAT_Script_LoadExport.m*: Load exported data (sect. 3.6). Select the exported *.mat file and GOAT will display the exported data.
- *GOAT_Script_IndexComposite.m*: Produce composites from indices (learn about Indices in FILTERS, sect. 3.5). For a given Index, this script outputs composites of the Index, the climatology (i.e. no index) and a significance plot of the Index composite.
- *GOAT_Script_MIP_LINK_DRS_1_3_1.m*: A tool for linking data in CMIP5 archiving standard. See more at Section 6.XXX.

TimeStep

TimeStep enables repeating actions an editable (*TimeStep* edit box) number of times. This feature can be used to browse quickly through data or produce simple animations in time (by using a large number of time steps). It is possible to add a minimal pause between each *TimeStep* display, in the SETTINGS '*TimeStep pause*' pane (thereby controlling the speed of the animation). Pressing the right or left arrows (keypress = right and left arrow keys) is equivalent to pressing the GO button and repeating the latest analysis one time increment forward or backwards in time, respectively. The increment can be set in a dropdown menu which adjusts according to the Calendar.

3.4 SETTINGS GUI



Topography [None | Atmosphere | Ocean | All]
Using MATLAB's 'topo', it is possible to display above sea level topography ('Atmosphere'), below sea level topography ('Ocean') or bellow and above sea level ('All'). Some fields are automatically assigned customized topography display settings (e.g., for precipitation rate ['*pr*'] - above sea level topography is shown).

Topography interval (m)
It is possible to set the topography contour interval in units of m. The default value is 1000m.

Topography linewidth [{1}]
Sets topography contour linewidth.

Fix-Z Buffer [{10%}]
When the *Fix-Z* option is on, an editable buffer (10% by default) of the absolute maximal value of the presented data ('*nanmax(abs(G.Data(:)))*')is added to the colormap axis. This is done to allow consecutive plots to be captured within the current colormap limits.

Smooth Climatology [{None} | 1:2:31 for Daily | 1:2:11 for Monthly]
It is possible to smooth 'Daily', or 'Monthly' climatology by applying a centered running-average of n = 1,3,5,7, ... points.

Use NanMean
Apply the function 'nanmean' instead of 'mean' (i.e. ignore Nan values) to averaged data in space or time. This Flag is automatically set to 1 for some datasets known to include Nan values.

***General* (dropdown menu)**

GUIDE

launch the GOAT GUIDE, also located in *GOAT/Documents/GOAT_GUIDE.pdf*.

Update GOAT

Update to the latest GOAT version. This will close the current GOAT session.

Edit Constants

Edit or view the physical constants: Earth's radius, gravitational constant, rotational frequency, specific heat at constant pressure, moist specific heat at constant pressure, latent heat of vaporization, ideal dry gas constant, ideal vapor gas constant. The editable values are not saved after GOAT is closed. In order to permanently edit these values, edit the variables under *G.Consts* in *GOAT_Defaults.m*. In order to prevent edited values for being overwritten during version updates, add the appropriate variables to *GOAT_MyDefaults.m*.

TimeStep Pause (sec)

Assign a minimal delay (seconds) between consecutive plots in *TimeStep* mode.

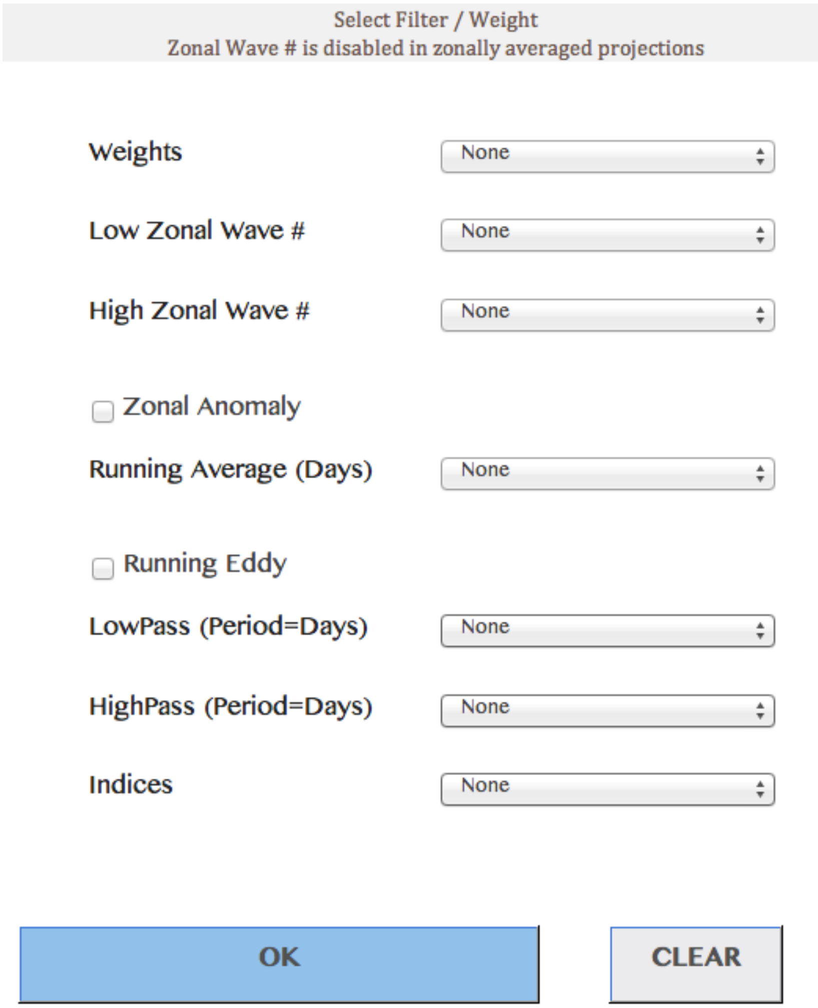
Vertical Level Type

Assign vertical level type (*'Pressure (mb)'*, *'Height (m)'*, etc.).

Links

Links to the GOAT homepage, DataDownload Tool, and other useful sites. It is possible to add, remove, or edit links in *GOAT_MyDefaults.m* by editing the fields *G.Settings.webLinks* and *G.Settings.webLinksTitle*.

3.5 FILTERS GUI



When any of the FILTERS options is other than 'None', the FILTERS button changes color to green. The FILTERS options are reset (and the color of the FILTERS button set back to grey) when a new Calendar or new dataset are selected and can also be reset by pressing the CLEAR button.

Weights

A list of preset weight functions can be added to any field. These are a scalar Factor, and the functions $\cos(\phi)$, $-\cos(\phi)$, $\left(a\cos(\phi)\right)^{-1}$, and $-\left(a\cos(\phi)\right)^{-1}$. The factor menu accepts algebraic expressions.

Low Zonal Wave # / High Zonal Wave #

Zonally filter fields to show wave numbers between Low and High Wave #. The filtering is done using simple fft transformation to spectral space.

Zonal Anomaly

Show the difference from the zonal mean.

Running Average (Calendar)

Perform a centered running average of the data in time, where the number of points corresponds to the current Calendar (Months, Days, etc.)

Running Eddy

Show difference from running average.

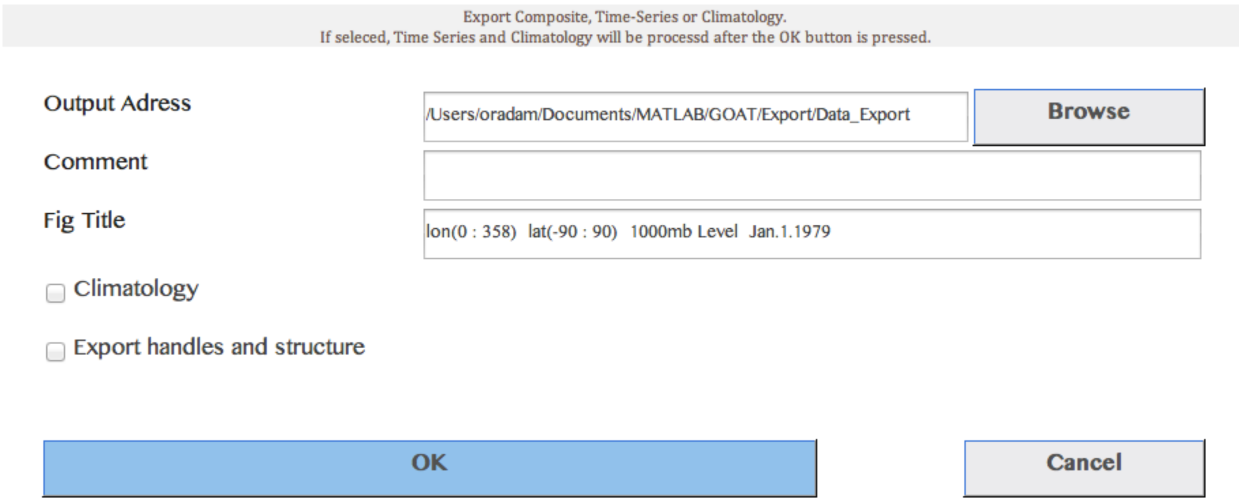
LowPass / HighPass (Period = Calendar)

Perform Low and high pass filtering of the data in time. The filtering is done by simple transformation of the data series to spectral space.

Indices

Indices are logical vectors (or vectors with values 0 or 1) of the same length as G.Addresses.Default (the current Calendars cell array of address strings). Each Calendar folder includes an ('*Indices*') folder (not shown in the list of fields), reserved for indices particular for the dataset and Calendar. The Indices menu lists the indices present in this folder and also allows to browse for an index places elsewhere.

3.6 EXPORT GUI



The latest analysis is exported to a folder containing a *.fig figure file, a pdf figure file and a structure D of the exported data.

Output Address

By default, analysis is exported to the folder '*…GOAT/Export/Data_Export*'. This address can be edited using the edit box or by browsing to an existing folder.

Comment (optional)

Add a descriptive comment to the exported data.

Fig Title

Edit (or delete) the exported figure title (pdf and fig files). By default, the exported figure is given the title appearing in the GUI message are.

Climatology

When checked, the climatology ('*Monthly*' or '*Daily*') of the analysis time series is also calculated and exported. The climatology is not smoothed by default. It is possible to smooth the climatology using a centered running mean by setting the variable '*G.Limits.Default.DataClimSmooth*' at the SETTINGS menu ('*Smooth Climatology*' panel) or at *GOAT_Defaults.m*.

Export handles and structure

The figure handles and main GOAT structure 'G' can be exported in addition to the latest analysis.

Description of exported data

The latest analysis is exported to a structure named 'D' containing the following fields:

D.Comment: Comment entered at the comment edit box

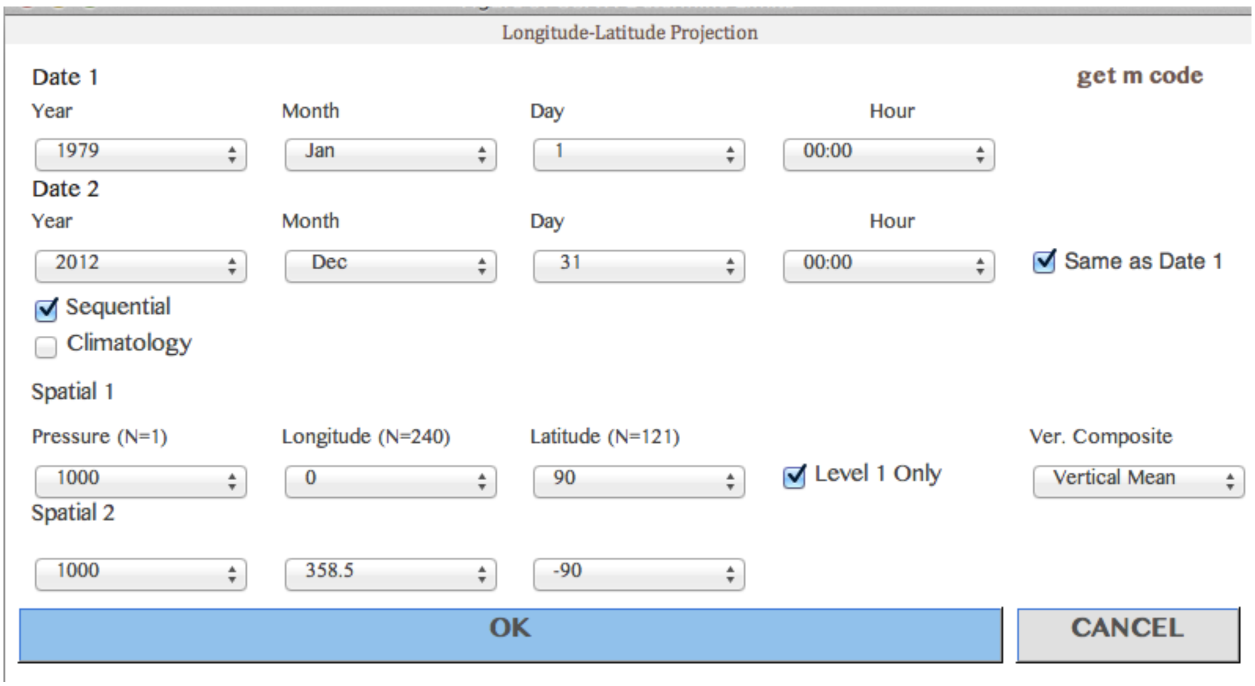
D.data = analysis data

D.x , D.y: horizontal and vertical axis coordinates
D.X , D.Y: meshed D.x and D.y
D.xlabel: horizontal axis label
D.ylabel: vertical axis label
D.datalabel: field name
D.Dates: the dates used to calculate time series and composite;
D.FilteredDates: if an index was used, FilteredDates are the dates for which Index=1
D.Filter: Filter structure with properties of filters used
D.infoMessage: the message appearing at the GUI message area;
D.ProjectionType: projection index (1=Latlon LandCover, 2=Lat-Lon, 3=Lat-Level, etc..)
D.Limits.Default.sameP: Level Flag (0/1) (see section 4.7)
D.Limits.Default.VerMeanInt: vertical averaging index (see section 4.7)
D.lon: longitude un-truncated dimension of the analyzed (as it appears in Dims.mat)
D.lat: latitude un-truncated dimension of the analyzed (as it appears in Dims.mat)
D.P: vertical un-truncated dimension of the analyzed (as it appears in Dims.mat)
D.lonlabel: x-dimension label
D.latlabel: y-dimension label
D.Plabel: vertical dimension label
D.Units: field units
D.Description: field description
D.Limits.Default.Lat1: first index of truncated latitude
D.Limits.Default.Lat2: second index of truncated latitude
D.Limits.Default.Lon1: first index of truncated longitude
D.Limits.Default.Lon2: second index of truncated longitude
D.Limits.Default.P1: first index of truncated vertical dimension
D.Limits.Default.P2: second index of truncated vertical dimension
D.DataClim: Climatology data
D.DataSeries: time series of data. If time the time series output size exceeds 1GB it is broken into files DataSeries_1,2,3… with accompanying Dates_1,2,3… files.
D.DataSeriesAnom: the time series anomaly relative to the climatology DataClim. As in the case of DataSeries, if the output size of DataSeriesAnom exceeds 1GB it is broken into consecutive files DataSeriesAnom_1,2,3…
D.Hold: Held data and held data properties

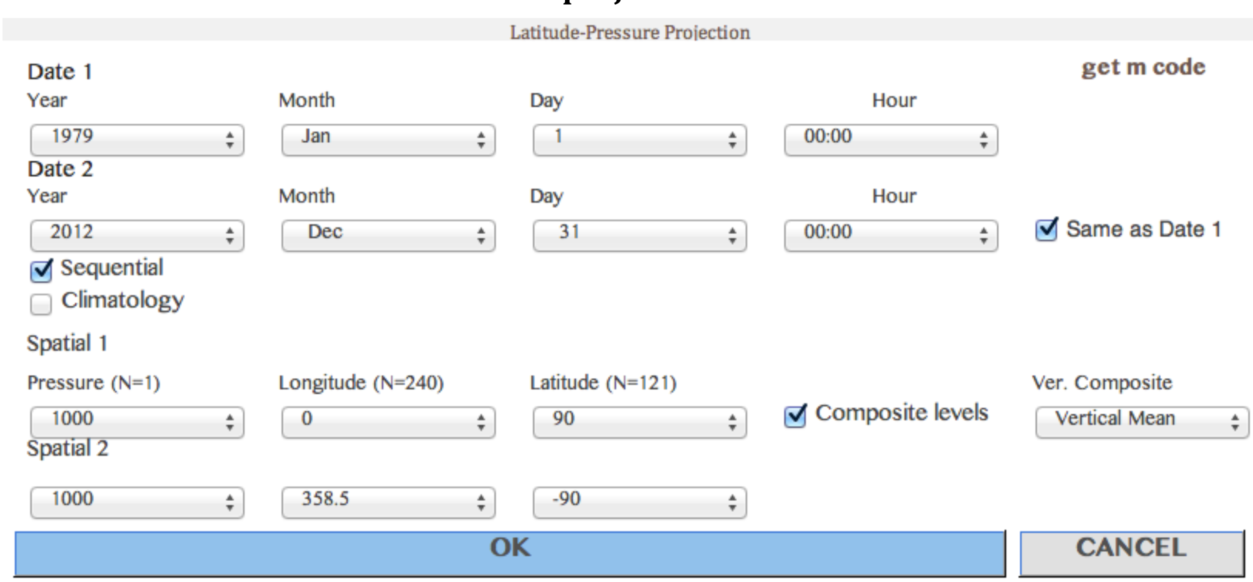
3.7 GO and the LIMITS GUI

Temporal and spatial subsets of the selected field are set at the upper and lower parts of the LIMITS GUI, respectively. The message area at the top of the GUI gives a description of the projection and in some cases, warnings when the spatial temporal subset conflict with the FILTER properties. The spatial and temporal limits present the dimensions and available data particular to the chosen field. The available flags (checkboxes), temporal and spatial panels differ for each projection, Calendar and vertical coordinate. Examples of the LIMITS GUI in each projection are shown below:

LIMITS GUI for lat-lon and polar stereographic projections



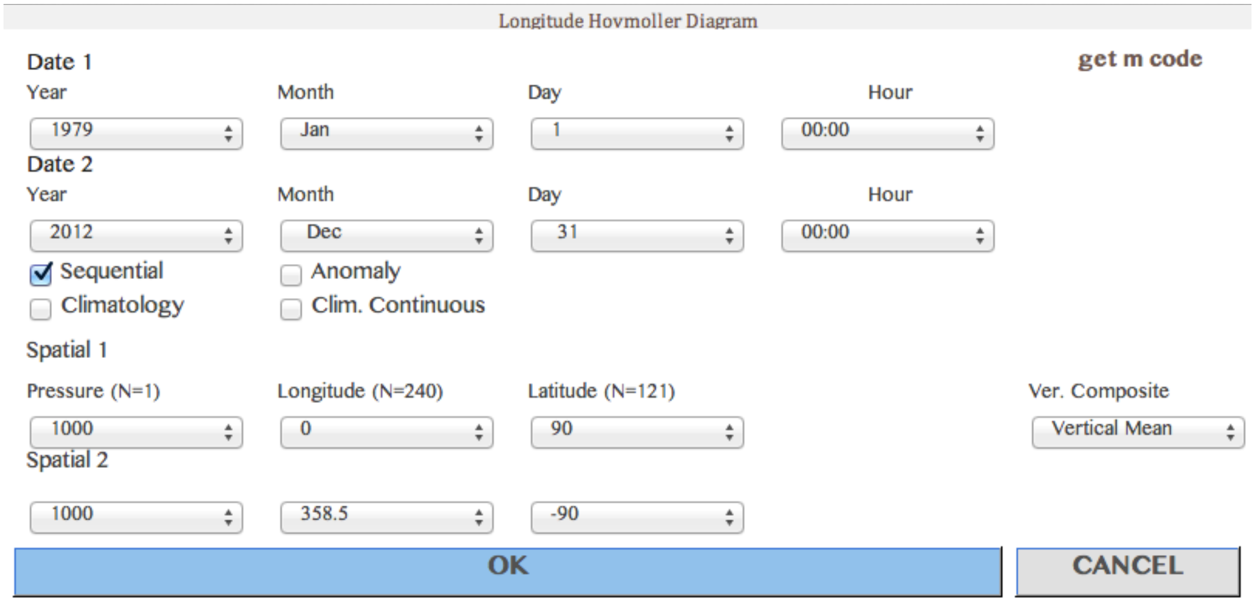
LIMITS GUI for lat-Level and lon-Level projections



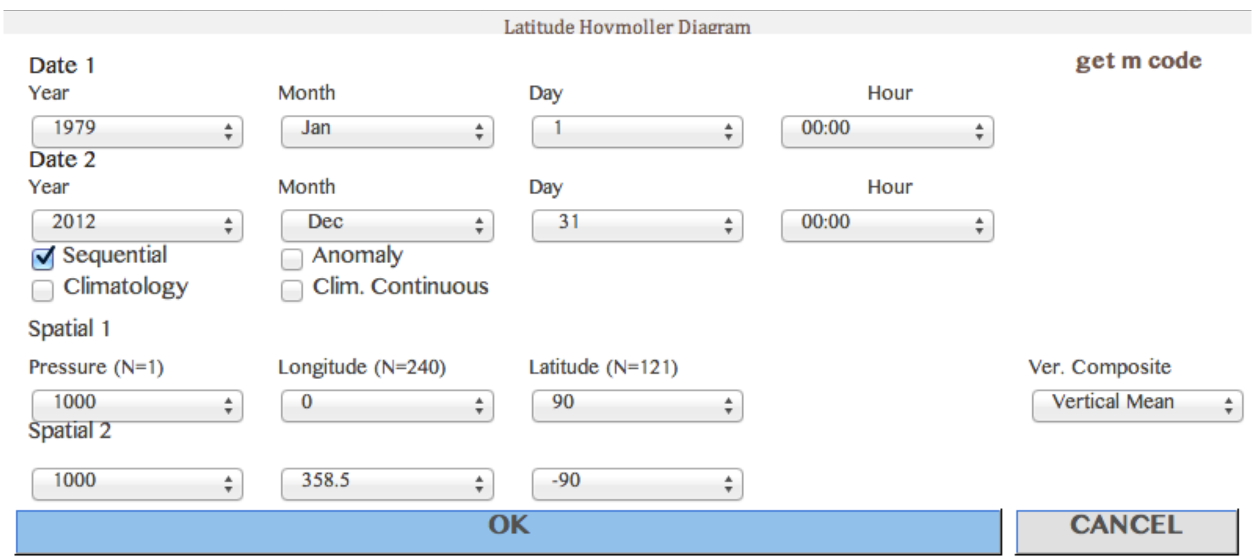
Hovmöller LIMITS GUIs differ from other projections in the following properties:

- No '*Same_as_Date_1'* checkbox
- An additional '*Anomaly*' checkbox
- An additional '*Clim. Continuous*' checkbox

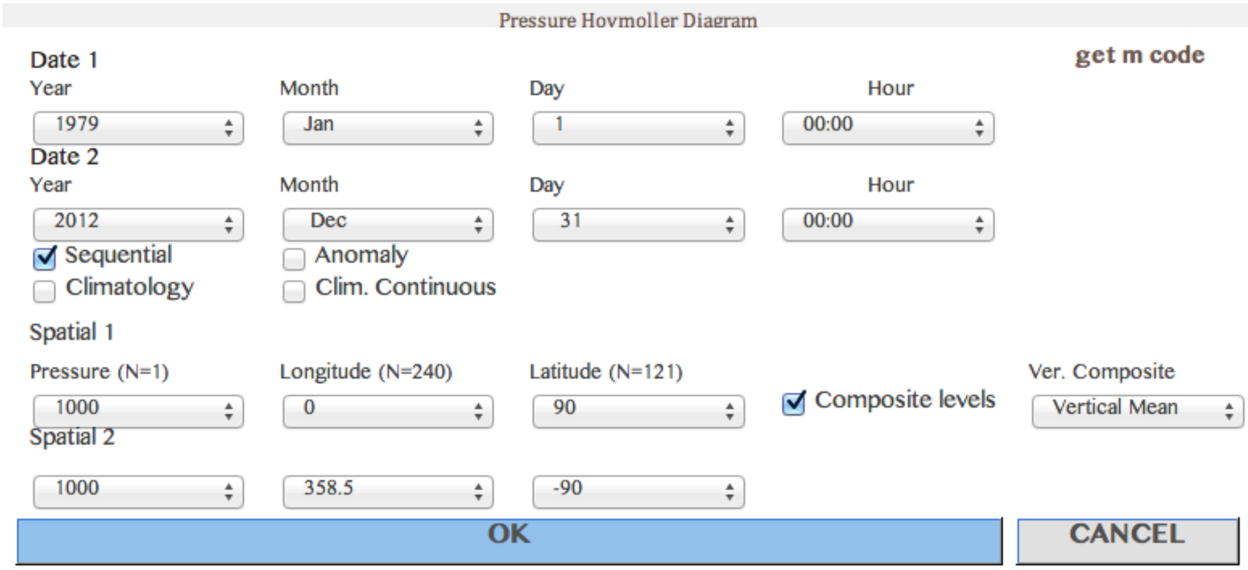
LIMITS GUI for lon-Hovmöller projection



LIMITS GUI for lat-Hovmöller projection



LIMITS GUI for Level-Hovmöller projection



Temporal Subsets

Date 1 and Date 2 (lists)
Select beginning and end dates of data subset. By default, the Date 1 and date 2 show the first and last available dates for the selected field.

Same as Date 1 (checkbox)
This flag sets Date 2 to be equal to Date 1, thereby allowing a fast way of creating a single date subset. This flag is not available for Hovmöller plots.

Sequential / Climatology (checkbox)
Subsets can be either in sequential or climatology mode. Consider the following case:
Date 1: Year = 1979, Month = Jun, Day = 1, Hour = 00:00
Date 2: Year = 1989, Month = Aug, Day = 31, Hour = 00 ;
Same_as_Date_1 = off.
In sequential mode, the temporal subset will be composed of all dates between Jun.1.1979 at 00:00 hours and Aug.31.1989 at 00:00 hours.
In Climatology mode, the temporal subset will include only the dates between June.1 and Aug.31 for the selected years (i.e., June,July,August for the years 1979:1989).
If Same_as_Date_1 is 'on' in Climatology mode, the temporal subset will be composed of the month and day of Date 1 for the selected year range (year of Date 1 and Year of Date 2)

Anomaly (checkbox, only for Hovmöller plots)
When checked, a time series of the anomaly (from climatology) is shown.

Clim. Continuous (checkbox, only for Hovmöller plots)
When checked, it shows climatology in a continuous manner. For example, in the example given above (*Sequential / Climatology*), when checked in Climatology mode, a time series of JJAJJAJJA… for the years 1979:1989 will be shown.

Spatial Subsets

Spatial 1 / Spatial 2
Select spatial subset

Spatial Flag
The spatial flag varies for different projections.
Level 1 Only (Lat-Lon and Polar Stereographic projections): when checked, the subset will include only Level 1. When unchecked, the subset will be composed of a vertical composite of the levels between Level 1 and Level 2. The type of vertical composite is set in the Ver. Composite dropdown menu.
Composite Levels (Lat-Level, Lon-Level and Level-Hovmöller plots): when unchecked, the subset includes the specified levels between Level 1 and Level 2. When checked, the subset will include a vertical composite (set by *Ver. Composite*) of the specified levels between Level 1 and Level 2.

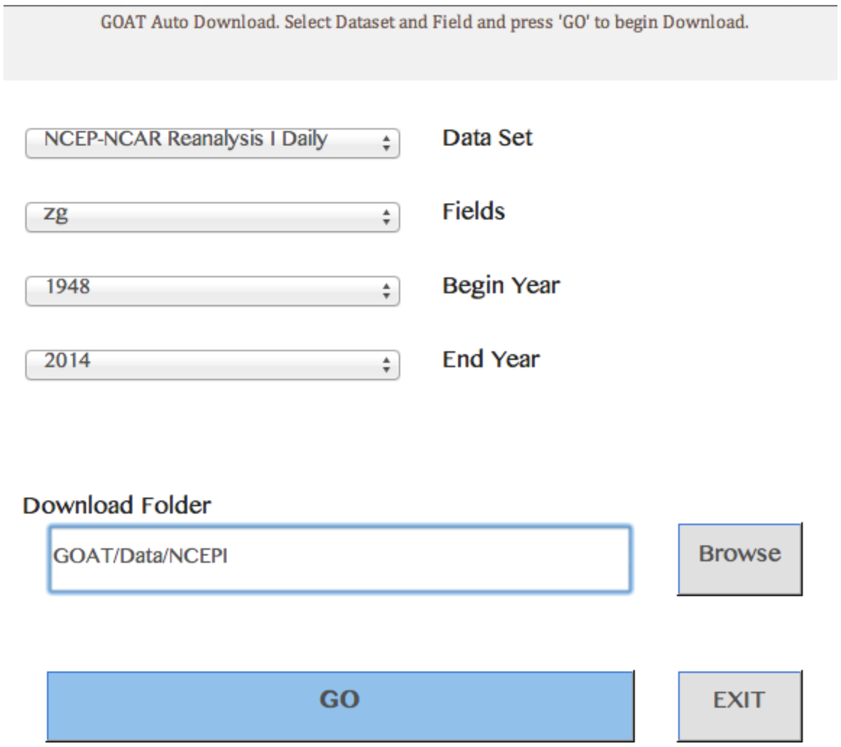
Ver. Composite
Vertical Mean: calculate simple (level weighted) vertical mean (or nanmean if nanmean flag is on) between of the selected levels.
Mass weighted Vertical Integral Boer: Calculate mass weighted (*1/g* for isobaric and sigma coordinates, water density for oceanic '*Depth (m)*' vertical levels and 1 for other vertical level types) vertical integral using the Boer β function (G.J.Boer, 1982: Diagnostic Equations in Isobaric Coordinates. *Monthly Weather Review*, **110**, 1801-1820). The Boer β matrix (defined as 1 when pressure < surface-pressure and 0 when pressure > surface-pressure) is automatically calculated and saved (but not shown in the fields list) when the geopotential height field is downloaded in any of the Auto Data Download codes.
Mass weighted Vertical Integral Nan: perform a vertical integral while setting Nan values to 0.

get m code
When pressed, the code required to produce the calculation activated by pressing 'OK' is printed onto the command window. This code serves as a good starting point for any code in the GOAT coding environment. This code, generated by *GOAT/Code/Utils/GOAT2Code_Limits.m* is described in detail in sect. 5. Copying and pasting this code to a new m file and running the code, will produce the same calculation as pressing the OK button.

4.1 Auto Data-Download tool

Linking may not be possible or preferable for some datasets. It is possible to download and install datasets in MAT archiving format using the automated data-download utility *GOAT_AutoDownload.m*. This utility can be launched by either pressing the *DataDownload* shortcut or by typing *GOAT_AutoDownload* at the command window.

A screen shot of the AutoDownload tool GUI is shown below



From top to bottom, the GUI elements are:
message area: as in the main GOAT GUI, it provides instructions in some cases and describes the progress of the download process.
DataSet: a dropdown menu listing datasets available for download.
Fields: A list of the fields available for download. An option to download all fields is also added at the end of the list.
Begin Year/End Year: First and last years of data to be downloaded.
Download Folder: the main data folder. The default folder is *GOAT/Data/DatasetName*. The Download Folder can be edited at the edit box or browsed by the pressing the Browse button.
GO: begin download.
EXIT: close GUI and Exit.

4.2 Auto Download Protocol (ADP)

The utility *GOAT_AutoDownload.m* reads scripts located at
…GOAT/Utils/DataManagementCode/AutoDownload/Datasets/GOAT_AutoDownload_(Dataset).m

These scripts are written in a generic format (herein referred to as the Auto Download Protocol - ADP), allowing users to write customized download scripts.
The fields and functions listed below need to exist in an *AutoDownload script*. These are divided into three 'modes': 'Meta', 'Field' and 'Download'. The following fields need to exist in the 'Meta' mode:

D.AutoDownloadDataset.Active: 0 or 1. When 0 the dada set will not be available via the DataSet drop down menu.

D.AutoDownloadDataset.Name: Dataset name as it appears in the Dataset dropdown menu.

D.AutoDownloadDataset.FolderName: a unique dataset name, used as the default Download Folder name (*NCEPI* in the above GUI example).

D.AutoDownloadDataset.DescriptionTip: A string which appears as the tooltip string when hovering above the Data Set dropdown menu.

D.AutoDownloadDataset.CurrentYear: when 1, the current year is set as the End Year (enables downloading the most recent data). When 0, the End Year is set independently for each field.

D.AutoDownloadDataset.TimeSelect: when 1, the GUI enables selecting the Begin and End years. When 0, the Begin and End Year dropdown menus are disabled. This feature is used for datasets that need to be downloaded in one batch.

D.AutoDownloadDataset.Fields.Names: A cell string array of field names appearing in the Fields dropdown menu.

Each fieldname appearing in *D.AutoDownloadDataset.Fields.Names* must be assigned the following attributes in the 'Field' mode:

D.AutoDownloadDataset.VerticalOrdinate: one of 5 possible vertical ordinates supported by GOAT, '*Pressure (mb)*' | '*Depth (m)*' | '*Height (m)*' | '*Sigma*' | '*Potential Temperature (K)*'

D.AutoDownloadDataset.LevelType (Optional): information regarding the vertical ordinate of the field

D.AutoDownloadDataset.FirstYear: first year of data available for download

D.AutoDownloadDataset.LastYear: last year of data available for download (replaced by current year when *CurrentYear* Flag is 1)

D.AutoDownloadDataset.TimeResolution: Calendar

D.AutoDownloadDataset.FieldDescription: Appears as a tooltip string when hovering above the Fields drop down menu. Used as the Field Description in the *Dims.mat* file.

D.AutoDownloadDataset.Units: Field Units.

D.AutoDownloadDataset.url (optional): Reference url address

A generic download code is to be placed in the 'Download' mode. Several examples of AutoDownload scripts already exist in GOAT. These can be used as templates and examples. In particular, *GOAT_AutoDownload_NCEPI.m* contains code for downloading data from two different sources and serves as a good example code. Look for further code documentation within the script.

5. GOAT coding environment

GOAT functions and scripts are organized by underline folders. This makes it easy to browse through GOAT functions. For example, by typing
>> GOAT_Script_
at the command window and pressing the tab key, all GOAT Scripts become visible via MATLAB's auto completion utility. In addition, all functions have help texts.

All GOAT variables and settings are centralizes in the main GOAT structure, G. This simplifies calling functions by enabling the syntax *G = function_name(G)* as well as enabling a search within the structure via MATLABs auto completion utility. However, the complexity of G makes is difficult to browse through, if one is not sure exactly what to look for. The function GOAT_INFO_LIMITS displays the current spatial and temporal settings and provides a list of the relevant variables with some descriptions and tips. Similarly, the function GOAT_INFO_PLOT displays the current plot settings and provides guidance and tips for using the GOAT plotting function GOAT_plotUpdate.

The examples below provide a basic guide of the GOAT coding environment.

5.1 *get m code*

The *get m code* button is located at the top right corner of the LIMITS GUI (see section 3.7). Pressing the '*get m code*' calls the function *GOAT/Code/Utils/GOAT2Code_Limits.m* which produces the code for performing the analysis which would commence if the *GO* button is pressed. The code is printed onto the command window. The printed code begins and ends with the statements:

```
% - - - code generated by GOAT2Code_Limits.m – TimeStamp - - - - - -
  •

  •

% - - - code end – TimeStamp - - - - - - -
```

Close GOAT, copy the text between these statements, and paste to a new m-file.

Example 1: Retrieving data

The example described below retrieves the zonal surface wind ('*uas*') from the NCEP-I DataSet between the dates Jan.1.1948 and March.31.1948, and plots it in the *Lat-Lon LandCover* projection.

The code is written in MATLAB's 'cell' code-sections. If you are not familiar with code-sections, it is worth spending a few minutes reading about it (*http://www.mathworks.ch/ch/help/matlab/matlab_prog/run-sections-of-programs.html*).

The initial section describes the structure of the code. It also mentions that it is recommended but not obligatory to work from the …*GOAT/Code* directory. If the paths to the folders
…GOAT/Code

...GOAT/Scripts
...GOAT/MyScripts
and all their subfolders are included in the MATLAB path (set upon a successful installation of GOAT), there should be no problem working from any directory.

```
% It is recommended though not mandatory to work from the ... GOAT/Code
  directory
% When not working from the GOAT/Code directory make sure that the following
  folders
% and all their subfolders are added to the matlab path:
% ... GOAT/Code,   GOAT/Scripts,   and GOAT/MyScripts
% cell-1: Initiate GOAT and set: Dataset, Field and Calendar
% cell-2: Set temporal limits
% cell-3: Set spatial limits and Filters/weights
% cell-4: Retrieve and compile data
% cell-5: plot
% cell-6: save
```

In the first cell, the GOAT environment is initiated and DataSet, Calendar, and Field are set.

```
%% cell-1: Initiate GOAT and set: Dataset, Field and Calendar
G = GOAT_Environment('enter_project_path') ; % Initiate GOAT Environment
```

All GOAT codes begin with the function *GOAT_Environment*, which sets up the main GOAT structure, G. It is possible to add a temporary project path. This path will not be saved after MATLAB is closed.

```
G = GOAT_Initialize_Dataset(G,'/GOAT/Data/NCEP-I','Daily') ; % Go through
  contents of Data folder and Calendar. Link existing fields, temporal span,
  etc.
```

The command
G = GOAT_Initialize_Dataset(G , 'DataSet_Address' , 'Calendar')
is equivalent to selecting a DataSet and Calendar at the GOAT GUI. This function surveys the DataSet's Calendars, real and virtual Fields, Field spans and spatial dimensions. The dataset and Calendar can be either in MAT or LINK archive formats.

```
G = GOAT_Initialize_Field(G,'uas')        ; % get spatial and temporal
  dimensions of field, display settings, topography, and file addresses. Link
  if necessary
```

The command
G = GOAT_InitializeField(G,'FieldName')
is equivalent to selecting a field in the Field GOAT GUI panel. The FieldName can refer to a real or virtual field. This function gathers the spatial and temporal dimensions of the field, as well as metadata properties (Units, FieldDescription, Field metadata).

```
Year0 = 1948 ; % first year of the field uas
```

The first year of the archived dataset is stated for reference. This value is automatically found and stored in *G.Limits.Year0.*

The temporal definitions of the data are set in cell-2.

```
%% cell-2: Temporal limits
% Temporal Checkboxes/Flags
G.Limits.Default.Day2same   = 0 ; % 1/0 Set Date 2 same as Date 1
```

Setting *G.Limits.Default.Day2same* to 1 or 0 is equivalent to checking or unchecking the 'Save as Date 1' checkbox in the GOAT LIMITS GUI.

```
G.Limits.Default.Sequential = 1 ; % 1/0 Sequential or Seasonal averaging Flag
```

Setting *G.Limits.Default.Sequential* to 1 or 0 is equivalent to checking the 'Sequential' or 'Climatology' checkboxes, respectively.

```
G.Limits.Default.ClimatologyContinuous = 0 ; % present seasonal data
% sequentially in Hovmöller plots
```

Setting *G.Limits.Default.ClimatologyContinuous* to 1 or 0 is equivalent to checking or unchecking the '*Clim. Continuous*' checkbox in the GOAT LIMITS GUI (relevant only for Hovmöller plots).

```
G.Limits.Default.Anom  = 0      ; % display anomalies in Hovmoller plots
```

Setting *G.Limits.Default.Anom* to 1 or 0 is equivalent to checking or unchecking the '*Anomaly*' checkbox in the GOAT LIMITS GUI (relevant only for Hovmöller plots).

Begin (Date 1) and end dates are set below

```
year1  = 1948 ; % yyyy
month1 = 1    ; % 1:12
day1   = 1    ; % 1:31
hour1  = 0    ; % 0:3:21
year2  = 1948 ;
month2 = 3    ;
day2   = 31   ;
hour2  = 0    ;
```

For performance optimization considerations, GOAT uses a predefined array of address strings spanning the Calendar timespan of the selected field. This predefined string of addresses is found in *G.Addresses.Default*. After defining the temporal span of the data to be retrieved, it is necessary to locate the indices of the relevant dates within *G.Addresses.Default*. The indices *G.Limits.S1* and *G.Limits.S2* are the indices of Date 1 and Date 2 within *G.Addresses.Default*.

```
% G.Limits.S1 and G.Limits.S1 are the temporal indices used by GOAT to locate
% files in the GOAT archiving standard
G.Limits.S1 =
GOAT_getDailySerial(year1,month1,day1,hour1,G.TimeFieldVirtual) ;
if G.Limits.Default.Day2same && G.Limits.Default.Sequential
    G.Limits.S2 = G.Limits.S1 ;
else
G.Limits.S2 =GOAT_getSerial(year2,month2,day2,hour2,G.TimeFieldVirtual,Year0);
end
```

The function GOAT_getSerial retrieves the index of the date specified by year, month, day, hour, for a given Calendar. The variable *G.TimeFieldVirtual* is equal to the Calendar entered in *GOAT_Initialize_Dataset*. The retrieved index is in reference to the first Calendar data of the year Year0.

🐐 The spatial definitions of the field are set in cell-3.

```
%% cell-3: Spatial limits
% Checkboxes/Flags
G.ProjectionType = 1 ; % 1) lat-lon with coast 2) lat-lon 3) lat-height
% 4)lon-height 5) hovmoller-lon 6) hovmoller-lat 7) hovmoller height
% 8/9) NH/SH polar stereographic
```

🐐 The index G.ProjectionType determines the projection, by order of appearance in the Projection panel in the GOAT GUI.

```
G.Limits.Default.sameP = 1      ; % Projection 1,2,8,9) 1/0 set level2 =
% level1 | Projection 3,4,7) 1/0 composite levels
```

🐐 Setting the *Limits.Default.sameP* flag to 1 or 0 is equivalent to checking or unchecking the levels flag in the GOAT LIMITS GUI.

```
G.Limits.Default.VerMeanInt = 1 ; % 1) Vertical Mean 2) Integral using Boer
% Beta 3) set Nan values to zero and vertically integrate
```

🐐 The index *G.Limits.Default.VerMeanInt* determines the type of vertical averaging/ integration, by order of appearance in the *Ver. Composite* dropdown menu.

```
G.Settings.NanMeanFlag = 1  ; % use mean(x) when 0 and nanmean(x) when 1
```

🐐 When *G.Settings.NanMeanFlag* is set to 1, GOAT uses nanmean instead of mean for averaging operations.

```
Plevel1 = 1000 ; % GOAT will use the nearest level to this value
Plevel2 = 1000 ; % GOAT will use the nearest level to this value
Lon1    = 0   ; % use values from -360 to 360. GOAT will use the nearest
  values to the ones specified
Lon2    = 358.125   ; % use values from -360 to 360. if Lon2 < Lon1 the
% interval Lon2 < lon < Lon1 will not be used.
Lat1    = 88.542   ; % use values from -90 to 90. GOAT will use the nearest
% values to the ones specified.
Lat2    = -88.542   ; % Lat2 does not have to be larger than Lat1.
```

🐐 Set the spatial truncation of the data. GOAT will find the indices closest to the specified levels, longitude and latitude limits specified. NOTE: the calculation is sensitive to the order of the longitude limits (Lon 1 and 2) but not to the levels and latitude limits (Plevel 1 and 2 and Lat 1 and 2). For example, setting Lon1 = 100 and Lon2 = 200 will result in the retrieval of data between 100E and 200E longitudes. Setting Lon1 = 200 and Lon2 = 100 will result in the retrieval of data between 160W and 100E longitudes.

```
G.Limits.Default.P1   = dsearchn(double(G.P),Plevel1) ;
G.Limits.Default.P2   = dsearchn(double(G.P),Plevel2) ;
G.Limits.Default.Lat1 = dsearchn(G.lat,Lat1)  ;
G.Limits.Default.Lat2 = dsearchn(G.lat,Lat2)  ;
G.Limits.Default.Lon1 = dsearchn(G.lon,Lon1)  ;
G.Limits.Default.Lon2 = dsearchn(G.lon,Lon2)  ;
```

🐐 Assign appropriate indices to entered limits.

```
% Filters, Weights and Indices:
% No Filters, Weights or Indices Selected
G.Filter.Index = ones(length(G.Addresses.Default),1) ; % default index
```

🐐 No filters set in this example. Filter properties set at the FILTERS GUI will appear in this section. An Index filter (the length of *G.Default.Addresses*) of all ones is set by default.

🐐 Retrieval and compilation of the data is done in cell-4. The function *GOAT_Compile_data_Series* produces a time series (stored in *G.DataSeries*) of the selected data and a mean field (*G.Data*). The dates of the time series are stored in *G.Dates*. The filtered (using the index *G.Filter.Index*) dates are stored in *G.FilteredDates*. In the present example *G.Dates* and *G.FilteredDates* are the same. The original longitude, latitude and level vectors are stored in *G.lon*, *G.lat* and *G.P*, respectively. The truncated longitude and latitude vectors are stored in *G.x* and *G.y*, respectively. Meshed longitude and latitude vectors are stored in *G.X* and *G.Y*. This makes it straight forward to produce *contour* (or *contourf*) plots of the data using *contour(G.X,G.Y,G.Data')*.

```
%% cell-4: retrieve and compile data
G = GOAT_Compile_data_Series(G) ;
Data  = G.Data              ; % Composited data
DataSeries = G.DataSeries ; % Time Series
Dates = G.FilteredDates    ; % Time Series Dates
x = G.x ; y = G.y          ; % x and y vectors
X = G.X ; Y = G.Y          ; % meshed x and y vectors. Use contourf(X,Y,Data')
  to plot data.
```

🐐 For time series that span more than one year, it is straight forward to calculate the climatology and time series anomaly from climatology. A smoothing (centered running average) factor of the climatology is assigned at *G.Limits.Default.DataClimSmooth*. Calculating the climatology is done using the function *GOAT_Compile_Climatology*. Once the climatology is calculated, the anomaly from climatology is calculated using *GOAT_Compile_ClimAnomaly*. The climatology and anomaly time series are stored in *G.DataClim* and *G.DataSeriesAnom*, respectively.

```
%% Optional: calculate climatology and anomaly (from climatology)
% G.Limits.Default.DataClimSmooth = 1 ; % Smooth climatology using a centered
  running mean (G.Limits.Default.DataClimSmooth = 3 5 7 ...). 1 = 'none'
% G = GOAT_Compile_Climatology(G)         ; % calculated climatology (monthly,
  daily or diurnal) from Time Series
% DataClim       = G.DataClim         ; % Climatology derived from the Time
  Series
% G = GOAT_Compile_ClimAnomaly(G)     ; % Anomaly from Climatology (monthly,
  daily or diurnal)
% DataSeriesAnom = G.DataSeriesAnom   ; % time series of the climatological
  anomaly of the Time Series
```

🐐 Plot properties are set cell-5.

```
%% cell-5: plot Right Hand Side and 'Settings' Plot Settings
% Right Hand Side and 'Settings' Plot Settings
G.Settings.ContourIdx        = 1        ; % Contour Type Idx







G.Settings.LineWidth         = -0.5 ; % Darken_Edge | None | .5 | 1 | 1.5 ...
```

```
G.topo.TypeFlag             = 2 ; % 1=None | 2=Atmo | 3=Ocean | 4=Atmo+Ocean
G.topo.interval = 1000 ;
switch G.topo.TypeFlag
  case {1,2}
  G.topo.intervalmin = 0     ;
  G.topo.intervalmax = 8000  ;
  case 3
  G.topo.intervalmin = -8000 ;
  G.topo.intervalmax =  0    ;
  case 4
  G.topo.intervalmin = -8000 ;
  G.topo.intervalmax = 8000  ;
end
G.topo.topovec = G.topo.intervalmin:G.topo.interval:G.topo.intervalmax ;
zeroidx = dsearchn(G.topo.topovec',0) ;
G.topo.topovec = G.topo.topovec(G.topo.topovec~=G.topo.topovec(zeroidx))  ;
G.topo.TopotLineWidthIdx = 2 ; % 0.5 | 1 | 1.5 | 2 | 2.5 ...
% Create Figure and Plot
Fig = figure('position',G.GUI.pos.Main)  ; % create a figure with the same
   size as the GOAT figure
set(0,'CurrentFigure',Fig)               ; % make sure it prints on new Fig
GOAT_plotUpdate(G)                         ; % Plot without GUI features
```

By default, all 2D plots are displayed as surface plots using the *contourf* function. It is possible to use other mapping functions using the index *G.Settings.ContourIdx*. This option is not enabled in the GUI. Not all mapping functions are guaranteed to work smoothly. The various available mapping functions are:
G.Settings.ContourIdx = 1: contourf
G.Settings.ContourIdx = 2: contour
G.Settings.ContourIdx = 3: surf
G.Settings.ContourIdx = 4: surfc
G.Settings.ContourIdx = 5: mesh
G.Settings.ContourIdx = 6: meshc
G.Settings.ContourIdx = 7: meshz
G.Settings.ContourIdx = 8: imagesc

```
colormapName = 'RdBu11'      ; % ColorMap Name as it appears in the GUI
G.Settings.ColorMapIdx = find(strcmp(G.Settings.ColorMap,colormapName))  ; %
   actual colormap is found in G.Settings.ColorMaps.colormapName
```

A map index, *G.Settings.ColorMapIdx*, needs to be specified. The string array *G.Settings.ColorMap* contains the names of all the maps available in GOAT. The actual colormaps are found in *G.Settings.ColorMaps*, with matching names.

```
G.Settings.ReverseColorMap   = 1    ; % 0/1 Reverse Colormap checkbox
```

Invert colormap checkbox (1/0)

```
G.Settings.CoastLineWidth    = 0.5  ; % 0 | .5 | 1  ...
```

Set coast line width, with indices as they appear in the GUI ('None', .5, 1, …)

```
G.Settings.XgridFlag         = 0    ; % A flag for displaying the X Grid Lines
G.Settings.YgridFlag         = 0    ; % A flag for displaying the Y Grid Lines
```

```
G.Settings.SymZLimitsFlag    = 1    ; % A flag for setting limits symmetric
   with respect to zero (good for divergent colormaps)
G.Settings.FixZLimitsFlag    = 0    ; % A flag for fixing limits (useful in
   auto time steps)
G.Settings.LogLevelsFlag     = 0    ; % log(P) in lat-P,Lon-P projections
G.Settings.SinLatFlag        = 0    ; % plot sin(latitude) in projections
   involving latitudelog
```

X-grid, Y-grid, Sym-Z, Fix-Z and Log-Y checkboxes (1/0)

```
G.Settings.ContourLevelStep  = 0    ; % contour step. 0 = auto
```

Set contour step level. When = 0, MATLAB decides.

```
G.Settings.LineWidth         = -0.5 ; % Darken_Edge | None | .5 | 1 | 1.5 ...
```

Set contour line width, as it appears in the GUI. when *LineWidth* < 0, GOAT uses *GOAT_darken_edge.m* to darken the edges of surface contours. *LineWidth* = 0 refers to 'linestyle' = 'none'. *LineWidth* > 0 sets the line width.

```
G.topo.TypeFlag              = 2 ; % 1=None | 2=Atmo | 3=Ocean | 4=Atmo+Ocean
G.topo.interval = 1000 ;
switch G.topo.TypeFlag
  case {1,2}
  G.topo.intervalmin = 0     ;
  G.topo.intervalmax = 8000  ;
  case 3
  G.topo.intervalmin = -8000 ;
  G.topo.intervalmax =  0    ;
  case 4
  G.topo.intervalmin = -8000 ;
  G.topo.intervalmax = 8000  ;
end
G.topo.topovec = G.topo.intervalmin:G.topo.interval:G.topo.intervalmax ;
zeroidx = dsearchn(G.topo.topovec',0) ;
G.topo.topovec = G.topo.topovec(G.topo.topovec~=G.topo.topovec(zeroidx))  ;
G.topo.TopotLineWidthIdx = 2 ; % 0.5 | 1 | 1.5 | 2 | 2.5 ...
```

Set topography display definitions, as they appear in the GUI. The zero contour level is removed from topography and replaced with the coast line.

```
% Create Figure and Plot
Fig = figure('position',G.GUI.pos.Main)  ; % create a figure with the same
size as the GOAT figure
set(0,'CurrentFigure',Fig)               ; % make sure it prints on new Fig
GOAT_plotUpdate(G)                         ; % Plot without GUI features
```

Create figure and plot. The function *GOAT_plotUpdate* performs all of the plotting in the GOAT GUI.

Optional: cell-6 shows an example of how to save and print the data.

```
%% cell-6: Save
if strcmp(G.GOAT_Directory,cd) % If working from . . . GOAT/Code
    ExportFolder = fullfile('..','Export','') ;
else
    ExportFolder = 'Export' ; mkdir(ExportFolder) ;
end
```

```
set(gcf,'PaperPositionMode','auto','papersize',[30  18])  ;
print('-dpdf','-r70',fullfile(ExportFolder,'SavedFigure','')) ;
save(fullfile(ExportFolder,'SavedFigure.fig'),'Fig')
save(fullfile(ExportFolder,'SavedData.mat'),'Data')  ;
```

🐐 If working from the GOAT/Code directory, GOAT will export data to GOAT/Export.

Example 2: Install linked data as MAT

The following example shows how to generate code for installing a LINK archive to MAT.

🐐 As shown before, the DataFolder, Calendar, and field are set in the first section. Note that in the following example the List Calendar is used so that a temporal span of years months or days is not required.

```
%% cell-1: Initiate GOAT and set: Dataset, Field and Calendar
G = GOAT_Environment('enter_project_path') ; % Initiate GOAT Environment
G = GOAT_Initialize_Dataset(G,'GOAT/Data_LINK/FGOALS-s2/r0i0p0','List') ; % Go
through contents of Data folder and Calendar. Link existing fields, temporal
span, etc.
G = GOAT_Initialize_Field(G,'areacello')        ; % get spatial and temporal
dimensions of field, display settings, topography, and file addresses. Link if
necessary
```

🐐 The list span of the desired fields is set in section 2.

```
%% cell-2: Temporal limits
% Temporal Checkboxes/Flags
G.Limits.Default.Day2same   = 1 ; % 1/0 Set Date 2 same as Date 1
G.Limits.Default.Sequential = 1 ; % 1/0 Sequential or Seasonal averaging Flag
G.Limits.Default.ClimatologyContinuous = 0 ; % present seasonal data
sequentially in Hovmoller plots
G.Limits.Default.Anom  = 0      ; % display anomalies in Hovmoller plots
ListFolder = 'fx'    ;
G.Addresses.LinearIdx = find(strcmp(G.Addresses.Linears,ListFolder))    ;
G.Limits.S1 = 1 ; % sequential index in linear time from 1 to end
if G.Limits.Default.Day2same
  G.Limits.S2 = G.Limits.S1 ;
else
  G.Limits.S2 = 1 ;
end
```

🐐 The function *GOAT_ALP_Install* installs the linked data in MAT archive format. The export address of the field is set in *G.ALP.ExportAddress*. An overwrite flag (1/0), *G.ALP.OverWriteFlag* indicates whether to overwrite existing MAT archive files during installation. When possible, setting *G.ALP.BoerBetaFlag* to 1 will install the BoerBeta field in the same folder and the field being installed.

```
%% cell-3: Install Linked Data
G.ProjectionType     = 10 ;
G.ALP.ExportAddress  = GOAT/Data_LINK/FGOALS-s2/r0i0p0/List/areacello ;
G.ALP.TimeResolution = 'List' ;
G.ALP.OverWriteFlag  = 0 ;
G.ALP.BoerBetaFlag   = 0 ;
G = GOAT_ALP_Install(G)  ; % Install as MAT archive format from OPeNDAP server
   or nc file
```

5.2 Virtual Fields

In GOAT lingo, Virtual fields refer to fields derived from other existing fields. It is possible to add an unrestricted number of such fields. Each Virtual field is created in a Virtual Field function, placed at
GOAT/MyScripts/MyVirtualFields/GOAT_VirtualField_(FunctionName).m

Virtual field function follow a protocol similar to the Auto Download Protocol (ADP). Shown below is an example of a virtual field that calculates Angular Momentum from the zonal wind field.

```
function …
  [G,data]=GOAT_VirtualField_AngularMomentum(Mode,G,datafile,DataFolder, …
                                                      TimeField,Aidx)
switch Mode
    case 'Active'
        G.VirtualFields.ActiveFlag =  1 ; % | 1 for 'yes' and 0 for 'no'
    case 'VirtualFieldName'
        G.VirtualFields.MyName = 'M' ;
    case 'RequiredFieldsList'  % The existing fields required for the
  calculation of the virtual field
        G.VirtualFields.RequiredFieldsList = {'ua'} ;
    case 'ReferenceField' % a field with similar Dims
        G.VirtualFields.ReferenceField = 'ua' ;
    case 'Dims'
        G.VirtualFields.DataFieldUnits = 'm^2/s' ;
        G.VirtualFields.FieldDescription = 'Total Angular Momentum' ;
    case 'Load'
        G.DataField = 'ua' ;
    [data,G] = …
  GOAT_load_data(fullfile(DataFolder,TimeField,G.DataField,datafile),G,Aidx);
        data = G.Consts.EarthRadius^2.*G.Consts.Omega.*G.Vectors.cos2mat
                  + G.Consts.EarthRadius.*G.Vectors.cosmat.*data ;
        G.DataField = 'M' ;
end
```

GOAT communicates with the VirtualField function via different 'Modes' (string), entered as the first input argument. The function outputs the GOAT structure G (which is also the second input argiment) and the virtual field data frame. By order of appearance, the input arguments after G are, the local file address with reference to the field folder (e.g., '1979/10100.mat'), the Data folder, Calendar, and the index of the file in the address string vector of the current analysis.

The following fields and functions must exist in a VirtualField function
G.VirtualFields.ActiveFlag: when 1, this field appears in the GOAT GUI Fields dropdown menu
G.VirtualFields.MyName: The name of the virtual field appears in the dropdown menu
G.VirtualFields.RequiredFieldsList: real fields than need to exist in order for the virtual field to be derived.
G.VirtualFields.ReferenceField: a field from which to 'borrow' the *Dims.mat* file.
G.VirtualFields.DataFieldUnits: units of the virtual field
G.VirtualFields.FieldDescription: Description of the virtual field

The code for deriving the field is places in the 'Load' Mode. In brief, the derivation function performs the following operations:
- sets the current field to '*ua*' (zonal wind).
- retrieves the data (which can be in MAT or LINK format) using the function GOAT_load_data.
- calculates (and stores in '*data*') angular momentum using the built-in consts and using the spatial vectors and matrices created for each loaded field
- re-sets current field to 'M'

GOAT comes with many virtual field function which can be used as templates for additional virtual fields.

5.3 Example scripts

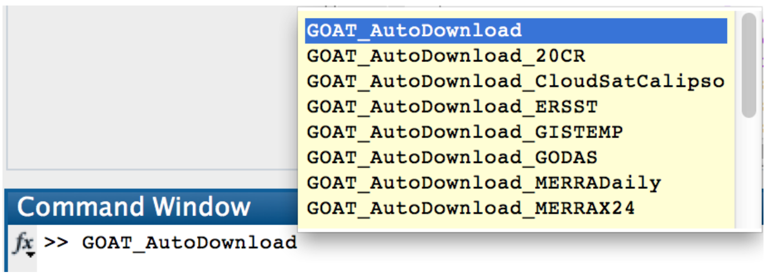
Several example scripts are included in the default GOAT version. These can be found in the GOAT function folder GOAT_Examples. Current examples are
GOAT_Examples_3DMean.m
An example of how to calculate a the time mean of a 3D field.

GOAT_Examples_CreateIndex.m
An example of how to create and save a filter date index for a given field.

GOAT_Examples_TimeSeriesSurvey.m
An example of how to perform and analysis that surveys several fields or datasets

6. GOAT Scripts and Toolboxes

GOAT functions are organized by underline folders. For example, by pressing typing at the command window
>>> GOAT_AutoDownload
and pressing the 'tab' key, all of the GOAT_AutoDownload scripts become visible via MATLAB's auto completion tool, as shown below



GOAT Toolboxes and functionalities are likewise organised.

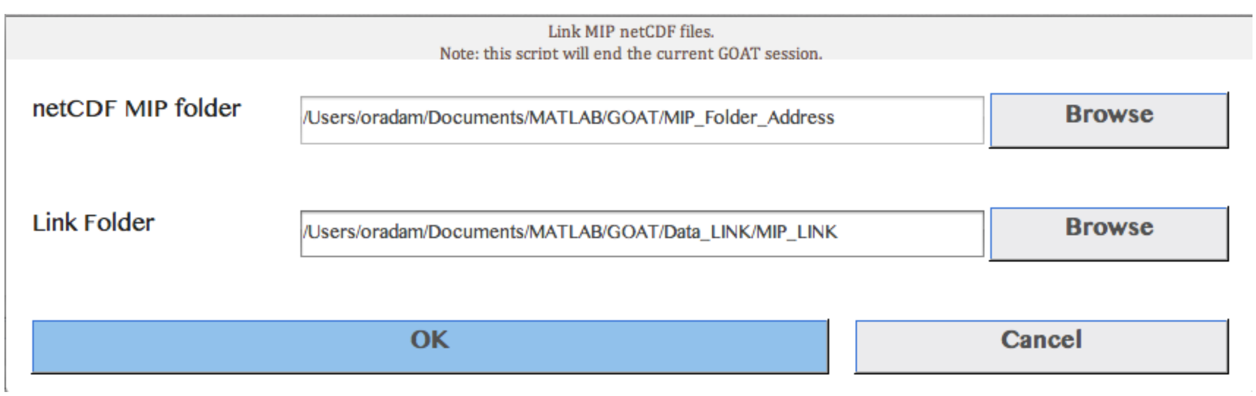
6.1 CMIP5 toolbox

The GOAT_CMIP5 toolbox consists of the following functions:

GOAT_CMIP5_LINK

an alias for the afore mentioned GOAT Script *GOAT_Script_MIP_LINK_DRS_1_3_1.m.* As shown below, this function presents a GUI requesting the user to assign

1. MIP Folder: a folder containing CMIP5 files (conforming to DRS 1.3.1 protocol)
2. A folder where LINK archives are to be created. It is best practice to use the same folder for all CMIP LINK archives.



The function then finds and links all of the CMIP5 files in theMIP folder, saving the LINK archive at the Link Folder. The function terminates the current GOAT session when run from the GUI Script panel.
Note: the linking process may be slow, as GOAT verifies each datafile and assigns links for each frame.

GOAT_CMIP5_FolderContent
[table , fileaddresses] = GOAT_CMIP5_FolderContent(Folder , filter_options)
This function returns a '*table*' and a list of '*fileaddresses*' of the attributes of the CMIP5 files contained in the folder address 'Folder'. A table of attributes is also displayed at the command window. The first input argument must be a folder address. additional filtering options are:

'*variable*' or '*var*': a string or array of strings listing variable names. E.g., {'*pr*' , '*tas*'} means look only for files containing the precipitation and surface temperature fields. By default no variables are filtered.

'*experiment*' or '*exp*': a string or array of strings of experiments. E.g., {'*amip*' , '*piControl*'} means look only for files associated with the *amip* and *piControl* experiments. By default no experiments are filtered.

'*model*' or '*mod*': a string or array of strings of climate model names to be included in the search. By default no model is filtered.

'*realm*' or '*rea*': a string or array of strings of Realms (e.g., 'Amon' , Omon', etc.) to be included in the search. By default no Realms is filtered.

'*ensemble*' or '*ens*': a string or array of strings of Ensembles (e.g., 'r1i1p1','r1i1p2', etc.) to be included in the search. By default no Ensemble is filtered.

'*begin*' or '*year1*': an integer. Show only files containing data dated later than the specified year.

'*end*' or '*year2*': an integer. Show only files containing data dated earlier than the specified year.

GOAT_CMIP5_parseName
P = GOAT_CMIP5_parseName(filename)
This function parses the name of a CMIP5 file and returns the attributes in a structure (P):

P.VarName (string): variable name.
P.RealmName (string): realm.
P.ModelName (string): climate model.
P.ExperimentName (string): experiment.
P.EnsembleName (string): ensemble.
P.TimeSpan (string): the time span string as it appears in the file name
P.TimeResolution (string): the equivalent GOAT Calendar ('*Daily*', '*Daily365*', '*Daily360*', etc.)
P.Year1 (integer): begin year
P.Month1 (integer): begin month
P.Day1 (integer): begin day
P.Year2 (integer): end year
P.Month2 (integer): end month
P.Day2 (integer): end day

GOAT_CMIP5_filter_wget_fields
FileAddress = GOAT_CMIP5_filter_wget_fields(wgetfile,FieldsToInclude,outputaddress)

A very common and useful method of downloading CMIP5 files is by using *wget*. For some servers, the automatically generated *wget* scripts contain more variables than desired. This function returns a *wget* file (‘*FileAddress*’) which includes on the the variables listed in ‘*FieldsToInlcude*’. Input arguments are:

wgetfile: address of original wget file
FieldsToInclude: a string or string array of variable names to be included in the wget file
outputaddress (optional): the edited wget file address. If not entered, the edited wget file will be created at the same folder as the original wget file, with ‘*_filtered*’ appended to the file name.

A.1 Scripts used by GOAT

GOAT incorporates several freely available scripts. In some cases, the scripts' names are appended with '*GOAT_*' initials. This is done in order to ensure version compatibility and to allow minor modifications in these scripts. In no way does incorporating any of the scripts in GOAT or appending the scripts with any initials imply authorship by Ori Adam. The list of scripts used by GOAT is given below:

- *cbrewer* tool by Charles Robert (2011): cbrewer scripts are appended with the initials '*GOAT_*' with some minor compatibility modifications in the code.
- *cejulienColorMaps.m*: original version by Julien Emile-Geay, LDEO, 2006.
- *cprintf.m*: by Yair Altman (2012).
- *csvimport.m*, by Ashish Sandanandan (2009). Name changed to csvimport_g.m.
- *findjobj.m*: by Yair Altman (2007).
- *jheapcl* tool: Java Heap cleaner tool by Davide Tabarelli (2012).
- *inpaintn.m*: interpolate over nan values. By Damien Garcia (2009).
- *shortcut_tools*, by Health and Safety Laboratory, 2010. Add MATLAB shortcuts.

A.2 References

G.J, Boer, 1982: Diagnostic Equations in Isobaric Coordinates. *Mon. Wea. Rev.*, **110**, 1801-1820.