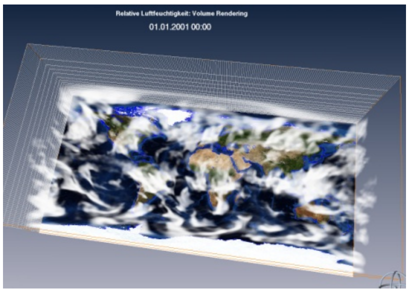
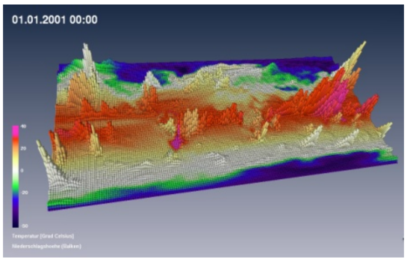
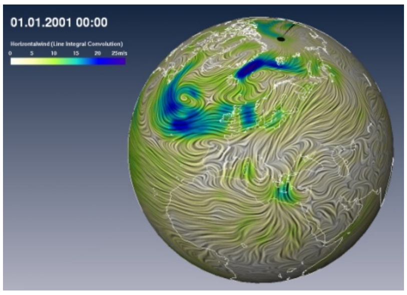
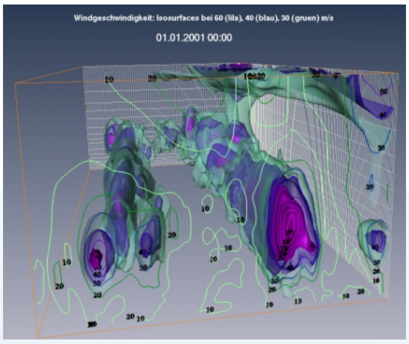


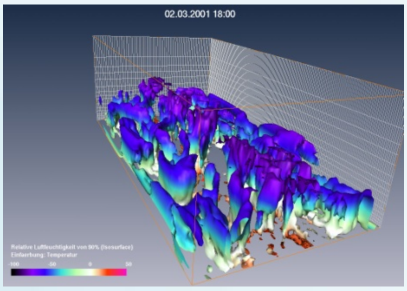


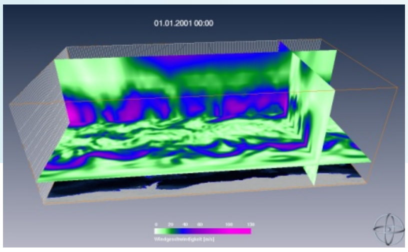



Windgeschwindigkeit: Isosurfaces bei 60 (lila), 40 (blau), 30 (gruen) m/s
01.01.2001 00:00

Tutorial

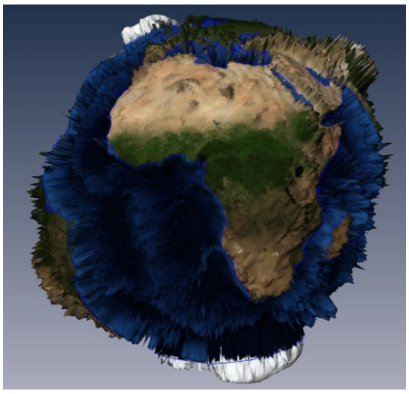
Interactive 3D Visualization in Earth System Research with Avizo Green 8.1





Michael Böttinger
Karin Meier-Fleischer
DKRZ

Carmen Ulmen
CliSAP



Version: 1.9, 08/09/2014

Contact: **Michael Böttinger**

Exzellenzcluster "Climate System
Analysis and Prediction (CliSAP)",
Universität Hamburg
Grindelberg 5
D-20144 Hamburg
Germany

Email: office.clisap(at)zmaw.de
http://www.clisap.de

Deutsches Klimarechenzentrum (DKRZ)
Bundesstrasse 45a
D-20146 Hamburg
Germany

Email: boettinger(at)dkrz.de
http://www.dkrz.de/

Contents

1. About *Avizo* and this Tutorial

1.1. Introduction

The 3D data visualization software ***Amira*** was developed primarily for the visualization of medical, chemical or biological data. A typical application: on the basis of computer tomography data, physicians can interactively extract and render 3D representations of skin, bones or tumors.

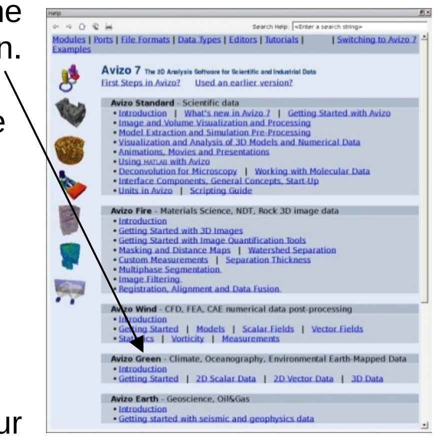
In order to better deal with application area specific requirements, the system was later divided into two branches: a life sciences product line, *Amira,* and a line more focused on physical sciences and industrial applications called *Avizo*.

Starting in 2007, *Avizo* was extended for applications in earth sciences – initiated and coordinated by the German Climate Computing Centre (DKRZ). As a result, an extension package called ***Avizo Green*** (or formerly ***Amira Climate Viz***) is now available for the visualization of observed or simulated data in meteorology, oceanography or other earth system sciences. Among other new features, the Green edition includes the capability to directly read NetCDF CF data files and interactively visualize the data - with different map projections, using different visualization methods and, in combination with the geographic context, the earth topography and texture or continent outlines.

1.2. Contents of this Document

This document specifically describes the different visualization tools useful for time-dependent gridded climate model or satellite data available with *Avizo Green*.

You can also find a short version of this tutorial in the *Avizo* help section.


Help
Search Help: <Enter a search string>
Modules | Ports | File Formats | Data Types | Editors | Tutorials | | Switching to Avizo 7
Examples

Avizo 7 The 3D Analysis Software for Scientific and Industrial Data
First Steps in Avizo? Used an earlier version?

Avizo Standard - Scientific data
- Introduction | What's new in Avizo 7 | Getting Started with Avizo
- Image and Volume Visualization and Processing
- Model Extraction and Simulation Pre-Processing
- Visualization and Analysis of 3D Models and Numerical Data
- Animations, Movies and Presentations
- Using MATLAB with Avizo
- Deconvolution for Microscopy | Working with Molecular Data
- Interface Components, General Concepts, Start-Up
- Units in Avizo | Scripting Guide

Avizo Fire - Materials Science, NDT, Rock 3D image data
- Introduction
- Getting Started with 3D Images
- Getting Started with Image Quantification Tools
- Masking and Distance Maps | Watershed Separation
- Custom Measurements | Separation Thickness
- Multiphase Segmentation
- Image Filtering
- Registration, Alignment and Data Fusion

Avizo Wind - CFD, FEA, CAE numerical data post-processing
- Introduction
- Getting Started | Models | Scalar Fields | Vector Fields
- Statistics | Vorticity | Measurements

Avizo Green - Climate, Oceanography, Environmental Earth-Mapped Data
- Introduction
- Getting Started | 2D Scalar Data | 2D Vector Data | 3D Data

Avizo Earth - Geoscience, Oil&Gas
- Introduction
- Getting started with seismic and geophysics data

In contrast to the tutorial document available within the software, this document is completely updated and extended by several chapters:

- a section about data pre-processing with *cdo* and reading metadata with ncdump (chapter 2.2),
- a section about 3D vector field visualization (chapter 6)
- a section about producing mpeg1 films from your animations including camera paths (chapters 7).
- It also works with datasets with more time steps in order to produce more interesting animations.

1.3. Tutorial Data and other Prerequisites

Registered users of the DKRZ visualization server ***halo*** will find all tutorial data files and example projects in the folder:

/work/kv0653/Tutorial_AvizoGreen/

All atmosphere data sets used in the tutorial were acquired through the World Data Center for Climate (WDCC) Database. Please bear in mind that these data may be used freely for research only. Commercial use of the data is not allowed. Detailed information on the ***terms of use*** of the World Data Center for Climate (WDCC) data can be found at:

- http://cera-www.dkrz.de/WDCC/ui/docs/TermsOfUse.html

Additionally, some Atlantic Ocean datasets from MIT General Circulation Model runs performed by Prof. Detlef Stammer and Dr. Nuno Serra (Institute of Oceanography, University of Hamburg) are included. We thank Prof. Stammer for his authorization to use these data in this tutorial. Of course, commercial use is not allowed either. You can find further information on this ocean circulation model at:

- http://mitgcm.org/

Unlike with many other visualization applications, *Avizo* - as well as all other interactive 3D-Applications based on OpenGL - needs a 3D-graphics card on the system where the application is started. For DKRZ users, the visualization server ***halo*** - a powerful visualization cluster equipped with high-end-graphics cards - is available to use *Avizo* virtually on your own workstation. This is accomplished by means of *Remote-3D-Rendering* (in our case with *VirtualGL and TurboVNC*, see http://www.virtualgl.org/), a technique where the rendered content of the 3D window is continuously read back from the graphics card and sent to the client workstation.

More detailed information on the *DKRZ visualization server **halo*** can be found here:

- http://www.dkrz.de/Nutzerportal-en/doku/halo

2. Data Input and Pre-Processing

2.1. NetCDF file format and related software

One of the most important features of the Green edition is the *NetCDF* reader module, which allows for direct work with gridded simulation data stored in the *NetCDF* file format, or, more specifically, *NetCDF* files which follow the *NetCDF CF-1.0* metadata convention (“**C**limate **F**orecast”). Detailed information on *NetCDF* and the *NetCDF* library can be found at:

- http://www.unidata.ucar.edu/software/netcdf/index.html

For more information on the *CF* metadata convention see

- http://cf-pcmdi.llnl.gov/

Other file formats used by the community, such as WMO’s *GRIB* format, can also be read by the module, but in this case the data file is first automatically being converted to *NetCDF* (and has to be stored on the hard disk), before the data can be visualized.

Avizo uses the publicly available tool “*cdo*” (“Climate Data Operators”) for this implicit data conversion. This powerful tool is developed by the Max-Planck-Institute for Meteorology with the aim to simplify the processing and analysis of climate model data. With *cdo*, you can also convert data from IEG, EXTRA, SERVICE, or HDF5 file formats into NetCDF. Beneath file conversion, *cdo* offers many other important features such as regridding, arithmetic, statistics and so on.

Downloads, documentation, and access to the mailing lists of *cdo* (= Climate Data Operators) are available at:

- https://code.zmaw.de/projects/cdo
- https://code.zmaw.de/files/cdo/html/1.6.0/cdo.pdf

On DKRZ systems, *cdo* and NetCDF are available by default. When you work on other systems, it might be a good idea to install NetCDF and *cdo*.

There is another set of operators to manipulate NetCDF files which is called NetCDF Operators (nco). You might find its attribute editor (ncatted -a) especially useful in some cases. The nco documentation is available at:

- http://nco.sourceforge.net/#Definition
- http://nco.sourceforge.net/nco.pdf

2.2. NetCDF Metadata with ncdump

The NetCDF metadata convention being used with Avizo Green is *NetCDF CF-1.0* (CF = Climate Forecast). In order to display NetCDF metadata, you can use *ncdump*, either with the -h option, if you only want to explore the header with the dimensions, the variables, and the global attributes, or the -c option, which additionally includes the grid coordinates:

```
ncdump -h filename.nc
ncdump -c filename.nc
```

Questions:

- How many and which variables are included in the NetCDF file `ECHAM5_OM_A1B_2001_2D`?
- How many time steps are covered in `ECHAM5_OM_A1B_2001_2D`? Which time period do they represent exactly?
- In which unit is the data of the variable containing the vertically integrated water vapor of the atmosphere in `ECHAM5_OM_A1B_2001_2D`?
- Which cdo commands were used latest to process or produce `ECHAM5_OM_A1B_2001_2D`?
- How many vertical levels are available in `ECHAM5_OM_A1B_2001_2D` and `ECHAM5_OM_A1B_2001_3D` respectively?

Answers

2.3. *Data pre-processing with cdo*

In order to visualize climate model data with *Avizo*, you may first have to do some data processing. In this paragraph we list some *cdo* commands which we found very useful with respect to the pre-processing – depending on the climate model type and area of interest. For more information please refer to the *cdo* documentation mentioned above. Detailed examples of NetCDF pre-processing with *cdo* can be found in the annex of this tutorial (chapter 8).

- **NetCDF format**: Conversion from IEG, GRIB, or other climate data formats to the NetCDF file format is done by the *-f nc* option:

  ```
  cdo -f nc copy <inputfile> <outputfile.nc>
  ```

- **Missing values**: Missing values are used to mask out grid cells that should not be visualized, such as grid cells on land in case of ocean model data. Since *Avizo* cannot read *NaN* values (**N**ot **a** **N**umber), make sure that missing values are defined differently, e.g., by using a missing value of -9.e+33.

  Substitute an old by a new missing value:

  ```
  cdo setmissval,miss <inputfile> <outputfile>
  ```

  Declare a constant as missing value:

  ```
  cdo setctomiss,c <inputfile> <outputfile>
  ```

- **Relative time**: If your NetCDF file was stored with absolute time values ("day as %Y%m%d.%f"), you will need to convert it to relative time (e.g. "days since 1989-6-15 12:00" or "hours since 2001-01-01 0:00") by using the *–r* option. Otherwise, *Avizo* won't be able to annotate the date and/or time in the viewer window.

  ```
  cdo -r copy <inputfile> <outputfile
  ```

- **Geographic coordinates of global data**: Global data in geographic coordinates with longitudes in the range [0°; 360°] (standard output of the ECHAM model) should be transformed into the range [-180°; +180°] by using the selection operator `selindexbox` with the following parameters for the western and eastern longitudes:

  ```
  lon1 = (number of longitudes / 2) +1
  lon2 = number of longitudes / 2
  ```

  Example for the T63 grid:

  ```
  cdo selindexbox,97,96,1,96 <inputfile> <outputfile>
  ```

- **Hybrid model levels**: For the vertical axis of atmospheric models, *Avizo* supports height and pressure levels. If the vertical axis of your data is described by *hybrid* model levels (terrain following vertical coordinates at the bottom, pressure levels at the top, interpolated in between), the data must be interpolated onto HEIGHT [m] or PRESSURE [Pa] levels before reading it with *Avizo*. The number and values of those height or pressure levels can be defined explicitly (plevels, hlevels). During this process you may receive missing values (Grid cells with no data), e.g., in areas with mountains in case of atmospheric data.

  ```
  cdo ml2pl,plevels <inputfile> <outputfile>
  cdo ml2hl,hlevels <inputfile> <outputfile>
  ```

- **Levels of ocean data**: Ocean circulation models like MPI-OM, MITgcm, GECCO or ECOHAM provide output files with positive levels, which are understood as “depth in meters” in the oceanographers’ community. However, Avizo would interpret those levels as heights and would visualize the 3D data above-ground (in the atmosphere) rather than subsurface (in the ocean). Therefore, we have to change the sign of all ocean level data from positive to negative before loading the NetCDF files into Avizo.

  ```
  cdo setaxis,zaxis.txt <inputfile> <outputfile>
  ```

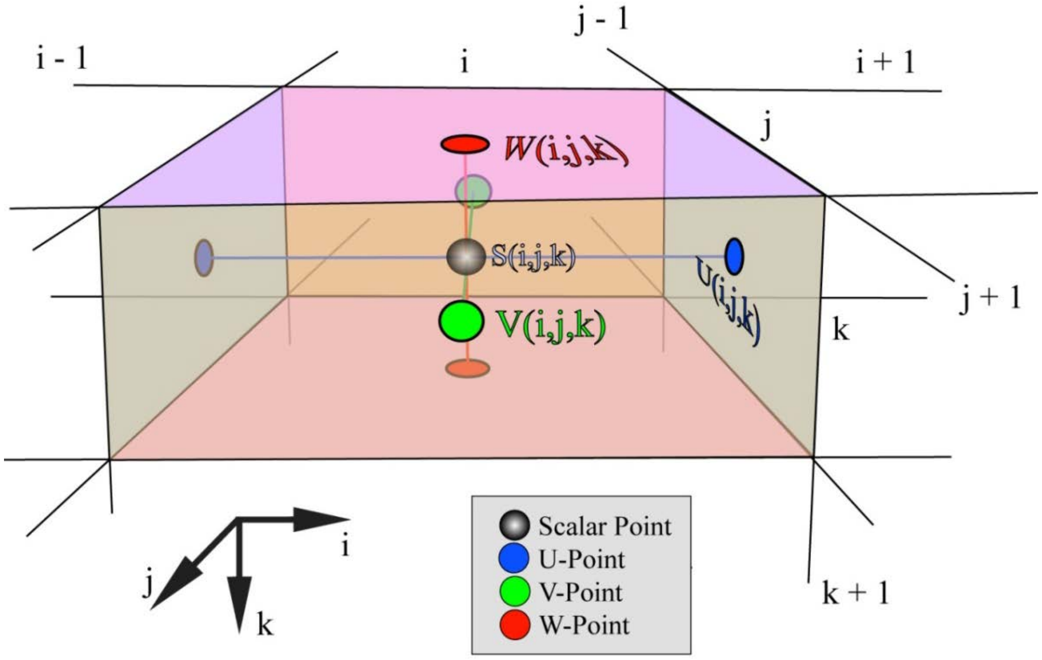
- **Time interpolation for smooth animation**: Depending on the type of data and the time interval the data has been stored with, the spatial patterns and peak values of some quantities can differ drastically from time step to time step. In order to achieve a smooth animation, a bilinear interpolation along the time axis might be useful. In addition, a time interpolation makes sense if your output variables differ in their time resolution (e.g. particle concentrations in a 1 hour resolution, wind: 6 hourly), but shall be visualized in the same animation. This can be done by the cdo operators `intntime` and `inttime`:

  ```
  cdo -intntime,n <inputfile> <outputfile>
  cdo -inttime,date,time,increment <inputfile> <outputfile>
  ```

Special treatment of vector components

- **Interpolation to scalar grid:**
  Often scalar output data is represented by values in the center of the grid cells, as shown in grey in the figure below, while, in contrast, vector

components are defined on the outer faces of the grid cells. In the example (Grid of the curvilinear MPI-OM Ocean Model) below, u is defined on the right face (blue), v on the front face (green) and w on the top face (red).



MPI-OM: Grid definition for scalar and vector quantities

For the representation of vector fields with *Avizo*, all three vector components of a grid cell need to be defined at the very same geographic coordinate. If your data does not fulfill this condition, you may want to interpolate the vector components onto the “scalar grid” (cell centers). This is imperative for the 2D display (combination of the horizontal wind components u and v only) as well as for the 3D display (combination of all wind components u, v and w).

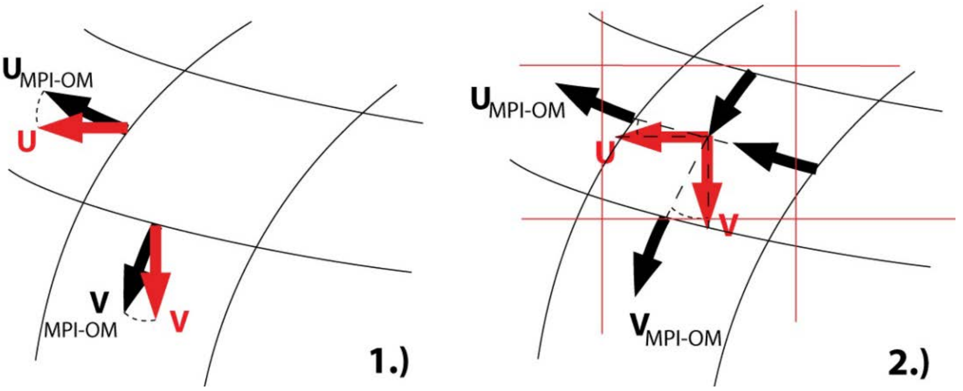
cdo offers several horizontal interpolation methods as well as some generic descriptions of regular or gaussian grids, which will often be sufficient for visualization purposes. Bilinear interpolation (*remapbil)* or distance weighted interpolation *(remapdis)* might be a good starting point, but keep in mind what the chosen method does with your data.

```
cdo remapbil,<scalarGrid> <inputfile> <outputfile>
```

For the interpolation of the vertical velocity component to the central point of your grid cells use the *intlevel* operator and the vertical coordinates of the according scalar grid:

```
cdo intlevel,levels <inputfile> <outputfile>
```

- **Rotation of horizontal velocity u, v:**
  For all *curvilinear* (e.g. MPI-OM, MITgcm) and *rotated* (e.g. REMO, CLM) model grids, a rotation of the horizontal velocity components has to be applied to turn the vectors from the local model grid onto the geographic grid, so that u gives the zonal component (W-E-direction) and v the meridional component (S-N-direction) of the velocity vector.



Rotation can be done by the cdo commands *rotuvb* and *mrotuvb* (the latter is not yet documented). *mrotuvb* combines the horizontal interpolation to the scalar grid with the rotation in one operator.

For REMO:

```
cdo rotuvb,u,v <inputfile> <outputfile>
```

where u and v are the vector components located in the inputfile.

For MPI-OM:

```
cdo mrotuvb <inputfile1> <inputfile2> <outputfile>
```

where the inputfiles should each contain only one variable – with the first vector component variable in inputfile1 and the second in inputfile2 (on two different grids).

- **Units of vertical velocity w**: If the vertical wind velocity component of the atmospheric model is only available as the pressure change per time in Pa/s (ECHAM5: variable “vertical velocity” *omega*, code 135), it is essential to compute the vertical wind speed in m/s in order to obtain the same unit as for the horizontal wind components. Only on that condition can the three wind components be combined to a resulting 3D spatial vector. The new cdo operator `vertwind` (also not yet documented) can help you do this. In your inputfile you need the following variables:

code	***description***
130	temperature [K]
133	specific humidity [kg/kg]
134	surface pressure [Pa]
135	vertical velocity [Pa/s]

```
cdo vertwind <inputfile> <outputfile>
```

- **Interpolation to rectilinear or regular grid:**
For 3D visualization packages like *Avizo*, it is computationally easier to work with rectilinear or regular grids compared with curvilinear grids. Additionally, several Avizo modules do not work with curvilinear grids yet, especially the vector modules. Therefore, interpolation onto rectilinear or regular grids is recommended – especially for vector components.

```
cdo remapbil,<rectilinearGrid> <inputfile> <outputfile>
```

See annex chapter 8 for detailed examples of NetCDF pre-processings.

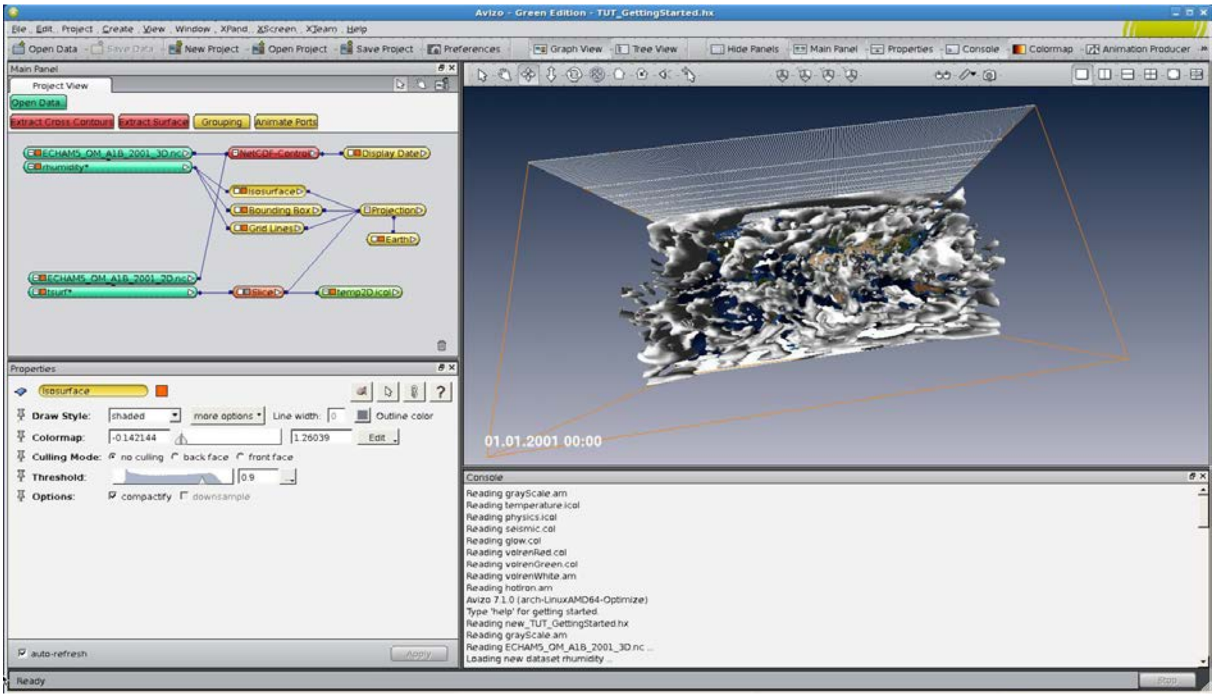
3. Avizo Green: Getting started

3.1. *Avizo's User Interface*

We assume that you have started a VirtualGL/TurboVNC session on the DKRZ visualization server *halo*. Open a *Terminal* window (right click on the empty desktop background and → ***Open Terminal)*** and type in *avizo8* (Avizo Green Version 8). When you run *avizo* for the first time, you will have to accept the "terms of use", otherwise the *Graphical User Interface* (GUI) of *Avizo* will open directly.

In order to get familiar with *Avizo's* GUI and to get in touch with the 3D navigation in the viewer window, we start by loading an existing example project:

- Click on ***File* → *Open Project*** (or on the *Open Project* button in the main toolbar) and load the project called **TUT_GettingStarted.hx**. This will take a few seconds.



Draw Style: shaded · more options · Line width: 0 · Outline color
Colormap: -0.142144 · 1.26039 · Edit
Culling Mode: ● no culling ○ back face ○ front face
Threshold: 0.9
Options: ☑ compactify ☐ downsample

```
Reading grayScale.am
Reading temperature.icol
Reading physics.icol
Reading seismic.col
Reading glow.col
Reading volrenRed.col
Reading volrenGreen.col
Reading volrenWhite.am
Reading hotIron.am
Avizo 7.1.0 (arch-LinuxAMD64-Optimize)
Type 'help' for getting started.
Reading new_TUT_GettingStarted.hx
Reading grayScale.am
Reading ECHAM5_OM_A1B_2001_3D.nc ...
Loading new dataset rhumidity ...
```

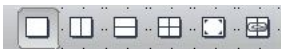
Avizo presents you four areas within one large window:

- In the **Viewer** area (or window) you see a 3D visualization of the relative humidity as an isosurface (for 90% humidity) from the ECHAM5/MPI-OM global model (IPCC-scenario A1B for the year 2001).
- Top left: The **Main Panel** shows the project with loaded data sets, colormaps, the Earth, Projection and some visualization modules and their data connections.
- Bottom left: In the **Properties** area you can set and vary the parameters of the module that is currently active in the Main Panel. For example, if you click on the module *Isosurface* in the Main Panel, its related properties are displayed in the Properties area.

- Bottom right: The **Console** gives feedback from the processes *Avizo* is currently performing and can also be used as a command line to interactively type in TCL-commands. You can switch the console window on and off by clicking on the Console button in the toolbar.

Help information can be found in a separated viewer by

a) typing help in the console,
b) clicking ***Help → User's Guide***,
c) clicking on ? Help in the main toolbar,
d) pressing the F1 button,
e) clicking the question tag ? in the Properties area. This will lead you to the help page concerning the active module (in the above figure e.g. help information of the Isosurface module).

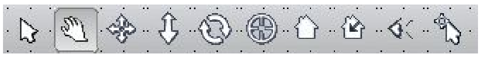


The main viewer can be divided into two visualization windows which are tiled vertically ◫ or horizontally ⊟ or could even be divided into four tiles⊞. With two or four tiled viewers you can either watch your data from different angles or different variables at the same time. Or you could use the tiles to show different time steps in parallel. To change the visibility of an object in all viewers by changing the visibility of an object in one viewer, you can use ⊟ .

The icon □ provides a full screen mode. In order to finish the full screen mode and get back to the original layout, you have to do a right-click in the viewer window and click on *FullScreen*.

3.2. Spatial Navigation

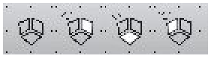
Get familiar with the navigation bar above the viewer. The different glyphs change between different navigation or interaction modes.



- With the Trackball ✋ active (default value), you may turn the visualization intuitively and explore the visible objects from all viewing angles. If “Spin animation” is activated: when pressing the left mouse button, accelerate the mouse (left button still pressed) a bit and release the button while it still moves, the objects in view will continuously keep turning. You can easily halt it by clicking again.
- With the Translate Tool ✥ activated, you can translate the objects up, down, or left and right respectively.
- When you move your mouse downwards with the Zoom tool ⇳ activated, you pull the objects to yourself, so that you zoom in. Moving your mouse upwards is

like pushing the figure away which results in zooming out. The Zoom tool does not react to mouse movements in left or right direction.

- By clicking the Rotate tool, your objects rotate a fixed angle clockwise around the normal axis of your screen.
- The Seek tool allows you to click on a target point in order to tell *Avizo* to fly to that point and bring the target into focus.
- Use the Home tool to get back to a certain view you had defined earlier with the Set Home tool.
- You might use the Set Home tool to define a certain view you like in order to quickly come back to it later.
- With this button you can switch the virtual camera between orthographic mode and perspective mode – similar to a telephoto and wide angle lens.
- To pick any shape in any shape viewer to select the corresponding module in the Project Viewer use the button.

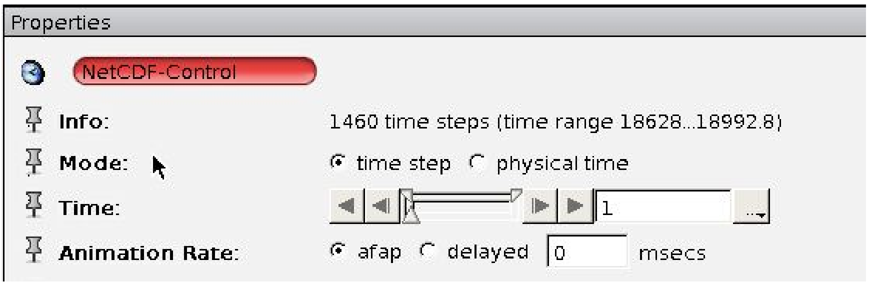


- You will need the left button (View All) to get an overview of the viewer content if you are lost during spatial navigation or nothing is visible anymore.
- Clicking the third button (XY) provides you a top view of your data.
- The second (YZ) and fourth button (XZ) provide lateral views.

Up to now you only did spatial navigation through the very first time step of your data stream. In this case this is the simulated relative humidity for January 1st, 2001 at 0:00 am. In the following chapter, we will spend some time on the temporal navigation.

3.3. Time Animation

- In the Main Panel click on the NetCDF-Control module. In the corresponding Properties, you will find that the NetCDF file contains 1460 time steps (= 365 days * 4 time steps).



Properties

NetCDF-Control

Info: 1460 time steps (time range 18628...18992.8)

Mode: ● time step ○ physical time

Time: 1

Animation Rate: ● afap ○ delayed 0 msecs

- Keep the **Mode** port on *time step*. The *physical time* stands for the days since the reference date and is hardly useful here.
- Move the *Time* slider rightwards and watch the evolution in the viewer. With the “Step forward” button ▶ next to the *Time* slider you can go through the year 2001 step-by-step. By clicking the “Play forward” button ▶ you can run the whole animation. Both buttons are also available for the reverse direction.
- Right-clicking in the Time slider provides you with a context menu that offers playing the animation once, playing it in repeat mode (Loop) e.g. for exhibitions or fairs and even playing it forward and backwards alternately (Swing).
- Slow the animation down to one time step per 3 seconds by changing the *Animation Rate* from afap (”as fast as possible“ ☺) to 3000 msecs delayed.

3.4. Toggle modules on or off

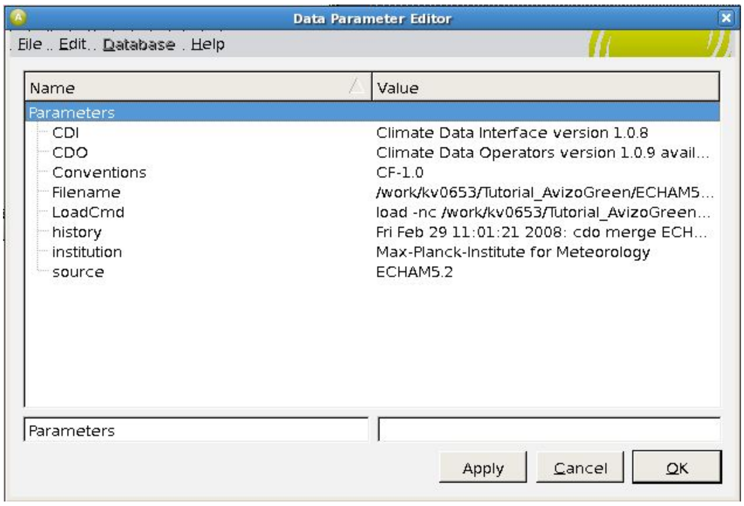
In the Main Panel, you may have noticed the second ECHAM5-MPI-OM data set in the project called tsurf (= surface temperature), which is, however, not visualized in the Viewer. That is because loaded data sets can be activated and deactivated by pressing their toggles:

- In the Main Panel you can activate the data set of the temperature by clicking on the small grey box for the variable *tsurf**. Alternatively, you might click the grey toggle box in the *Slice* Properties.
- As the temperature data are visualized on the same level as the underlying earth, switch the earth off by clicking on the orange box of the Earth module.
- You can also switch off or on the orange Bounding Box, the white grid at the back plane (module Grid Lines) or the *Isosurface* module.

3.5. Find metadata

Metadata of the whole NetCDF file

- In the Main Panel, activate the module *ECHAM5_OM_A1B_2001_3D.nc* and click on the Data Parameter Editor ▤ in the Properties.



Name	Value
Parameters	
CDI	Climate Data Interface version 1.0.8
CDO	Climate Data Operators version 1.0.9 avail…
Conventions	CF-1.0
Filename	/work/kv0653/Tutorial_AvizoGreen/ECHAM5…
LoadCmd	load -nc /work/kv0653/Tutorial_AvizoGreen…
history	Fri Feb 29 11:01:21 2008: cdo merge ECH…
institution	Max-Planck-Institute for Meteorology
source	ECHAM5.2

Parameters

In the Data Parameter Editor you may find information on the server path of the NetCDF file (Filename) and – very useful in practice – on the cdo processing steps applied before (history) in a descending chronological order (from new to old).

- Close the Data Parameter Editor with OK.

Metadata of individual variables

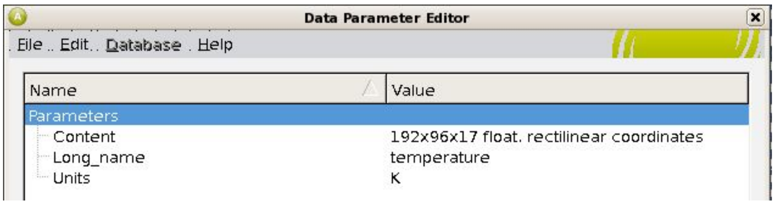
- Activate the variable *rhumidity** in the Main Panel window. In the Properties window you will now find the dimension of the variable, its min and max values and the underlying grid type.

  The grid of the atmospheric data has 192 x 96 x 17 grid points, meaning there are data for 17 levels. The min value of the current (!) time step is -0.142144, the max value is 1.26039. The grid is rectilinear.

- For a comparison click on the variable *tsurf** in the Main Panel window. As is visible in the Properties, this variable consists of the same amount of horizontal grid points but is supplied for one vertical level only. Min and max values of the first time step are 221.787 and 313.941.

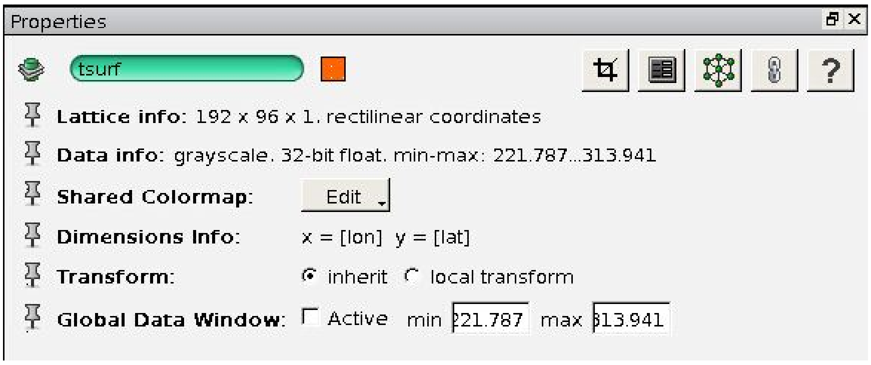
Please notice that an identical number of grid points along the x and y axis does not necessarily mean that the two data sets match together. They might cover a completely different area or differ in the distance between the grid points. Even if they are identical concerning coverage area and grid sizes, the data values might possibly stand for the grid cell center or one of the grid cell corners or even erect as vector components from the grid walls. In our case they actually match spatially, but keep those problems in mind.

- In the Properties click on the Data Parameter Editor button . In the Data Parameter Editor you will find additional metadata of this variable, e.g. its long name (“surface temperature”) and its unit (“K” - Kelvin).



Name	Value
Parameters	
Content	192x96x17 float, rectilinear coordinates
Long_name	temperature
Units	K

- Close the Data Parameter Editor with OK.
- Have a look at the Global data window port in the Properties window. As min and max values you will find the same values as displayed behind Info (representing only the current time step). But for the design of a colormap e.g. we will need the min and max values of all 1460 time steps together.



Lattice info: 192 x 96 x 1, rectilinear coordinates

Data info: grayscale, 32-bit float, min-max: 221.787...313.941

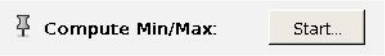
Shared Colormap: Edit

Dimensions Info: x = [lon] y = [lat]

Transform: ● inherit ○ local transform

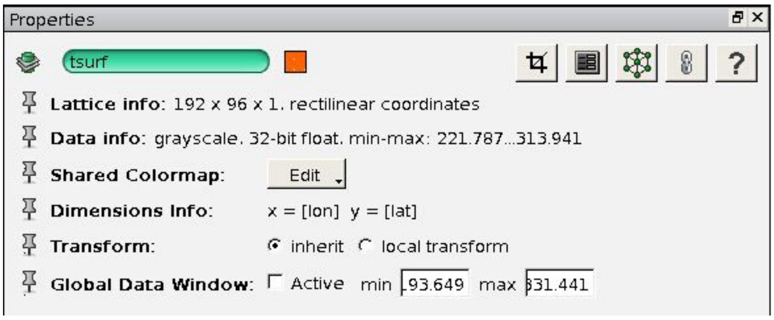
Global Data Window: □ Active min 221.787 max 313.941

- To compute them, scroll down the properties of the activated NetCDF-Control module, and click on *Start* at the **Compute Min/Max** port. In the bar below the Console window you can keep track of Avizo going through all time steps and comparing all current min and max values with the lowest and highest value found before.



Compute Min/Max:

- Go back to the Properties of the variable *tsurf*. In its “Global data window” fields you will now find the min and max values that have just been computed. The lowest surface temperature which occurred during the 1460 time steps is 193.649 Kelvin, the highest value is 331.441 Kelvin. Activate the Global data window now.



Lattice info: 192 x 96 x 1, rectilinear coordinates

Data info: grayscale, 32-bit float, min-max: 221.787...313.941

Shared Colormap: Edit

Dimensions Info: x = [lon] y = [lat]

Transform: ● inherit ○ local transform

Global Data Window: □ Active min 193.649 max 331.441

3.6. Colormaps and Colormap Legends

Colormap file format

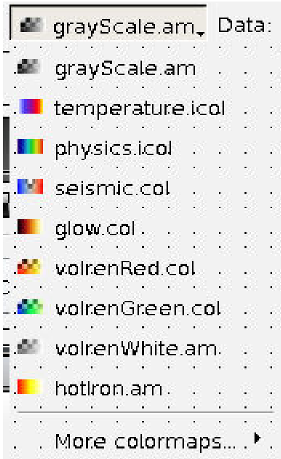
A *colormap* is a sequence of RGB-tuples, where every tuple specifies a color by a red, green and blue value in the *RGB color model*. Each value ranges from 0.0 to 1.0 and is represented by a floating point value. A fourth value, the so-called alpha value, defines opacity. It also ranges from 0.0 to 1.0, where 0.0 means that the color is fully transparent, and 1.0 that the color is full opaque. A colormap usually stores 256 different RGB-tuples, but other sizes are possible too.

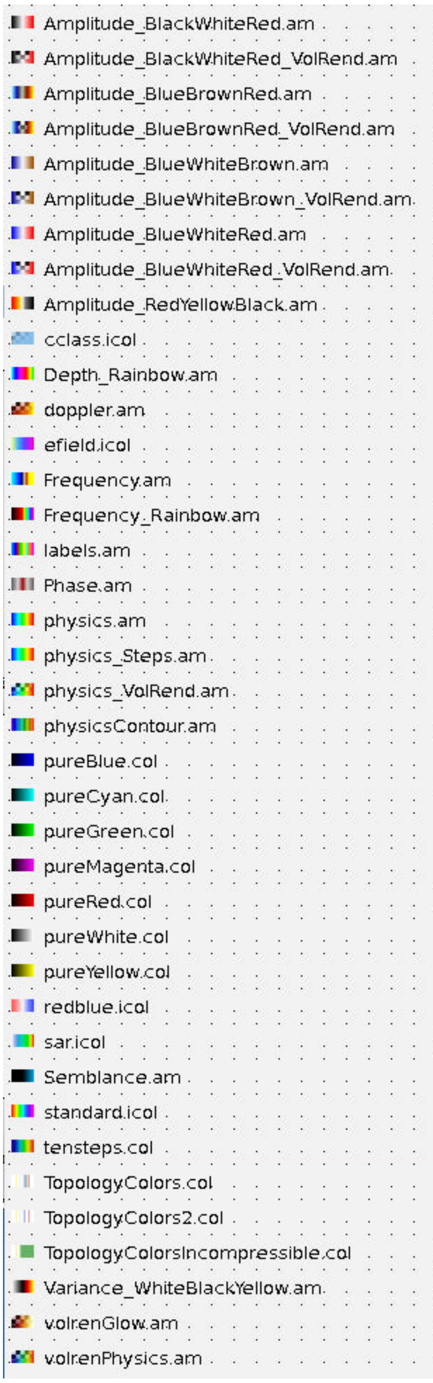
temp2D.icol:

```
# AmiraMesh 3D ASCII 2.0

# CreationDate: Thu Feb 28 14:44:53 2008


define Lattice 256

Parameters {
   MinMax 221.787 313.941,
   ContentType "Colormap"
}

Lattice { float[4] Data } @1

# Data section follows
@1
0 0 0 1
0 0 0.231373 1
0.0071144 0 0.23919 1
0.0142288 0 0.247007 1
0.0213432 0 0.254824 1
0.0284576 0 0.262641 1
0.035572 0 0.270458 1
……
……
```

The colormap file also contains the value range (minimum and maximum) of the data to be displayed.
The example temp2D.icol colormap file set the value range from 221.787 to 313.941 (in this case temperature in Kelvin). The number of colors is set to 256 (Lattice).

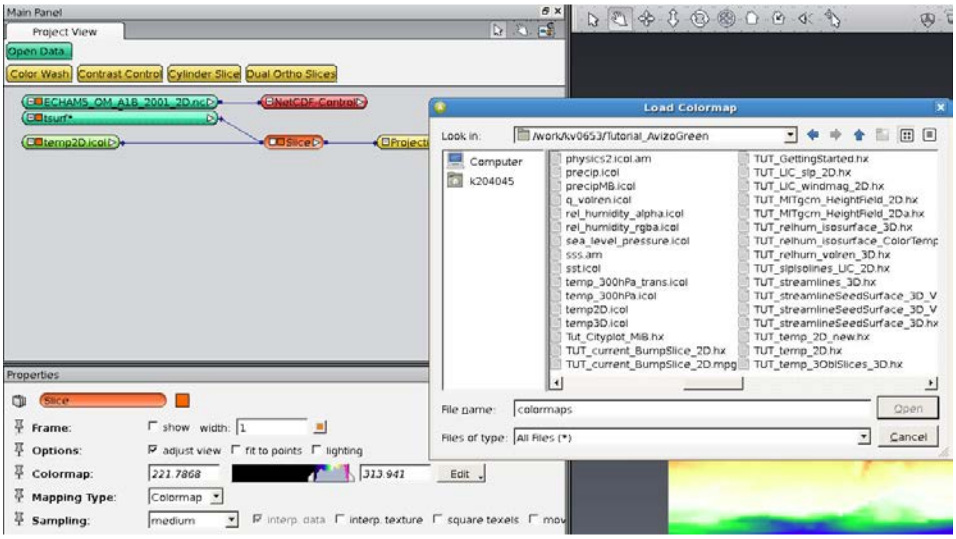
Pre-defined colormaps





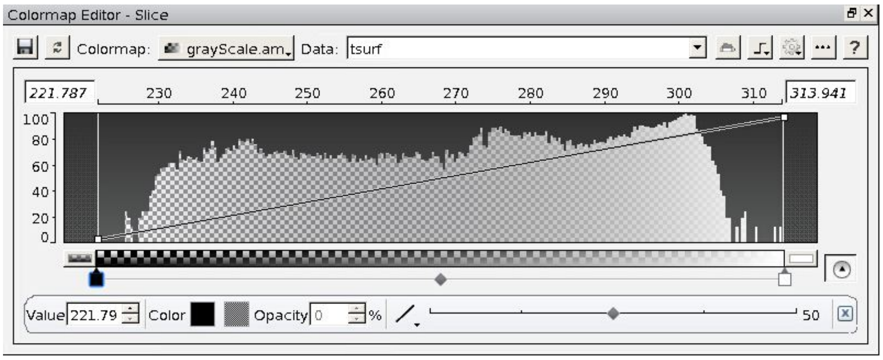
Load Colormap

In the properties window of the Display type, e.g. Slice, click on the button on the right side of the Colormap line ***Edit → Options → Load colormap***. Now, you can select a colormap file, e.g. /work/kv0653/Tutorial_AvizoGreen/temp2D.icol. Further details on the *Colormap Editor* can be found in its menu bar by clicking on Help.

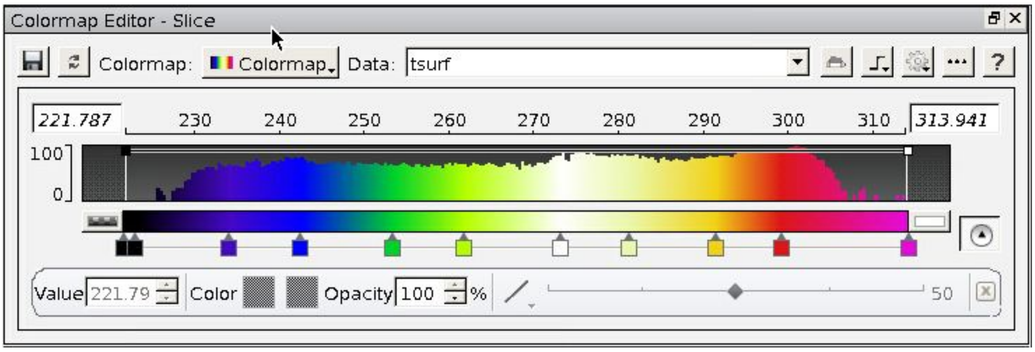


Edit Colormap

The default colormap of AvizoGreen is set to grayscale. To change a colormap, AvizoGreen provides the *Edit colormap* feature to define colors and their appropriate data values. In the Properties Window click on the Edit button on the right side of the Colormap line ***Edit → Options → Edit colormap***.



You can see a histogram with the value distribution of the variable, which can be graphed either on a logarithmic (default) or linear scale. The colormap is provided with colors defined by the boxes below and opacity defined by the line across the histogram. New colors could be inserted by left mouse click under the color histogram. The color key squares could be moved to the desired color or value.

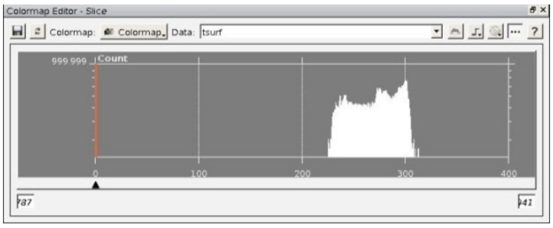


Set extreme values manually

Just change the minimum and maximum values on the top left and right hand side in the Colormap Editor to the desired values.

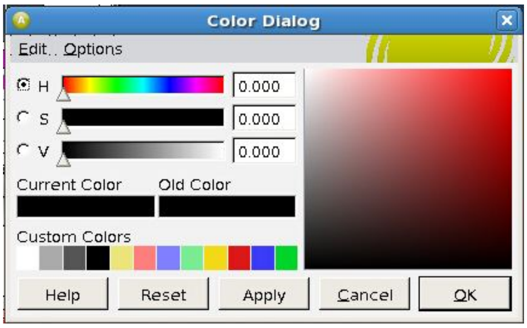
Adjust the range of the data

Click *Adjust range* in the *Colormap Editor* in order to adjust the domain of your colormap to the domain of your data set.



Create new color value (key value)

Click below the colorbar on one of the “key value” boxes and move the cursor a little bit horizontally. You will notice that the “key value” changes gradually. When you release the mouse button, a new key value / color pair is defined. In order to change the color, click on the color on the right side of the *Value* field. The Color Dialog will pop up where you can change the color values by moving the sliders or clicking into the color field.



H 0.000
S 0.000
V 0.000

Delete color value (key value)

Move the mouse onto the key value which you want to delete and right click.

Save Colormap

To save the colormap for your dataset, click on the 💾 button. Select “non-indexed icol” for the data format. The default location, where Avizo tries to save colormaps, is currently the “colormaps” subdirectory of the Avizo system installation - *where you shouldn’t be able to write anything*. So make sure to choose a directory owned by you!

Create an own Colormap

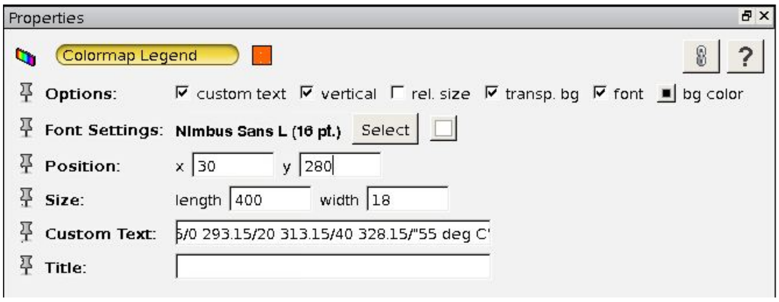
In the Main Panel, right-click on an empty space ***Create Object → Other → Colormap → Create***.

Configure your colormap and save it using the file format “non-indexed icol”.

Colormap Legend

The Colormap Legend, also known as Color Labelbar, displays the colormap with the appropriate data values as an icon in the Viewer. To append a colormap legend, right-click on the Colormap module in the Main Panel ***Annotate* → *Colormap Legend* → *Create***. A new module will be added to the Main Panel window.

The Colormap Legend could be displayed in a horizontal or vertical direction. The default setting is horizontal; to change the direction to vertical, activate the vertical toggle.



Properties

Colormap Legend

Options: ☑ custom text ☑ vertical ☐ rel. size ☑ transp. bg ☑ font ■ bg color

Font Settings: **Nimbus Sans L (16 pt.)** Select

Position: x 30 y 280

Size: length 400 width 18

Custom Text: 5/0 293.15/20 313.15/40 328.15/"55 deg C'

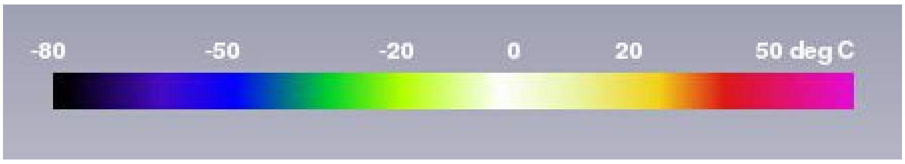
Title:

You can modify the default Colormap Legend annotation (the minimum value, the maximum value and the value in between) to any other alphanumerical annotation by activating the “custom text” toggle in the **Options** port.

For example:

```
223.15/-50 253.15/-20 273.15/0 293.15/20 313.15/"40 deg C"
```

```
[and press ENTER!]
```



This results in five instead of three entries. Each entry consists of a data value, a slash (“/“) and the text that shall be visible at the corresponding position. If you want to add the units at the top entry of the colormap, you might set the text “40 deg C” in quotation marks to indicate that blank characters are included. Special characters like ° (degrees) are not accepted by *Avizo*, unfortunately.

Hint: Of course there are also other ways to achieve a °C instead of K annotation, for example:

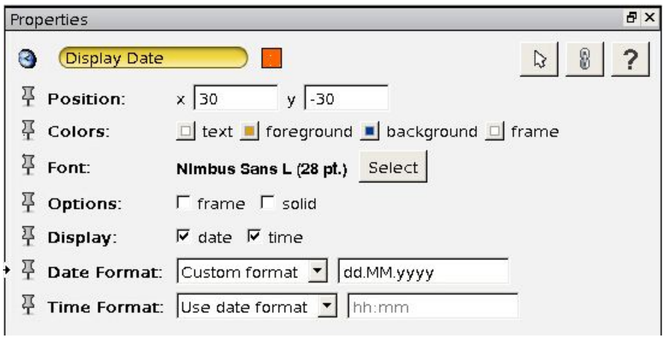
a) use *cdo subc* to subtract the constant value 273.15 from the temperature data before visualizing the data

b) use the Avizo module “Arithmetic” to do the conversion on the fly. But be careful, this may slow down the animation since the operation has to be done again and again for every new time step.

3.7. Caption, Annotation and Display Date

Date and time display

The date and time of the simulation data could be displayed in the Viewer window: In the main panel, right-click on the NetCDF-Control module and then click ***Annotate* → *Display Date* → *Create***. A new module called *Display Date* is added.



Properties

Display Date

Position: x 30 y -30

Colors: □ text ■ foreground ■ background □ frame

Font: Nimbus Sans L (28 pt.) Select

Options: □ frame □ solid

Display: ☑ date ☑ time

Date Format: Custom format dd.MM.yyyy

Time Format: Use date format hh:mm

If *Avizo* produces an error (“Invalid date”), your NetCDF time format might be given in absolute time format instead of relative times (see chapter 2).

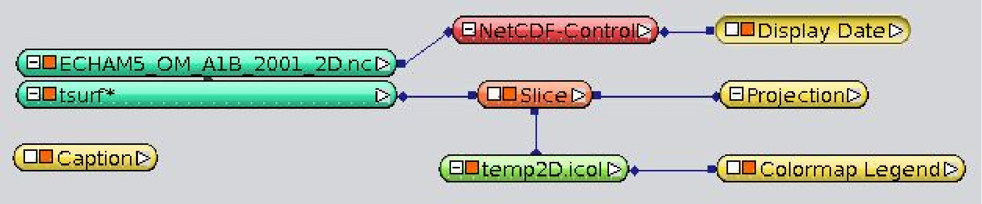
You can change the date format or abstain from the display of the weekday. In the Properties of the *Display Date* module, click on *Custom format* in the **Date Format** port and enter *dd.MM.yyyy* for the German data format or *MM.dd.yyyy* for the English data format (don’t forget to press ENTER!).

In the **Time Format** port, also choose *Custom format* and keep the default setting *hh:mm*.

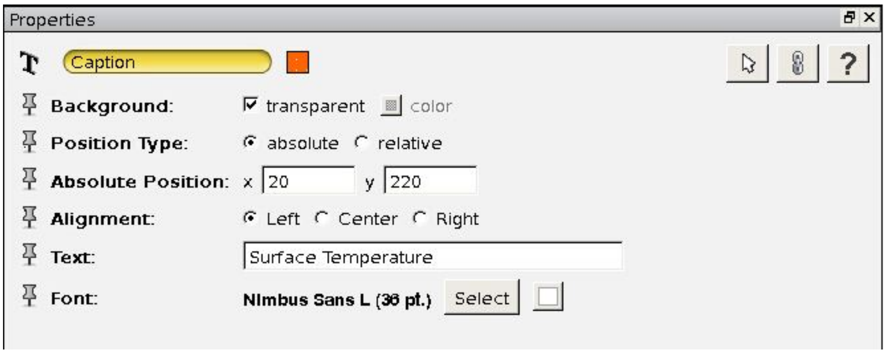
In the **Position** port you may vary the position of the time display in the viewer window. Test some values to arrange the time display in the top left corner. You can also use negative values.

Captions

To create a text field to act as a title, sub-title or any text string, you can use the caption feature of Avizo. In the Main Panel, right-click ***Create Object* → *Annotations* → *Caption* → *Create***. *Avizo* will add a new module called *Caption in the Main Panel*.



The default annotation text is the current date and time.


Properties

Caption

Background: ☑ transparent □ color

Position Type: ● absolute ○ relative

Absolute Position: x 20 y 220

Alignment: ● Left ○ Center ○ Right

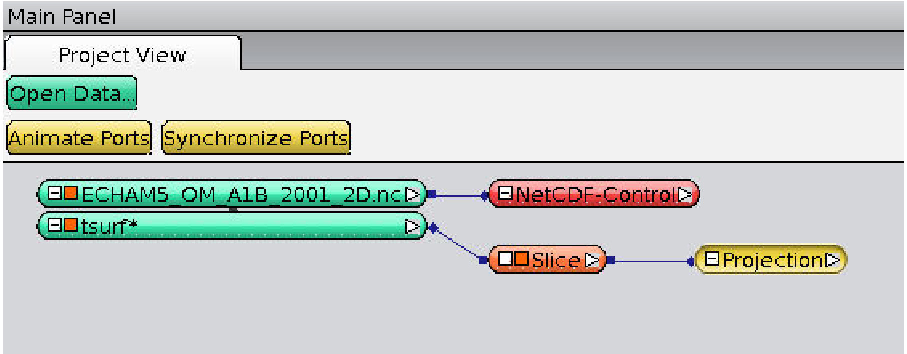
Text: Surface Temperature

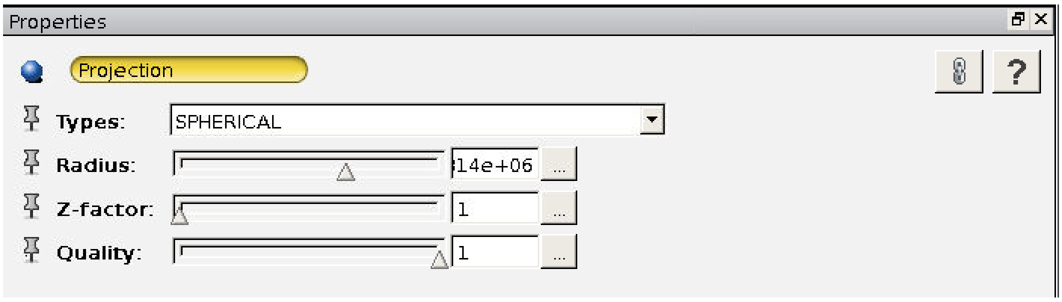
Font: Nimbus Sans L (36 pt.) Select

3.8. *Projections*

AvizoGreen provides a wide range of geographical projections, but only the common projections are mentioned here.

First you have to load your data and select a Display Type, like Slice. In the properties window of the Slice Display Type click on the Button which appends the Projection module to the Main Panel.




Properties

Projection

Types: SPHERICAL

Radius: 14e+06

Z-factor: 1

Quality: 1

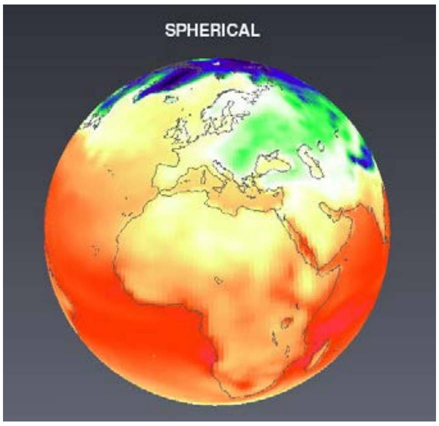
Click on the Projection module, move the mouse to the Projections Properties window and click on the *Types* selector to choose the projection type.

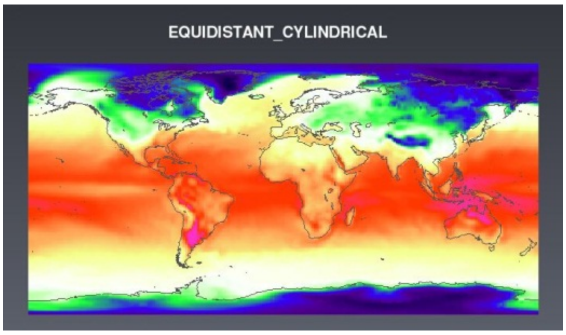
List of the most common Projections in AvizoGreen

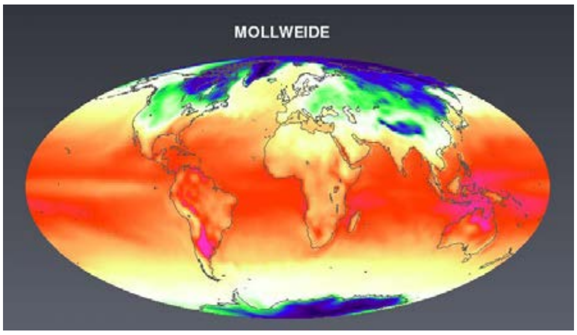
- None
- SPHERICAL
- EQUAL_AREA_CYLINDRICAL
- ECKERT (I II III IV V VI)
- EQUIDISTANT_CYLINDRICAL
- LAMBERT_EQUAL_AREA_CONIC
- MERCATOR
- MOLLWEIDE
- ROBINSON
- UNIVERSAL_POLAR_STEREOGRAPHIC

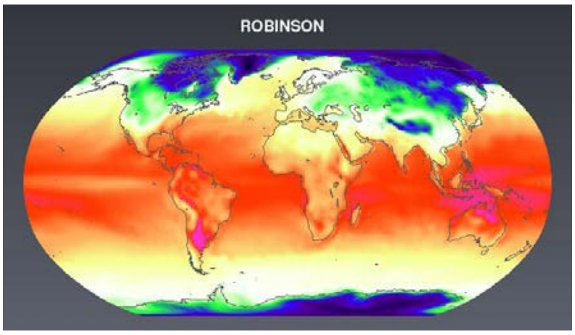
Tip: Don't forget to project!

Example plots of projections







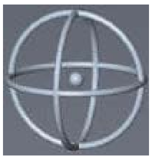


3.9. Preferences

a) When starting Avizo for the first time, it makes sense to switch off some potentially confusing ports in the properties windows:

Edit* → *Preferences* → *Layout in the ***Project View*** section switch OFF "*Show connection ports in Properties Area*"

b) If you decide that you need the trackball in the viewer window you can switch it on:



Edit* → *Preferences* → *Layout in the ***Viewer gadgets*** section switch ON "*Show the camera trackball*"

c) In case you wish to have the viewer window as a separate window, you can do so in the Preferences as well:

Edit* → *Preferences* → *Layout in the ***Windows*** section switch ON "*Show viewer in top level"*

d) If you want Avizo to save the viewer positions and sizes within the Avizo project in order to start with the same window layout when reopening the project, then:

Edit* → *Preferences* → *Layout in the ***Windows*** section switch ON "Save *window layout on exit*"

e) To enable the spinning function (letting the Earth turn automatically when giving it a kick):

Right-click in the viewer window → ***Preferences* → *Spin animation (on)***

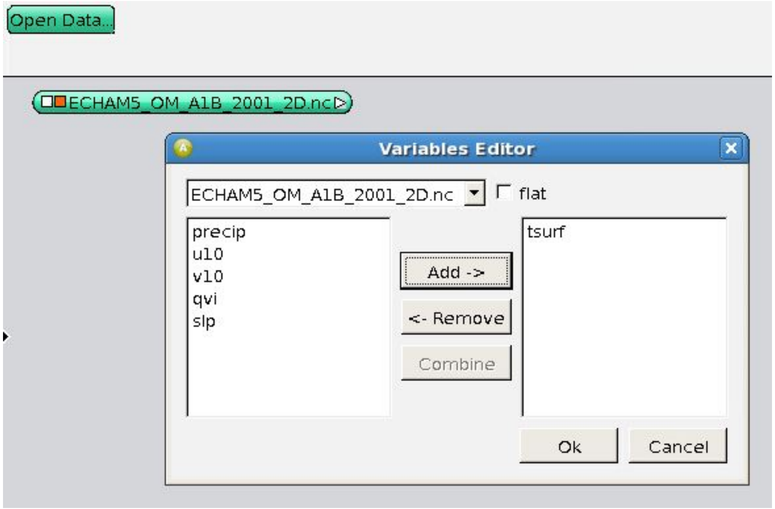
Let's stop this preface and go straight into details. We will now load data on our own, visualize them, and build our own projects, with both 2D data 3D data.

	2D	***3D***
Scalar data	chapter 4	chapter 4
Vector data	chapter 5	chapter 6

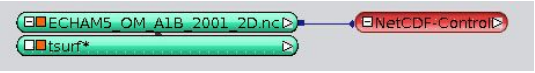
4. Visualization of 2D and 3D Scalar Data

4.1. Surface temperature as Slice

- In the Main Panel click *Open Data* and load the NetCDF file *ECHAM5_OM_A1B_2001_2D.nc*.
- The *Variables Editor* opens and lists the variables available in the NetCDF file. In the *Variables Editor* choose the variable *tsurf* (surface temperature) by clicking *Add* and confirm with *Ok*.



Avizo loads the selected variables of the data set, reports this in the *Avizo* Console, and represents the data set with green modules in the Main Panel window – in this case one for the NetCDF file as a whole and one for the tsurf* variable. [Tip: You can access the *Variables Editor* later, e.g. to select other variables, by clicking on the green NetCDF file module in the Main Panel window and then on the button in the according Properties panel.]
Additionally, *Avizo* automatically connects each loaded NetCDF file to a red NetCDF-Control module, which you might use e.g. for navigation through time.



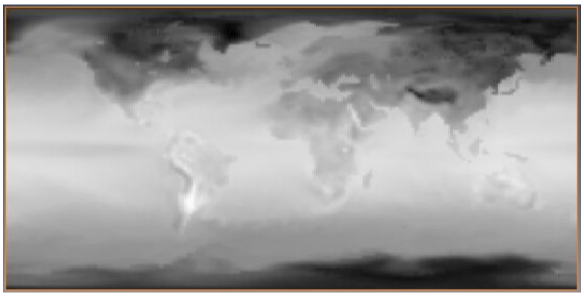
Questions concerning metadata:

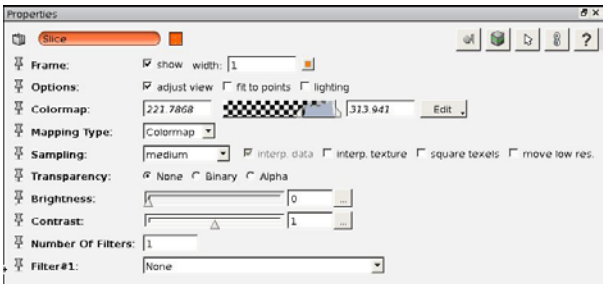
- How many time steps are available?
- What are the min and max values in the first time step and over all time steps?
- What kind of grid is it?
- Which *cdo* commands were used last and when to process the file?
- In which physical unit is the temperature provided?

Answers

- To visualize the temperature, right-click on the variable and then click ***Display* → *Slice*** and select ***Create***.

  In the Main Panel, Avizo connects the variable *tsurf** with the new Slice module. In the viewer, a first visualization of the global surface temperature is visible – up to now only in black and white (gray scale).





Properties

Slice

Frame: ☑ show width: 1

Options: ☑ adjust view ☐ fit to points ☐ lighting

Colormap: 221.7868 313.941 Edit

Mapping Type: Colormap

Sampling: medium ☑ interp. data ☐ interp. texture ☐ square texels ☐ move low res.

Transparency: ● None ○ Binary ○ Alpha

Brightness: 0

Contrast: 1

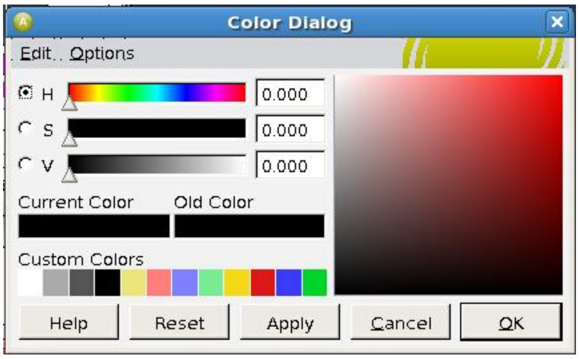
Number Of Filters: 1

Filter#1: None

4.2. *Defining a Colormap*

- In the Properties of the **Slice**, the **Mapping Type** should be *Colormap*.
- In order to create your own colormap, right-click into the Main Panel and select ***Create Object* → *Other* → *Colormap*** and select ***Create***. To connect the new colormap to the Slice module, click on the little white square on the left side of the Slice module and select "Colormap". Move the mouse cursor to the Colormap module you have created and click into it. Now a connection between both modules is established, and the Slice turns white.

  [**Hint:** Alternatively, you can select your newly created colormap as a **shared colormap** for the variable you're about to visualize. Click on the white square of the tsurf module, select "shared colormap" and then click on your new colormap module. All visualization modules connected to tsurf will now use the same color mapping.]
- You may want to edit the colormap: define combinations of physical values and the respective colors. In the **Slice** Properties right-click on Colormap and select ***Options* → *Edit colormap***. The *Colormap Editor* opens.
- Click ***Adjust range*** in the *Colormap Editor* in order to adjust the domain from your colormap to the data range of your variable.
- Click below the colorbar on one of the "key value" boxes and move the cursor a little bit horizontally. You will notice that the "key value" changes gradually. When you release the mouse button, a new key value / color pair is defined. In order to change the color, click on the color on the right side of the Value field. The Color Dialog will pop up where you can change the color values by moving the sliders or clicking on the color field.



Color Dialog

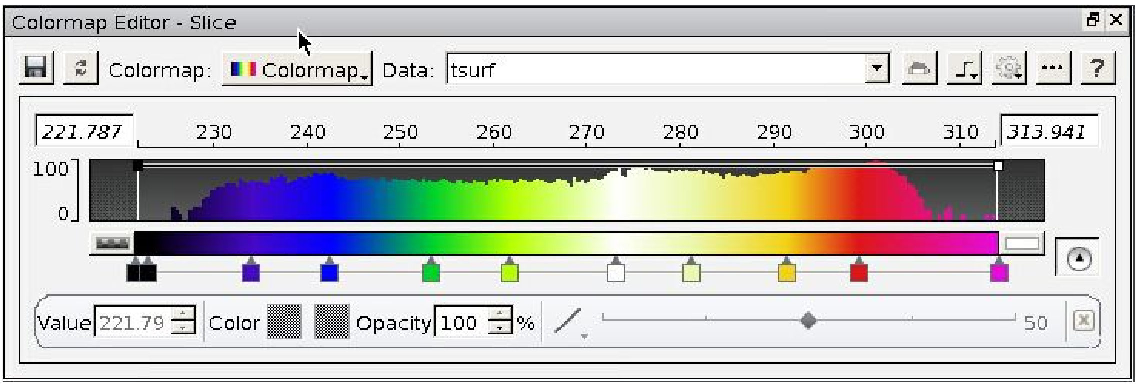
● H 0.000

○ S 0.000

○ V 0.000

- For example, choose a white color for 273 Kelvin (= 0°C), then Green-Blue-Purple-Black for values below 273 K and Yellow-Orange-Red-Pink for values larger than 273 K.
- Move the line of opacity upwards to the top to visualize all colors with full opacity (100%) [default]. To get more transparency, move the line downwards and see what happens.
- You may also fill the two boxes for extreme values: on the left hand side choose black, on the right hand side choose pink. When clicking ***Adjust range**, Avizo* has only loaded the min and max values of the current time step and it may so happen higher or lower values occur in other time steps. Such values above and below the selected data range will be visualized with the respective colors of the two boxes.

  [Tip: Alternatively, you may compute the absolute min and max values of all time steps in the Properties of the NetCDF-Control module, note them (Global data window of *tsurf** module) and enter them manually in the *Colormap Editor* as limiting left and right values.]
- Save your colormap as *temp2D.icol*. Select “non-indexed icol” for the data format. The default location, where Avizo tries to save colormaps, is currently the “colormaps” subdirectory of the Avizo system installation - where you shouldn’t be able to write anything. So, make sure to choose a directory owned by you!


Colormap Editor - Slice

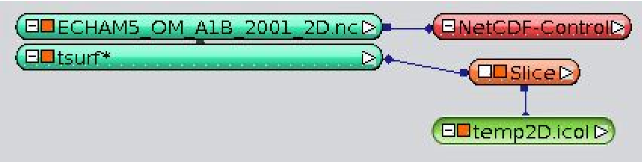
Colormap: Colormap Data: tsurf

221.787 313.941

Value 221.79 Color Opacity 100 %

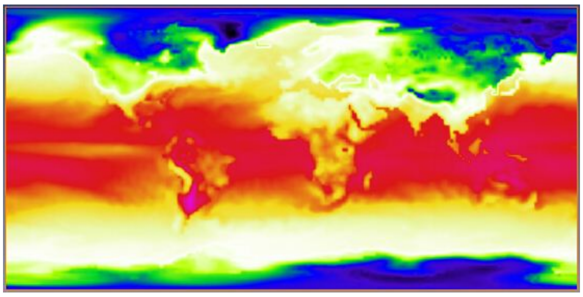
- Close the *Colormap Editor.*

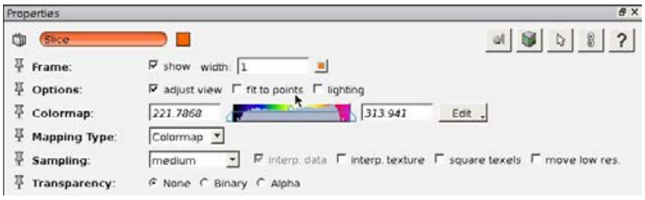
The network in your Main Panel window should look something like this now:



- The module ***Slice*** shows the temperature as grid cells. Zoom close to the surface to view the individual pixels.
- In the Properties window you may change the sampling resolution (from coarse to finest) or even choose an interpolated texture resulting in a less pixelated visualization. You should be aware, though, that Slice does not display the

original grid sampling of the data. You may also notice that the rendering speed will decrease with increasing resolution used.





Properties

Slice

Frame: ☑ show width: 1

Options: ☑ adjust view ☐ fit to points ☐ lighting

Colormap: 221.7868 313.941 Edit

Mapping Type: Colormap

Sampling: medium ☑ interp. data ☐ interp. texture ☐ square texels ☐ move low res.

Transparency: ● None ○ Binary ○ Alpha

4.3. Saving your Project

Avizo can save the visualization application you have just compiled as a “project”: a script file with the extension *.hx, which enables you to later reopen the very same application again.

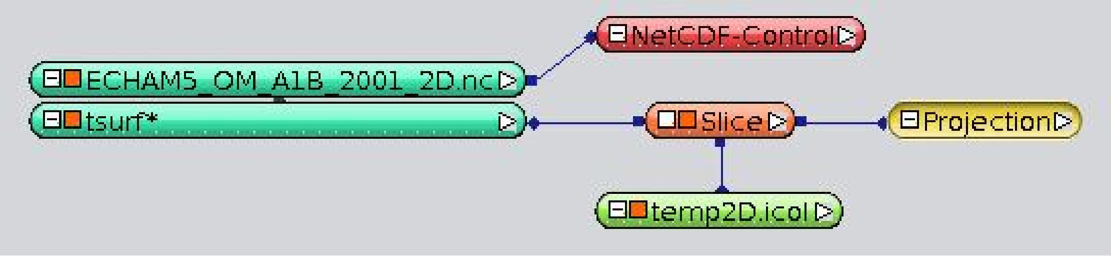
- Click on ***File → Save Project As*** where you may choose the file name *temp_2D.hx*.

In case of any problems or uncertainties you can find a similar project in the tutorial folder under the name *TUT_temp_2D.hx*.

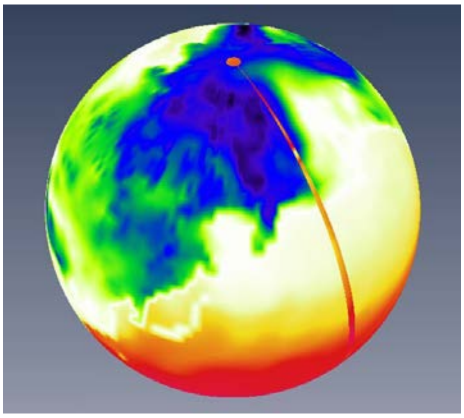
4.4. Using Projections

Up to now, no geographic projection has been applied to our visualization. *AvizoGreen*, however, offers a huge number of projections of your choice:

- In the Slice Properties click the projections button. Avizo uses a spherical projection (globe) as default.



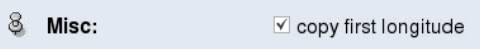
- For orientation reasons click *View All* in the navigation bar.
- Explore the earth from all sides. What is striking?



Since the ECHAM5-OM1 global models grid cell centers lie in the range between 88.6°N to 88.6°S, you will have “holes” at the north and south pole. Unfortunately, they cannot easily be closed by means of interpolation or extrapolation.

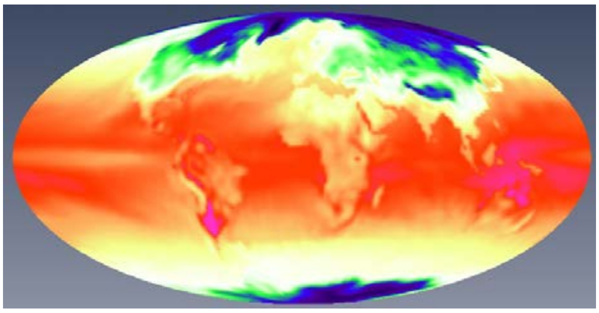
Additionally, *Avizo* does not close the gap at the model borders in the Pacific Ocean as it does not know that the first column of the data (at ~ -180°) and the last column (at ~ +180°) are topologically direct “neighbors”. This problem can easily be solved:

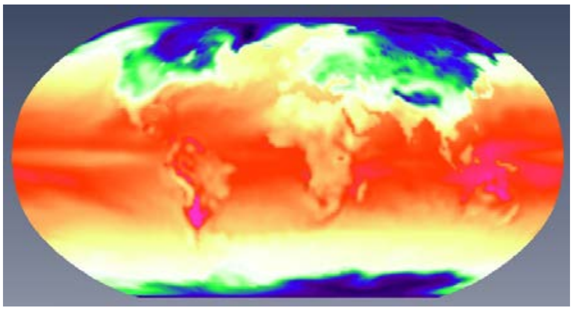
- At the bottom of the NetCDF-Control Properties activate the *copy first longitude* toggle of the **Misc** port.



Misc: ☑ copy first longitude

- In the Properties of the Projection module test *MOLLWEIDE* projection („a recumbent egg“, left figure) and the *ROBINSON* projection (right figure) of the **Types** port.



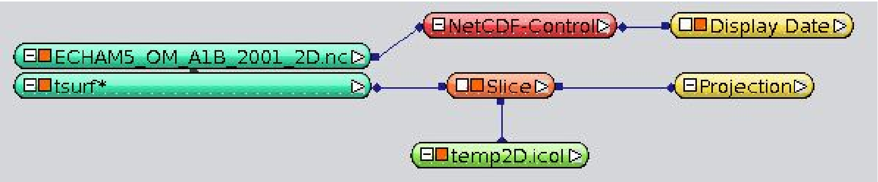


- You may also want to test the *EQUIDISTANT_CYLINDRICAL Projection*. What’s the difference between this projection and the unprojected visualization? Answer.
- Save your project in this Equidistant Cylindrical Projection since we want to continue with this projection in the next chapter.

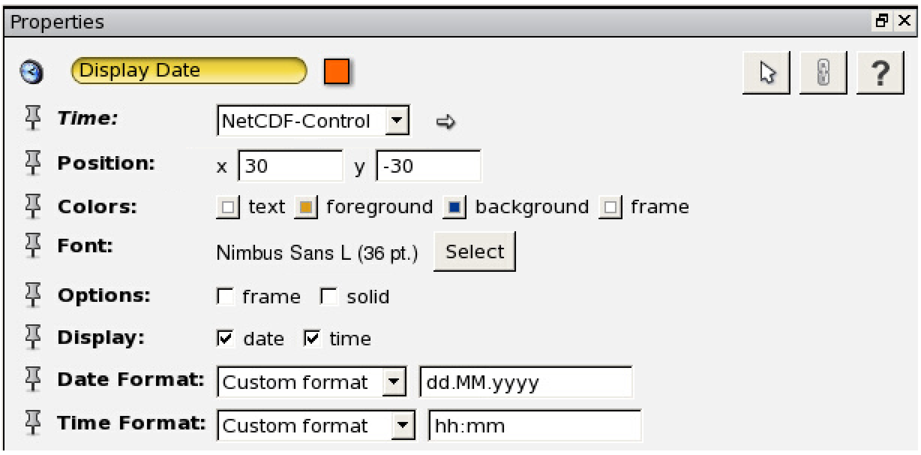
4.5. Legend: Display Date, Colormaps and Annotations

Date and time display

- In order to display the date and time of the simulation data in the viewer window, right-click on the NetCDF-Control module and then click ***Annotate* → *Display Date* → *Create***.



- Test it by starting the animation in the NetCDF-Control Properties. If *Avizo* produces an error ("Invalid date"), your NetCDF time format might be given in absolute time format instead of relative times (see chapter 2).
- You may want to change the date format or abstain from the display of the weekday. In the Properties of the Display Date module click on *Custom format* in the **Date Format** port and enter *dd.MM.yyyy* for the German data format or *MM.dd.yyyy* for the English data format (and press ENTER!).
- In the **Time Format** port also choose *Custom format* and keep the default setting *hh:mm*.
- Enlarge the **Point Size** from 14 to 36, choose font *Nimbus Sans L* .
- In the **Position** port you may vary the position of the time display in the viewer window. Test some values to arrange the time display in the top left corner. You can also use negative values.



Properties

Display Date

Time: NetCDF-Control

Position: x 30 y -30

Colors: □ text ■ foreground ■ background □ frame

Font: Nimbus Sans L (36 pt.) Select

Options: □ frame □ solid

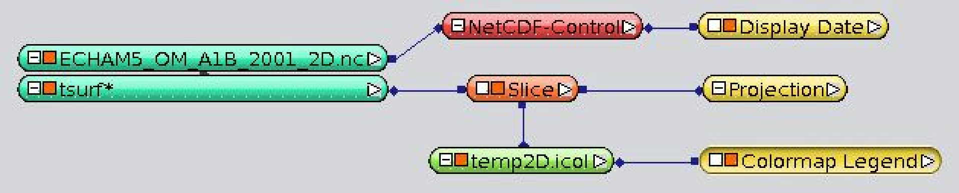
Display: ☑ date ☑ time

Date Format: Custom format dd.MM.yyyy

Time Format: Custom format hh:mm

Colormap Legend

- In the Main Panel window right-click on the colormap module *temp2D.icol* and then click ***Annotate* → *Colormap Legend* → *Create***.

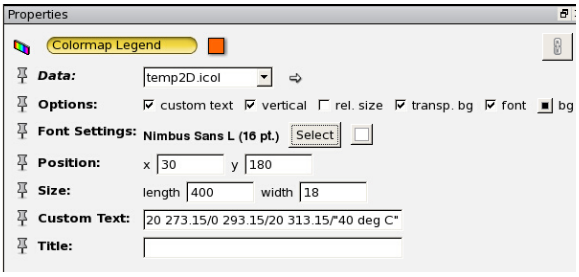


- In the *Colormap Legend* Properties activate the *vertical* toggle in the **Options** port to change the display from horizontal to vertical. This might be suggested here as the temperature mainly alters with the latitudes.
- You can change the default colormap annotation (the minimum value, the maximum value and the value in the middle between both numbers) to any other alphanumerical annotation by activating the “custom text” toggle the **Options** port.
- The default entry in the input field for the custom format results in the annotation “Low – Medium – High“. This can be changed to absolute data values, and you could also use “Degrees Celsius” instead of “Kelvin” - which might be more comprehensive for non-meteorologists. Enter the following text in the **Custom Text** port:

```
223.15/-50 253.15/-20 273.15/0 293.15/20 313.15/"40 deg C"

[and press ENTER!]
```

This results in five instead of three entries. Each entry consists of a data value, a slash (“/“) and the text that shall be visible at the corresponding position.

- Since the colormap is a bit too short for five entries, we shall enlarge it by entering a *length* of *400* in the **Size** port.



Properties

Colormap Legend

Data: temp2D.icol

Options: ☑ custom text ☑ vertical ☐ rel. size ☑ transp. bg ☑ font ■ bg

Font Settings: Nimbus Sans L (16 pt.) Select

Position: x 30 y 180

Size: length 400 width 18

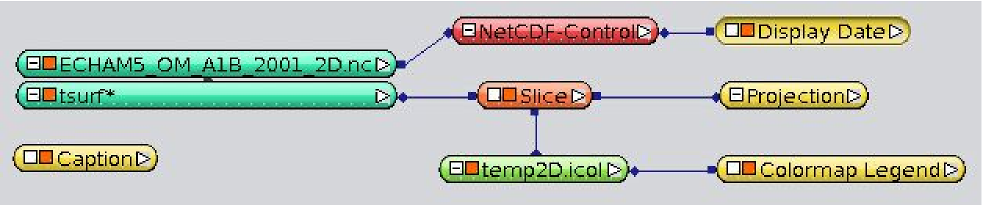
Custom Text: 20 273.15/0 293.15/20 313.15/"40 deg C"

Title:

- You may want to enlarge the annotation text by activating the font menu and change the settings to font *Nimbus Sans L* bold and 16 pt. size.

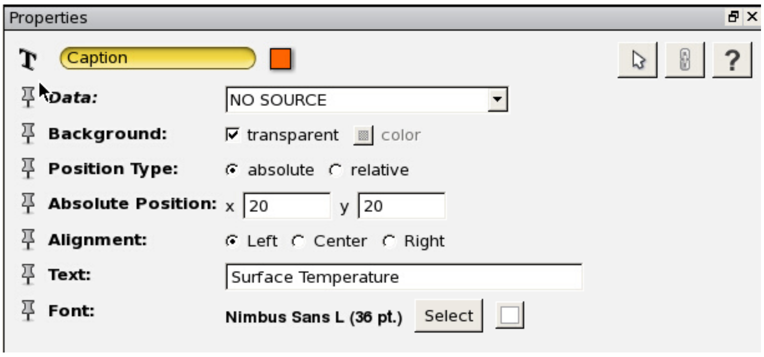
Display of Captions

- To complete this legend with an annotation, click ***Create Object*** **→** ***Annotations*** **→** ***Caption*** **→** ***Create*** in the menu bar. In the Main Panel, *Avizo* adds a new module again.



- The default annotation text is the current date and time. In the Caption Properties you may change it to “*Temperature”* in the **Text** port.

- Move the Caption to the bottom of the colormap by changing the position parameters in the Properties area (drag & drop is not possible in the main viewer). Please note: The Caption module is not connected to the Colormap module and in the case of repositioning the legend, you will have to move both elements separately.



Properties

Caption

Data: NO SOURCE

Background: ☑ transparent □ color

Position Type: ● absolute ○ relative

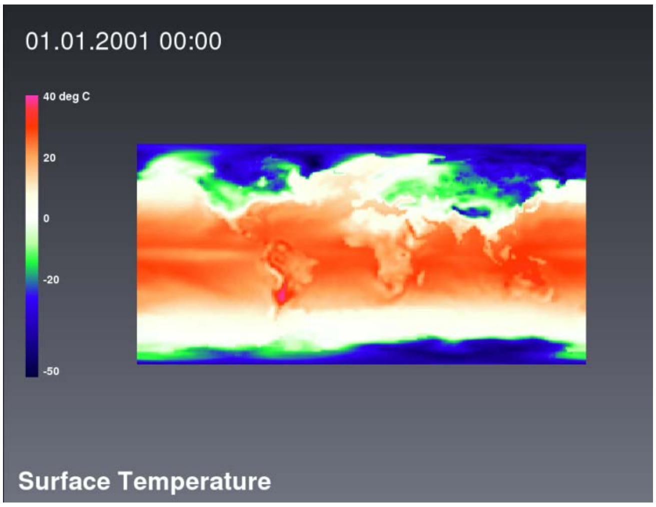
Absolute Position: x 20 y 20

Alignment: ● Left ○ Center ○ Right

Text: Surface Temperature

Font: Nimbus Sans L (36 pt.) Select

The main Viewer should now look something like this:

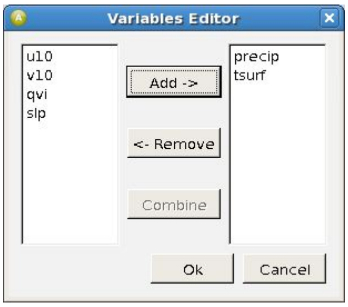


- Save your project as *temp_layout_2D.hx* (File → Save Project As).

In case of uncertainties you will find the current project in the tutorial folder under the file name *TUT_temp_layout_2D.hx*.

4.6. Precipitation as Bar Chart Slice

- Add the precipitation data set by clicking on the NetCDF file module *ECHAM5_OM_A1B_2001_2D.nc* again, and open the *Variables Editor* in the Properties window. Then add the variable ***precip*** to the list on the right side and click Ok.

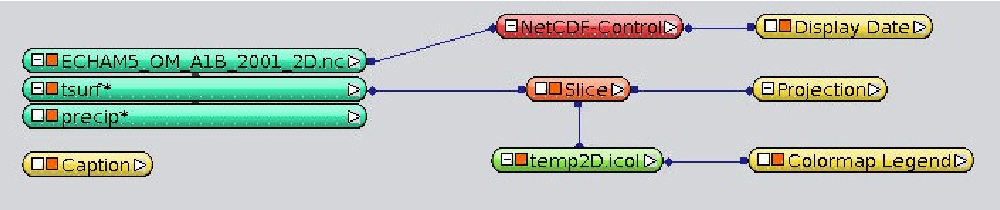


Variables Editor

u10
v10
qvi
slp

precip
tsurf

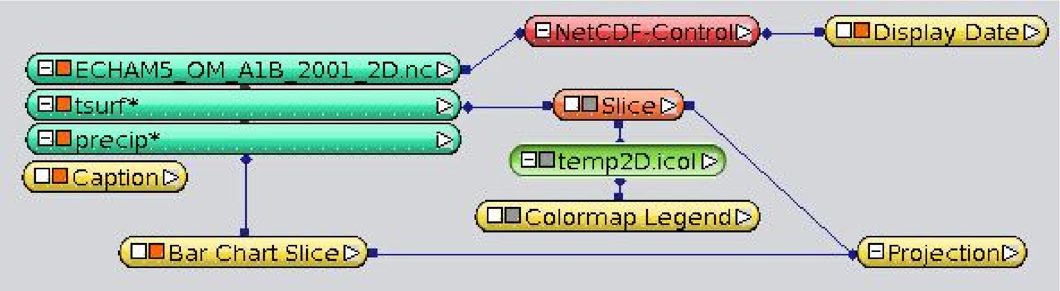
The second variable is added to the NetCDF module in the Main Panel:



- Turn off the Slice with the temperature data, the colormap and the annotation by clicking their toggles. (How? See chapter 3.4)

Now, we want to visualize the precipitation as a 3D bar diagram with different bar lengths representing the precipitation value in each grid cell. This visualization method is especially intuitive for variables which only can take positive values. In the case of rainfall it reproduces our perception of a „height of precipitation".

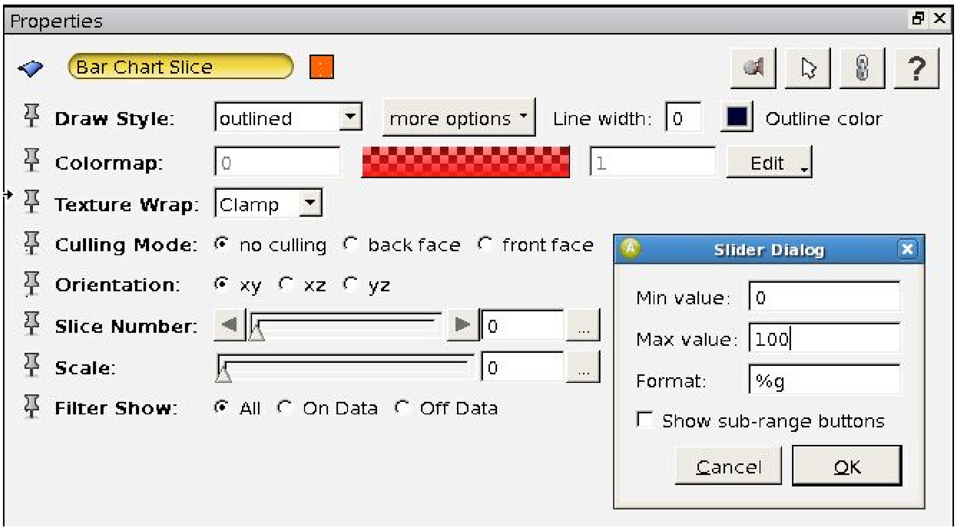
- In the Main Panel window, right-click the variable ***precip****, then click ***Display* → *Bar Chart Slice* → *Create***. Avizo adds it as a new module.
- In the *Bar Chart Slice* Properties, click on the projection symbol . Avizo then connects the Bar Chart Slice module to the existing Projection module.



- In case you have a perpendicular XY-View of your map in the viewer window, incline it a bit with the trackball to make sure the bars erect from the earth surface as wished.

Unfortunately, up to now, the "bars" only look like flat squares, even from an inclined view. You will have to apply a reasonable vertical scaling:

- In the *Bar Chart Slice* Properties, move the **Scale** slider to 1. This scaling seems to be insufficient since the "bars" are still flat.
- To enlarge the value range of this slider, click on the field on its right hand side.
- In the opening *Slider Dialog*, enter the *Max value 100* and confirm with OK.



Properties

Bar Chart Slice

Draw Style: outlined | more options | Line width: 0 | Outline color

Colormap: 0 | 1 | Edit

Texture Wrap: Clamp

Culling Mode: ● no culling ○ back face ○ front face

Orientation: ● xy ○ xz ○ yz

Slice Number: 0

Scale: 0

Filter Show: ● All ○ On Data ○ Off Data

Slider Dialog

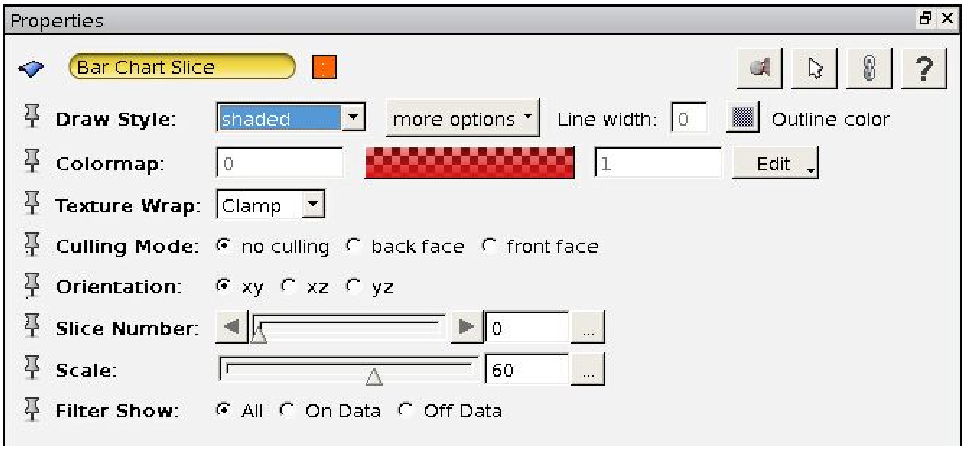
Min value: 0

Max value: 100

Format: %g

□ Show sub-range buttons

- Move the slider to the right again. Now the range is sufficient to scale our bars. Set the scale to *60*.
- Additionally, change the **Draw Style** from *outlined* to *shaded*.



Properties

Bar Chart Slice

Draw Style: shaded | more options | Line width: 0 | Outline color

Colormap: 0 | 1 | Edit

Texture Wrap: Clamp

Culling Mode: ● no culling ○ back face ○ front face

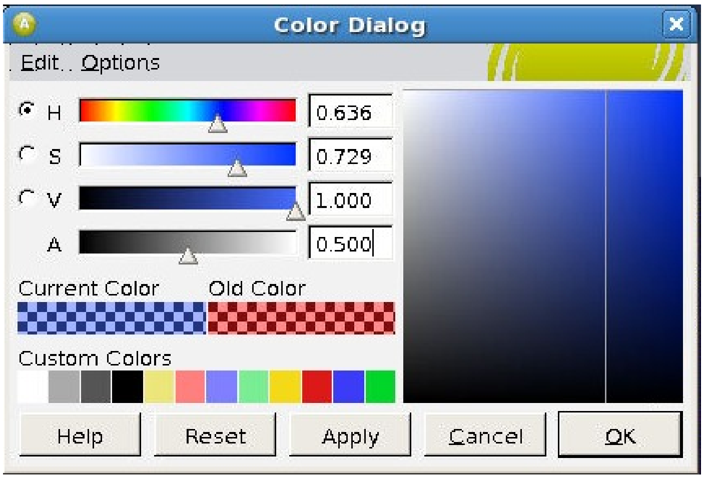
Orientation: ● xy ○ xz ○ yz

Slice Number: 0

Scale: 60

Filter Show: ● All ○ On Data ○ Off Data

- Precipitation is now represented by red bars. Blue would be more suitable for rain. Change the bar color into blue by double-clicking the red colormap in the *Bar Chart Slice* Properties.
- In the opening **Color Dialog** shift the triangle of the H port (H = hue) to a Blue and reduce the saturation (= S port) a bit. Click OK.



Color Dialog

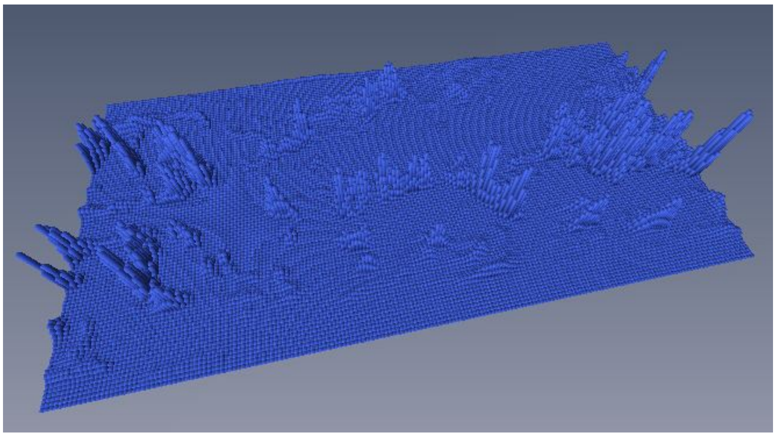
● H 0.636

○ S 0.729

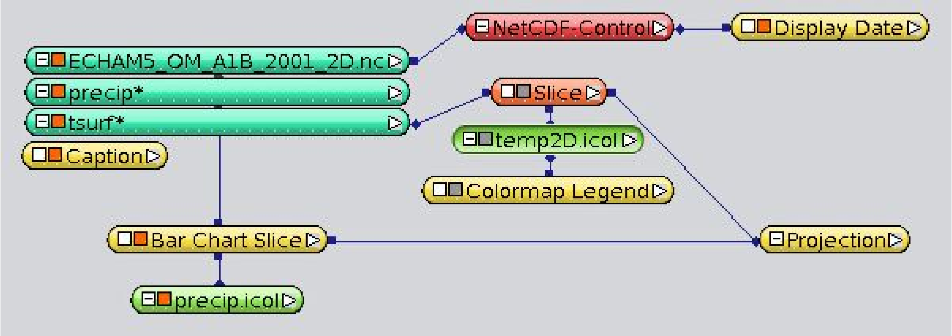
○ V 1.000

A 0.500

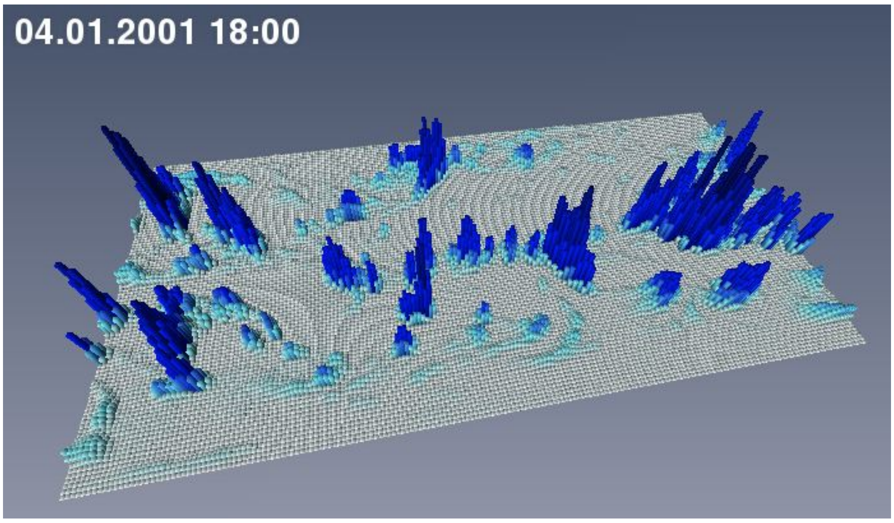
The height of precipitation is now visualized by blue bars.



In practice, you probably would not use such a monochrome diagram. You can also create a colormap and connect it to the *Bar Chart Slice* module in order to add color mapping. Here, we are going to take a pre-defined colormap: right-click into the colormap, select ***Options → Load colormap***, open the tutorial folder and select *precip.icol*.



If you move forward to time step 16, your visualization will look like this:



Global Data Window

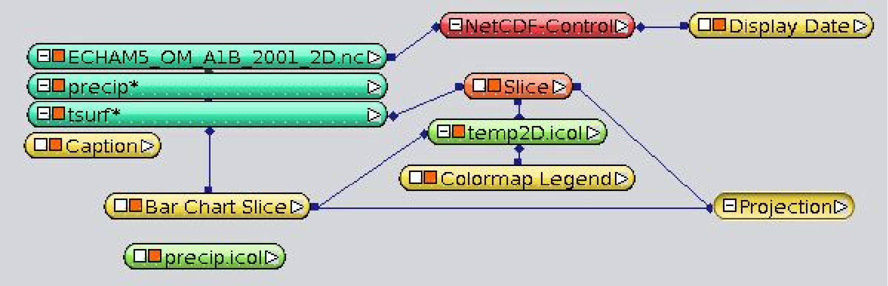
Up to now, the heights of the bars have only been comparable at different locations within one single time step. Bar heights of different time steps are not comparable because their maximum and minimum heights depend on the data range of the individual time step. If you want to compare the bar heights of different time steps, you must define a *Global data window* in the Properties of the corresponding variable (here: *precip*), that covers the whole data range of all time steps.

In case of multivariate data, a very useful option is the use of a colormap to visualize a second variable. For example, a *Bar Chart Slice* showing the absolute precipitation could be colorized by the percentile change of precipitation relative to a certain time period (ColorField). In this case, we could colorize the bars by using a bipolar colormap (e.g. Brown – Orange – White for negative percentages, White - Light Blue – Dark Blue for positive percentages).

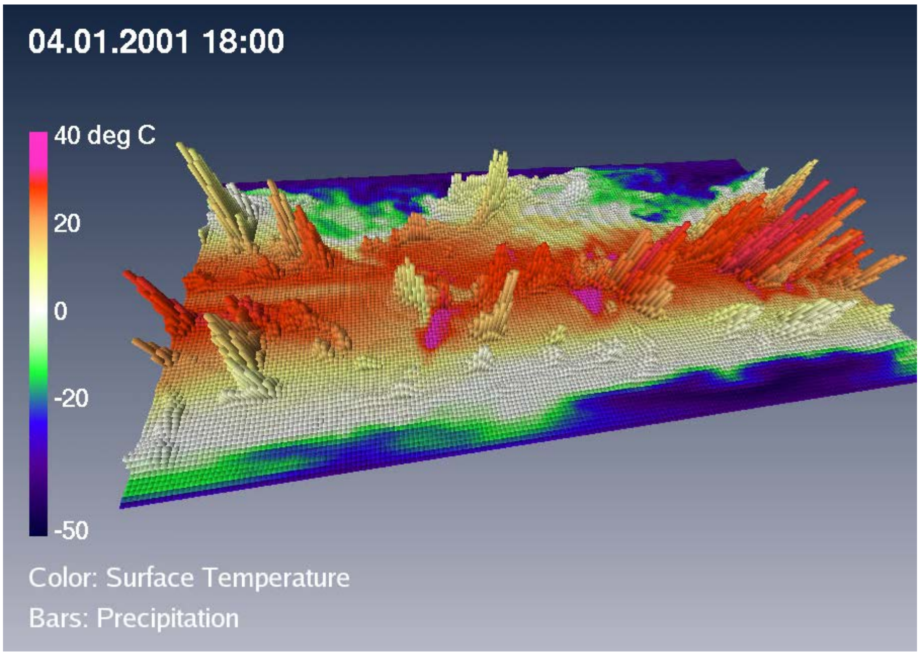
In our tutorial we are not going to visualize such anomalies, but, in order to demonstrate the feature, we would rather use the loaded temperature variable to colorize the *Bar Chart Slice*.

4.7. Bivariate Bar Chart Slice: Precipitation and Temperature

- Switch the temperature colormap and annotation on again. (How? See chapter 3.4)
- In the *Bar Chart Slice* module, click on the small white box, there on ColorField and connect the arising line with the *tsurf** module.
- By clicking on the white box again, connect the Colormap port of the *Bar Chart Slice* with the colormap *temp2D.icol*. The precipitation colormap is unconnected now.
- In the *Bar Chart Slice* module, right-click on the colormap and click on ***Adjust range to → tsurf***.



- Add a second annotation (***Create → Annotations → Caption → Create***). Enter the text "*Bars: Precipitation"* in the Caption Properties and arrange the annotations like this:

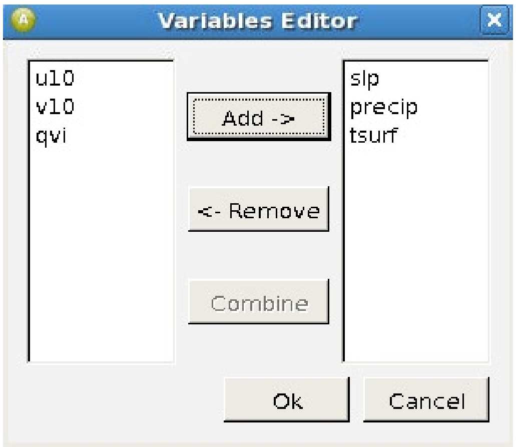


- Start the time animation in the NetCDF-Controls Properties.
- Save your project in your own folder as *temp_prec_2D.hx* (File → Save Project As).

This project can also be found in the tutorial folder (filename *TUT_temp_prec_2D.hx*).

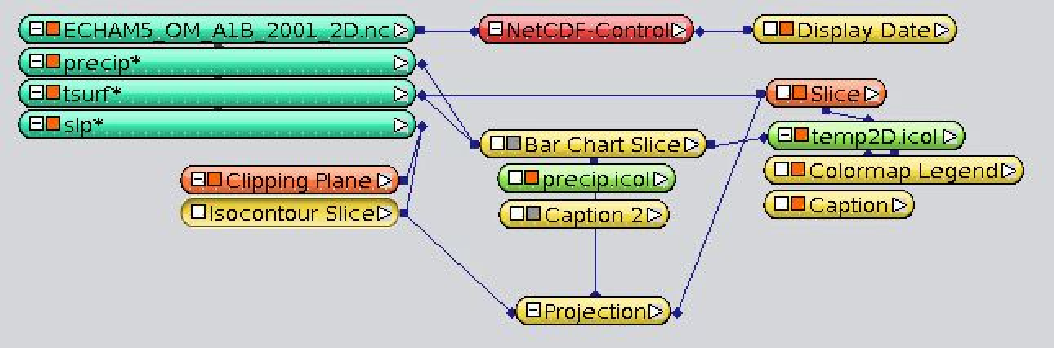
4.8. Sea Level Pressure as Isocontour Slice

- Load the mean sea level pressure as a third variable: Click on the NetCDF file *ECHAM5_OM_A1B_2001.nc* in the Main Panel window, then press ▦ to load the *Variables Editor*, where you can add the variable slp (= sea level pressure).

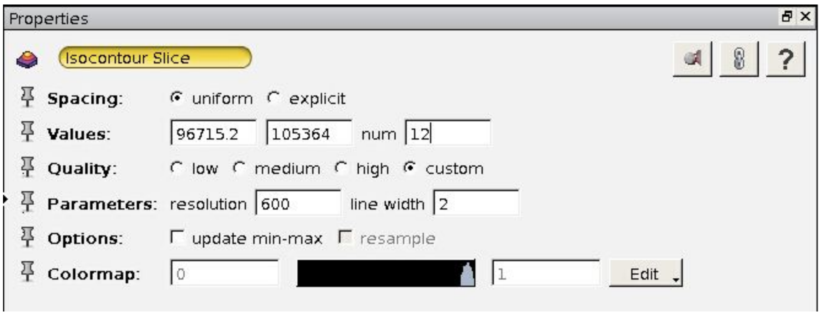


- Switch off the *Bar Chart Slice* and its corresponding Caption 2 via their toggles. Switch on the *Slice* again with the temperature data.

- Right-click the variable *slp**, then ***Display* → *Isocontour Slice* → *Create***. This time, Avizo adds two coupled modules to the Main Panel window – an empty *Clipping Plane* and an *Isocontour Slice* module.
- As usual, you will have to project the new *Isocontour Slice* module first before you can see its visualization properly. Click the projection button to connect it to the *Projection* module.



- In the *Isocontour Slice* Properties change the color from red to black (double-click in the colormap).
- Enlarge the *resolution* to *600*. This will enhance the isoline resolution, but decrease the render performance.
- Keep the *line width* at *2*.
- Set the number of isolines (*num in the Values line*) to *12*. Now *Avizo* evenly divides the value range by this number. As a result, the Isolines are chosen at evenly distributed hPa-values.



Properties

Isocontour Slice

Spacing: ● uniform ○ explicit

Values: 96715.2 105364 num 12

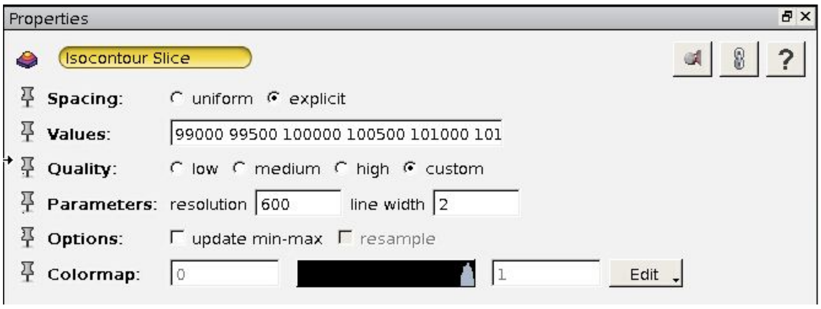
Quality: ○ low ○ medium ○ high ● custom

Parameters: resolution 600 line width 2

Options: □ update min-max □ resample

Colormap: 0 1 Edit

- You can explicitly define the values for which you wish to have isolines on your own by switching the **Spacing** port from *uniform* to *explicit*.
- In the **Values** port enter 13 pressure levels in Pascal (not hPa!) from 99000 to 105000 separated by a blank. You will have to press ENTER at the end before Avizo can apply them.



Properties

Isocontour Slice

Spacing: ○ uniform ● explicit

Values: 99000 99500 100000 100500 101000 101

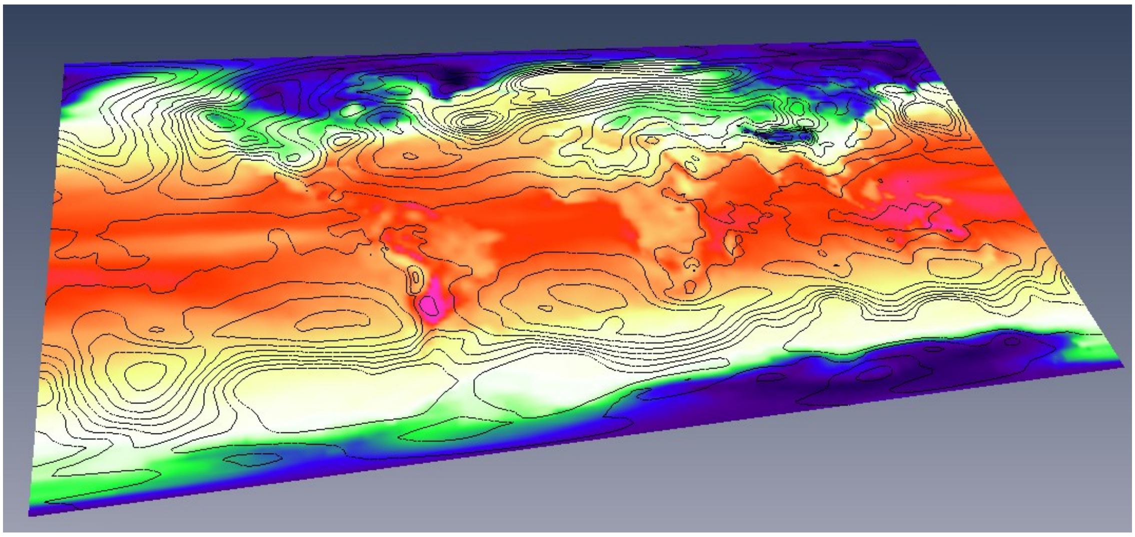
Quality: ○ low ○ medium ○ high ● custom

Parameters: resolution 600 line width 2

Options: □ update min-max □ resample

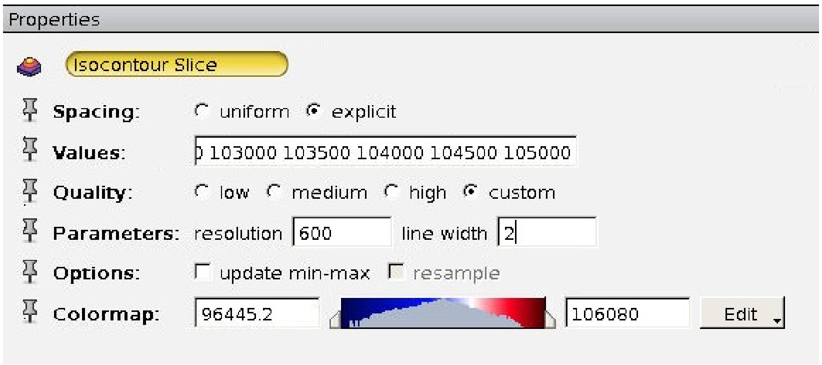
Colormap: 0 1 Edit

In the viewer window, temperature and sea level pressure should be visible somewhat like this (time step 1):



This visualization clearly shows where pressure gradients are high or low. But you won’t be able to quickly see if there is high-pressure or low-pressure in a certain region. Therefore it might be suitable to colorize the Isolines.

- Right-click on the colormap in the *Isocontour Slice* Properties and there click ***Options → Load Colormap***. Load a pre-defined colormap called *sea_level_pressure.icol* from the tutorial folder. High-pressure is displayed in red colors, low-pressure in blue colors, while 1013.25 hPa is displayed in white.



Properties

Isocontour Slice

Spacing: ○ uniform ● explicit

Values: 0 103000 103500 104000 104500 105000

Quality: ○ low ○ medium ○ high ● custom

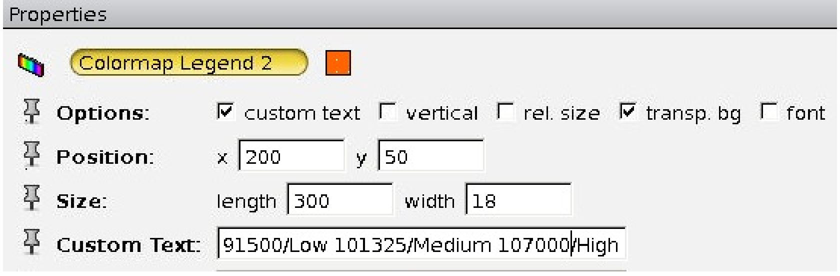
Parameters: resolution 600 line width 2

Options: □ update min-max □ resample

Colormap: 96445.2 106080 Edit

- Add a legend for the sea level pressure in the viewer window by right-clicking on the *sea_level_pressure.icol* module in the Main Panel window; then click ***Annotate → Colormap Legend → Create***.

- You may annotate the second colormap with *custom text, e.g. “High*” for 107000 Pa, “*normal*” for 101325 Pa and “*Low*” for 91500 Pa.



Properties

Colormap Legend 2

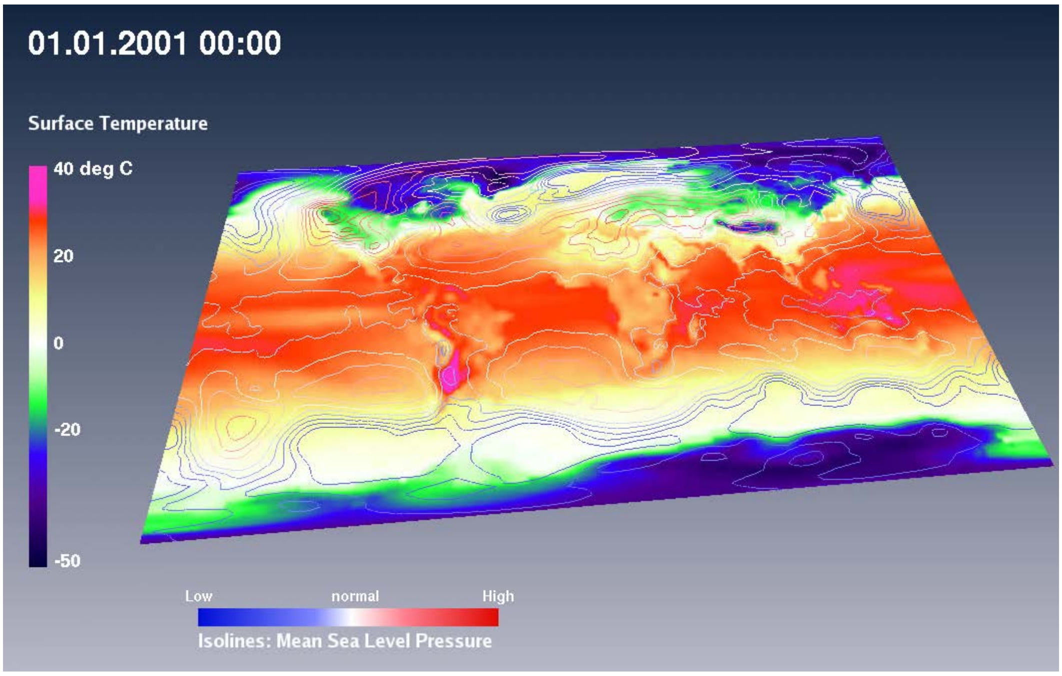
Options: ☑ custom text ☐ vertical ☐ rel. size ☑ transp. bg ☐ font

Position: x 200 y 50

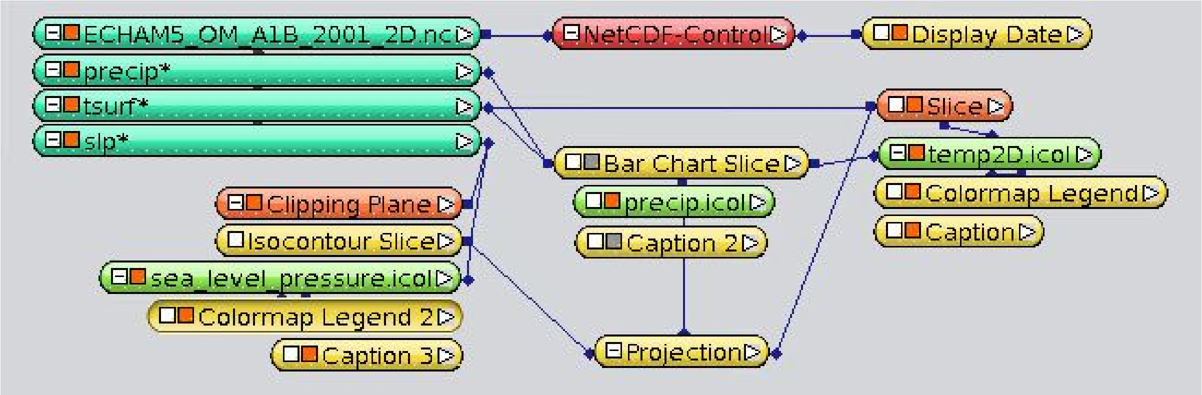
Size: length 300 width 18

Custom Text: 91500/Low 101325/Medium 107000/High

- You may also add an annotation to the legend again (***Create Object → Annotations → Caption***), e.g. “*Isolines: Sea Level Pressure”*.
- Move the position of the two colormaps and its annotations according to the following figure.



Your project structure has become quite complex already, but on the other hand, some of the modules are currently inactive:



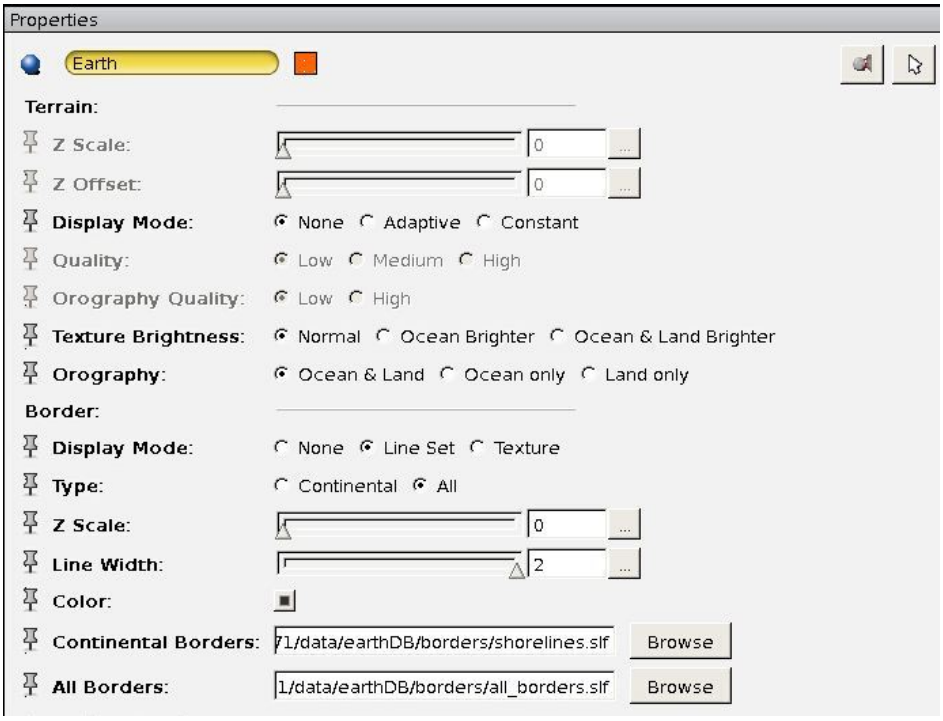
- Save your project as *temp_slp_2D.hx* (File → Save Project As).

This project version is called *TUT_temp_slp_2D.hx* in the tutorial folder.

4.9. Continental and Political Outlines

Until now, in our visualization you could not easily figure out the geographic context of the visualization. For example, coast lines would help to show the geographic mapping of the data. This might be important to compare the temperature and pressure patterns onshore and offshore.

- You can easily add a standard *Earth* module to your project by clicking ***Create Object → Other → Earth → Create*** from the main menu bar.
- As always, connect the *Earth* module to the Projection module by clicking the projection button.
- In the *Earth* Properties, switch off the terrain (*Terrain* > *Display Mode* > *None*), but switch on the coast lines (*Border* > *Display Mode* > *Line Set*).
- Change the **Color** of the continental outlines from blue to dark grey (click in the color rectangle of the *Earth* Properties).
- If you also like to see the political outlines (borderlines), click *Border* > *Type* > *All*.
- Change the **Line Width** of the borders from 1 to 2. In order to do so, you will again have to configure the maximum value of the slider (click on button on the right side next to the line width value).



Properties

Earth

Terrain:

Z Scale: 0

Z Offset: 0

Display Mode: ● None ○ Adaptive ○ Constant

Quality: ● Low ○ Medium ○ High

Orography Quality: ● Low ○ High

Texture Brightness: ● Normal ○ Ocean Brighter ○ Ocean & Land Brighter

Orography: ● Ocean & Land ○ Ocean only ○ Land only

Border:

Display Mode: ○ None ● Line Set ○ Texture

Type: ○ Continental ● All

Z Scale: 0

Line Width: 2

Color: ■

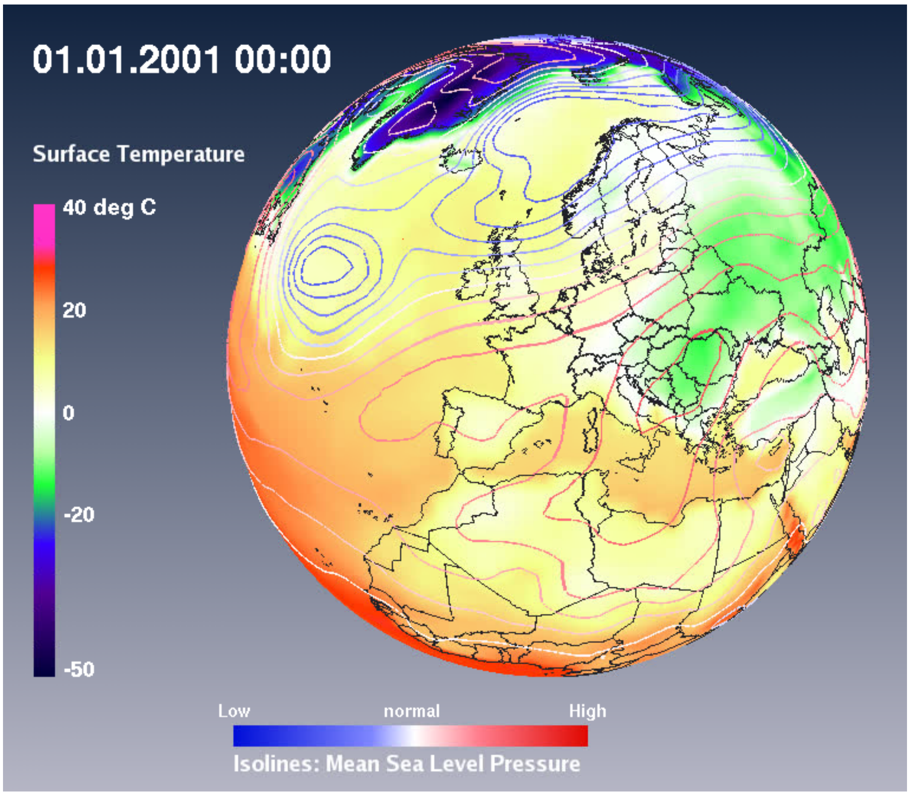
Continental Borders: 71/data/earthDB/borders/shorelines.slf Browse

All Borders: 1/data/earthDB/borders/all_borders.slf Browse

- Change the flat earth into a globe, that is, change to spherical projection (click on the *Projection* module and change the Types entry to *SPHERICAL*, then click *View All* in the navigation bar).

- If not yet done, fill the Pacific gap by activating *Copy first longitude* in the **Misc** port of the *NetCDF-Controls* Properties.
- Save your project (File → Save Project).

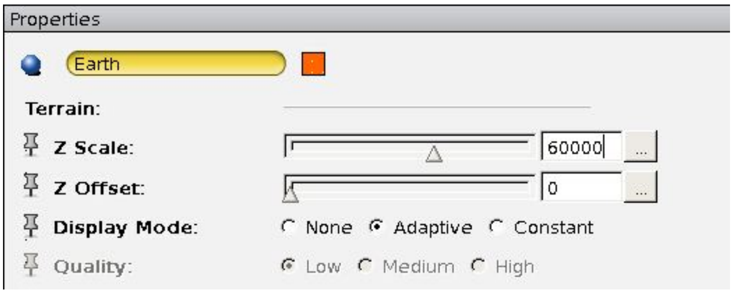
This project version is called *TUT_temp_slp_Earth_2D.hx* in the tutorial folder.



4.10. Topography & Bathymetry

The same Earth module that we just got to know in chapter 4.9 can also be used to construct a background topography and bathymetry for visualizations.

- Start with an empty project and click ***Create Object → Other → Earth → Create*** in the main menu bar.
- Project the earth by clicking the Projection button 🌐.
- In the *Projection* Properties, set the ***Z-factor*** port to *1*.
- In the *Earth* Properties, set ***Terrain*** > ***Z Scale*** to *60000*. To do this, you will have to increase the default max value of the slider e.g. to *100000* (click on the button right to *Z Scale*, the *Slider Dialog will appear*). Topography (onshore heights) and bathymetry (offshore depths) become visible.



Properties

Earth

Terrain:

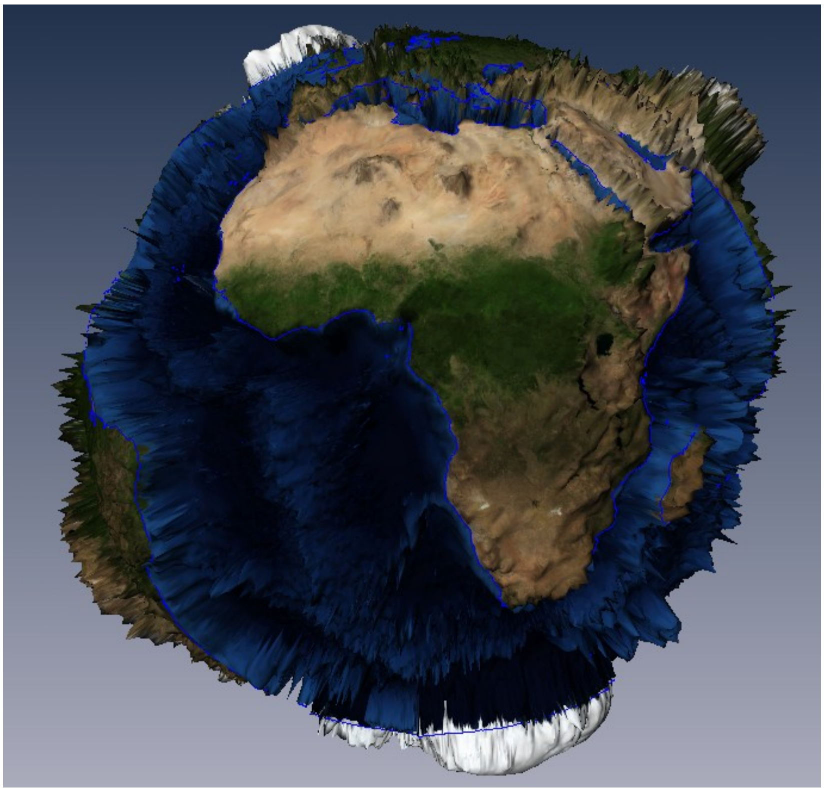
Z Scale: 60000

Z Offset: 0

Display Mode: ○ None ● Adaptive ○ Constant

Quality: ● Low ○ Medium ○ High

- If you further increase the ***Z Scale*****,** the earth begins to look like a hedgehog. In this figure, the *Z Scale* is *200000*.



- Keep the ***Display Mode*** on *Adaptive* resolution.
- Zoom closer. Depending on the display scale, three different resolutions for the topography and bathymetry are loaded.
- The adaptive resolution technique will cause a delay whenever the next level or another tile of the high resolution topography data is loaded while you navigate. Therefore you may want to change the ***Display Mode*** to *Constant > Low resolution* for cases where you don't need the high resolution of the topography data, or *Constant > High resolution* for videos with smooth Camera Paths close to the earth's surface.

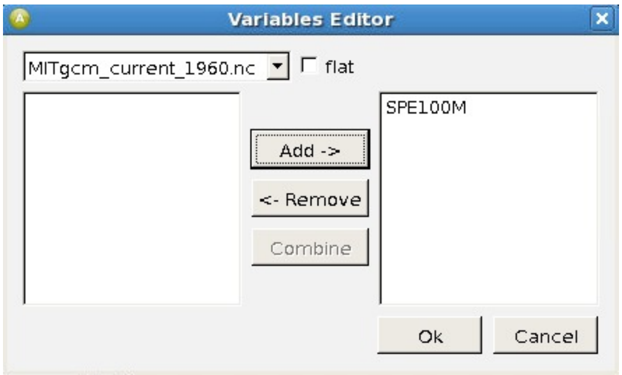
4.11. Velocity Magnitude as Embossed Slice

Let’s build a completely new project now and work with velocity data from an ocean circulation model called MITgcm. You can find further information about this model at

- http://mitgcm.org/

By means of the *Embossed Slice* module, we can visualize slices through scalar data with a combined color and *bump* shading. As a result, the maxima and minima of the 2D data slice will look like a colorized relief structure, like mountains at high data values and valleys at low data values, although the rendering is still only based on 2D slices through the data field. The *Embossed Slice* module efficiently utilizes the capabilities of the visualization server’s 3D graphics card - with the result of a quite high rendering speed.

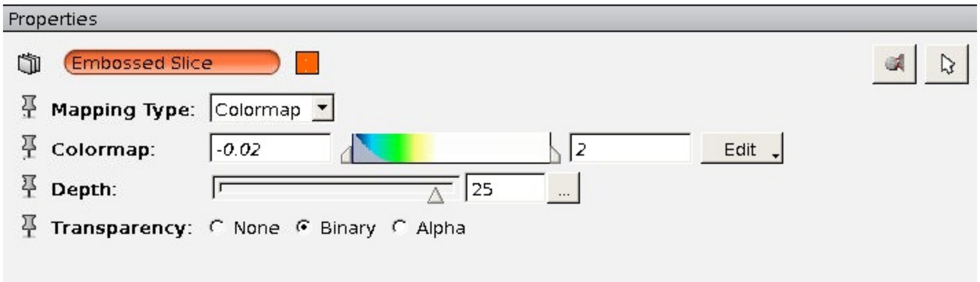
- Start with an empty Avizo project, load the NetCDF file *MITgcm_current_1960.nc* from your tutorial folder and select the variable SPE100M (current speed in 100 meter depth).



MITgcm_current_1960.nc

SPE100M

- Right-click the *SPE100M** variable ***Display → Embossed Slice → Create***
- Project the *Embossed Slice* by clicking the Projection button.
- Load a pre-defined colormap called *current_speed.icol* from your tutorial folder and connect it to the *Embossed Slice* module.
- Keep the Mapping Type *Colormap*.
- Add the earth topography by clicking ***Create Object → Other → Earth → Create*** from the main menu bar and project the *Earth* module as well. In its Properties, window set its scale to 30000 so that topography arises and bathymetry is pushed down to better see the *Embossed Slice*.
- Up to now, the *Embossed Slice* does not show any bump shading yet. Add the bump shading effect by shifting the Depth slider in the *Embossed Slice* Properties. Positive and negative values will result in shadowing from different sides. You might choose a Depth value of *25* for this example.



Properties

Embossed Slice

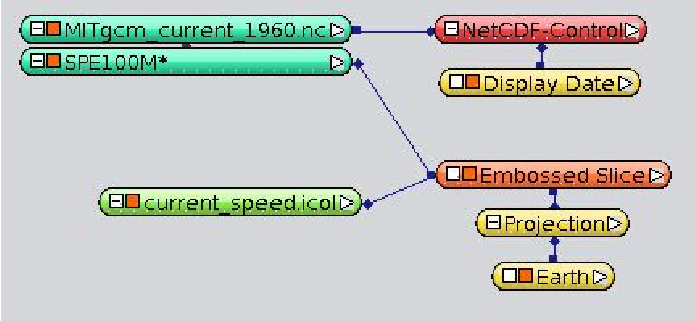
Mapping Type: Colormap

Colormap: -0.02 2 Edit

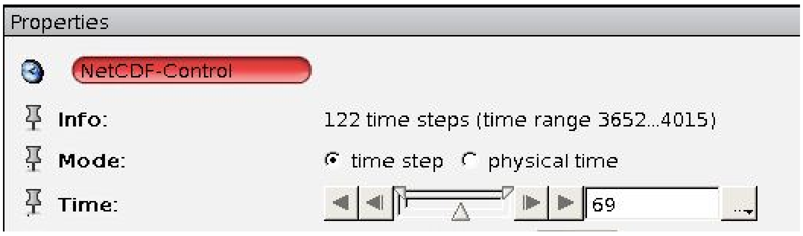
Depth: 25

Transparency: ○ None ● Binary ○ Alpha

- Finally, add time information to your viewer window by right-clicking the NetCDF-Control → ***Annotate*** → ***Display Date*** → ***Create***. This time choose a custom date format *MMM yyyy* which results in *"Jan 1960",* for example.



- You should take some time to look through all the time steps in the NetCDF-Control module by pressing the play button there:



Properties

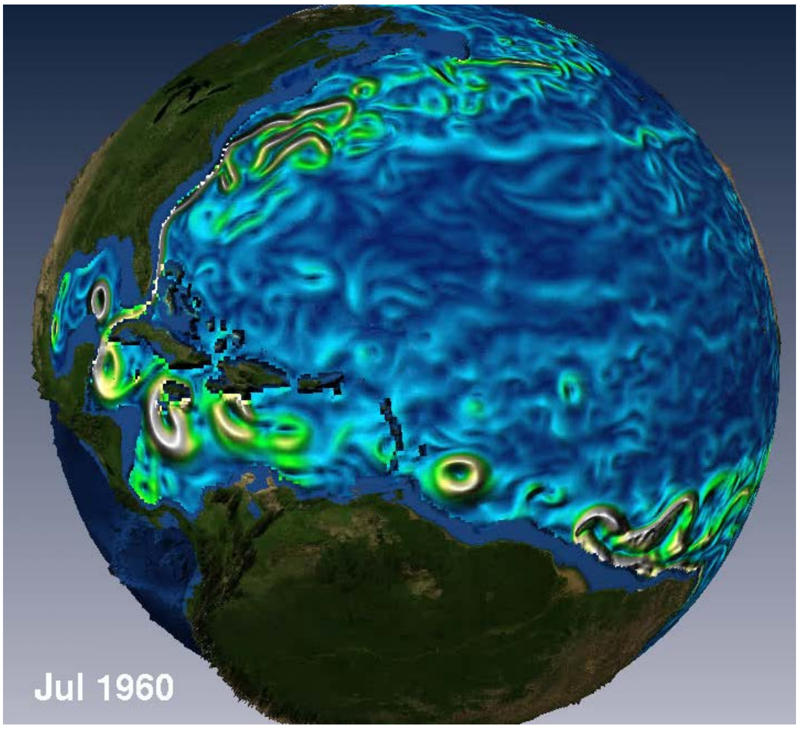
NetCDF-Control

Info: 122 time steps (time range 3652...4015)

Mode: ● time step ○ physical time

Time: 69

The following figure shows the resulting visualization of time step 69:



- Save your project as *current_Embossed_Slice_2D.hx.*

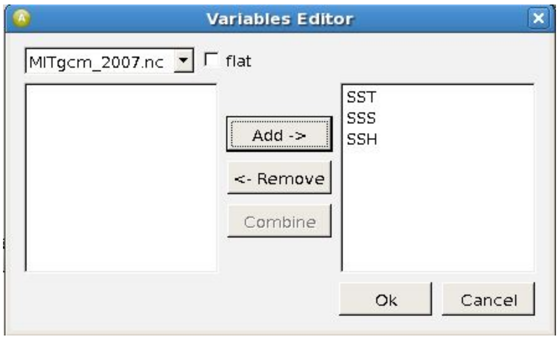
The pre-defined project is called *TUT_current_Embossed_Slice_2D.hx*.

4.12. Sea Surface as Height Map Slice

The *Height Map Slice* module works similar to *Bar Chart Slice*, which we have used in chapters 4.6 and 4.7 to visualize precipitation. Both use the 3D height to visualize 2D scalar data. The main difference: the Height Map Slice module renders a smoothly elevated surface (according to the values of the 2D Input field), whereas the Bar Chart Slice module displays the data as 3D bars. Both modules can be used in a uniform color, colorized according to the input field or with a second variable.

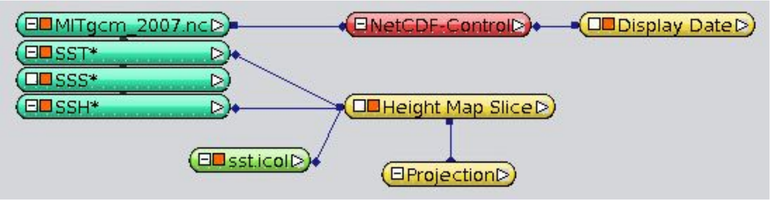
We want to continue to work with the ocean data from the MITgcm model. In this exercise we want to construct a Height Map Slice from the variable sea surface height (SSH) and assign either sea surface temperature (SST) or sea surface salinity (SSS) as a ColorField. In this way we can visualize two variables (either SSH and SST or SSH and SSS) at the same time and examine their interrelation.

- Click Open Data from your Main Panel window, select the NetCDF file MITgcm_2007.nc from your tutorial folder and add all three variables (SST, SSS, SSH) to the right side. Confirm with Ok.
- In the Main Panel window, right-click the variable sea surface height (SSH), then click ***Display → Height Map Slice → Create***.

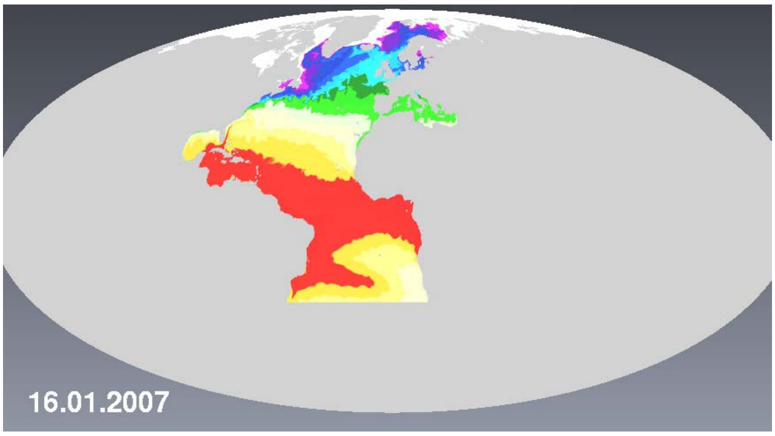


- Click the white small rectangle of the new *Height Map Slice* module, select the input *ColorField* (via the white rectangle on the left side of the module) and connect the line to the SST variable of the NetCDF file. The *Height Map Slice* module should now be connected with two variables.
- Project the *Height Map Slice* module by clicking and choose the Type *MOLLWEIDE* from the *Projection* Properties.
- In the *Height Map Slice* Properties, keep the *shaded* Draw Style. To assign a pre-defined colormap, click ***Edit → Options → Load colormap*** and choose *sst.icol* from your tutorial folder. In the Height Map Slice module click on the colormap and ***Adjust range to → sst***.
- Display the date information by right-clicking the NetCDF-Control module in the Main Panel window ***Annotate → Display Date → Create***. Choose the date format of your choice but deactivate time information since we only have 10 datasets per month and do not need time information to distinguish the time steps.

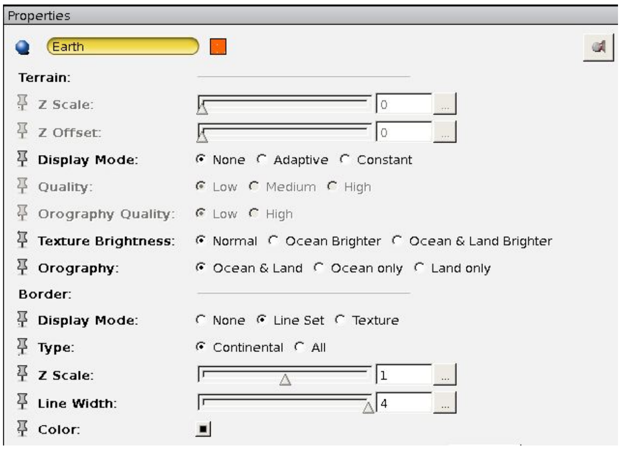
Your Main Panel and viewer window should look similar to the following figures now.



As you can see, our NetCDF file only covers the Atlantic Ocean and the Arctic region. All other oceans and all land areas are filled with missing values, which are visualized here in grey. When you change the Height Map Slice settings to a *transparent* Draw Style, the grey areas would turn transparent. But let's keep the shaded Draw Style for this exercise.



- For better orientation, you might add the coastal shorelines by clicking ***Create Object → Other → Earth → Create*** from the main menu bar, then selecting *Terrain > Display Mode > None* and *Border > Line Set* from the *Earth* Properties. Choose a black color for the shorelines and a *Line Width* of 4. Don't forget to enable the ***Projection*** for this module as well.



Properties

Earth

Terrain:

Z Scale: 0

Z Offset: 0

Display Mode: ● None ○ Adaptive ○ Constant

Quality: ● Low ○ Medium ○ High

Orography Quality: ● Low ○ High

Texture Brightness: ● Normal ○ Ocean Brighter ○ Ocean & Land Brighter

Orography: ● Ocean & Land ○ Ocean only ○ Land only

Border:

Display Mode: ○ None ● Line Set ○ Texture

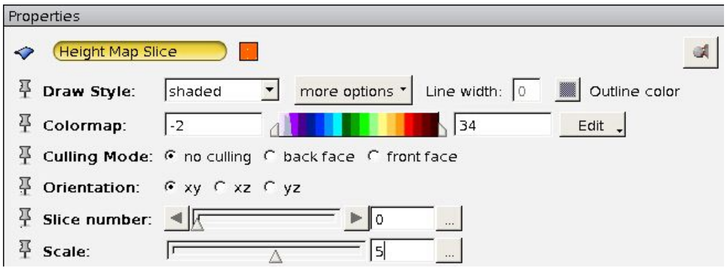
Type: ● Continental ○ All

Z Scale: 1

Line Width: 4

Color: ■

- Up to now, *the Height Map Slice* rendering is flat – which you might have noticed when navigating through your visualization with the trackball. To make use of the central advantages of the *Height Map Slice* module, configure the *Scale* port of the *Height Map Slice* Properties to a Max value of 10 and choose a scale of 5.



Properties

Height Map Slice

Draw Style: shaded | more options | Line width: 0 | Outline color

Colormap: -2 | 34 | Edit

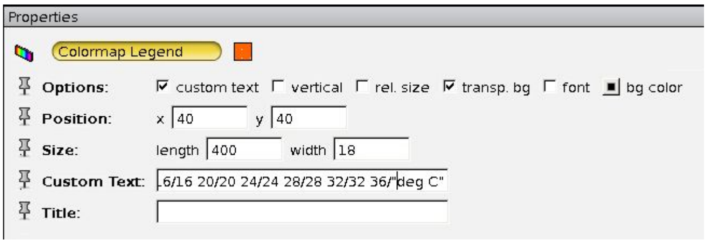
Culling Mode: ● no culling ○ back face ○ front face

Orientation: ● xy ○ xz ○ yz

Slice number: 0

Scale: 5

- Add a colormap legend to your viewer window by right-clicking the sst.icol module in the Main Panel window: ***Annotate → Colormap Legend → Create***. You could set the colormap annotations to steps of 4 degrees and set the parameters as follows:



Properties

Colormap Legend

Options: ☑ custom text ☐ vertical ☐ rel. size ☑ transp. bg ☐ font ■ bg color

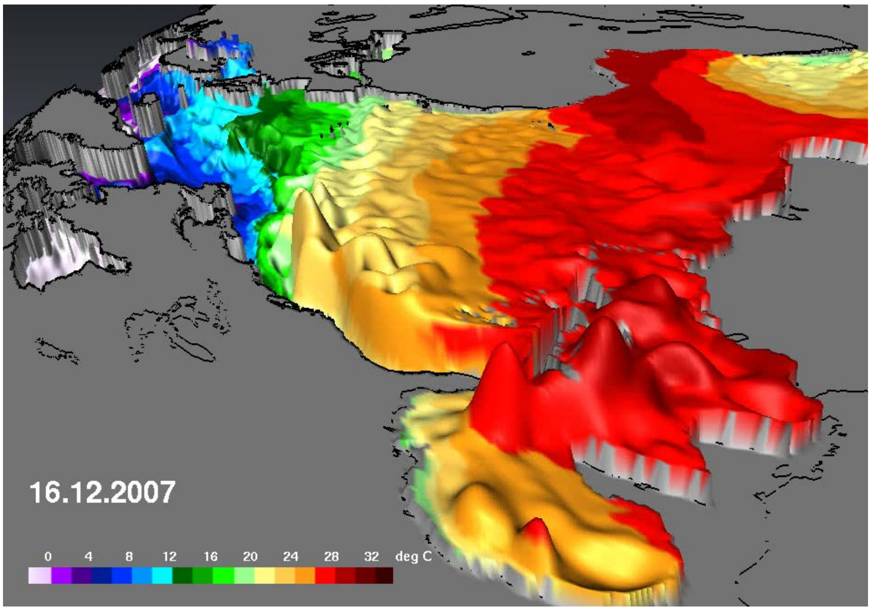
Position: x 40 y 40

Size: length 400 width 18

Custom Text: .6/16 20/20 24/24 28/28 32/32 36/"deg C"

Title:

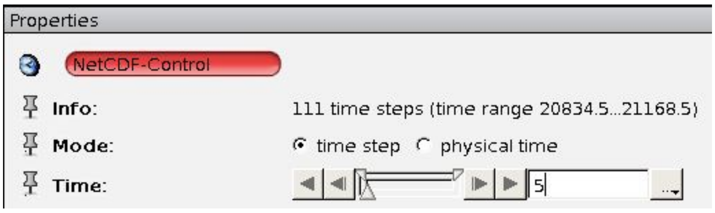
- Take the trackball and turn the visualization to a perspective view from the Caribbean Ocean to Europe, with the eddies in the Gulf of Mexico in the very front, and a straight view along the Gulf stream and its continuation to the north east, the Northern Atlantic stream.



Global Data Window

Similarly to the situation with the *Bar Chart Slice*, the heights of the *Height Map Slice* are only comparable at different locations within one single time step. Maximum and minimum heights depend on the data range of the individual time step. If you want to compare the heights of different time steps, you must define a *Global data window* in the Properties of the corresponding variable (here: *SSH*), that covers the whole data range of all time steps.

- At this point, you should take some time to animate your visualization. In the *NetCDF-Control* Properties, click the Play button of the Time port. As you will realize, this is computationally quite intensive compared to the *Embossed Slice* method: now the frame rate is quite slow. We would first have to export an mpeg movie in order to later look through all time steps quickly enough to be able to directly see the seasonal changes.



Properties

NetCDF-Control

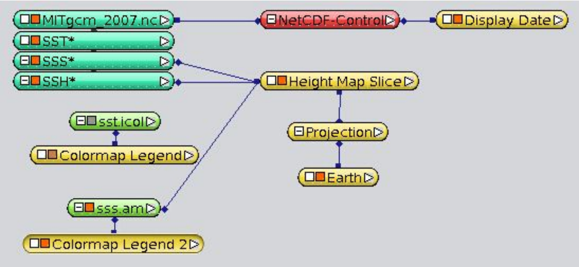
Info: 111 time steps (time range 20834.5...21168.5)

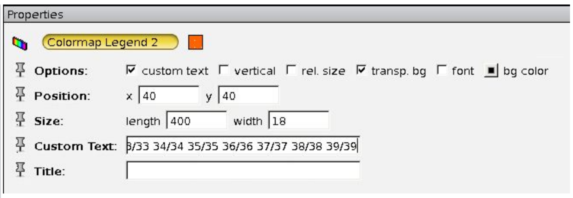
Mode: ● time step ○ physical time

Time: 5

With this project once set, you can easily change the coloring from sea surface temperature (SST) to sea surface salinity (SSS):

- Simply take the connection between *the Height Map Slice* and the *SST* variable and connect it to the *SSS* variable instead.
- Due to the different data range and meaning you will have to assign a different colormap: Click *Open Data*, select the colormap *sss.am* from your tutorial folder and connect the *Colormap* port of the Height Map Slice module to *sss.am*.
- Switch off the *sst.icol* colormap, display the *sss.am* colormap instead and define the parameters in the *Colormap Legend 2* Properties as follows:




Properties

Colormap Legend 2

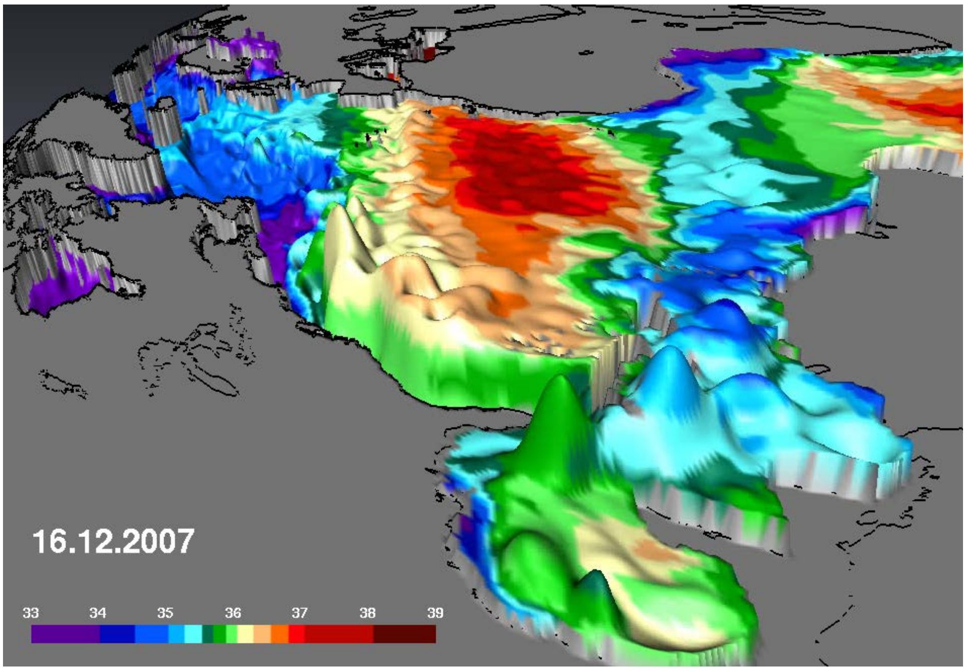
Options: ☑ custom text ☐ vertical ☐ rel. size ☑ transp. bg ☐ font ■ bg color

Position: x 40 y 40

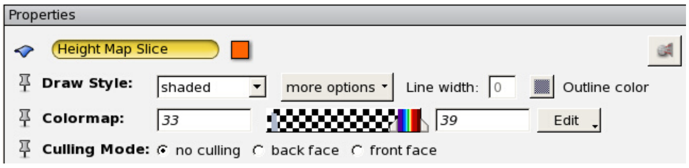
Size: length 400 width 18

Custom Text: 3/33 34/34 35/35 36/36 37/37 38/38 39/39

Title:



- Adjust the range to SSS: in the *Height Map Slice* module right-click on the colormap > *Adjust range to* > *SSS* and set the minimum value to 33*.*


Properties

Height Map Slice

Draw Style: shaded | more options | Line width: 0 | Outline color

Colormap: 33 | 39 | Edit

Culling Mode: ● no culling ○ back face ○ front face

Tip: When rendering a *Height Map Slice*, Avizo also displays an orange boundary line, which may disturb you depending on the chosen projection. To remove this line, type into the console:

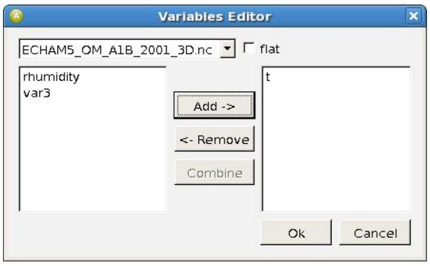
```
“Height Map Slice” frame off
```

- Save your project as *MITgcm_Height_Map_Slice.hx*.

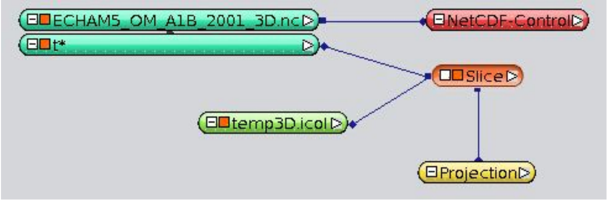
As usual, you will find this project pre-defined, called *TUT_MITgcm_Height_Map_Slice.hx* in the tutorial folder.

4.13. 3D Temperature on a movable Slice

- Start with an empty project and load the NetCDF file *ECHAM5_OM_A1B_2001_3D.nc* (*Open Data* button).
- In the *Variables Editor* select the variable *t* (temperature). As usual in the Main Panel window, three modules are added: one for the NetCDF file, one for the variable *t** and one *NetCDF-Control* module.

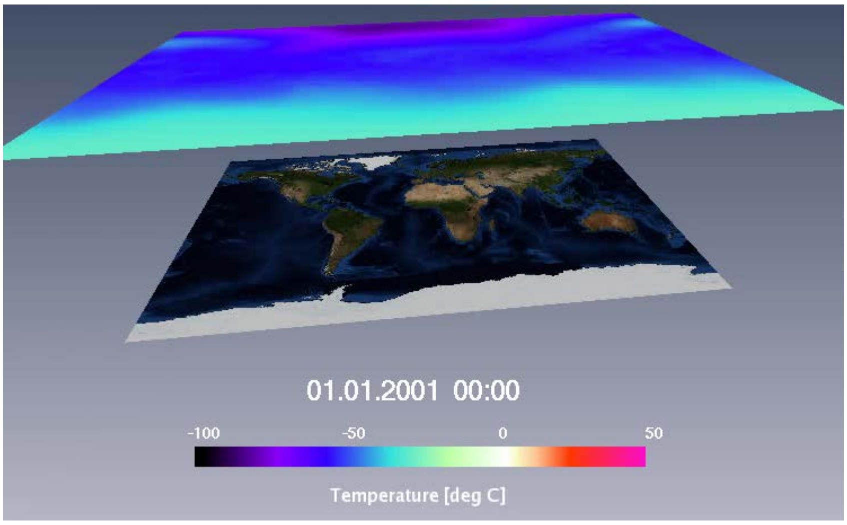


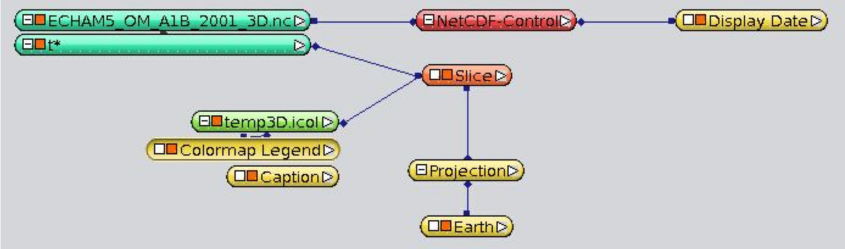
- Right-click the *t** variable module, and then click ***Display* → *Slice* → *Create***.
- Project the Slice by clicking . Select the projection EQUIDISTANT_CYLINDRICAL for this exercise and reduce the **Z factor** to *0.5*.
- Let's colorize the Slice in the module's Properties panel: As **Mapping Type** select *Colormap* (default), then click in the Colormap entry on *Edit > Options > Load colormap* and load the colormap *temp3D.icol* from the tutorial directory.



Question: Why is the colormap *temp2D.icol*, which we have used in the earlier exercises, not suitable for our 3D data? Answer

- Take the trackball to move the camera to a more declined perspective and move the slider of the **Translate** port to the right. You can move the Slice up and down and visualize the temperature at 17 different pressure levels now.
- When you add the *Earth* module (***Create Object* → *Other* → *Earth* → *Create***), display date and time information (right-click the variable *NetCDF-Control* → ***Annotate* → *Display Date* → *Create***), add a colormap legend entry (right-click temp3D.icol → ***Annotate* → *Colormap Legend* → *Create***) and include an annotation. (***Create Object* → *Annotations* → *Caption* → *Create***).
- Your visualization should look like this:



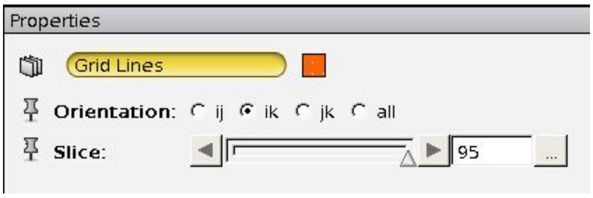


- Save your project as temp_movSlice_3D.hx.

The example tutorial project is called *TUT_temp_movSlice_3D.hx*.

4.14. Bounding Box & Grid

- In the Main Panel window, right-click *t** and then ***Annotate* → *Bounding Box* → *Create***.
- Project the *Bounding Box* with . In the viewer window, orange boundary outlines appear, representing the spatial extent of the 3D data set.
- Additionally, add a visualization of the grid by right-clicking *t** and then ***Display* → *Grid Lines* → *Create***.
- Apply the projection to the grid with .
- In the corresponding Properties, choose the **ik** orientation instead of the default **ij** orientation. The grid is now displayed at the front side of the *Bounding Box*.
- You may move the grid to the back by selecting **Slice** 95.
- If not yet done, move the *Slice* a bit down again in its slider in order to be able to see the back side of the grid.
- You can create a second grid for the left side with a second *Grid Lines* module (**Orientation** *jk*, **Slice** *0*). Don't forget the projection…

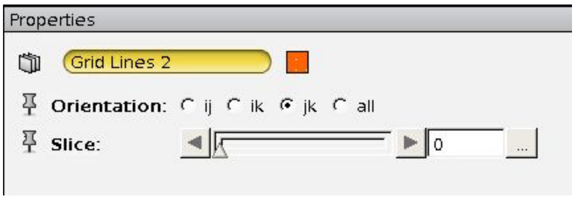


Properties

Grid Lines

Orientation: ○ ij ● ik ○ jk ○ all

Slice: 95

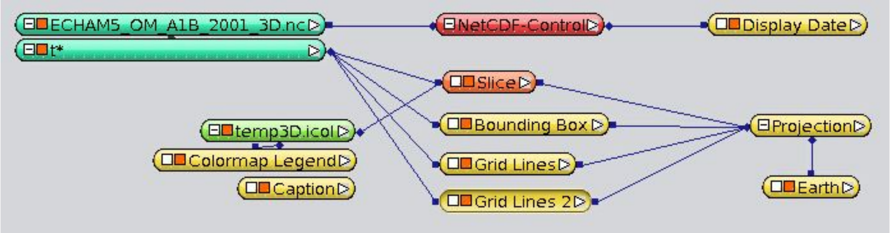


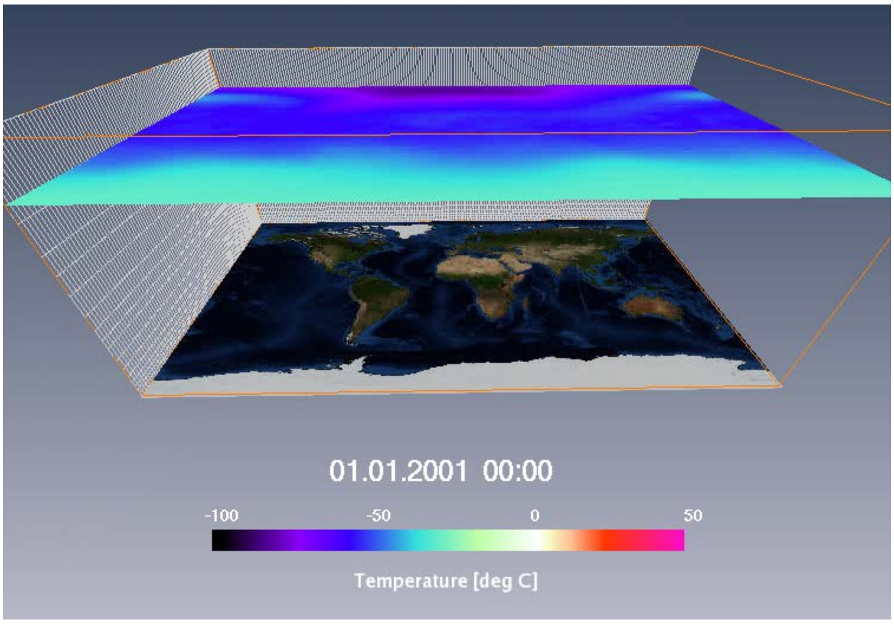
Properties

Grid Lines 2

Orientation: ○ ij ○ ik ● jk ○ all

Slice: 0





4.15. Ortho Slice, Slice, Embossed Slice – Comparison of 2D Slices

Avizo offers three different modules for 2D slices in 3D space. In chapters 4.1 and 4.13 we got to know the *Slice*, in chapter 4.11 the *Embossed Slice*. Additionally, there is an *Ortho Slice*. What are the differences?

Ortho Slice and *Slice* are quite similar, as both of them can operate in three slice orientations (xy, xz, yz). But they differ concerning the grids they accept as input: Ortho Slice can only be used in uniform (regular) grids and is thus not available at all when your data grid is not regular (uniform coordinates). *Slice* is applicable for rectilinear and curvilinear grids, as it internally executes a resampling to a regular grid first.

Since our ECHAM5-OM data uses a rectilinear grid, we cannot create an *Ortho Slice*. *Embossed Slice* differs from *Ortho Slice* and *Slice* in some ways: First, only horizontal slices (xy-direction) are supported. Second, it always interpolates the color between the grid points. Its advantage is (third), that high gradients can much better be visualized due to its shading technique. Also, the *Embossed Slice* shading efficiently makes use of the graphics hardware and is very fast compared to the other modules.

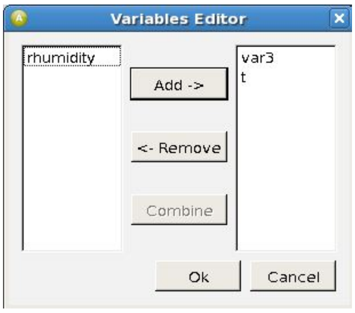
Table 1: Comparison of Ortho Slice, Slice and Embossed Slice.

Module	**Spatial Orientations**	**Remarks**
Ortho Slice	xy, xz, yz	“uniform coordinates” only (= regular grids, grid cells are squares or cubes). For non-uniform grids, Ortho Slice is not available.
Slice	xy, xz, yz	Applicable for “rectilinear coordinates” (grid cells are rectangles or cuboids) and with „curvilinear coordinates” (grid boundaries are curved). Automatic resampling on a uniform grid, optionally in 4 spatial resolutions from “coarse” to “finest”. In case of “uniform coordinates”, adoption of the input-grid, no change of the spatial resolution possible.
Embossed Slice	only xy	Applicable for “rectilinear cords” (grid cells are rectangles or cuboids) and with „curvilinear cords” (grid lines are curved). No display of single-colored grid cells but interpolation between grid points with continuous smooth color interpolation. Possibility of Shading for height and depth reliefs.

4.16. Combination of several Slices

When you think it might help to understand your data, you can combine multiple combinations of slices within one display, e.g. one slice for each 2D plane. We will try this for the wind velocity and visualize the two west wind zones of the northern and southern hemisphere this way.

- Activate the NetCDF file *ECHAM5_OM_A1B_2001_3D.nc* in your Main Panel window, open the *Variables Editor*, add variable *var3* (wind speed) to the right side list and confirm with *Ok*.



Variables Editor

rhumidity

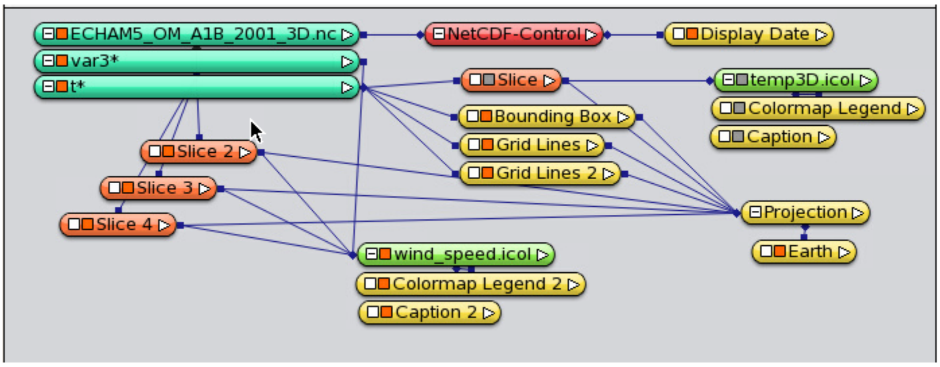
var3
t

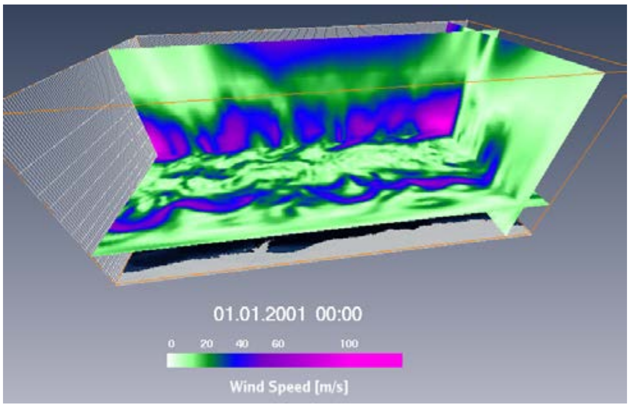
- Switch off the *Slice* of the temperature variable as well as its colormap and annotation.

- Create three new *Slices* connected to *var3** and project them all. In their Properties panels, select the following parameters:

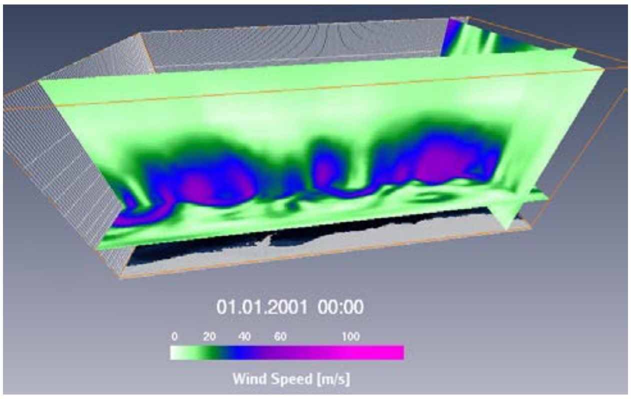
	Orientation	*Mapping Type*	*Translate*
Slice 2	*xy*	*Colormap*	*4*
Slice 3	*xz*	*Colormap*	*67*
Slice 4	*yz*	*Colormap*	*180*

- To colorize them, load the pre-defined colormap *wind_speed.icol* from the tutorial directory into your Main Panel window. Now click on the white square of the var3 module and select "SharedColormap". Connect the rubberband line with the colormap module. Notice that all Slice modules are now connected to the same colormap.
- Display the colormap *wind_speed.icol* as a horizontal legend entry (right-click *wind_speed.icol* → ***Annotate*** → ***Colormap Legend*** → ***Create***) with annotations at 0, 20, 40, 60, 100 and 130 m/s.
- Add the title *Wind Speed [m/s]* for the colormap (***Create Object*** → ***Annotations*** → ***Caption*** → ***Create***), so that, finally, your *Main Panel* and *Viewer* windows look like these:

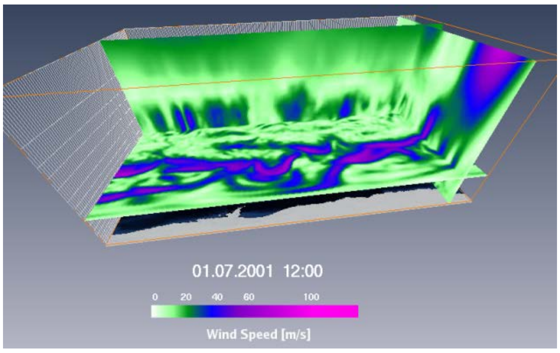


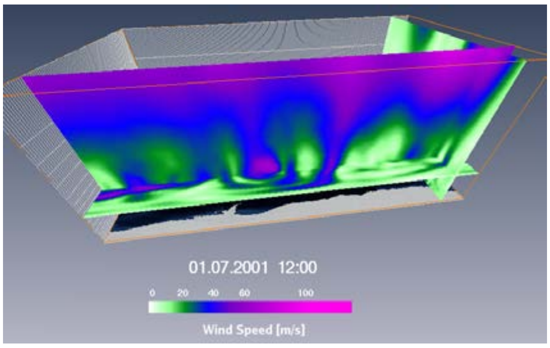


- Try to get an overview of the 3D wind field by moving the three *Slices* up and down, left and right, to the front and to the back. The jet streams in the upper troposphere of the Northern and Southern hemisphere are very eye-catching. In the arctic stratosphere you can also see a polar eddy in this winter situation, whereas wind speeds in the Antarctic stratosphere are lower.



- Move the time slider of the *NetCDF-Control* Properties to a summer situation, e.g. 1st Jul 2001. Now the wind speeds in the arctic stratosphere are quite small, whereas the Antarctic stratospheric eddy is extremely strong.





- Save your visualization as *windspeed_3Slices_3D.hx*.

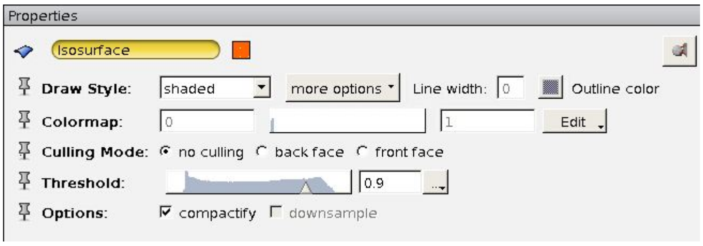
In the tutorial folder, *TUT_windspeed_3Slices_3D.hx* provides you with a current version of the project.

4.17. Isosurface

Isosurfaces are the 3D equivalents to Isocontours in 2D. We are going to visualize the relative humidity as an isosurface with varying thresholds.

- Switch off the three *Slice*s of the wind velocity as well as the corresponding *Colormap Legend* and *Caption*.
- Activate the NetCDF file *ECHAM5_OM_A1B_3D.nc*, open the *Variables Editor* and add the variable *rhumidity* (relative humidity) to the Main Panel window.
- Right-click *rhumidity**, then ***Display* → *Isosurface* → *Create*** and project the *Isosurface* .

- In the **Threshold** port of the *Isosurface* Properties choose 0.9 to visualize the atmospheric areas with a relative humidity of 0.9 or more (≥ 90%).
- Click ***Apply*** to compute this *Isosurface*, or click *auto-refresh* to tell Avizo to recompute it after each parameter change automatically.
- You may change the default yellow color to white, which is closer to our idea of cloudy areas (double-click into the colormap).
- Try different **Draw Styles**, particularly, *shaded* and *transparent*.



Properties

Isosurface

Draw Style: shaded | more options | Line width: 0 | Outline color

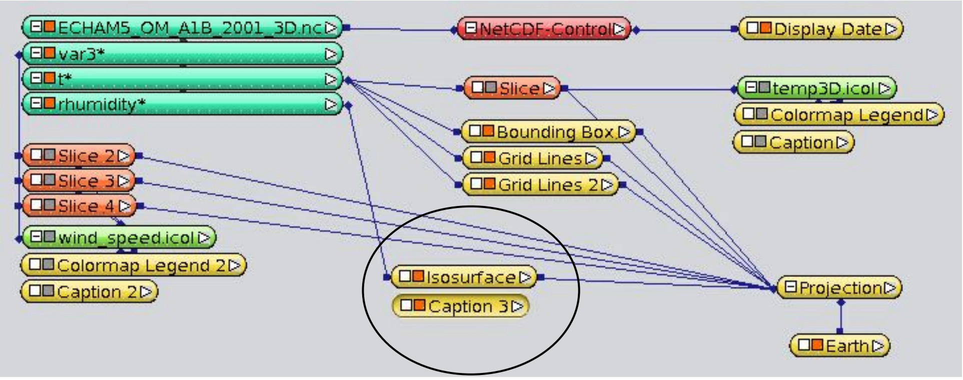
Colormap: 0 | 1 | Edit

Culling Mode: ● no culling ○ back face ○ front face

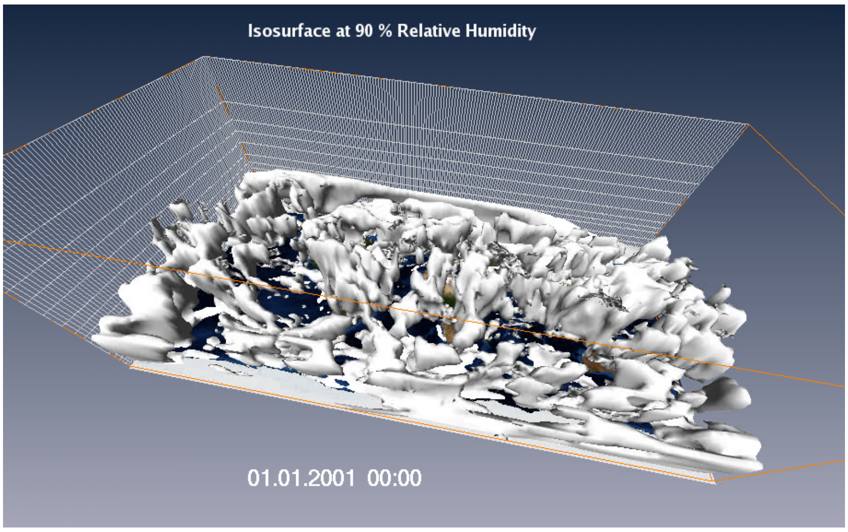
Threshold: 0.9

Options: ☑ compactify ☐ downsample

- Annotate with „*Isosurface at 90 % Relative Humidity*" (Create > Caption).



- Use the mouse to turn the figure to get a better 3D idea of the spatial distribution of the relative humidity.



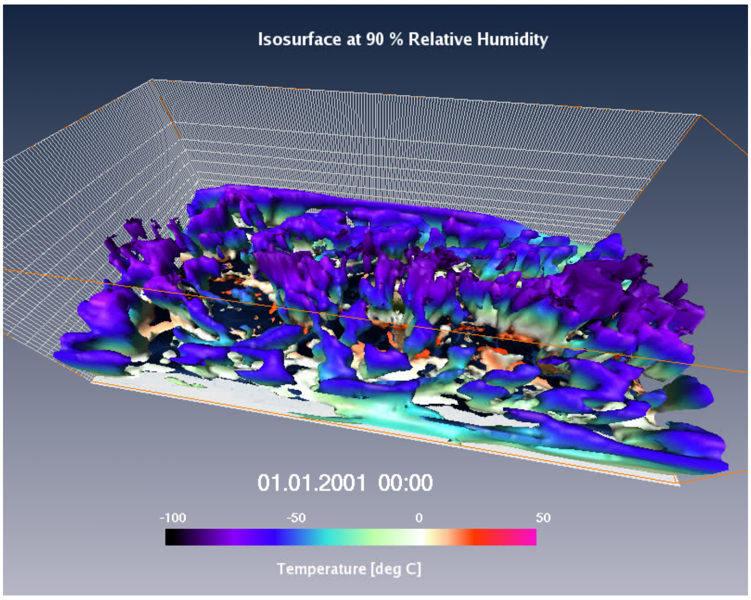
At a glance you can see that a relative humidity of 90% only occurs in the troposphere, as in the upper grid levels no isosurfaces are displayed at all.

- Move the Threshold slider further to the right to 1.0 (100% humidity) and observe that the isosurface is getting smaller and smaller.
- In contrast, at a threshold of 50%, it gets quite large and you can even recognize an isosurface in the stratosphere at the North Pole.
- Save this visualization as *relhum_Isosurface_3D.hx*.

The tutorial folder provides you with the corresponding project *TUT_relhum_Isosurface_3D.hx*

4.18. Bivariate Visualization: Isosurface & Temperature

Similar to the visualizations we have already done with the *Bar Chart Slice* and the *Height Map Slice* modules, we can also colorize the *Isosurface* according to the values of a second variable. We are going to visualize the temperature of the 90%-isosurface here.



- In the Main Panel window, click into the small white box of the *Isosurface* module, then on ColorField and connect the arising line with the variable *t* (temperature).
- Connect the colormap port of the *Isosurface* module with the colormap *temp3D.icol* which is also already loaded in your project. Don't forget to adjust the range. Switch on the temperature colormap and annotation again.

- Save your project (File → Save Project).

This stage of the project is provided as *TUT_relhum_Isosurface_ColorTemp_3D.hx* in the tutorial folder.

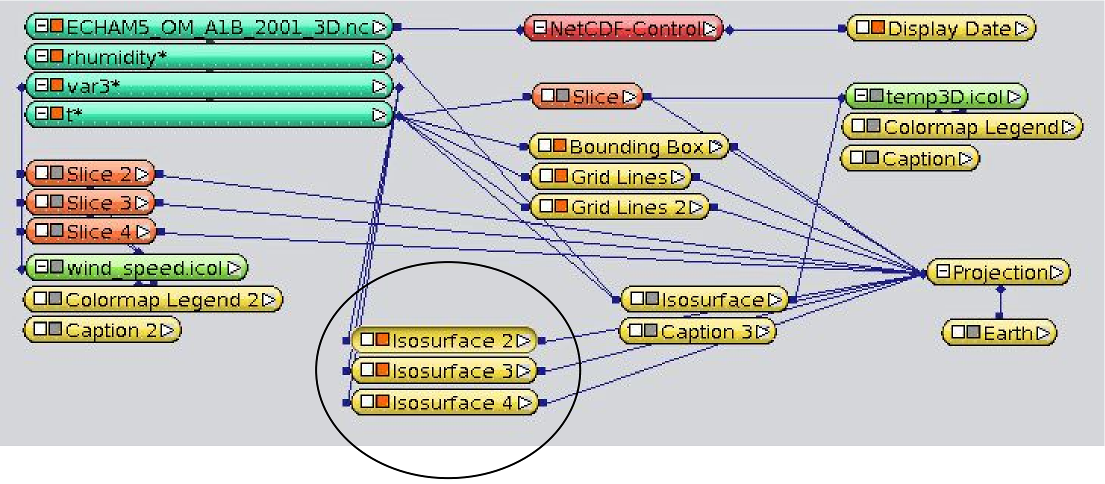
4.19. Nested Isosurfaces

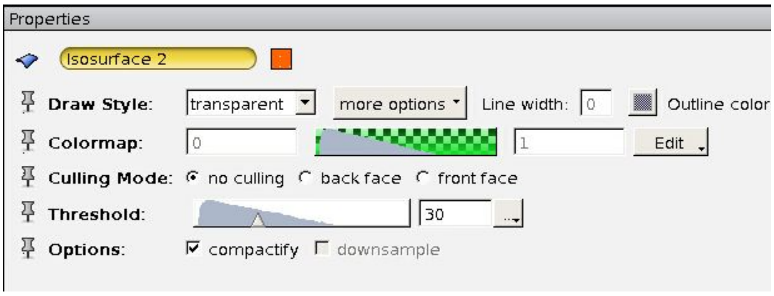
In 2D you can easily visualize a high gradient by displaying isolines for many different values. By definition, the isolines don't obscure or cross each other. Doing the same with isosurfaces in 3D is a bit more complicated. Isosurfaces for different values of the same quantity will enclose each other.

Nevertheless, you can *nest* several isosurfaces if you reduce the opacity of the outer ones. It also might be a good idea to use different colors for the different isosurface values additionally. This technique helps in visualizing the spatial distribution of the field for different values at the same time.

We are going to nest three isosurfaces for the wind velocity (variable var3) which already exists in our project.

- Switch off the rhumidity* *Isosurface*, its *Caption*, the temperature *Colormap Legend* and its caption, and for this example also the *Earth* module.
- Right-click *var3** (wind velocity), create three Isosurfaces and project them. In the Main Panel window they are now called *Isosurface 2*, *Isosurface 3* and *Isosurface 4*.
- Choose a threshold of *30 m/s* for *Isosurface 2*, a *transparent* Draw Style and a transparent green (double-click into the colormap). Then click *Apply* and *auto-refresh*.





Properties

Isosurface 2

Draw Style: transparent | more options | Line width: 0 | Outline color

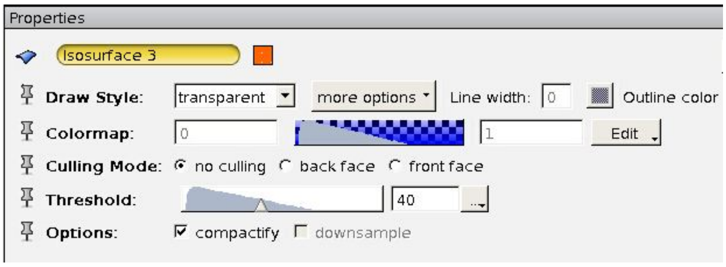
Colormap: 0 | 1 | Edit

Culling Mode: ● no culling ○ back face ○ front face

Threshold: 30

Options: ☑ compactify ☐ downsample

- Define a threshold of *40 m/s* for *Isosurface 3*, choose the *transparent* Draw Style and a transparent blue in the colormap. Again press *Apply* and *auto-refresh*.



Properties

Isosurface 3

Draw Style: transparent | more options | Line width: 0 | Outline color

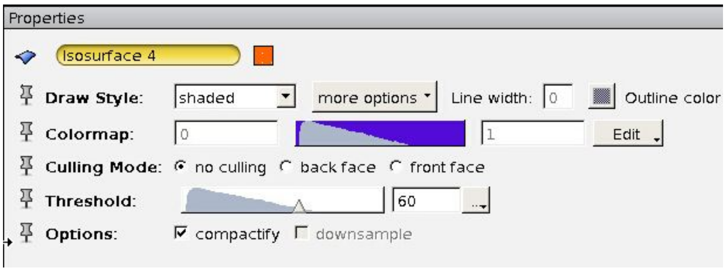
Colormap: 0 | 1 | Edit

Culling Mode: ● no culling ○ back face ○ front face

Threshold: 40

Options: ☑ compactify □ downsample

- Isosurface4 shall include all regions with a minimum velocity of *60 m/s*. Since this isosurface will be the inner one in this case, choose the Draw Style *shaded*. After double-clicking the colormap, select a purple with full opacity. Then click *Apply* and *auto-refresh*.



Properties

Isosurface 4

Draw Style: shaded | more options | Line width: 0 | Outline color

Colormap: 0 | 1 | Edit

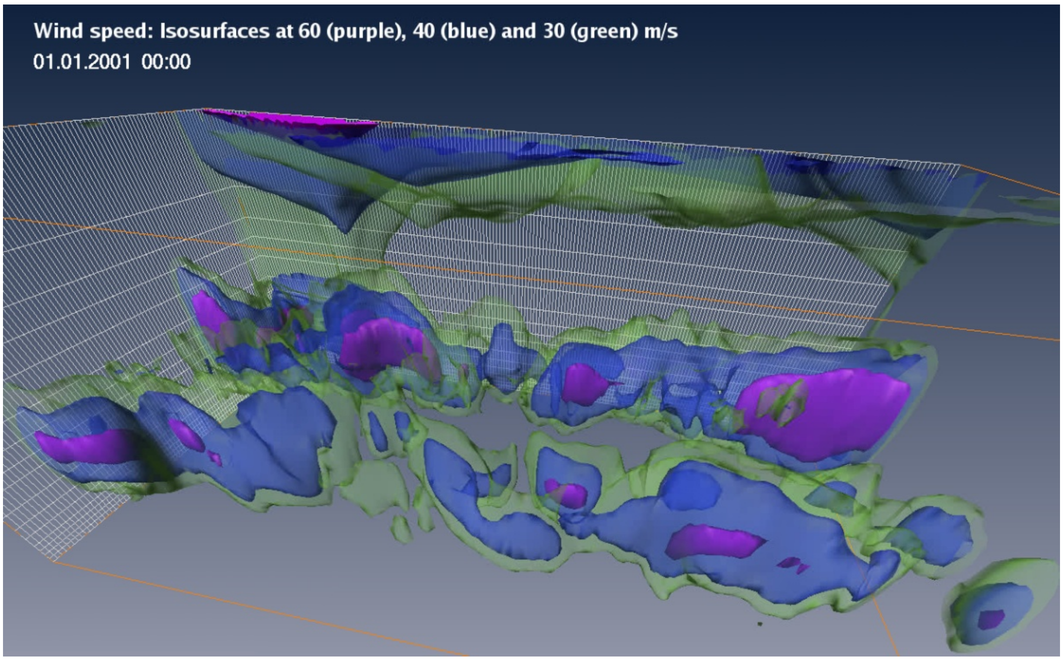
Culling Mode: ● no culling ○ back face ○ front face

Threshold: 60

Options: ☑ compactify □ downsample

- You may have noticed that the semi-transparent isosurfaces are not yet rendered correctly. The semi-transparent surfaces enclosed by others are not visible. Avizo has different ways to handle transparency; the default setting is “Blend Delayed”, which is relatively fast and sufficient for only one semi-transparent interface in the line of sight. For this example, however, you will have to select the mode “Sorted Layer Delayed” to achieve a correct rendering of multiple semi-transparent interfaces: In the main window’s menu, select ***View → Transparency → Sorted Layers Delayed.*** Switch back and forth to see the difference.

- In a top middle position of the viewer window add an annotation (Create > *Caption*) saying “*Wind speed: Isosurfaces for 60 (purple), 40 (blue) and 30 (green) m/s*“.


Wind speed: Isosurfaces at 60 (purple), 40 (blue) and 30 (green) m/s
01.01.2001 00:00

High wind velocities in the northern and southern west wind zones are quite eye-catching this way. Additionally, in the upper levels of the northern hemisphere, there are some flat isosurfaces lying on top of each other. Wind velocity increases again with height.

- Save your project as *windspeed_IsosurfNest_3D.hx*.

As usual, you'll also find this project in the tutorial directory:
TUT_windspeed_IsosurfNest_3D.hx

4.20. Annotated Isolines

In chapter 4.8 we have used *Isocontour Slice* already. *Avizo Green* offers a second variant: *Isocontour Annotated Slice*. However, keep in mind that isoline annotations are quite useful e.g. for screenshots in print publications, but absolutely unsuitable for animations. Used within animations, annotations will change their position at each time step, resulting in a kind of chaotic visualization.

We are going to add *Isocontour Annotated Slice* to the nested isosurfaces visualization of the west wind zones of the last chapter.

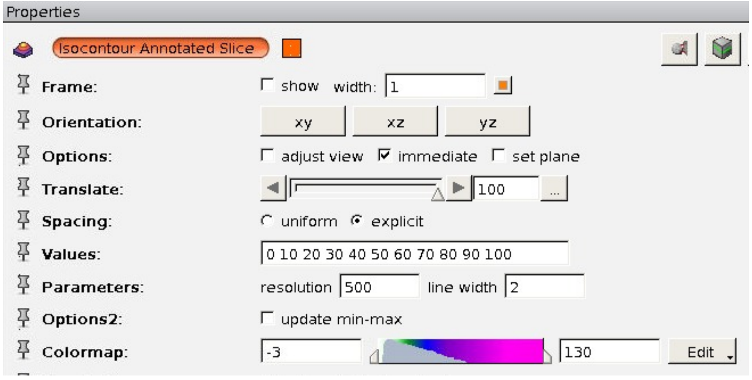
- In the Main Panel window right-click *var3** (wind velocity), then ***Display*** → ***Isocontour Annotated Slice*** → ***Create*** and enable the *Projection* for the new module.
- You may display the *Isocontour Annotated Slice* in the same colormap as the *Slice*s of chapter 4.16. Click into the white box of the module *Isocontour Annotated Slice*, then on *Colormap* and connect the arising line with the colormap *wind_speed.icol*. Make sure that *wind_speed.icol* is switched on (red box).

Properties: Adjustment of the Isolines

- In the *Isocontour Annotated Slice* Properties, first change the **Spacing** from *uniform* to *explicit* and enter the **Values** for which isolines shall be visualized. Separate the values by blanks:

```
10 20 30 40 50 60 70 80 90 100  (ENTER)
```

- Select **Orientation** *yz* to allow a side view into the west wind jet streams.
- Move the **Translate** slider to the very right position on *190* and take the trackball to turn the figure in the viewer window so that you can look at the right side of the bounding box (see figure below).
- In the **Parameters** port increase the *resolution* to *500* in order to smooth them. The default *line width* of *2* is fine.



Properties

Isocontour Annotated Slice

Frame: □ show width: 1

Orientation: xy xz yz

Options: □ adjust view ☑ immediate □ set plane

Translate: 100

Spacing: ○ uniform ● explicit

Values: 0 10 20 30 40 50 60 70 80 90 100

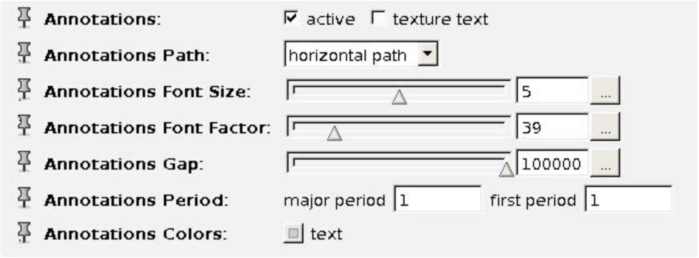
Parameters: resolution 500 line width 2

Options2: □ update min-max

Colormap: -3 130 Edit

Properties: Adjustment of the Isolines' Annotation

- In the *Isocontour Annotated Slice* Properties, make sure that *active* behind the **Annotations** port is switched on. If turned off, no annotation will be drawn at all.
- Test some value combinations for the three Annotations ports **Font Size**, **Font Factor** and **Gap**. The combination of *font size* and *font factor* determines the size of the displayed data figures. *Annotations gap* determines the distance between two annotation entries along an isoline – a larger gap results in fewer annotations. Unfortunately, in most combinations you'll either end up without any visible annotation or with black isolines swamped by annotations. Some practice will help. For the current projection settings, the following parameter combination gives a reasonable annotation:



Annotations: ☑ active □ texture text

Annotations Path: horizontal path

Annotations Font Size: 5

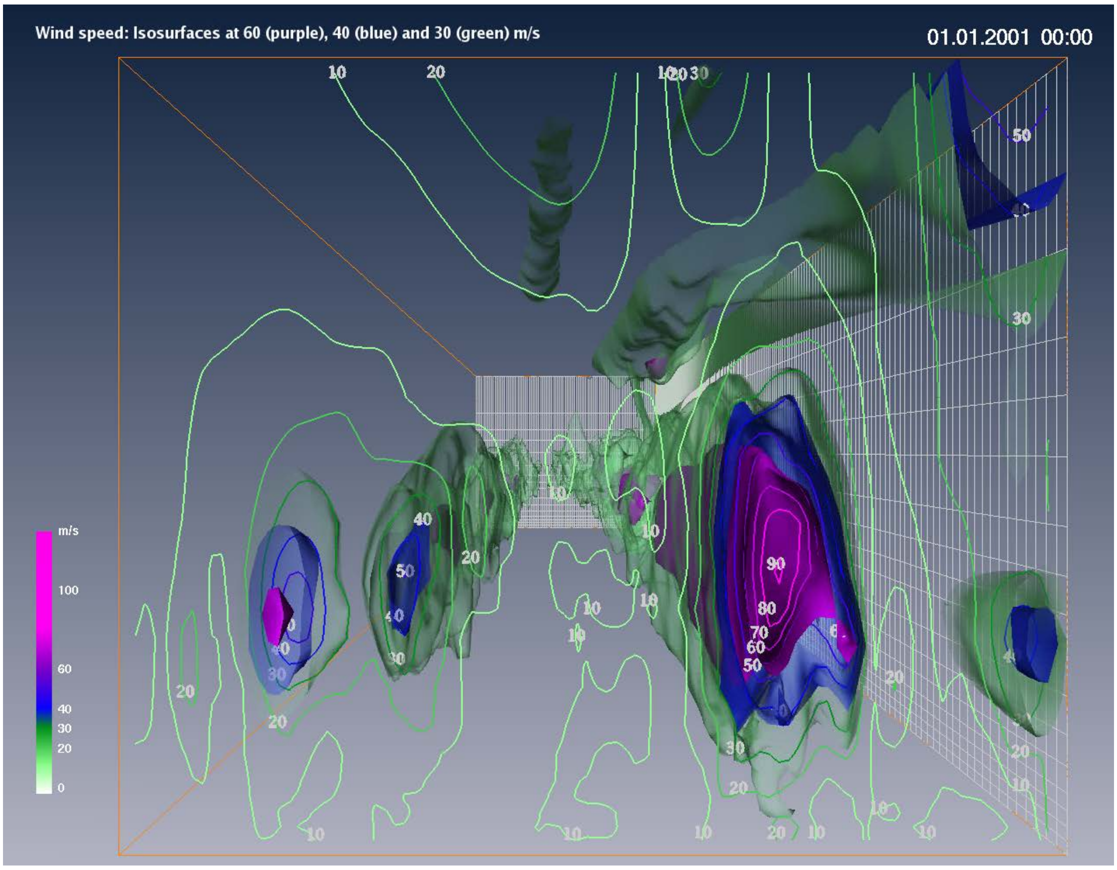
Annotations Font Factor: 39

Annotations Gap: 100000

Annotations Period: major period 1 first period 1

Annotations Colors: text

- In the **Annotations Period** port, set the *major period* to *1* in order to annotate each of the isolines you explicitly entered. By keeping the default value 3, only every third isoline will receive labels.
- Test the **Annotations Path**: With *horizontal path* selected, all data figures are shown parallel to the earth surface, which is parallel to the y-axis, with *vertical path*, respectively, parallel to the z-axis, and in *tangential path*, annotations are tangential to its isolines, resulting in a different orientation for each figure.


Wind speed: Isosurfaces at 60 (purple), 40 (blue) and 30 (green) m/s
01.01.2001 00:00
m/s
100
60
40
30
20
0

- You might want to view this visualization in more detail and enlarge it to full screen mode. To do so, use the fifth button of this series: □ ◫ ⊟ ⊞ ⛶ ⊡.
- Remember, this presentation is unsuitable for video export, but you may want to take a snapshot. You will find the snapshot button 📷 above the viewer window. Pressing this button opens a window, where you can set the filename, format and size of the snapshot.
- Save the current project as *windspeed_AnnoIsocontour_3D.hx*.

In case of any problems, you may want to have a closer look at the project *TUT_windspeed_AnnoIsocontour_3D.hx* in the tutorial folder …

4.21. Volume Rendering

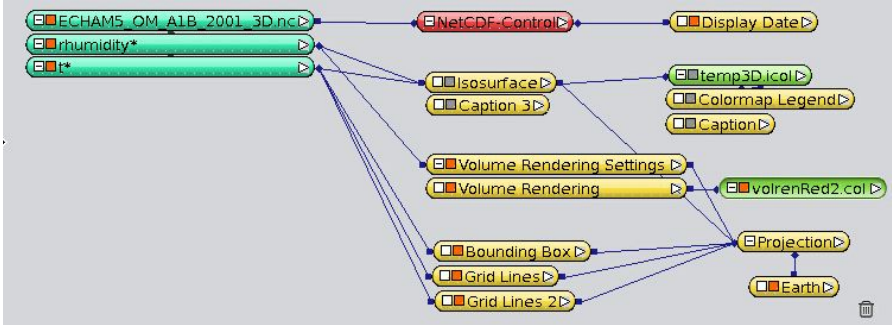
In the Volume Rendering method, the physical values for all grid cells (or “voxels”) are mapped to opacity (alpha) and color values by using a so called **transfer function**. In our case, the transfer function is defined by our *Colormap*, which assigns RGBA (red, green, blue, alpha) values to the physical values. In the volume rendering method, the viewing space is traversed, starting from the front to the rear, and the composed RGBA result is projected on every correspondent pixel of the frame buffer.

Ultimately, a smooth transition between areas with high and low opacity is typical for this method, which is very suitable for the visualization of liquids or gaseous phenomena.

Please notice that Volume Rendering is only available for data with uniform or rectilinear grids, but not for those with curvilinear grids!

Our data have rectilinear coordinates, so we can visualize, for example, the relative humidity as Volume Rendering. This is an alternative to the Isosurface representation shown in chapters 4.17 and 4.19. As mentioned above, this module can offer a more intuitive visualization for liquids or gases which don't show clearly defined interfaces.

- Remove the *Isocontour Annotated Slice*, the three nested *Isosurface*s and all four *Slice*s including the wind speed *Colormap Legend* and *Caption*s from your Main Panel window – we do not need them now. You might also use the *Variables Editor* to completely remove the variable *var3** (wind speed) from the NetCDF file module. Make sure that the *Bounding Box* and Grid Lines are switched off and the *Earth* is switched on.
- Right-click *rhumidity**, then ***Display → Volume Rendering → Create*** and project the new module.



For each data value, its color and transparency is defined by means of a color lookup table (colormap), which includes alpha values (transparency values) in addition to RGB-colors. Avizo provides four predefined colormaps with alpha values. One of them, *volrenWhite.am*, is assigned by default.

The 3D impression is generated by putting 2D slices back-to-back in series. The more 2D slices displayed, the higher the resolution of the visual representation gets. But the volume rendering also gets slower because more graphical objects have to be rendered. For large data sets, this computationally intensive rendering might not be suited for time animation and spatial navigation.

- Rendering Quality: In the *Volume Rendering Settings* Properties, you may want to enable the "Advanced Settings". Then try the slider of the *Sampling Quality* port to increase and decrease the resolution of the volume rendering. For a nice rendering at screen resolution, a value of 1 might be a good start. If more detail is needed, you can increase the maximum value of the Sampling Quality also to higher values.

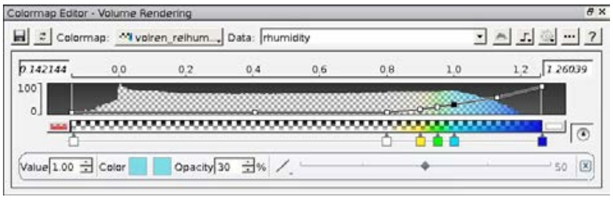
Since the 2D slices do not exactly run through the model grid points, its values must be defined – either by taking the value of the closest grid point (*nearest*) or by linear interpolation between the grid points (*linear or cubic*).

- In the Volume Rendering Properties, keep **Interpolation Advanced** setting *linear*.

Design of the Voltex Colormap

- In the Volume Rendering Properties, click Colormap → ***Edit*** → ***Options*** → ***Edit Colormap***. In the *Colormap Editor* design a colormap which is completely transparent (Opacity = 0%) for a relative humidity less than 80% and with an opacity of 100% for the max value:

Relative Humidity	***Opacity [%]***	***Color***
0.8	0	white
0.9	10	yellow
0.95	20	green
1.05	30	cyan
Max value	100	blue


Colormap Editor - Volume Rendering

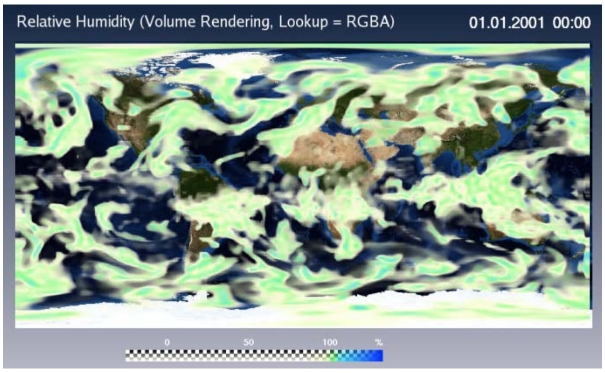
Colormap: volren_relhum... Data: rhumidity

0.142144 1.26039

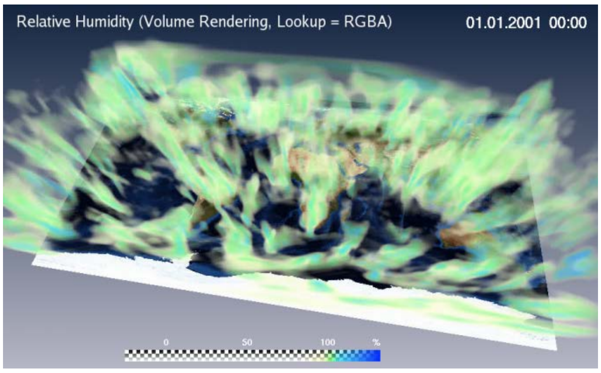
Value 1.00 Color Opacity 30 %

- Save the colormap in your tutorial folder as *volren_relhum.icol* (non-indexed).

XY Ortho View:



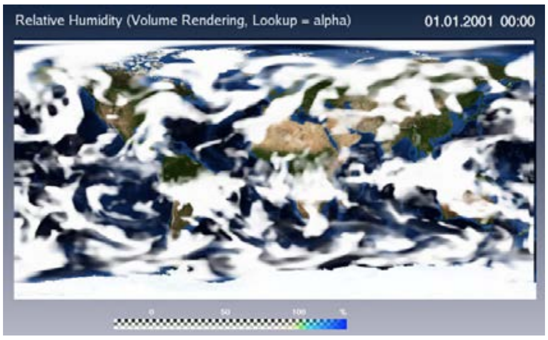
Declined Perspective View:



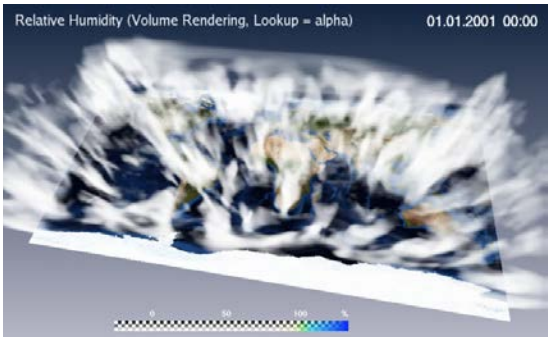
Variation of Lookup Tables

- Our example is even more intuitive if we only use transparency values and dismiss all colors. In the *Volume Rendering* Properties, change the **Colormap Lookup** port from *rgba* (Red-Green-Blue-Alpha) mode to *alpha* mode. The information of the three channels Red, Green and Blue is discarded now (all pixels become white) and only the alpha channel is used for data representation – for both absorption and emission. Regions with a humidity higher than 80% are partly opaque now.

XY Ortho View:



Declined Perspective View:



- In the **Colormap Lookup** port, you may test *luminance alpha*. This configuration does not use any Red-Green-Blue channels information, either. The alpha channel is used for absorption only, whereas for emission, the *luminance* is taken into account (light intensity per area), a photometric term, that human beings perceive as lightness. In our case, moist regions get grey. Change to the *alpha* mode again.
- With the **Alpha Scale** slider you can scale the transparency for the whole visualization. Reduce the opacity from *1* to *0.3*.
- Compare your Volume Rendering results with the *Isosurface* you have created in chapter 4.17 – check different *Isosurface* thresholds of relative humidity.
- Save your project as *relhum_volren_3D.hx*.

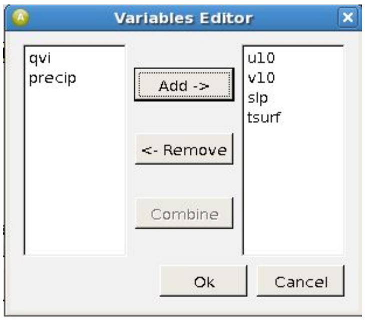
For comparison, you'll find the current project in the tutorial folder: filename *TUT_relhum_volren_3D.hx*.

5. Visualization of 2D Vector Data

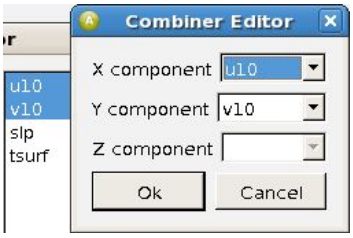
5.1. *Horizontal wind as vector arrows*

Typical vector fields in climate research are the flow fields: the wind and the ocean currents. In the 2D wind case, we may only be interested in the wind components u and v, each in m/s. To visualize the horizontal wind as a vector, we need to combine the two vector components u and v in the NetCDF reader.

- Reopen your *Avizo* project *TUT_temp_slp_Earth_2D.hx* from the tutorial folder. This was the result of chapter 4.9. Save it as *temp_slp_uv_2D.hx* into your private folder for this chapter.
- In the Properties of your NetCDF file *ECHAM5_OM_A1B_2001_2D.nc,* click on the Variables Editor, remove the variable *precip* as we do not need it anymore, and add the two 10 meter wind components *u10* and *v10*.



- On the right side of the Variables Editor, select *u10* and *v10* again and click the *Combine* button. If u10 is assigned to the X component and v10 to the Y component, confirm with Ok. Otherwise swap their assignment.



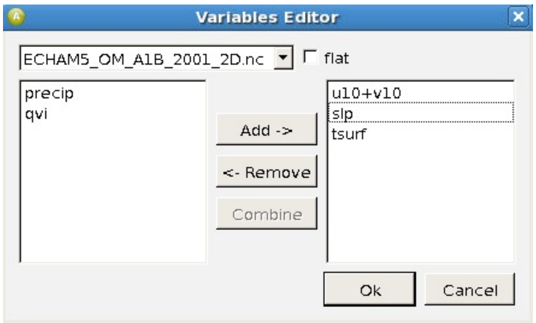
Combiner Editor

X component u10

Y component v10

Z component

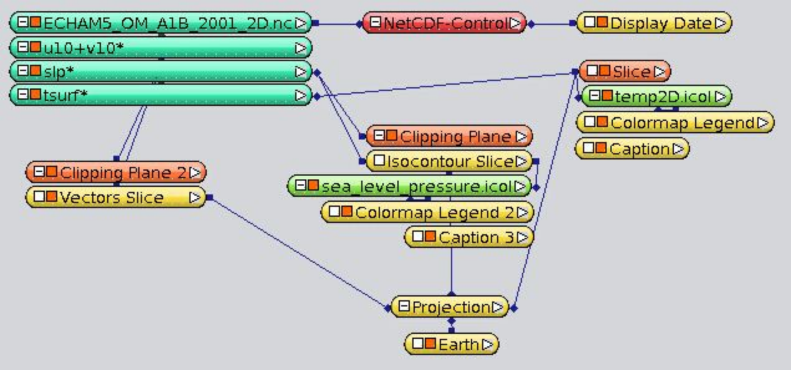
- Load the resulting horizontal wind vector *u10+v10** into the Main Panel window by clicking *Ok* in the *Variables Editor*.



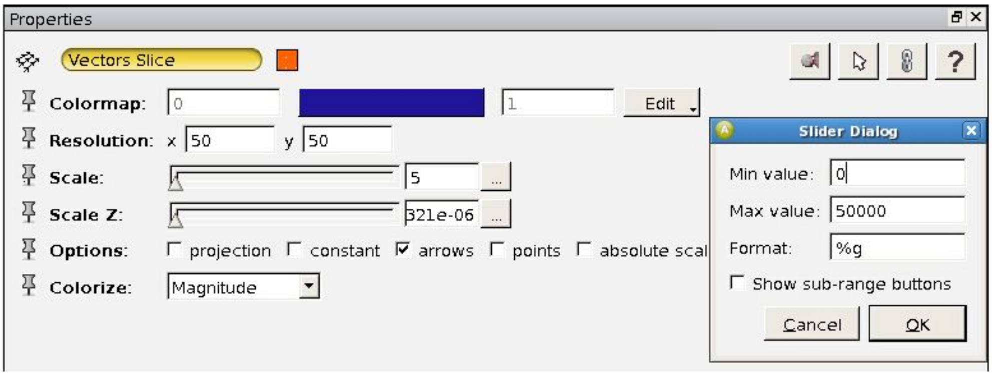
Variables Editor

ECHAM5_OM_A1B_2001_2D.nc ☐ flat

- In your Main Panel window, delete the unused Bar Chart Slice and Caption 2 modules. We do not need them anymore. You can either delete them with the delete button on your keyboard or drag and drop them in the trash in the bottom right corner of your Main Panel window.
- Now we want to visualize the horizontal wind field with vector arrows. To do this, right-click *u10+v10**, then click ***Display → Vectors Slice → Create***.
- The wind vectors also need to be projected onto the globe, so click the small white box of the Vectors Slice module, click Projection and drop the new connection line on the Projection module. [Or simply click as usual.]



- In the *Vectors Slice* Properties, shift the *Scale* slider to the right until it reaches the value 5. This value range seems to be too small again to display any vector arrows, so right-click into the 5, click *Configure* there and set the *Max value* in the *Slider Dialog* to *50000*. Confirm with *OK*.



Properties

Vectors Slice

Colormap: 0 1 Edit

Resolution: x 50 y 50

Scale: 5

Scale Z: 321e-06

Options: □ projection □ constant ☑ arrows □ points □ absolute scal

Colorize: Magnitude

Slider Dialog

Min value: 0

Max value: 50000

Format: %g

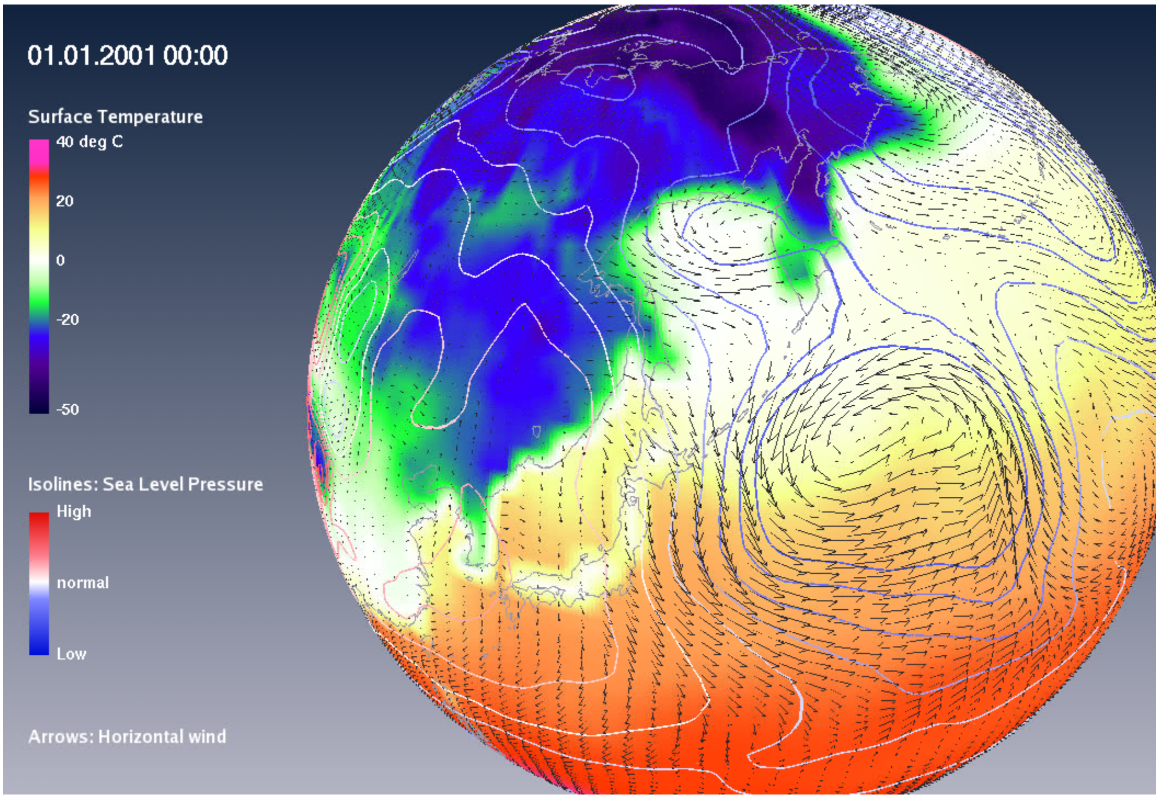
□ Show sub-range buttons

Cancel OK

- Move the slider further right. Now the arrows become visible on the globe.
- Change the color of the arrows into a dark grey (double-click into the colormap of the *Vectors Slice* Properties).
- Deactivate the countries' borders in the *Earth* Properties and choose a lighter grey for the shorelines to avoid confusion with the vectors.
- By default, *Avizo* displays 50 x 50 arrows on the globe (see *Vectors Slice* Properties > Resolution). Set the resolution to *200 x 200* arrows. Note that *Avizo* doesn't display the vector arrows by using the original grid! Internally, an interpolation onto a regular grid is done before visualization.
- If you think the vector arrows are too thin, then you can broaden them by typing in the console:

  ```
  “Vectors Slice” setLineWidth 2
  ```

  The default line width is 1.
- Activate and deactivate the toggle *constant* in the Vectors Slice module. Switched on, all arrows have the same length, meaning they only represent the wind direction, but not the wind magnitude. When "constant" is switched off, the wind magnitude is additionally represented by the length of the arrows.
- Turn the globe with the trackball. Search for strong high-pressure and low-pressure regions in the northern and southern hemispheres. As you might have expected, in the northern hemisphere the wind flows into a low-pressure region counterclockwise (wind backs) and out of a high-pressure region clockwise (wind veers) – and vice versa in the southern hemisphere. Wind magnitudes are higher in low-pressure regions than in high-pressure regions.
- You may also want to add a legend entry for the wind vectors. Click ***Create Object → Annotations → Caption → Create*** and choose the text *"Arrows: Horizontal wind"* and arrange your layout elements similar to the following figure (showing the north-western Pacific in time step 1).



- Save your project (called *temp_slp_uv_2D.hx*).

  This project can also be found in the tutorial folder (filename: *TUT_temp_slp_uv_2D.hx*).

Colorizing of vector arrows

You might want to represent the magnitude of the vector field (the horizontal wind speed here) not only by the length of the arrows but additionally by a color ramp.

- Make sure that the Colorize port in the *Vectors Slice* Properties is set to Magnitude (which is the default).
- Right-click into some empty space of the Main Panel, then click ***Create Object → Other → Colormap → Create***.
- Design your colormap as you like it in the *Colormap Editor*, then save it in your private directory.
- Connect your designed colormap with the Vectors Slice module by clicking on the white rectangle of the *Vectors Slice* module, then clicking *Colormap* and release the new connection line over your new colormap module.
- Finally, adjust the data range of your colormap to the data range of the horizontal wind speed data (e.g. *Vectors Slice* Properties > *Colormap* > *Edit* > *Adjust range*).
- Maybe you want to switch off the temperature Slice and instead switch on the Earth Terrain (*Earth* Properties > Display Mode = Constant, Quality = Low).

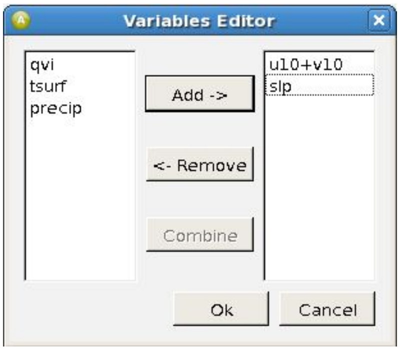
- Save your project again.

In the tutorial folder this is project *TUT_temp_slp_uvCol_2D.hx*.

5.2. Line Integral Convolution (LIC)

We are now going to represent the horizontal wind field by a dense directional representation - similar to the display of many stream lines. We will use the Line Integral Convolution (LIC) method, which yields a nice continuous representation of a static 2D vector field by convolving a random noise texture along streamlines. Experimentally, you could achieve an image similar to a LIC rendering by visualizing a magnetic field with iron filings.

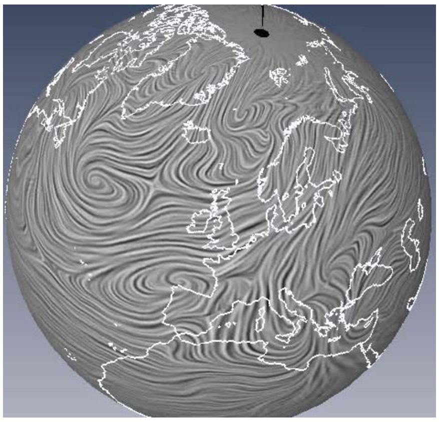
- In the Main Panel window, remove the variable *tsurf** via the Variables Editor.



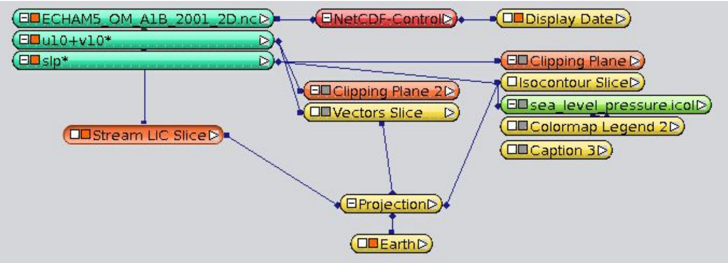
Variables Editor

qvi
tsurf
precip

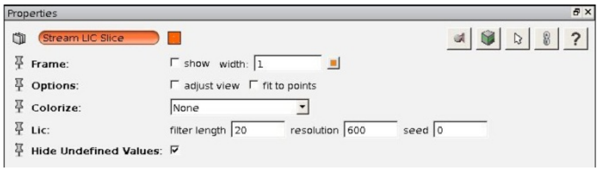
u10+v10
slp



- Switch off all the modules except for Display Date and the Earth to get a better idea of what the LIC is going to look like.
- Right-click the variable *-u10-v10**, and then Display > **Stream LIC Slice**.
- Connect the *Stream LIC Slice* module with the *Projection* module.



- At the bottom of the *Stream LIC Slice* Properties, click ***Apply*** to compute them. You might also enable ***auto-refresh*** to tell *Avizo* to apply each parameter change automatically without pressing Apply each time. But note – the LIC texture is computed mainly on the CPU, which can take some time for any image update.
- Increase the *resolution* to *600*.



Properties

Stream LIC Slice

Frame: □ show width: 1

Options: □ adjust view □ fit to points

Colorize: None

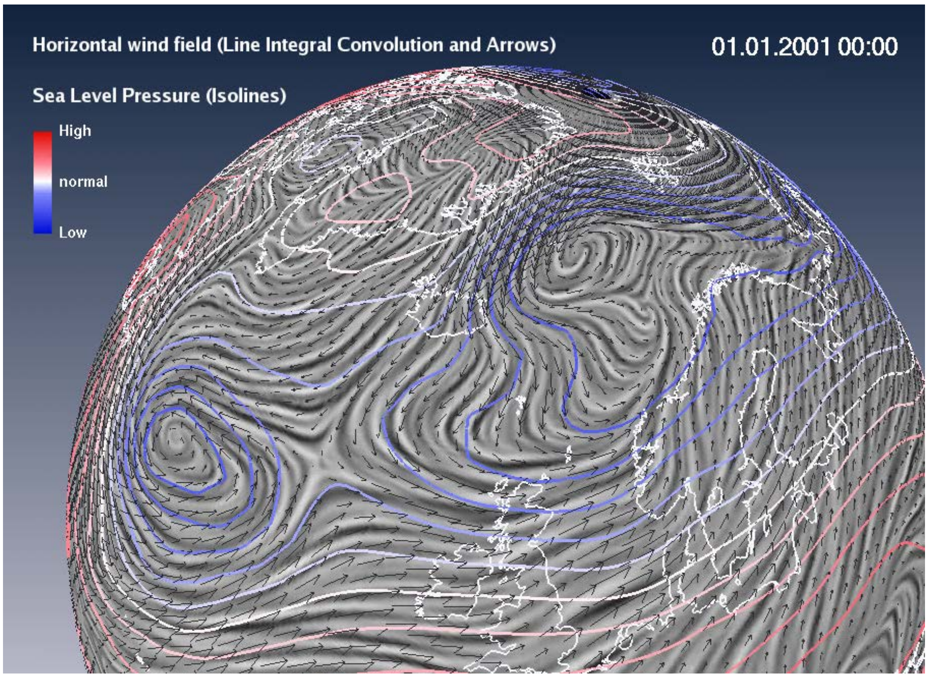
Lic: filter length 20 resolution 600 seed 0

Hide Undefined Values: ☑

- Keep the *filter length* default of *20*. The higher the filter length is, the more coherent the greyscale distribution along the field lines. Therefore, larger values are more attractive. A filter length of 0 results in a noise pattern without any directional information.

Movies including the Stream LIC Slice
In the Stream LIC Slice Properties you'll find a port called *seed* that allows control of the generation of the noise pattern. This is useful in case you want to produce videos with a time dependent LIC. With a seed value of 0 (default), a different random noise pattern is used for each rendered image. With a seed value different from 0, the same noise pattern is used for the computation of each time step, yielding a continuous and temporally consistent vector field visualization.

- Change the color of the shorelines to white (*Earth* Properties).
- Add a legend entry (***Create Object* → *Annotations* → *Caption* → *Create***) with the text *"Horizontal wind (Line Integral Convolution and Arrows)"* and again switch on the *Isocontour Slice* and *Vectors Slice* modules as well as the *Colormap Legend* and *Caption* of the sea level. Now you can easily explore the interrelation between pressure and wind field. You may want to re-arrange the legend entries as follows:


Horizontal wind field (Line Integral Convolution and Arrows)
01.01.2001 00:00
Sea Level Pressure (Isolines)
High
normal
Low

Remark: In the spherical projection, you will notice that the Stream LIC Slice module cannot smoothly connect the most western boundary of the data field with the eastern boundary – even when you have activated *Copy first longitude* in the NetCDF-Control. This is caused by the method itself.

- Save your project as slp_Isocontour_Slice_LIC.hx.

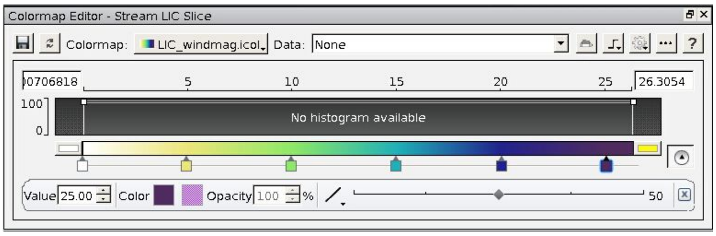
In the tutorial folder you can find a corresponding project called TUT_slp_Isocontour_Slice_LIC.hx.

5.3. Bivariate Visualization: LIC & Wind Magnitude

The LIC texture can also be colorized – as we did for the Bar Chart Slice in chapter 4.7. We are going to colorize the LIC with the magnitude of the wind velocity first.

- In the **Colorize** port of the *Stream LIC Slice* Properties select *Magnitude.*
- Right-click into the free space of the Main Panel window, then click ***Create Object* → *Other* → *Colormap* → *Create***. Connect the Stream LIC Slice module with this new colormap.
- In the *Stream LIC Slice* Properties, click ColorMap 1 > *Edit* > *Adjust range* first. Then you may click *Edit* > *Options* > *Edit Colormap* to open the *Colormap Editor* and create a colormap like this one:

0 m/s	White
5 m/s	Light yellow
10 m/s	Light green
15 m/s	Cyan
20 m/s	Dark blue
25 m/s	Purple

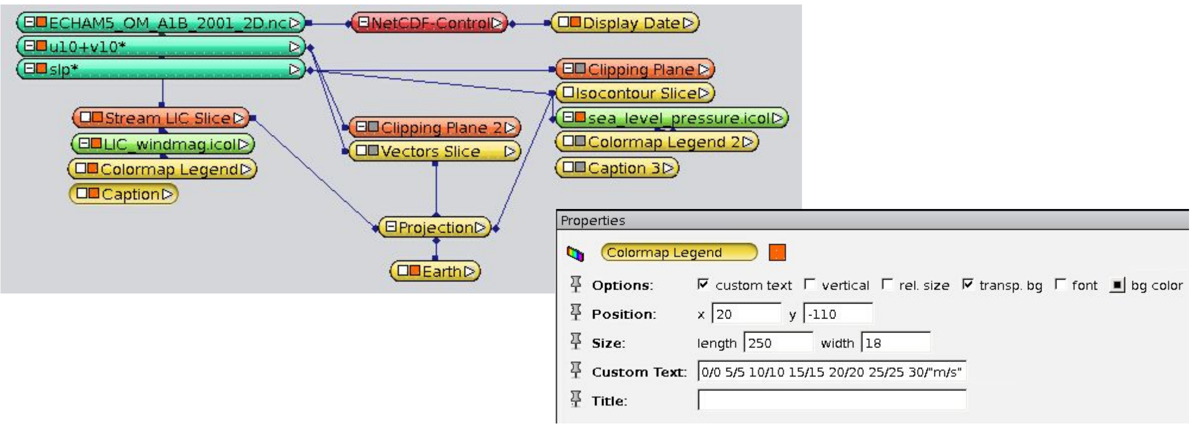

Colormap Editor - Stream LIC Slice

Colormap: LIC_windmag.icol Data: None

0706818 26.3054

Value 25.00 Color Opacity 100 %

- Save this colormap as non-indexed with the filename *LIC_windmag.icol*.
- Close the *Colormap Editor* by pressing *OK*.
- Add a legend entry for this colormap (*Colormap Legend*) in horizontal orientation and include the units "m/s" in quotation marks at position 30.



Properties

Colormap Legend

Options: ☑ custom text ☐ vertical ☐ rel. size ☑ transp. bg ☐ font ■ bg color

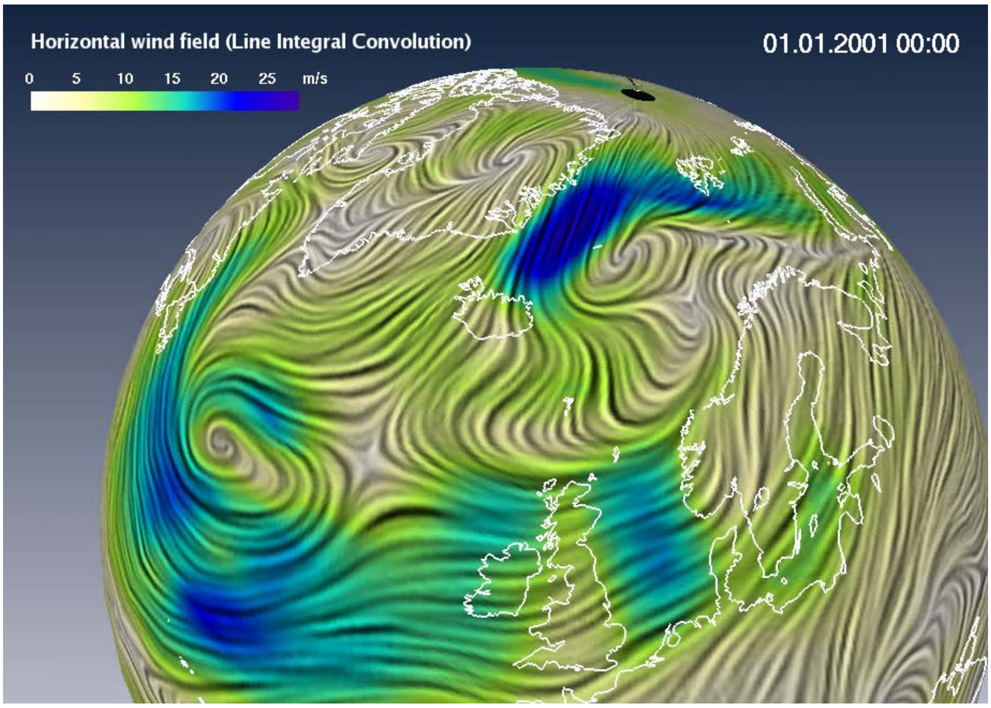
Position: x 20 y -110

Size: length 250 width 18

Custom Text: 0/0 5/5 10/10 15/15 20/20 25/25 30/"m/s"

Title:

- Adjust the date, text and colormap positions according to the following figure:



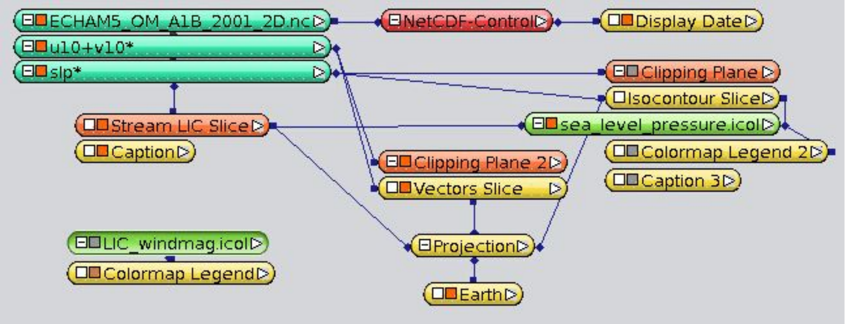
- Save this project as *LIC_windmag_2D.hx*.

In case of uncertainties you will also find a similar project in the tutorial folder (filename *TUT_LIC_windmag_2D.hx*).

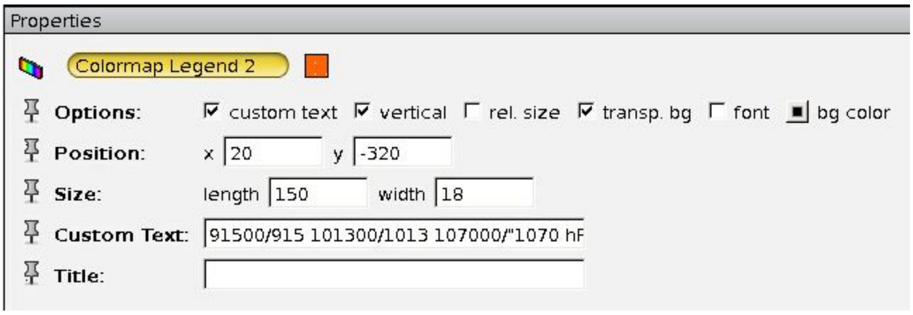
5.4. Bivariate Visualization: LIC & Sea Level Pressure

Alternatively, you may want to colorize the LIC with a different variable, e.g. with the sea level pressure:

- In the **Colorize** port of the *Stream LIC Slice* Properties select *ColorField*.
- In the Main Panel window, click on the small white box of the Stream LIC Slice, select *ColorField* and connect the arising line with the *slp** variable of the NetCDF module.
- Detach the connection between the *Stream LIC Slice* module and the colormap *LIC_windmag.icol* and connect it to colormap *sea_level_pressure.icol* instead.
- In the *Stream LIC Slice* module *ColorMap 1 > Edit > Adjust range to > slp*
- Switch off *LIC_windmag.icol* and switch on *sea_level_pressure.icol*.



- Instead of a relative colormap annotation like “high – normal – low”, you can also define min/max values and by hand convert the unit: 103000 Pa = 1013 hPa. In the *Colormap Legend 2* Properties, you can include the units with the help of quotation marks:



Properties

Colormap Legend 2

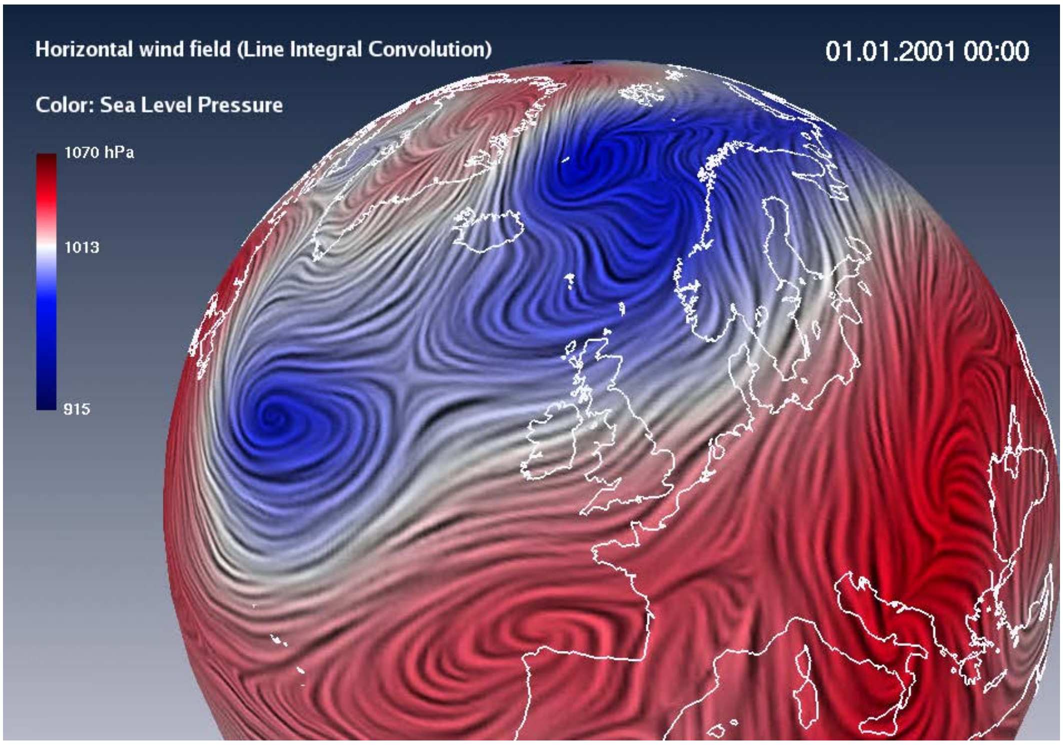
Options: ☑ custom text ☑ vertical □ rel. size ☑ transp. bg □ font ■ bg color

Position: x 20 y -320

Size: length 150 width 18

Custom Text: 91500/915 101300/1013 107000/"1070 hF

Title:


Horizontal wind field (Line Integral Convolution)
01.01.2001 00:00
Color: Sea Level Pressure
1070 hPa
1013
915

- Save your project as *LIC_slp_2D.hx*.

As usual, you can find the according project, *TUT_LIC_slp_2D.hx*, in the tutorial folder.

6. Visualization of 3D Vector Fields

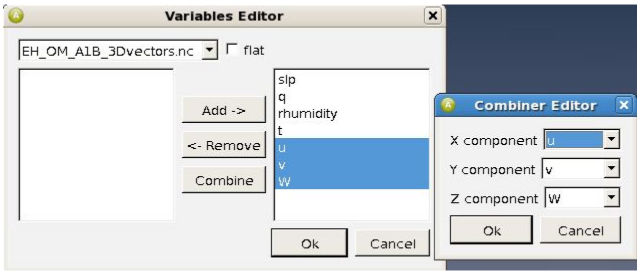
In this chapter, we are finally going to visualize a 3D wind field of the North Atlantic and European area up to a height of 300 hPa (9 levels) – using vector arrows in space (see chapter 6.1), streamlines (see chapter 6.2) and trajectories (see chapter 6.4). To enhance our understanding of the complex meteorological situation in the troposphere, we additionally visualize some scalar variables like pressure, temperature, and humidity.

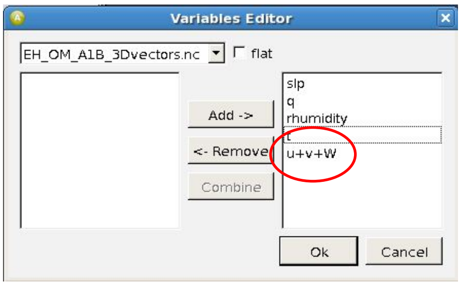
As a reminder: ***Streamlines*** are lines in a velocity field of a steady stream, whose tangents locally always have the same direction as the velocity vectors. ***Trajectories***, however, describe the paths of a particle for the time dependent case. For steady vector fields, streamlines and trajectories are identical. For time dependent vector fields, e.g. 3D-wind fields, streamlines visualize the field (as if it was stationary) at a certain point of time, whereas the trajectories show the dynamics within a certain time period.

6.1. Vector Arrows in Space

- Load the NetCDF file *EH_OM_A1B_3Dvectors.nc* and add all of the variables in the *Variables Editor*:
  - slp (sea level pressure) is a 2D scalar dataset,
  - t (temperature), q (specific humidity), and rhumidity (relative humidity) are 3D scalar datasets and
  - u, v, W are the three vector components of the wind, which we will focus on in this exercise.

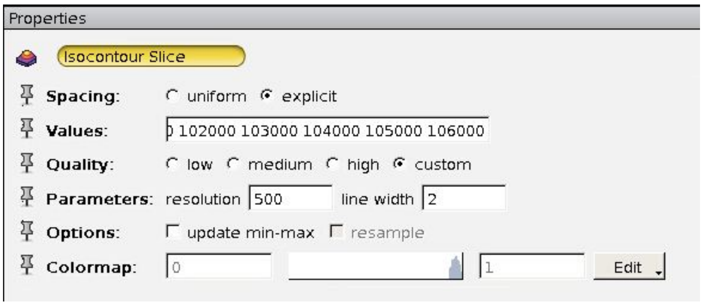
  On the right hand side of the *Variables Editor*, select u, v, W and click *Combine* to load the *Combiner Editor* and construct the resulting wind vector *u+v+W** there (also see chapter 5.1).





- Add a date and time entry in your viewer window (Right-click NetCDF-Control > ***Annotate → Display Date → Create***).

- Right-click the vector *u+v+W**, then click ***Bounding Box*** and project it via .
- Load a standard earth (***Create Object → Other → Earth → Create***) and project it. Select the *EQUIDISTANT_CYLINDRICAL* projection.
- Display the sea level pressure as Isocontour Slice (slp* → ***Display → Isocontour Slice → Create***) and enable projection for this module :



Properties

Isocontour Slice

Spacing: ○ uniform ● explicit

Values: 0 102000 103000 104000 105000 106000

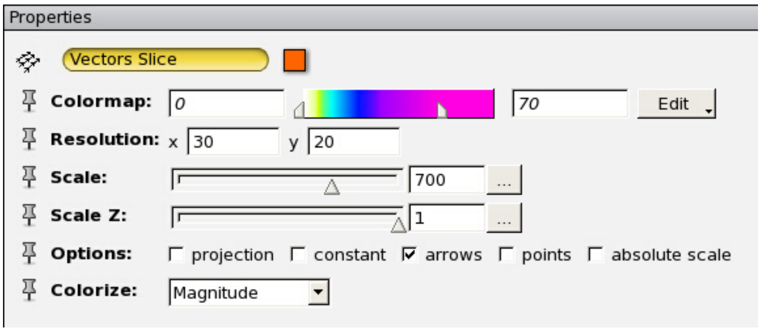
Quality: ○ low ○ medium ○ high ● custom

Parameters: resolution 500 line width 2

Options: □ update min-max □ resample

Colormap: 0 1 Edit

- To add a plane with vector arrows, right-click the *u+v+W** variable, click ***Display* → *Vectors Slice* → *Create*** and enable projection also for this module. Load the colormap *windspeed_300hPa.icol* from the tutorial folder to colorize the arrows, and use the following additional parameters:



Properties

Vectors Slice

Colormap: 0 70 Edit

Resolution: x 30 y 20

Scale: 700

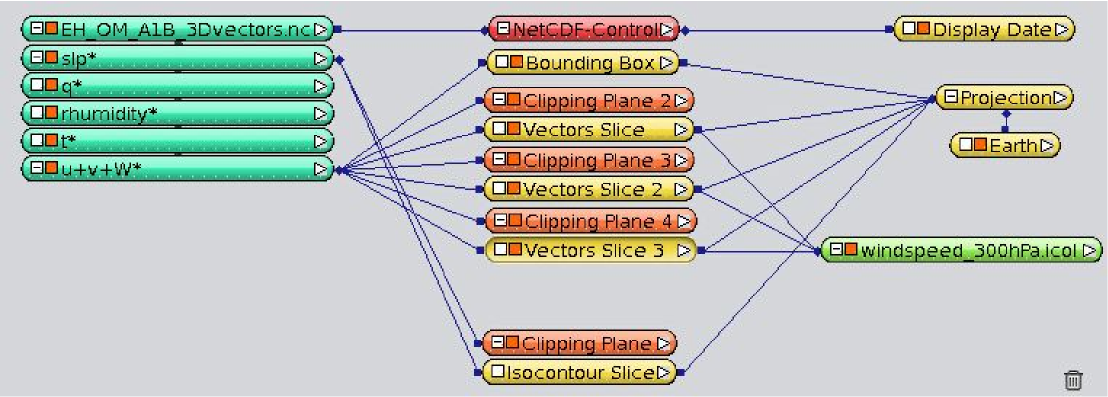
Scale Z: 1

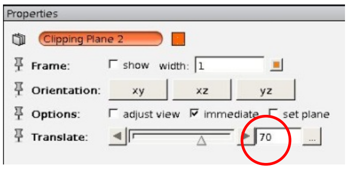
Options: □ projection □ constant ☑ arrows □ points □ absolute scale

Colorize: Magnitude

In contrast to chapter 5.1, where we have introduced the vector arrows for the 2D case, the 3D arrows are now not necessarily parallel to the x-y-plane since the vertical wind component is also used.
With the ***Scale Z*** parameter of the module, the vertical vector component W can be scaled with factors different from “1”, such as to the “700” chosen above. Be careful, this leads to physically incorrect representations! A scaling of 0 would be useful for cases where only the horizontal velocity is analyzed. Caution: In the current version of *Avizo* , the default for the *Scale Z* parameter is 8.99321 E-06.

- To get a better picture of vertical wind distribution, add two more vector planes, each in xy orientation. Project each of them, move them with the ***Translate*** slider to 70, 45 and 20 and make sure that they are all connected to the same colormap (*windspeed_300hPa.icol*):





Properties

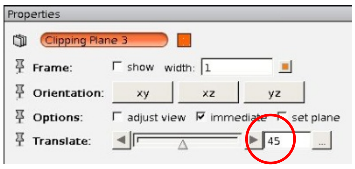
Clipping Plane 2

Frame: □ show width: 1

Orientation: xy xz yz

Options: □ adjust view ☑ immediate □ set plane

Translate: 70



Properties

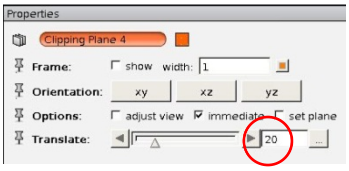
Clipping Plane 3

Frame: □ show width: 1

Orientation: xy xz yz

Options: □ adjust view ☑ immediate □ set plane

Translate: 45



Properties

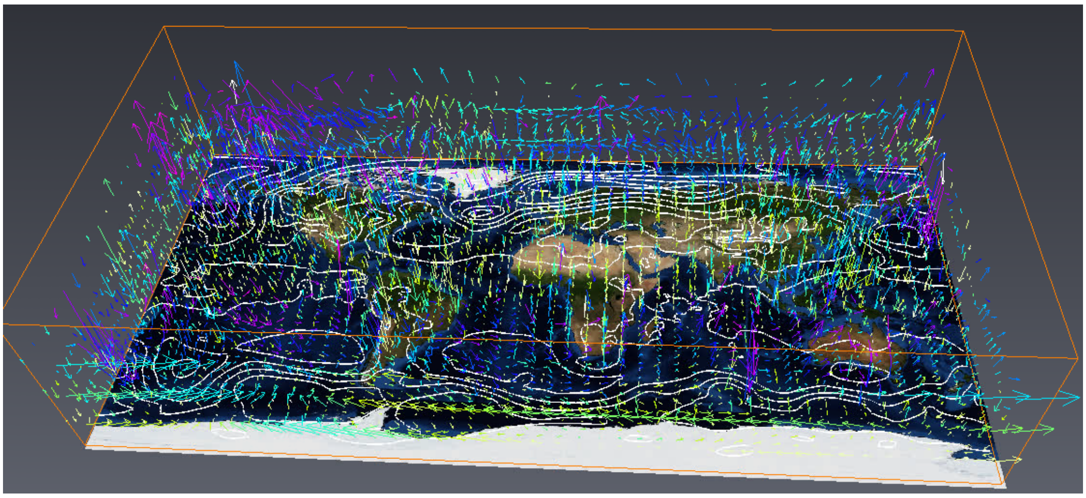
Clipping Plane 4

Frame: □ show width: 1

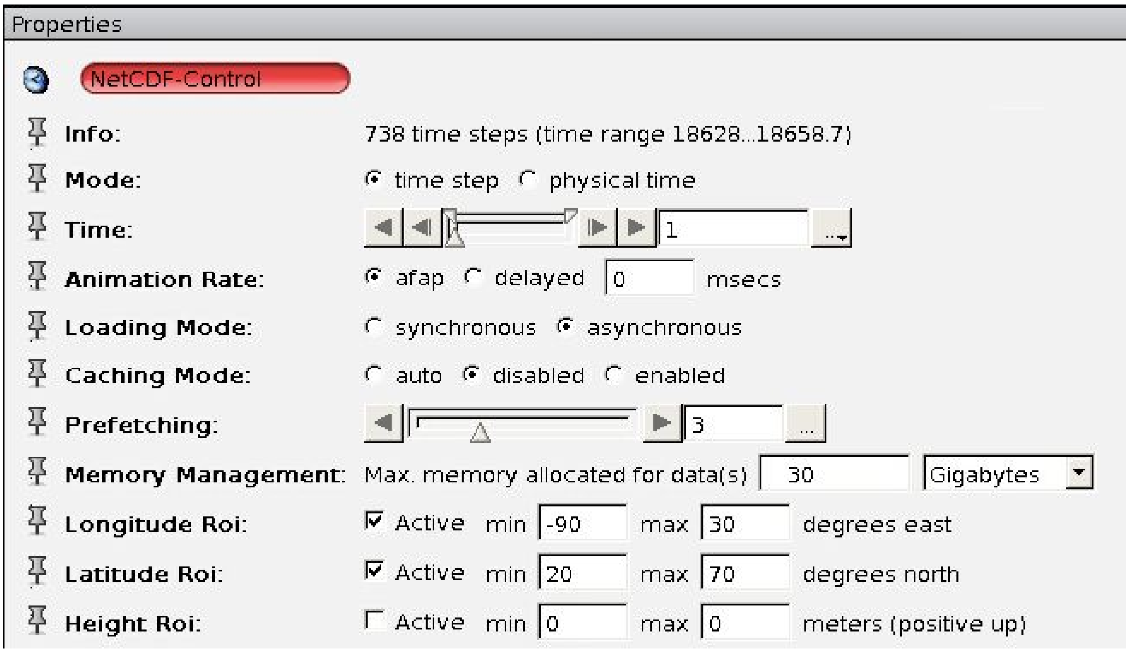
Orientation: xy xz yz

Options: □ adjust view ☑ immediate □ set plane

Translate: 20



- Since this representation gets quite confusing, let’s focus on a region of interest (***Roi***), which shall be the North Atlantic and Europe. You don’t have to crop your dataset outside of Avizo to do so, instead you can define your *Roi* in the *NetCDF-Control* Properties:



Properties

NetCDF-Control

Info: 738 time steps (time range 18628...18658.7)

Mode: ● time step ○ physical time

Time: 1

Animation Rate: ● afap ○ delayed 0 msecs

Loading Mode: ○ synchronous ● asynchronous

Caching Mode: ○ auto ● disabled ○ enabled

Prefetching: 3

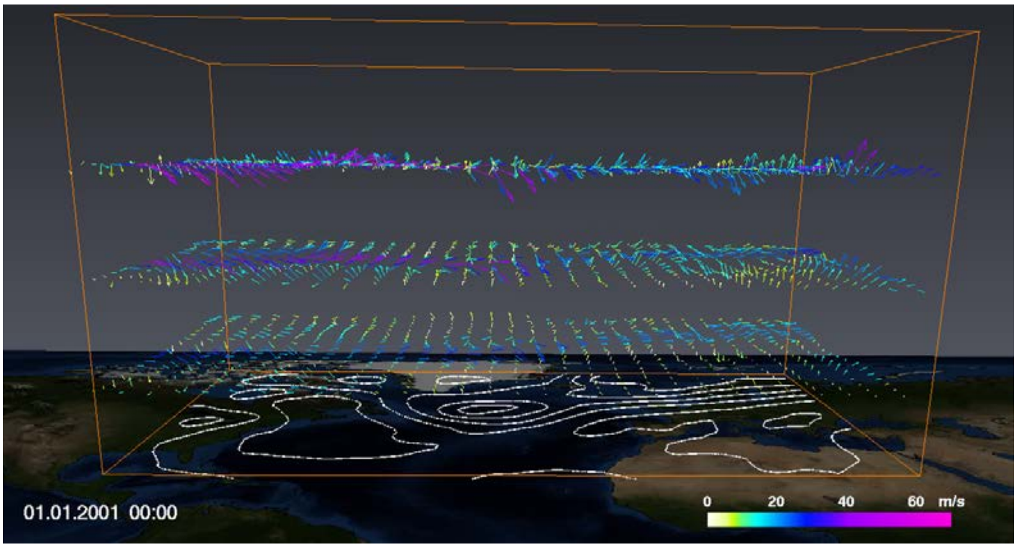
Memory Management: Max. memory allocated for data(s) 30 Gigabytes

Longitude Roi: ☑ Active min -90 max 30 degrees east

Latitude Roi: ☑ Active min 20 max 70 degrees north

Height Roi: □ Active min 0 max 0 meters (positive up)

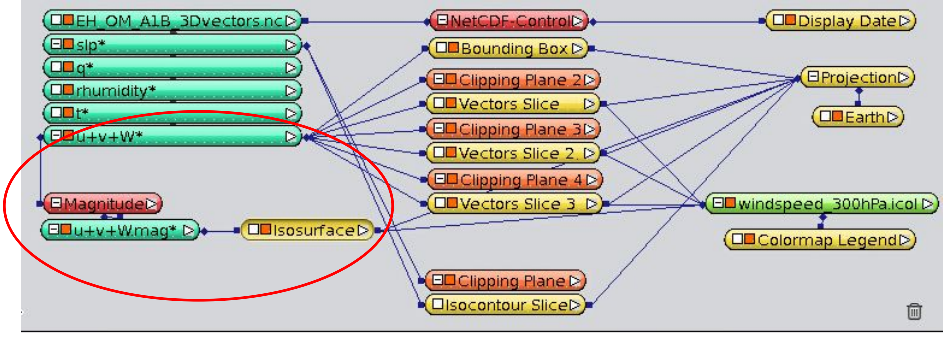
The bounding box, as well as all other visualizations of that data file, are limited to the longitudes and latitudes you have defined. Unfortunately, Avizo currently resets the **Scale Z** parameters while selecting the region of interest, so you will have to set these parameters again (and again).

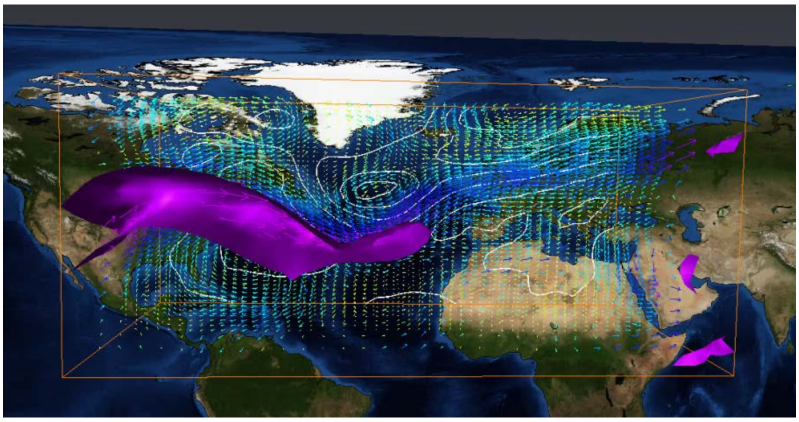


Remark: Due to a bug in the current version of *Avizo* (8.0), you cannot animate the wind vectors with a z-scaling other than the default value. After progressing one time step (and after reloading your whole project), you will have to reset your z scales manually again, because Avizo automatically sets them back to the default.

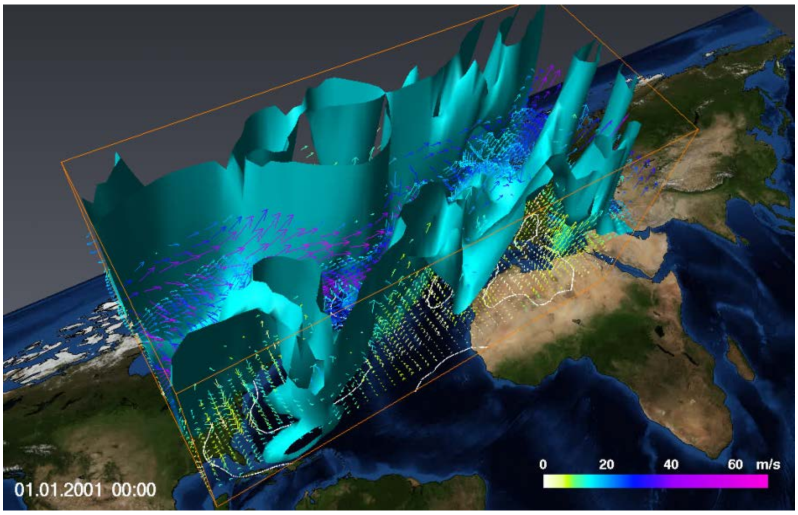
Adding a wind speed scalar

- To underline the 3D impression of the wind velocity, we can add an *Isosurface* of the wind speed. Since isosurfaces can only be computed from *scalar* data, we have to compute the *magnitude* of our vector field first. This can easily be done by right-clicking the *u+v+W** variable, then clicking ***Compute* → *Magnitude* → *Create***. Select the new module u+v+W.mag* in your Main Panel window then and create an *Isosurface*. Project it , select a threshold of 30 m/s, press auto-refresh and connect its Colormap port to the *windspeed_300hPa.icol* colormap. Change the threshold between values of 30 and 60 m/s and watch the isosurface growing and shrinking.





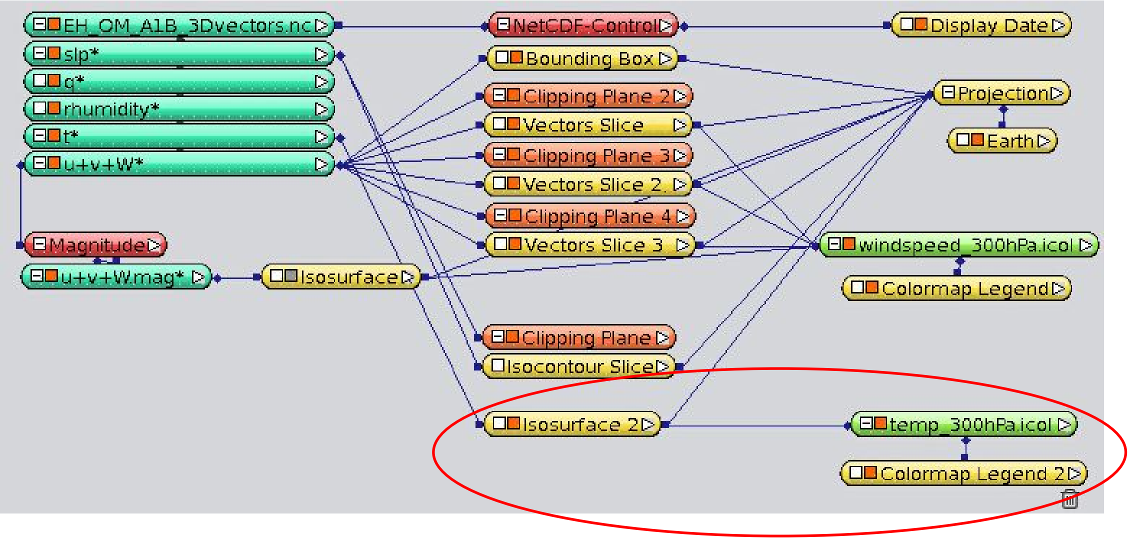
Threshold = 50 m/s
XY View
Time step 1



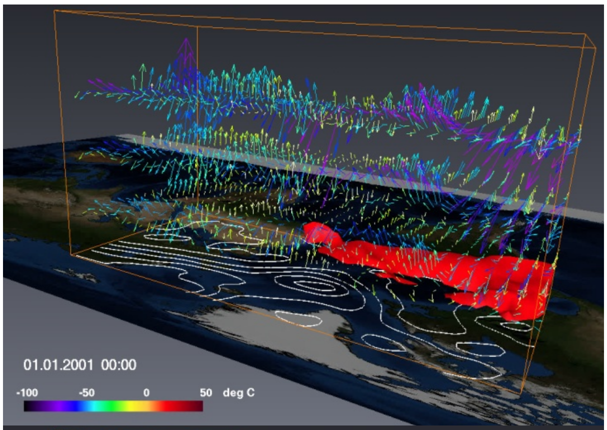
Threshold = 15 m/s
Lateral view
Time step 1

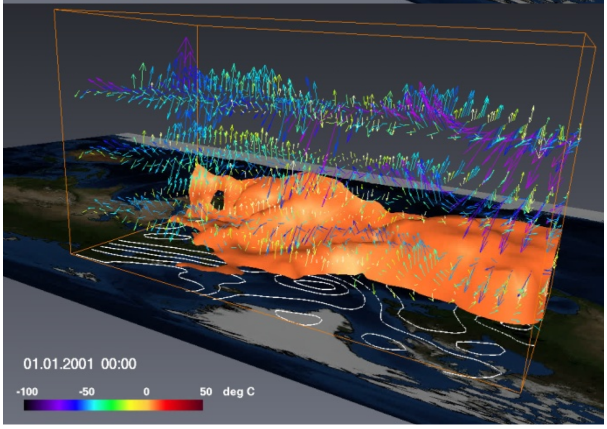
Adding temperature – the frontal system

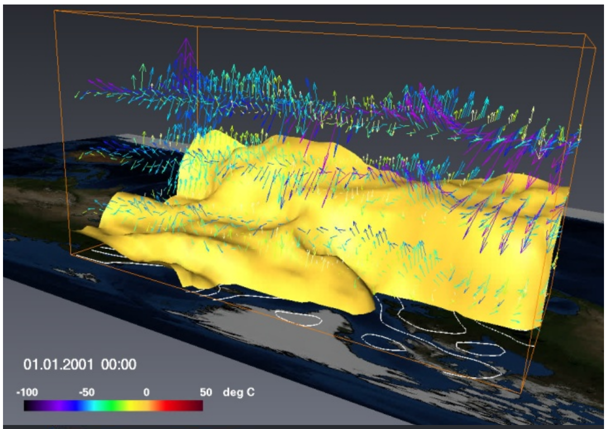
- Once you have a good overview over the spatial features of the wind field, you may want to add other variables to your visualization. Switch off the wind isosurface and construct a new isosurface for the *t** variable (temperature), which can give you an idea about the frontal system when you change the threshold from 290 to 220 Kelvin. Import the colormap *temp_300hPa.icol* from your tutorial folder and use it for the isosurface.

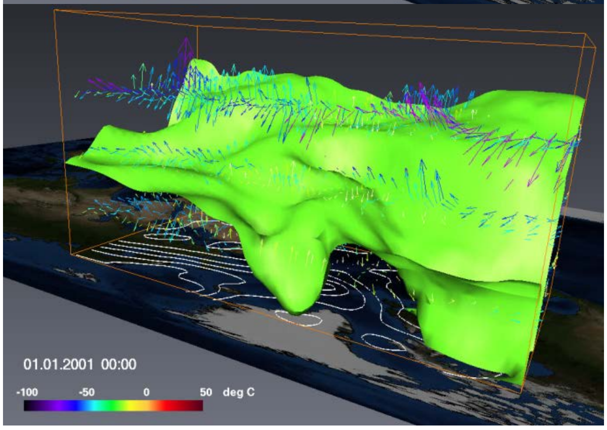


The following four screenshots show the frontal system from a northwestern perspective (from Greenland to Africa): Temperature Isosurfaces at 290 K, 280 K, 270 K and 250 K.



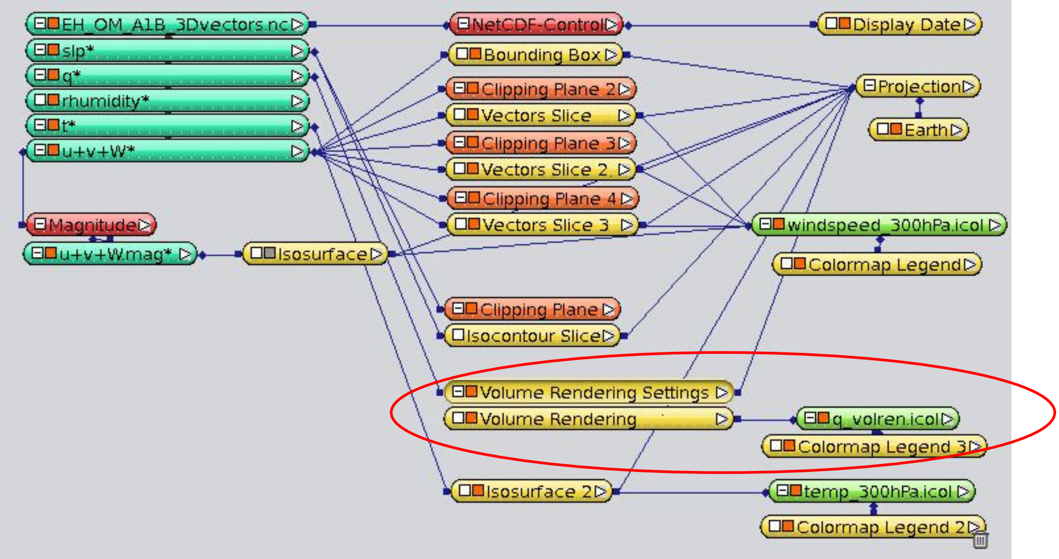




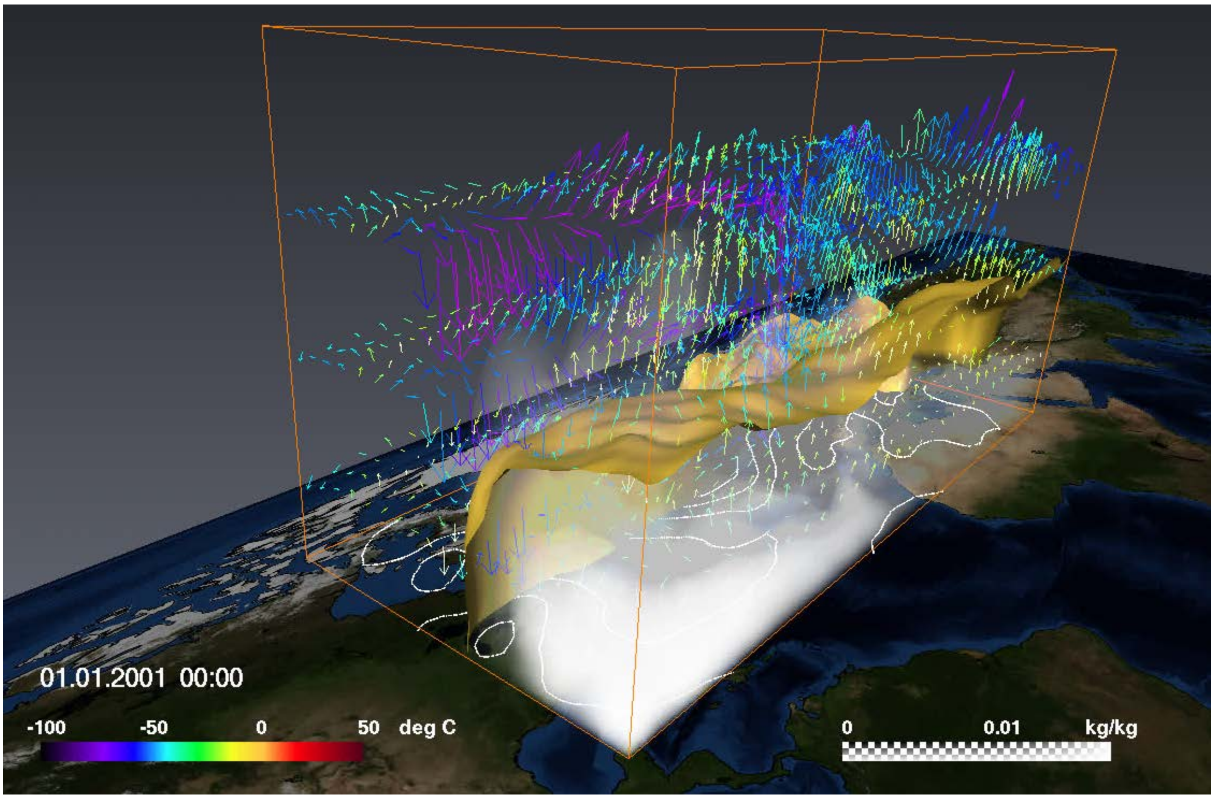


Adding Humidity

- Up to now we have visualized three scalar variables – sea surface pressure, wind field and temperature. Let’s add a volume rendering of the specific humidity q (in kg/kg): right-click *q** → ***Display*** → ***Volume Rendering*** → ***Create***, project it and choose *Sampling Quality* = 500 in the *Volume Rendering Setting* module (Note, you will again have to enable “Advanced Settings” in order to access this parameter!). Load the colormap *q_volren.icol* from your tutorial folder and connect it to your *Volume Rendering* module.



In the following screenshot you can see that warm and humid air masses are transported from the south west to the cyclone over the Atlantic. The north and north eastern regions of our Roi are quite dry.



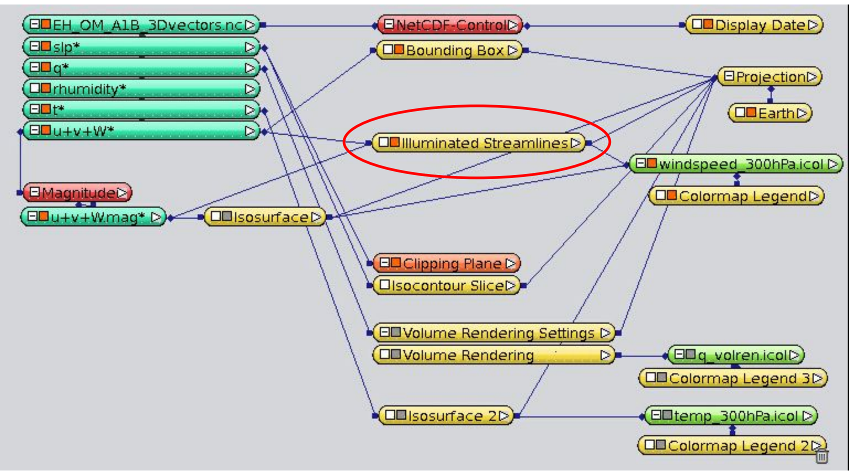
Save your project as *vectorArrows_3D.hx*.

A similar script called *TUT_ vectorArrows_3D.hx* is located in the tutorial folder.

6.2. Illuminated Streamlines

In this exercise, we want to replace the three vector arrow planes by streamlines.

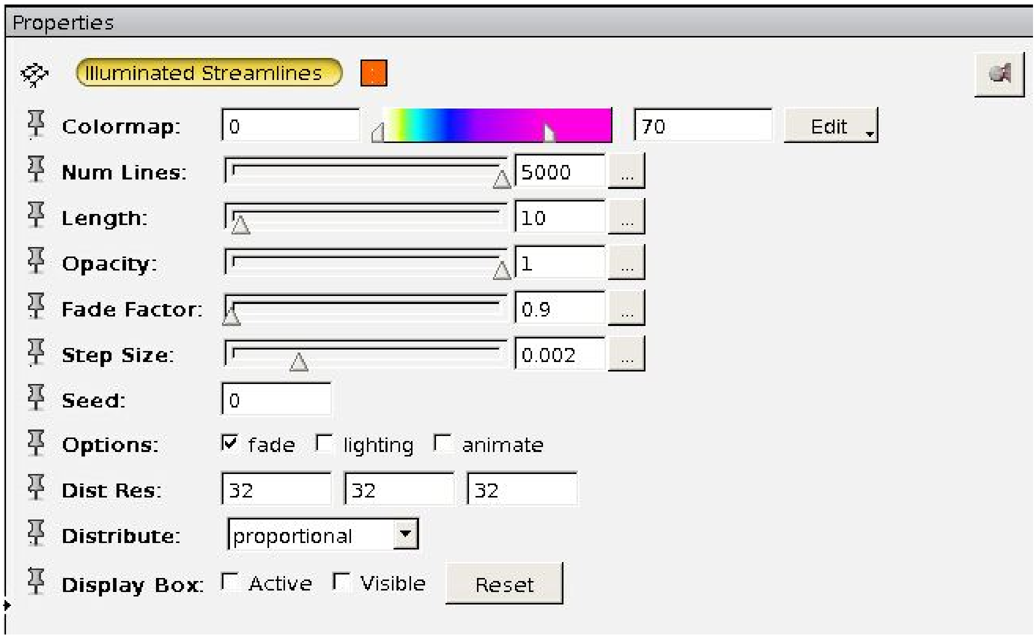
- You can either go on working with your project from chapter 6.1 or load the pre-defined project *TUT_vectorArrows_3D.hx* and save it in your folder (you do not have writing access in the tutorial folder). Remove the three Vectors Slice modules and their empty planes. Switch off the temperature isosurface and the volume rendering of the specific humidity – we won't need them until the end of this exercise.
- Right-click the *u+v+W** vector, then click ***Display → Illuminated Streamlines → Create*** and project the resulting module by clicking as well. To colourize the streamlines by the wind speed magnitude (scalar), connect the *ColorField* port of the Illuminated Streamlines module with the *u+v+W.mag** variable and the *Colormap* port with *windspeed_300hPa.icol*.



- In the Illuminated Streamlines Properties you can define the number of streamlines that shall be visualized. Set **Num Lines** to *5000*.
- Each streamline shall have a **Length** of *10*.
- The streamline shall be completely visible (**Opacity** = *1*).
- In the **Options** port, activate the *fade* and deactivate the *lighting* options. By *fading out* the streamlines, you get an intuitive impression of the stream direction since they get greyer and more transparent at their ends.
- With the fade mode active, an additional port called **Fade Factor** appears, which you can use to define how quickly the opacity shall be reduced along a streamline, or in other words, how quickly the transparency shall increase. The first streamline segment always has the opacity value you have defined in the Opacity port. The opacity of the second segment is the same value multiplied by the fade factor; that of the third segment is the opacity value of the second element multiplied by the fade factor and so on. You may set the fade factor to *0.9* in our example, so that the transparency of each segment increases by 10% relative to the previous one.

Remark to fading: You won't be able to see any fading effect when using short streamlines (of only a few line segments) combined with a small reduction of opacity, because the last line segment will still have a very high opacity then. So, generally speaking, the smaller the length of your streamlines, the smaller the fading factor should also be.

- Choose a very small **Step Size** of *0.002*. This is important for the animation of the streamlines later.
- In the **Distribute** port change the distribution of the seed points from *homogeneous* to *proportional*. Thereby, seed points are distributed proportional to the magnitude of the vectors, so that more streamlines are passing through regions with high wind velocity than regions with lower wind velocity.



Properties

Illuminated Streamlines

Colormap: 0 70 Edit

Num Lines: 5000

Length: 10

Opacity: 1

Fade Factor: 0.9

Step Size: 0.002

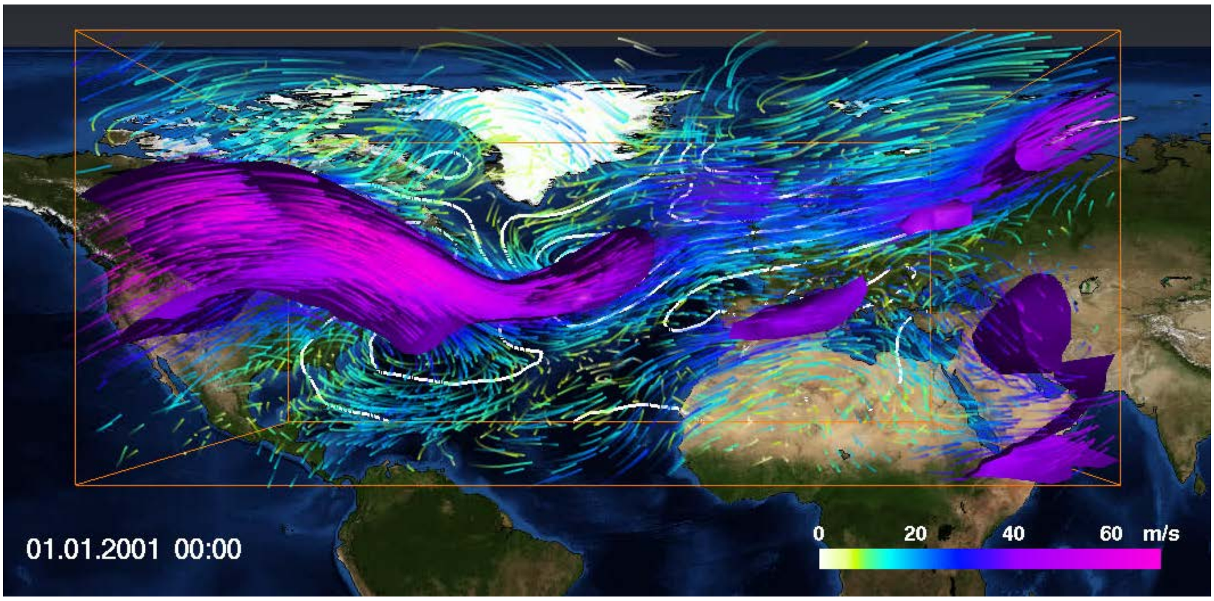
Seed: 0

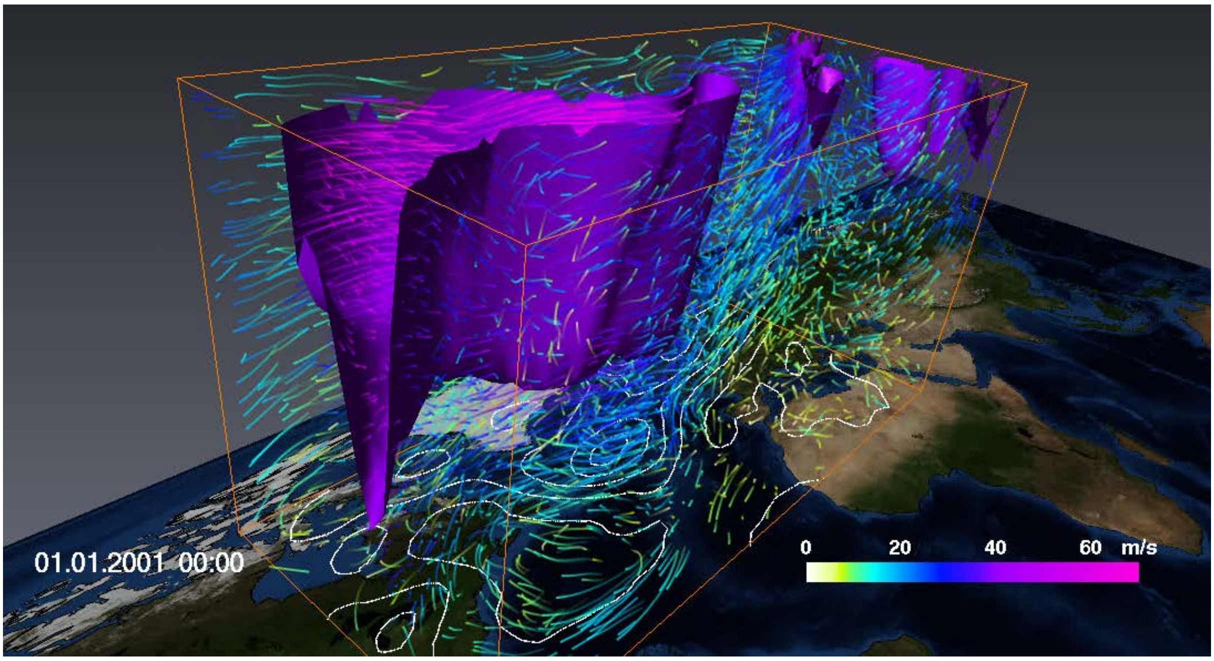
Options: ☑ fade ☐ lighting ☐ animate

Dist Res: 32 32 32

Distribute: proportional

Display Box: ☐ Active ☐ Visible Reset





- Unfortunately, the **Line Width** of the streamlines cannot be changed in the *Properties* window, but you can use a *Console* command to do so. The default value for the LineWidth is 1. With the following command you can increase it:
  ```
  “Illuminated Streamlines” setLineWidth 2
  ```
- In the **Options** port, test the *lighting* mode. Each streamline gets illuminated now, increasing the 3D impression (but from some viewing angles it gets too grey and dark this way).
- Switch on your temperature isosurface and the volume rendering of the specific humidity again including their colormaps to observe the interrelation of the four displayed variables.
- Alternatively, the streamlines could be coloured by the temperature variable: In the Main Panel window, right-click the *Illuminated Streamlines* module, connect its *ColorField* port to *t** and its *Colormap* port to *temp_300hPa.icol*.

Animation of streamlines

- The streamlines can be animated for a fixed time by clicking *animate* in the *Options* port. The result looks like the advection of particles in a steady wind field.
- If the streamlines move too fast, you can decrease their velocity to the minimal value of 1 by entering a Console command (speed depends on the Step size):
  ```
  “Illuminated Streamlines“ setAnimationSpeed 1
  ```
- To slow the animation down further, you may reduce the **Step Size** e.g. to *0.001*.
- Save your project as *streamlines_3D.hx*.

The corresponding project in the tutorial folder is called *TUT_streamlines_3D.hx* again.

6.3. Using a Surface to seed Streamlines

You might have thought that streamlines are quite interesting, but that you cannot easily locate their position in space: which streamlines are in the front and which are further in the background of your visualization? When too many streamlines are concurrently visible in the scene, they will occlude each other - which makes it even more difficult to understand the structures. Additionally, the interactive graphics performance decreases because a large amount of geometric information has to be processed and rendered.

In this chapter we are going to use a so-called *Seed Surface* in order to “filter the data” before rendering by means of a data driven streamline placement method. A *Seed Surface* is a surface which is used to define the area where the streamlines are started (“seeded”). With a cleverly selected *Seed Surface*, the quality of streamline visualizations can be drastically improved, because the amount of visual clutter in the scene is minimized while the important structures are highlighted.

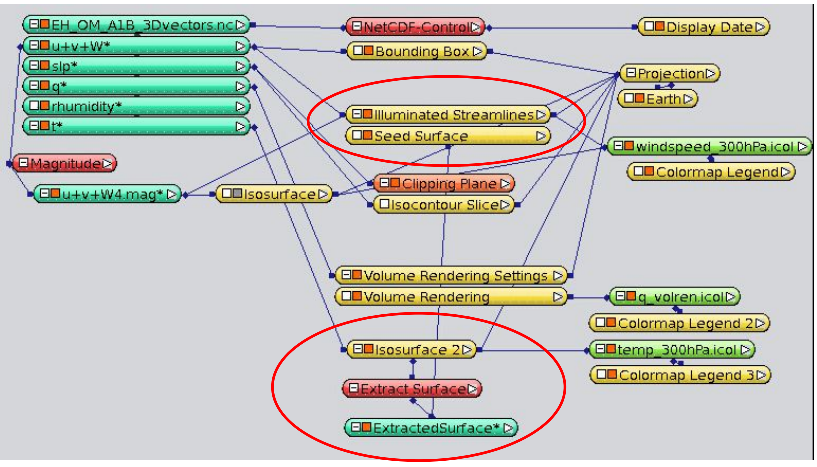
Seed Surfaces can be derived from isosurfaces or Height Map Slices. As an example, let us have a closer look at the wind field close to the frontal system of the 5°C temperature isosurface. In order to use this isosurface as a *Seed Surface*, we first have to convert it into *Avizo’s* internal *surface* format with the help of the *Extract Surface* function.

Extraction of an Avizo Surface

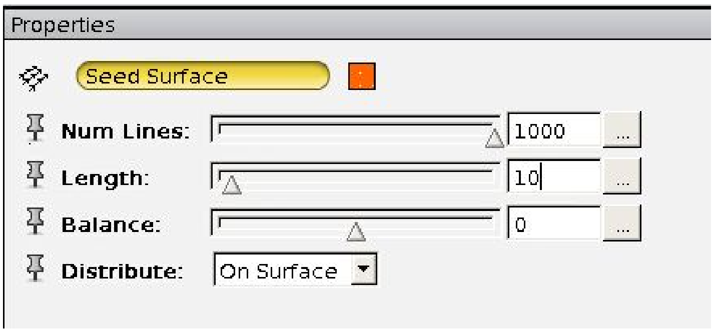
- Set your temperature isosurface to a threshold of 278.15 K (= 5°C).
- Right-click the Isosurface and press *Extract Surface*. This is a red module that can construct a surface in the Avizo surface format with the extension **.surf*.
- In the *Extract Surface* Properties click *Apply* to compute this surface. The computed surface is the green data module called *ExtractedSurface* in your Main Panel window.

Application of a Seed Surface

- In order to compute streamlines of the wind field with seed points across this 5°C surface, right-click the *Illuminated Streamlines* module in your Main Panel window, select *Seed Surface* and click *Create*. The new *Seed Surface* module is attached to the Illuminated Streamlines module and is automatically connected to the first surface found in your project – which is the *ExtractedSurface* module:



- In the *Seed Surface* Properties, choose 1000 lines with a length of 10 line segments to be distributed on the surface. If the distribution mode *“At Vertices”* is chosen, the lines start at each vertex of the surface. If *“On Surface”* is chosen, a user-defined number of streamlines are placed across the surface. The latter is what we need here.
- Retain the default *balance* value of 0, which results in streamlines that are equally long in forward and backward direction. A value of -1 would indicate that the streamlines extend in a backwards direction only, whereas a value of +1 would indicate that they extend in forward direction only.
  Tip: A balance value of +1 might be very useful if you need a screenshot for a print publication because you cannot animate your streamlines in order to show their direction. But if you want to examine your data interactively or produce a video, then animation (with *balance* = 0) is the better way to show the direction of the streamlines.
- Click *Apply* and *auto-refresh* in the *Seed Surface* Properties to compute the new streamlines.



Properties

Seed Surface

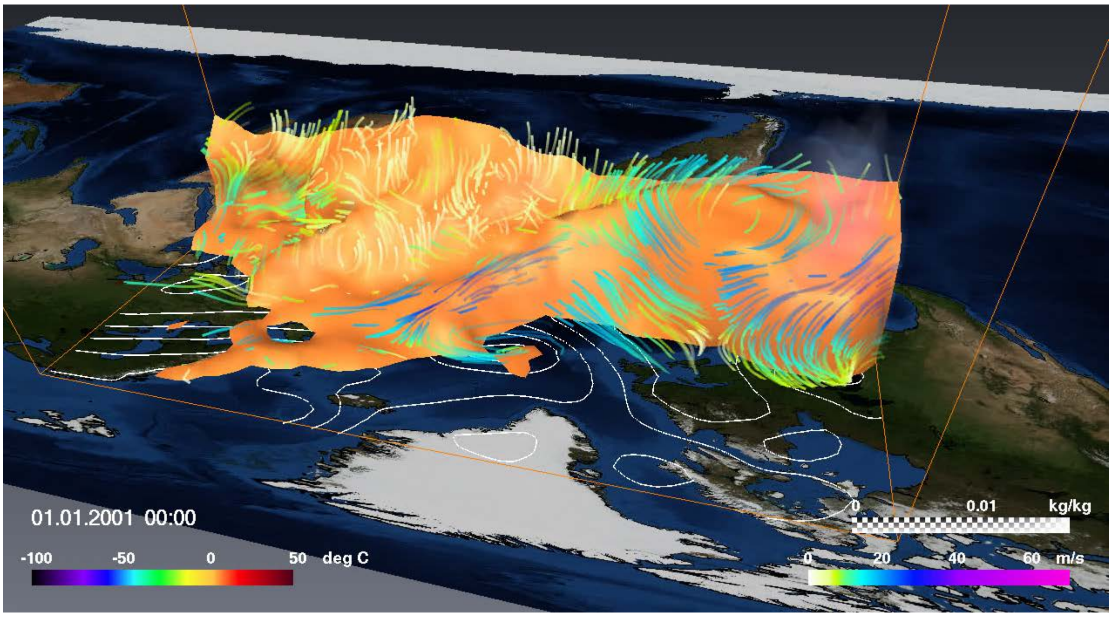
Num Lines: 1000

Length: 10

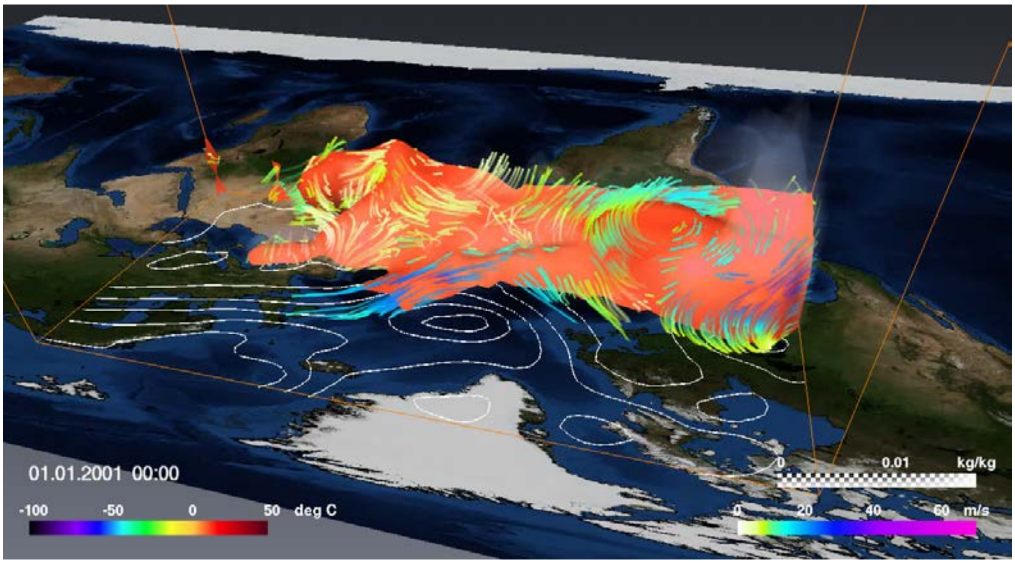
Balance: 0

Distribute: On Surface

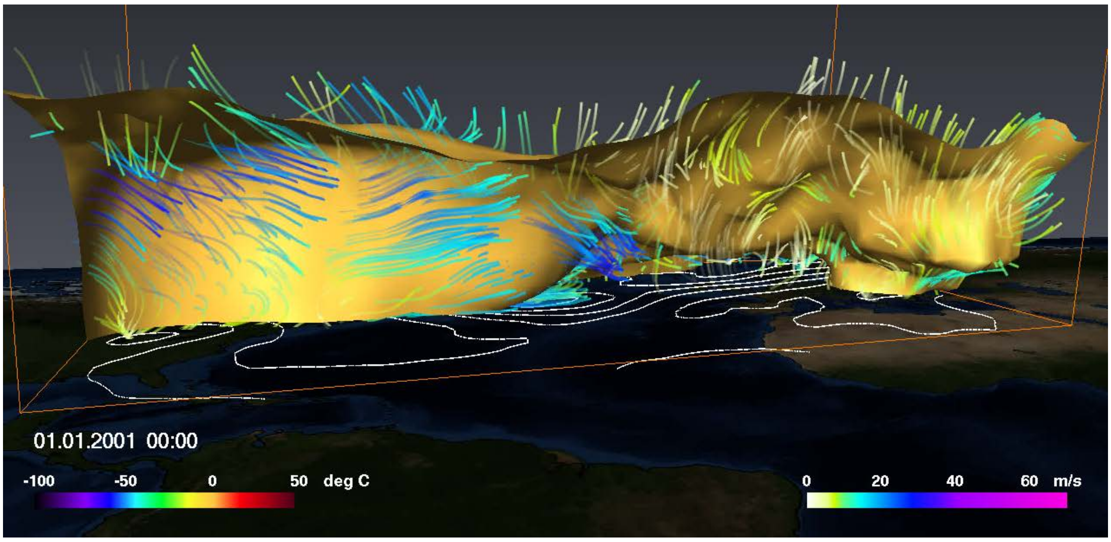
- The streamlines of the *Seed Surface* are being constructed in addition to the streamlines of the *Illuminated Streamlines* module. So, if you want to visualize only the streamlines seeded by this surface, you will have to set the number of lines (Num Lines) in the *Illuminated Streamlines* module to 0.



- In the *Illuminated Streamlines* Properties animate your streamlines now. This gives you a vital overview of the wind system close to the 5°C frontal isosurface.
- In the red *Extract Surface* module activate auto-refresh. Now you can change the threshold of the temperature isosurface, e.g. to 283.15 (= 10°C, see figure below) and both the *ExtractedSurface* and the *Seed Surface* will be refreshed on the fly.



- Or you might be interested in the wind field from the southern side of the front. Below you can see the southern view on the 273.15 K (= 0°C) surface.



- Save your project as streamlines_*SeedSurface_3D.hx*.

The corresponding project in the tutorial folder is called *TUT_streamlines_SeedSurface_3D.hx*.

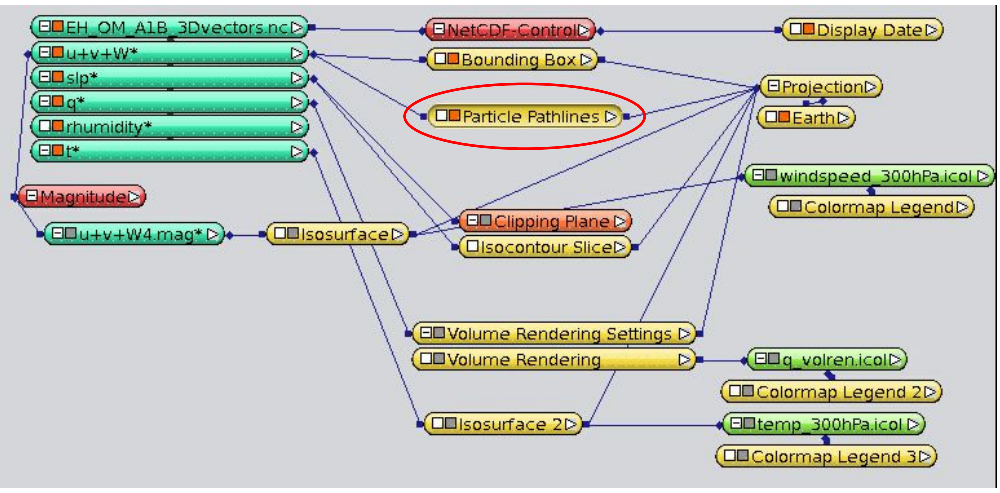
Tip: If you want to reuse one of these extracted surfaces again in another project, you can save them in the *"HxSurface binary"* format **.surf*. To do this: right-click on the green *ExtractedSurface > Save Data As*.

6.4. Trajectories

As described above, trajectories are defined as pathlines of “weightless” particles in time dependent flow fields. The 3D velocity field is used to compute the advection of these particles.

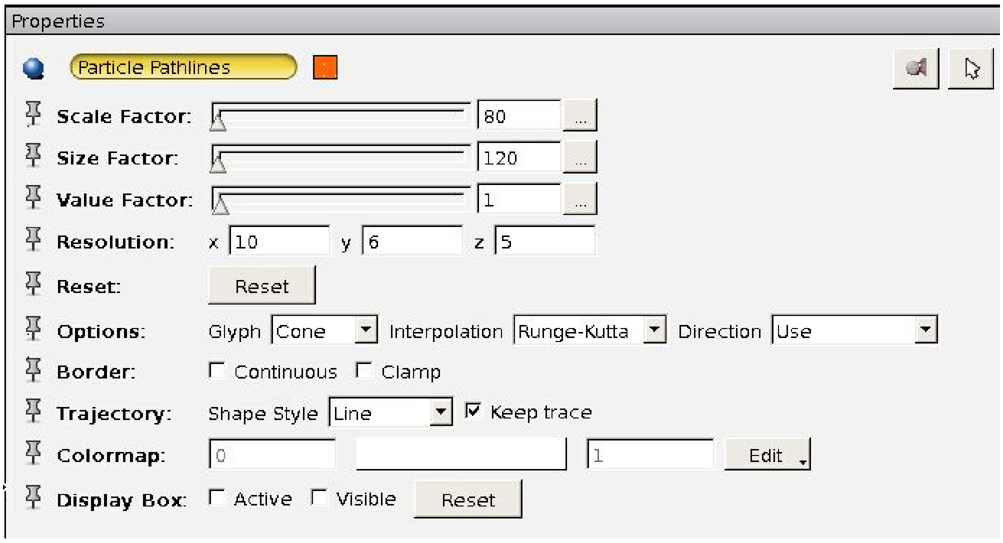
In Avizo, the *Particle Pathlines* module allows us to visualize the advection of particles and their trajectories. For the visualization of the particles itself, two options are available: cones (default) or spheres. Cones have the advantage that they can show the local wind direction, similar to the wind arrows of the Vectors Slice module, and in contrast to the vector arrows, the cone glyphs are 3-dimensional. The path of the particles can be rendered as simple lines or tube-like structures.

- Load the project *TUT_streamlines_3D.hx* from the tutorial directory and save it as *trajectories_3D.hx* in your own folder. Delete the *Illuminated Streamlines* module and switch off the temperature and specific humidity modules.
- Right-click the vector *u+v+W**, then click ***Display → Particle Pathlines → Create*** and enable projection for the module. Your Main Panel window looks like this now:



- In the *Particle Pathlines* Properties, choose a **Scale Factor** of *80*. This factor determines the diameter of the cone base area.
- Choose a **Size Factor** of *120*. It regulates the height of the cones, which should be a bit bigger than their diameter.
- Keep the **Value Factor** to 1. The particle velocity is multiplied by this factor when particles are advected (by pressing the play button in the NetCDF-Control module).
- Set the number of cones to *10 x 6 x 5* (**Resolution**).
- In the **Options** port, keep the default setting for *Glyph* (*Cone) and Direction (Use)*, but set the *Interpolation* to *Runge Kutta* (which is more complex than the *Euler* method). Keep the Direction mode “*Use*”, which means that the particles (here cones) are oriented according to the local velocity vector.

- In the **Trajectory** port, select the **Trajectory Shape Style** *Line* and activate *Keep trace*.
- Change the colour of the cones from yellow to opaque white (double-click in the colormap) to increase the contrast to the underlying earth.



Properties

Particle Pathlines

Scale Factor: 80

Size Factor: 120

Value Factor: 1

Resolution: x 10 y 6 z 5

Reset: Reset

Options: Glyph Cone Interpolation Runge-Kutta Direction Use

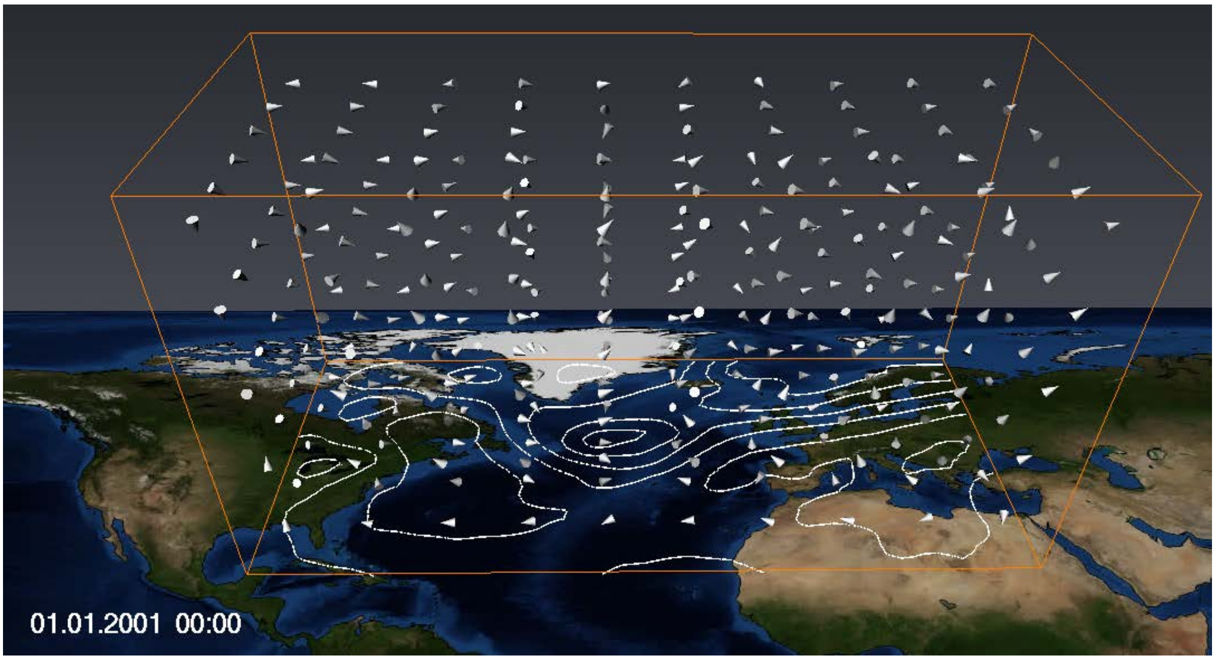
Border: □ Continuous □ Clamp

Trajectory: Shape Style Line ☑ Keep trace

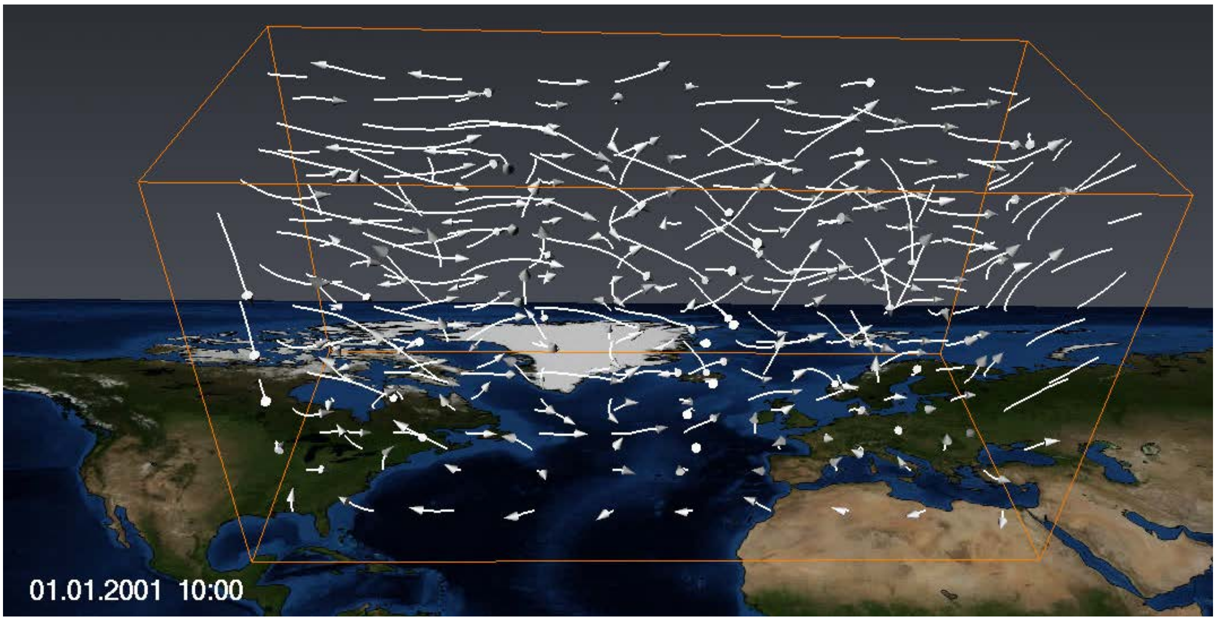
Colormap: 0 1 Edit

Display Box: □ Active □ Visible Reset

In your viewer window you should see 10 x 6 x 5 = 300 small cones, showing the wind direction of the current time step (here: base time step 1).



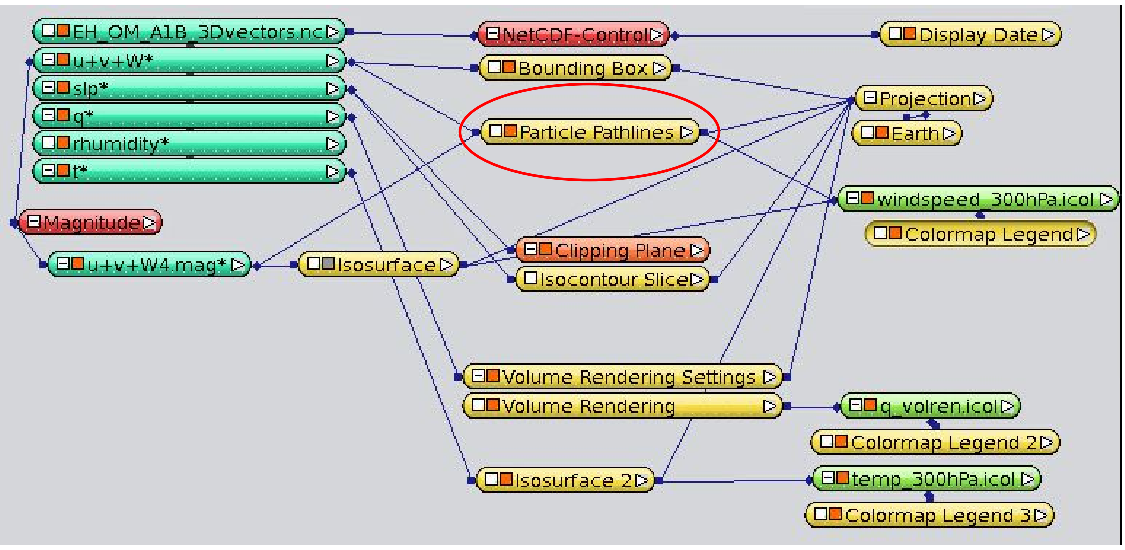
- In the *NetCDF-Control* Properties move forward from time step 1 to time step 2. The cones will move according to the local wind direction and velocity. After 10 hours, that is at time step 11 (01.01.2001 10:00), the cone particles have moved along these trajectories:



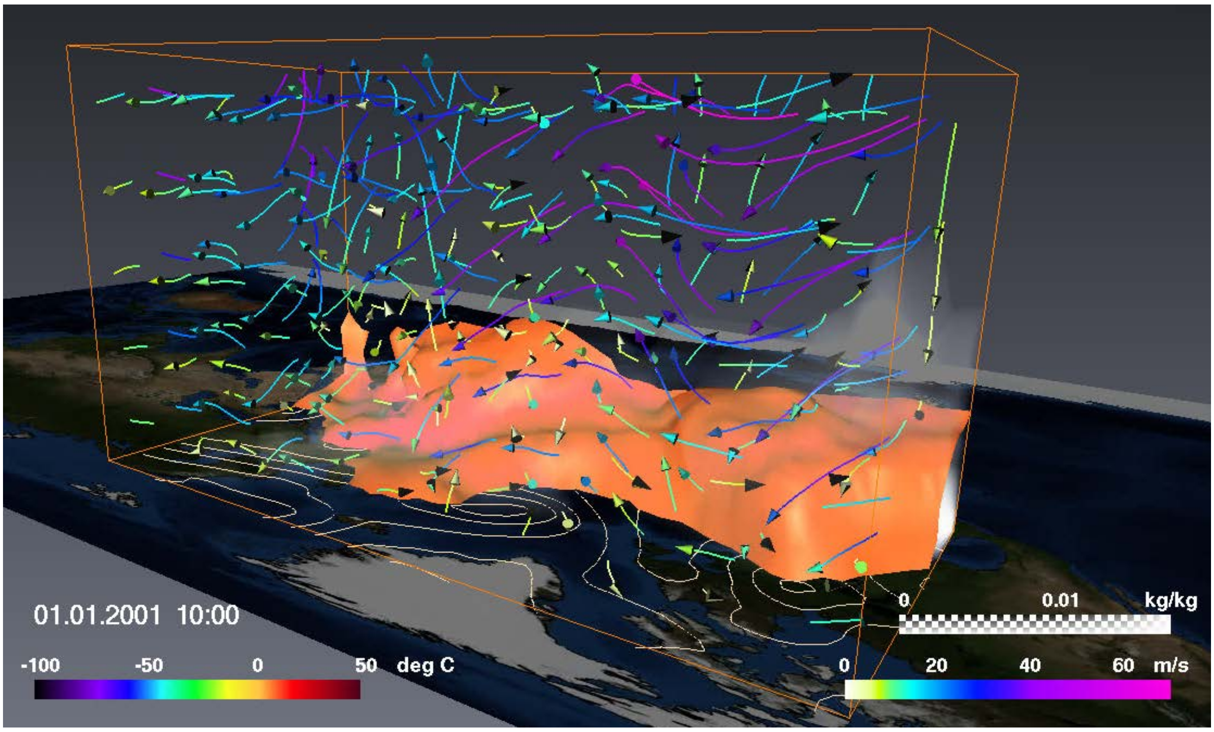
- Save your project as *trajectories_3D.hx*.

The corresponding project in the tutorial folder is called *TUT_trajectories_3D.hx*.

- As possible within most methods, you can also colorize the Cones and trajectories: In the Main Panel window, right-click the *Particle Pathlines* module, connect the ColorField to the wind speed magnitude *u+v+W.mag*,* and connect the Colormap port to *windspeed_300hPa.icol*.



- You will have to reset the cone particles in the *Particle Pathlines* Properties first and go through the time steps again to colorize the lines of the cones.
- If you also switch on the slp Isocontour Slice, the temperature isosurface and the specific humidity volume rendering again, you'll get the following figure:



The current project can also be found in the tutorial folder with the filename *TUT_trajectories_3D_Trj.hx*. You may also have a look into the video *V_trajectories.mpg* showing the movement of the trajectories from time step 1 to 11.

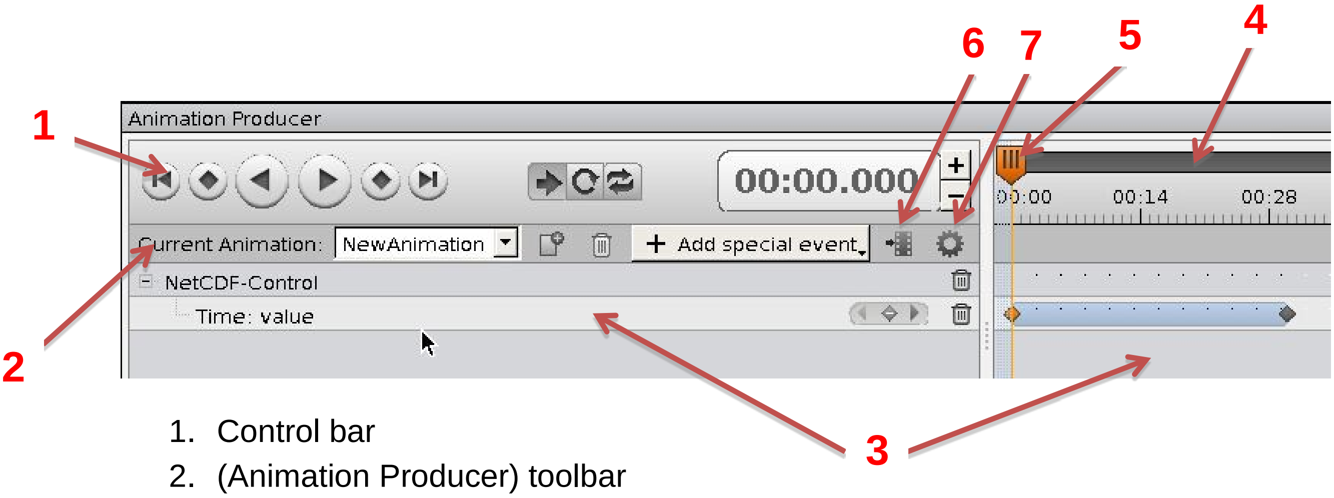
7. Animations and Videos

If you want to produce a simple video with a visualization of time dependent data with a fixed view, you will only need the *MovieMaker*.

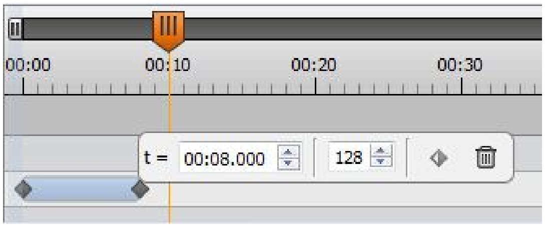
More complex animations or series of animations can be set up with the *Animation Producer* module. Animation Producer was first introduced with Avizo 7.1 and replaces the deprecated DemoMaker module.
With the *Animation Producer*, you can combine and synchronize time animations, camera rotations, and movements of 2D slices etc., as well as switch modules on or off. Basically, all ports (parameters) of the active modules can be changed. Even complex animations, like time animation combined with rotation of the view or a moving camera position can be accomplished by using the *Camera Orbit* (see chapter 7.1) or the *Camera Path* (see chapter 7.2) module.

After you have finalized the choreography of your animation in the *Animation Producer*, the result can be saved in form of an MPEG-1 video or as a sequence of single image files (see chapter 7.4).

The Animation Producer module is activated or deactivated by pressing the Animation Producer button Animation Producer in the toolbar.



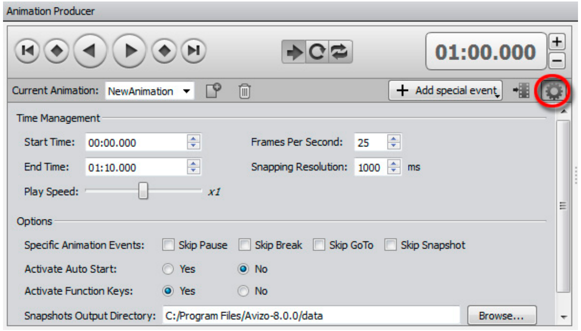
1. Control bar
2. (Animation Producer) toolbar
3. Event list
4. Timeline
5. Master Time Slider
6. “Movie Creation” Menu
7. “More Options” Menu



A new widget becomes visible, hosting the Animation Producer’s user interface. Clicking on the stopwatch button creates a new keyframe in the Animation Producer timeline and the event is listed in the left panel of the user interface. If you hold the mouse cursor above the small orange diamond symbol in the timeline panel, it will activate a small input field where you can adjust the time and the accompanying value for the port that it’s associated with.

In order to adjust the schedule, you can simply drag the diamond icon to the desired position on the timeline.

Time Management: In order to define the length of your animation storyboard, you can open the “More Options” menu (7.) and set parameters such as the start and end time and number of frames per second.



Animation Producer

01:00.000

Current Animation: NewAnimation

Time Management

Start Time: 00:00.000

End Time: 01:10.000

Play Speed: x1

Frames Per Second: 25

Snapping Resolution: 1000 ms

Options

Specific Animation Events: □ Skip Pause □ Skip Break □ Skip GoTo □ Skip Snapshot

Activate Auto Start: ○ Yes ● No

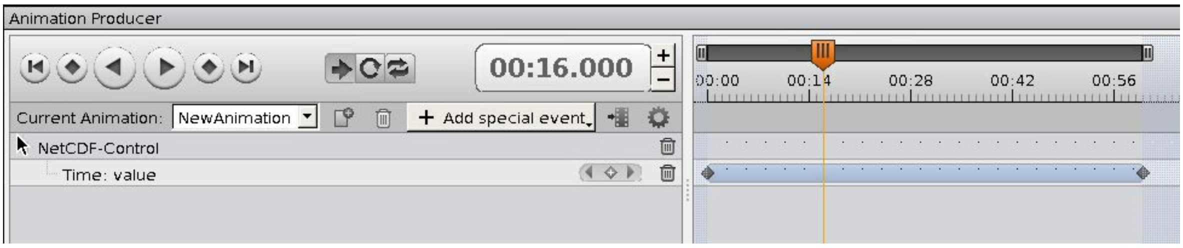
Activate Function Keys: ● Yes ○ No

Snapshots Output Directory: C:/Program Files/Avizo-8.0.0/data

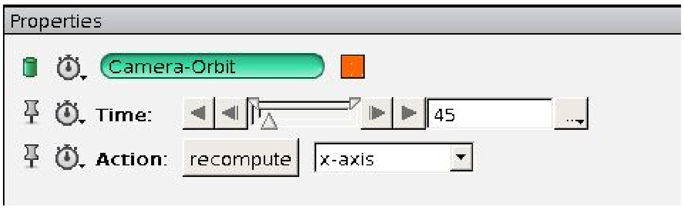
7.1. Toggle Modules on/off and Camera Rotation

We are going to animate our project of chapter 4.7 (*Bar Chart Slice* of precipitation colorized by the temperature field) for the summer months June to August 2001. In this animation, we will start with the temperature *Slice* only in an XY view (top view), then rotate it 45° backward and at the same time switch on the *Bar Chart Slice* and finally watch oblique views from the left and right side. For a better orientation you may want to view the final video in advance in a Media Player: *prec_temp_2D.mpg*.

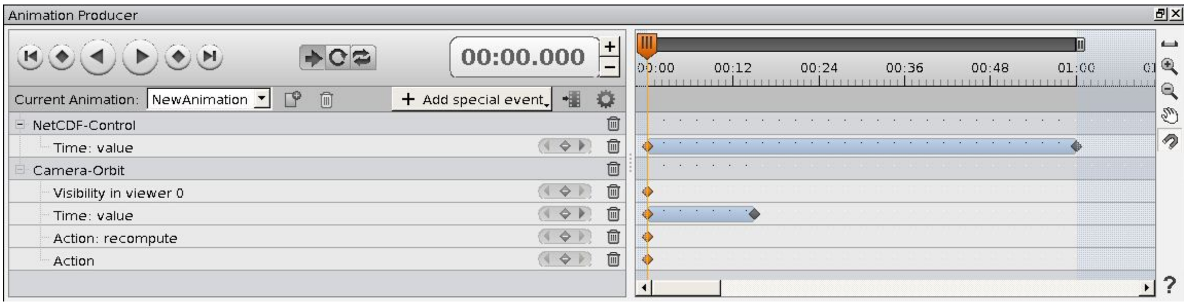
- Open the project *TUT_temp_prec_2D.hx* from your tutorial folder.
- Switch off the *Bar Chart Slice* and its *Caption 2* and switch on the *Slice* including its colormap and *Caption*.
- Put the *Slice* into *XY View*.
- Open the NetCDF-Control Properties panel.
- From the toolbar, click on the ***Animation Producer*** button Animation Producer. We want to animate the months June to August within the whole time period, that is, time steps **605** (01.06.2001 0:00) to **972** (31.08.2001 18:00):
    - In the NetCDF-Control Properties panel click on the stopwatch button left of **Time** and choose *Time: value*. A new keyframe appears in the left panel of the **Animation Producer** user interface, called NetCDF-Control -- Time: value.
    - In the Animation Producer user interface, move the mouse above the orange diamond icon and set the second value to 605.
    - Move the mouse on the timeline to 60 seconds; double-click in the line of the NetCDF-Control Time: value, which will insert a new keyframe. Move the mouse above the new keyframe and change the second value to 972 (the first value should be t=01:00:000).
    - Move the Master Time Slider back to the start (t=0) and click on the play button in the control bar.



- During the time 00:00 and 00:15 the *Slice* should be inclined 45° backwards.
  - In the main menu bar, click ***Project*** → ***Create Object*** → ***Animations And Scripts*** → ***Camera Orbit*** → ***Create***.
  - In the *Camera Orbit* Properties, you may test the five offered 360° rotations. Finally, select *x-axis,* set the size of the Viewer Window and its contents as you like, and click *recompute*.

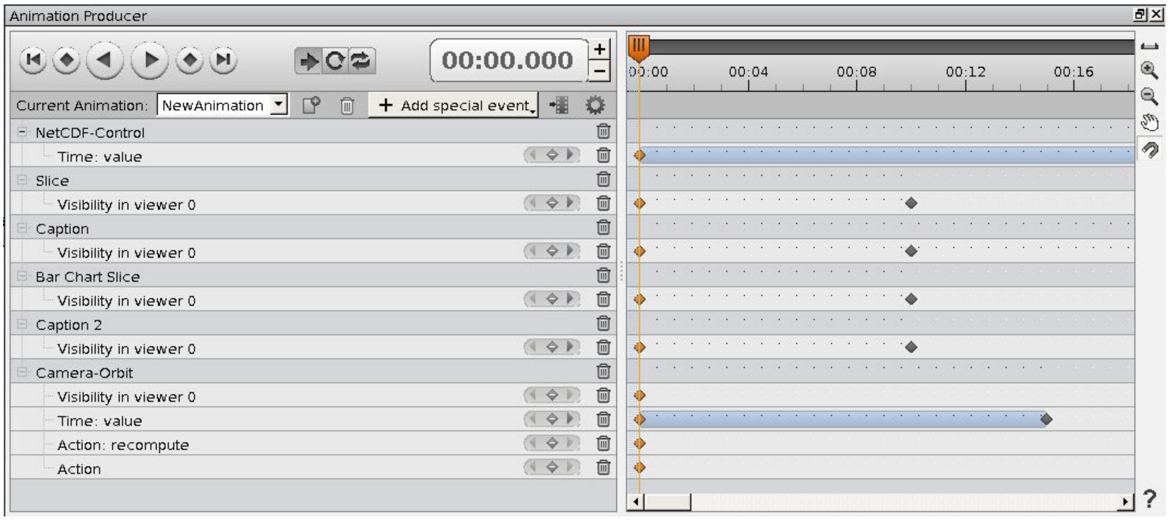


  - In the *Camera-Orbit Properties* click on the stopwatch left to **Camera-Orbit** and choose ***Visibility in viewer 0***.
  - In the *Camera-Orbit Properties* click on the stopwatch left to **Time** and choose ***Time: value***.
  - In the *Camera-Orbit Properties* click on the stopwatch left to **Action** and choose ***Action***.
  - In the *Camera-Orbit Properties* click on the stopwatch left to **Action** and choose ***Action: recompute***.
  - In the Animation Producer user interface event list, move the mouse above the first orange diamond icon of the *Camera-Orbit Time: value* keyframe. The first value should be 00:00:000 and the second 0.00. At 15 seconds, double-click in the timeline event list and move the mouse above the new keyframe icon to set the first value to 00:15:000 and the second value to 45.0.
  - Move the mouse above the orange diamond icon of the *Camera-Orbit Action* entry. The second value must be *x-axis*.
  - Move the Master Time Slider back to the start (t=0), click *recompute* in the *Camera-Orbit Properties*, and click on the play button in the control bar.



- After 10 seconds of the rotation, we want to switch *on* the *Bar Chart Slice* and its *Caption* and switch *off* the *Slice*. We must add four events to do so:

- In the Caption Properties click on the stopwatch left to **Caption** *and select **Visibility in viewer 0***.
- In the Animation Producer user interface event list move the mouse above the first orange diamond icon of the Caption keyframe. The first value should be 00:00:000 and the toggle should be set to **on**.
- At 10 seconds, double-click in the timeline event list and move the mouse above the new keyframe icon to set the first value to 00:10:000 and set the toggle to **off**.
- Respectively, you can switch off the Slice using the stopwatch left of the Properties of *Slice – Visibility in viewer 0.* Go on as previously described.
- In the Caption 2 Properties, click on the stopwatch left of **Caption 2** *and select **Visibility in viewer 0***.
- In the Animation Producer user interface’s event list, move the mouse above the first orange diamond icon of the Caption 2 keyframe. The first value should be 00:00:000 and the toggle should be set to **off**.
- At 10 seconds, double-click in the timeline event list and move the mouse above the new keyframe icon to set the first value to 00:10:000 and set the toggle to **on**.
- Respectively, you can switch on the Bar Chart Slice using the stopwatch left to the Properties of *Bar Chart Slice – Visibility in viewer 0*.
- Now, play the animation. Don’t forget to click *recompute* in the *Camera-Orbit Properties* before you start the animation.



After the inclined view on the Bar Chart Slice is reached by the time 15 seconds, we want to change the camera position again and watch the animation from both sides. To do so, we rotate the figure around its z-axis about 40° to the right, go back, and then rotate it by 40° to the left.

In the Animation Producer’s user interface:

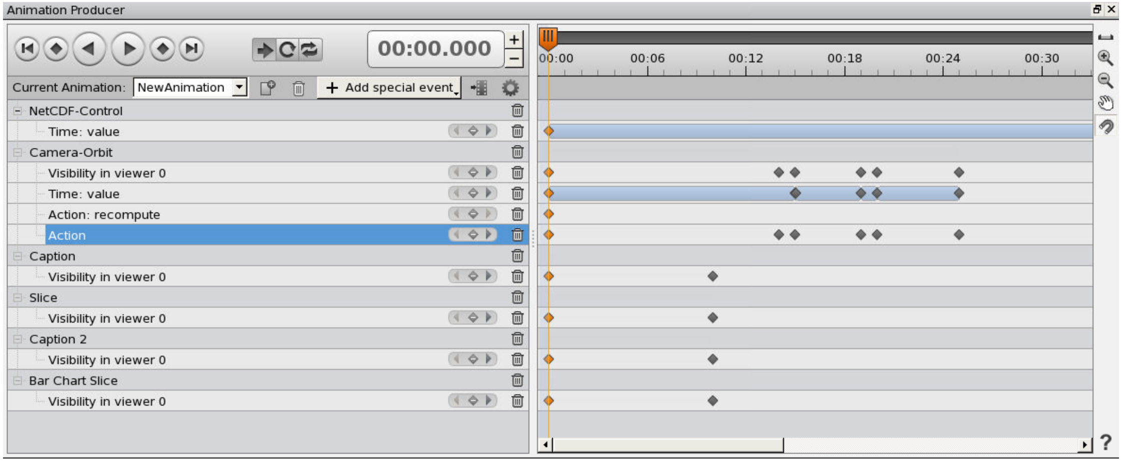
- Insert a new keyframe to the ***Camera-Orbit Action*** (double-left-click) at t=14s. The second value must be ***x-axis***.

- Insert a new keyframe to the ***Camera-Orbit Visibility in viewer 0*** at t=14s and toggle it ***off***.

- Insert a new keyframe to the ***Camera-Orbit Action*** (double-left-click) at t=15s. The second value must be ***z-axis***.
- Insert a new keyframe to the ***Camera-Orbit Visibility in viewer 0*** at t=15s and toggle it ***on***.
- Insert a new keyframe to the ***Camera-Orbit Time: value*** at t=15.040s and set the second value to 0.0.

- Insert a new keyframe to the ***Camera-Orbit Action*** (double-left-click) at t=19s. The second value must be ***z-axis***.
- Insert a new keyframe to the ***Camera-Orbit Visibility in viewer 0*** at t=19s and toggle it ***off***.
- Insert a new keyframe to the ***Camera-Orbit Time: value*** at t=19s and set the second value to 40.0.
- Insert a new keyframe to the ***Camera-Orbit Action*** (double-left-click) at t=20s. The second value must be ***z-axis***.
- Insert a new keyframe to the ***Camera-Orbit Visibility in viewer 0*** at t=20s and toggle it ***on***.
- Insert a new keyframe to the ***Camera-Orbit Time: value*** at t=20s and set the second value to 360.0.

- Insert a new keyframe to the ***Camera-Orbit Action*** (double-left-click) at t=25s. The second value must be ***z-axis***.
- Insert a new keyframe to the ***Camera-Orbit Visibility in viewer 0*** at t=25s and toggle it ***off***.
- Insert a new keyframe to the ***Camera-Orbit Time: value*** at t=25s and set the second value to 320.0.



Current Animation: NewAnimation

00:00.000

- Watch the whole animation by pressing the *play* button. If the rotations differ from what you have intended, make sure that you have put the view into starting position (*View XY*) and that you have clicked *recompute* in the *Camera Orbit Properties* in advance.
- Save your project.

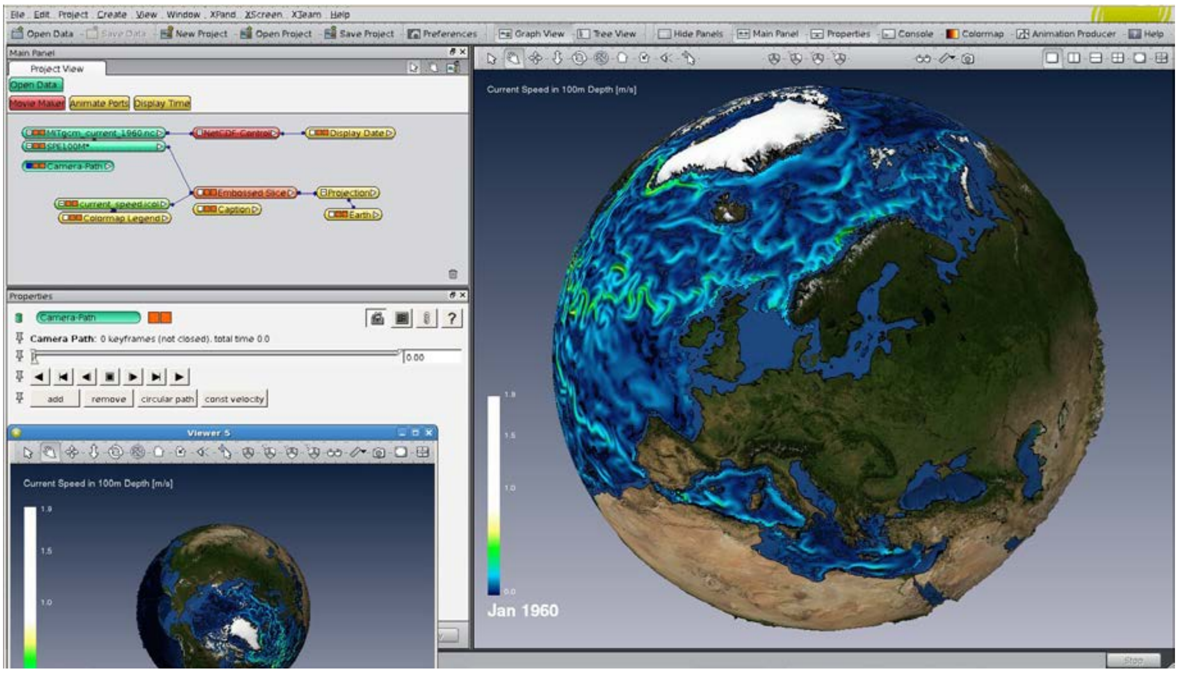
You can find this project in your tutorial folder: *AnimProd_temp_prec_2D.hx*. To export this animation to an mpeg video file, see chapter 7.4.

7.2. *“Flying Cameras”: Using Camera Path*

The *Camera Path* module allows you to move the virtual camera through space, while the *Camera Orbit* module, which we got to know in the last chapter, only allows to rotating the camera around the center of the viewing space.

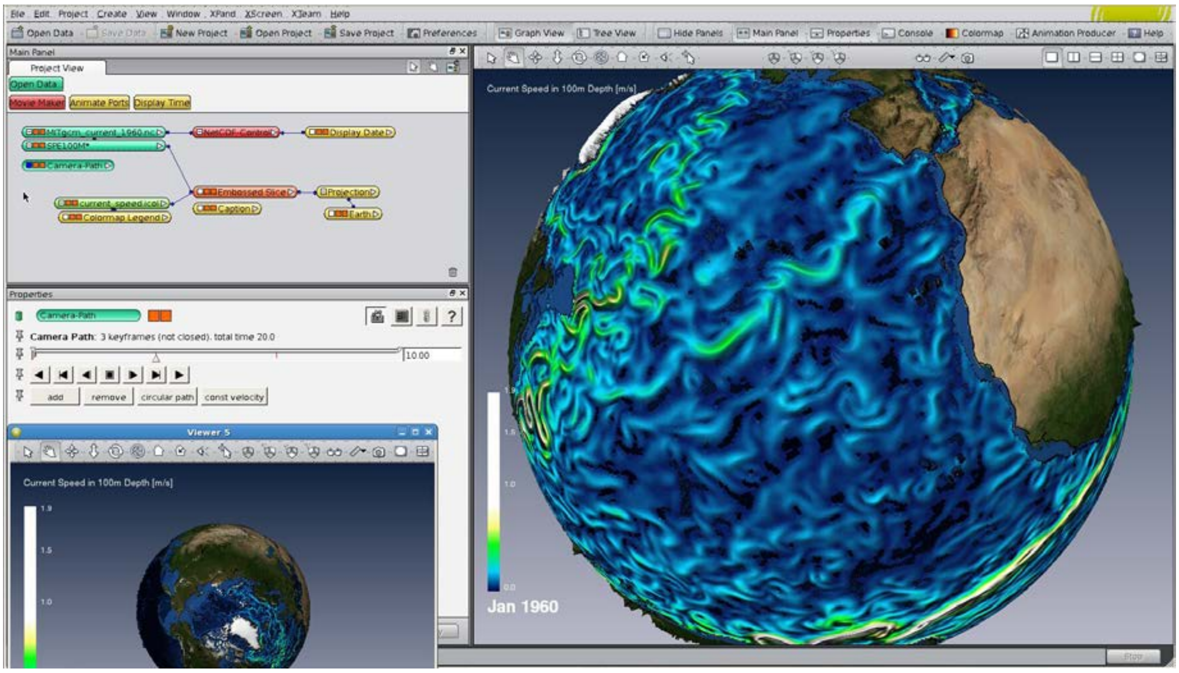
We are going to use the *Camera Path* module for our visualization of chapter 4.11→ (Current speed as *Embossed Slice*). We will set up a camera path, starting from a European view flying to a perspective view onto the Gulf of Mexico.

- Open the project *TUT_current_Embossed Slice_2D.hx* (result of chapter 4.11). Choose a view onto Europe to start with.
- Click ***Create → Animations And Scripts → Camera Path → Create***.
- To define a path, open the *Camera Path Editor* from the corresponding Properties panel. A new viewer, *Viewer 5*, opens. This could be used to visualize the positioning of your main camera (viewer 0) relative to the objects in your visualization. Move it aside, so that you can completely see your main Viewer.
- Zoom in a little bit, so that the whole *Earth* fills the viewer, and choose this view as the first keyframe of your *Camera Path* by clicking *add* in the Properties panel. *Avizo* adds a red vertical line in the *Camera Path* time slider at position 0.00 and jumps to the next position (10.00).



Camera Path: 0 keyframes (not closed), total time 0.0

- Then, use the navigation tools (Trackball, Translate, Zoom, …) to add two further views as keyframes.
  - One view across the Atlantic Ocean.

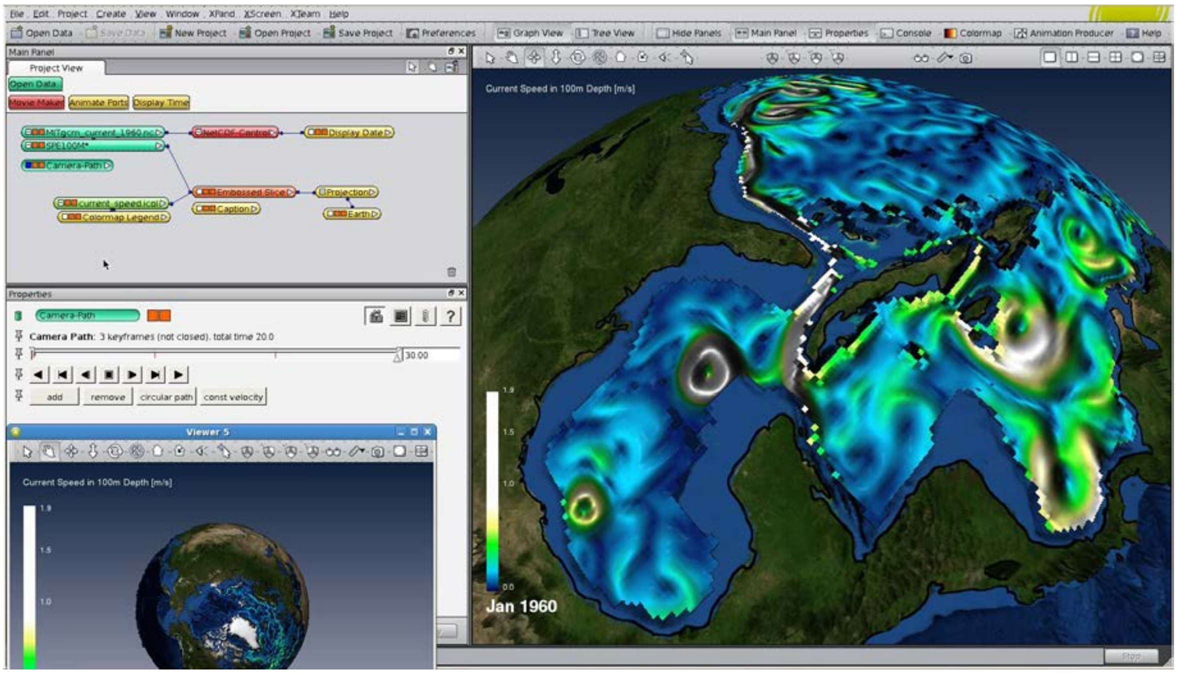


Camera Path: 3 keyframes (not closed). total time 20.0

10.00

add remove circular path const velocity

- View on the Caribbean Sea



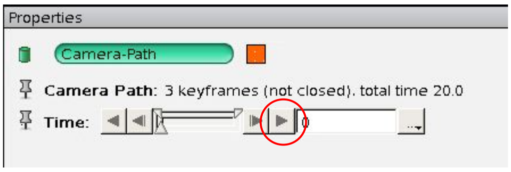
Camera Path: 3 keyframes (not closed). total time 20.0

30.00

add remove circular path const velocity

- If you want to change one of your keyframes, you can use the forward and backward buttons in the Properties window. Use these ⏮ ⏭ to jump to the last or next keyframe (red line). Use these ◀ ▶ to go forward and backward in small individual time steps to watch the frames *Avizo* computes automatically between the keyframes you have defined. Use the outer pair of buttons ◀ ▶ to watch your flight as an animation that you could stop by clicking ■.
- You might realize that the camera moves faster between keyframes 1 and 2 than keyframes 2 and 3. If you wish to get a constant flight velocity, click the button *“const velocity”* – then *Avizo* computes a smoother flight.

- After closing *Viewer 5*, you can watch your flight again by clicking the play button in the *Camera Path* Properties.



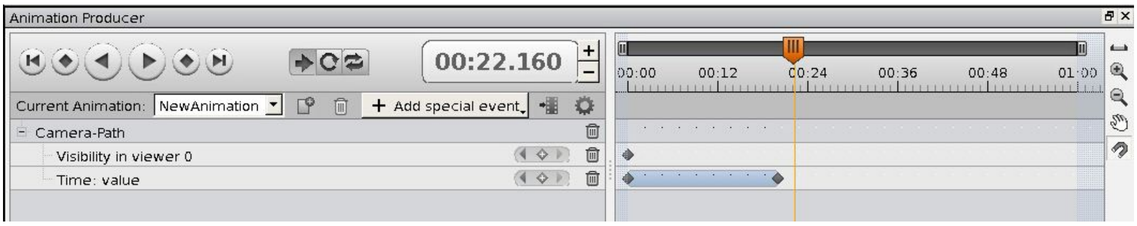
Properties

Camera-Path

Camera Path: 3 keyframes (not closed). total time 20.0

Time: 0

- After defining the camera path, you should save it. To do this: click ***File* → *Save Project*** and *Avizo* will prompt you to save the *Camera Path*. Choose *PathToCaribbean.civ* as the file name.
- Once the Camera Path is saved, you can include it in an animation. Click *Animation Producer*
- In the *Camera Path Properties*, select the stopwatch left of Camera Path and click on *Visibility on viewer 0*. Click on the stopwatch left of *Time* and choose *Time: value*. In the Animation Producer event list, insert a new keyframe and set the first value to 00:20:000 and the second to 20.0



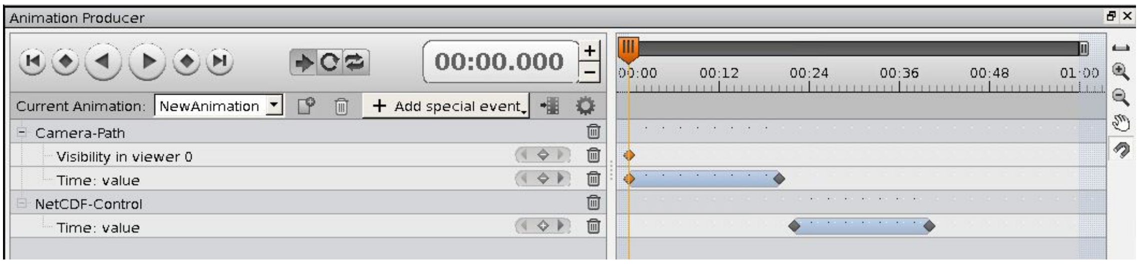
Animation Producer

00:22.160

Current Animation: NewAnimation

- Camera-Path
  - Visibility in viewer 0
  - Time: value

- After we have reached the Caribbean, we freeze the camera position for a short time of 2 seconds and then play a time animation from time steps 1 to 30 (Jan to March). Select the *NetCDF-Control Properties*. Click on the stopwatch left of *Time* and choose *Time: value*. In the Animation Producer event list, select the keyframe of the NetCDF-Control entry and set the first value to 00:22:000 and the second to 1.0. Move the mouse to 00:40 on the NetCDF-Control keyframe line and double-left-click to create a new keyframe. Set the first value to 00:40:000 and the second to 30.0.



Animation Producer

00:00.000

Current Animation: NewAnimation

- Camera-Path
  - Visibility in viewer 0
  - Time: value
- NetCDF-Control
  - Time: value

- Save your project.

The pre-defined version is called *AnimProd_current_Embossed_Slice_2D.hx.* To produce a movie from this demo, see chapter 7.4 for details.

7.3. *Moving 2D slices; changing Threshold and Transparency of Isosurfaces*

In this exercise you will combine four kinds of animation:

a) You will learn how to shift three differently oriented *Slices* of wind velocity.

b) You will be able to change the threshold of an *Isosurface* that begins by representing a wind velocity of 60 m/s and “grows” to an Isosurface of 30 m/s.

c) Three nested *Isosurfaces* representing the wind speed at 30, 40 and 60 m/s will be visualized, **alt**ernating between completely opaque and transparent.

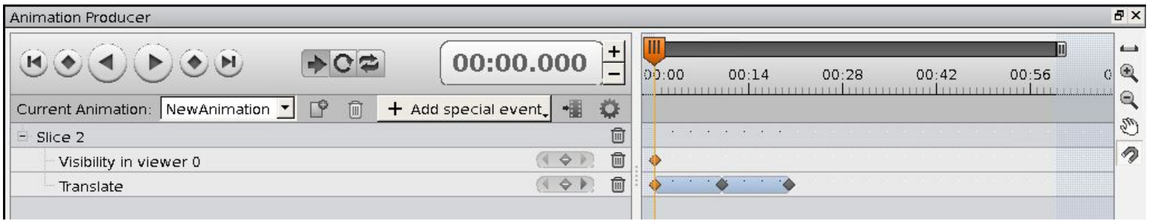
d) At the end of this animation, take the time dependence of the data into account again and show the change of wind velocity during the first 60 time steps in January (1.1.2001 0:00 – 15.1.2001 18:00).

To get an idea of the animation, you may watch it in advance. Have a look at *V_windspeeds.mpg*.

- Open the tutorial project *TUT_windspeeds_3D.hx* (which is basically the result of chapter 4.19 with a somewhat simplified and modified Main Panel window).
- Enable the *Animation Producer* user interface in your project (click on the ***Animation Producer** button in the main toolbar*).

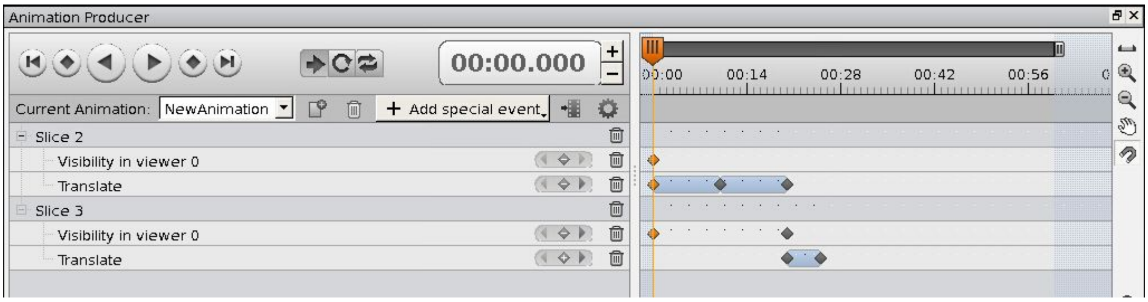
a) Moving three Slices in xy, xz and yz orientation:

- *Create a new Caption (**Create** → **Annotations**→ **Caption** → **Create**) “Moving Slices in xy, xz and yz orientation” on top of the viewer.*
- Switch ***on*** *Slice 2* (the one in XY orientation) and *Slice 3*, Slice 4 and *Slice* ***off***. Switch ***off*** all *Isosurfaces*.
- In the *Slice 2 Properties* click on the stopwatch left of ***Slice 2*** and select ***Visibility in viewer 0***. Click on the stopwatch left to ***Translate***.
- Until the point of time 00:10, Slice 2 shall be moved to the top of the grid (level 16) and then back down to level 4 until the time 00:20. In the Animation Producer event list of ***Slice 2 Translate,*** move above the keyframe and set the first value to 00:00:000 and the second to 0.0. Move the mouse to 00:10, insert a new keyframe (l-click) and set the first value to 00:10:000 and the second to 16.0. Move the mouse to 00:20, insert a new keyframe (left-click) and set the first value to 00:20:000 and the second to 4.0.

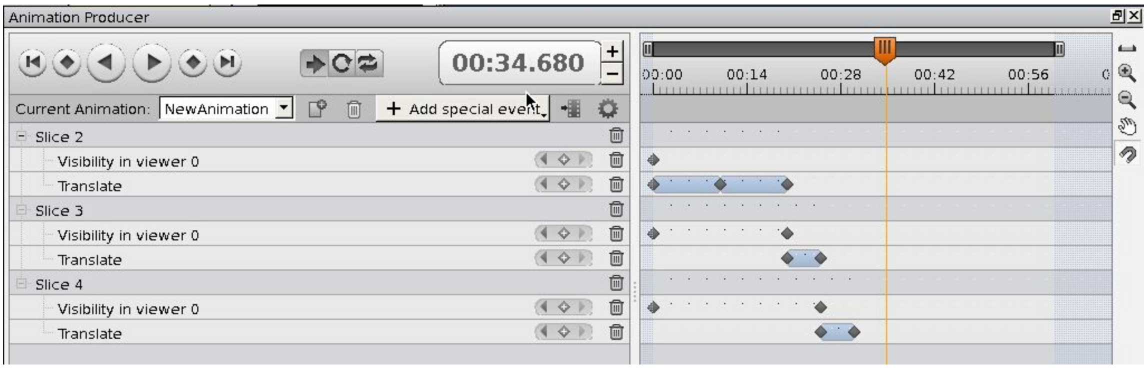


Current Animation: NewAnimation

- Go on as described before and add the ***Slice 3*** in XZ orientation and move it from rear side (position 95) to position 70 during the time period 00:20 to 00:25.

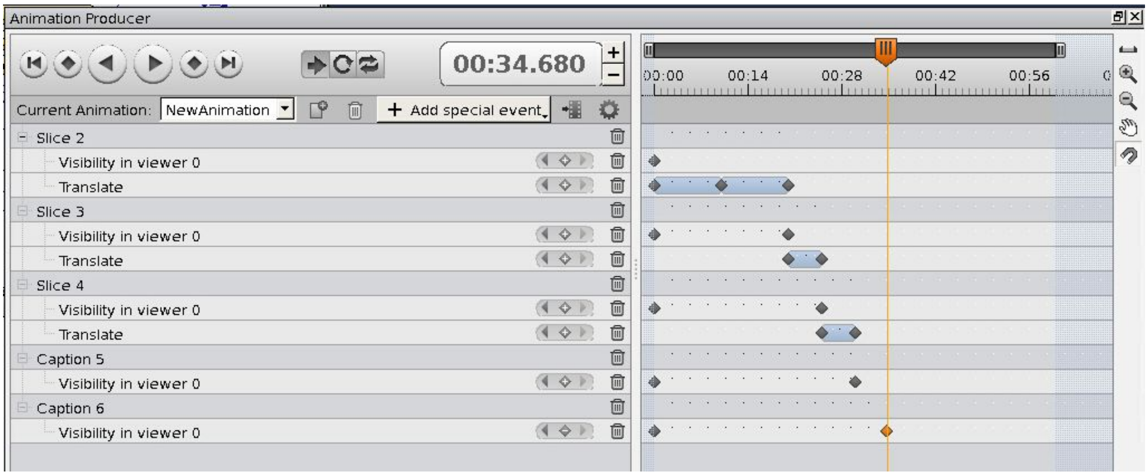


- Now, switch ***on*** the ***Slice 4*** in YZ orientation and move it from the left wall to position 175 during time interval 00:25 to 00:30.

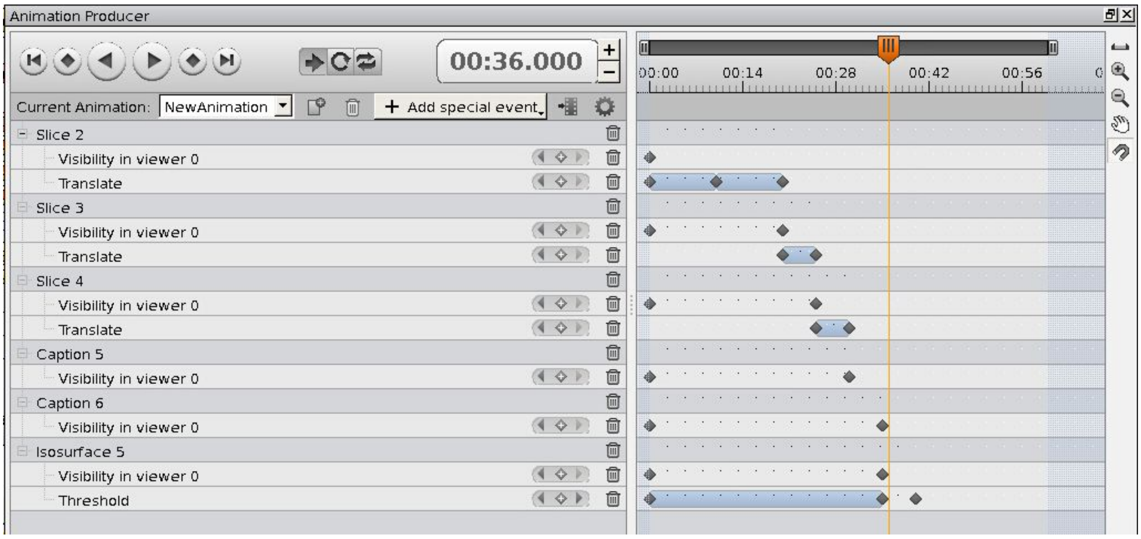


b) Growing isosurface

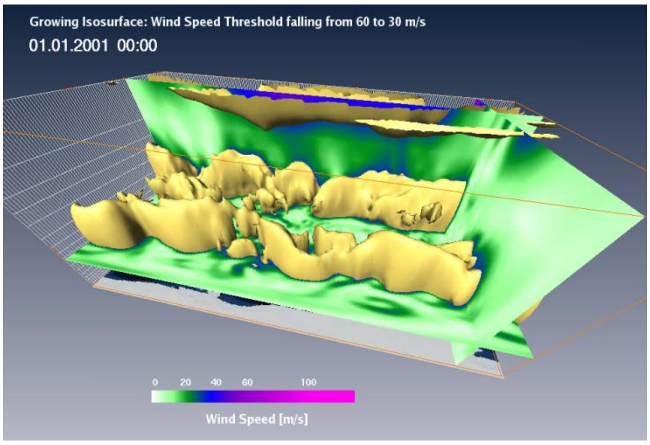
- Change the titles visible in your Viewer window – switch ***off*** the ***Caption*** saying *“Moving Slices in xy, xz and yz orientation”* and create a new Caption (***Create → Annotations → Caption → Create***) saying *“Growing Isosurface: Wind Speed Threshold falling from 60 to 30 m/s”.*



- Additionally we create a new *Isosurface*, set the *Draw Style* to *shaded* and the colormap to a single yellow color. Reduce its threshold during time period 00:30 to 00:40 from 60 to 30 m/s. Don’t forget to set auto refresh in the new Isosurface’s Properties.



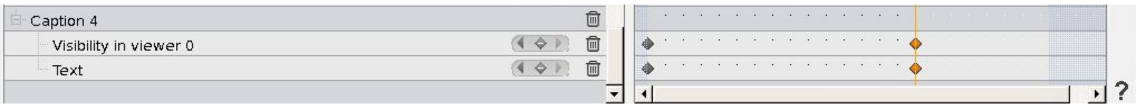
- Test the animation. At time 00:40 your viewer window should look something like this:


Growing Isosurface: Wind Speed Threshold falling from 60 to 30 m/s
01.01.2001 00:00
0
20
40
60
100
Wind Speed [m/s]

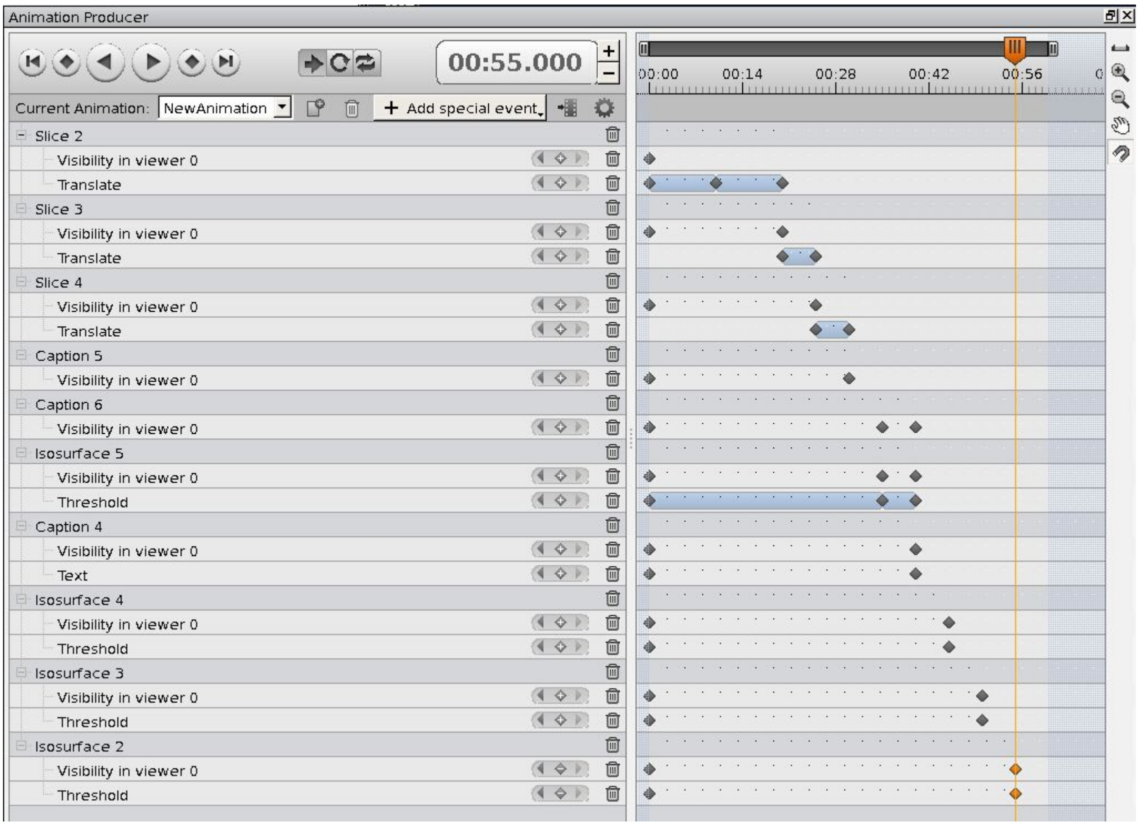
- In the *Animation Producer*, switch ***off*** the yellow *Isosurface* and its *Caption* at time 00:40.

c) Nested Isosurfaces

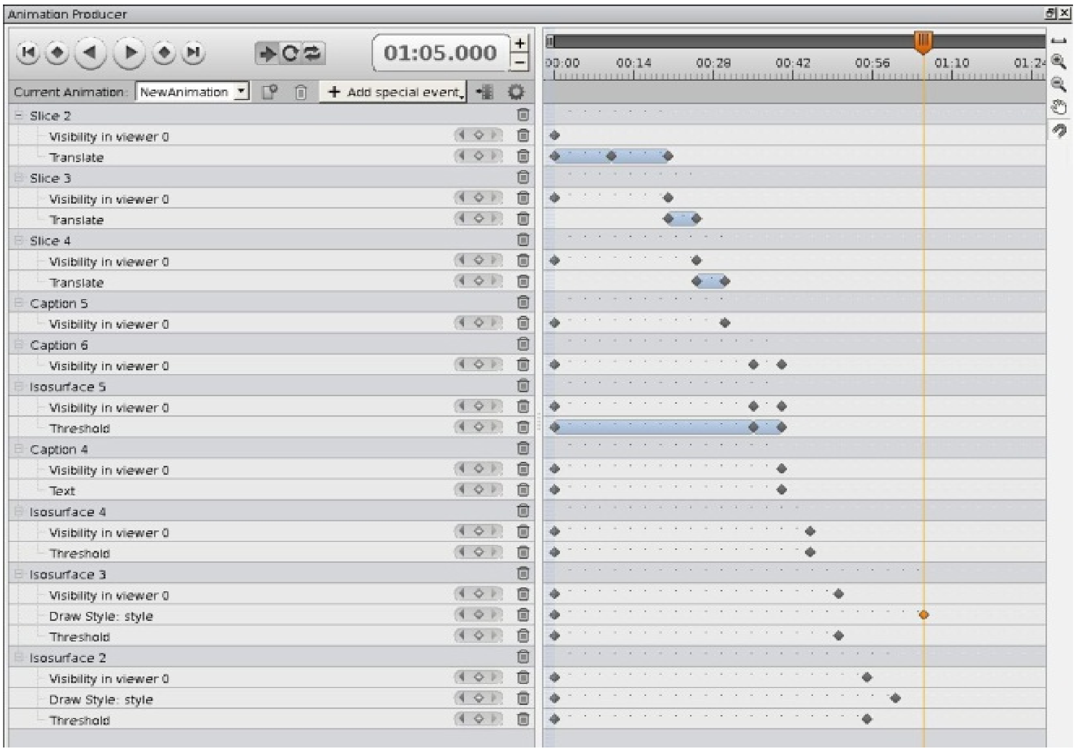
- Switch ***on*** *Caption 4* and change it to *“Nested Isosurfaces at 60 (purple), 40 (blue) and 30 (green) m/s”.*



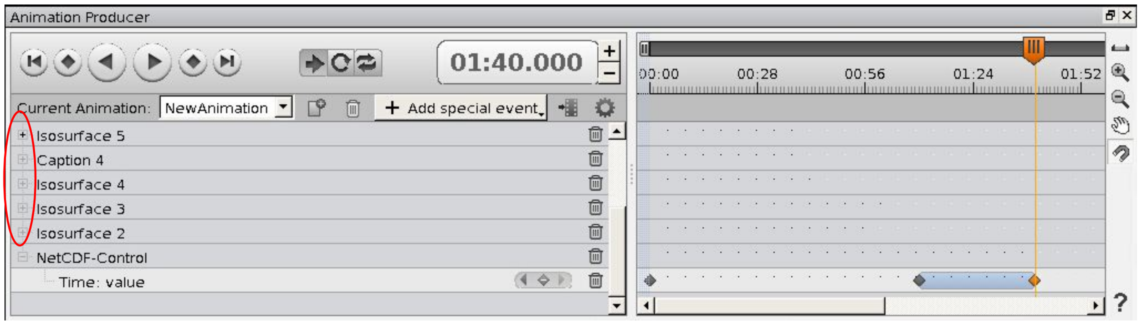
- At time points 00:45, 00:50 and 00:55, the isosurfaces, one after the other, shall be switched on: the purple isosurface (60 m/s), the blue one (40 m/s) and finally the green one (30 m/s).



- But in this way, only the outer isosurface will be visible during the animation. With all three isosurfaces switched on, change the ***Draw Style*** of the outer green one from *shaded* to *transparent* at time 01:00 and for the blue one at time 01:05. To do this, you have to click the stopwatch left of the *Draw Style* in the *Isosurface Properties* and choose *Draw Style: style*. In the Animation Producer event list, change the draw styles from *shaded* to *transparent*. Keep the inner purple isosurface opaque (shaded). Now, all three isosurfaces are visible at the same time (nested).



- During time interval 01:10 to 01:40, you may now animate the first 124 time steps representing January 2001. In the Animation Producer set the Master Time Slider to 01:10. Select the *NetCDF-Control Properties* and click on the stopwatch left of *Time* and choose *Time: value*. Set the second value in the right keyframe of the NetCDF-Control entry of the Animation Producer to 124. For a better overview, it is possible to collapse the Animation Producer event list entries.

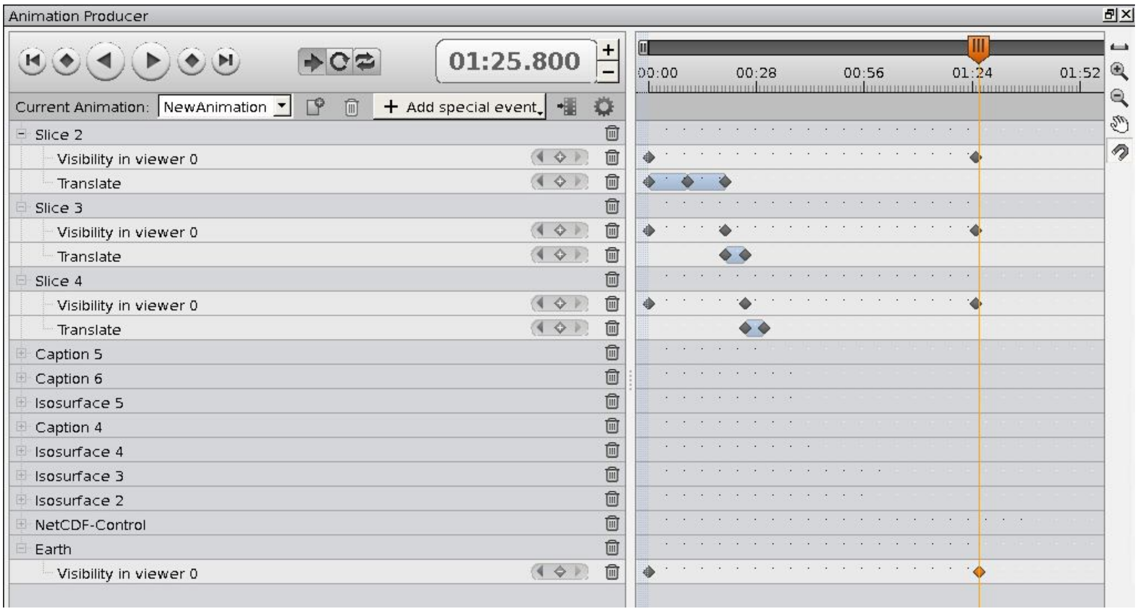


Animation Producer

01:40.000

Current Animation: NewAnimation

- The three *Slices* and the underlying *Earth* might bother you while watching the jetstreams closely – so turn them ***off*** at time point 01:25 for the last part of your animation. To turn ***off*** the Earth you have to select the stopwatch to the left of *Earth* in the *Earth Properties*. Choose *Visibility in viewer 0*.



Animation Producer

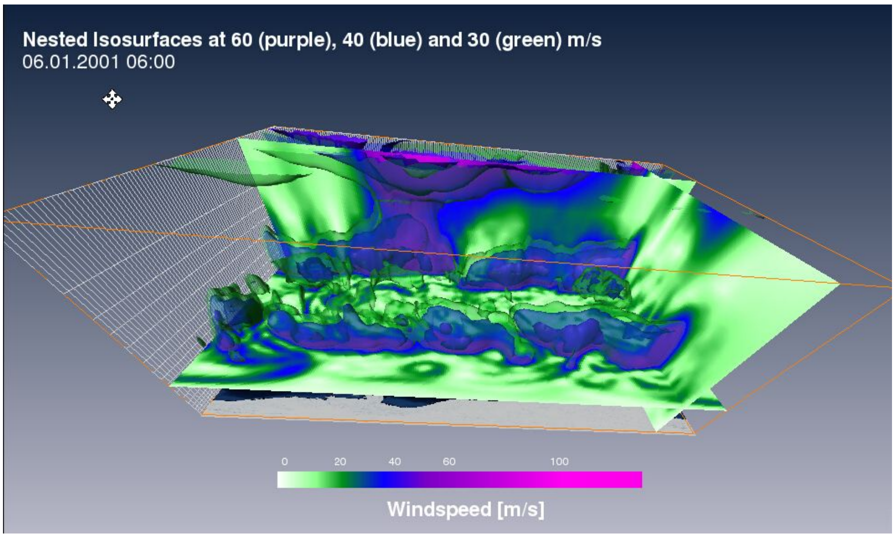
01:25.800

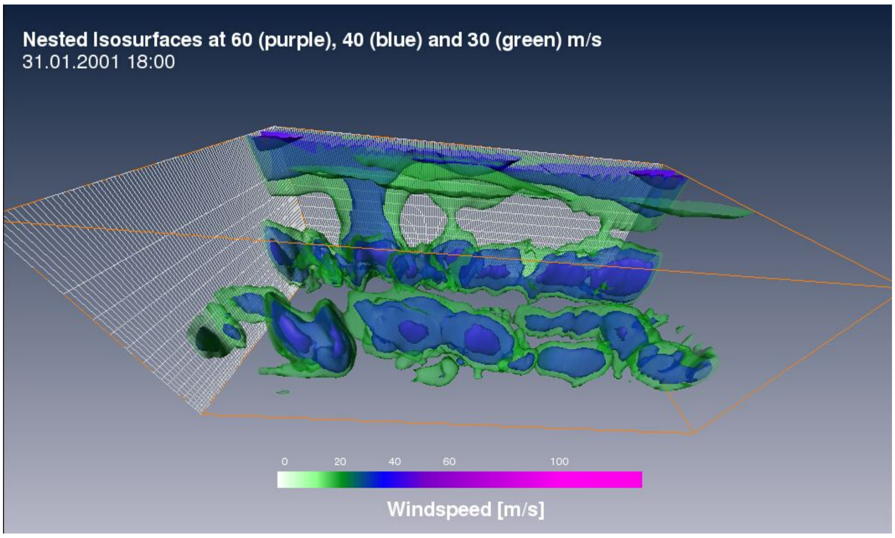
Current Animation: NewAnimation

- Finally, play the animation in the Animation Producer and check if everything works as you wish.

Remark: One of the limitations of the *Animation Producer* is that you should not delete or rename any modules of your project **after** the *Animation Producer* animation was set up. Internally, *Animation Producer* produces a script which contains the names of all used modules. Even the deletion and re-insertion of a module with the same name can lead to mismatches.

At the times 01:15 and 01:40, your visualization should look similar to these figures:


Nested Isosurfaces at 60 (purple), 40 (blue) and 30 (green) m/s
06.01.2001 06:00
0
20
40
60
100
Windspeed [m/s]


Nested Isosurfaces at 60 (purple), 40 (blue) and 30 (green) m/s
31.01.2001 18:00
0
20
40
60
100
Windspeed [m/s]

- In the *Animation Producer click on the Play Loop* button to set the animation to a *repeat mode*. An animation re-starting at the beginning automatically might be useful for presentations at fairs or conference stands. But please remember to reset the animation to *Play Once* mode before starting a video export … The *Play Swing* button will endlessly run your animation back and forth.
- Save your project, including your *Animation Producer* “choreography”, as *windspeeds_3D.hx*.

This project stage can also be found in the tutorial folder (*AnimProd_windspeeds_3D.hx*). If you want to export this animation to a video file, continue in chapter 7.4.

7.4. Video Export (MovieMaker)

With *Avizo*, you can produce *mpeg1* videos only – other video formats like *mpeg2*, *mpeg4*, *mov*, and *avi* are not supported up to now. In principle, all videos are created by the *MovieMaker* module. During the export, the animation is played step-by-step; snapshots of your current screen are taken automatically and concatenated one after the other to the *.mpg video file.

Generally, many different digital video formats are commonly used. They differ e.g. in the frame rate, the resolution, the bandwidth and the encoding algorithm. Depending on the application (web, *PowerPoint* presentations, TV, cinema, podcast, etc.), different requirements in respect to quality and file size have to be met. Although *mpeg1* is a relatively old standard with some limitations and, to be honest, not the best codec quality, it has a great advantage for us: It runs basically on every PC and can be imported into *Microsoft's PowerPoint*.

For more information you might start with these links:

- http://en.wikipedia.org/wiki/MPEG-1
- http://www.paradiso-design.net/videostandards_en.html
- http://www.b-i-t-online.de/archiv/2000-01/nach2.htm (German only)
- http://www.biff-filmfestival.de/bildformate.html (German only)

Before creating a video, it is useful to turn your screensaver off (e.g. Linux: Applications > Preferences > Screensaver) because the snapshot function of the *MovieMaker* always takes a “photo“ of the current screen, so after some time the screensaver may override your animation … and you will then have to start over from scratch with your video export.
Unfortunately, you cannot work within other windows within the same X-Display (or RGS Session) – you will have to keep the Avizo window open until the end of the video export.

7.4.1. *Videos of simple Time Animations*

A simple time animation can be exported quickly without making a detour via the *Animation Producer*. Now you are going to do this to get the result of chapter 4.11. You may preview the video you are producing here in a video player: *V_current_Embossed_Slice_2D.mpg*

- Load the project *TUT_current_Embossed_Slice_2D.hx* from the tutorial folder.
- In the Main Panel window, right-click the *NetCDF-Control*, then click ***Compute → MovieMaker → Create***.
- In the **Filename** port of the *MovieMaker Properties* define a path and filename to which the video shall be stored. You might call your video *current_Embossed_Slice.mpg*.
- Set the target screen ***resolution*** for your video. You can choose one of the following ways:
    - Click *Custom* in the **Size** port and enter a **Resolution** of e.g. X = *768*, Y = *576*. This solution has two disadvantages: First, your layout will be different in other resolutions and you cannot see the new layout in advance. Some elements might get partly or completely lost, others will be too big or too small.
    - Alternatively, you can change the viewer size by hand (drag & drop the viewer sides) and hope to catch a width of 768 and a height of 576. Remark: You will have to activate another module in your Main Panel window and then go back

to the MovieMaker Properties to see which current resolution has changed, because *Avizo* does not refresh the parameters on the fly.

- As a third option, you can change the current resolution of your viewer by typing the following command in the console:

  ```
  viewer setSize 768 576
  ```

  But this command will only work when the viewer is in top-level position. To set your viewer in top-level position, click *Edit > Preferences > Layout > Show viewer in top-level window*. But this might end up causing slightly different values in the *MovieMaker's current resolution* "because of the window decoration and the window frame" (respective section of *Avizo* online help).

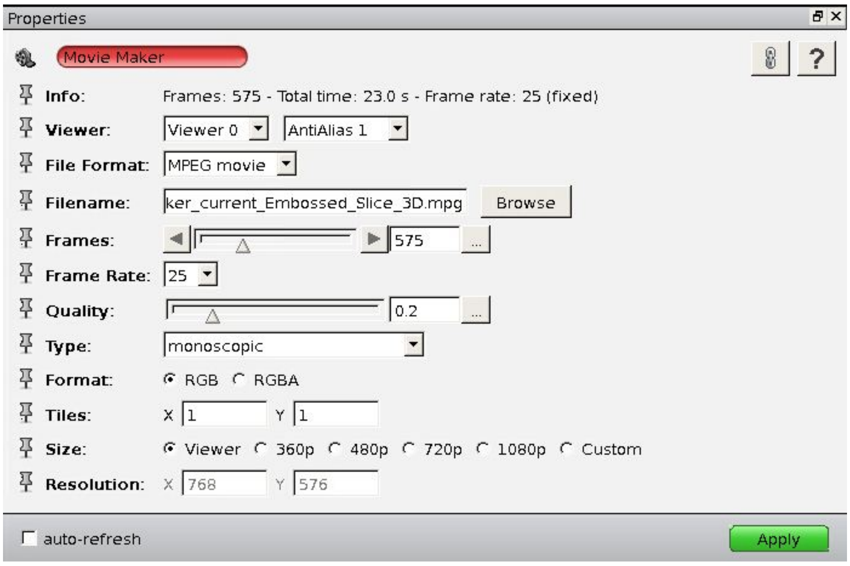
- Now you only want to animate January to September, defined by the first 92 time steps. In the *NetCDF-Control Properties*, right-click into the Time slider, click *Configure*, and enter a *Sub Max value* of *92*. When playing the animation now, it only runs until *Sep 1960*.
- Back in the *MovieMakers Properties*, set the ***Frame Rate*** to *25* frames per second. 25 frames per second are fast enough to give the impression of fluent motion.
- Set the total number of ***Frames*** to *575*

$$\frac{92 timesteps \bullet 25\frac{frames}{\sec}}{4\frac{timesteps}{\sec}} = 575\, frames$$

  so that your mpeg1 video gets a duration of 23 seconds:

$$\frac{575\, frames}{25\frac{frames}{\sec}} = \frac{92 timesteps}{4\frac{timesteps}{\sec}} = 23\sec$$

- In this animation, there are only moderate changes from frame to frame. We have used a fixed camera position and there are no drastic changes in the data from time step to time step. Therefore you can take a *compression* ***Quality*** of 0.2 or even less. The smaller the compression quality, the stronger the compression, and the smaller the size of your resulting mpeg1 file.
- Finally, in the **Viewer** port of the *MovieMaker Properties*, select *AntiAlias1* as standard. This parameter helps to reduce flickering. If your animation flickers much – e.g. in case of camera movements – you should even choose *AntiAlias3* or even *AntiAlias4* here.
- Click *Apply* to start the export.



Info: Frames: 575 - Total time: 23.0 s - Frame rate: 25 (fixed)
Viewer: Viewer 0 | AntiAlias 1
File Format: MPEG movie
Filename: ker_current_Embossed_Slice_3D.mpg
Frames: 575
Frame Rate: 25
Quality: 0.2
Type: monoscopic
Format: ● RGB ○ RGBA
Tiles: X 1 Y 1
Size: ● Viewer ○ 360p ○ 480p ○ 720p ○ 1080p ○ Custom
Resolution: X 768 Y 576
□ auto-refresh

Exercise:

A dataset with time stepping of 3 hours shall be exported to a video file for June – with a frame rate of 25 frames per second playing one day in a second. How many frames need to be chosen? What is the difference when you have 6 hourly data?
Answer

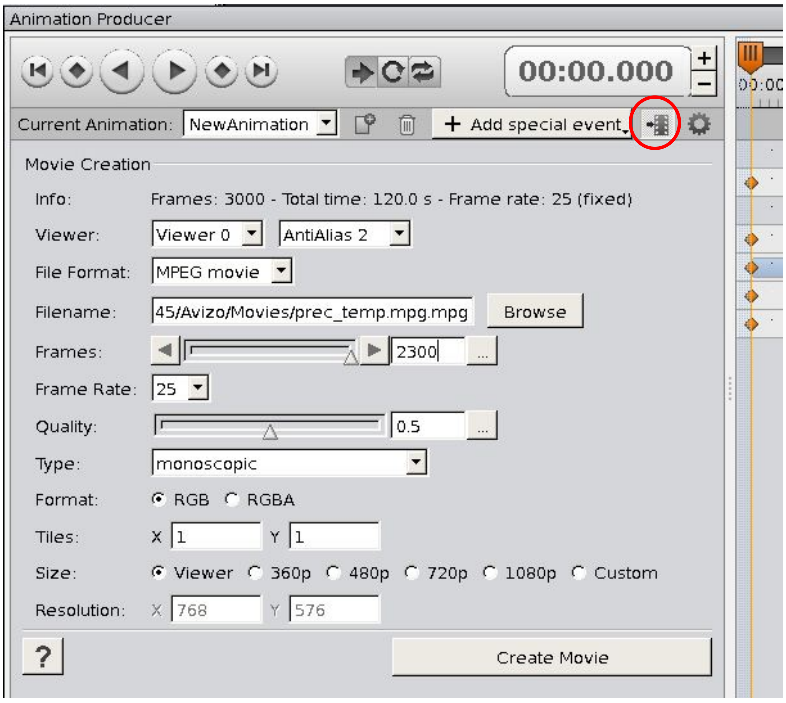
7.4.2. *Videos of Complex Animations*

This chapter shows how to export the animations that have been prepared with the *Animation Producer* in chapter 8.

Export of AnimProd_temp_prec_2D.hx
Export of AnimProd_current_Embossed_Slice_2D.hx
Export of AnimProd_windspeeds_3D.hx
Export of AnimProd_streamlines_3D.hx

Export of AnimProd_temp_prec_3D.hx

- Load project *AnimProd_prec_temp_2D.hx* (File > Open Project).
- In the *Animation Producer* module, click on the *Movie Creation* button. The panel contents switch to *Movie Creation* which represents the *MovieMaker* module.
- Enter *prec_temp.mpg* as **Filename** *in Movie Creation*.
- Change the **Resolution** to 768 x 576 (see chapter 9.1 for details).
- Choose *AntiAlias2* and a compression **Quality** of 0.5, since the visual content of this video changes quickly.
- Set the total number of **Frames** to 2300, so that your video gets a total length of 92 seconds. With 4 time steps/day, we get 4 * (30+31+31= 92 days) = 368 time steps. In the resulting video, this yields 4 time steps per second.
- Click *Create Movie* to create the video.



Animation Producer

00:00.000

Current Animation: NewAnimation

+ Add special event

Movie Creation

Info: Frames: 3000 - Total time: 120.0 s - Frame rate: 25 (fixed)

Viewer: Viewer 0 | AntiAlias 2

File Format: MPEG movie

Filename: 45/Avizo/Movies/prec_temp.mpg.mpg Browse

Frames: 2300

Frame Rate: 25

Quality: 0.5

Type: monoscopic

Format: ● RGB ○ RGBA

Tiles: X 1 Y 1

Size: ● Viewer ○ 360p ○ 480p ○ 720p ○ 1080p ○ Custom

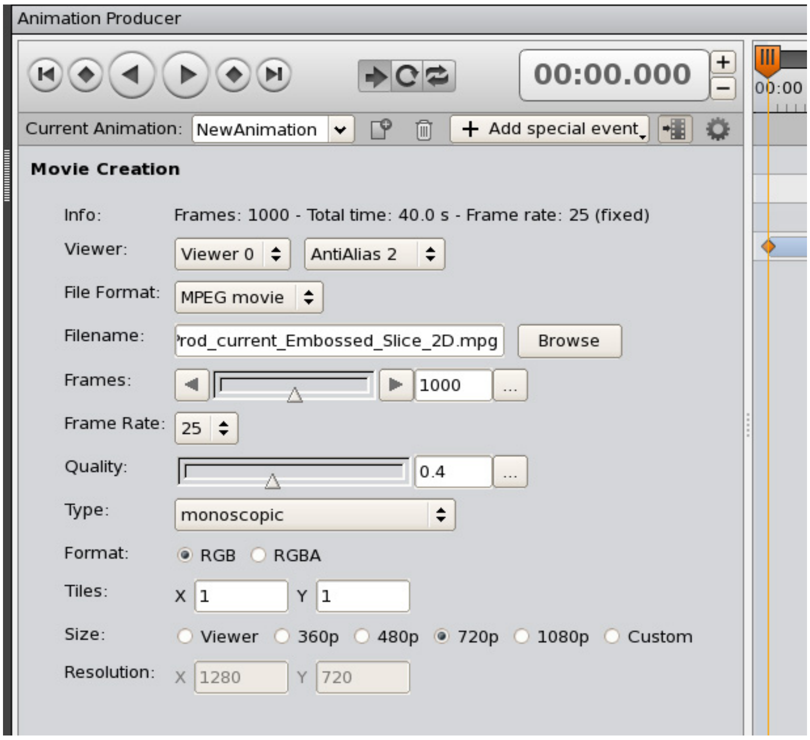
Resolution: X 768 Y 576

Export of AnimProd_current_Embossed_Slice_2D.hx

- Reload your project *AnimProd_current_Embossed_Slice_2D.hx* (File > Open Project).
- Remove the orange boundary frame in the visualization by typing the following command in the console:

  ```
  "Height Map Slice" frame off
  ```

  Hint: Even if you did this earlier in the exercise – you'll have to repeat it after re-opening the project, since Avizo unfortunately does not save this command in the script.
- In the *Animation Producer* module click on the *Movie Creation* button.
  - In *Movie Creation* enter the **Filename** *AnimProd_current_Embossed_Slice_2D.mpg*
- You might want to select the following parameters and click *Create Movie*.



Current Animation: NewAnimation

Movie Creation

Info: Frames: 1000 - Total time: 40.0 s - Frame rate: 25 (fixed)

Viewer: Viewer 0 AntiAlias 2

File Format: MPEG movie

Filename: ʼrod_current_Embossed_Slice_2D.mpg

Frames: 1000

Frame Rate: 25

Quality: 0.4

Type: monoscopic

Format: ● RGB ○ RGBA

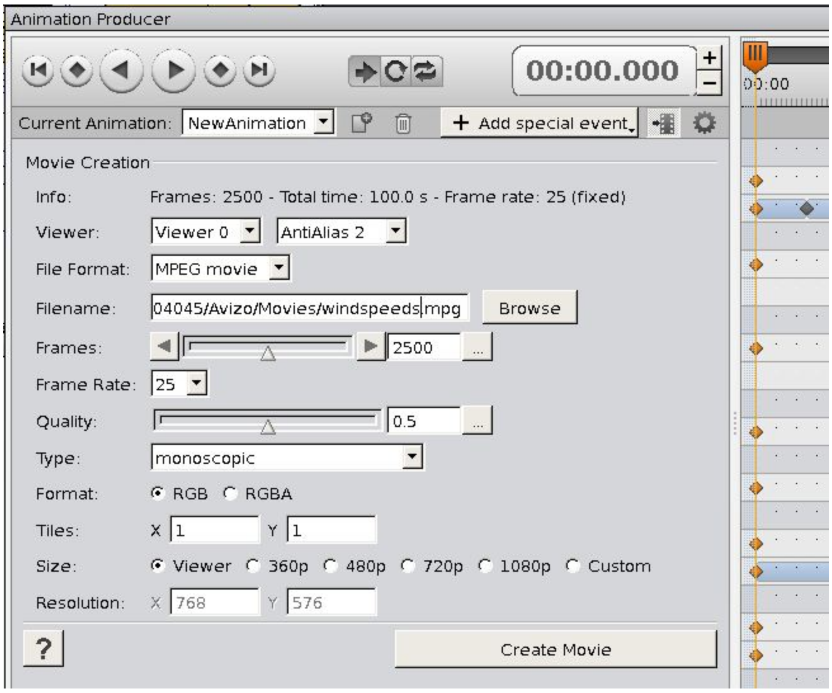
Tiles: X 1 Y 1

Size: ○ Viewer ○ 360p ○ 480p ● 720p ○ 1080p ○ Custom

Resolution: X 1280 Y 720

Export of AnimProd_windspeeds_3D.hx

- Reload your project *AnimProd_windspeeds_3D.hx* (File > Open Project).
- In the *Animation Producer* module, click on the *Movie Creation* button .
- In *Movie Creation,* enter the path and **Filename** *windspeeds.mpg*.
- Define the following parameters and start the export by pressing the *Create Movie* button.



Current Animation: NewAnimation

Movie Creation

Info: Frames: 2500 - Total time: 100.0 s - Frame rate: 25 (fixed)

Viewer: Viewer 0 AntiAlias 2

File Format: MPEG movie

Filename: 04045/Avizo/Movies/windspeeds.mpg

Frames: 2500

Frame Rate: 25

Quality: 0.5

Type: monoscopic

Format: ● RGB ○ RGBA

Tiles: X 1 Y 1

Size: ● Viewer ○ 360p ○ 480p ○ 720p ○ 1080p ○ Custom

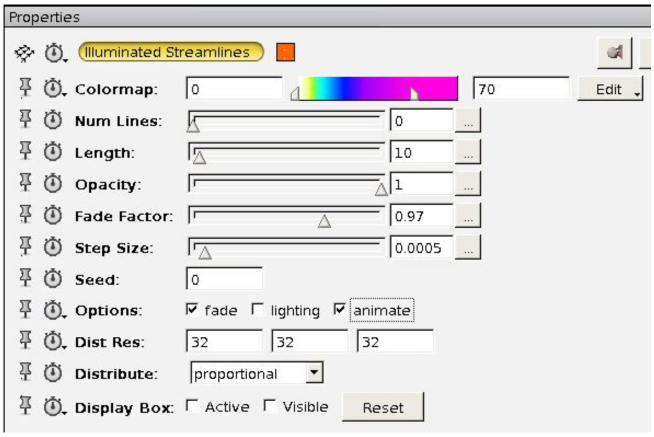
Resolution: X 768 Y 576

Export of AnimProd_streamlineSeedSurface_3D.hx

- Reload your project *AnimProd_streamlineSeedSurface_3D.hx* (File > Open Project).
- Before exporting a video from these streamlines, we should significantly reduce the speed of the animated streamlines, because the frequency in which screenshots are taken during the video export is much slower than the frequency in which *Avizo* delivers picture updates of the streamline animations. To reduce the speed of the streamlines, first type in your console:

```
“Illuminated Streamlines” setAnimationSpeed 1
```

Second, reduce the step size in the *Illuminated Streamlines* Properties to 0.0005. Additionally, you will have to prolong the streamlines in the *Seed Surface* Properties because they are extremely short now:



Properties

Illuminated Streamlines

Colormap: 0 70 Edit

Num Lines: 0

Length: 10

Opacity: 1

Fade Factor: 0.97

Step Size: 0.0005

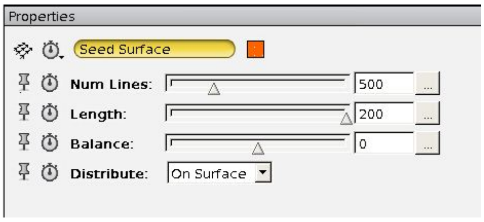
Seed: 0

Options: ☑ fade ☐ lighting ☑ animate

Dist Res: 32 32 32

Distribute: proportional

Display Box: ☐ Active ☐ Visible Reset



Properties

Seed Surface

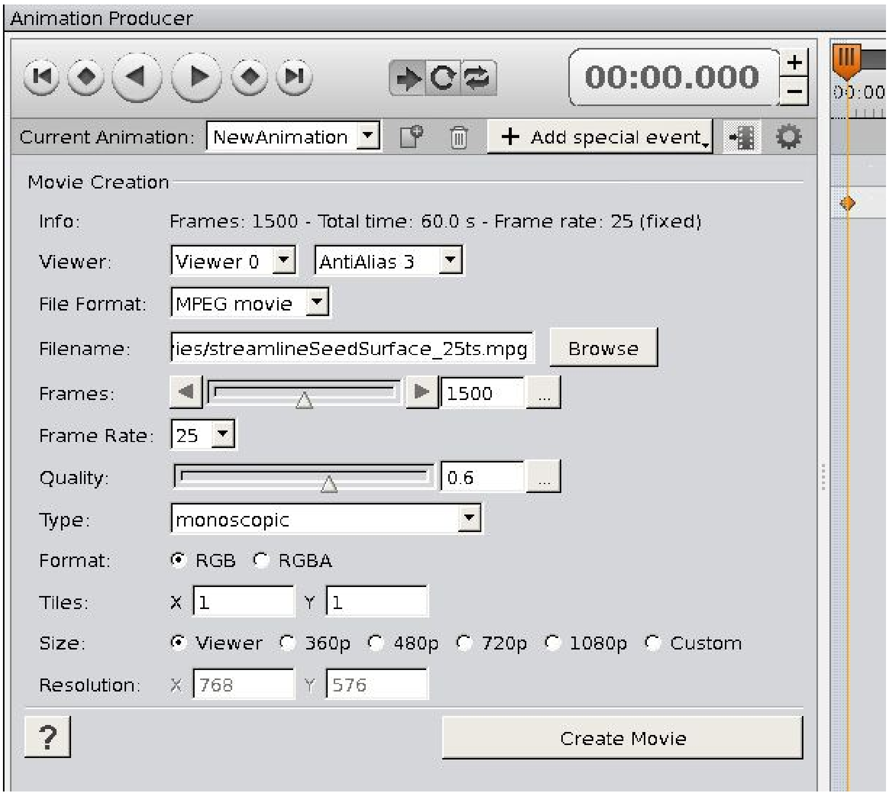
Num Lines: 500

Length: 200

Balance: 0

Distribute: On Surface

- In the *NetCDF-Control* set a sub-range of 1 to 25 only.
- In the *Animation Producer* module click on the *Movie Creation* button.
- In *Movie Creation* enter the path and **Filename** *streamlineSeedSurface_25ts.mpg*.
- In *Movie Creation* select the following parameters; your video will show animated streamlines of one time step and then move on to the next time step (the streamlines one hour later):



Current Animation: NewAnimation

Movie Creation

Info: Frames: 1500 - Total time: 60.0 s - Frame rate: 25 (fixed)

Viewer: Viewer 0 AntiAlias 3

File Format: MPEG movie

Filename: ies/streamlineSeedSurface_25ts.mpg

Frames: 1500

Frame Rate: 25

Quality: 0.6

Type: monoscopic

Format: ● RGB ○ RGBA

Tiles: X 1 Y 1

Size: ● Viewer ○ 360p ○ 480p ○ 720p ○ 1080p ○ Custom

Resolution: X 768 Y 576

7.4.3. Playing back mpeg1 videos

There are three common ways of playing and presenting your mpeg1 videos:

a) Use the **mplayer** command on the server *halo* by typing

```
mplayer <<videofile.mpg>>
```

b) Download the *.mpg file onto your local computer and play the file with a video player like **VLC Media Player**, **Windows Media Player** or others.

c) Download the *.mpg file and include it in a **PowerPoint** presentation (Insert > Video > From File).

Remark: Since older PowerPoint versions do NOT include the *.mpg file in the *.ppt file but only link to the *.mpg video file, you must make sure that the video file *.mpg is in the same folder as the *.ppt file. Remember to copy both files when you give your talk from a different computer than yours.

Annex

7.5. Answers

(1) Metadata with ncdump:

- There are 6 variables in this NetCDF file including the surface temperature *tsurf*, the precipitation *precip*, the 10m wind components *u10*, *v10*, the vertically integrated water vapour *qvi* and the mean sea level pressure *slp*.
- The file spans 1460 time steps (see under *dimensions*), in hours since 1.1.2001 at 0:00 (see variables > double time (time) > units = “hours since 2001-01-01 0:00“). Further down, under data > time, you can see that time is given in 6-hour time steps until 8754 hours after the reference date, which is 31.12.2001 18:00. Thus, the file exactly covers the entire year 2001.
- The unit of variable *qvi* (vertically integrated water vapour) is kg/m².
- Under *global attributes* > *history* you can see that this NetCDF file has been created by merging 6 individual variables (merge). Before, at least one of those files was converted into relative time format (-r), its longitudes were reset from the value range [0;360] to the value range [-180;180] (selindexbox), and the original GRIB-Format was transformed into NetCDF file format (-f nc).
- The NetCDF file *ECHAM5_OM_A1B_2001_2D.nc* does not contain a vertical dimension (neither height levels nor pressure levels); all 6 variables are 2-dimensional and time dependent (latitude, longitude, time). The NetCDF file *ECHAM5_OM_A1B_2001_3D.nc* however, includes 17 vertical levels (see *dimensions*), namely, the following pressure levels in hPa (there given in Pa): 1000, 925, 850, 775, 700, 600, 500, 400, 300, 250, 200, 150, 100, 70, 50, 30, 10.

(2) Metadata in Avizo:

- There are 1460 time steps available (see time slider in the NetCDF-Control Properties).
- Min-Max values at the beginning are 221.787 and 313.941 over all time steps 193.649 and 331.441.
- The grid is *rectilinear*.
- Processing history, e.g. cdo commands used to produce this file: see in the Parameter *history* in the NetCDF file module’s *Data Parameter Editor*.
- Temperature data are given in *Kelvin*.

(3) Without *Projection* applied, the data is visualized in grid coordinates (192 x 96 grid points). As the ECHAM data was stored on a Gaussian grid (which approximates to a cylindric equidistant grid in geographic coordinates), the Cylindric Equidistant projection yields the same display as with no projection applied. For other grids such as curvilinear grids, the difference is clearly visible.

(4) The colormap *temp2D.icol* covers the temperatures over all time steps on the earth’s surface. The colormap *temp3D.icol* shall map all time steps in 17 pressure levels of troposphere and stratosphere, so that especially the *Min* value is much lower.

Variable	***Farbmappe***	***Global Data Window***	
		Min	***Max***
tsurf (2D dataset)	temp2D.icol	193.649	331.441
t (3D dataset)	temp3D.icol	170.174	326.664

(5) The trick is to first figure out how long the video should last when it is played back. When a day should have the duration of one second, then 30 days (month of June) will last for 30 seconds. If the video has a total length of 30 seconds and is played back with 25 frames per second (= FrameRate): 25 * 30 = **750 Frames** have to be stored in the *.mpg file.

Avizo computes internally how to distribute the time steps to the frames exactly. For a file with 3-hourly data, 8 time steps have to be distributed over 25 frames. So, the data values always change after 3 (or 4) frames. If your dataset consists of 6-hourly data, 4 time steps have to be distributed over 25 frames, so the values always change after 6 (or 7) frames.

7.6. *Examples of data pre-processing*

Open a new terminal on the server *halo* and change into the tutorial subdirectory containing the pre-processing example files:
/work/kv0653/Tutorial_AvizoGreen/preprocessing

Example 1: Time unit, longitude range

According to chapter 2.2 type:
`ncdump -c ex1_relTime_lonRange.nc`

This provides the following output (shortened at the end):

```
netcdf ex1_relTime_lonRange {
dimensions:
        lon = 192 ;
        lat = 96 ;
        lev = 17 ;
        time = UNLIMITED ; // (124 currently)
variables:
        double lon(lon) ;
                lon:long_name = "longitude" ;
                lon:units = "degrees_east" ;
                lon:standard_name = "longitude" ;
        double lat(lat) ;
                lat:long_name = "latitude" ;
                lat:units = "degrees_north" ;
                lat:standard_name = "latitude" ;
        double lev(lev) ;
                lev:long_name = "pressure" ;
                lev:units = "Pa" ;
        double time(time) ;
                time:units = "day as %Y%m%d.%f" ;
        float t(time, lev, lat, lon) ;
                t:long_name = "temperature" ;
                t:units = "K" ;
                t:code = 130 ;
                t:table = 128 ;
                t:grid_type = "gaussian" ;

// global attributes:
                :CDI = "Climate Data Interface version 1.2.1" ;
                :Conventions = "CF-1.0" ;
                :history = "Wed Nov 26 09:57:42 2008: cdo seldate,2001-01-01,2001-01-
31 EH5_OM_A1B_1_STP.nc EH5_OM_A1B_1_STP_Jan.nc\n",
                    "Wed Nov 26 09:52:47 2008: cdo -f nc copy EH5_OM_A1B_1_STP.grb
EH5_OM_A1B_1_STP.nc" ;
                :source = "ECHAM5.2" ;
                :institution = "Max-Planck-Institute for Meteorology" ;
                :CDO = "Climate Data Operators version 1.2.1
(http://www.mpimet.mpg.de/cdo)" ;
data:

 lon = 0, 1.875, 3.75, 5.625, 7.5, 9.375, 11.25, 13.125, 15, 16.875, 18.75,
    20.625, 22.5, 24.375, 26.25, 28.125, 30, 31.875, 33.75, 35.625, 37.5,
    39.375, 41.25, 43.125, 45, 46.875, 48.75, 50.625, 52.5, 54.375, 56.25,
    58.125, 60, 61.875, 63.75, 65.625, 67.5, 69.375, 71.25, 73.125, 75,
    76.875, 78.75, 80.625, 82.5, 84.375, 86.25, 88.125, 90, 91.875, 93.75,
    95.625, 97.5, 99.375, 101.25, 103.125, 105, 106.875, 108.75, 110.625,
    112.5, 114.375, 116.25, 118.125, 120, 121.875, 123.75, 125.625, 127.5,
    129.375, 131.25, 133.125, 135, 136.875, 138.75, 140.625, 142.5, 44.375,
    146.25, 148.125, 150, 151.875, 153.75, 155.625, 157.5, 159.375, 161.25,
    163.125, 165, 166.875, 168.75, 170.625, 172.5, 174.375, 176.25, 78.125,
    180, 181.875, 183.75, 185.625, 187.5, 189.375, 191.25, 193.125, 195,
    196.875, 198.75, 200.625, 202.5, 204.375, 206.25, 208.125, 210, 11.875,
    213.75, 215.625, 217.5, 219.375, 221.25, 223.125, 225, 226.875, 228.75,
    230.625, 232.5, 234.375, 236.25, 238.125, 240, 241.875, 243.75, 45.625,
    247.5, 249.375, 251.25, 253.125, 255, 256.875, 258.75, 260.625, 262.5,
    264.375, 266.25, 268.125, 270, 271.875, 273.75, 275.625, 277.5, 79.375,
    281.25, 283.125, 285, 286.875, 288.75, 290.625, 292.5, 294.375, 296.25,
    298.125, 300, 301.875, 303.75, 305.625, 307.5, 309.375, 311.25, 13.125,
    315, 316.875, 318.75, 320.625, 322.5, 324.375, 326.25, 328.125, 330,
    331.875, 333.75, 335.625, 337.5, 339.375, 341.25, 343.125, 345, 46.875,
    348.75, 350.625, 352.5, 354.375, 356.25, 358.125 ;

 lat = …;

 lev = … ;

 time = … ;
}
```

According to chapter 2.3 we'll have to do two conversions:

a) Change the time format of the NetCDF file from absolute to relative time.
b) Change the value range of the longitudes from [0;360] to [-180;+180].

You can use cdo operator piping to do so:

```
cdo -r copy -selindexbox,97,96,1,96 ex1_relTime_lonRange.nc
avizo1.nc
```

Example 2: Hybrid levels

For the vertical axis of atmospheric models, *Avizo* supports height and pressure levels. If the vertical axis of your data is described by *hybrid* model levels (terrain following vertical coordinates at the bottom, pressure levels at the top, interpolated in between), the data must be interpolated onto HEIGHT [m] or PRESSURE [Pa] levels before reading it with *Avizo*. The number and values of those height or pressure levels can be defined explicitly (plevels, hlevels). During this process you may receive missing values (Grid cells with no data), e.g. in areas with mountains in case of atmospheric data.

```
cdo ml2pl,plevels <inputfile> <outputfile>
cdo ml2hl,hlevels <inputfile> <outputfile>
```

Example 3: Ocean levels

Our third example file includes concentrations of phytoplankton, zooplankton and nitrate in 24 ocean levels which have positive depths:

```
ncdump -c ex3_oceanLevels.nc
```

```
netcdf ex3_oceanLevels {
dimensions:
        longitude = 88 ;
        latitude = 82 ;
        levels = 24 ;
        time = UNLIMITED ; // (1 currently)
variables:
        double longitude(longitude) ;
                longitude:long_name = "Longitude" ;
                longitude:units = "degrees_east" ;
                longitude:standard_name = "longitude" ;
        double latitude(latitude) ;
                latitude:long_name = "Latitude" ;
                latitude:units = "degrees_north" ;
                latitude:standard_name = "latitude" ;
        double levels(levels) ;
                levels:long_name = "Level" ;
                levels:units = "meters" ;
                levels:positive = "down" ;
        double time(time) ;
                time:units = "days since 2004-01-01 00:00" ;
        float phc(time, levels, latitude, longitude) ;
                phc:long_name = "Phytoplankton" ;
                phc:_FillValue = -9999.f ;
        float zoc(time, levels, latitude, longitude) ;
                zoc:long_name = "Zooplankton" ;
                zoc:_FillValue = -9999.f ;
        float n3n(time, levels, latitude, longitude) ;
                n3n:long_name = "Nitrate" ;
                n3n:_FillValue = -9999.f ;

// global attributes:

...

data:

 longitude = ... ;

 latitude = ... ;

 levels = 10, 15, 20, 25, 30, 35, 40, 45, 50, 60, 75, 100, 150, 200, 300,
    400, 500, 600, 700, 800, 1000, 2000, 3000, 4000 ;

 time = 0 ;
}
```

According to the CD1.0 metadata convention [http://cf-pcmdi.llnl.gov/documents/cf-conventions/1.0/cf-conventions.pdf – page 14], the levels can have a positive sign but will be interpreted as depths if the attribute ‘`positive = "down"`’ is added, as we did in our example file `ex3_oceanLevels.nc` above.

```
axis_name:units = "meters" ;
axis_name:positive = "down" ;
```

Unfortunately, Avizo does not yet consider this in the current version, although it is CF1.0 convention. Instead, we have to make sure that the ocean levels have negative signs in order to visualize them below sea-surface.

Open an editor (“gedit” for instance) and define a simple text file (here called zaxis.txt) with a description of the new z-axis with negative height levels:

```
cat zaxis.txt
zaxistype = height
size      = 24
levels    = -10 -15 -20 -25 -30 -35 -40 -45 -50 -60 -75 -100
-150 -200 -300 -400 -500 -600 -700 -800 -1000 -2000 -3000
-4000
```

Then use the setzaxis command to assign this new z-axis to your ocean NetCDF file:

```
cdo setzaxis,zaxis.txt ex3_oceanLevels.nc avizo3.nc
```

Example 4a/b: Time interpolation

You might look into the NetCDF files of this example with `ncdump -c` to view the time variable data, but it is not very convenient to read. Time data is much easier to read with `cdo sinfo` because all time steps are printed in “`yyyy-MM-dd hh:mm`” format, then:

```
cdo sinfo ex4a_timeInt.nc
```

The last section of the output provides information on the time steps:

```
Time axis :  12 steps
  RefTime = 1900-01-01 00:00  Units = days  Calendar = STANDARD
YYYY-MM-DD hh:mm   YYYY-MM-DD hh:mm   YYYY-MM-DD hh:mm   YYYY-MM-DD hh:mm
2007-01-16 12:00   2007-02-15 00:00   2007-03-16 12:00   2007-04-16 00:00
2007-05-16 12:00   2007-06-16 00:00   2007-07-16 12:00   2007-08-16 12:00
2007-09-16 00:00   2007-10-16 12:00   2007-11-16 00:00   2007-12-16 12:00
```

The dataset obviously has 12 time steps with one time step for the mid of each month in the year 2007. To smooth the animation we can interpolate on 10 time steps per month with the following command:

```
cdo intntime,10 ex4a_timeInt.nc avizo4a.nc
```

The original times are preserved and 9 time steps are added between one time step and the next. So, the output NetCDF file has 12 + 11*9 = 111 time steps. Note that after the last original time step (mid of December), no further time steps are added:

```
cdo sinfo avizo4a.nc
```

```
Time axis :  111 steps
  RefTime = 1900-01-01 00:00  Units = days  Calendar = STANDARD
YYYY-MM-DD hh:mm   YYYY-MM-DD hh:mm   YYYY-MM-DD hh:mm   YYYY-MM-DD hh:mm
2007-01-16 12:00   2007-01-19 10:48   2007-01-22 09:36   2007-01-25 08:24
2007-01-28 07:12   2007-01-31 06:00   2007-02-03 04:48   2007-02-06 03:36
2007-02-09 02:24   2007-02-12 01:12   2007-02-15 00:00   2007-02-17 22:48
2007-02-20 21:36   2007-02-23 20:24   2007-02-26 19:12   2007-03-01 18:00
2007-03-04 16:48   2007-03-07 15:36   2007-03-10 14:24   2007-03-13 13:12
2007-03-16 12:00   2007-03-19 13:12   2007-03-22 14:24   2007-03-25 15:36
2007-03-28 16:48   2007-03-31 18:00   2007-04-03 19:12   2007-04-06 20:24
2007-04-09 21:36   2007-04-12 22:48   2007-04-16 00:00   2007-04-19 01:12
2007-04-22 02:24   2007-04-25 03:36   2007-04-28 04:48   2007-05-01 06:00
2007-05-04 07:12   2007-05-07 08:24   2007-05-10 09:36   2007-05-13 10:48
2007-05-16 12:00   2007-05-19 13:12   2007-05-22 14:24   2007-05-25 15:36
2007-05-28 16:48   2007-05-31 18:00   2007-06-03 19:12   2007-06-06 20:24
2007-06-09 21:36   2007-06-12 22:48   2007-06-16 00:00   2007-06-19 01:12
2007-06-22 02:24   2007-06-25 03:36   2007-06-28 04:48   2007-07-01 06:00
2007-07-04 07:12   2007-07-07 08:24   2007-07-10 09:36   2007-07-13 10:48
2007-07-16 12:00   2007-07-19 14:24   2007-07-22 16:48   2007-07-25 19:12
2007-07-28 21:36   2007-08-01 00:00   2007-08-04 02:24   2007-08-07 04:48
2007-08-10 07:12   2007-08-13 09:36   2007-08-16 12:00   2007-08-19 13:12
2007-08-22 14:24   2007-08-25 15:36   2007-08-28 16:48   2007-08-31 18:00
2007-09-03 19:12   2007-09-06 20:24   2007-09-09 21:36   2007-09-12 22:48
2007-09-16 00:00   2007-09-19 01:12   2007-09-22 02:24   2007-09-25 03:36
2007-09-28 04:48   2007-10-01 06:00   2007-10-04 07:12   2007-10-07 08:24
2007-10-10 09:36   2007-10-13 10:48   2007-10-16 12:00   2007-10-19 13:12
2007-10-22 14:24   2007-10-25 15:36   2007-10-28 16:48   2007-10-31 18:00
2007-11-03 19:12   2007-11-06 20:24   2007-11-09 21:36   2007-11-12 22:48
2007-11-16 00:00   2007-11-19 01:12   2007-11-22 02:24   2007-11-25 03:36
2007-11-28 04:48   2007-12-01 06:00   2007-12-04 07:12   2007-12-07 08:24
2007-12-10 09:36   2007-12-13 10:48   2007-12-16 12:00
```

Remark: Technically, we could have used the operator `inttime` (instead of `intntime`) as well to produce e.g. a daily dataset (`cdo intntime,2007-01-16,12:00,1day`). But then we do not preserve the original time steps of all months.

Let's have a look at a second example. This time we have a 6-hourly data set of horizontal wind components u and v which we would like to interpolate to a 1-hourly dataset.

```
cdo sinfo ex4b_timeInt.nc
```

```
   Time axis :  124 steps
     RefTime = 2001-01-01 00:00  Units = hours  Calendar = STANDARD
   YYYY-MM-DD hh:mm   YYYY-MM-DD hh:mm   YYYY-MM-DD hh:mm   YYYY-MM-DD hh:mm
   2001-01-01 00:00   2001-01-01 06:00   2001-01-01 12:00   2001-01-01 18:00
   2001-01-02 00:00   2001-01-02 06:00   2001-01-02 12:00   2001-01-02 18:00
   2001-01-03 00:00   2001-01-03 06:00   2001-01-03 12:00   2001-01-03 18:00
   2001-01-04 00:00   2001-01-04 06:00   2001-01-04 12:00   2001-01-04 18:00
   2001-01-05 00:00   2001-01-05 06:00   2001-01-05 12:00   2001-01-05 18:00
   2001-01-06 00:00   2001-01-06 06:00   2001-01-06 12:00   2001-01-06 18:00
   2001-01-07 00:00   2001-01-07 06:00   2001-01-07 12:00   2001-01-07 18:00
   2001-01-08 00:00   2001-01-08 06:00   2001-01-08 12:00   2001-01-08 18:00
   2001-01-09 00:00   2001-01-09 06:00   2001-01-09 12:00   2001-01-09 18:00
   2001-01-10 00:00   2001-01-10 06:00   2001-01-10 12:00   2001-01-10 18:00
   2001-01-11 00:00   2001-01-11 06:00   2001-01-11 12:00   2001-01-11 18:00
   2001-01-12 00:00   2001-01-12 06:00   2001-01-12 12:00   2001-01-12 18:00
   2001-01-13 00:00   2001-01-13 06:00   2001-01-13 12:00   2001-01-13 18:00
   2001-01-14 00:00   2001-01-14 06:00   2001-01-14 12:00   2001-01-14 18:00
   2001-01-15 00:00   2001-01-15 06:00   2001-01-15 12:00   2001-01-15 18:00
   2001-01-16 00:00   2001-01-16 06:00   2001-01-16 12:00   2001-01-16 18:00
   2001-01-17 00:00   2001-01-17 06:00   2001-01-17 12:00   2001-01-17 18:00
   2001-01-18 00:00   2001-01-18 06:00   2001-01-18 12:00   2001-01-18 18:00
   2001-01-19 00:00   2001-01-19 06:00   2001-01-19 12:00   2001-01-19 18:00
   2001-01-20 00:00   2001-01-20 06:00   2001-01-20 12:00   2001-01-20 18:00
   2001-01-21 00:00   2001-01-21 06:00   2001-01-21 12:00   2001-01-21 18:00
   2001-01-22 00:00   2001-01-22 06:00   2001-01-22 12:00   2001-01-22 18:00
   2001-01-23 00:00   2001-01-23 06:00   2001-01-23 12:00   2001-01-23 18:00
   2001-01-24 00:00   2001-01-24 06:00   2001-01-24 12:00   2001-01-24 18:00
   2001-01-25 00:00   2001-01-25 06:00   2001-01-25 12:00   2001-01-25 18:00
   2001-01-26 00:00   2001-01-26 06:00   2001-01-26 12:00   2001-01-26 18:00
   2001-01-27 00:00   2001-01-27 06:00   2001-01-27 12:00   2001-01-27 18:00
   2001-01-28 00:00   2001-01-28 06:00   2001-01-28 12:00   2001-01-28 18:00
   2001-01-29 00:00   2001-01-29 06:00   2001-01-29 12:00   2001-01-29 18:00
   2001-01-30 00:00   2001-01-30 06:00   2001-01-30 12:00   2001-01-30 18:00
   2001-01-31 00:00   2001-01-31 06:00   2001-01-31 12:00   2001-01-31 18:00
```

We can do the time interpolation in one of the following two ways (resulting in the same output file):

```
cdo intntime,6 ex4b_timeInt.nc avizo4b.nc
cdo inttime,2001-01-01,00:00,1hour ex4b_timeInt.nc avizo4b.nc
```

Example 5: Horizontal grid interpolation onto the scalar grid

The NetCDF file example6a.nc has a rectilinear grid with 1°x1° spatial resolution. There are two longitudinal and two latitudinal dimensions defined (and two depth dimensions which we will work on later – see example 6 for vertical grid interpolation). The dimensions LONGITUDE_T and LATITUDE_T define the grid center points – used for the scalar variable SALT for instance. The dimension LONGITUDE_U is used for the u wind component defined at the left grid wall. The dimension LATITUDE_V is used for the v wind component defined at the front grid wall respectively.

```
ncdump -c ex5_vectorGrids.nc
```

```
netcdf ex5_vectorGrids {
dimensions:
        LONGITUDE_T = 360 ;
        LATITUDE_V = 160 ;
        LATITUDE_T = 160 ;
        LONGITUDE_U = 360 ;
        DEPTH_T = 23 ;
        DEPTH_W = 24 ;
variables:
        double LONGITUDE_T(LONGITUDE_T) ;
                LONGITUDE_T:long_name = "longitude" ;
                LONGITUDE_T:units = "degrees_east" ;
                LONGITUDE_T:standard_name = "longitude" ;
        double LATITUDE_V(LATITUDE_V) ;
                LATITUDE_V:long_name = "latitude" ;
                LATITUDE_V:units = "degrees_north" ;
                LATITUDE_V:standard_name = "latitude" ;
        double LATITUDE_T(LATITUDE_T) ;
                LATITUDE_T:long_name = "latitude" ;
                LATITUDE_T:units = "degrees_north" ;
                LATITUDE_T:standard_name = "latitude" ;
        double LONGITUDE_U(LONGITUDE_U) ;
                LONGITUDE_U:long_name = "longitude" ;
                LONGITUDE_U:units = "degrees_east" ;
                LONGITUDE_U:standard_name = "longitude" ;
        double DEPTH_T(DEPTH_T) ;
                DEPTH_T:units = "meters" ;
        double DEPTH_W(DEPTH_W) ;
                DEPTH_W:units = "meters" ;
        float V(DEPTH_T, LATITUDE_V, LONGITUDE_T) ;
                V:long_name = "V VELOCITY" ;
                V:units = "m/sec" ;
                V:long_name_mod = "T=01-JAN-1952 05:45:31-DEC-2001 17:45@AVE" ;
                V:history = "From iter23" ;
                V:_FillValue = -1.e+23f ;
        float SALT(DEPTH_T, LATITUDE_T, LONGITUDE_T) ;
                SALT:long_name = "SALINITY" ;
                SALT:units = "PSU" ;
                SALT:long_name_mod = "T=01-JAN-1952 05:45:31-DEC-2001 17:45@AVE" ;
                SALT:history = "From iter23" ;
                SALT:_FillValue = -1.e+23f ;
        float U(DEPTH_T, LATITUDE_T, LONGITUDE_U) ;
                U:long_name = "U VELOCITY" ;
                U:units = "m/sec" ;
                U:long_name_mod = "T=01-JAN-1952 05:45:31-DEC-2001 17:45@AVE" ;
                U:history = "From iter23" ;
                U:_FillValue = -1.e+23f ;
        float W(DEPTH_W, LATITUDE_T, LONGITUDE_T) ;
                W:long_name = "W VELOCITY" ;
                W:units = "m/sec" ;
                W:long_name_mod = "T=01-JAN-1952 05:45:31-DEC-2001 17:45@AVE" ;
                W:history = "From iter23" ;
                W:_FillValue = -1.e+23f ;

// global attributes:
...

data:
```

```
LONGITUDE_T = 0.5, 1.5, 2.5, 3.5, 4.5, 5.5, 6.5, 7.5, 8.5, 9.5, 10.5, 11.5,
   12.5, 13.5, 14.5, 15.5, 16.5, 17.5, 18.5, 19.5, 20.5, 21.5, 22.5, 23.5,
   24.5, 25.5, 26.5, 27.5, 28.5, 29.5, 30.5, 31.5, 32.5, 33.5, 34.5, 35.5,
   36.5, 37.5, 38.5, 39.5, 40.5, 41.5, 42.5, 43.5, 44.5, 45.5, 46.5, 47.5,
   48.5, 49.5, 50.5, 51.5, 52.5, 53.5, 54.5, 55.5, 56.5, 57.5, 58.5, 59.5,
   60.5, 61.5, 62.5, 63.5, 64.5, 65.5, 66.5, 67.5, 68.5, 69.5, 70.5, 71.5,
   72.5, 73.5, 74.5, 75.5, 76.5, 77.5, 78.5, 79.5, 80.5, 81.5, 82.5, 83.5,
   84.5, 85.5, 86.5, 87.5, 88.5, 89.5, 90.5, 91.5, 92.5, 93.5, 94.5, 95.5,
   96.5, 97.5, 98.5, 99.5, 100.5, 101.5, 102.5, 103.5, 104.5, 105.5, 106.5,
   107.5, 108.5, 109.5, 110.5, 111.5, 112.5, 113.5, 114.5, 115.5, 116.5,
   117.5, 118.5, 119.5, 120.5, 121.5, 122.5, 123.5, 124.5, 125.5, 126.5,
   127.5, 128.5, 129.5, 130.5, 131.5, 132.5, 133.5, 134.5, 135.5, 136.5,
   137.5, 138.5, 139.5, 140.5, 141.5, 142.5, 143.5, 144.5, 145.5, 146.5,
   147.5, 148.5, 149.5, 150.5, 151.5, 152.5, 153.5, 154.5, 155.5, 156.5,
   157.5, 158.5, 159.5, 160.5, 161.5, 162.5, 163.5, 164.5, 165.5, 166.5,
   167.5, 168.5, 169.5, 170.5, 171.5, 172.5, 173.5, 174.5, 175.5, 176.5,
   177.5, 178.5, 179.5, 180.5, 181.5, 182.5, 183.5, 184.5, 185.5, 186.5,
   187.5, 188.5, 189.5, 190.5, 191.5, 192.5, 193.5, 194.5, 195.5, 196.5,
   197.5, 198.5, 199.5, 200.5, 201.5, 202.5, 203.5, 204.5, 205.5, 206.5,
   207.5, 208.5, 209.5, 210.5, 211.5, 212.5, 213.5, 214.5, 215.5, 216.5,
   217.5, 218.5, 219.5, 220.5, 221.5, 222.5, 223.5, 224.5, 225.5, 226.5,
   227.5, 228.5, 229.5, 230.5, 231.5, 232.5, 233.5, 234.5, 235.5, 236.5,
   237.5, 238.5, 239.5, 240.5, 241.5, 242.5, 243.5, 244.5, 245.5, 246.5,
   247.5, 248.5, 249.5, 250.5, 251.5, 252.5, 253.5, 254.5, 255.5, 256.5,
   257.5, 258.5, 259.5, 260.5, 261.5, 262.5, 263.5, 264.5, 265.5, 266.5,
   267.5, 268.5, 269.5, 270.5, 271.5, 272.5, 273.5, 274.5, 275.5, 276.5,
   277.5, 278.5, 279.5, 280.5, 281.5, 282.5, 283.5, 284.5, 285.5, 286.5,
   287.5, 288.5, 289.5, 290.5, 291.5, 292.5, 293.5, 294.5, 295.5, 296.5,
   297.5, 298.5, 299.5, 300.5, 301.5, 302.5, 303.5, 304.5, 305.5, 306.5,
   307.5, 308.5, 309.5, 310.5, 311.5, 312.5, 313.5, 314.5, 315.5, 316.5,
   317.5, 318.5, 319.5, 320.5, 321.5, 322.5, 323.5, 324.5, 325.5, 326.5,
   327.5, 328.5, 329.5, 330.5, 331.5, 332.5, 333.5, 334.5, 335.5, 336.5,
   337.5, 338.5, 339.5, 340.5, 341.5, 342.5, 343.5, 344.5, 345.5, 346.5,
   347.5, 348.5, 349.5, 350.5, 351.5, 352.5, 353.5, 354.5, 355.5, 356.5,
   357.5, 358.5, 359.5 ;

LATITUDE_V = -80, -79, -78, -77, -76, -75, -74, -73, -72, -71, -70, -69,
   -68, -67, -66, -65, -64, -63, -62, -61, -60, -59, -58, -57, -56, -55,
   -54, -53, -52, -51, -50, -49, -48, -47, -46, -45, -44, -43, -42, -41,
   -40, -39, -38, -37, -36, -35, -34, -33, -32, -31, -30, -29, -28, -27,
   -26, -25, -24, -23, -22, -21, -20, -19, -18, -17, -16, -15, -14, -13,
   -12, -11, -10, -9, -8, -7, -6, -5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5, 6,
   7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25,
   26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43,
   44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61,
   62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79 ;

LATITUDE_T = -79.5, -78.5, -77.5, -76.5, -75.5, -74.5, -73.5, -72.5, -71.5,
   -70.5, -69.5, -68.5, -67.5, -66.5, -65.5, -64.5, -63.5, -62.5, -61.5,
   -60.5, -59.5, -58.5, -57.5, -56.5, -55.5, -54.5, -53.5, -52.5, -51.5,
   -50.5, -49.5, -48.5, -47.5, -46.5, -45.5, -44.5, -43.5, -42.5, -41.5,
   -40.5, -39.5, -38.5, -37.5, -36.5, -35.5, -34.5, -33.5, -32.5, -31.5,
   -30.5, -29.5, -28.5, -27.5, -26.5, -25.5, -24.5, -23.5, -22.5, -21.5,
   -20.5, -19.5, -18.5, -17.5, -16.5, -15.5, -14.5, -13.5, -12.5, -11.5,
   -10.5, -9.5, -8.5, -7.5, -6.5, -5.5, -4.5, -3.5, -2.5, -1.5, -0.5, 0.5,
   1.5, 2.5, 3.5, 4.5, 5.5, 6.5, 7.5, 8.5, 9.5, 10.5, 11.5, 12.5, 13.5,
   14.5, 15.5, 16.5, 17.5, 18.5, 19.5, 20.5, 21.5, 22.5, 23.5, 24.5, 25.5,
   26.5, 27.5, 28.5, 29.5, 30.5, 31.5, 32.5, 33.5, 34.5, 35.5, 36.5, 37.5,
   38.5, 39.5, 40.5, 41.5, 42.5, 43.5, 44.5, 45.5, 46.5, 47.5, 48.5, 49.5,
   50.5, 51.5, 52.5, 53.5, 54.5, 55.5, 56.5, 57.5, 58.5, 59.5, 60.5, 61.5,
   62.5, 63.5, 64.5, 65.5, 66.5, 67.5, 68.5, 69.5, 70.5, 71.5, 72.5, 73.5,
   74.5, 75.5, 76.5, 77.5, 78.5, 79.5 ;

LONGITUDE_U = 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17,
   18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35,
   36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53,
   54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71,
   72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89,
   90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105,
   106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119,
   120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 130, 131, 132, 133,
   134, 135, 136, 137, 138, 139, 140, 141, 142, 143, 144, 145, 146, 147,
   148, 149, 150, 151, 152, 153, 154, 155, 156, 157, 158, 159, 160, 161,
   162, 163, 164, 165, 166, 167, 168, 169, 170, 171, 172, 173, 174, 175,
   176, 177, 178, 179, 180, 181, 182, 183, 184, 185, 186, 187, 188, 189,
   190, 191, 192, 193, 194, 195, 196, 197, 198, 199, 200, 201, 202, 203,
   204, 205, 206, 207, 208, 209, 210, 211, 212, 213, 214, 215, 216, 217,
   218, 219, 220, 221, 222, 223, 224, 225, 226, 227, 228, 229, 230, 231,
   232, 233, 234, 235, 236, 237, 238, 239, 240, 241, 242, 243, 244, 245,
    246, 247, 248, 249, 250, 251, 252, 253, 254, 255, 256, 257, 258, 259,
    260, 261, 262, 263, 264, 265, 266, 267, 268, 269, 270, 271, 272, 273,
    274, 275, 276, 277, 278, 279, 280, 281, 282, 283, 284, 285, 286, 287,
    288, 289, 290, 291, 292, 293, 294, 295, 296, 297, 298, 299, 300, 301,
    302, 303, 304, 305, 306, 307, 308, 309, 310, 311, 312, 313, 314, 315,
    316, 317, 318, 319, 320, 321, 322, 323, 324, 325, 326, 327, 328, 329,
    330, 331, 332, 333, 334, 335, 336, 337, 338, 339, 340, 341, 342, 343,
    344, 345, 346, 347, 348, 349, 350, 351, 352, 353, 354, 355, 356, 357,
    358, 359 ;

 DEPTH_T = 5, 15, 27.5, 45, 65, 87.5, 117.5, 160, 222.5, 310, 435, 610,
    847.5, 1160, 1542.5, 1975, 2450, 2950, 3450, 3950, 4450, 4950, 5450 ;

 DEPTH_W = 0, 10, 20, 35, 55, 75, 100, 135, 185, 260, 360, 510, 710, 985,
    1335, 1750, 2200, 2700, 3200, 3700, 4200, 4700, 5200, 5700 ;
}
```

To interpolate U and V onto the scalar grid you might first extract the variable SALT to use this separate NetCDF file as a grid description file:

```
cdo selname,SALT ex5_vectorGrids.nc SALT.nc
cdo remapbil,SALT.nc ex5_vectorGrids.nc scalarGrid_UV.nc
```

`ncdump -h scalarGrid_UV.nc` shows the result

```
netcdf scalarGrid_UV {
dimensions:
        LONGITUDE_T = 360 ;
        LATITUDE_T = 160 ;
        DEPTH_T = 23 ;
        DEPTH_W = 24 ;
variables:
        double LONGITUDE_T(LONGITUDE_T) ;
                LONGITUDE_T:long_name = "longitude" ;
                LONGITUDE_T:units = "degrees_east" ;
                LONGITUDE_T:standard_name = "longitude" ;
        double LATITUDE_T(LATITUDE_T) ;
                LATITUDE_T:long_name = "latitude" ;
                LATITUDE_T:units = "degrees_north" ;
                LATITUDE_T:standard_name = "latitude" ;
        double DEPTH_T(DEPTH_T) ;
                DEPTH_T:units = "meters" ;
        double DEPTH_W(DEPTH_W) ;
                DEPTH_W:units = "meters" ;
        float V(DEPTH_T, LATITUDE_T, LONGITUDE_T) ;
                V:long_name = "V VELOCITY" ;
                V:units = "m/sec" ;
                V:long_name_mod = "T=01-JAN-1952 05:45:31-DEC-2001 17:45@AVE" ;
                V:history = "From iter23" ;
                V:_FillValue = -1.e+23f ;
        float SALT(DEPTH_T, LATITUDE_T, LONGITUDE_T) ;
                SALT:long_name = "SALINITY" ;
                SALT:units = "PSU" ;
                SALT:long_name_mod = "T=01-JAN-1952 05:45:31-DEC-2001 17:45@AVE" ;
                SALT:history = "From iter23" ;
                SALT:_FillValue = -1.e+23f ;
        float U(DEPTH_T, LATITUDE_T, LONGITUDE_T) ;
                U:long_name = "U VELOCITY" ;
                U:units = "m/sec" ;
                U:long_name_mod = "T=01-JAN-1952 05:45:31-DEC-2001 17:45@AVE" ;
                U:history = "From iter23" ;
                U:_FillValue = -1.e+23f ;
        float W(DEPTH_W, LATITUDE_T, LONGITUDE_T) ;
                W:long_name = "W VELOCITY" ;
                W:units = "m/sec" ;
                W:long_name_mod = "T=01-JAN-1952 05:45:31-DEC-2001 17:45@AVE" ;
                W:history = "From iter23" ;
                W:_FillValue = -1.e+23f ;
```

Example 6: Vertical level interpolation

The example NetCDF file here is the result of the section with example 5 where we did the horizontal grid interpolation. The file still has two depth dimensions: DEPTH_T for the cell centers of the scalar variables and DEPTH_W for bottom cell wall where the vector component W is originally defined.

`ncdump -c ex6_vertInt.nc` (shortened output listed)

```
netcdf ex6_vertInt {
dimensions:
        LONGITUDE_T = 360 ;
        LATITUDE_T = 160 ;
        DEPTH_T = 23 ;
        DEPTH_W = 24 ;
variables:
        double LONGITUDE_T(LONGITUDE_T) ;
                LONGITUDE_T:long_name = "longitude" ;
                LONGITUDE_T:units = "degrees_east" ;
                LONGITUDE_T:standard_name = "longitude" ;
        double LATITUDE_T(LATITUDE_T) ;
                LATITUDE_T:long_name = "latitude" ;
                LATITUDE_T:units = "degrees_north" ;
                LATITUDE_T:standard_name = "latitude" ;
        double DEPTH_T(DEPTH_T) ;
                DEPTH_T:units = "meters" ;
        double DEPTH_W(DEPTH_W) ;
                DEPTH_W:units = "meters" ;
        float V(DEPTH_T, LATITUDE_T, LONGITUDE_T) ;
                V:long_name = "V VELOCITY" ;
                V:units = "m/sec" ;
                V:long_name_mod = "T=01-JAN-1952 05:45:31-DEC-2001 17:45@AVE" ;
                V:history = "From iter23" ;
                V:_FillValue = -1.e+23f ;
        float SALT(DEPTH_T, LATITUDE_T, LONGITUDE_T) ;
                SALT:long_name = "SALINITY" ;
                SALT:units = "PSU" ;
                SALT:long_name_mod = "T=01-JAN-1952 05:45:31-DEC-2001 17:45@AVE" ;
                SALT:history = "From iter23" ;
                SALT:_FillValue = -1.e+23f ;
        float U(DEPTH_T, LATITUDE_T, LONGITUDE_T) ;
                U:long_name = "U VELOCITY" ;
                U:units = "m/sec" ;
                U:long_name_mod = "T=01-JAN-1952 05:45:31-DEC-2001 17:45@AVE" ;
                U:history = "From iter23" ;
                U:_FillValue = -1.e+23f ;
        float W(DEPTH_W, LATITUDE_T, LONGITUDE_T) ;
                W:long_name = "W VELOCITY" ;
                W:units = "m/sec" ;
                W:long_name_mod = "T=01-JAN-1952 05:45:31-DEC-2001 17:45@AVE" ;
                W:history = "From iter23" ;
                W:_FillValue = -1.e+23f ;

// global attributes:
...

data:

...

 DEPTH_T = 5, 15, 27.5, 45, 65, 87.5, 117.5, 160, 222.5, 310, 435, 610,
    847.5, 1160, 1542.5, 1975, 2450, 2950, 3450, 3950, 4450, 4950, 5450 ;

 DEPTH_W = 0, 10, 20, 35, 55, 75, 100, 135, 185, 260, 360, 510, 710, 985,
    1335, 1750, 2200, 2700, 3200, 3700, 4200, 4700, 5200, 5700 ;
}
```

→ **Step1**: To interpolate the vertical velocity W onto the cell centers you will have to list the DEPTH_T levels together with the intlevel command, applied only to variable W:

```
cdo intlevel,5,15,27.5,45,65,87.5,117.5,160,222.5,310,435,610,
    847.5,1160,1542.5,1975,2450,2950,3450,3950,4450,4950,5450
    –selvar,W ex6_vertInt.nc out6.nc
```

`ncdump –c out6.nc liefert` (gekürzt):

```
netcdf out6 {
dimensions:
        LONGITUDE_T = 360 ;
        LATITUDE_T = 160 ;
        lev = 23 ;
variables:
        double LONGITUDE_T(LONGITUDE_T) ;
                LONGITUDE_T:long_name = "longitude" ;
                LONGITUDE_T:units = "degrees_east" ;
                LONGITUDE_T:standard_name = "longitude" ;
        double LATITUDE_T(LATITUDE_T) ;
                LATITUDE_T:long_name = "latitude" ;
                LATITUDE_T:units = "degrees_north" ;
                LATITUDE_T:standard_name = "latitude" ;
        double lev(lev) ;
                lev:long_name = "generic" ;
                lev:units = "level" ;
        float W(lev, LATITUDE_T, LONGITUDE_T) ;
                W:long_name = "W VELOCITY" ;
                W:units = "m/sec" ;
                W:long_name_mod = "T=01-JAN-1952 05:45:31-DEC-2001 17:45@AVE" ;
                W:history = "From iter23" ;
                W:_FillValue = -1.e+23f ;

// global attributes:
...

data:
...
lev = 5, 15, 27.5, 45, 65, 87.5, 117.5, 160, 222.5, 310, 435, 610, 847.5,
    1160, 1542.5, 1975, 2450, 2950, 3450, 3950, 4450, 4950, 5450 ;
}
```

→ **Step 2**: Since intlevel results in a default dimension called “lev” with the two attributes long_name = „generic” and units = „levels”, we have to adjust them all to our demands. This can be done with the help of the ncrename (-d to rename dimensions or –v to rename variables) and ncatted (attribute editor to add, delete or modify attribute values) commands of the NetCDF Operators (nco):

```
ncrename -d lev,DEPTH_T out6.nc
ncrename -v lev,DEPTH_T out6.nc
ncatted -a long_name,DEPTH_T,m,c, "height" out6.nc
ncatted -a units,DEPTH_T,m,c, "meters" out6.nc
```

→ **Step 3**: Finally, we can merge the new variable W of the file out6.nc with the other variables of our original input file ex6_vertInt.nc (reduced by the old W variable version):

```
cdo selvar,U,SALT,V ex6_vertInt.nc noW.nc
cdo merge noW.nc out6.nc out6_final.nc
```

`ncdump -c out6_final.nc liefert` (gekürzt):

```
netcdf out6_final {
dimensions:
        LONGITUDE_T = 360 ;
        LATITUDE_T = 160 ;
        DEPTH_T = 23 ;
variables:
        double LONGITUDE_T(LONGITUDE_T) ;
                LONGITUDE_T:long_name = "longitude" ;
                LONGITUDE_T:units = "degrees_east" ;
                LONGITUDE_T:standard_name = "longitude" ;
        double LATITUDE_T(LATITUDE_T) ;
                LATITUDE_T:long_name = "latitude" ;
                LATITUDE_T:units = "degrees_north" ;
                LATITUDE_T:standard_name = "latitude" ;
        double DEPTH_T(DEPTH_T) ;
                DEPTH_T:units = "meters" ;
        float V(DEPTH_T, LATITUDE_T, LONGITUDE_T) ;
                V:long_name = "V VELOCITY" ;
                V:units = "m/sec" ;
                V:long_name_mod = "T=01-JAN-1952 05:45:31-DEC-2001 17:45@AVE" ;
                V:history = "From iter23" ;
                V:_FillValue = -1.e+23f ;
        float SALT(DEPTH_T, LATITUDE_T, LONGITUDE_T) ;
                SALT:long_name = "SALINITY" ;
                SALT:units = "PSU" ;
                SALT:long_name_mod = "T=01-JAN-1952 05:45:31-DEC-2001 17:45@AVE" ;
                SALT:history = "From iter23" ;
                SALT:_FillValue = -1.e+23f ;
        float U(DEPTH_T, LATITUDE_T, LONGITUDE_T) ;
                U:long_name = "U VELOCITY" ;
                U:units = "m/sec" ;
                U:long_name_mod = "T=01-JAN-1952 05:45:31-DEC-2001 17:45@AVE" ;
                U:history = "From iter23" ;
                U:_FillValue = -1.e+23f ;
        float W(DEPTH_T, LATITUDE_T, LONGITUDE_T) ;
                W:long_name = "W VELOCITY" ;
                W:units = "m/sec" ;
                W:long_name_mod = "T=01-JAN-1952 05:45:31-DEC-2001 17:45@AVE" ;
                W:history = "From iter23" ;
                W:_FillValue = -1.e+23f ;

// global attributes:
:CDI = "Climate Data Interface version 1.3.0" ;
:Conventions = "CF-1.0" ;
:history =
     "Mon Mar 02 14:30:47 2009: cdo merge ex6_noW.nc out6.nc out6_final.nc\n",
     "Mon Mar  2 14:28:51 2009: ncatted -a long_name,DEPTH_T,m,c,height out6.nc\n",
     "Mon Mar  2 14:28:33 2009: ncatted -a units,DEPTH_T,m,c,meters out6.nc\n",
     "Mon Mar  2 14:24:15 2009: ncrename -d lev,DEPTH_T out6.nc\n",
     "Mon Mar  2 14:22:01 2009: ncrename -v lev,DEPTH_T out6.nc\n",
     "Mon Mar 02 13:49:41 2009: cdo
     intlevel,5,15,27.5,45,65,87.5,117.5,160,222.5,310,435,610,847.5,1160,1542.5,19
     75,2450,2950,3450,3950,4450,4950,5450 -selvar,W ex6_vertInt.nc out6.nc\n",
     "Wed Nov 26 17:58:07 2008: cdo remapbil,SALT.nc example5a.nc example6.nc\n",
     "Wed Nov 26 17:10:56 2008: cdo selname,U,V,W,SALT iter23m.cdf
     ../Tutorial_AvizoGreenPack/preprocessing/example5b.nc\n",
     "FERRET V5.70   19-Dec-05" ;
:CDO = "Climate Data Operators version 1.3.0 (http://www.mpimet.mpg.de/cdo)" ;

data:

DEPTH_T = 5, 15, 27.5, 45, 65, 87.5, 117.5, 160, 222.5, 310, 435, 610,
    847.5, 1160, 1542.5, 1975, 2450, 2950, 3450, 3950, 4450, 4950, 5450 ;
}
```

Example 7: MPI-OM workflow including scalar grid interpolation and grid rotation

We use the MPI-OM ocean model to show the whole workflow of vertical and horizontal grid interpolation to the scalar grid and grid rotation. The input NetCDF file is called `ex7_mpiom_gr15.nc`.

The vector components and scalar variables are all stored with the same fictive lon / lat values. For visualization and previous transformations we need the MPI-OM GR15 grid definitions that can be found in two different longitudinal ranges in the following *halo* folders:

a) longitude range [0°;360°] in
`/Main Panel/data/MPIOM/GR15/`

b) longitude range [-180°;+180°] in the subfolder
`/Main Panel/data/MPIOM/GR15/GRID_-180_180/`

In each folder you'll find the following four grid definitions:

`GR15u.nc`	Grid of zonal vector component
`GR15v.nc`	Grid of meridional vector component
`GR15w.nc`	Grid of vertical vector component
`GR15s.nc`	Grid of scalar variables (cell centre points)

→ Step 1: Interpolation of vertical velocity WO

Selection of the variable WO
Setting all missing values to 0.0 (land areas in each level)

```
cdo selvar,WO -setmisstoc,0.0 ex7_mpiom_gr15.nc gr15_WO.nc
```

Vertical interpolation to the scalar levels

```
cdo intlevel,6,17,27,37,47,57,69,83,100,123,150,183,220,263,
310,363,420,485,560,645,740,845,960,1085,1220,1365,1525,1700,
1885,2080,2290,2525,2785,3070,3395,3770,4195,4670,5170,5720
gr15_WO.nc gr15_WO_s.nc
```

Assign scalar grid in longitude range [-180°;+180°], since this is necessary for visualization with Avizo

```
cdo setgrid,/Main Panel/data/MPIOM/GR15/GRID_-180_180/GR15s.nc
-setmisstoc,0.0 gr15_WO_s.nc gr15_WO_sG.nc
```

→ Step 2: Interpolation of horizontal velocities UKO, VKE and grid rotation

Selection of the zonal vector component UKO and the meridional vector component VKE each with its individual grid. Grid longitude range [0°;360°] is used because the finally piped operator `mrotuvb` only works with these longitudes. The operator `mrotuvb` does the horizontal interpolation as well as the rotation. The output

NetCDF file ukovke_nswe.nc contains both variables, UKO and VKE, both in North-South-West-East (_nswe) coordinates.

```
cdo mrotuvb -setgrid,/Main Panel/data/MPIOM/GR15/GR15u.nc
-selvar,UKO ex7_mpiom_gr15.nc
-setgrid,/Main Panel/data/MPIOM/GR15/GR15v.nc
-selvar,VKE ex7_mpiom_gr15.nc ukovke_nswe.nc
```

For visualization with Avizo we need the longitude range [-180°;+180°] again.

```
cdo setgrid,/Main Panel/data/MPIOM/GR15/GRID_-180_180/GR15s.nc
-setmisstoc,0.0 ukovke_nswe.nc ukovke_nsweG.nc
```

→ Step 3: Scalar mask creation

Due to the interpolation processes, we have “lost” the top data level in step 1 as well as data lines along the ocean-land mask in step 2. These “lost” ocean cells have been set to 0.0 like all other land-missing values. To localize the original land-missing values and to be able to reset their 0.0 values to missing values we have to create a scalar mask. The zeros in the “lost” ocean cells we maintained then, since we need the same missing value mask for all three vector components in Avizo. We could build a mask of the lowest common grid of the three vector components. But instead we prefer to use a scalar mask to be able to colorize the vector arrows, streamlines, or trajectories with a second scalar variable in Avizo. To colorize the vector with an additional scalar variable, all vector components and the scalar variable have to have the very same missing value mask – which should then be the scalar mask.

To create this mask, we first select the very first time step from our input NetCDF file ex7_mpiom_gr15.nc. Secondly, we select a scalar variable - we use SAO (salinity) here because it does not have any values of 0.0. Thirdly, we define the scalar grid in longitude range [-180°;+180°], which is needed for visualization in Avizo. Fourthly and fifthly, we set this scalar grid file to 1.0 for all cells with a non-missing value and to 0.0 for all cells with a missing value. Finally, we rename the variable from SAO to mask_s.

```
cdo setname,mask_s -setmisstoc,0.0 -ifthenc,1.
-setgrid,/Main Panel/data/MPIOM/GR15/GRID_-180_180/GR15s.nc
-selvar,SAO -seltimestep,1 ex7_mpiom_gr15.nc GR15_MASK_S.nc
```

→ Step 4: Mask application on UKO, VKE, WO

The mask, as created in step 3, is now used on the variables UKO and VKE, in their results of step 2, and WO, in the result of step 1. The results of steps 1 and 2 are kept unchanged for all scalar cells (mask = 1.0) and are set to missing for all “non-scalar” cells (mask = 0.0).

```
cdo ifthen GR15_MASK_S.nc -selvar,UKO ukovke_nsweG.nc
gr15_UKO_ROT_SPV.nc
```

```
cdo ifthen GR15_MASK_S.nc -selvar,VKE ukovke_nsweG.nc
gr15_VKE_ROT_SPV.nc
```

```
cdo ifthen GR15_MASK_S.nc gr15_WO_sG.nc gr15_WO_sG_SPV.nc
```

→ Step 5: Merging of UKO, VKE, WO

```
cdo merge gr15_UKO_ROT_SPV.nc gr15_VKE_ROT_SPV.nc
gr15_WO_sG_SPV.nc gr15_uvw_vis.nc
```

Example 8: Vertical velocity units

In this example the vertical velocity is given in Pa/s, but we need it in m/s to load u, v and w with the same units as a 3D vector into Avizo.

```
ncdump -h ex8_wUnit.nc
```

```
netcdf ex8_wUnit {
dimensions:
        lon = 192 ;
        lat = 96 ;
        lev = 17 ;
        time = UNLIMITED ; // (124 currently)
variables:
        double lon(lon) ;
                lon:long_name = "longitude" ;
                lon:units = "degrees_east" ;
                lon:standard_name = "longitude" ;
        double lat(lat) ;
                lat:long_name = "latitude" ;
                lat:units = "degrees_north" ;
                lat:standard_name = "latitude" ;
        double lev(lev) ;
                lev:long_name = "pressure" ;
                lev:units = "Pa" ;
        double time(time) ;
                time:units = "hours since 2001-01-01 00:00" ;
        float aps(time, lat, lon) ;
                aps:long_name = "surface pressure" ;
                aps:units = "Pa" ;
                aps:code = 134 ;
                aps:table = 128 ;
                aps:grid_type = "gaussian" ;
        float q(time, lev, lat, lon) ;
                q:long_name = "specific humidity" ;
                q:units = "kg/kg" ;
                q:code = 133 ;
                q:table = 128 ;
                q:grid_type = "gaussian" ;
        float t(time, lev, lat, lon) ;
                t:long_name = "temperature" ;
                t:units = "K" ;
                t:code = 130 ;
                t:table = 128 ;
                t:grid_type = "gaussian" ;
        float u(time, lev, lat, lon) ;
                u:long_name = "u-velocity" ;
                u:units = "m/s" ;
                u:code = 131 ;
                u:table = 128 ;
                u:grid_type = "gaussian" ;
        float v(time, lev, lat, lon) ;
                v:long_name = "v-velocity" ;
                v:units = "m/s" ;
                v:code = 132 ;
                v:table = 128 ;
                v:grid_type = "gaussian" ;
        float omega(time, lev, lat, lon) ;
                omega:long_name = "vertical velocity" ;
                omega:units = "Pa/s" ;
                omega:code = 135 ;
                omega:table = 128 ;
                omega:grid_type = "gaussian" ;

// global attributes:   ... }
```

We can use the cdo operator *vertwind* to convert the vertical velocity omega (in Pa/s) into the vertical velocity W (in m/s). All variables which are necessary for this conversion (t, q, asp, omega) are already included in example6.nc and have the correct units and code numbers.

Note that the operator *vertwind* needs these exact code numbers as attributes. If your data set doesn’t have any code attributes you can include them with the help of the attribute editor of the NetCDF operators (nco) in advance (ncatted –a code, …), see link in chapter 2.1).

```
cdo vertwind ex8_wUnit.nc W_mps.nc
```

Finally, you can merge the variable W with the original dataset because you need all the wind components in the same NetCDF file in order to load the wind as a 3D vector into Avizo:

```
cdo merge ex8_wUnit.nc W_mps.nc avizo8.nc
```

Example 9: Horizontal grid interpolation to a rectilinear grid

The NetCDF file of this example has a curvilinear grid (dimensions x and y, variables lon and lat providing the transformation from x/y to lon/lat) with wind stress tau vector components from the MITgcm model.

```
ncdump -h ex9_curvGrid.nc
```

```
netcdf ex9_curvGrid {
dimensions:
        x = 750 ;
        y = 864 ;
        nv = 4 ;
        time = UNLIMITED ; // (10 currently)
variables:
        double lon(y, x) ;
                lon:long_name = "longitude" ;
                lon:units = "degrees" ;
                lon:standard_name = "grid_longitude" ;
                lon:bounds = "lon_bounds" ;
        double lon_bounds(y, x, nv) ;
        double lat(y, x) ;
                lat:long_name = "latitude" ;
                lat:units = "degrees" ;
                lat:standard_name = "grid_latitude" ;
                lat:bounds = "lat_bounds" ;
        double lat_bounds(y, x, nv) ;
        double time(time) ;
                time:units = "days since 1901-01-15 00:00" ;
        float TAUX(time, y, x) ;
                TAUX:long_name = "zonal wind stress" ;
                TAUX:coordinates = "lon lat" ;
                TAUX:history = "From data8" ;
                TAUX:_FillValue = -1.e+34f ;
        float TAUY(time, y, x) ;
                TAUY:long_name = "meridional wind stress" ;
                TAUY:coordinates = "lon lat" ;
                TAUY:history = "From data8" ;
                TAUY:_FillValue = -1.e+34f ;
```

This curvilinear grid shall be interpolated onto a rectilinear grid (0.25° x 0.25° spatial resolution = 1440 x 720 grid cells) in order to be able to use the full capacities of the Avizo Green:

```
cdo remapbil,r1440x720 ex9_curvGrid.nc rectiGrid.nc
```

7.7. Listing of Tutorial Projects

Chapter	***Filename of Avizo Project*** /work/kv0653/Tutorial_AvizoGreen_7.1/…	***Filename of exported Video*** /work/kv0653/Tutorial_AvizoGreen_7.1/**videos/…**
3	TUT_GettingStarted.hx	
4.3	TUT_temp_2D.hx	
4.5	TUT_temp_layout_2D.hx	
4.7	TUT_temp_prec_2D.hx	
4.8	TUT_temp_slp_2D.hx	
4.9	TUT_temp_slp_Earth_2D.hx	
4.11	TUT_current_Embossed_Slice_2D.hx	V_current_Embossed_Slice_2D.mpg
4.12	TUT_MITgcm_Height_Map_Slice_2D.hx	
4.13	TUT_temp_movSlice_3D.hx	
4.16	TUT_windspeed_3Slices_3D.hx	
4.17	TUT_relhum_Isosurface_3D.hx	
4.18	TUT_relhum_Isosurface_ColorTemp_3D.hx	
4.19	TUT_windspeed_IsosurfNest_3D.hx	
4.20	TUT_windspeed_AnnoIsocontour_3D.hx	
4.21	TUT_relhum_volren_3D.hx	
5.1	TUT_temp_slp_uv_2D.hx TUT_temp_slp_uvCol_2D.hx	V_temp_slp_uv_2D.mpg
5.2	TUT_slp_Isocontour_Slice_LIC.hx	
5.3	TUT_LIC_windmag_2D.hx	
5.4	TUT_LIC_slp_2D.hx	V_LIC_slp.mpg
6.1	TUT_vectorArrows_3D.hx	V_vectorArrows.mpg
6.2	TUT_streamlines_3D.hx	
6.3	TUT_streamline_SeedSurface_3D.hx	V_streamlineSeedSurface_25ts.mpg
6.4	TUT_trajectories_3D.hx TUT_trajectories_3D_Trj.hx	V_trajectories.mpg
7.1	AnimProd_temp_prec_2D.hx	V_temp_prec.mpg
7.2	AnimProd_current_Embossed_Slice_2D.hx	
7.3	AnimProd_windspeeds_3D.hx	V_windspeeds.mpg

In the videos folder you will find further interesting videos that have been produced with the Avizo Animation Producer and Movie Maker (and some have been post-processed with Adobe Premiere Pro, a powerful video editing software).