Interpreting and Analyzing Model Output II: Postprocessing Scripts

Here we discuss the use of postprocessing scripts to speed the analysis of model output. I've written a few shell scripts that use the NCO utilities to process netcdf files directly. The NCO utilities do all the work; the scripts just take care of the repetitive tasks. The NCO utilities users guide can be found here:

http://nco.sourceforge.net/nco.html

You can do a lot with these relatively simple to use command line arguments.

For our purposes, recall the the model output is stored in:

```
/scratch/nyuID/model_output/[simulation_ID]_d[final_date]
```

Simulation ID is the name you gave your run, say, t42l20e_hs, and final date is the final date of the run. Each directory containst atmos_daily.nc and atmos_average.nc. For an example, we'll assume you've run the model for 1300 days, each run containing 100 days out output.

```
/scratch/nyuID/model_output/t42l20e_hs_d00100
/scratch/nyuID/model_output/t42l20e_hs_d00200
...
/scratch/nyuID/model_output/t42l20e_hs_d01300
```

0) Get the scripts!

Start in your home directory

```
cd ~
```

You'll store and work with the scripts here:

```
mkdir models/scripts
```

Get the scripts from my directory:

```
cp ~epg2/models/scripts/transfer-data.sh models/scripts
cp ~epg2/models/scripts/create_post_processing_job_scripts models/scripts
cp ~epg2/models/scripts/template-compute_all.sh models/scripts
cp ~epg2/models/scripts/template_tracers.sh models/scripts
```

You also need pressure level interpolation scripts. Copy these from my home directory as well.

First, make a place to put the code. "riga" comes from GFDL -- they name their versions after cities, it alphabetical order. The model codes are from an earlier "memphis" version.

```
mkdir models/code/riga
cp -r ~epg2/models/code/riga/plevel_interpolation models/code/riga/
```

The second line gets you the pressure level interpolation scripts and source code. I've already compiled it for use on the hpc!

You'll execute or submit all your scripts from the `models/scripts/` directory. So go there to work.

1) transfer-data.sh

The first script organizes all the output, giving each file a unique name and putting it in one place. It's quite similar to the set up for a simulation. Specify the simulation ID, the start and end dates, and the delta increment. You must also tell it the directory where the output is stored, in this case:

```
/scratch/epg2/model_output
```

And where you'd like the data to put written out, for example:

```
/home/epg2/scratch_output/idealized_simulations
```

It will then go through the model output directories one by one, and shift them to a new directory, remove fms.x, excess input files, and most of the restart files -- it keeps every 5th one, plus the last day of the intergraion. Here's how to set it up:

```
#!/bin/bash

# This script should transfer data, and clean up duplicate files.
```

```
# experiment to transfer
exp=t42l20e_hs
# start and end days, and the number of days per file
start_day=0
end_day=1300
delta_day=100

# where the data is being moved from
indir="/scratch/epg2/model_output/"

# where the data is being moved to
outdir="/home/epg2/scratch_output/idealized_simulations"
```

With these settings, it will produce a new directory:

```
/home/epg2/scratch_output/idealized_simulations/t42l20e_hs/
```

Inside it, you'll find

```
ls /home/epg2/scratch_output/idealized_simulations/t42l20e_hs/

> average_data    raw_data

ls /home/epg2/scratch_output/idealized_simulations/t42l20e_hs/raw_data/

atmos_1day_d00100.nc
atmos_1day_d00200.nc
....
atmos_1day_d01300.nc
```

Similarly, if you look in average_data, you'll find all the atmos average files, listed by end date.
The restart and log files are found in

```
/home/epg2/scratch_output/idealized_simulations/completed_runs/
```

Take a look at the script to see how it works, etc.. The rest of the scripts will assume the data is organized as by this script.

2) Postprocessing scripts with NCO utilities.

So that you can run these scripts with the queue, they are set up like the create_runscript. In this case, the "create" script is:

models/scripts/create_post_processing_job_script

Set this file up like create_runscript. In the beginning, you need to set your username, and tell it where to find your model output files, and where to put the finished data.

```
#--- things that you'll less often change

set user              = epg2 # your user name
# location tells it where to find the input files, and where to put
the output
set indir           = /home/$user/scratch_output/idealized_simulations
set outdir          = /home/$user/scratch_output/idealized_simulations
# a place to work, write temporary files, etc.
set workdir         = /home/$user/workspace
```

The "indir" should be where the transfer-data.sh script put you model output -- note that I've matched to to the discussion above! It'll know to look for the simulation inside here. The "outdir" is where it'll write the output. Again, just give it the overall location. Often, you can put the output in the same place.

Make sure that you set up the pressure level interpolation scripts, as specified above.

After the initial set up, the main things you'll change

```
#!/bin/csh -f

# Set the run parameters here.  This script will create a job script.
#-----------------------------------------------------------------#
# Define the experiment and NYU/NCAR variables, if applicable
```

```
set job_id     = t42l20e_hs # experiment name
set t_start    = 0     # day to begin (0 for cold start)
set t_end      = 1300     # final day of integration
set delta      = 100       # time for each run (days)

set wall_clock_max = 01:00:00   # wall clock max (in hours:minutes:seconds)
set postprocessing_task  = compute_all
     # 'compute_all' to compute all the eddy statistics
     # 'interpolate_uvwT' to interpolate u,v,T, and w to pressure levels
     # 'tracers' to compute the age of air tracer statistics
     # 'delta_tracer' to compute just the delta tracer statistics

# which pressure layers you want (in Pa, not hPa) NO DECIMALS!
#L20
set plevs = "2000 7500 12500 17500 22500 27500 32500 37500 42500 47500 52500
57500 62500 67500 72500 77500 82500 87500 92500 97500"
```

job_id is the simulation id, t_start, t_end, and delta are as with other scripts. Setting the wall_clock_max is similar to for running the model; this will generally go pretty quick though, especially for low resolution runs. And it only uses 1 processor -- its not parallel.

The “postprocessing_task” is what you want to do. The options I’ve given you for now are “compute_all” and “tracers.” The first takes zonal means, and computed eddy statistics for dynamical variables u, v, w, and T. The latter works with your tracers.

Lastly, you need to set the pressure levels. It will interpolate to pressure levels, instead of the sigma or hybrid coordinates. You want levels that are close to your sigma levels, but it’s okay to change them a bit to give yourself nicer values (say, 225 instead of 223.7, etc.) These are the sigma levels from the standard t42l20e_hs integration, in hPa, not Pa!

p =

18.3940
73.5759

124.1593
174.4021
224.5358
274.6205
324.6791
374.7220
424.7547
474.7806
524.8015
574.8188
624.8333
674.8456
724.8563
774.8655
824.8737
874.8809
924.8874
974.8931

I've adjusted these a bit for the pressure levels shown above. The output will be on pressure levels. For flat integrations like this, it doesn't make much of a difference, but when you add topography, it will!

You first execute the create_post_processing_job_script, and this makes a job.[based on your simulation] script (`job.compute_all_t42l20e_hs_d1300`) for the case above, which you can submit the the queue or run directly (as long as it's not too large!)

To start, make a small script like that above, using the compute_all option. Get an interactive session, and execute the job.

./job.compute_all_t42l20e_hs_d1300

The output will appear in:

```
/home/epg2/scratch_output/idealized_simulations/t42l20e_hs/plev_data/
```

Actually, the script should tell you where the output is, and what it's called. You'll find a number of files, each with one variable, say

```
height_zonave_plev_d0_1300.nc
omega_zonave_plev_d0_1300.nc
temp_zonave_plev_d0_1300.nc
t_sq_zonave_plev_d0_1300.nc
ucomp_zonave_plev_d0_1300.nc
u_sq_zonave_plev_d0_1300.nc
uv_zonave_plev_d0_1300.nc
uw_zonave_plev_d0_1300.nc
vcomp_zonave_plev_d0_1300.nc
v_sq_zonave_plev_d0_1300.nc
vt_zonave_plev_d0_1300.nc
w_sq_zonave_plev_d0_1300.nc
wt_zonave_plev_d0_1300.nc
```

Each file contains the variable specified, either the zonal mean of a variable, the variance, or the eddy covariance. Take a look ....