OBLIMAP User Guide

version 1.0
accompanying OBLIMAP 2.0

Thomas Reerink

Institute for Marine and Atmospheric research Utrecht, Utrecht University, 3508 TA Utrecht, The Netherlands

November 21, 2016

Abstract

This OBLIMAP User Guide accompanies the OBLIMAP 2.0 release. This User Guide presents an overview of the new features in OBLIMAP 2.0. It describes the minimum system requirements, installation instructions for various platforms, and how to run OBLIMAP with a configuration file. The main goal of this User Guide is to provide the description of the 67 configuration variables of OBLIMAP. The scientific methods underlying OBLIMAP are described in *Reerink et al.* [2010, 2016].

Contents

1 Introduction

OBLIMAP is a mapping technique for exchanging climate fields between GCMs and ice models [*Reerink et al.*, 2010].

The output fields of a Global Circulation Model (GCM) can be mapped to an ice model (IM) with the OBLIMAP package. Mapping in the reverse direction, from an IM to a GCM, is possible as well with the OBLIMAP package. As such it acts as a coupler between grids which are based on geographical coordinates and grids which represent a flat surface.

With 'mapping' the combination of projecting and interpolation is meant. First the 'departure' grid points are projected on the 'destination' surface. Thereafter the field values of these projected points (their projected coordinates are known now) are interpolated at the destination grid points. In the latter step a certain search and weighting strategy is used. OBLIMAP incorporates two main search strategies to find the nearest projected points for each destination grid point: The quadrant and the radius method [*Reerink et al.*, 2010]. Both search methods are combined with the distance weighting strategy which is based on *Shepard* [1968].

GCM fields which are defined on a grid representing the Earth surface can be mapped with OBLIMAP to an IM grid which simply coincides with a flat surface,

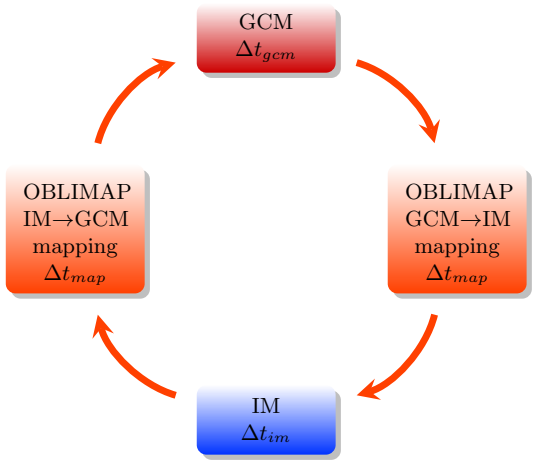

Figure 1: *Schematic representation of an embedded IM within a GCM run. The GCM and IM are coupled with the embedded OBLIMAP routines at each coupling interval Δt_{map}. The GCM and the IM evolve with their own time step Δt_{gcm} and Δt_{im} depending on their specific stability criteria. In fact these time steps might be flexible over time.*

and vice versa in the reverse direction. The Earth surface is usually represented by a sphere or the Earth ellipsoid (WGS84) with geographical coordinates. Various kinds of geographical grids exist, they might be regularly and irregularly spaced, both situations are supported by OBLIMAP. IM grids are often equidistantly spaced, however OBLIMAP 2.0 supports irregularly spaced IM grids as well.

OBLIMAP has been designed in order to couple climate models with ice sheet models, but it is a generic tool and can be used for various coupling problems. A coupled ice model returns, among other variables, the ice extent and the topography to a GCM, which in return are used by the GCM and results in a more realistic GCM simulation.

With OBLIMAP 2.0 ice models and GCMs can be coupled both 'off-line' and 'on-line'. In an off-line coupling approach the results of a GCM run are mapped to the IM and with them a new IM run is initiated. As soon as the IM run is finished the IM results are mapped to the GCM and a new GCM run is initiated, and so on.

In an on-line coupling approach the GCM calls each Δt_{map} the OBLIMAP GCM to IM mapping routine, run the ice model routine and call the OBLIMAP IM to GCM mapping routine. While the GCM and the IM evolve with their own time step Δt_{gcm} and Δt_{im}, as schematically shown in Fig. 1. This embedded coupling of an IM with a GCM requires the addition of a few lines in the GCM code, because the GCM has to call these routines. The fact that this on-line coupling is possible now, is one of the main achievements of the redesign of OBLIMAP 2.0 with respect to OBLIMAP's prior version. This meets the growing demand in climate science to couple ice models to GCMs.

The mathematical kernel of the package is described in *Reerink et al.* [2010], and their supplementary material contains the first open source code of OBLIMAP. The most important new OBLIMAP 2.0 features and algorithms are described in

Reerink et al. [2016], accompanied by the OBLIMAP 2.0 open source code and this OBLIMAP User Guide as supplementary material.

2 Why oblique projections?

With an oblique projection one can obtain the same projection quality for areas which lie asymmetric around the north or the south pole as with a polar projection for symmetric oriented areas. In fact, with an oblique projection any arbitrary area of interest can be optimal projected, with the smallest deviation of mapped distances in the mapped surface. With OBLIMAP it is simple and convenient to choose the precise area of interest, its orientation, and the number of desired grid points in each grid direction. This enables a quick way of mapping your topographic & forcing data sets in exactly the same way, so both data sets can be used as input for the same experiment. For example a local projection of the Antarctic Peninsula would be far off from optimal if the polar stereographic projection will be used. The same applies for areas like the Himalayas, Svalbard or Patagonia, but also for entire Greenland or parts of it. Applying an optimal projection is straightforward with OBLIMAP by using an oblique projection. An optimal projection with OBLIMAP depends on the selected grid extent, spanned by the grid numbers and the grid sizes, and the choice of the best α [See Eq. (2.2) & Fig. 3 *Reerink et al.*, 2010] for that configuration.

3 The quadrant and radius interpolation

The quadrant and radius method are both inverse squared distance weighting interpolation methods based on Donald Shepard's famous paper 'A two-dimensional interpolation function for irregularly-spaced data' which he published in 1968. His introduction discusses the shortcomings of the bilinear (called double linear), bicubic and other interpolation methods if applied to irregularly-spaced data.

The inverse squared distance weighting function has a few very practical advantages when interpolating spatial data, it is suited to identically treat: (1) regular and irregular spaced grid nodes, (2) 1D, 2D and 3D spatial grids, (3) any curved destination surface, i.e. the surface of a sphere and ellipsoid or the flat surface, (4) any number of weighting contributions. The weighted average is based on weighting the inverse squared distances of all the selected contributions. Although the weighting factors are default taken equal to the inverse *squared* distance as recommended by *Shepard* [1968], it is possible (also in OBLIMAP) to replace this exponent = 2 which makes it *squared*, by each exponent equal or larger than 1 instead.

This pure inverse squared distance weighting function has to be combined with a limiting influence distance in order to select only nearby points, as described by *Shepard* [1968]. The main reason for this limitation is the computational performance, but it also avoids the need of suppressing a possibly biased average in case there is a relative large number of distant points involved.

As noted by *Shepard* [1968] there are many ways to limit the number of contributions. In OBLIMAP we implemented two methods to limit the influence of distant contributions, based on three typical situations encountered by mapping: (1) a coarse grid is mapped on a fine grid, (2) a grid is mapped on a grid with a similar resolution, (3) a fine grid is mapped on a coarse grid.

The first and second situation are addressed by OBLIMAP's default interpolation method, the quadrant interpolation method, which draws a cross through the considered destination point, and selects in each quadrant the nearest projected contribution. It is a relative arbitrary choice to divide the surrounding area in four segments, in fact it could be divided in any number of segments. The choice for four segments is slightly inspired by the bilinear interpolation which also uses four surrounding points. This selection method does effectively shadow other contributions in the same segment/direction in a simple way. Note that with an increasing number of segments the shadowing becomes more direction sensitive.

In the third situation, in which a fine grid is mapped on a coarse grid, OBLIMAP uses the radius interpolation method which selects those contributions which lay within a certain radius. A reasonable radius typically equals half the departure grid size resolution. The basic idea is that the coarse destination grid cell obtains a representable average value. Because the the number of selected contributions increases approximately squared with an increasing selecting radius (given a constant node density), more distant points are selected but they weight squared inverse. This squared and inverse squared effect compensates and makes that the radius method generates a representable average estimate.

We preferred selecting within a radius over selecting the n nearest points which is another well known method, because the latter requires sorting which is notorious computational expensive for large n and complicates the interpretation of the results in situations with masks and data gaps, and also does not directly match the area size of the destination grid cell.

As the weighting function itself just weights over the number of detected contributions, segments are allowed to stay empty. Therefore the method is robust for destination grid domain edges, mapped departure grid domain edges, data gaps, and masked points where the mask is also allowed to differ per field and per vertical layer, and all these different masks are even allowed to change in time.

In both interpolation methods the distances between the considered destination point and a projected departure point are calculated over the destination surface along the great circle. Available surface curvatures in OBLIMAP are the surface of a sphere, an ellipsoid and of a flat plane.

Default OBLIMAP uses the quadrant interpolation method, but the radius interpolation method is automatically selected if the resolution of the destination grid is four times coarser than the resolution of the departure grid. The interpolation can be configured manually as well, for that and for all defaults see the OBLIMAP User Guide.

Independent of the interpolation method which has been used in the scan phase,

the nearest point assignment can be used in the post scan phase and will match with masks which change in time and differ per field and layer.

4 New features in OBLIMAP 2.0

The kernel, the projection and the selection of the points which contribute to each mapped grid point, remained in OBLIMAP 2.0 identical to the method which is described in *Reerink et al.* [2010]. Except for the grid edge extention as mentioned in the last sentence in Sect. 2.3.2 of *Reerink et al.* [2010] which is left behind in OBLIAMP 2.0, and the BUG fixes as reported in Sect. 10.

GCM fields can be mapped on an IM grid, and vice versa. Each unique GCM grid - IM grid combination requires a so called 'scan phase' if the mapping is conducted for the first time for this combination of grids. During this scan phase the most essential mapping information for each mapped point is written to the scanned indices and distances file: the SID file. Note that the interpolation at each mapped point might use a different number of projected points (contributions). For each of this contributions the following data is stored: 1. The grid indices at the departure grid of the projected point. 2. The relative distance between the projected point and the considered mapped point. The header of the SID file describes the precise data format and contains the configuration of the scan determining parameters. Default this SID file can be found in the directory:
oblimap-package/oblimap-sid-files/

OBLIMAP 2.0 reads this file which contains the most essential mapping information and stores its data in a large internal object: the dynamic data object (DDO). This DDO is used for the final mapping of the field(s). If the SID file already exists from a previous equal mapping session, then the scan phase can be skipped by using this SID file. Note that the scan phase only depends on the grid configurations, the selected projection method and the selected interpolation method. So in fact the actual field values do not matter during the scan phase.

The new features in OBLIMAP 2.0 are summarized below:

- With the fast scan method the scanning of contributions is conducted in small local grid boxes around a certain pivot contribution. The pivot is a near contribution of a previous scanned point. The pivot is known in most cases, if not a general search is performed which will be optionally messaged. The size of the local box is default determined dynamically by OBLIMAP or it can be specified by the user, in that case it should be large enough, see Sect. 9.2.33).

- The mapping of large grid combinations has become feasible with the fast scan option, because of the dramatic performance improvement in the scan phase.

- With masked mapping points which are part of an invalid mask will not contribute to the interpolation. The invalid mask covers the area for which

the field values of a certain user specified field matches with a certain user specified invalid value.

- With multiple masked mapping each field can use its own mask. This mask is allowed to vary in time, and might vary per vertical layer in case the masking is based on a spatial 3D field.
- The redesign of OBLIMAP allows embedded calling of the OBLIMAP mapping routines.
- The introduction of a dynamic data object avoids superfluous reading of the SID file. This improved the post scan phase performance, in particular in combination with multiple record, multiple field and multiple layer (3D) mapping.
- On-line coupling of an IM with a GCM is possible by using the OBLIMAP routines embedded. A prior off-line scan is recommended for both mapping directions, so the embedded OBLIMAP mapping routines can use the fast post scan mapping relying on the dynamic data object.
- Nearest point assignment, a post scan alternative to the quadrant and radius interpolation method. I.e. each destination node obtains the field value of the nearest projected point, which implies that no interpolation is required. This option can be considered in case both grids have about the same resolution.
- With multiple field mapping several fields can be mapped simultaneously, where the number of fields is unlimited.
- The mapping of spatial 3D fields while simultaneously spatial 2D fields can be mapped. Each 2D or 3D field might also contain the unlimited time dimension.
- With multiple record mapping several fields can be mapped simultaneously for all records, where the number of fields and records are unlimited. In the embedded case the fields will be mapped at each coupling time step.
- The precise calculation with the Vincenty method of the great distance over the ellipsoidal arc is added as an option.
- With the automatic scan option the scan parameters are determined by OBLIMAP itself.
- Automatic OBLIMAP advice concerning the best and optimal estimates for the four scan parameters:
  1 Is the data set cyclic in longitude or not?
  2 Which interpolation method should be used: the quadrant or the radius method?
  3 What is the best size for the search radius if the radius method is used?
  4 What is the optimal α for the considered grid?
- Extended OBLIMAP messaging on user and programmer level, including four levels of message intensity.

- Automatic dimension shape determination while reading the netcdf input files, so awkward spatial and time dimension settings will not bother the user any longer.
- The option to read the IM grid coordinates from a netcdf file. This enables mapping on an irregular IM grid.
- The origin of the IM coordinates is allowed to be anywhere. Before it was fixed to the IM grid center.
- Separate configuration files have to be used for each mapping direction. This is more transparent to work with, and reduced the number of config variables.
- To generate high flexible netcdf output, an option is added which reduces dummy dimensions, i.e. dimensions which have size 1 can be omitted in the created netcdf file.
- When an IM field is mapped to a GCM field it will by default be merged with an existing (pre-mapped) GCM field. However, in case such a GCM field is not available, the option exists to ignore the reading of certain (pre-mapped) GCM fields.
- Shifted mapping (off-centered mapping), but not recommended. This makes only sense for remapping of a product which was mapped by another package where shifts were applied to fit the area. Off-centered or shifted mapping conflicts with the OBLIMAP vision, because with OBLIMAP it is straightforward to perform an optimal projection by choosing the right λ_M and ϕ_M [See Fig. 2 and Sec. 2.2.2 in *Reerink et al.*, 2010].
- Rotational and inverse rotational mapping (local ice model to ice model mapping). I.e. a specific part of the ice model grid can be mapped to a local higher resolution ice model grid, and vice versa. Instead of a projection the mapping will conduct a rotation over an angle θ which can be specified by the user.
- Remapping an irregular distributed grid by using the rotational mapping with $\theta = 0$. In case the grid coordinates of a certain netcdf input file are unequally spaced, it is possible to map these irregular distributed grid points on an equal distance grid by using the unit projection with $\theta = 0$ with the rotational mapping option. Note that this rotational mapping is designed for the flat surface grids, i.e. the distances are calculated for the flat plane. For remapping irregular distributed grids on the Earth surface the distances should be measured over the great circles, which is not implemented as an option for the rotational mapping.
- This OBLIMAP User Guide accompanies OBLIMAP 2.0, which has been added to guide on user level.

5 Minimum system requirements

The following software should be installed on your system to be able to compile OBLIMAP and to examine the results:

- gfortran
- netcdf
- ncview
- nco
- csh
- nedit or any other text editor, to edit your configuration files

6 Cross platform compatibility

OBLIMAP is written in Fortran and should compile at any platform. Makefile include files are added for the following platforms:

- linux Ubuntu with the gfortran compiler (default for the OBLIMAP-package). The final OBLIMAP 2.0 release has been tested on the latest available LTS version of Ubuntu, Ubuntu 16.04.
- mac OS X 10.10 with the gfortran-mp-4.9 compiler
- unix server platforms with the gfortran and the intel ifort compiler

OBLIMAP compiles for both the gfortran and intel compiler without any warning and debug complains, as proven by the Valgrind debugger.

7 Compiling OBLIMAP

To compile the OBLIMAP Fortran source, one has to include the correct Makefile.include file in the Makefile. Check which Fortran compiler you are using. The default Makefile.include is the

Makefile.gfortran

which is included at the top of the Makefile, so e.g. Ubuntu users don't have to change anything. However, other Makefile.include examples for the mac and for some linux server platforms can be included by changing the top lines in the Makefile. The available Makefile.include files are listed below:

Makefile.gfortran

```
Makefile.gfortran-mp-4.9-mac
Makefile.gfortran-lisa
Makefile.ifort-lisa
Makefile.ftn-cca-ecmwf
Makefile.ifort-stampede
```

CCA is a Cray XC30 High Performance Computer (HPC) at ECMWF. LISA is one of SURF SARA's HPC linux platforms. SURF SARA is a company which supports computational science in The Netherlands on a national level. Stampede is one of XSEDE's HPC platforms in the U.S.A.

In the Makefile.include one should check the NETCDF path, in the default Makefile.gfortran this is:

```
NETCDF_DIR = /usr/local
```

After these checks or adjustments one can compile the OBLIMAP source by

```
cd src/
make all
```

8 Running OBLIMAP

8.1 Mapping

Default running should be done from the oblimap-package/ directory with for example:

```
./src/oblimap_gcm_to_im_program config-files/oblimap/ccsm-to-im/config_oblimap_ccsm_to_im_greenland
```

and for the inverse direction with:

```
./src/oblimap_im_to_gcm_program config-files/oblimap/im-to-ccsm/config_oblimap_im_to_ccsm_greenland
```

Or with the tiny script:

```
./oblimap-to-and-fro-mapping.csh
```

8.2 The verification script

One can make use of the small verification script which helps to find the fastest scan parameter. Although the gain might be very limited to the default fast scan method. Actually this script is most useful for benchmarking and testing.

```
./verify-oblimap-scan-search-block-size.csh config-files/oblimap/ccsm-to-im/config_obli map_ccsm_to_im_greenland 3 2
```

8.3 Remapping

Data which has already been projected on an IM grid with for instance a polar projection, can be remapped with an oblique projection by a two step approach: In the first step the IM grid points are inverse projected to lon-lat coordinates with the oblimap_convert_program which is also available in the OBLIMAP-package:

./src/oblimap_convert_program config-files/oblimap/convert-for-ice2sea/config_oblimap_convert_ice2sea_01x01km

In the second 'usual mapping' step one can map the data with the preferred oblique projection, for example with:

./src/oblimap_gcm_to_im_program config-files/oblimap/ice2sea-to-im/config_oblimap_ice2sea_to_im_greenland_20x20km

Note that the data fields remain identical after the first step as they are not interpolated. So this remapping procedure interpolates only once on the final grid.

Applying the oblimap_convert_program to some IM data sets can also be useful for plotting IM fields with pythons basemap library.

9 Configuration files

OBLIMAP 2.0 works with a separate configuration file for each mapping direction. A configuration file, often called config file for short, is an ascii file containing the configuration variables which enables to configure each mapping. The number and order of the listed configuration variables in the configuration file is not prescribed. Those configuration variables which are not listed keep their predefined OBLIMAP settings. The format of the configuration file is based on Fortran NAMELISTs.

9.1 config file example

An example of a config file in Fig. 2 illustrates the use of the regular config variables. The user is free to choose the amount of fields which should be mapped. The field names for the input file have to be specified in the config file. On output these field names are allowed to be different. The fields are numbered starting at one. These numbers are used as the index of these config variables which are arrays themselves. These arrays allow a well structured way to contain the various field names. With similar array config variables the units and the longnames of the fields can be specified. Of course the same array index always corresponds to the same field number. A proper way to specify array like config variables is shown in the config example in Fig. 2. The example shows negative indices for the 'dimensional variables'. The first spatial dimension (here the x-axis) corresponds with index -1, the second spatial dimension (here the y-axis) corresponds with index -2, the time dimension corresponds with index -3, the maximum number of the first spatial dimension (here NX) corresponds with index -4, and the maximum number of the first spatial dimension (here NY) corresponds with index -5. The

```
! ./src/oblimap_gcm_to_im_program config-files/oblimap/ice2sea-to-im/↩
    config_oblimap_ice2sea_to_im_greenland_20x20km
! Mapping the ice2sea data on to an IM grid for Greenland.

&CONFIG

! Grid:
NLON_config                     =   2501   ! The number of GCM grid points in the longitude direction
NLAT_config                     =   3001   ! The number of GCM grid points in the latitude  direction

NX_config                       =     77   ! The number of IM grid points in the x-direction
NY_config                       =    141   ! The number of IM grid points in the y-direction
dx_config                       =  20000   ! The           IM grid size   in the x-direction (in meter)
dy_config                       =  20000   ! The           IM grid size   in the y-direction (in meter)

! Projection:
lambda_M_config                 =    319   ! The longitude coordinate of the middle point of projection
phi_M_config                    =     72   ! The latitude  coordinate of the middle point of projection
alpha_stereographic_config      =      7.1 ! alpha determines the standard parallel of the projection
choice_projection_method_config = 'oblique_stereographic_projection_ellipsoid_snyder'

! Scanning:
scanning_mode_config            = .TRUE.
sid_filename_config             = './oblimap-sid-files/sid-file-ice2sea-to-im-greenland-20x20km.txt'
choice_quadrant_method_config   = .FALSE.

! Interpolation:
R_search_interpolation_config   = 8000
shepard_exponent_config         = 2
nearest_point_assignment_config = .FALSE.

! The input data below is obtained by:
!  ./src/oblimap_convert_program config-files/oblimap/convert-for-ice2sea/config_oblimap_convert_ice2sea_01x01km
gcm_input_filename_config       = './data/ice2sea/ice2sea-greenland-geometry-1km-updated-at-lon-lat.nc'
gcm_record_range_config         = 1,1

! File and field properties:
number_of_mapped_fields_config  = 5
im_field_name_config(1)         = 'Surface Elevation'
im_field_name_config(2)         = 'Ice Thickness'
im_field_name_config(3)         = 'Bedrock Elevation'
im_field_name_config(4)         = 'longitude'
im_field_name_config(5)         = 'latitude'
gcm_field_name_config(-4)       = 'latitude'          ! Should match with the gcm input file
gcm_field_name_config(-2)       = 'longitude'         ! Should match with the gcm input file
gcm_field_name_config(1)        = 'Surface Elevation' ! Should match with the gcm input file
gcm_field_name_config(2)        = 'Ice Thickness'     ! Should match with the gcm input file
gcm_field_name_config(3)        = 'Bedrock Elevation' ! Should match with the gcm input file
gcm_field_name_config(4)        = 'longitude'         ! Should match with the gcm input file
gcm_field_name_config(5)        = 'latitude'          ! Should match with the gcm input file

im_field_unit_config(-4)        = 'meter'
im_field_unit_config(-2)        = 'meter'
im_field_unit_config(1)         = 'meter'
im_field_unit_config(2)         = 'meter'
im_field_unit_config(3)         = 'meter'
im_field_unit_config(4)         = 'degrees'
im_field_unit_config(5)         = 'degrees'

im_field_longname_config(-4)    = 'y-axis (meter)'
im_field_longname_config(-2)    = 'x-axis (meter)'
im_field_longname_config(1)     = 'Surface Elevation (meter)'
im_field_longname_config(2)     = 'Ice Thickness (meter)'
im_field_longname_config(3)     = 'Bedrock Elevation (meter)'
im_field_longname_config(4)     = 'longitude (degrees)'
im_field_longname_config(5)     = 'latitude (degrees)'

! Each field can use a masked mapping based on the invalid value pattern of one of the mapped fields.
! Usually the masked mapping will be based on the invalid value pattern of the considered field itself.
! No masked mapping will be conducted in case this value is negative or zero.
field_which_determines_invalid_value_mask_config(1) = -1
field_which_determines_invalid_value_mask_config(2) = -2
field_which_determines_invalid_value_mask_config(3) =  3
field_which_determines_invalid_value_mask_config(4) = -4
field_which_determines_invalid_value_mask_config(5) = -5

invalid_input_value_config(3)   = -9999

im_created_filename_config      = './input_fields/greenland/initial-greenland-ice2sea-20x20km.nc'

! Messaging:
protect_file_overwriting_config = .FALSE.
oblimap_message_level_config    = 0

! Additional allocation:
oblimap_allocate_factor_config  = 110
/
```

Figure 2: *Example of an OBLIMAP config file.*

time dimension corresponds with the unlimited record dimension which is common practice in netcdf formats, and therefore don't have to be specified. The possibility to specify all properties of the netcdf fields, enables direct reading of any netcdf file by OBLIMAP.

The next section describes in detail all config variables and their options which can be used in the config file. The ⋆-labeled config variables are usually present in a config file, where the presence of some of them depend on the mapping direction. The remaining config variables are often omitted in the config file, and their default settings are used. However, as soon more specific demands emerge, like masked mapping, one can add them to the config file as well.

9.2 Specifying the functionality of each scanning config variable

This section lists all 'scanning config variables', i.e. these config variables determine the scanning. Changing them makes no sense if the scan phase is switched off and an earlier created SID file is read. OBLIMAP will give warning messages in case the scanning variables in the config file differ from those logged in the header of the SID file.

9.2.1 gcm_input_filename_config ⋆

The filename of the GCM netcdf input file.

9.2.2 im_input_filename_config ⋆

The filename of the IM netcdf input file.

9.2.3 NLON_config ⋆

The number of grid points corresponding with the longitude dimension in the GCM file.

9.2.4 NLAT_config ⋆

The number of grid points corresponding with the latitude dimension in the GCM file.

9.2.5 NX_config ⋆

The number of grid points corresponding with the X dimension in the IM file.

9.2.6 NY_config ⋆

The number of grid points corresponding with the Y dimension in the IM file.

9.2.7 dx_config ⋆

The grid size corresponding with the X dimension in the IM file.

9.2.8 dy_config ⋆

The grid size corresponding with the Y dimension in the IM file.

9.2.9 choice_projection_method_config

The desired projection method can be selected here. The projections are based on *Snyder* [1987] and *Reerink et al.* [2010]. The oblique stereographic projection and the Lambert equal-area projection are both available in combination with the Earth represented by a sphere or by the WGS84-ellipsoid. Additionally a rotation projection is available for any rotation in a flat plane, which enables to embed a local IM within a larger scale IM. For each of these projections the inverse projection is available as well.

9.2.10 earth_radius_config

Projections which use a sphere as destination surface, use this radius. The default value for the Earth radius is 6371221 meter in OBLIMAP.

9.2.11 ellipsoid_semi_major_axis_config

The semi major ellipsoid axis or equatorial radius is used in the projections which use an ellipsoid as destination surface. OBLIMAP takes by default the WGS84 value for the Earth equatorial radius, which is 6378137 meter.

9.2.12 ellipsoid_eccentricity_config

The eccentricity of the ellipsoid is used in the projections which use an ellipsoid as destination surface. OBLIMAP takes by default the WGS84 value for the eccentricity of the Earth ellipsoid, which is 0.08181919084262149.

9.2.13 lambda_M_config ⋆

The λ_M specifies the longitude of the middle point of interest M of the mapped area [See Fig. 2 and Sec. 2.2.2 in *Reerink et al.*, 2010].

9.2.14 phi_M_config ⋆

The ϕ_M specifies the latitude of the middle point of interest M of the mapped area [See Fig. 2 and Sec. 2.2.2 in *Reerink et al.*, 2010].

9.2.15 alpha_stereographic_config ⋆

Specifies the exact (oblique) stereographic projection. I.e. the stereographic projection with the standard parallel at α degrees [See Figure 3 and Section 2.2.2 in *Reerink et al.*, 2010]. If the mapped area is small then an optimal α [See Eq. (2.2) in *Reerink et al.*, 2010] will be a few degrees only. It should be mentioned here that this optimal α is based on the entire rectangular grid, however in case the grid is chosen relatively spacious, e.g. in order to cover an asymmetric peninsula, then a slightly lower α might be on average even more appropriate in relation to the considered area. If oblimap_message_level_config >= 1 then OBLIMAP will inform about an optimal α with respect to the masked area as well.

9.2.16 theta_rotation_projection_config

Specifies the rotational angle θ in case the rotation projection is selected.

9.2.17 shift_x_coordinate_rotation_projection_config

Specifies the shift along the x-axis of the local IM relative to the x-center of the IM in case the rotation projection is selected.

9.2.18 shift_y_coordinate_rotation_projection_config

Specifies the shift along the y-axis of the local IM relative to the y-center of the IM in case the rotation projection is selected.

9.2.19 enable_shift_im_grid_config

Recommended default: false. Should only be used for remapping of a non-OBLIMAP off centered mapping.

9.2.20 shift_x_coordinate_im_grid_config

Recommended default: 0. Should only be used for remapping of a non-OBLIMAP off centered mapping.

9.2.21 shift_y_coordinate_im_grid_config

Recommended default: 0. Should only be used for remapping of a non-OBLIMAP off centered mapping.

9.2.22 alternative_lambda_for_center_im_grid_config

Recommended default: λ_M. Should only be used for remapping of a non-OBLIMAP off centered mapping.

9.2.23 alternative_phi_for_center_im_grid_config

Recommended default: ϕ_M. Should only be used for remapping of a non-OBLIMAP off centered mapping.

9.2.24 unit_conversion_x_ax_config

In case the units of the x-axis in the input netcdf file differ from meter, this config variable determines the conversion factor. E.g. to convert from kilometers to meters: unit_conversion_x_ax_config = 1000. Default this value is 1.

9.2.25 unit_conversion_y_ax_config

In case the units of the y-axis in the input netcdf file differ from meter, this config variable determines the conversion factor. E.g. to convert from kilometers to meters: unit_conversion_y_ax_config = 1000. Default this value is 1.

9.2.26 use_prefabricated_im_grid_coordinates_config

With this option one can choose to read the IM grid coordinates from a netcdf file. This is obligatory in case the IM grid is irregularly spaced. Default this config variable is set to false, and a regular IM grid is generated. In case the IM grid is read from a file, the horizontal spatial coordinates are allowed to be equally or irregularly spaced in 1D or 2D.

9.2.27 prefabricated_im_grid_filename_config

The filename of the netcdf file which contains the IM grid coordinates in case these coordinates are read from a file if use_prefabricated_im_grid_coordinates_config = true.

9.2.28 scanning_mode_config ⋆

Each unique grid combination requires a 'scan phase' in case the mapping is conducted for the first time for this grid combination. During this scan phase the most essential mapping information, the relative distance of the projected points to the considered point and the indices of the projected points at the departure grid, is written to the SID file. The default location of this file is the directory: oblimap-package/oblimap-sid-files/. The header of this file contains a description of the

file format and a description of the essential config variables and their used values which determined the result of the scanning. To emphasize: the scan phase only depends on the grid configurations, the projection and the interpolation methods. So it doesn't matter what the actual field values are during the scan phase.

9.2.29 level_of_automatic_oblimap_scanning_config

This option allows to use the best estimates by OBLIMAP itself for the four scan parameters. Raising level_of_automatic_oblimap_scanning_config includes for each higher level an additional automatic scan parameter, in the order as specified below:

0: no automatic scan parameter is used
1: data_set_is_cyclic_in_longitude
2: choice_quadrant_method_config
3: R_search_interpolation_config
4: alpha_stereographic_config

OBLIMAP will message, if oblimap_message_level_config > 1, which scan parameters are overruled by this automatic values. The default is level_of_automatic_oblimap_scanning_config $= 3$.

9.2.30 data_set_is_cyclic_in_longitude

If level_of_automatic_oblimap_scanning_config $= 0$, the data_set_is_cyclic_in_longitude can be set manually. It should be TRUE for GCM to IM mapping if the GCM data set is cyclic in longitude, i.e. the GCM grid covers the entire 0-360 degrees longitude range. However, it is recommended, safe and convenient to leave this to the OBLIMAP package.

9.2.31 choice_quadrant_method_config

Specifying which interpolation method has to be used during the scan phase: the quadrant method or the radius method. With the quadrant method the nearest mapped points in the four surrounding quadrants are selected and these four points are interpolated by a Shepard distance-weighted averaging [*Shepard*, 1968]. With the radius method all points within a certain radius R_search_interpolation_config are interpolated by a Shepard distance-weighted averaging. Both interpolation methods are described extensively in *Reerink et al.* [2010]. No matter which of those interpolation methods is used during the scan phase, the 'nearest point assignment' can always be used in the post scan phase. OBLIMAP will inform the user in case it would be preferable to switch between the quadrant and the radius interpolation method. Raising oblimap_message_level_config will also provide more advice on this issue.

9.2.32 R_search_interpolation_config

The radius within mapped points contribute to the interpolation, in case the radius interpolation method is used. If oblimap_message_level_config >= 1 then OBLIMAP will message a proper value, which is 0.8 times half the grid size. This config variable is expressed in meters.

9.2.33 scan_search_block_size_config

This config variable is added in OBLIMAP 2.0, and is a key variable in the fast scan method. In principle scan_search_block_size_config can have any positive integer value starting from 0: it is the number of grid points within a local search block at the departure grid. Within this block all grid points are scanned to find the nearest projected contributions. Besides the scan_search_block_size_config can have three special values. If scan_search_block_size_config = -1 at each destination point a full search over all departure grid points is performed, this is the classic way of OBLIMAP's first release and still useful for benchmarking and testing. If scan_search_block_size_config = -2 OBLIMAP determines for each destination grid point a propper scan search block size, though this works fine for most situations the most robust method which is slightly slower will be used by setting scan_search_block_size_config = -3. In this case the same method as with scan_search_block_size_config = -2 is used but with an additional dynamic component: The internal estimated scan search block size is used, whereafter the scan search block size is raised each step until no new contributions are found. This scan_search_block_size_config = -3 is the most robust and convenient option for fast scan method, this option is used by the automatic scanning (which itself is default). The fast scan method allows the mapping of large grid combinations. In our high resolution case for Greenland we gained more than a factor 200.000 in computational time.

Though the scan_search_block_size_config = -3 option is strongly recommended, the fastest scanning might be achieved by using the smallest correct value for scan_search_block_size_config. However, the disadvantage in this case is that a check or a save range for scan_search_block_size_config is required. The convenient csh script verify-oblimap-scan-search-block-size.csh, which is distributed within OBLIMAP 2.0, facilitates this check.

9.2.34 scan_search_block_size_step_config

If the default dynamic scan_search_block_size_config = -1 is used, the local search block will be increased as long new contributions are found. This local block will be increased in the plus and minus direction of both grid directions by scan_search_block_size_step_config. The minimum value should be 2, which is also the recommended default. Actually there is no need to change this setting, it is included as a config variable for benchmarking and tests only.

9.2.35 vincenty_method_for_ellipsoid_config

In case one prefers a precise calculation of the relative distances over the ellipsoidal surface towards the mapped contributions, these distances have to be calculated by Vincenty's formula for the ellipsoid geodesic. In OBLIMAP the inverse numerical approximation of [*Vincenty*, 1975a] is implemented, for further reading we refer to [*Karney*, 2012; *Vincenty*, 1975]. With vincenty_method_for_ellipsoid_config = true the distances on the ellipsoid will be calculated by using Vincenty's method within the scan phase. Default these distances over the ellipsoid are approximated by distances over the sphere, assuming that the differences are small because the contribution points are near on the globe compared to the Earth radius. The disadvantage of using Vincenty's method is the increase in computational time during the scan phase.

9.2.36 sid_filename_config

This scanned indices and distances file (the SID file) is created during the scan phase. The SID file is read only once and contains the most essential mapping information. Whereafter its content, the indices of the projected points and the relative distance of the projected points to the considered point, is stored in the dynamic data object (DDO). This DDO is used for the final mapping of the field(s). If the SID file already exists from a previous equal mapping session, then the scan phase can be skipped by using this SID file.

9.2.37 backward_sid_filename_config

In case the OBLIMAP routines are used in an on-line coupling approach two different SID files are required, one for each mapping direction. The recommended on-line coupling implementation will use the SID files resulting from prior off-line conducted scans in both directions. Whereupon the on-line coupling only uses the post scan phase by using the SID files with the file names corresponding with the config variables scanned_projection_data_filename_config and the backward_sid_filename_config.

9.2.38 oblimap_allocate_factor_config

In the radius interpolation method the number of 'contributions' is unknown on forehand. The default allocation is ample, but in case it appears insufficient the allocation can be extended by setting this factor larger than one.

9.3 Specifying the functionality of each post scan config variable

This section list all 'post scan config variables'. If desired, these post scan variables can be changed without repeating the scan phase.

9.3.1 oblimap_message_level_config

The message level of OBLIMAP is default set to zero. With level zero all warning and error messages will be provided, and a minimum of messages about the progress of OBLIMAP informs the user. A final message shows which resulting netcdf file has been created. For more extended OBLIMAP messaging this config variable can be put to 1, 2 or 3 for an increasing amount of messages. These will consider the amount of grid points which are involved in the mapping, if there are no contributing points found for certain points, and if a slower scan was required at certain points.

9.3.2 suppress_check_on_scan_parameters_config

With this option it is possible to suppress the messages which are generated by the check on the scan parameters. The scan parameter values in the header of the SID file are compared with those in the used config file, in order to prevent discrepancies between the scan parameter values during the production of the SID file and the scan parameter settings in the config file in case the scan mode is switched off. However, in an on-line coupling experiment the embedded oblimap routines in both mapping directions are used. In that case both SID file headers will be compared to the config file which leads inevitable to ambiguities for this check. So in that case it is better to omit this check and suppress the irrelevant warnings.

9.3.3 nearest_point_assignment_config

If this option is used each destination node obtains the value of the nearest projected source point, disregarding any other contribution. This method can be combined with masked mapping, in that case a destination node will be always invalid if the nearest projected point has an invalid mask. The nearest point assignment is a post scan alternative to the quadrant and radius interpolation method. Regardless which interpolation method has been used during the scan phase, the 'nearest point assignment' can always be used in the post scan phase. This option can be considered in case both grids have about the same resolution.

9.3.4 shepard_exponent_config

The value of the exponent e in *Reerink et al.* [2010] concerning the distance weighting in both the quadrant and radius interpolation method. The default value is 2 [See Sec. 2.3.1 in *Reerink et al.*, 2010; *Shepard*, 1968].

9.3.5 invalid_input_value_config ⋆

This config variable is an array of integers. The array index corresponds to the mapped field number. This config variable specifies for each field the value of the 'invalid value' in the input field. These field values will be omitted in the colour scaling in netcdf representing tools like ncview or in images which are created by

python scripts. But more important the mask for the masked mapping is shaped by the pattern of the invalid_input_value_config values.

9.3.6 invalid_output_value_config

This config variable is an array of integers. The array index corresponds to the mapped field number. This config variable specifies for each field the value of the 'invalid value' in the output field. These field values will be omitted in the colour scaling in netcdf representing tools like ncview or in images which are created by python scripts. The invalid_output_value_config = -9999 by default.

9.3.7 field_which_determines_invalid_value_mask_config

This config variable is an array of integers. The array index corresponds to the mapped field number. With this config variable the masked mapping can be switched on or off for each individual mapped field. In case the value for a certain field is zero or negative, this field will be mapped without using a mask. In case the value for a certain field equals the field number of this field (which is expected to be the most common situation for a masked mapping), this field will be masked mapped based on the invalid value pattern of the considered field itself, i.e. points with an invalid value do not participate in the mapping. However, it is also possible to conduct a masked mapping of a certain field which is based on the invalid value pattern of another mapped field, in that case the field number of this 'other' field has to be specified as the value of field_which_determines_invalid_value_mask_config of the considered mapped field. E.g. specifying field_which_determines_invalid_value_mask_config(2) = 1 means that the second field is masked mapped based on the invalid value pattern of the first field, where in this case the value of the invalid_input_value_config(2) determines the invalid value pattern in this first field.

9.3.8 invalid_value_mask_criterion_config

This config variable is an array of integers. The array index corresponds to the mapped field number. With this config variable the masking criterion can be set to option 1 or 2 for each field. With option 1 (default) the destination point obtains an invalid value in case the nearest projected departure point has an invalid value. With option 2 all invalid contributions will be ignored and as long there is any valid contribution available they will be used for the interpolation at the destination point. If no valid contribution is detected then the point obtains an invalid value.

9.3.9 gcm_record_range_config ⋆

Two numbers separated by a comma specify the GCM record range for which the mapping has to be performed. So 1,10 will just map the first ten GCM records of the specified fields. For example 5,5 will map just only GCM record 5, which will

be in the netcdf output file IM record 1. Analogue, 4,9 will map 6 GCM records: the fourth up to the ninth, which will show up as the IM record range 1-6 in the netcdf output file.

9.3.10 im_record_range_config ⋆

Two numbers separated by a comma specify the IM record range for which the mapping has to be performed. So 1,10 will just map the first ten IM records of the specified fields. For example 5,5 will map just only IM record 5, which will be in the netcdf output file GCM record 1. Analogue, 4,9 will map 6 IM records: the fourth up to the ninth, which will show up as the GCM record range 1-6 in the netcdf output file.

9.3.11 number_of_vertical_layers_config

In case one of the fields which has to be mapped is a spatial 3D field, this number_of_vertical_layers_config has to equal the number of vertical layers corresponding with this spatial 3D field. In case other spatial 2D variables are mapped simultaneously, they will be detected as such and they will be mapped as spatial 2D fields. Beside their spatial dimension, these fields are allowed to contain an additional unlimited time dimension. Usually spatial 2D fields are mapped, so default the number_of_vertical_layers_config = 1.

9.3.12 number_of_mapped_fields_config ⋆

Specifies the number of fields which should be mapped.

9.3.13 ignore_reading_pre_mapped_fields_config

This config variable is an array of logicals. The array index corresponds to the mapped field number. For each of the mapped fields, the option exists to ignore the field values of the pre-mapped input file for the areas which are not involved in the mapping. Instead the 'invalid value' is assigned to the area which is not involved in the mapping. This config variable enables the mapping of a variable which is not present in the pre-mapped file.

9.3.14 gcm_field_name_config ⋆

This config variable is an array of character strings. The array index corresponds to the mapped field number. This config variable contains the name of each GCM field.

9.3.15 gcm_field_unit_config ⋆

This config variable is an array of character strings. The array index corresponds to the mapped field number. This config variable contains the unit of each GCM field.

9.3.16 gcm_field_longname_config ⋆

This config variable is an array of character strings. The array index corresponds to the mapped field number. This config variable contains the long name description of each GCM field.

9.3.17 im_field_name_config ⋆

This config variable is an array of character strings. The array index corresponds to the mapped field number. This config variable contains the name of each IM field.

9.3.18 im_field_unit_config ⋆

This config variable is an array of character strings. The array index corresponds to the mapped field number. This config variable contains the unit of each IM field.

9.3.19 im_field_longname_config ⋆

This config variable is an array of character strings. The array index corresponds to the mapped field number. This config variable contains the long name description of each IM field.

9.3.20 prefabricated_im_grid_field_name_config

This config variable is an array of character strings. The array index corresponds to the mapped field number. This config variable contains the name of each prefabricated IM grid field. Note that it makes only sense to adjust the default prefabricated_im_grid_field_name_config(-2) = 'x' and prefabricated_im_grid_field_name_config(-4) = 'y', because only this two dimension variables are read if use_prefabricated_im_grid_coordinates_config = true.

9.3.21 field_factor_config

This config variable is an array of reals. The array index corresponds to the mapped field number. This config variable contains a conversion factor for each mapped field. Default all factors equal 1.

9.3.22 field_shift_config

This config variable is an array of reals. The array index corresponds to the mapped field number. This config variable contains a conversion shift for each mapped field. Default all shifts equal 0.

9.3.23 gcm_created_filename_config ⋆

The filename of the created GCM netcdf file.

9.3.24 im_created_filename_config ⋆

The filename of the created IM netcdf file.

9.3.25 reduce_dummy_dimensions_config

With this option dummy dimensions, i.e. dimensions which have size 1, are omitted by an external call from fortran to the nco command ncwa which does an dimension average over this dummy dimension. The ncwa command omits dimensions with size 1. To make this option work nco should be installed, if not the dummy dimension reduction just won't work, without effecting the OBLIMAP results, in that case the results will still contain the dummy dimensions.

9.3.26 use_double_instead_of_float_in_netcdf

With this config variable the netcdf files can be written in double precision (default false). Netcdf files in double precision are of course also doubled in size.

9.3.27 synchronize_netcdf_writing

With this config variable the netcdf writing can be synchronized after each record by using synchronize_netcdf_writing = true (default). Which is an advantage in case the program is aborted because all fields which are calculated until then are written. But it can be significantly slower.

9.3.28 protect_file_overwriting_config

With this config option one can prevent that existing results are overwritten without a program warning and stop.

9.3.29 enable_color_messaging_in_terminal

With this config variable the coloring of OBLIMAP's error and warning messages in the terminal can be switched on by enable_color_messaging_in_terminal = true (default). For platforms which do not support terminal colors it can be switched off to avoid unreadable terminal output.

9.4 Specifying the range of each config variable

In the configuration files (config files) several Fortran NAMELIST-options can be specified. It is strongly recommended to use a config file and to specify all the config variables for your experiment. However, all config variables are initialised by a default value in the src/oblimap_configuration_module.f90 in case no config file is used or in case a config variable is omitted in the config file. Examples of config files can be found in the config-files/oblimap/ directory.

The order of the config variables in the config file doesn't matter. If by accident a config variable is specified more than once, the lowest listed value in the config file is taken.

Each config variable and its useful range is listed in Fig. 3. The Maximum Number of Fields (MNF) which can be mapped simultaneously is 100 by default, but can be increased to any positive integer in src/oblimap_configuration_module.f90 before (re)compiling the OBLIMAP code.

10 BUG fixes for OBLIMAP's first release

Two BUG's have been fixed for the oblique Lambert equal-area projection in OBLIMAP's first release. *Snyder* [1987]'s polar aspect formula's for the stereographic and the Lambert equal-area projection are added for the case they are used in combination with the ellipsoid. The projections which are taken from *Snyder* [1987, p. 154-163 and p.182-190] have been verified with the examples as provided by *Snyder* [1987] at p. 312-319 and p. 332-337.

11 Further OBLIMAP developments

OBLIMAP 2.0 has been developed as part of the ICEDYN-package for ice models. This external oblimap-package contains the oblimap core, which has been tested intensively. Beside the oblimap core other oblimap extensions have been developed as part of the ICEDYN-package. In one example a flow line model is embedded into a large scale 2D or 3D model. This includes the mapping on a flow line curve from the large scale model to the flow line model and vice versa. Further it includes the determination of a flow line profile of the curve, based on ice fluxes or ice thickness or a combination of them in a high flexible user design. In another example the low resolution atmosphere - ocean model CLIMBER 2.4 is embedded in one of ICEDYN's ice models by embedded oblimap coupling.

OBLIMAP is a light weight package which easily installs and runs on a laptop. The OBLIMAP 2.0 release doesn't make use of any MPI parallel implementation. However, a parallel MPI implementation using an effective domain decomposition is desirable for the full and fast scan method. A successful start of this development path has been made at the Polar HPC Hackathon 2016 at the XSEDE conference in

```
gcm_input_filename_config                       = any CHARACTER string (maximum of 256 characters)
im_input_filename_config                        = any CHARACTER string (maximum of 256 characters)
NLON_config                                     = any positive INTEGER
NLAT_config                                     = any positive INTEGER
NX_config                                       = any positive INTEGER
NY_config                                       = any positive INTEGER
dx_config                                       = any positive REAL
dy_config                                       = any positive REAL
choice_projection_method_config                 = 'oblique_stereographic_projection'
                                                  'oblique_stereographic_projection_snyder'
                                                  'oblique_stereographic_projection_ellipsoid_snyder'
                                                  'oblique_lambert_equal-area_projection_snyder'
                                                  'oblique_lambert_equal-area_projection_ellipsoid_snyder'
                                                  'rotation_projection'
earth_radius_config                             = any positive REAL (meter)
ellipsoid_semi_major_axis_config                = any positive REAL (meter)
ellipsoid_eccentricity_config                   = any REAL between 0 and 1
lambda_M_config                                 = any REAL between   0 and 360 degrees
phi_M_config                                    = any REAL between -90 and  90 degrees
alpha_stereographic_config                      = any REAL between   0 and  90 degrees
theta_rotation_projection_config                = any REAL between   0 and 360 degrees
shift_x_coordinate_rotation_projection_config   = any REAL (meter)
shift_y_coordinate_rotation_projection_config   = any REAL (meter)
enable_shift_im_grid_config                     = TRUE or FALSE
shift_x_coordinate_im_grid_config               = any REAL (meter)
shift_y_coordinate_im_grid_config               = any REAL (meter)
alternative_lambda_for_center_im_grid_config    = any REAL (degrees)
alternative_phi_for_center_im_grid_config       = any REAL (degrees)
unit_conversion_x_ax_config                     = any REAL
unit_conversion_y_ax_config                     = any REAL
use_prefabricated_im_grid_coordinates_config    = TRUE or FALSE
prefabricated_im_grid_filename_config           = any CHARACTER string (maximum of 256 characters)
scanning_mode_config                            = TRUE or FALSE
level_of_automatic_oblimap_scanning_config      = INTEGER in the range 0 to 5
data_set_is_cyclic_in_longitude_config          = TRUE or FALSE
choice_quadrant_method_config                   = TRUE or FALSE
R_search_interpolation_config                   = any positive REAL (meter)
scan_search_block_size_config                   = any INTEGER starting from -3 (where -3, -2 and -1 are special cases, -3 is default)
scan_search_block_size_step_config              = any positive INTEGER starting from 2 (with 2 the recommended default)
vincenty_method_for_ellipsoid_config            = TRUE or FALSE
sid_filename_config                             = any CHARACTER string (maximum of 256 characters)
backward_sid_filename_config                    = any CHARACTER string (maximum of 256 characters)
oblimap_allocate_factor_config                  = any positive REAL
oblimap_message_level_config                    = INTEGER in the range 1 to 3
suppress_check_on_scan_parameters_config        = TRUE or FALSE
nearest_point_assignment_config                 = TRUE or FALSE
shepard_exponent_config                         = any positive REAL, usually 2
invalid_input_value_config                      = DIMENSION(  1:MNF): array with:  any REAL
invalid_output_value_config                     = DIMENSION(  1:MNF): array with:  any REAL
field_which_determines_invalid_value_mask_config = DIMENSION(  1:MNF): array with:  any INTEGER (if > 1, the field is masked mapped)
invalid_value_mask_criterion_config             = DIMENSION(  1:MNF): array with:  INTEGER in the range 1 to 2
gcm_record_range_config                         = DIMENSION(2): positive INTEGER's:
                                                   format: starting desired record number, ending desired record number
im_record_range_config                          = DIMENSION(2): positive INTEGER's:
                                                   format: starting desired record number, ending desired record number
number_of_vertical_layers_config                = any positive INTEGER (default 1)
number_of_mapped_fields_config                  = any positive INTEGER, i.e. up to MNF = 100, but MNF can be increased in
                                                   oblimap_configuration_module.f90
ignore_reading_pre_mapped_fields_config         = DIMENSION(MND:MNF): array with:  TRUE or FALSE
gcm_field_name_config                           = DIMENSION(MND:MNF): array with:  any CHARACTER string (maximum of 128 characters)
gcm_field_unit_config                           = DIMENSION(MND:MNF): array with:  any CHARACTER string (maximum of 128 characters)
gcm_field_longname_config                       = DIMENSION(MND:MNF): array with:  any CHARACTER string (maximum of 256 characters)
im_field_name_config                            = DIMENSION(MND:MNF): array with:  any CHARACTER string (maximum of 128 characters)
im_field_unit_config                            = DIMENSION(MND:MNF): array with:  any CHARACTER string (maximum of 128 characters)
im_field_longname_config                        = DIMENSION(MND:MNF): array with:  any CHARACTER string (maximum of 256 characters)
prefabricated_im_grid_field_name_config         = DIMENSION(MND:MNF): array with:  any CHARACTER string (maximum of 128 characters)
field_factor_config                             = DIMENSION(  0:MNF): array with:  any REAL
field_shift_config                              = DIMENSION(  0:MNF): array with:  any REAL
gcm_created_filename_config                     = any CHARACTER string (maximum of 256 characters)
im_created_filename_config                      = any CHARACTER string (maximum of 256 characters)
reduce_dummy_dimensions_config                  = TRUE or FALSE
use_double_instead_of_float_in_netcdf           = TRUE or FALSE
synchronize_netcdf_writing                      = TRUE or FALSE
protect_file_overwriting_config                 = TRUE or FALSE
enable_color_messaging_in_terminal              = TRUE or FALSE
```

Figure 3: *Listing for each config variable the permitted options or range of valid options.*

Miami. A well scalable parallel domain decomposition has been implemented for the scan phase, the results remain bitwise identical for a changing number of processors. In particular in case future applications include on-line coupling of adaptive GCM and/or IM grids, a parallel implementation of this scan phase is required in order to minimize the scan time each time one of the grids change.

12 licence OBLIMAP 2.0

13 Code availability

The source code of OBLIMAP can be downloaded from the OBLIMAP svn repository through

svn checkout https://svn.science.uu.nl/repos/project.oblimap

or by a git checkout from OBLIMAP's Github through

git clone https://github.com/oblimap/oblimap-2.0

14 Feed-back

Feed-back on this OBLIMAP User Guide and on the OBLIMAP-package will be appreciated, and can be sent to tjreerink@gmail.com. Update messages might be low frequent published on the OBLIMAP website: http://oblimap.wordpress.com/.

References

Karney, C. F. F. (2012), Algorithms for geodesics, *Journal of Geodesy*, *87*(1), 43–55, doi:10.1007/s00190-012-0578-z.

Reerink, T. J., M. A. Kliphuis, and R. S. W. van de Wal (2010), Mapping technique of climate fields between gcm's and ice models, *Geoscientific Model Development*, *3*(1), 13–41, http://www.geosci–model–dev.net/3/13/2010/, doi:10.5194/gmd-3-13-2010.

Reerink, T. J., W. J. van de Berg, and R. S. W. van de Wal (2016), Oblimap 2.0: a fast climate model–ice sheet model coupler including online embeddable mapping routines, *Geoscientific Model Development*, *9*(11), 4111–4132, http://www.geosci–model–dev.net/9/4111/2016/, doi:10.5194/gmd-9-4111-2016.

Shepard, D. (1968), A two-dimensional interpolation function for irregularly-spaced data, *Proceedings-1968 ACM National Conference*, pp. 517–524.

Snyder, J. P. (1987), Map projections: A working manual, *Tech. rep.*, USGS, http://pubs.er.usgs.gov/usgspubs/pp/pp1395.

Vincenty, T. (1975), Geodetic inverse solution between antipodal points., *Tech. rep.*, Richard Rapp Geodetic Science Ohio State University, http://geographiclib.sf.net/geodesic-papers/vincenty75b.pdf.

Vincenty, T. (1975a), Direct and inverse solutions of geodesics on the ellipsoid with application of nested equations., *Surv Rev*, (23(176)), 88–93 [addendum: Surv Rev 23(180):294 (1976)].