Data Assimilation Workflow for CTSM using DART

Tim Hoar

Data Assimilation Research Section, NCAR

©UCAR 2018
All about “multi-instance”

DART is an **ensemble** data assimilation framework.

The multi-instance capability of CESM is *absolutely perfect* for this. If you haven’t used CESM in this capacity you are missing out.

The vast majority of the nuisance of running N instances (AKA ‘ensemble members’) is fundamentally transparent because of the multi-instance capability. There is a bit of setup overhead, but it’s well worth it …

A giant **Thank You** to the entire SEWG team.
The Basic Workflow for CTSM (CLM)

1. Set up and test a multi-instance case that has nothing to do with DART or DA*. The forecast length should be the desired assimilation cycle.
2. Prepare some observations, synthetic or real ...
3. Customize the DART assimilation script.
4. Enable data assimilation in the case from step 1.
5. Run.

Of course, each of these has steps … and DART has examples.

*We are currently trying to remove the need for SourceMods to support DA – we are very close.
Workflow for the visual learners:

Thanks to Andy Fox for the graphic
DART has heavily documented examples for a free run of CLM where each instance uses a unique data atmosphere.

setenv resolution f09_f09
setenv compset 2000_DATM%GSWP3v1_CLM50%BGC-CROP_SICE_SOCN_MOSART_SGLC_SWAV
setenv num_instances 80

```
./create_newcase --res ${resolution} \  
--mach ${machine} \  
--compset ${compset} \  
--case ${caseroot} \  
--project ${project} \  
--run-unsupported \  
--ninst ${num_instances} \  
--multi-driver
```
1. multi-instance setup and submission

<lots of xmlchange commands> REFDATE, STARTDATE, layout etc.

./case.setup

0[1669] cheyenne5:/<4>clm5_f09_freerun8/run > diff datm_in_0001 datm_in_0002
25,30c25,30
< 'datm.streams.txt.CPLHISTForcing.nonSolarFlux_0001 1998 1998 2010',
< 'datm.streams.txt.CPLHISTForcing.State1hr_0001 1998 1998 2010',
< 'datm.streams.txt.CPLHISTForcing.State3hr_0001 1998 1998 2010',
< 'datm.streams.txt.presaero.clim_2000_0001 1 1 1',
< "datm.streams.txt.topo.observed_0001 1 1 1"

<optional> check namelists with ./preview_namelists
<optional> stage a collection of restart files in the RUNDIR

./case.build
1. multi-instance setup and submission

./case.submit

- Test, Rinse, Repeat
- If you are creating an initial ensemble, it’s nice to output restarts every month to quantify the progress of the ensemble spread.
  - One way is to start with a single instance repeated many times and force each with a different data atmosphere.
2. Prepare some observations

DART has tools to convert many kinds of observations from their raw format (netCDF, BUFR, prepBUFR, HDF, csv, … )

It’s convenient to chunk the observations into ‘assimilation-sized’ files and tag each with the CESM time of the intended restart. If the model stops at midnight, the filenames could be `input_obs.2000-01-06-00000` (for example) and have all the observations you want to assimilate at that midnight.

The DART example scripts presume the observations have been staged.
3. Customize the DART scripts

*CESM2_0_DART_config* is quite simple and heavily internally documented. This script is normally copied to the CASEROOT directory and run interactively. It has four main functions:

1. Copies the DART executables to EXEDIR
2. Copies the DART run-time resources RUNDIR
3. Copies the DART scripts and namelist to CASEROOT
4. Uses *xmlquery, xmlchange* for additional customizations
4. Enable the assimilation

`CESM2_0_DART_config` then does the following:

```
./xmlchange DATA_ASSIMILATION=TRUE
./xmlchange DATA_ASSIMILATION_SCRIPT=${CASEROOT}/assimilate.csh
```

And prints some instructions to the screen. These instructions are also put into a file in the CASEROOT to be consulted later if necessary.

The CESM run script is configured to invoke the DA script if the model forecast was successful. Automatically. Baked into CESM 2.0.
No modifications necessary.
5. Run

./case.submit

As far as the workflow for performing a data assimilation experiment is concerned, *that’s it!* There are lots of diagnostics that can be explored to see if the assimilation is effective - but that’s another *workshop*!

Until things are confirmed to be working, I leave the short-term archiving off. Kevin Raeder has been working closely with Alice Bertini to support DA output in the archiving process.

There are more things to consider once the system is running …
Efficiency, Efficacy Concerns

• Each instance should run on as few nodes as possible to be cost effective.
  • At f09_f09 ... 4 Cheyenne nodes-instance times 40 instances ...
  • Task layout is a challenge, run all components sequentially on all PEs?

• Multiple DA cycles per job execution are naturally supported and help minimize wait time in the queue ... be aware of filesystem quotas or limitations.
  • ./xmlchange DATA_ASSIMILATION_CYCLES=10

• Prognostic variables may be dynamically specified in a DART namelist.
  • Determining which variables to update may be a challenge. Last I checked, there were more than 480 variables in a CLM5 BGC-CROP restart file.

• History files may be used to specify diagnostic CLM variables that are useful for forward operators – both grid-based and vector-based history files.
How much ensemble spread is needed?

Getting a proper initial ensemble is an area of active research.

1. Replicate an equilibrated state $N$ times.
2. Use a unique (and different!) *realistic* forcing for each to induce separate model trajectories.
3. Run them forward for “a long time”.

DART has tools to explore how much spread we NEED to capture the uncertainty in the system.
Some unobserved state variable. e.g. live root carbon, dead root carbon, canopy water …

Directly from ensemble member 1

Result of the forward observation operator for ensemble member 1

The plane defining the relationship between the observation and the model – as defined by the ensemble.

Could be Soil Temperature
updated unobserved variables

Directly from ensemble member 2

"observation" from ensemble member 2
In our assimilations, we typically use \(~80\) ensemble members. Now, we can calculate out observation increments any way we want.
c) Which means the unobserved Posterior should be:

b) which projects to here:

a) The “observation” Posterior for member 1

The plane defining the relationship between the observation and the model – as defined by the ensemble.
updating unobserved variables

The plane defining the relationship between the observation and the model – *as defined by the ensemble.*

Any part of the model: snow cover fraction, root carbon, canopy water … Could even be a model parameter!

Could be Soil Temperature