NCO

Table of Content

Introduction

The netCDF Operators (NCO) are a set of command line tools for manipulating netCDF files. The advantage over cdos is, that dimension and variables names are not changed but only the wished changes are performed.

NCOs also enable the user to change names of attributes, variables and dimensions and the values of attributes and values.

There is a good introduction on http://nco.sourceforge.net/nco.html.

Online Man Pages can be found on http://linux.die.net/man/1/nco.

Installation notes

Installation Basics:

see NetCDFSetup

Changes in configure and configure.ac

```
       nco_udunits2_xml=${UDUNITS2_PATH}/share/udunits
/udunits2.xml
       LDFLAGS="${LDFLAGS} -L${UDUNITS2_PATH}/lib"
```

to

```
      # Some systems install 64bit libraries into $prefix/lib64
instead of
      #  $prefix/lib. If NCO libraries are installed into
$prefix/lib64,
      #  UDUNITS presumably was installed in
$UDUNITS2_PATH/lib64 as well.
      #  Therefore, we take `basename $libdir`
(libdir=$prefix/lib|lib64),
      #  which is 'lib' or 'lib64', and append it to
UDUNITS2_PATH. If this
      #  directory does not exist, we append 'lib' to
UDUNITS2_PATH.
      #   (Daniel Neumann, 3rd June 2017)
       nco_udunits2_xml=${UDUNITS2_PATH}/share/udunits
/udunits2.xml
       if test -d "${UDUNITS2_PATH}/`basename $libdir`"; then
        LDFLAGS="${LDFLAGS} -L${UDUNITS2_PATH}/`basename
$libdir`"
      else
        LDFLAGS="${LDFLAGS} -L${UDUNITS2_PATH}/lib"
      fi
```

Basic Examples of NCO operators to fix netCDF files:

source: https://www.earthsystemcog.org/projects/dcmip-2012/netcdf

Rename a variable name via ncrename
Example: rename the variable (-v) 'temp' to the new name 'T' in file file.nc, the NCO command will not (-h) be appended as a global 'history' attribute:

```
ncrename -h -v temp,T file.nc
```

Rename a coordinate variable via ncrename
Example: In order to rename a coordinate variable 'level' to 'lev' so that it remains a

coordinate variable, you must separately rename both the dimension (-d) and the variable (-v):

```
ncrename -h -d level,lev -v level,lev in.nc
```

Changing or adding global attributes via ncatted
Example: Add the global attributes 'horizontal_resolution' and 'grid' and set them to 'medium' and 'latlon'. If a global attribute with the same name is present is will be overwritten (o), if it is not present it will be created, the information 'medium' or 'latlon' is a character string (c)

```
ncatted -h -a horizontal_resolution,global,o,c,"medium" -a
grid,global,o,c,"latlon" file.nc
```

Even more attributes can be added or changed at once.

Delete all existing global attributes via ncatted
Example: Delete (d) all global attributes in file in.nc

```
ncatted -h -a ,global,d,, in.nc
```

Changing or adding attributes of a variable via ncatted
Example: Change or add the 'long_name' attribute of the 'T' variable to 'Temperature' in file file.nc

```
ncatted -h -a long_name,T,o,c,Temperature file.nc
```

Example: Change or add the 'units' attribute of the coordinate variable 'lev' and set it to 'level', set the 'positive' indicator of the 'lev' variable 'lev' to 'down' which indicates that the 'lev' fields increases in the downward direction in file file.nc

```
ncatted -h -a units,lev,o,c,level -a positive,lev,o,c,down
file.nc
```

Exclude variables from the netCDF file and create a new output file via ncks
Example: Create the netCDF file out.nc containing all variables and any associated coordinates, except (-x) the listed variables (-v), from the netCDF in.nc

```
ncks -h -x -v co2vmr,n2ovmr,isccp_prs,isccp_prstau,isccp_tau
in.nc out.nc
```

Extract variable from a netCDF file via ncks
Example: Extract variables time and PS from netCDF in.nc. If out.nc does not exist it will be created. Otherwise you will be prompted whether to append to or to overwrite out.nc. All relevant coordinate information will be kept.

```
ncks -h -v time,PS in.nc out.nc
```

More Examples

add/modify/delete Attributes

See http://linux.die.net/man/1/ncatted for the man page of the command 'ncatted' which we will use below.

Use case: We have a variable 'lat' with the attribute 'long_name' = 'lat'. We want to change this value to 'latitude'. The file's name is 'results.nc'.

```
ncatted -a long_name,lat,m,c,latitude results.nc
results_modified.nc
# syntax: -a att_nm,var_nm,mode,att_type,att_val
#  -a         = do something with an attribute
#  mode m     = modify
#  att_type c = character
```

Use case: We have variables 'lat' and 'lon' with the attribute 'long_name' = 'lat' and 'lon', respectively. We want to change these values to 'latitude' and 'longitude'. The file's name is 'results.nc'.

```
ncatted -a long_name,lat,m,c,latitude results.nc -a
long_name,lon,m,c,longitude results.nc results_modified.nc
# syntax: see above
```

Use case: We want to add an attribute 'units' to the variable 'lat'. The unit should be 'degrees north'. The file's name is 'results.nc'.

```
ncatted -a unit,lat,c,c,"degrees north" results.nc
results_modified.nc
# syntax: -a att_nm,var_nm,mode,att_type,att_val
#  mode c = create
#  att_type c = character
```

Use case: We want to add an attribute 'coordinates' to all variables. The value should be "lat lon". We need this attribute if there are variables 'lat' and 'lon' in the same netCDF file and we want ncview to recognise these variables to be the geographical coordinates of our other variables. See here for another solution.

```
ncatted -a coordinates,,c,c,"lat lon" myFile.nc
# Add attribute 'coordinates' to every variable in the file
myFile.nc.

ncatted -a coordinates,,c,c,"lat lon" myFile.nc newFile.nc
# Add attribute 'coordinates' to every variable but save the
modified version
# in a new file (a leave myFile.nc untouched).
# If newFile.nc already exists, the used is prompted to choose
between
# 'overwrite', 'cancel' and 'append'(?).

ncatted -O -a coordinates,,c,c,"lat lon" myFile.nc newFile.nc
# Add attribute 'coordinates' to every variable but save the
modified version
# in a new file (a leave myFile.nc untouched).
# Overwrite newFile.nc if it already exists (without
confirmation or warning).
```

Sum two files

Assume we have two input file INFILE1 and INFILE2 with the variables VAR1, VAR2 and VAR3. We want to sum INFILE1 and INFILE2 in a way that the output file OUTFILE has the files VAR1, VAR2 and VAR3 which are the sums of the variables of INFILE1 and INFILE2. This is quite simple with cdo but they change attribute and file names which might be annoying.

```
ncbo --op_typ='+' -v VAR1,VAR1,VAR3 INFILE1 INFILE2 OUTFILE
```

Renaming of Atts/Vars/Dims

See http://linux.die.net/man/1/ncrename for the man page of the command 'ncrename' which we will use below.

Use case: We have a file 'results.nc' with gridded (lon-lat grid) model output. Longitudes and latitudes are saved in variables names 'x' and 'y', respectively. The corresponding dimensions are names 'x' and 'y'. We want to change their names into 'lon' and 'lat'.

```
ncrename -d x,lon -d y,lat -v x,lon -v y,lat results.nc
results_modified.nc
# syntax: -d old_name,new_name
#  -d = dimension renaming
# syntax: -d old_name,new_name
#  -v = variable renaming
```

Change values of variables

See http://linux.die.net/man/1/ncap2 for the man page of the command 'ncap2' which we will use below.

Use case: We want to set all values of the variable 'MYVAR' in the file 'results.nc' to 0.

```
ncap2 -s "MYVAR=0.0" results.nc results_modified.nc
# syntax: -s script
#  -d     = use an in-line script 'script';
#  script = can be some arithmetic expression, e.g."
#               "MYVAR=0.0"
#               "T2=T*T", if variable T already exists
```

Print out data

See http://linux.die.net/man/1/ncks for the man page of the command 'ncks' which we can be used for this purpose.

```
ncks # = nc kitchen sink

TODO
```

modify dimensions

Remove Dimension

Assume there is a netCDF file myFile.nc with variables LAT and LON which have the dimensions (time x layer x row x col). 'time' and 'layer' are 1 and we want to remove them (key words: "remove dimension", "drop dimension"). How does it work? We use the nco averager (ncwa) to do this.

```
ncwa -O -a time,layer myFile.nc newFile.nc
```

The solution was found here: http://netcdf-group.1586084.n2.nabble.com/delete-colapsed-dimension-td7574610.html

There seems to be sometimes a problem with this solution: http://stackoverflow.com/questions/20215529/delete-a-dimension-in-a-netcdf-file

Reorder Dimension

Now assume we want to exchange the LAT and LON dimensions in our newFile.nc (= transpose the data). For this purpose we use ncwa (netCDF weighted averager).

```
ncwa -a LON,LAT newFile.nc newerFile.nc
# The output file is NOT optional, here.
# or the other way around: ncwa -a LAT,LON newFile.nc
newerFile.nc
```

record dimension

The situation may occure that we have to set a record dimension.

```
ncks –mk_rec_dmn TSTEP inFile.nc outFile.nc
```

Ncks can also remove the record dimension (--no_rec_dmn or--fix_rec_dmn).

Powerful ncap2

ncap2 is the arithmetic processor of the NCOs. However, it can do much more than just doing calculations. For example, it can create dimensions, attributes and variables.

Assume we have got a file SSEMIS.test.C.2008001 which contains sea salt emissions in J and K mode mass of Na+, Cl-, SO4- and H2O (and number and surface area). We want to add I mode emissions. In order to be CMAQ-compatible, we need to increase the VAR

dimension (number of variables) and we need to modify some global variables.The new file should be called SSEMIS.test.F.2008001.

We will do the task with ncap2 as follows

- clone ACLJ, ANAJ, ASO 4J, AH2OJ, NUMACC and SRFACC into ACLI, ANAI, ASO4I, AH2OI, NUMATKN, SRFATKN (we can modify the values later)
- modify attributes of the cloned variables (ACLI:long_name="ACLI " and NOT "ACLJ "
- add a dimension VAR2 which is 18 instead of 12
- set global attribute NVARS to 18
- add variable AATFLAG2 which has VAR2 as second dimensions instead of VAR
- copy attributes from TFLAG to AATFLAG2

Then we

- remove TFLAG and sort variables alphabetically (is done automatically if we do NOT append -a) with ncks,
- rename AATFLAG2 to TFLAG and VAR2 to VAR with ncrename and
- modify the global attribute VAR-LIST with ncatted.

Questions and answers

- ***Q:*** Why do we name a new TFLAG variable AATFLAG2 and not just TFLAG2?
- ***A:*** *Later, our new TFLAG needs to be the first variable in the netCDF file. By naming th new TFLAG as AATFLAG2 it is the first one after sorting (ncks call).*

- ***Q:*** Why is there a long expression in the ncatted call?
- ***A:*** *The VAR-LIST variable needs to have the same order than the variables in the netCDF file - TFLAG excluded. Additionally, variable name + succeeding whitespaces need to be exactly 16 characters. Thus, we*
  - get the header of the netCDF file (ncdump),
  - get those lines containing 'float' (grep) (these lines contain the variable),
  - replace the parts before ('\tab float ') and after ('(TSTEP,LAY,ROW,COL) ;') each variable name by 16 white spaces after the variable name (sed) ('z' is used as tracer),
  - cut the length of variable name plus white space to 15 characters (sed) and
  - replace newline ('\n') by one whitespace (' ').

```
# call a script which contains the tasks described above
ncap2 -O -S calc05.nco SSEMIS.test.C.2008001
SSEMIS.test.E.2008001

# remove TFLAG and sort variables alphabetically (is done
automatically if we do NOT append -a)
ncks -O -x -v TFLAG SSEMIS.test.E.2008001 SSEMIS.test.F.2008001

# rename AATFLAG2 to TFLAG and VAR2 to VAR
ncrename -v AATFLAG2,TFLAG -d VAR2,VAR SSEMIS.test.F.2008001

# modify the global attribute VAR-LIST
ncatted -a 'VAR-LIST',global,m,c,"`ncdump -h
SSEMIS.test.F.2008001 | grep 'float' | sed -r 's/\tfloat(
)([^(]*)\([^)]*\) ;/z\2                /g' | sed -r 's/z([a-zA-
Z0-9 ]{15})[ ]*$/\1/g' | tr '\n' ' '`" SSEMIS.test.F.2008001
```

The calc05.nco file looks as follows.

```
defdim("VAR2",18);
global@NVARS=18;

ACLI=ACLJ/4;
ACLI@long_name="ACLI            ";
ACLI@var_desc = "hourly ACLI sea-salt emission rate
";

ASO4I=ASO4J/4;
ASO4I@long_name="ASO4I           ";
ASO4I@var_desc = "hourly ASO4I sea-salt emission rate
";

ANAI=ANAJ/4;
ANAI@long_name="ANAI            ";
ANAI@var_desc = "hourly ANAI sea-salt emission rate
";

AH2OI=AH2OJ/4;
AH2OI@long_name="AH2OI           ";
AH2OI@var_desc = "hourly AH2OI sea-salt emission rate
";

NUMATKN=NUMACC/4*1000;
NUMATKN@long_name = "NUMATKN         " ;
NUMATKN@var_desc = "hourly ATKN sea-salt emission rate
";

SRFATKN=SRFACC/4*1000;
SRFATKN@long_name = "SRFATKN         " ;
SRFATKN@var_desc = "hourly SRFATKN sea-salt emission rate
";

AATFLAG2[$TSTEP,$VAR2,'$DATE-TIME']=0;
for(*idx=0;idx<$VAR2.size-1;idx++) {
  AATFLAG2(,idx:(idx+1),)=TFLAG(,0:1,);
}
AATFLAG2@units=TFLAG@units;
AATFLAG2@long_name=TFLAG@long_name;
AATFLAG2@var_desc=TFLAG@var_desc;
```

Finished (smile) .

Download:

Files (input, output and tmp):

- SSEMIS.BOX06_cb05tucl_ae5_aq_std_cdl_EMEP.2008150.col_S__2008001 (input for attached full script)
- SSEMIS.test.C.2008001 (input for example above)
- SSEMIS.test.E.2008001 (temporary file)
- SSEMIS.test.F.2008001 (output file)

Scripts and further files:

- make.SSEMIS.test.sh (main (full) script)
- make.SSEMIS.test.R (called within the full script; not needed for the example above)
- calc05.nco (needed for the ncap2 call)

Merge CONC files and remove hour 25

For some of the features described in solution 1, at least NCO version 4.2.1 is needed. See Introduction for details on which versions are installed were.

Use case: We have three CONC files of three consecutive days with each 25 time steps. We want to merge the variables NO, NO2 and O3 in time. Problem: If we just use "ncrcat" (without -d) or "cdo mergetime" we need to create temporary files or, in the case of cdo, construct a really long expression. Additionally, keeping TFLAG is not possible with cdo because it is confused by the dimensions of TFLAG (thinks that there are to sets of spatial dimensions).

Solution 1 (shortest)

```
ncrcat -O -D 3 -d TSTEP,0,,25,24 -v TFLAG,NO,NO2,O3
CONC_2013001 CONC_2013002 CONC_2013003 CONC_2013_merged
```

Brief parameter description:

```
    -O: overwrite existing files
    -D 3: Debug Level 3 = verbose (http://nco.sourceforge.net
/nco.html#Command-Line-Options)
    -d TSTEP,0,,25,24: drop each 25th time step (see below for
details, ncrcat -d)
    -v TFLAG,NO,NO2,O3: take only the named variables
```

Comment on ncrcat -d

See http://nco.sourceforge.net/nco.html#Hyperslabs, http://nco.sourceforge.net/nco.html#Stride and http://nco.sourceforge.net/nco.html#Subcycle for -d parameter description. Mainly the keyword subcycle is of interest, here. Brief summary on what "-d TSTEP,0,,25,24" means:

- for dimension TSTEP in the finally merge file ("TSTEP,")
- start at index 0 and go to the end ("0,,", empty field means 'the end')
- in intervals of 25 time steps do something ("25,")
- take the first 24 values ("24") (== drop the 25th value)

If we left out ",24" in the end, we would extract every 25th hour.

Solution 2 (basic)

The alternative cdo expression would be:

```
myTimeSteps='1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20
,21,22,23,24'
myVars='NO,NO2,O3'
cdo mergetime -seltimestep,${myTimeSteps} -selname,${myVars}
CONC_2013001 -seltimestep,${myTimeSteps} -selname,${myVars}
CONC_2013002 -seltimestep,${myTimeSteps} -selname,${myVars}
CONC_2013003 CONC_2013_merged
```

This expression seems to be straight forward. This expression can easily be constructed for 100 and more input files. However, there is an upper threshold for the number of chained operators. For n input files we have already n * 2 + 1 operators chained. If we additionally want to extract the bottom layer only (-sellevel,1), we arrive at n * 3 + 1 operators.

ncap2

RAM variables / temporary variables

source: nco v4.6.9 manual, section *RAM-variables*

Unlike regular variables, RAM variables are never written to disk. Hence using RAM variables in place of regular variables (especially within loops) significantly increases execution speed. Variables that are frequently accessed within for or where clauses provide the greatest opportunities for optimization. To declare and define a RAM variable simply prefix the variable name with an asterisk (*) when the variable is declared/initialized. To delete RAM variables (and recover their memory) use the ram_delete() method. To write a RAM variable to disk (like a regular variable) use ram_write().

```
*temp[$time,$lat,$lon]=10.0;     // Cast
*temp_avg=temp.avg($time);      // Regular assign
temp.ram_delete();              // Delete RAM variable
temp_avg.ram_write();           // Write Variable to output

// Create and increment a RAM variable from "one" in Input
*one++;
// Create RAM variables from the variables three and four in
Input.
// Multiply three by 10 and add it to four.
*four+=*three*=10; // three=30, four=34
```

-- DanielNeumann - 04 May 2017

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