GOES16 - George's Notes

Initial Notes

- The data comes in from the LDM in the following format:

```
TIRC15 KNES 081937 PAI
```

- The current configuration (as of 3/16/2017) of the LDM is

```
NOTHER  (TI....) (....) (....)(..) (...)        PIPE    -close  /home/ldm
/bin/chomp.py /var/autofs/mnt/rapdmg1/data/goesr/\3/\4/\1_\2_\3\4_\5.nc
```

This produces files with the following format:

/var/autofs/mnt/rapdmg1/data/goesr/DDHH/MM/ZZZZSS_KKKK_DDHHMM_TTT.nc where

D = day

H = hour

M = minute

Z = region (TIRC is CONUS, TIRU is hemispheric and TIRP is perhaps Puerto Rico, **TISL = ?, TISM = ?**)

S = sensor

K = center ID, i.e. KNES (Center ID List)

T = tile ID

For example, /var/autofs/mnt/rapdmg1/data/goesr/1602/15/TISM09_KNES_160215_PAA.nc

- The new format follows:

/var/autofs/mnt/rapdmg1/data/goesr/YYYY/MMDD/HH/tiles/SSS/YYYYMMDD_HHmm_ZZZZ_SSS_TTT.nc where

Y = year

M = month

D = day

S = sensor (preceded with S)

H = hour

m = minute

Z = region

T = tile ID

For example, /var/autofs/mnt/rapdmg1/data/goesr/2017/0316/02/tiles/S09/20170316_0215_TISM_S09_PAA.nc

- The new configuration in etc/pqact.conf would be:

```
NOTHER    (TI..)(..) .... (..)(..)(..) (...)    PIPE    -close    /home/ldm
/bin/prepare_goes-r.py /var/autofs/mnt/rapdmg1/data/goesr/%Y/%m\3/\4/tiles
/S\2/%Y%m\3_\4\5_\1_S\2_\6.nc /home/ldm/logs
```

This was determined using this website as reference: pqact.conf

Archive Old Data

The old data was moved to /var/autofs/mnt/rapdmg1/data/goesr-old to be archived to the HPSS.

The data will be moved to the HPSS in a tarball: /RAPDMG/GOESR/2017/goes_r_2017_03_09-16.tar.gz

Tiling

Resources

ncview /var/autofs/mnt/rapdmg1/data/goesr/2017/0317/S10/0727_TIRC_PAE.nc

http://netcdf-group.1586084.n2.nabble.com/concatenate-netcdf-files-td3209242.html

https://sourceforge.net/p/nco/discussion/9830/thread/2782cde1/

http://www.unidata.ucar.edu/mailing_lists/archives/thredds/2011/msg00270.html

- A user from a forum (link above) wrote an NCL script to combine data. It would have to be modified to work with our data. The script is here:

Click here to expand...

```
;
--------------------------------------------------------------------
--------
;
; NCL demo program to spatially join two Netcdf Daymet tiles.
;
```

```
; 2012-oct-09   By Dave Allured, NOAA/PSD/CIRES Climate Analysis
Branch.
;       Infill of missing 2-D coordinates is not yet provided.
;
; Notes:
;
; This version requires that the two input tiles have at least
; one grid point in common.
;
; This method can generally be extended in both X and Y to any
; number of tiles, by calculating the appropriate offsets.
;
;
-------------------------------------------------------------------
--------
load "$NCARG_ROOT/lib/ncarg/nclscripts/csm/contributed.ncl"
begin
   var     = "prcp"
   file2   = "11737_1980/" + var + ".nc"
   file1   = "11917_1980/" + var + ".nc"
   outfile = var + ".1980.nc"
   f1 = addfile (file1, "r")        ; open input files
   f2 = addfile (file2, "r")
   x1 = f1->x               ; read 1-D coordinate vectors
   y1 = f1->y
   x2 = f2->x
   y2 = f2->y
   i1a = max ((/ 0, ind (x2 .eq. x1(0)) /))  ; compute starting
offsets
   i2a = max ((/ 0, ind (x1 .eq. x2(0)) /))  ; uses missing value
trick
   j1a = max ((/ 0, ind (y2 .eq. y1(0)) /))
   j2a = max ((/ 0, ind (y1 .eq. y2(0)) /))
   i1b = i1a + dimsizes (x1) - 1         ; compute ending offsets
   i2b = i2a + dimsizes (x2) - 1
   j1b = j1a + dimsizes (y1) - 1
   j2b = j2a + dimsizes (y2) - 1
   nxout = 1 + max ((/ i1b, i2b /)) ; combined grid size
   nyout = 1 + max ((/ j1b, j2b /))
   print ("nxout, nyout = " + nxout + ", " + nyout)
   ntimes = dimsizes (f1->time)     ; allocate output arrays
   dims2  = (/ nyout, nxout /)
   dims3  = (/ ntimes, nyout, nxout /)
   sample = f1->$var$(0,0,0)
   olat = new (dims2, typeof (f1->lat), "No_FillValue")
   olon = new (dims2, typeof (f1->lon), "No_FillValue")
   xout = new (dims3, typeof (sample),  sample@_FillValue)
   print ("Overlay first  tile into arrays.")
   olat(j1a:j1b,i1a:i1b)   = f1->lat    ; overlay 2-D coordinates
   olon(j1a:j1b,i1a:i1b)   = f1->lon
   xout(:,j1a:j1b,i1a:i1b) = f1->$var$  ; overlay main array
   print ("Overlay second tile into arrays.")
   olat(j2a:j2b,i2a:i2b)   = f2->lat    ; overlay 2-D coordinates
   olon(j2a:j2b,i2a:i2b)   = f2->lon
   xout(:,j2a:j2b,i2a:i2b) = (/ where (ismissing (f2->$var$), \
      xout(:,j2a:j2b,i2a:i2b), f2->$var$) /)
                        ; overlay main array; use mask
                    ; to preserve existing data
   xout&x(i2a:i2b) = (/ x2 /)       ; 1-D coordinates were
excluded by
   xout&y(j2a:j2b) = (/ y2 /)       ; masking; must copy explicitly
   print ("Write output file: " + outfile)
   if (isfilepresent (outfile)) then    ; overwrite any previous
file
      system ("rm " + outfile)
   end if
   out = addfile (outfile, "c")     ; create new output file
   copy_VarAtts (f1, out)       ; copy the global attributes
   time_stamp  = systemfunc ("date")    ; add history attribute
   out@history = time_stamp + ": Tiles joined by join.daymet.1009.
ncl"
   delete (out@tileid)          ; fix up the tile ID attribute
   out@tileid = (/ f1@tileid, f2@tileid /)
; Write variables in approximately the same order as in the input
files.
; Some variables are copied directly because they are not affected
; by the join process.
; Coordinate variables get copied automatically.
   out->lambert_conformal_conic = f1->lambert_conformal_conic
   out->lat = olat          ; write the merged 2-D coordinates
   out->lon = olon
   out->yearday   = f1->yearday     ; copy auxiliary time arrays
   out->time_bnds = f1->time_bnds
   out->$var$ = xout            ; write the main data array last,
                    ; to improve speed
   delete (out->x@_FillValue)       ; remove vestigial attributes
   delete (out->y@_FillValue)
end
exit
```

0727_TIRC_PA[A-F].nc are 1024 x 1024

0727_TIRC_PA[G-I].nc are 1024 x 512

- We will have to figure out how to combine these. We could merge 2 of them to get 1024 x 1024 but then we have 1 left over!
- The assumption of the last bullet is incorrect. See section "Grid Layouts" below for how tiles are arranged.

The following script was used to combine the tiles by x, but the result was a tall set of data instead of the correct shape (as expected). We will need to figure out how the tiles fit together then add logic to combine them the correct way (by x then by y)

Click here to expand...

```
#! /bin/bash

ts=( A B C D E F ) #G H I )

for t in ${ts[@]}; do
  echo ncecat -O /home/ldm/rapdmg1/data/goesr/2017/0317/S10
/0727_TIRC_PA$t.nc 0727_TIRC_PA$t.nc
  echo ncpdq -O -a x,record 0727_TIRC_PA$t.nc 0727_TIRC_PA$t.nc
  echo ncwa -O -a record 0727_TIRC_PA$t.nc 0727_TIRC_PA$t.nc
done

cmd="ncrcat "

for t in ${ts[@]}; do
  cmd=$cmd"0727_TIRC_PA"$t".nc "
done

cmd=$cmd" out.nc"
echo $cmd
```

The following script was written to combine TIRC sensor 10 data:

Click here to expand...

Can be improved to be more generic to tile other sensor data

```
#! /usr/bin/env python

import os

filename="/home/ldm/rapdmg1/data/goesr/2017/0320/S10
/20170320_1527_TIRC_S10_PAG.nc"
filename = filename[0: -5]

grid = []
grid.append(["AA", "AB", "AC"])
grid.append(["AD", "AE", "AF"])
grid.append(["AG", "AH", "AI"])

fcmd = "ncrcat "
for y in grid:
         rcmd = "ncrcat "
         rout = "1527_TIRC_S10_P"
         for x in y:
                  path = "{:s}{:s}.nc".format(filename, x)
                  basename = os.path.basename(path)
                  dirname = os.path.dirname(path)
                  tmpdir = "{:s}/tmp".format(dirname)
                  if not os.path.exists(tmpdir):
                           os.mkdir(tmpdir)
                  tmppath = "{:s}/{:s}".format(tmpdir,basename)
                  cmd = "ncecat -O {:s} {:s}".format(path, tmppath)
                  print cmd
                  cmd = "ncpdq -O -a x,record {:s} {:s}".format
(tmppath, tmppath)
                  print cmd
                  cmd = "ncwa -O -a record {:s} {:s}".format(tmppath,
tmppath)
                  print cmd
                  rcmd += tmppath + " "
                  rout += x+"_"
         merged_file = "{:s}/{:s}.nc".format(tmpdir,rout[0: -1])
         rcmd += merged_file + " "
         print rcmd
         print "\n"
         cmd = "ncecat -O {:s} {:s}".format(merged_file, merged_file)
         print cmd
         cmd = "ncpdq -O -a y,record {:s} {:s}".format(merged_file,
merged_file)
         print cmd
         cmd = "ncwa -O -a record {:s} {:s}".format(merged_file,
merged_file)
         print cmd
         print "\n"
         fcmd += merged_file + " "

fcmd += " {:s}/1527_TIRC_S10_FULL.nc".format(tmpdir)
print fcmd
```

Timing

- I ran a script to combine tiles based on filename. I passed the script ~/rapdmg1/data/goesr/2017/0320/S01/20170320_1542_TIRC_S01_PAY.nc and it correctly combined all 1542_TIRC_S01 tiles (30 of them) into a single NetCDF file
- I will run the script on the 5 combinations of tiles on otho and icculus to compare the time it takes to process.

tiles	Run on otho	Run on icculus
9		
30	1 hour 40 min	1 hour 18 min
108	3 hour 23 min	
16		
4		

The conclusion is this method will take too long for our purposes. I will investigate using NCL instead.

NetCDF File Cleanup

- We want to remove unused global attributes that are related to tiles after tiling since they are not relevant.

https://sourceforge.net/p/nco/discussion/9830/thread/62e97cb1/

https://sourceforge.net/p/nco/discussion/9830/thread/e6a01e3e/

```
ncatted -a tile_center_longitude,global,d,, outfile
```

This command works.The above mentions that the tool should be able to handle regex to remove multiple global attributes, but this fix was added for nco 4.5.1. otho and icculus both currently run nco 4.4.8.

We want to remove the following attributes:

- tile_center_longitude
- tile_center_latitude
- tile_row_offset
- tile_column_offset
- product_tile_width
- product_tile_height
- history (lists every tile filename)

Also, the chunking info looks incorrect in the final output file

```
chunk dimensions:
    y   = 1024 // unlimited
    x   = 1024
...
Chunking Info:      [ 1024 <y | unlimited> x 1024 <x> ]
```

Grid Layouts

TIRC

Every 30 minutes on 12 and 42

Sensors 4, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16

3x3 = 9 tiles

AA	AB	AC
AD	AE	AF
AG	AH	AI

Where A-F are 1024x1024 (xy) and G-I are 1024x512

TIRC - Sensor 1, 3, 5

6x5 = 30 tiles

AA	AB	AC	AD	AE	AF
AG	AH	AI	AJ	AK	AL
AM	AN	AO	AP	AQ	AR
AS	AT	AU	AV	AW	AX
AY	AZ	BA	BB	BC	BD

Where all tiles are 1024x1024

TIRC - Sensor 2

12x9 = 108 tiles

AA	AB	AC	AD	AE	AF	AG	AH	AI	AJ	AK	AL	AM
AN	AO	AP	AQ	AR	AS	AT	AU	AV	AW	AX	AY	AZ
BA	BB	BC	BD	BE	BF	BG	BH	BI	BJ	BK	BL	BM
BN	BO	BP	BQ	BR	BS	BT	BU	BV	BW	BX	BY	BZ
CA	CB	CC	CD	CE	CF	CG	CH	CI	CJ	CK	CL	CM
CN	CO	CP	CQ	CR	CS	CT	CU	CV	CW	CX	CY	CZ
DA	DB	DC	DD	DE	DF	DG	DH	DI	DJ	DK	DL	DM
DN	DO	DP	DQ	DR	DS	DT	DU	DV	DW	DX	DY	DZ
EA	EB	EC	ED	EE	EF	EG	EH	EI	EJ	EK	EL	EM

Where all tiles are 1024x1024

TIRP

Every 30 minutes on 06 and 36

S01	S02	S03	S04	S05	S06	S07	S08	S09	S10	S11	S12	S13	S14	S15	S16
4	16	4	1	4	1	1	1	1	1	1	1	1	1	1	1

TIRP - Sensor 2

AA	AB	AC	AD
AE	AF	AG	AH
AI	AJ	AK	AL
AM	AN	AO	AP

Where all tiles are 1024x1024 except AD, AH, AL, and AP are 768x1024

TIRP - Sensors 1, 3, 5

AA	AB
AC	AD

Where

AA is 1024x1024

AB is 896x1024

AC is 1024x896

AD is 896x896

TIRU

Every 30 minutes on 06 and 36

4 tiles for each sensor

AA	AB
AC	AD

where

AA is 1024x1024

AB is 784x1024

AC is 1024x784

AD is 784x784

TISI and TISJ

TISI = 30°N Lat. < 45°N and 75°W < Long. 90°W

TISJ = 30°N Lat. < 45°N and 60°W < Long. 75°W

every 6 minutes from 0000

S01	S02	S03	S04	S05	S06	S07	S08	S09	S10	S11	S12	S13	S14	S15	S16
4	9	4	1	4	1	1	1	1	1	1	1	1	1	1	1

S05 occasionally out (based on 3/19/2017 data feed)

TISI - Sensors 1, 3, 5

AA	AB
AC	AD

where

AA is 1024x1024

AB is 39x1024

AC is 1024x392

AD is 39x392

TISI - Sensor 2

AA	AB	AC
AD	AE	AF
AG	AH	AI

where

All tiles are 1024x1024 except

AC,AF are 78x1024

AG, AH are 1024x786

AI is 78x786

Script to generate the number of tiles per sensor statistics

Click here to expand...

```
#! /usr/bin/env python
import os
import glob
regions = ("TISI", "TISJ", "TIRC", "TIRP", "TIRU")
top_dir = "/home/ldm/rapdmg1/data/goesr/2017/0319"
for r in regions:
  print r
  line = ""
  for sensor in range(1,17):
    line += "S{:02d}  ".format(sensor)
  print line
  for h in range(0,24):
    for m in range(0,60,6):
      items=[]
      for sensor in range(1,17):
        sdir="{:s}/S{:02d}".format(top_dir,sensor)
        files="{:s}/*{:02d}{:02d}_{:s}*".format(sdir,h,m,r)
        items.append(len(glob.glob(files)))
      all_zero = True
      for i in items:
          if i != 0:
              all_zero = False
      if not all_zero:
        print "{:02d}{:02d}".format(h,m)
        line = ""
        for i in items:
            line +=str(i) + "    "
        print line
  print "\n"
```

Script to generate tile dimension information

These charts were created using the following code that can be modified to read other times and regions

Click here to expand...

```
#! /bin/bash

for f in /home/ldm/rapdmg1/data/goesr/2017/0320/S02
/20170320_1527_TIRC_S02_P*; do
  echo $f
  ncl_filedump $f | grep -A 3 "^dimensions"
  ncl_filedump $f | grep tile | grep offset
done
```

Other Notes

- If the chomp.py script is run given a file that doesn't exist, the exception causes an error instead of handling it properly

```
Traceback (most recent call last):

  File "/home/ldm/bin/chomp.py", line 11, in <module>

    if exc.errno != errno.EEXIST:

NameError: name 'errno' is not defined
```

- The metadata describing how to recompose original SCMI products with the tiles are described in section 4.2. (4.0 is only 4 section in the PDF document)
-

Projections

- It looks like different regions have different projection information.

TISI - lambert_projection

TIRU - fixedgrid_projection

TIRP - mercator_projection

TIRC - lambert_projection

TISJ - lambert_projection

- I added code to the NCL script to check the region and add the appropriate projection variable. This will have to be modified to add new regions as they become available.

Ncview

https://www.myroms.org/forum/viewtopic.php?t=1930

From: http://meteora.ucsd.edu/~pierce/docs/ncview.README

```
3) Ncview puts copies of the colormap files (files with extension
   ".ncmap") in a system-wide directory, whose location defaults to
   "/usr/local/lib/ncview".  If you want to change this location,
   edit the Imakefile and change the value of "NCVIEW_LIB_DIR"
   accordingly.  If you don't want any system-wide directory at
   all, comment the line out; you can then put the .ncmap files
   in one of 3 places: 1) In a directory named by the environmental
   variable "NCVIEWBASE"; 2) In your home directory, if you don't
   define NCVIEWBASE; 3) In the directory which you are running
   ncview from.
```

GeoServer

- The NetCDF plugin for GeoServer supports gridded NetCDF files having dimensions following the COARDS convention (custom, Time, Elevation, Lat, Lon).
    - We will need to modify these files to include these dimensions, as they currently only have x,y
    - Two dimensional non-independent latitude-longitude coordinate variables aren't currently supported, i.e. lat(x,y) and lon(x,y)
- Followed instructions here GeoServer and PostGIS Setup/Configuration
    - On satops3, use /data/goes16 instead of /d1/nnew
    - Did not modify jetty file
    - Copied pthread libs from /usr/lib64
    - The following command failed:

```
./configure --with-zlib=/data/goes16/GeoServer/NetCDF-libs --prefix=/data
/goes16/GeoServer/NetCDF-libs --enable-threadsafe --with-pthread=/usr
```

        - g++ not found
        - SNAT installed g++ and the configure command succeeded
    - configure failed for netcdf-c-4.3.3.1

```
configure: error: Can't find or link to the z library. Turn off netCDF-4
and      opendap with --disable-netcdf-4 --disable-dap, or see config.log
for errors.
```

        - Need to have z library installed – This should have been found since it was installed into /data/goes16/GeoServer/NetCDF-libs
        - I had SNAT install z lib, however it failed because it couldn't find hdf5. I need to figure out why it isn't finding these libraries.
        - I reran the command by calling configure with the environment variables being set in the call instead of setting them in the terminal environment and it worked.

```
CPPFLAGS=-I/data/goes16/GeoServer/NetCDF-libs/include LDFLAGS=-L/data
/goes16/GeoServer/NetCDF-libs/lib ./configure --prefix=/data/goes16
/GeoServer/NetCDF-libs
```

    - I did not run export when setting LDFLAGS originally and when I tried to set it with export, I had a typo so it wasn't set properly. User error.
  - Need Java installed to be able to start GeoServer – Java is in /usr/bin. Unsetting the JAVA_HOME variable solves this issue
  - I tried to start the server again but it failed with the following:

```
bash-4.1$ ./startup.sh
GEOSERVER DATA DIR is /data/goes16/GeoServer/Data
WARNING: Module not found [ssl]
2017-03-31 18:37:11.733:INFO::main: Logging initialized @179ms
2017-03-31 18:37:11.876:INFO:oejs.Server:main: jetty-9.2.13.v20150730
2017-03-31 18:37:11.890:INFO:oejdp.ScanningAppProvider:main: Deployment
monitor [file:/data/goes16/GeoServer/geoserver-2.11.0/webapps/] at
interval 1
2017-03-31 18:37:12.700:INFO:oejw.StandardDescriptorProcessor:main: NO JSP
Support for /geoserver, did not find org.eclipse.jetty.jsp.JettyJspServlet
2017-03-31 18:37:12.705:WARN:oejw.WebAppContext:main: Failed startup of
context o.e.j.w.WebAppContext@567a4593{/geoserver,file:/data/goes16
/GeoServer/geoserver-2.11.0/webapps/geoserver/,STARTING}{/geoserver}
java.lang.reflect.InvocationTargetException
```

    - It appears that ssl and/or jetty is needed.

HERE MARKS WHERE AARON STARTED WORK

- The Exception above was due to an incomplete GeoServer data directory. This was resolved by copying $GEOSERVER_HOME/data_dir/* to $GEOSERVER_DATA_DIR/. The previous contents were moved into $GEOSERVER_DATA_DIR/data-dir-20170403/. More work will follow regarding configuring this default data dir to use our data, for now it is just running with default settings to ensure it is running as expected
- Another Exception was thrown (WARN [serverStartup] - Nc4Iosp: NetCDF-4 C library not present (jna_path='/usr/local/lib/', libname='netcdf')... which related to not being able to find the netCDF API lib in LD_LIBRARY_PATH. The LD_LIBRARY_PATH was updated to point to /data/goes16/GeoServer/NetCDF-libs/**lib/**.
- GeoServer started up correctly but http://satops3.rap.ucar.edu:8080/geoserver/web/ is not reachable from external hosts. Using the 'wget' command from satops3 it is clear that GeoServer is running and receiving requests, but contacting the host from other (external) hosts was not working. Stephen said that the firewall rules need to be updated. As a workaround ssh -L can be used to access and configure the GeoServer web UI