1. Preparing the cluster

(a) Install the Korn shell

The model uses a lot of Korn shell scripts and needs the model user to have to Korn shell as the login shell. Th installation didn't install this by default so we have to do it manually.

cd /tftpboot/rpm

rpm -i pdksh-5.2.14-12.i386.rpm

cpush --source=pdksh-5.2.14-12.i386.rpm --destination=/tmp/pdksh-5.2.14-12.i386.rpm

cexec -c "rpm -i /tmp/pdksh-5.2.14-12.i386.rpm"

(If the pdksh rpm file isn't in /tftpboot/rpm then you will have to copy it off the lab server).

(b) Add a new user

/usr/sbin/useradd -s /bin/ksh <newuser>

Change the password as appropriate.

(c) Re-install the Portland compiler

The Portland compiler has an annoying feature of a time delay between compilations. The time delay gets gre evaluation period draws to a close.

When compiling more than 500 routines this can slow down your compilation a great deal so it is necessary to the Portland compiler to prevent this.

cd /root/pgi
./install

Choose option 5 and install the compiler where you put it before.

(d) Copy the model off the lab server

Logon as the model user and install the model code.

scp student@140.105.19.181:/home/school/case_study_B.tgz .
tar -xvzf case_study_B.tgz
rm case_study_B.tgz
cd umdir
mv * ..
mv .* ..
cd ..

. ./.kshrc

and now you are ready to begin!

2. Compile gcom (the message passing library)

cd $HOME/um/gcom/rel_1m1s5x5/build/
make -f Makefile.linux
mv libgcom1m1s5x5_mpi.a ..

If this doesn't work first time it may be necessary to remove the *.o, *.f and *.a files and start again.

Check libgcom1m1s5x5_mpi.a exists before proceeding.

3. Compile the small executables

The small executables are used for reformatting and creating model dumps between different data formats

cd $HOME/source
./compile_execs

Test pumf (**p**rint **u**nified **m**odel **f**ile)out on a dataset.

cd $HOME/HADCM3L/startdumps

See which pumf you are referencing:
which pumf
It should be $HOME/um/vn4.5/utils/pumf

pumf acfob.astart

should produce something like:

```
Header output in:,/home/um/tmp/pumf_head.29272
Field output in:,/home/um/tmp/pumf_field.29272
```

see the output from the second output line - i.e.

more $HOME/tmp/pumf_head.29272

and you should see something such as:

```
FILE STATUS
===========
OPEN:  File acfob.astart to be Opened on Unit 20 Exists


Maximum Field Size =    8192
```

```
!!!! STASH_MSTR
/home/um/um/vn4.5/ctldata/STASHmaster/STASHmaster_A
!!!! STASH_MSTR
/home/um/um/vn4.5/ctldata/STASHmaster/STASHmaster_O
!!!! STASH_MSTR
/home/um/um/vn4.5/ctldata/STASHmaster/STASHmaster_S
!!!! STASH_MSTR
/home/um/um/vn4.5/ctldata/STASHmaster/STASHmaster_W

FIXED LENGTH HEADER
-------------------
Dump format version-32768
UM Version No         405
Atmospheric data
On hybrid levels
Over global domain
Instantaneous dump
Exp No =     1 Run Id =     0
360-day calendar
Arakawa B grid
                  Year  Month Day Hour Min  Sec DayNo
Data time     = 1991    9    1    0    0    0  331
Validity time = 2835   12    1    0    0    0  331
Creation time = 2000    7   19   10   16   55*****
                      Start   1st dim  2nd dim  1st parm  2nd parm
Integer Consts          257       29               29
Real Consts             286       38               38
Level Dep Consts        324       19        6      19         6
Row Dep Consts          438       73        3      73         3
Column Dep Consts   -32768   -32768   -32768        0         0
Fields of Consts    -32768   -32768   -32768        0         0
Extra Consts        -32768   -32768                 0
History Block       -32768   -32768                 0
CFI No 1            -32768   -32768                 0
CFI No 2            -32768   -32768                 0
CFI No 3            -32768   -32768                 0
Lookup Tables          657       64      225       64       225
```

and much more.....

Check with one of the tutors if you don't see the above output.

4. Compile the model

cd $HOME/source
./compile_model

The compilation should take around five minutes.

There should be a warning for the routine fill3a.f:

```
compiling fill3a.f
PGF90-W-0164-Overlapping data initializations of l_in_climat (fill3a.f)
PGF90-W-0164-Overlapping data initializations of l_in_climat (fill3a.f)
```

```
PGF90-W-0164-Overlapping data initializations of l_in_climat (fill3a.f)
PGF90-W-0164-Overlapping data initializations of l_in_climat (fill3a.f)
PGF90-W-0164-Overlapping data initializations of l_in_climat (fill3a.f)
  0 inform,   5 warnings,   0 severes, 0 fatal for r2_set_aerosol_field
```

This is due to multiple definitions - not a problem really.

The link step will list unfound routines:

```
mpif90 blkdata.o umshell1.o libum1.a \
-noinhibit-exec -Bstatic -Wl,-warn-once -L. -L../../um/gcom/rel_1m1s5x5 -
L/home/um/mpich-1.2.3/lib -lmpich -lgcom1m1s5x5_mpi -o
/home/um/PUM_Output/vn4.5/datam.xaaqg/xaaqg.exe
pgf90-warning-Unknown option passed to linker: -noinhibit-exec
libum1.a(atmstep1.o): In function `atm_step_':
atmstep1.o(.text+0x1444): undefined reference to `iau_ctl_'
libum1.a(initphy1.o): In function `initphys_':
initphy1.o(.text+0xc9): undefined reference to `swlkin_'
initphy1.o(.text+0x1ef): undefined reference to `lwlkin_'
libum1.a(setlscl1.o): In function `setlscld_':
setlscl1.o(.text+0xaa5): undefined reference to `rhcrit_calc_'
libum1.a(varctl1.o): In function `var_ctl_':
varctl1.o(.text+0xd36): undefined reference to `var_umprocessing_'
libum1.a(writdm1a.o): In function `writdump_':
writdm1a.o(.text+0x1bd3): undefined reference to `buffout_shmem_'
libum1.a(zonmctl1.o): In function `zonmctl_':
zonmctl1.o(.text+0x2224): undefined reference to `zonm_atm_'
libum1.a(readdm1a.o): In function `readdump_':
readdm1a.o(.text+0x203c): undefined reference to `buffin_shmem_'
libum1.a(readdm1a.o): In function `readacobs_':
readdm1a.o(.text+0x3104): undefined reference to `buffin_acobs_'
libum1.a(pp2griba.o): In function `pp2grib_':
pp2griba.o(.text+0x1102): undefined reference to `coder_'
libum1.a(stwork1a.o): In function `stwork_':
stwork1a.o(.text+0x2222): undefined reference to `stocgt_'
stwork1a.o(.text+0x2494): undefined reference to `stwvgt_'
libum1.a(ac_ctl1.o): In function `ac_ctl_':
ac_ctl1.o(.text+0x3427): undefined reference to `swapbounds_shmem_'
ac_ctl1.o(.text+0x3b93): undefined reference to `ac_'
ac_ctl1.o(.text+0x4887): undefined reference to `stratq_'
libum1.a(chemctl1.o): In function `chem_ctl_':
chemctl1.o(.text+0xa14): undefined reference to `gravsett_'
chemctl1.o(.text+0x133d): undefined reference to `sulphur_'
chemctl1.o(.text+0x13a4): undefined reference to `new2old_'
chemctl1.o(.text+0x142a): undefined reference to `sootscav_'
libum1.a(cldctl1.o): In function `cld_ctl_':
cldctl1.o(.text+0xfba): undefined reference to `area_cld_'
libum1.a(inacctl1.o): In function `in_acctl_':
inacctl1.o(.text+0x43e): undefined reference to `ac_init_'
inacctl1.o(.text+0x501): undefined reference to `var_umsetup_'
libum1.a(rad_ctl1.o): In function `rad_ctl_':
rad_ctl1.o(.text+0x4802): undefined reference to `swrad_'
rad_ctl1.o(.text+0x65e9): undefined reference to `swdkdi_'
rad_ctl1.o(.text+0x7c20): undefined reference to `lwrad_'
libum1.a(vdf_ct1.o): In function `vdf_ctl_':
vdf_ct1.o(.text+0xcf6): undefined reference to `vdif_ctl_'
libum1.a(coex1a.o): In function `coex_':
```

```
coex1a.o(.text+0x83): undefined reference to `cri2ibm_'
coex1a.o(.text+0x1e2): undefined reference to `ibm2cri_'
libum1.a(coex1a.o): In function `coex2_':
coex1a.o(.text+0xc41): undefined reference to `strmov_'
libum1.a(coex1a.o): In function `instin_':
coex1a.o(.text+0x279e): undefined reference to `movbit_'
```

Anything other than these missing 30 routines indicates a problem that will need resolving before the ne

5. Running the model

cd $HOME
umsubmit

The default is a two processor job which will run on the first two nodes in your cluster.

While the model is running it puts some output in the directory $HOME/umui_out

The last file in this directory (use ls –lrt) is the current model run output. To see the output as the model pr

tail –f <filename>

The last line should have shomething like:

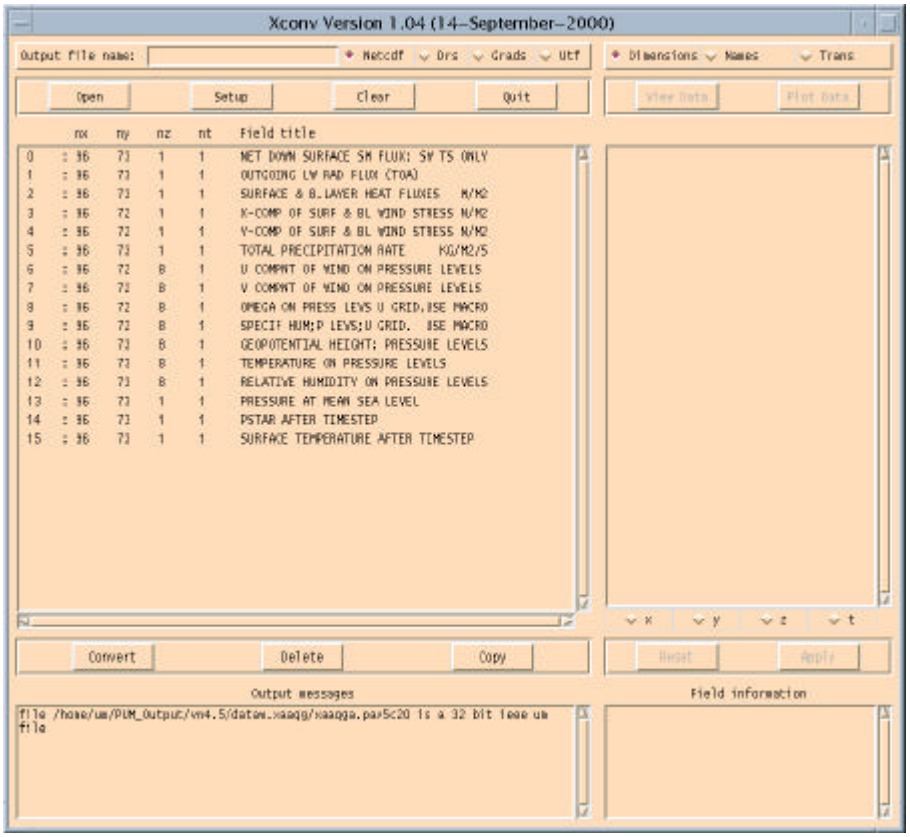
```
ATMOS TIMESTEP 12
```

With one instance of the above per processor.

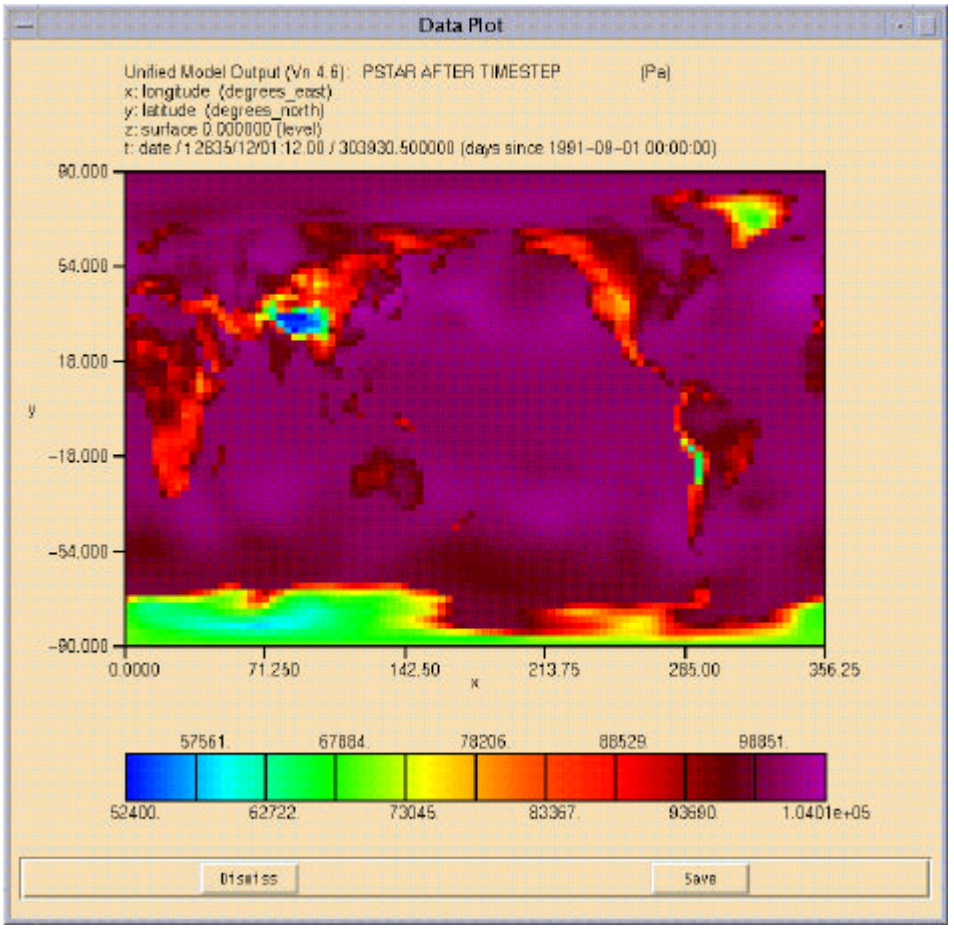
When the program has run for one model day (48 timesteps) you can view the results:

cd $HOME/PUM_Output/vn4.5/dataw.xaaqg

xconv -i xaaqga.pav5c20



Double left click on a field and then click "Plot data" on the upper right.



Pstar after the timestep.

6. Experiments to try

i) Change the number of processors.

cd $HOME/umui_jobs/xaaqg

Edit the file SUBMIT and change the lines

```
NMPPN=1           # N-S decomposition
NMPPE=2           # E-W decomposition
```

change the number of processors accordingly.
Do timing runs on your cluster 1,2,3 and 4 processors to get a feel for how the model scales across fast et

The start and end time for the model run are in $HOME/umui_out. Type ls -lrt to get the latest output file. Do that the timings are in minutes and seconds.

ii) Change the optimisations

cd $HOME/source

Edit the compile line in compile_model.

Run the model again (as above) to see what effect they have.

One you may want to try is:

-fast -tp p6 –Mvect=prefetch

iii) Compare the two runs above with that from a Myrinet cluster with 850 Athlon processors:

configuration	minutes/climate day	speedup
1x1	7.07	-
2x1	4.04	1.74
2x2	2.33	3.03
3x3	1.09	6.49
4x4	0.72	9.82

A graph might be instructive… Should you be plotting speedup or minutes/model day?

iv) Compare the results of your runs using cumf

Move the output file xaaqga.pav5c20 to one side and do another run with different optimisations or processor

Compare the results with the cumf untility:

cumf xaaqga.pav5c20 <original file>

Look at the difference map – any comments?

iv) Turn on profiling.

cd $HOME/source

Edit compile_model to use the Portland profiler. You may also need to recompile the gcom libraries...

Where does the model spend most of it's time? Have a look at the source code in $HOME/source/model

Could the model be made to run significantly faster?

(v) Look at the model using Vampir

Add the link option to the file makefile.link in $HOME/source/model and relink.

Re-run the code and look at the output using vampir.

(vi) Recompile using the Intel compiler

This is for the brave hearted only!

Try recompiling the model and gcom with the Intel compiler. Can you get the model to run?

Is it faster or slower than the Portland compiler for this code?