Subject: NetCDF file object
Posted by Martin Schultz on Wed, 06 Sep 2000 12:33:37 GMT
View Forum Message <> Reply to Message

Hi all,

I am pleased to announce a working version of my NetCDF file object. Currently, it only supports read access, but I am planning to add methods for adding, changing, or deleting variables from a file, and for creating a new file with data that you pass to the object.

Usage is fairly simple, yet quite powerful and relatively fast as far as I can tell:

```
thefile = obj_new('mgs_ncdffile')   ; optional keyword arguments
data = thefile->GetData(variable=<string or string array>)  ;
optional keywords
obj_destroy, thefile
```

This is the minimum set of commands that you need. If you use the GetData method without the AsStructure flag, the data will be returned as mgs_basevariable objects, and you will need to call the Get method of these objects to actually access the data -- the advantage is that you get all the metadata as well. If you set the AsStructure keyword, GetData will return a structure with the variable names as tags and the data as values (similar to my ncdf_read function). Note that dimension variables that are used by any of the requested variables are automatically included in either the object array or the structure. If you don't want them, set the NoDimensions keyword.

The real power of the file object is the easiness how you can access subsets ("hyperslabs") of the data: Simply specifiy the range of the dimensions you want to limit as keywords. If you supply float (or double) values, these are interpreted as dimension values, alternatively you can supply integer values which are then interpreted as firstindex, lastindex, and stride. The respective subset for one dimension will be applied to all variables which use this dimension and the dimension variable itself, so you always get a consistent set of variables.
Examples:

(1)
```
  ozone =
thefile->GetData(var='O3',lon=,lat=,lev=30,time=,/AsStructure)
```

This will return a structure with (dimensions arbitrary):

o3 : fltarr
lon : fltarr
lat : fltarr
lev : 982.5
time : fltarr[12]
containing ozone over Europe (except the very west) for the 31st model level and for 12 time points roughly at the center of each month. Circular dimensions are currently not supported properly, i.e. you cannot wrap the longitude around.

(2)
co =
thefile-> GetData(var=['CO','EMFLX-CO'],time=0,/AsStructure,/NoDimensi ons)

This will return a structure containing:
co : fltarr[128,64,31]
emflx_co : fltarr[128,64]

I would be happy if some of you could try it out and tell me what you think. You will need at least the following files from my website

http://www.mpimet.mpg.de/~schultz.martin/idl/libmgs_objects. html

mgs_ncdffile__define.pro
mgs_basefile__define.pro
mgs_container__define.pro
mgs_basevariable__define.pro
mgs_baseobject__define.pro

and from my tools library (.../libmartin_schultz.html)

chkstru.pro
open_file.pro

Cheers,
Martin

PS: Ben (as my first registered user for the container object), I changed the Get method of MGS_Container so that it returns named objects in the order they are requested. You can set the Sroted keyword to revert to the old behaviour of returning them in the order they are stored.

PPS: If you are interested in learning about OOP, you can try to adapt the ncdffile object to read a different file format. This should be fairly easy if you plan an interface say for HDF or HDF-EOS, because their file structure philosophy is similar to netcdf. Note that most of the object power is actually coded in the basefile object, and the ncdffile object only contains routines that are specific to the netcdf format (file size of ncdffile is 17kB, whereas basefile has 50kB). I'll be happy to help you and to adopt your file formats in my library ;-)

--

[[[[[[[[[[
[[Dr. Martin Schultz   Max-Planck-Institut fuer Meteorologie
[[
[[Bundesstr. 55, 20146 Hamburg
[[
[[phone: +49 40 41173-308
[[
[[fax: +49 40 41173-298
[[
[[martin.schultz@dkrz.de
[[
[[[[[[[[[[