Visualisation of climate model data

Summary

Tools for working with climate data sets are often provided as source code which needs to be compiled, together with their dependencies, directly on the machine used for the the climate simulation. This requires understanding of how to work and configure common approaches to building software on Linux systems. The task outlined below concerns building a common post-processing tool for climate model data sets, `cdo`, to include options for visualisation.

Background

Modern climate models (CMs) are used to understand how the Earth's climate evolves over time and how it responds to changed conditions such as the increasing amount of CO2 in the atmosphere. A CM typically consist of a number of interacting components, where each component models a specific part of the Earth's climate such as the atmosphere, the ocean, land vegetation, etc..

The direct output from such climate simulation will be, for each component, a number of binary files that will need further processing (post-processing) before being analyzed. The initial post-processing involves operations such as concatenating separate files to single time series, creating averages over areas and time periods, creating statistical measures of the data. The final step in the post-processing is to visualize the processed data sets.

A common "work horse" for the initial post-processing is the UNIX-styled command line tool "`cdo`" [1]. `cdo` can also be interfaced with the plotting library `Magicks++`[2], interfacing the latter library allows the entire post-processing chain, both processing the data and viualising the processed data, to be executed with a single tool, "`cdo`".

Task outline

`The cdo` binary is usually available and can be installed from Linux repositories, however, this binary is typically not built with `Magics++` support. The task is to compile `cdo` from its source code and to include support for `Magics++`. This requires understanding of the common build procedures on Linux systems, such as "`cmake`" and "`./configure + make`".

Further, output from CMs is large and post-processing is preferably done locally, without having to transfer the data to a different machine. This implies that `cdo` needs to be compiled on the computer where the simulation took place using the default (non-root) user, and that additional dependencies, e.g., libraries to read the various CM binary formats, may need to be prepared/compiled as well.

Requirements

- Compile `cdo`, `Magics++` and other denpendencies as a non-root user in a Linux environment
- Use GNU Compiler Collection or Intel compilers

- Include support libraries for the two most common data formats used for CM output, GRIB [3] and netCDF[4]
- Note that `cdo` requires external libraries to be compiled as thread safe

Bibliography

1: MPI, CDO: Climate Data Operator, 2015, http://www.mpimet.mpg.de/cdo
2: ECMWF, Magics++, 2015, https://software.ecmwf.int/wiki/display/MAGP/Magics
3: ECMWF, GRIB API, 2015, https://software.ecmwf.int/wiki/display/GRIB/Home
4: Unidata, NetCDF: Network Data Common Form, 2015,
https://www.unidata.ucar.edu/software/netcdf/