Getting started with CLM + ParFlow

by Aditi Bhaskar
with significant input from Ian Ferguson, Vibhava Srivastava, and Reed Maxwell
14 February 2012, revised 4 September 2012

This document is loosely a continuation of the Getting Started with ParFlow document posted at http://www.umbc.edu/cuere/BaltimoreWTB/modeling.html (loosely because this document uses a more recent ParFlow version).

Now that you have gotten started with ParFlow, you may be interested in time-variable precipitation forcing. The most currently supported way of doing this is by using CLM, the land surface processes model NCAR Community Land Model, which has been coupled with ParFlow. The BCPressure part of the code is currently not configured to read a time series of pfb files as input (although this functionality existed in a previous ParFlow version, version unknown).

1. Building ParFlow with CLM
   - 1.1. To build ParFlow with CLM, simply include this flag in your configure line: --with-clm.
   - 1.2. This document uses parflow.r633. This version gave a compile error with pgf90, which was fixed by commenting out a stray print line in $PARFLOW_DIR/pfsimulator/clm/drv_readvegtf.F90 (line 61).
   - 1.3. CLM test error.
     - 1.3.1. If you get the following error and failure when running the /test/clm/clm.tcl test file:

```
At line 84 of file open_files.F90
Fortran runtime error: Bad ACCESS parameter in OPEN statement.
```

       The following lines need to be added to the clm.tcl file (~ line 290 in r633):

```
pfset Solver.CLM.Print1dOut                          False
pfset Solver.CLM.BinaryOutDir                        False
pfset Solver.WriteCLMBinary                          False
```

2. Pre-processing of CLM input files
   There are a four additional input files that you need to create in order to run CLM with ParFlow.
   - 2.1. drv_vegm.dat contains land cover data for your ParFlow grid
     - 2.1.1. Land cover data download and processing
       - 2.1.1.1. I used the MODIS IGBP 2007 global vegetation classification, which can be downloaded for any spatial extent from http://webmap.ornl.gov/wcsdown/wcsdown.jsp?dg_id=10004_31. Some GIS processing is necessary in order to find the dominant classification type for each of the model domain grid cells. The method developed by Thomas Myers, CUERE undergraduate student (UMBC) is outlined below.
       - 2.1.1.2. Once we imported the ASCII file into ArcMap as a grid, we had to determine a method of assigning each 500 m grid cell a dominant vegetation

type. The vegetation cell size is smaller than 500 m. This can be achieved using Zonal Statistics tool within the Spatial Analyst toolset. There is a "MAJORITY" option that is particularly useful in this scenario. The only hitch in this process is that we need a zone layer with a unique identifier for each zone. To solve this, we used the ETGeowizards tool (http://www.ian-ko.com/ET_GeoWizards/gw_main.htm) to generate a vector grid based on the extent of the domain. To create a unique identifier, generate a list of sequential numbers in the Field Calculator. This can be done using the following Python Code.

```
rec=0
def autoIncrement():
global rec
pStart = 1 #adjust start value, if req'd
pInterval = 1 #adjust interval value, if req'd
if (rec == 0):
rec = pStart
else:
rec = rec + pInterval
return rec
```

Then in the text box at the bottom of the field calculator where it says "YOUR_FIELDNAME=", simply type in the name of the function, which in this case is AutoIncrement().

2.1.2. Creating drv_vegm.dat

2.1.2.1. The resulting text file from 2.1.1 is used, along with a text file of the latitude and longitudes, as inputs into the landuse.f90 file. This file was written by Vibhava Srivastava, University of Florida. In this file, the sand, clay, and color values are constant for all cells. Sand and clay were previously used for the hydrology part of CLM, but have been replaced by ParFlow, and so dummy values are used. In the resulting output file, there should be one column per row with a value of 1.0, and all other values of 0.0. This is because ParFlow only allows one land cover type per grid cell.

2.1.2.2. In this version of ParFlow, drv_vegm.dat does not need to be distributed. If you are using an older version, dist_vegm.f90 distributes drv_vegm.dat for the appropriate number of processors.

2.2. Met forcing files.

2.2.1. Download NLDAS primary forcing data for your time period of interest. This is hourly data of precipitation, temperature, wind speed, etc. that is used to drive the land surface model (in our case, we will use it to force CLM).

The download URL is http://mirador.gsfc.nasa.gov/cgi-bin/mirador/LocationTimeAttribute_Search.pl?tree=project&project=NLDAS&&dataGroup=NLDAS-2%20Forcing&dataset=NLDAS_FORA0125_H.002&version=002&CGISESSID=93e66d8620f430dcc2b99d5383ca1c21

2.2.2. Note that wgrib needs to be installed to work with NLDAS data and for the next step. An executable is included in the zip file that comes with this file, and the source code can be downloaded from:

http://www.cpc.ncep.noaa.gov/products/wesley/wgrib.html

2.2.3. Ian Ferguson shared three Fortran codes which are used to process NLDAS data. For example, 1D forcing would only use the lat/long location of the center of your domain, and create a time series over time (text file). The 2D code creates a gridded output (.pfb) with one timestep per file. The instructions for using these are in their headers.

2.3. drv_clmin.dat is the CLM input file

2.3.1. It was recommended to me to only change a few lines in this file. These lines are below. The ending time can be set far in the future, because the simulation ending time is controlled by ParFlow. The drv_clmin.dat has to be modified for CLM restarts, discussed in section 4.

```
vegtf          drv_vegm.dat      Vegetation Tile Specification File
vegpf          drv_vegp.dat          Vegetation Type Parameter
metf1d         gunpat.nldas.clm.txt    Meterologic input file-
                                       valdai 3 hr
outf1d         gp.output.txt         CLM output file
poutf1d        gp.para.out.dat       CLM 1D Parameter Output File
rstf           gp.rst.               CLM active restart file
startcode      2                     1=restart file,2=defined
sss            00                     Starting Second
smn            00                     Starting Minute
shr            00                     Starting Hour
sda            01                     Starting Day
smo            10                     Starting Month
syr            2000                   Starting Year
ess            59                     Ending Second
emn            59                     Ending Minute
ehr            23                     Ending Hour
eda            30                     Ending Day
emo            09                     Ending Month
eyr            2500                   Ending Year
```

2.4. drv_vegp.dat is the vegetation type file. I did not change anything in this file from what is in the /test/clm/clm.tcl test file.

3. Setting up ParFlow.CLM runs

3.1. Tcl file modifications

3.1.1. These below lines should be added to use CLM with 1D forcing. Even if ParFlow is built with CLM, without these lines, CLM will not be used.

```
pfset Solver.LSM                          CLM
pfset Solver.CLM.MetForcing               1D
pfset Solver.CLM.MetFileName   gunpat.nldas.clm.txt
pfset Solver.CLM.MetFilePath              ./
```

3.1.2. The below lines copy the drv_clmin.dat file based on the number of processors you are using. These lines are copied from the /test/clm/clm.tcl test file. However, do not also copy the portion from the test file that copies the drv_vegm.dat files. These files cannot be distributed for multiple processors by simply copying. The distribution of drv_vegm.dat is discussed in section 2.1.3.

```
set num_processors [expr [pfget Process.Topology.P] * [pfget
Process.Topology.Q] * [pfget Process.Topology.R]]
for {set i 0} { $i <= $num_processors } {incr i} {
```

```
file delete drv_clmin.dat.$i
file copy drv_clmin.dat drv_clmin.dat.$i
}
```

3.1.3. You should set the dump Interval to time units. If it is set to a negative number, as is sometimes used for ParFlow without CLM (meaning write out solver steps instead of timesteps), the numbering between ParFlow and CLM will be off if there are any backtracks.

```
pfset TimingInfo.DumpInterval    1
```

3.2. CLM length and time units

CLM and its input files are currently configured to run at a 1 hour timestep. To modify this, you will need to modify the NLDAS F90 code for interpolations. CLM also assumes your ParFlow length units are in meters.

4. CLM restarts

Unless you are able to run your entire model in one simulation, you will likely need to restart your model. When using ParFlow. CLM, this requires more steps than when just using ParFlow. This is partly because CLM keeps track of 'real' time, and writes its own restart files. Below is a checklist for ParFlow.CLM restarts.

4.1. Checklist for ParFlow.CLM restarts:

4.1.1. Decide at what time you are restarting. Since CLM dumps restart files every 24 hours (at midnight GMT), your restart needs to be at a multiple of 24 hours from your initial time (in this example case hour 168). Copy this restart file and other required CLM files to your new run's directory.

4.1.2. Copy the restart time's pressure output to the new run's directory, and change the ParFlow IC in the tcl file. Also add pfdist and pfundist lines as needed for the ICPressure file.

```
pfset ICPressure.Type                PFBFile
pfset ICPressure.GeomNames           domain
pfset Geom.domain.ICPressure.FileName
"gpclm.out.press.00168.pfb"

pfdist gpclm.out.press.00168.pfb
pfundist gpclm.out.press.00168.pfb
```

4.1.3. Change lines 55 and 73 in drv_clmin.dat to 1's.

```
startcode      1           1=restart file,2=defined
clm_ic         1           1=restart file,2=defined
```

4.1.4. Update the PF tcl file's start count.

```
pfset TimingInfo.StartCount      168
```

4.1.5. Update the CLM start step in the tcl file, equal to start count +1:

```
pfset Solver.CLM.IstepStart            169
```

4.1.6. You can check that CLM is starting at the right time by looking in the file <runname>.out.txt, which states the CLM start time in real time.

4.2. Note that if you want to add more hours to your met file (i.e. the run has completed all the lines, corresponding to hours, in a 1D CLM met file), you need to append subsequent hours' met information at the end of your existing file. Do not create a new met file, because it will not start at the correct line.