NetCDF/CF file format specification for LPJ-GUESS

This document describes the NetCDF format understood by LPJ-GUESS when using the CF input module. CF is a standard for storing climate and forecasting data. Many CF compliant NetCDF files should work with LPJ-GUESS with no or only minor modifications, but since the CF standard is quite permissive we have some restrictions.

A file following the CF conventions can usually be recognized by using the "ncdump -h" command on the file, there should be a "Conventions" attribute stating that the file is a CF file.

Details about the CF standard can be found here:

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

Especially the conventions document:

- http://cf-pcmdi.llnl.gov/documents/cf-conventions/latest-cf-conventions-document-1

As the specification states, any COARDS file is a valid CF file, and it also links to the COARDS spec:

- http://ferret.wrc.noaa.gov/noaa_coop/coop_cdf_profile.html

However, the easiest way to create LPJ-GUESS compatible NetCDF files is probably by example. If you have a CF compliant data set, the easiest way to see if it works with LPJ-GUESS is simply to try it, LPJ-GUESS will complain if some requirement isn't met (with a hopefully helpful error message).

Examples

The following is a CDL specification of a monthly precipitation variable (CDL is the text based definition you see when you run ncdump -h on a NetCDF file):

```
netcdf prec {
dimensions:
        lon = 288 ;
        lat = UNLIMITED ; // (192 currently)
        time = 132 ;
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 time(time) ;
                time:units = "days since 1850-01-01 00:00:00" ;
                time:calendar = "365_day" ;
        float prec(lat, lon, time) ;
                prec:long_name = "Precipitation" ;
                prec:standard_name = "precipitation_flux" ;
                prec:units = "kg m-2 s-1" ;
                prec:cell_methods = "time: mean (interval: 30
days)" ;

// global attributes:
                :Conventions = "CF-1.4" ;
}
```

The above example has a simple grid with independent lat/lon dimensions. The next is an example file from Rossby Centre, with a rotated pole grid mapping (daily temperature forcing):

```
netcdf temp {
dimensions:
        rlat = 57 ;
        rlon = 51 ;
        time = UNLIMITED ; // (366 currently)
variables:
        double lat(rlat, rlon) ;
                lat:standard_name = "latitude" ;
                lat:long_name = "latitude" ;
                lat:units = "degrees_north" ;
        double lon(rlat, rlon) ;
                lon:standard_name = "longitude" ;
                lon:long_name = "longitude" ;
                lon:units = "degrees_east" ;
        double rlat(rlat) ;
                rlat:standard_name = "grid_latitude" ;
                rlat:long_name = "latitude in rotated pole
grid" ;
                rlat:units = "degrees" ;
                rlat:axis = "Y" ;
        double rlon(rlon) ;
                rlon:standard_name = "grid_longitude" ;
                rlon:long_name = "longitude in rotated pole
grid" ;
                rlon:units = "degrees" ;
                rlon:axis = "X" ;
        float temp(time, rlat, rlon) ;
                temp:grid_mapping = "rotated_pole" ;
                temp:_FillValue = 1.e+20f ;
                temp:standard_name = "air_temperature" ;
                temp:long_name = "Near-Surface Air
Temperature" ;
                temp:units = "K" ;
                temp:coordinates = "lon lat" ;
                temp:missing_value = 1.e+20f ;
                temp:cell_methods = "time: mean" ;
                temp:comment = "daily-mean near-surface
(usually, 2 meter) air temperature." ;
        double time(time) ;
                time:standard_name = "time" ;
                time:units = "days since 1949-12-01 00:00:00" ;
```

```
                time:calendar = "standard" ;
                time:long_name = "time" ;
                time:bounds = "time_bnds" ;
                time:axis = "T" ;

// global attributes:
                :Conventions = "CF-1.4" ;
}
```

Note that the variable is indexed with rlat and rlon, but the coordinate variables lat and lon contain the real coordinates (this is because the lat and lon coordinate variables aren't independent in this grid). Both formats can be understood by LPJ-GUESS (as well as a third form called "reduced grid", which probably should be avoided, see section 5.3 of the CF spec for an example).

The first example uses a calendar with 365 days per year. The second uses the "standard" calendar, which includes leap-days, LPJ-GUESS will simply skip these days if such a calendar is used.

Performance, dimension ordering and chunking

Note that the dimensions are ordered differently in these two examples (prec(lat, lon, time) vs. temp(time, rlat, rlon)). LPJ-GUESS doesn't care technically about the order, but having lat/lon first is much more efficient, especially for large files, since that's the order LPJ-GUESS will go through the simulation. Fortunately it's easy to change the order of dimensions with the nco command ncpdq. For instance:

```
ncpdq --permute rlat,rlon,time in.nc out.nc
```

Sometimes, changing the order of the dimensions might not be an option. For instance:

- Some programs can't work with the files unless time is the first dimension (ncview for instance)
- ncpdq loads the whole file in memory in order to reorder it, if your file is larger than available RAM memory it wont work
- If you want to distribute the files in a way so that both extracting time series (like LPJ-GUESS does) and extracting maps is reasonably efficient.

If any of the above applied, you may need to look at rechunking your files instead of changing the order of dimensions. Rechunking can be done with the standard NetCDF tool nccopy. For more information, see documentation for nccopy and these articles:

- Chunking Data: Why it Matters
- Chunking Data: Choosing Shapes

Specification

This specification is divided into two parts. First general restrictions on any CF file read by LPJ-GUESS, and second more domain specific requirements for the model (such as "temperature should be given in Kelvin").

General restrictions

- Make sure extraction of time series is efficient (see performance section above). Many NetCDF data sets are organized for efficient extraction of maps and horribly slow when it comes to extracting time series.
- The main variable and time coordinate must be numeric NetCDF classic data types (see the NetCDF standard). Specifically, data stored as strings or 64 bit integers is not supported.
- The main variable may have up to 4 dimensions. One or two spatial dimensions, one time dimension and an optional "extra" dimension which could be anything (height above ground for instance). At the moment LPJ-GUESS will expect only spatial and time dimensions for all its forcing data.
- The only calendar types supported are "standard", "gregorian", "proleptic_gregorian", "noleap" and "365_day" (See CF spec for a description of what that means). The standard and gregorian calendars are only supported from year 1583 (after the Julian/Gregorian switch) as per CF recommendations.
- If data for a location contains "missing values" (see CF and COARDS documents), that whole location will be skipped by LPJ-GUESS. Coordinate variables may not contain missing values.
- The library should be able to read in the types of coordinate systems described in 5.1 (independent lat,lon), 5.2 (two-dimensional lat,lon) and 5.3 (reduced horizontal grid), of the CF document. Regarding 5.3, we support the way it's described there, but not the "Compression by gathering" example referenced there, which works quite differently, we do not recognize the compress attribute. Section numbers above refer to version 1.6 of the CF spec.

LPJ-GUESS domain specific restrictions

Daily forcing:

- All variables within a data set are assumed to have the same spatial and temporal domain (they should all contain exactly the same grid cells and timesteps).
- Spatial coordinates are expected to represent the center of the grid cell.
- There shall be no holes in the temporal axis, each time step is exactly 24h before the next.
- Forcing data for temperature, precipitation and insolation is required.
- Daily maximum and minimum temperature is optional (needed if LPJ-GUESS is to be run with BVOC calculations switched on).
- Forcing data for soil codes, CO2 and Nitrogen deposition is _not_ included. LPJ-GUESS will get this from other sources.

Requirements for the variables:

- The temperature file shall contain a data variable named temp, with the standard_name attribute "air_temperature". Unit K. The data shall represent near surface temperature only, 2m above ground (so no height dimension).

- The precipitation file shall contain a data variable named prec, with the standard_name attribute "precipitation_flux" or "precipitation_amount". Unit kg m-2 s-1 (for flux) or kg m-2 (for amount). Note that different standard names will cause LPJ-GUESS to interpret the data differently.
- The insolation file shall contain a data variable named insol, with one of these standard_name attributes:
  surface_downwelling_shortwave_flux_in_air,
  surface_downwelling_shortwave_flux,
  surface_net_downward_shortwave_flux or cloud_area_fraction. Unit W m-2 except for cloud_area_fraction which is a fraction and should have unit "1". Note that different standard names will cause LPJ-GUESS to interpret the data differently (although the first two are aliases).
- The maximum temperature file (if included) shall contain a data variable named max_temp, with standard_name and unit as temperature above. The data shall represent daily maximum of near surface air temperature.
- The minimum temperature file (if included) shall contain a data variable named min_temp, with standard_name and unit as temperature above. The data shall represent daily minimum of near surface air temperature.

The standard_name attributes and their meaning is defined in the CF metadata standard names list:

http://cf-pcmdi.llnl.gov/documents/cf-standard-names/standard-name-table/23/cf-standard-name-table.html

Sub-daily forcing (not implemented yet):

- Within each day, the time steps must be of equal duration.
- All days have the same number of time steps.
- Sub-daily is only supported for temperature and insolation (so precipitation should still be provided as a daily variable)
- If sub-daily forcing is used, max and min temperature is no longer required (even with BVOC).
- Temperature works the same as daily temperature (same standard_name, same unit).
- For insolation, only net shortwave radiation is supported.

Monthly forcing:

- Temperature, precipitation and insolation works the same as in daily forcing (same standard_names, same units).
- Max and min temperature (if included) should not just be monthly versions (i.e. monthly extremes), but rather monthly means of the daily extremes. If a data set already includes these variables, it should be possible to find out whether it represents monthly extremes or monthly means of daily extremes by looking at the cell_methods.
- If the 'rainonwetdays' feature of LPJ-GUESS is to be used, we also require number of wet days (per month). This should be expressed with the standard_name (brace yourself)
  number_of_days_with_lwe_thickness_of_precipitation_amount_above_threshold. The CF standard explains how that one works, it's a bit special. The file shall contain a data variable named wetdays. TODO: come up with a suitable threshold.

Wiki: NetCDF/CF file format (last edited 2014-04-22 08:59:24 by joes)