cdo - Climate Data Operators

Introduction

The climate data operators (cdo) are a set of command line function to modify, print out and evaluate content of netCDF (and other) files.

In contrast to NCOs, the cdos automatically change names of assumed spatial and temporal dimensions to standard names ('x', 'y' and 'time') and do not keep old names (e.g. 'COL', 'ROW' and 'TSTEP'). This is often not desired. On the other hand, cdos are simple to use. Moreover, cdos offer functions

to concatenate several files to one (e.g. We have 30 days of simulation and a netCDF file per day with results but we want to have all 30 days of results in one file.), to select and modify single variables, to perform lots of statistical analysis, to spatially interpolate data, and to rotate/transform coordinate systems.

Each cdo call, a set of 'operators' is attached which indicate which operations should be performed. The operators can be interlaced arbitrarily with the exception of some operators and technical problems (see 'Notes' section).

Documentation, html: https://code.zmaw.de/projects/cdo/embedded/1.6.3/cdo.html

Reference Card to print out: https://code.zmaw.de/projects/cdo/embedded/1.6.3/cdo_refcard.pdf

Notes

Operators like 'merge', which can accept a non-fix number of files, can only be used as left-most (last processed) operator.

Parallelisation and/or combination of to much operators may cause segmentation faults. To prevent this, one can (a) split the operators from one to several cdo calls and save interim results in temporary netCDF files or (b) add '-L' to the cdo call to deactivate parallel processing.

Basic Examples

We use one example file below. If you want to reproduce the examples you can download it here: METCRO2D_cd24_2008002_onlyWind .

We can have a look into the netCDF file via ncview and ncdump:

```
# see structure of the file but not values:
ncdump -h METCRO2D_cd24_2008002_onlyWind
```

Output:

```
neumannd@e6530L:~/Downloads$ ncdump -h
METCRO2D_cd24_2008002_onlyWind
netcdf METCRO2D_cd24_2008002_onlyWind {
dimensions:
   x = 112 ;
   y = 106 ;
   time = UNLIMITED ; // (25 currently)
variables:
   double time(time) ;
      time:standard_name = "time" ;
      time:units = "day as %Y%m%d.%f" ;
      time:calendar = "proleptic_gregorian" ;
   float WSPD10(time, y, x) ;
      WSPD10:long_name = "WSPD10          " ;
      WSPD10:units = "M/S             " ;
      WSPD10:var_desc = "wind speed at 10 m
" ;
   float WDIR10(time, y, x) ;
      WDIR10:long_name = "WDIR10          " ;
      WDIR10:units = "DEGREES         " ;
      WDIR10:var_desc = "wind direction at 10 m
" ;

// global attributes:
      :CDI = "Climate Data Interface version 1.5.3
(http://code.zmaw.de/projects/cdi)" ;
      :Conventions = "CF-1.4" ;
      :history = "Mon Jun 01 10:15:42 2015: cdo
selname,WSPD10,WDIR10 /home/neumannd/storage/M3HOME/data/lmmcip
/CD24/2008/METCRO2D_cd24_2008002
METCRO2D_cd24_2008002_onlyWind" ;
      :IOAPI_VERSION = "$Id: @(#) ioapi library version 3.0
OpenMP enabled $                             " ;
      :EXEC_ID = "lmmcip
" ;
      :FTYPE = 1 ;
      :CDATE = 2010177 ;
      :CTIME = 233803 ;
      :WDATE = 2010177 ;
      :WTIME = 233803 ;
      :SDATE = 2008002 ;
      :STIME = 0 ;
      :TSTEP = 10000 ;
      :NTHIK = 1 ;
      :NCOLS = 112 ;
      :NROWS = 106 ;
      :NLAYS = 1 ;
      :NVARS = 58 ;
      :GDTYP = 2 ;
      :P_ALP = 30. ;
      :P_BET = 60. ;
      :P_GAM = 10. ;
      :XCENT = 10. ;
      :YCENT = 55. ;
      :XORIG = -1398000. ;
      :YORIG = -1191000. ;
      :XCELL = 24000. ;
      :YCELL = 24000. ;
      :VGTYP = 2 ;
      :VGTOP = 2000.f ;
      :VGLVLS = 1.f, 0.994898f ;
      :GDNAM = "LM_METCRO_CD24_C" ;
      :UPNAM = "LM_METCRO       " ;
      :VAR-LIST = "*SCHNIPP-SCHNAPP*" ;
      :FILEDESC = "*SCHNIPP-SCHNAPP*" ;
      :CDO = "Climate Data Operators version 1.5.3
(http://code.zmaw.de/projects/cdo)" ;
}
```

Or:

```
# see content of the file graphically:
ncview METCRO2D_cd24_2008002_onlyWind &
```

Now we come to some applications of cdos.

Extract and rename variables

The file METCRO2D_cd24_2008002_onlyWind has two variable: WSPD10 and WDIR10. The first one contains the wind speed and the second one the wind direction in 10 m height. We want to extract the wind speed WSPD10, only, and rename it to u10.

```
cdo selname,WSPD10 METCRO2D_cd24_2008002_onlyWind tmp01.nc  #
extract WSPD10
cdo chname,WSPD10,u10 tmp01.nc output_ex01.nc               #
rename WSPD10 to u10
```

Alternatively to the second command we might also use expr operator.

```
cdo expr,u10=WSPD10 tmp01.nc output_ex01b.nc               #
rename WSPD10 to u10
```

The expr operator is for performing calculations. However, it can also be used for renaming. The is also the operator exprf which needs a input file containing the calculation(s). The latter is quite useful if we want to clone a variable in one file. Therefore we create a text file which we call myCalc_ex01 with the following content:

```
u10a=WSPD10;
u10b=WSPD10;
u10c=WSPD10;
```

Do not forget the ';' in the end of each line!

```
cdo exprf,myCalc_ex01 tmp01.nc output_ex01c.nc              #
rename WSPD10 to u10a, u10b and u10c (three times the same
variable)
```

The file output_ex01c.nc contains three variables, now, which have the same content. This might be useful below ... (smile) .

Combine operators

Now we combine the two operations performed above. If we chain operators we have to write a '-' in front of every operator except of the first one.

```
cdo chname,WSPD10,u10 -selname,WSPD10
METCRO2D_cd24_2008002_onlyWind output_ex02.nc  # extract WSPD10
```

You will see that output_ex02.nc and output_ex01.nc have the same structure and content.

If, then and else

For some application we want to know where the wind speed is above 7.5 m/s. Places where the wind speed is below that value, an NA should be inserted. We do it at first step by step and then in one command.

```
# here we create a MASK which is 1 everywhere where u10 > 7.5
and 0 everywhere else.
cdo gtc,7.5 output_ex01.nc tmp03a_mask.nc
# now we use the MASK to decide which values from
output_ex01.nc to take. These values are written into
output_ex03.nc
cdo ifthen tmp03a_mask.nc output_ex01.nc output_ex03.nc
```

If we put everything into one command, we get:

```
cdo ifthen -gtc,7.5 output_ex01.nc output_ex01.nc
output_ex03b.nc
```

Now we want to do something similar as above:

```
if u10 > 7.5 then keep u10
else (u10 <= 7.5) set the wind speed to 7.5
```

then we do:

```
# cdo ifthenelse TEST THEN_CASE ELSE_CASE OUTPUT
cdo ifthenelse -gtc,7.5 output_ex01.nc output_ex01.nc
-expr,u10=7.5+0*u10 output_ex01.nc output_ex03c.nc
```

Or in individual steps:

```
# make MASK
cdo gtc,7.5 output_ex01.nc tmp03b_mask.nc

# for the THEN condition nothing special is to do

# for the ELSE conditions we have to generate a field filled
with 7.5; one way doing so it to make a dummy calculation (also
done above):
cdo expr,u10=7.5+0*u10 output_ex01.nc tmp03c_constWind.nc
# alternatively one might use ifnothenc,7.5
cdo ifnotthenc,7.5 tmp03b_mask.nc tmp03d_constWind.nc
# Please have a look on both files in order to see the
difference between them. Both work in this case!

# finally, we put everything together:
cdo ifthenelse tmp03b_mask.nc output_ex01.nc
tmp03c_constWind.nc output_ex03c.nc
```

But be careful when the variables have different names!!! See below for this Fallstrick (smile) .

Fallstricke with IF

We do the same as in example 3 but we rename the wind speed field in one file

```
cdo ifthen -gtc,7.5 output_ex01.nc -chname,u10,WSPD10
output_ex01.nc output_ex04a.nc
```

This still works fine. But please have a look on the command line output anyway.

```
cdo ifthen: Started child process "gtc,7.5 output_ex01.nc
(pipe1.1)".
cdo ifthen: Started child process "chname,u10,WSPD10
output_ex01.nc (pipe1.2)".
cdo(3) chname: Processed 296800 values from 1 variable over 25
timesteps. ( 0.03s )
cdo(2) gtc: Processed 296800 values from 1 variable over 25
timesteps. ( 0.03s )
cdo ifthen: Processed 593600 values from 2 variables over 50
timesteps. ( 0.03s )
```

But if we new put METCRO2D_cd24_2008002_onlyWind instead of output_ex01.nc into the expression

```
cdo ifthen -gtc,7.5 output_ex01.nc
METCRO2D_cd24_2008002_onlyWind output_ex04b.nc
```

we get the following output onto the command line:

```
cdo ifthen: Started child process "gtc,7.5 output_ex01.nc
(pipe1.1)".
cdo ifthen: Filling up stream1 >-gtc,7.5 output_ex01.nc< by
copying the first record.
cdo(2) gtc: Processed 296800 values from 1 variable over 25
timesteps. ( 0.04s )
cdo ifthen: Processed 605472 values from 3 variables over 26
timesteps. ( 0.04s )
```

Please note the 'by copying the first record'. Which means that only the first time step of the gtc test is used for the if-then-else case. If we look with ncview into output_ex04a.nc and into output_ex04b.nc we see the difference (the first time step is the same but further time steps are not!). This is due to the fact, that the number of variables in the IF and THEN clauses differs: one in IF (u10) and two in THEN (WSPD10, WDIR10).

To correct our problem we need to rename and duplicate our variables in the IF clause. For that purpose we use the exprf function and create a text file myCalc_renameANDduplicate with the following content:

```
WSPD10=u10;
WDIR10=u10;
```

Now we modify our command:

```
cdo ifthen -exprf,myCalc_renameANDduplicate -gtc,7.5
output_ex01.nc METCRO2D_cd24_2008002_onlyWind output_ex04c.nc
```

We can have a look into the command line output again and see that "by copying the first record" does not appear anymore.

```
cdo ifthen: Started child process
"exprf,myCalc_renameANDduplicate -gtc,7.5 output_ex01.nc
(pipe1.1)".
cdo(2) exprf: Started child process "gtc,7.5 output_ex01.nc
(pipe2.1)".
cdo(3) gtc: Processed 296800 values from 1 variable over 25
timesteps. ( 0.04s )
cdo(2) exprf: Processed 296800 values from 1 variable over 25
timesteps. ( 0.04s )
cdo ifthen: Processed 1187200 values from 4 variables over 50
timesteps. ( 0.04s )
```

Text?

Expr/exprf with conditions (ifthenelse)

If you want to use if-then-else statements in expressions (using the expr and exprf operators), you need to use the "?:" operator. It is called a ternary condition (or more general ternary operator) because three input fields exists. If there would be only two (if-then statement) then it would be called a binary operator. The "?:" operator is a bit tricky to use.

Example: We have an input file which contains the variable INVAR. In the output file there should be a variable OUTVAR which is sqrt(INVAR)*2 when INVAR >= 4 and otherwise INVAR*3.

```
# correct
OUTVAR=((INVAR>=4) ? (sqrt(INVAR)*2) : (INVAR*3));

# wrong
OUTVAR=(INVAR>=4 ? (sqrt(INVAR)*2) : (INVAR*3));
# => error: "cdo aexprf (Abort): expr?expr:expr: First
expression is a constant but must be a variable!"

# wrong
((INVAR>=4) ? OUTVAR=(sqrt(INVAR)*2) : OUTVAR=(INVAR*3));
# => error: "syntax error!"

# wrong
OUTVAR=((INVAR>=4) ? sqrt(INVAR)*2 : INVAR*3);
# No error occurs but '*3' is applied to the 'then' and the
'else' statements.
# Thus, OUTVAR = sqrt(INVAR)*2*3 when INVAR >= 4 and OUTVAR =
INVAR*3 otherwise.
```

This shows, that one needs to be careful with placing brackets (better some more than too less).

-- DanielNeumann - 04 Aug 2017

Attachments 1

This topic: Main > WebHome > OverviewSoftwareTools > Cdo
Topic revision: