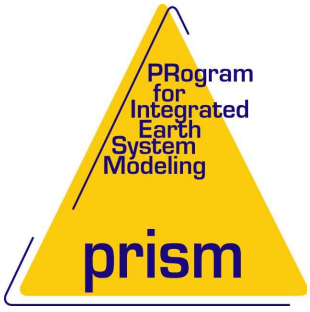
PRISM

Project for Integrated Earth System Modelling

An Infrastructure Project for Climate Research in Europe
funded by the European Commission
under Contract EVR1-CT2001-40012



Using OpenDX to Visualise PRISM Data

Edited by:
Camiel Severijns and Gabriella de Martino

PRISM-Report Series-18

1st Edition

(last change: December 17, 2004)

How to get assistance?

The individual work packages of the PRISM project can be contacted as listed below.
PRISM publications can be download from the WWW server of the PRISM project under the URL: <http://prism.enes.org/Results/Documents/>

Phone Numbers and Electronic Mail Adresses

Electronic mail adresses of the individual work packages are composed as follows :
prism_ *work package* @prism.enes.org

Name	**Phone**	***PRISM Work Package***
Camiel Severijns		*wp4a*
Gabriella de Martino		*wp4a*

Contents

List of Figures

List of Tables

Chapter 1

Using OpenDX to Visualise PRISM Data

1.1 Introduction

This document describes the modules that have been developed to enable the OpenDX tool to visualize data in the PRISM data format NetCDF(CF).

OpenDX is an open source high-end visualization tool that can be used to visualize simple data sets as well as to analyze complex, time-dependent data from disparate sources. It enables users to examine their data interactively via a graphical user interface. In addition, it provides the means to build interactive visualization applications. OpenDX provides functional modules to visualize data in many ways, to transform the data, and to interact with users and other applications. An application developed in OpenDX consists of a network of interconnected modules. Parts of such networks can be stored as macros for reuse. Completely new modules can be added as well by means of the module builder.

The standard PRISM data format, NetCDF CF-1.0, is not supported by the standard distribution of OpenDX. Therefore, a set of modules and macros has been developed that enables OpenDX to read files in this format. These modules and macros are described in section 1.2. This is followed with references to further information about OpenDX in section 1.3. The installation procedure for OpenDX and the additional modules is described in 1.4. The last section 1.5 describes the limitations of the current implementation of the modules.

1.2 NetCDF CF-1.0 Import Modules

Table 1.1 lists all PRISM specific OpenDX modules. These modules appear in the "Import and Export" category in the visual program editor (VPE) of OpenDX. You should read the section on the OpenDX data model in the OpenDX users guide before using the `NcCFReadComponent` and `NcCFCombine` modules. The modules are described in the following sections. Each section starts with a description of the module's functionality. This is followed by two tables describing the inputs and outputs of the module.

1.2.1 `NcCFCombine`

This module combines its inputs to a field or series of fields. All inputs should be arrays or series of arrays. In the latter case, the series must all have the same length. The arrays of the data, position and invalid inputs must all have the same number of elements. The array provided to the connection input should only refer to positions that are contained in the positions array.

Module	Description
`NcCFCombine`	Combines separate components output by the `NcCFReadComponent` module into a OpenDX field or series object
`NcCFFileInfo`	Retrieves the value of several standard file global attributes from a NetCDF CF-1.0 file
`NcCFInvertInZ`	Swaps the (scalar) data and the Z coordinates of a field with three dimensional positions
`NcCFLabels`	Uses the value of several standard variable attributes to create a set of text strings that can be used as captions and coordinate labels in plots
`NcCFListVariables`	Lists variables contained in a NetCDF CF-1.0 file
`NcCFReadComponent`	Uses the data from a variable in a NetCDF CF-1.0 file to create OpenDX data, position, connection, and invalid positions component objects
`NcCFRead`	Is a macro that combines the `NcCFReadComponent` and `NcCFCombine` modules to read common types of variables
`NcCFReplaceZ`	Replaces the Z coordinates in the position component of a field with the data component from another field
`NcCFSigmaCoordinate`	Computes mid-layer depths from layer thickness data. This module is useful when only layer thickness data is available from an isopycnic coordinate ocean model
`NcCFVariableInfo`	Provides meta information about a variable

Table 1.1: PRISM specifi c OpenDX modules appearing in the Import and Export category.

Input	Type	Description	Required	Default
`values`	array or series	The array(s) that will be the data component of the field(s)	Yes	-
`positions`	array or series	The array(s) containing the position data. The number of positions should be equal to the number of data values	Yes	-
`invalid`	array or series	The array(s) indicating invalid positions in the data. The number of invalid positions should be equal to the number of data values	No	-
`connections`	array or series	A list of connections between data positions. All references to positions should be valid. In addition, the element type of the connections should match the length of the position vectors. These requirements are not checked by the module	No	-

Output	Type	Description
`output`	field or series	A field or series of fields constructed from the components that were provided to the module

1.2.2 `NcCFFileInfo`

This module retrieves the value of the (optional) file global attributes as defined by the NetCDF CF-1.0 conventions. When an attribute is not defined in the file the corresponding output is set to the value “Unknown”. The module fails when the file does not exist or is not a NetCDF file.

Input	Type	Description	Required	Default
`filename`	string	The path to the file from which the information is to be retrieved	Yes	-

Output	Type	Description
`conventions`	string	The value of the global attribute “Conventions” or “Unknown”
`title`	string	The value of the global attribute “title” or “Unknown”
`institution`	string	The value of the global attribute “institution” or “Unknown”
`source`	string	The value of the global attribute “source” or “Unknown”
`history`	string	The value of the global attribute “history” or “Unknown”
`references`	string	The value of the global attribute “references” or “Unknown”
`comments`	string	The value of the global attribute “comments” or “Unknown”

1.2.3 `NcCFInvertInZ`

This macro converts a field of the form $f(x, y, z)$ to a field of the form $z(x, y, f)$. This is achieved without doing the actual inversion by exchanging the content of the scalar data component with the values of the third component of the position vectors. Only three dimensional scalar fields are modified. Other kinds of fields are passed without modification.

Input	Type	Description	Required	Default
`field`	field	The field to be inverted	Yes	-

Output	Type	Description
`output`	field	The input field with data and Z coordinates swapped

1.2.4 `NcCFLabels`

This module creates labels for the data and coordinate system of a variable in a NetCDF CF-1.0 file. The labels consist of the value of the `long_name` or `standard_name` attribute or the variable’s name (in this order) followed by the unit in which the data of coordinate is expressed enclosed by square brackets. The outputs of this module can be used as a caption in a plot or as labels of coordinate axes.

Input	Type	Description	Required	Default
`filename`	string	The path to a NetCDF file	Yes	-
`variablename`	string	The name of the variable in the file	Yes	-

Output	Type	Description
`variable`	string	A description the variable
`axislabels`	string list	The labels for the axes of the coordinate system

1.2.5 `NcCFListVariables`

This module creates a list of variables contained in a NetCDF CF-1.0 file. The variables included in the list can be constrained by kind and coordinate system. The kinds of variables correspond to those described in the NetCDF CF-1.0 conventions. This module is typically used in interactive programs, that are capable of visualizing only specific kinds of data, to present a list of variables from which the user can select one.

Input	Type	Description	Required	Default
`filename`	string	The path to the file to be examined	Yes	-
`kind`	string	The kind of variables to be included in the search. This input can have one of the following values: "All variables", "Plain variables", "Coordinate variables", "Auxiliary coordinate variables", "Multidimensional coordinate variables", "Boundary variables", "Grid mapping variables", and "Region variables". These values correspond to the kind of variables as defined by the NetCDF CF-1.0 conventions	No	"Plain variables"
`coordinates`	string	The kind of (spatial) coordinate system that the variables should have. This input can have one of the following values: "Any", "X", "Y", "Z", "XY", "XZ", "YZ", "XYZ"	No	"Any"

Output	Type	Description
`output`	string list	The names of all variables satisfying the selection criteria

1.2.6 `NcCFReadComponent`

This module reads the data for one of the components of a field or series of fields. The `start`, `end` and `increment` inputs are only meaningful when the variable is time dependent. Otherwise, these inputs are ignored. Note that time steps are counted starting from one.

According to the CF-1.0 conventions a variable's coordinate system can be described using coordinate variables as well as auxiliary coordinate variables. The `preference` input determines which coordinate variables are used to compute the positions component of a field when both types of coordinates are available.

The outputs of several instances of this module can be combined into a field (or series of fields) using the `NcCFCombine` module. The `NcCFRead` macro is such a combination that creates a complete field from a NetCDF variable and its associated information in one step.

Input	Type	Description	Required	Default
`filename`	string	The path to a NetCDF file	Yes	-
`variablename`	string	The name of a variable in the file	Yes	-
`what`	string	The component for which the data should be retrieved. Valid values are “Values”, “Invalid positions”, “Positions”, “Connections”	No	“Values”
`start`	integer	The first time step for which data should be read	No	First time step
`end`	integer	The last time step for which data should be read	No	Last time step
`increment`	integer	The time step interval at which data should be read	No	1
`preference`	string	The preferred coordinate system type. Valid values are “Auxiliary” and “Coordinates”	No	“Auxiliary”

Output	Type	Description
`output`	array or series	The data read. It is an array when the variable is time independent. Otherwise, this output is a series containing array objects

1.2.7 `NcCFRead`

This macro combines several instances of the `NcCFReadComponent` module with one instance of the `NcCFCombine` module to create a field or series of fields from a variable in a NetCDF CF-1.0 file. It handles most cases except for variables with irregular connections. In the case of irregular connections, the read_connections input should be set to 0. OpenDX’s `Connect` module can be used to compute connections using a triangulation algorithm.

Input	Type	Description	Required	Default
`filename`	string	The path to a NetCDF file	Yes	-
`variablename`	string	The name of a variable in the file	Yes	-
`start`	integer	The time step number of the first frame to read	No	First time step
`end`	integer	The time step number of the last frame to read	No	Last time step
`increment`	integer	The interval used to select frames from a time dependent variable	No	1
`read_connections`	integer	When this input is set to 1 the macro adds the connections components to the output but when this input is 0, the connections are not included	No	1
`preference`	string	The preferred kind of coordinate system. Valid values are “Auxiliary” and “Coordinates”	No	“Auxiliary”

Output	Type	Description
`output`	field or series	An OpenDX object constructed from the specified variable

1.2.8 NcCFReplaceZ

This macro replaces the Z coordinate of the position vectors of one field by the (scalar) data component of another field. It can be used to create fields that describe the relation between two quantities as a function of longitude and latitude without interpolation. Note that both input fields must have the same amount of data points.

Input	Type	Description	Required	Default
src	field	The field that contains the new Z coordinate values in its data component. The data component must be scalar	Yes	-
dst	field	The field in which the Z coordinates should be replaced	Yes	-

Output	Type	Description
output	field	The destination field with modified Z coordinates in it positions component

1.2.9 NcCFSigmaCoordinate

This module computes the mid-layer depth from layer thickness data produced by isopycnic ocean models as follows: let dp_i be the thickness and p_i the mid-layer depth of layer i, then $p_k = \sum_{i=1}^{k} dp_i - dp_k/2$. The output of this module can be passed to the NcCFReplaceZ macro to replace the z coordinate in positions data read using the NcCFReadComponent module. In this manner one can easily convert a field with isopycnal data to one with spatial data without interpolation.

Input	Type	Description	Required	Default
thicknesses	field or series	The layer thicknesses. This input must be a (series of) regularly connected field(s)	Yes	-

Output	Type	Description
depths	field or series	The mid-layer depths of the sigma layers at each position

1.2.10 NcCFVariableInfo

This module retrieves meta information about a variable in a NetCDF file. This meta information consists of all attributes defined by the NetCDF CF-1.0 conventions and information about the structure of the data grid. The long_name, standard_name and units attributes can be used to creating custom labels in a plot in the case that the output of the NcCFLabels is not satisfactory. The grid structure information can be used to select the proper type of visualization for the variable's data.

Input	Type	Description	Required	Default
filename	string	The path to a NetCDF file	Yes	-
variablename	string	The name of a variable in the file	Yes	-

Output	Type	Description
timesteps	integer	The number of time steps. This value greater than zero if the variable contains time dependent data. Otherwise this output is zero
ndimensions	integer	The number of spatial dimensions of the variables
long_name	string	The value of the long_name attribute. An empty string is returned when the attribute is not define for the variable
standard_name	string	The value of the standard_name attribute. An empty string is returned when the attribute is not define for the variable
units	string	The value of the units attribute. The string "Unknown"is returned when the attribute is not define for the variable
title	string	The value of the title attribute. An empty string is returned when the attribute is not define for the variable
institution	string	The value of the institution attribute. The string "Unknown"is returned when the attribute is not define for the variable
source	string	The value of the source attribute. The string "Unknown"is returned when the attribute is not define for the variable
history	string	The value of the history attribute. The string "Unknown"is returned when the attribute is not define for the variable
references	string	The value of the references attribute. The string "Unknown"is returned when the attribute is not define for the variable
comments	string	The value of the comments attribute. The string "Unknown"is returned when the attribute is not define for the variable
coordinates	string	Describes the (spatial) coordinate system of the variable. This output contains the characters X, Y and Z to indicate the presence of the corresponding coordinate axis

1.3 References

The OpenDX website, `http://www.opendx.org`, provides on-line access to all user and programmer documentation and mailing lists, and has links to additional macro packages. The OpenDX quick start guide describes how you can start using OpenDX. Further information about using OpenDX and its programming language can be found in the Users Guide. The standard modules are described in the Reference Guide. The highlights section of the OpenDX site might give you some ideas for visualizing your data. Its bookstore provides a list of books about OpenDX and visualization.

A macro package that is of interest to climate models is the Earth and Space Sciences package available at Cornell Theory Center (CTC, `http://www.tc.cornell.edu`). The CTC also has an on-line introduction to OpenDX.

The Department of Oceanography at Dalhousie University provides a tutorial showing about advanced visualization with OpenDX (`http://www.phys.ocean.dal.ca/docs/DX_tutorial.html`).

VISUALIZATION and IMAGERY SOLUTIONS provides commercial support.

1.4 Installing OpenDX and NetCDF CF-1.0 Modules

In this section we describe how OpenDX and the NcCFModules can be installed. We assume that you have downloaded the source code packages of both of these from the PRISM site, that you are using a Linux PC, and that the NetCDF and udunits libraries have been installed before in the (non-standard) directories `/usr/local/netcdf` and `/usr/local/udunits`. In this case you can configure, compile and

install OpenDX in your home directory by entering the following commands (assuming that you are using `bash`)

```
export CPPFLAGS="-I/usr/local/netcdf/include"
export LDFLAGS="-L/usr/local/netcdf/lib"
./configure --prefix=\$HOME/dx
make
make install
```

Note that when you are using `tcsh` instead of `bash` you should replace `export` by `setenv` and the equal sign by a space in this and the following command examples. If the NetCDF and udunits libraries are installed in a standard location the `export` command lines can be skipped altogether. Before you can test the installation of OpenDX by running it you need to include the directory `$HOME/dx/bin` in the search path of the shell

```
export PATH=\$HOME/dx/bin:$PATH
```

To start the visual program editor of OpenDX enter the command

```
dx -edit
```

Once OpenDX is installed, you are ready to configure, compile and install the NcCFModules. You can do this by entering the following commands

```
export CPPFLAGS="-I/usr/local/netcdf/include -I/usr/local/udunits/include"
export LDFLAGS="-L/usr/local/netcdf/lib -I/usr/local/udunits/lib"
./configure --with-dxroot=\$HOME/dx
make
make install
```

As before you can skip the `export` command lines when the NetCDF and udunits libraries are installed in a standard location. Before you can use the new modules in OpenDX you should set some environment variables in your `.bashrc` file

```
dxroot=$HOME/dx
export DXMACROS=.:$dxroot/dx/macros
export DXMDF=$dxroot/dx/modules_linux/NcCFModules.mdf
export DXMODULES=.:$dxroot/dx/modules_linux
unset dxroot
```

or in case of `tcsh` you should add the following to your `.tcshrc` file

```
set dxroot=$HOME/dx
setenv DXMACROS .:$dxroot/dx/macros
setenv DXMDF $dxroot/dx/modules_linux/NcCFModules.mdf
setenv DXMODULES .:$dxroot/dx/modules_linux
unset dxroot
```

Note that on non-Linux systems the directory names containing `_linux` should be replaced by the names corresponding to the operating system you are using. You can determine which names to use by listing the content of the `$HOME/dx/dx` directory when OpenDX is installed in `$HOME/dx`. You can test whether the modules have been installed successfully by starting OpenDX with

```
dx -edit
```

The NcCFModules should be listed in the “Import and Export” category if installation was successful.

1.5 Limitations

The visualization of climate model data using the advanced capabilities of OpenDX requires large amounts of computer resources. This implies that for optimal performance all data is best stored on local file systems and available computing power should be allocated to the rendering of the data as much as possible.

The latter is the reason that COCO has not been integrated into OpenDX although this is technically feasible. Of course files produced by COCO can be imported into OpenDX because they conform to the NetCDF CF-1.0 conventions. Since the NetCDF CF-1.0 modules access data files only via the NetCDF library, it should be possible to access remote files by linking the modules to the DODS netCDF client library. However, this has not been tested yet.

The NetCDF CF-1.0 modules for OpenDX are able to read all NetCDF files that conform to the NetCDF CF-1.0 conventions. However, in the current implementation the cell boundary information is ignored. These cell boundaries provide information about the connectivity of the grid. In the case of regularly connected grids - a common situation in data produced by climate models - the cell boundary information is redundant. In the case of irregularly connected grids the `Connect` module of OpenDX can be used to compute a triangulation of grid. An advantage of using the information on cell boundaries is that it most likely provides a faster way of determining the grid connectivity.