
Atmo. Sc.
Computing Resources
-
tips & tricks & examples

Bob Yantosca
Philippe Le Sager
Claire Carouge

OS = Linux

◆ Interactive Machines

- ◆ Tethys, Sol
- ◆ Pandora, Prometheus
- ◆ Argus

◆ Centos distro

- ◆ free RedHat
- ◆ good for server
- ◆ not that good for office

◆ Unix-like OS

- ◆ Interactive
- ◆ Multi-user
- ◆ Multi-tasking
- ◆ Mostly POSIX compliant

Software (1)

- ◆ Shell : main interaction
- ◆ Lots of utilities at the command line
 - ◆ for help : **man**, **info** (tab, u, n, p)
 - ◆ file manager
 - ◆ **cd**, **pwd**, **ls**, **mkdir**, **mv**, **rm**, **cp**, cat, touch
 - ◆ tar, zip/unzip, bzip2
 - ◆ process manager
 - ◆ top, ps, kill
 - ◆ internet (**ftp**, ssh, **wget**...) and your own program (./a.out)
 - ◆ and 100s more: **grep**, **find**, **sed**, **diff**, make, less,...

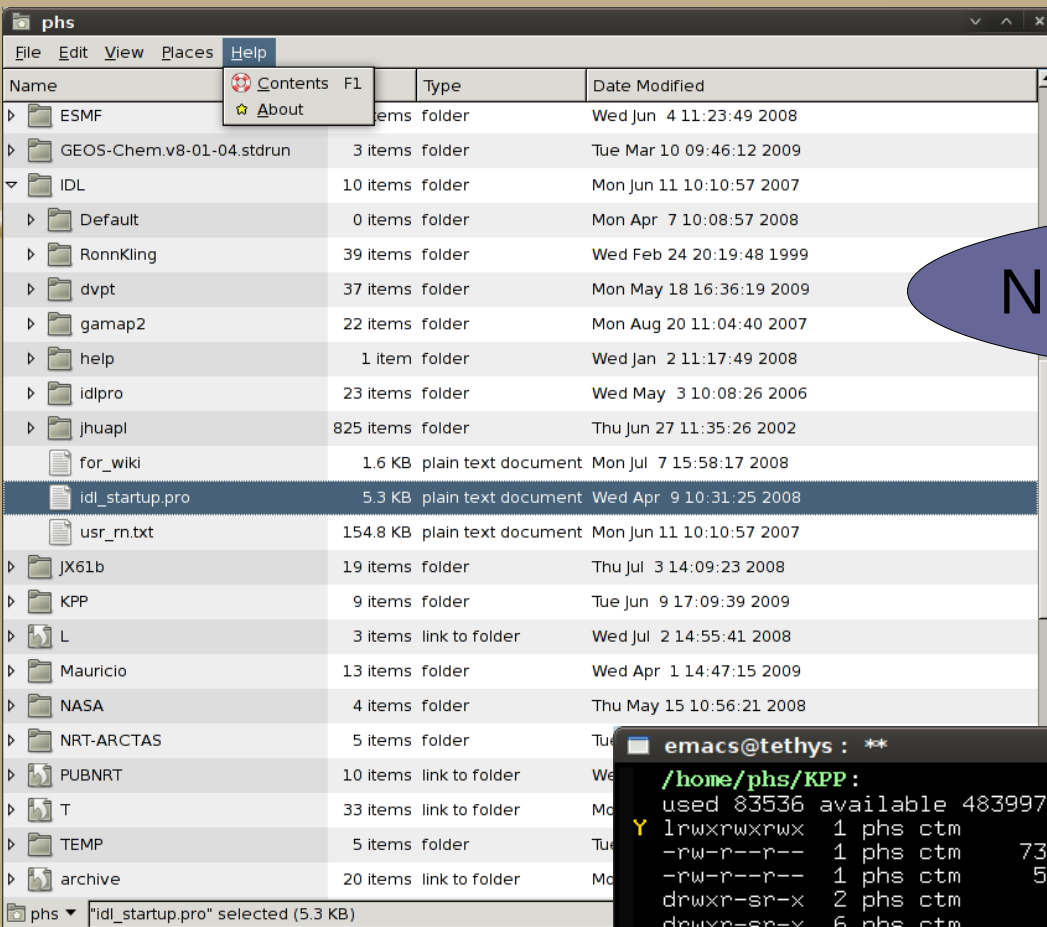
Software (2)

shell, Perl, python, IDL, R ...
=> command line is great to
experiment !

File management / system administration	<ul style="list-style-type: none">• shell (bash, tcsh)• perl, python, IDL (interpreted aka byte code)• GUI :: nautilus, OFM (a la Norton Commander), emacs (dir mode, nc.el)
text file editing	gedit, nedit, vim, emacs, nano
numeric programming	<ul style="list-style-type: none">• F95, C/C++ (compiled down to binary code = fast !)• Python (w/ NumPy)
data analysis	IDL, Splus, R, python, Matlab
imaging	convert, display, IDL, gimp (GUI)
office (doc, presentation)	latex (other options exist but you are better off on your local machine)
pdf, ps	Viewing : ghostview, gs Editing : ps2pdf converter, but vector graphic soft (illustrator, inkscape, ...) are not available

File Managers

Nautilus (from gnome desktop)



Emacs dired
(directory editor)

```
emacs@tethys: **
/home/phs/KPP:
used 83536 available 4839977968
Y lrwxrwxrwx 1 phs ctm      36 Jun 11 09:39 gckpp.kpp -> /home/phs/KPP/v8-02-01_43t/gckpp.kpp
-rw-r--r-- 1 phs ctm     73877 Jun  9 17:09 README
-rw-r--r-- 1 phs ctm     5569 Jun  9 17:08 for_kumaresh_paul
drwxr--sr-x 2 phs ctm      0 Jun  1 14:49 v8-02-01_54t
drwxr--sr-x 6 phs ctm      0 May 29 16:36 v8-02-01_43t
-rwxr--r-- 1 phs ctm    12594 Apr 20 12:11 geos2kpp_parser.pl
* -rw-r--r-- 1 phs ctm    36417 Apr 20 11:10 kpp_diff_Code.v7-04-10
-rw-r--r-- 1 phs ctm 84790219 Nov 20 2008 runs.tar.gz
drwxr--sr-x 8 phs ctm      0 Nov 19 2008 Code.v7-04-10
drwx--S--- 13 phs ctm      0 Nov 30 2006 kpp-2.2.1.December2006

/home/phs/KPP/v8-02-01_43t:
used 2624 available 4839977968
drwxr--sr-x 2 phs ctm      0 Jun  9 14:15 rosenbrock
drwxr--sr-x 2 phs ctm      0 Jun  3 15:47 lsode
-rw-r--r-- 1 phs ctm     115 Jun  3 15:47 gckpp.kpp
drwxr--sr-x 2 phs ctm      0 Jun  3 15:46 radau5
drwxr--sr-x 2 phs ctm      0 Jun  3 15:43 runge_kutta
-r--r--r-- 1 phs ctm 154032 May 29 15:07 globchem.dat
-r--r--r-- 1 phs ctm   2103 May 28 12:03 globchem.def
-r--r--r-- 1 phs ctm  32254 May 28 12:03 globchem.eqn
-r--r--r-- 1 phs ctm   4635 May 28 12:03 globchem.spc

/home/phs/KPP/v8-02-01_54t: ...

--:~* KPP Thu Jun 11 9:44AM (Dired by date Omit)--L24--C47--A11-----
```

Some of what's available

TETHYS

- ◆ FORTRAN
 - ◆ **gfortran, f90 & f95** (all link to gfortran)
 - ◆ **sunf90 & sunf95** (Sun Studio 12)
 - ◆ **ifort** (Intel Fortran 10)
- ◆ gs (pdf / ps)
- ◆ gedit, vim, emacs, nano
- ◆ **cvs**

SOL

- ◆ FORTRAN
 - ◆ **f95** (link to gfortran)
 - ◆ **gfortran** (GNU fortran 95)
- ◆ **ghostview**, gs (pdf / ps)
- ◆ gedit, **nedit**, vim, nano, emacs
- ◆ **IDL, Splus, R**

know your tools

- ◆ tutorial
- ◆ reference manuals
 - ◆ online/offline/book
 - ◆ better if **searchable**, let you **bookmark**
- ◆ user guides
- ◆ learning curve
 - a little bit everyday
 - understand what's going on, instead of simply applying recipes

fortran lfort 10 doc is here:

/opt/intel/fce/10.1.013/doc/Doc_Index.htm

Be Efficient (1)

teach yourself some tricks

- ◆ Shortcuts / history

aka there is a life after Ctrl-c and Ctrl-v
try “bindkey” at the command line

- ◆ Abbreviations / Completion

- ◆ Regexp

used everywhere (find, grep, sed,...)

- ◆ Macro

- ◆ Alias

alias gv "ghostview"

- ◆ Cheat sheet

Be Efficient (2)

bolster reuse

When writing a program

- ◆ Document
 - code (header, comments)
 - directory (include a README file)
- ◆ versatile (keywords in F90 and IDL)
- ◆ meaningful names
- ◆ several subroutines ..instead of one large prgm

“Geos-chem is really to be applauded for the really helpful webpage and the wiki. Then there's the way every Gamap procedure and every geos-chem module is commented and then commented again. I get a lot of mileage out of just reading the programs, and it cuts down on a lot of questions to y'all.”
a GC user

Miscellaneous

◆ Setting multiproc

setenv OMP_NUM_THREADS 2 (b/w 1-4 on tethys)

◆ or, in your code:

use omp_lib

call omp_set_num_threads(2)

◆ Using CVS

Set the local root directory for CVS version control

setenv CVSROOT /lustre/home/bmy/CVS

cvs co -d <dir-u-want> geos-chem -> to get the last version

cd <dir-u-want> && cvs update -> to update to the last version

TCSH SCRIPTS

All scripts examples are in `~phs/gforum`

In brief

- ◆ **Redirection:**

cmd | cmd2 to direct output of cmd to cmd2

cmd > file to direct output of cmd to file

>>, <<

- ◆ **Quotation**

“ and ' and \

- ◆ **Substitution**

` (back quote)

- ◆ **Variables**

\$var_name, \${var_name}

- ◆ **Script arguments**

\$1, \$2, ...

\$* (all arguments)

\$# (number of arguments)

Basic idea

to specify the log file and
rename the BPCH file (if it exists)

```
#!/bin/tcsh -f

##   Time-stamp: <2009-06-01 12:55:57 phs>

####-----
##   this is a simple script to run geos
####-----

set cmd=./geos

# ----- run
time $cmd >! ./output/logEXT

# ----- put the same extension if the output file is the generic ctm.bpch
if ( -e ctm.bpch ) mv ctm.bpch ./output/ctm.bpch.EXT

exit(0)
```

```
--:-- run_script_template Mon Jun 8 4:02PM (Shell-script[tcsh])--L10--C46--
```

Idea : pass one argument to specify the MONTH
 It creates a unique script (unique log and bpch with
 month nb) and unique INPUT.GEOS

```

LATION v8-02-01: GEOS-3-4-5 43 tracers
-----
MENU %%% :
, HMMSS : 11111111 111111
, HMMSS : 22222222 222222
: ./
file : ./restart.YYYYMMDDhh
rt file? : T
file(s) : ./restart.YYYYMMDDhh
ctory : /as/data/geos/GEOS_4x5/
subdir : AGRID/YYYY/MM/
subdir : GEOS_3/YYYY/MM/
subdir : GEOS_4_v4/YYYY/MM/
subdir : GEOS_5/YYYY/MM/
ssions etc: /as/data/geos/GEOS_1x1/
ctory : /home/phs/TEMP/
ls? : F
ields? : F
ropopause? : T
I0, J0 : 0 0
-----
J %%% :
tion : 3
ers : 43
-----> : TR# Name g/mole Tracer Members; () = emitted
: 1 NOx 46.0 NO2 (NO) NO3 HNO2
: 2 Ox 48.0 (O3) NO2 2NO3
: 3 PAN 121.0
: 4 CO 28.0 (CO)
: 5 ALK4 12.0 (4C)
: 6 ISOP 12.0 (5C)
: 7 HNO3 63.0 (HNO3)
: 8 H2O2 34.0
: 9 ACET 12.0 (3C)
: 10 MEK 12.0 (4C)
: 11 ALD2 12.0 (2C)
: 12 RCHO 58.0
: 13 MVK 70.0
: 14 MACR 70.0
: 15 PMN 147.0
: 16 PPN 135.0
: 17 R4N2 119.0
: 18 PRPE 12.0 (3C)

```

```

# CHECK that a month number is passed
if ($# == 0) then
    echo 'you must pass a month number (1-12)'
    exit(1)
endif

# DEFINE Date variables from input (month number)
# Shows simple calculations.
set Y1=2004
set Y2=$Y1

set M1=$1
@ M2= $1 + 1

if ($M1 == 12) then
    @ Y2++
    set M2=1
endif

if ($M1 <= 9) set M1=0`echo $M1`
if ($M2 <= 9) set M2=0`echo $M2`

# Modify input.geos accordingly
# Multiple calls to sed
sed -e "s:11111111 111111:${Y1}${M1}01 000000:" \
    -e "s:22222222 222222:${Y2}${M2}01 000000:" \
    input.geos_template > input.geos

# MODIFY run script (to get unique log file)
sed s/EXT/"$M1"/ run_script_template > runscript

# CHECK that restart file exist
set RESTART=$rundir/restart.${Y1}${M1}0100

if ( -e $RESTART == 0) then
    echo 'No restart file'
    exit(1)
endif

# RUN with SUBMIT
submit smp4 runscript

exit(0)

```

#!/bin/tcsh -f

Time-stamp: <2009-06-08 15:27:39 phs>

```
#####-----
## this is a script to run geos, and get unique outputs.
## Unique log and ctm.bpch name is done with appended extension.
## EXTension is script argument or DDD_hhmmss if no argument is passed.
#####-----
```

```
# ---- executable
set cmd=./geos
```

```
# ---- (re)set a dummy test file for timestamp
set testF=testFile
touch $testF
```

```
# ---- extension to singularize outputs (log and bpch)
set EXT=`date +%j`_`date +%k%M%S`
if($#argv > 0) set EXT=$1
```

```
# ----- run
time $cmd >! ./output/log$EXT
```

```
# ----- rename bpch output (version 1 - only ctm.bpch)
# if ( -e ctm.bpch ) mv ctm.bpch ./output/ctm.bpch.$EXT
```

```
# ----- rename bpch output (version 2 - most recent *bpch*)
# useful if bpch output is not ctm.bpch
set bpchfile=`find . -name '*bpch*' -newer ${testF} -print`
```

```
if ( ${#bpchfile} != 0 ) then
mv $bpchfile ./output/${bpchfile}$EXT
endif
```

^ run script

Main script >>>

Idea : pass 1 or 2 argument.

First one is passed to the run script to (have unique output names)

The second one set the number of processors

```
##
## "anythingelse" -> will be used as an extension for the log and
## ctm.bpch file (any *bpch*)
##
## No argument: Default outputs (log & ctm.bpch) have
## ----- unique extension DDD_hhmmss.
##
## 2nd argument: number of processors.
## -----
##
## Examples
## =====
## CNR clean # make clean only
##
## CNR # compile and run on 4 proc, w/ current date appended to outp
## CNR test1 # compile and run on 4 proc, w/ "test1" appended to outputs
## CNR test2 2 # compile and run on 2 proc, w/ "test2" appended to outputs
##
#####-----
```

```
# Define Run and Code Directories
set rundir=`pwd`
set codedir=${HOME}/`(basename $rundir | sed 's:run:Code:')`
```

```
# ----- compilation
cd $codedir

if ( $1 == "clean" ) then
build clean
exit(0)
endif
```

```
#build
make -f Makefile.ifort
```

```
# ----- move exe into run dir
mv geos $rundir
```

```
# check if compilation went fine (the check is done on mv success
# instead of build )
if ( $? != 0 ) then
echo "error in compilation"
exit(1)
endif
```

```
echo "compilation done"
```

```
# ----- run
cd $rundir

# nb of processors (deaful=4)
set np=4
if ($#argv >= 2) set np=$2

setenv OMP_NUM_THREADS $np
```

```
# pass 1st argument to script justrun
justrun $1
```

```
exit(0)
□
```

```

# ----- Define Run and Code Directories
#
# here we assume that the Code directory is in your home: ~/Code.<name>
# and this script is in the run directory <path>/run.<name>
#
# Shows: variable definition, substitution, pipe, env variable, sed,
set rundir=`pwd`
set codedir=${HOME}/`(basename $rundir | sed 's:run:Code:')`

# ----- compilation
cd $codedir

# ----- Clean & exit
foreach f ($*)
  if ( $f == "cleanonly" ) then
    build clean
    exit(0)
  endif
end

# ----- Clean & Compile
foreach f ($*)
  if ( $f == "clean" ) build clean
end

build

# ----- move exe into run dir
mv geos $rundir

# check if compilation went fine (the check is done on mv success
# instead of build, which returns 0 even when it fails )
if ( $? != 0) then
  echo "error in compilation"
  exit(1)
endif

echo "compilation done"

# ----- run
cd $rundir

```

Idea : pass one argument to a sed command to create a unique script (unique log and bpch) that we can submit.

```

#-----
# IDEA for a call to sed, to manipulate RUNSCRIPT
#-----
# MODIFY run script (to get unique log file)
sed s/EXT/"$1"/g run_script_template > runscript

# make it executable
chmod 744 runscript
#-----

submit smp4 runscript

exit(0)

```

```
--:-- CNS Mon Jun 8 3:59PM (Shell-script[
```

```

#-----
--:-- CNS Mon Jun 8 3:59PM (Shell-script[tcsh])--L37--C0-1, 2009

```