Home

This is the home page for the TOY08 Student Information space.

Week 3 (Computational Science Lab) Information

Course content

Lectures
Tutorials on tools and techniques
Lab work
Large numerical problems may consume years. We will only have one week!!!
Final student presentations on Friday (~10 - 15 minutes)

Problems

Please report any problems by email to toy4@ucar.edu

Logistics

Crypto cards & UCAS passwords

Please verify operation of your crypto card and UCAR password by ssh'ing to frost and storm0 ASAP using the procedures below. Report any problems by email to toy4@ucar.edu.

Frost:
ssh -l your_ncar_login_name frost.ucar.edu
Storm0:
Ssh -l your_ncar_login_name gate.ucar.edu
Storm0.scd.ucar.edu (simply type storm0.scd.ucar.edu at the prompt after ssh'ing to gate)

Note: As of 7/15 Frost accounts have not been activated. Students will be notified as soon as their accounts are active.

Venues

Main Seminar Room: Most lectures will be held in the MSR
SCD VisLab: Lab work and tutorials will be held in the SCD VisLab
Damon Room: Six workstations are located in the Damon room and may be used for remote visualization (the systems are equipped with the TurboVNC client)

VAPOR

For much of the work in week 3 you will be using the VAPOR visual data analysis tool (www.vapor.ucar.edu)

We highly recommend that you experiment with VAPOR prior to week 3 using some of the sample data sets we provide.

VAPOR Documentation

Documentation for VAPOR may be found www.vapor.ucar.edu/doc. In particular we recommend:

An Overview of VAPOR Data Collections
VAPOR Quick Start Guide

There are a couple of options for experimenting with VAPOR in advance of week 3

LOCAL: Use your own laptop

- 3D gfx hardware required
- Installation of VAPOR and test data required (documented on the VAPOR web site)

REMOTE: Use NCAR Data Analysis and Visualization resources

- Preferred method
- Use your laptop as remote TurboVNC client
    - TurboVNC client software installation required

or

- Use NCAR workstations in Damon Room (email room)

Sample data sets

Sample data sets may be found on the VAPOR web site:

www.vapor.ucar.edu/download (if running VAPOR on your laptop)

and on the storm cluster in /cxfs/DASG/clyne/TOY (if running VAPOR remotely)

NCAR Data Analysis and Visualization resources for remote viz

Information on using NCAR's data analysis resources remotly may be found at http://www.cisl.ucar.edu/hss/dasghttp://www.cisl.ucar.edu/hss/dasg in the following documents:

Quick Start Guide
Remote Visualization (Follow directions for "external" users)
VAPOR

Scratch disk space is available on the storm cluster in /ptmp/your_login_name

TurboVNC PCs

NCAR is providing six PCs (three linux and three windows) that are equipped with the TurboVNC software needed to support remote visualization with VAPOR. These PCs will be located in the Damon Room for the first two weeks of summer school. During the third week some, or all, of these systems will be moved to the VisLab as needed. Login information is written on the blackboard in the Damon Room.

DNS and LES simulation code notes for running on Frost

These are notes for setting up and using the Joe Werne's TRIPLE code.

1. setting up your user account

- change your umask to 022 in your ~/.cshrc file (or equivalent file if you
  are using a different shell). this will make it easier to share your results
  with your team members and with your instructors.

 - add the following to your path: /home/werne/PST/bin /home/werne/triple/bin

   the best way to do this is to add it to your .cshrc file:

     set path=(. /home/werne/PST/bin /home/werne/triple/bin $path)

- create the directory ~/triple/3d/bin

2. modifying codeit (the execution script for running the TRIPLE code)

- you must set the variable $TripleHomeUser in codeit to include your home
  directory:

  $TripleHomeUser = "/home/<usr>/triple";

- don't mess with the nearby variable $TripleHome.

- to run codeit (after codein is ready), do the following:

  codeit >& codeit_X.out &

  this will run codeit and save standard output to the specified file. the
  X is intended to be incremented for continuation runs.

3. preparing your work space to run.

- you must create the run-time directory for your project runs. a subdirectory
  name has been set already in the codein file for your project. to find it,
  look in codein and find the variable called DED. The variable $a{WORKDIR}
  is equivalent to /ptmp/<usr>/, so you can leave that as is. the tail is the subdirectory
  name you need to create (e.g., mkdir /ptmp/<usr>/R5kth60a09 - where <usr>
  is your user name).

4. editing codein to prepare a run

- codein holds all the input parameters for the run. all the parameters you
  may want to modify are the following:

NX, NY, NZ, ---------- grid size
NCPU, NCPUNOW, ---------- number of processors (keep equal)
NTOTAL, NSTEP0, NSTEP1, NSTEP2, NSTEP3, ---- output timing
PREVIFAR, CURRENT0, PROG, SCRIPT, ---------- run and executable names
QUEUE, QUEUE2, JOB_T, TOT_T ---------- run queue and total job time

- keep in mind that NZ/(2*NCPU) and NX/NCPU must be integers.

- likewise, all of the NTOTAL/NSTEP? must also be integers.

- the queues available to you are

debug: NCPU=<256; JOB_T,TOT_T=< 2 hours
reserved: NCPU=<1024; JOB_T,TOT_T=< 24 hours

keep QUEUE=QUEUE2 and JOB_T=TOT_T

- setting NSTEP1 about 50 or 60 is good

- setting NSTEP2 >= 120 or larger is good. this paramter controls writing 3D volumes. if you make it too small, we'll run out of disk space.

- for the KH DNS problems, max(NX,NY,NZ) will be (800,270,800). for GW DNS problems, max(NX,NY,NZ) will be (720,360,720). larger than this and you will have trouble finishing on time. you may be able to finish by leaving these set at these values, but when fine-scale turbulence weakens, you can choose smaller values. to find acceptable values, use the command "factors NX NY NZ", where NX, etc. are replaced with the values you want to use. values must be of the form 2^n*3^m*5^p, where n, m, p are integers. more will be said about how to choose NX, NY, NZ below in the section titled "choosing the best grid". for the LES problems, you can choose grids that are a factor of 5 smaller than the maximum values mentioned above (so you won't have any trouble finishing on time).

- PREVIFAR specifies the location of the previous run on the archival storage system. the tail of PREVIFAR specifies the local run name for the restart data. after a run, you must run a processing script to concatenate the output files, move them to archival storage, and prepare the restart data. you cannot continue a run until it is processes. processing the data is described below where SCRIPT is explained.

- CURRENT0 is the name of the run you are preparing.

- PROG is the name of the executable. you should change this from run to run. it resides in your ~/triple/3d/bin/ directory. if you reuse the name from a previous run, it will overwrite the previous value used, which might cause a problem if you want to save the executable for testing or reuse if something goes wrong.

- SCRIPT is the name of the script that submits the executable to frost's queueing system. codeit runs SCRIPT to get the job in the queue. you must run SCRIPT after the run is finished to process the restart data, concatenate the volume (and other) data, and move the results to archival storage. since you must run the script by hand to process the data, it is important that each run has a unique name for SCRIPT. to run it after the job is finished, do the following.

~/triple/3d/bin/SCRIPT cleanup >& SCRIPT-cleanup.out &

here SCRIPT should be replaced with its specified value in codein. collecing the output is a good idea so we can monitor problems, if they appear.

5. choosing the best grid

- NX,NY,NZ are chosen by you. you can just set them to the largest values i mentioned earlier and hope you are able to finish your project on time. i chose the problem size so this is feasible, but it could be tight. you can speed up the evolution in terms of wall-clock time by using the minimum number of spectral modes necessary. output saved in CURRENT0/o/CURRENT0.out.output helps you evaluate this. to take advantage of this, do the following:

cd CURRENT0/o/

for KH problems: grep -e "_max " *.output
for GW problems: grep _max_xz *.output

the output includes the time on the left and values of the ratio of the grid spacing over the kolmogorov scale for different methods of averaging. for the GW problem, this number is off by a factor of 1.333 (oops!), but that's not critical. just make sure you keep this value below about 1.34 for the GW solutions and 1.8 for the KH problems and your solutions will be well resolved.

note however that the kolmogorov scale only makes sense to use as a guide when the flow is fully turbulent. during the transitional flow in the beginning and at the end, required resolutions will be greater than that indicated by this ratio. it is always wise to augment statistical metrics with visual inspection when monitoring the evolving resolution requirements.

you can change NX, NY, and NZ from run to run, and the code will perform spectral interpolation when continuing with a different grid from the previous run.

6. examining your output

- you should look at the contents of the CURRENT0/o/CURRENT0.dat1 file as soon as it is available. it contains the following parameters: 1.time, 2.dt, 3.theta^2 (theta is the potential temperature, 4.u.u (u.u = u^2+v^2+w^2), 5.blah - ignore this, 6.cost, 7.sint, 8.cosv, 9.sinv (cost, sint, cosv, sinv are the coefficients for the sine and cosine parts of the primary gravity wave as it evolves. you'll need this to assess the wave amplitude versus time.), 10.max(theta), 11.min(theta), 12.max(u), 13.min(u), 14.max(v), 15.min(v), 16.max(w), 17.min(w), 18.max(vort_1), 19.min(vort_1), 20.max(vort_2), 21.min(vort_2), 22.max(vort_3), 23.min(vort_3), 24.theta_mid, 25.u_mid, 26.w_mid. this file is a 32-bit (i.e., single precision), fixed record length, direct access (i.e., it just has numbers in it - no header info).

- the volume data is only concatenated into single files after running the post-processing script. it is then moved into the archival storage system. you can find it by issuing a command like the following:

  archive ls KH/3d/R22003d/R2200Ri05_1
  archive get -C KH/3D/R2003d/R2200Ri05_1 R2200Ri05_1.txyz.03.001

  these commands are based on the msread and mswrite commands. if you know those, they work too.

  the volume files are 32-bit, fixed-record-length, direct-access files in a left-handed coordinate frame (Z,Y,X). for the KH problem, Z is vertical, Y is in the spanwise direction across the flow direction, and X is the streamwise direction. for the GW problem, Y points in the direction of the wave's phase normal. Z is in the direction of the wave group velocity, and Y is spanwiase, orthogonal to the group velocity.

7. so where are my codein and codeit files?

- codeit is in /ptmp/werne/R22003d/codeit -- take it and modify it as necessary. The file should be copied to the directory you created in step (3)

- codein: the codein files for your project are in the following locations:

  you will need to modify them to continue with the next run.

  1. /ptmp/werne/R22003d/codein
  2. /ptmp/werne/R24003d/codein
  3. /ptmp/werne/R34003d/codein
  4. see above. you must modify the following parameters in codein for this project: ISGS=2, FSGS=0 (not used), ASGS=2, CSGS=0.1 (not used for dynamic model)

  5. /ptmp/werne/R5kth60a09/codein
  6. /ptmp/werne/R5kth72a09/codein
  7. /ptmp/werne/R5kth80a09/codein
  8. see above. you must modify the following parameters in codein for this project: ISGS=2, FSGS=0 (not used), ASGS=2, CSGS=0.1 (not used for dynamic model)

8. Put a small dummy file on the mass store. It doesn't matter which file. For example go to the directory where you copied the "codein" file and run:

   msrcp codein mss:/<MY_LOGIN_NAME_IN_CAPITAL_LETERS>/dummy_file

   For Joe Werne the command would be:

msrcp codein mss:/WERENE/dummy_file

this will make sure your mass store directory exists. Type "msls" to make sure that this file was transfered.

9. Run codeit as described in (2) above (just type "codeit" in the directory where this script is located. This will create subdirectories for the old and new simulation.

this will automatically create a version file that we need to delete. First, delete the file called /ptmp/username/R*/R*/R*_2/R*_2.version (This assumes that you are going to use the R*_2 simulation to restart the new simulation.)

10. Copy the dat0 from the restart simulation to your /ptmp directory. For example from when you are in the /ptmp/username/R*/R* directory type:

cp -R /ptmp/werne/R*/R*/R*_2/dat0 R*_2/.

10. Now, copy the .version file from /ptmp/werne using

cp -R /ptmp/werne/R*/R*/R*_2/R*.version R*_2/.

11. Prepare a test job

Edit codein and change the name of PREVIFAR directory to the name of the simulation to use for the restart. If this is the first time editing codein, you just need to use the number following the underscore listed in CURRENT0 since this was the previous job. Then increment CURRENT0, PROG, AND SCRIPT by one. Finally, change SUBMIT to "no".

12. Run codeit again to make sure that the code compiles.

13. If successful, edit codein so that SUBMIT="yes", then type "codeit" -- you are a supercomputer!

14. How long is this run going to take? How long should I run?

The time it takes the code to complete on frost is given by:

wall time (in seconds) = C.NX.NY.NZ.NTOTAL ln(NX.NY.NZ) / NCPU

where C=1.48e-6.

For the KH DNS problems, try to get at least to t=150. If you can get as far as t=200, great! But to do that you will likely have to restart with a coarser grid when the resolution requirements have reduced.

For the GW DNS problems, try to get to t=50 or so. If you can get farther, great! Your goal is to get to a time where the max(vorticity components) are significant and fluctuating.

Data Analysis

The analysis you should do on your DNS/LES data is described in the HomeWork.pdf file located in frost:/ptmp/werne/student_info.

You will also find there an idl routine named lookdat1.pro that plots <uu+vv+ww> and Ri<tt> and the vorticity maxima versus time. Here <> refers to a volume average. Read the comments in the idl routine.

Preparing your simulation output data for analysis with VAPOR

To analyze your 3D output fields with VAPOR and IDL you will need to do the following:

- Copy the data you are interested in to the storm cluster. You only have 150GBs of space in your /ptmp directory on storm so don't copy more than you need.
- Transform the raw outputs from the simulation code into the VAPOR data format. The process for this is described by "VAPOR Documentation" section above

The basic steps for these operations are as follows:

1. Log on to one of the storm nodes (storm0, storm1... storm4) by ssh'ing through gate.ucar.edu:

- ssh my_login_name@gate.ucar.edu

at the gate.ucar.edu prompt type one of the storm node names. For example:

- storm2.scd.ucar.edu

2. From your storm window identify which files on the MSS archive you will copy to storm. The 3D volume file names contain the string "xyz" in them. The string the precedes the "xyz" indicates the variable name. For example, "uxyz" would be the U component of velocity. The files with the ".list" extension contain ascii text with other useful information about your run. The following command will list all of the files that you have:

- msls -lR

3. Change working directories to your storm /ptmp directory (all storm nodes see the same /ptmp file system):

- cd /ptmp/my_user_name

4. Copy the files from the MSS with the msrcp command. For example:

- msrcp mss:/USER_NAME_IN_CAPS/file_name file_name

5. Set up your environment to use the VAPOR commands:

- source /fs/local/bin/vapor-setup.csh

You can put the above source command in your .cshrc

6. Create a VAPOR .vdf file as described in the VAPOR documentation above with the command vdfcreate. The command:

- vdfcreate -level 2 -dimension 401x150x400 -varnames t:u:v:w -numts 20 R2200Ri05.vdf

would create a .vdf file named "R2200Ri05.vdf" that is ready to contain up to 20 time steps; 4 variables named "t", "u", "v", and "w"; for a volume with grid dimension 401x150x400; and would have two multi-resolution transformation levels. You may want more than two transform levels for larger data than 401x150x400. Note that X and Z are transposed in Joe's simulation code.

7. For each simulation output file transform it into a VAPOR Data Collection (VDC), associated with your .vdf file, using the raw2vdf command. For example, the command

- raw2vdf -swapbytes -varname t -ts 0 R2200Ri05.vdf R2200Ri05_1.txyz.00.001

would transform the temperature volume file, "R2200Ri05_1.txyz.00.001", inserting it at the first time step in the VDC. Note, the -swapbytes option is need to handle endian differences between Frost and the storm cluster.

8. Now you are ready to visualize with VAPOR!

IDL Scripts

The template IDL scripts that you will need to create derived quantities from you 3D data fields are on the storm cluster in /cxfs/DASG/clyne/TOY/idl

Copy all of these files to your /ptmp directory before using them. **Note, the scripts write data to the current working directory. Do NOT try to run them from your home directory. They should be run from /ptmp or you will run out of space.**

These scripts are intended to be used from data "exported" from VAPOR. If you want to read data in other formats you will have to modify them.

There are two scripts that are intended to be edited by you as needed:

vorticity_ie.pro : computes vorticity and writes the result to a vapor data set, vorticity_ie.vdf

eke_ie.pro : computes eddy kinetic energy and writes results to a vapor data set, eke_ie.vdf

To run either script you must first "export" data from VAPOR via the 'Data/Export to IDL' button. Be careful about exporting a region that is too large, or you will run out of memory, etc.

Multiple time steps

A number of you have asked how to calculate derived quantities (e.g. vorticity) for multiple time steps. There is a new script, vorticity_multi_ie.pro, that demonstrates how to accomplish this for vorticity. It's a quick hack and you'll need to edit to variables before you can use it: TSFIRST, and TSLAST. These are documented in the script. In order to use it you'll need new versions of expregions.pro and impregions.pro as well. All of these scripts may be found in

/cxfs/DASG/clyne/TOY/idl

Making animations out of jpeg files

All, you can create mpeg animations on the storm nodes with the ffmpeg command. The basic syntax is

```
ffmpeg -r FRAMERATE -qmax QUALITY -i FILEPREFIX%04d.jpg MOVIFILE.mp4
```

Where

FRAMERATE : is the movie play back rate in frames per second. I suggest 5 or 10
QUALITY: is an integer quality number. 2 will give the best results (what I recommend). Larger values will give poorer quality, but smaller files
FILEPREFIX: is the prefix for the jpeg files you generated with VAPOR. The '%04d' is used to match the index numbers of your jpegs.
MOVIEFILE: is simply the name of your output file.

For example:

```
ffmpeg -r 10 -qmax 2 -i otp%04d.jpg mymovie.mp4
```

Note, I believe the first jpeg file must be indexed 0000, for example myfile0000.jpg. Also, the jpegs must be in sequential order with no gaps

Also, you may or may not be able to play these from within powerpoint/keynote. If you can't, you should still be able to play them in a stand alone movieplayer (e.g. quicktime, windows media player)