Command line tips for Linux, OSX and Cygwin users of Esp-r

Dr. Jon W. Hand, ESRU
13 June 2011

Introduction

Graphic environments are great until you want to do something that isn't included. If you are using ESP-r within a Cygwin environment on a Windows or an Apple computer (or even a Linux computer) it is occassionally useful to put on a geek hat and do some magic on the command line.

Another reason for going geek is to sort out problems in your models of the 'it was working a few minutes ago' variety or the 'Fred sent me a model and I can't get it working' variety.

And sometimes a bit of geek magic allows you automate tasks - e.g. create scripts that run a dozen assessments and extract specific performane data.

And sometimes a remote IT support or simulation expert will be using a lot of jargon and asking you to type in commands for them. What the expert thinks is a simple statement like "in the command window go to the folder with the model configuration file" is loaded with implied knowledge that you might not have. And thus a 15 second task takes a lot longer or causes a lot of frustration. Wouldn't it be great to understand a bit of that jargon and more easily follow their instructions?

And sometimes you will want to archive models or find the difference in today's model and one from a week ago. Folk with an IT background know this stuff but if it becomes your job to manage projects this primer includes some of those IT skills.

And lastly, under the user interface is an industrial strength engine for virtual physics which can be coerced far beyond its normal bounds via the command line.

A command line primer

The command line facilties within Cygwin, OSX and Linux are similar so skills tend to be transferable. A subset of commands are equivalent to graphical interactions e.g. navigation within the folders of the computer, managing files, searching for text, archiving work, transferring models to colleagues and the like. This primer should cover most of what normal users of ESP-r need to know. Those who maintain the system and work with the source code will of course, need additional skills and should look into the ESP-r Developers Guide (see the ESRU web pages) as well as books on Unix and Linux.

Lets start with some high level concepts:

- The commands that will be described below are interpreted via a command interpretors (also known as a command shell). One interpretor is known as the bash shell and another is 'csh'. Most commands are interpreted the same way in both command shells although some of the optional parameters are slightly different.
- The command shell runs in either a command or graphic window. A command window only works with text based tasks and a graphic window can work with either text or graphic applications. For example:

  Cygwin
  : The inital command window (typiclly white text on a black background with a fixed width) that starts up supports text commands and can only be used to invoke ESP-r in a pure-text mode (for example to run automated tasks). If you try to invoke ESP-r in graphic mode from the initial command window nothing will happen or you will get a warning about cannot open display. To work with the graphic version of ESP-r you need to startup a graphic window via a command like:
    startxwin.sh
    which will start a graphic window (black text on a white background and with an X in the top border). You can resize the graphic window if you want. If you want you can have multiple graphic windows open at the same time. When you are finished close the graphic window (via the command exit). Then you can type exit in the original command window.

  OSX 10.5 and 10.6
  : OSX includes a default 'terminal' which can take either text or graphic commands (??). There is also an

X11 graphic window (typically called an xterm) which is available as part of the X11 environment. As X11 is required for ESP-r use you have a choice. X11 supports networked displays (where what you see in the xterm could be an application running on a different computer). This feature can be useful for accessing a compute server as well as for IT support staff to access your computer.

Linux

On linux machines the overall graphic environment is dictated by whether the computer is using KDE or Gnome. Each come with a different set of tools and several choices for graphic windows. As with Cygwin and OSX you can have multiple graphic windows running at the same time as well as networked displays.

- You will have a login account and an area to work in which is protected from the chaos of other users. On Linux and OSX it is possible to have multiple users working simultaneously on the computer. Sometimes it is useful to have an alternative user account for testing so that your normal working area is not altered. Cygwin is dependent on Windows for the user account so one must switch users but may not have two users simultaneously. Login accound names might cause problems if it includes spaces or unprintable characters.
- The permissions associated with files and folders can help as well as hinder your work. Files you get from another person might not have the correct permissions for use in your account and you might find that changes you make in your model might not be written to the model files. Commands to identify and correct permissions is thus a common task.

  Cygwin

  The underlying Windows platform supports a different concept of permissions for folders and files than OSX and Linux. For example is easier to overwrite and corrupt critical databases in Cygwin than it is in OSX or Linux. To change permissions set by another user one must switch to that user.

  OSX and Linux

  Since OSX and Linux are derived from Unix, both able to set permissions on files and folders at find resolution. Both support the concept of 'the current user', the group that the user is in, and 'others'. It is possible ensure that only specific staff are able to update databases. It is possible to set read and write permissions separately for the user, group and others. To change perissions set by another user one would typically use a sudo prefix to a command to temporarily become that user.
- Cygwin and OSX and Linux use a hierarchical directory structure beginning at "/" (equivalent to c: on a PC) which is the so called 'root' directory and everything else hangs off of that. There is no A:B:C:D:E: drive.

  Cygwin

  The Cygwin command window you will be able to navigate to folders in the A:B:C:... drives. The path within Cygwin becomes /cygdrive/c or /cygdrive/d. Note the use of / rather than \ when working within the Cygwin environment.
- Paths to files which are included in an ESP-r model are sensitive to the operating system they were created with. A model created with a 'native windows' version of ESP-r will include paths in the form C:\Esru\databases while ESP-r models created on Linux will follow the path pattern /home/fred/databases and on OSX will follow the pattern /Users/fred/databases.
  Path conventions thus complicate the task of moving a model between computers with different operating systems - a C:\Esru path will tend to confuse ESP-r running on a different type of computer. There are scripts which can convert models which you need to know about.
- Your HOME folder on Linux might be in /home/jackson (if you name is jackson). On OSX user folders would typically follow the pattern of /Users/jackson. Within Cygwin the users home tends to be C:\Documents and Settings\user_name although when setting up Cyginw there will be a /home/user_name.
- Case sensitivity differs in OSX and Linux and Cygwin:

  Cygwin

  Cyginw is not case sensitive for the names of files and folders (for a folder namde /home/fred/Src the commands:
  cd /home/fred/src and cd /home/fred/Src are equivalent.

  OSX

  By default OSX is not case sensitive. OSX drives can be formatted to support case sensitivity. This tends not to be a problem unles working with a model which was created on Linux which might have Readme.txt and readme.txt as separate files.

  Linux

  Linux is case sensitive. A folder named Src is different from a folder named src which is different from a folder named SRC.

- The ESP-r executables and databases and example models tend to be installed in standard locations for Cyginw (/usr/esru), OSX (/opt/esru), Linux (/opt/esru or /usr/esru). The layout of the ESP-r distribution is the same in each so the navigation skills needed to investigate the ESP-r distribution are common.

Examples

The following discussion will help you gain a basic understanding and serve as a reference. Items in <> would be replaced by your own text. Items in () are comments.

Commands tend to be terse - "cp" for copy, "mv" for move etc. Each has a dozen options so that copies can be recursive, file listings can be brief, extensive, sorted by some key. Those that remember the DOS command line will find the facilities are much richer (but also sensitive to capital and lower case).

It is possible to find out more about a specific command by giving the commands in the following form:

man <command name>

man ls

NAME
ls -- list directory contents

SYNOPSIS
ls [-ABCFGHLOPRSTUW@abcdefghiklmnopqrstuwx1] [file ...]

DESCRIPTION
For each operand that names a file of a type other than directory, ls displays its name as well as any requested, associated information. For each operand that names a file of type directory, ls displays the names of files contained within that directory, as well as any requested, associated information.

. . .
The following options are available:

-1 (The numeric digit ‘‘one’’.) Force output to be one entry per line. This is the default when output is not to a terminal.

-A List all entries except for . and ... Always set for the super-user.

. . .

So you can see that the command to list a directory has more than a dozen options. Over time you will identify which options are useful to you.

Moving around:

Given the following directories and files:

/ (root)
/home/jackson (the home of jackson)
/project_a (folder for a project)
/memos (folder for memos)
/fax_121196_gary, fax_1106_mark, fax

/email (folder for emails)

/usr/esru (the home of esp-r)
/opt/esru (alternative home of esp-r)
/esp-r
/bin (where esp-r executables live)
/bps, prj res (esp-r modules)

/training (where training models live)
/simple
/cfg
/bld_simple.cfg

When you log in you will be in your home folder. If you get confused about where you happen to be at any time then give the command

pwd

(print working folder) which will return information like: /home/jackson/memos

To move around use the command "cd <folder_name>" to go there. Ensure there is a space after the command and before the argument used by the command. To move to the set of folders used by esp-r use the command

cd /usr/esru/esp-r

The <folder_name> can be relative - in which case ".." means go up a level. To move to the folder above give the command

cd ..

The <folder_name> can also be absolute - so to go into the folder of esp-r executables (from your current location) to see how big they are you can use the command

cd /usr/esru/esp-r/bin

If no folder_name is given then you are returned to your home folder:

cd

The use of absolute paths within an ESP-r model can cause problems if you need to pass the model to a colleague. For example, if your model references the database
/home/fred/databases/fred-materials.db
and you send the model to a colleague who does not have a **/home/fred** on their machine then the model files would need to be edited to stipulate an alternative location for fred-materials.db.

Copying files:

In your home folder structure, say you wanted to copy a file (say fax_121109_gary) for use as the basis for a new fax.

cp fax_121109_gary fax_050210

the syntax is cp <soure> <destination>

If, while in your home folder you want to copy a model file in /usr/esru/esp-r/training/simple to your home folder for printing or editing then

cp /usr/esru/esp-r/training/simple/cfg/bld_simple.cfg .

in this case the "." is short for "where I am now". Because no name was given in the destination the copied file will have the same name as the source. If you wanted to change the name as you copied it you use the command:

cp /usr/esru/esp-r/training/simple/cfg/bld_simple.cfg ralph.cfg

You could also change the name by the command:

mv bld_simple.cfg ralph.cfg

where the syntax is mv <source> <destination>

Viewing files:

To view the contents of a text file, one option is to use the command:

more bld_simple.cfg

where the syntax is more <file name>

If the displayed text does not fit within the display then more will pause before displaying further pages of text.

Creating directories:

In the above folder structure, say you wanted to begin a new project. There is already a folder called project-a and if you want to create a project-b at the same level give the following commands:

cd /home/jackson
mkdir project-b

you can then move into this new folder via the command:

cd project-b

Finding out what is in a folder:

The normal way to find out the contents of a folder is to use the "ls" command. For example a listing of the contents of "memos" is

```
ls memos
Beattie              home_off_cover2       roosvlt_280196
bepac_lmia           home_off_cover3       roosvlt_280196.ps
home_off_cover       pcs210394
lambda: jon/textfR
```

Unless you know what these names represent you might want to find out more. A long listing is:

```
ls -l memos
total 44
-rw-r--r--  1 jon         1990 May 24  2003 Beattie
-rw-r--r--  1 jon         1665 May 30  2005 bepac_lmia
-rw-r--r--  1 jon         3102 Mar 12  2003 home_off_cover
-rw-r--r--  1 jon         4078 Feb  5  2004 home_off_cover2
-rw-r--r--  1 jon         4163 Mar 27  2005 home_off_cover3
-rw-r--r--  1 jon         2133 Nov 30  2004 pcs210394
-rw-r--r--  1 jon         1678 Oct 28 14:04 roosvlt_280196
-rw-r--r--  1 jon        22035 Oct 28 14:05 roosvlt_280196.ps
```

The "-rw-r--r--" represents the permissions and file types. In this case the user has read & write permission and the "group" of the user and "others" can read these files but they cannot modify them. The "-" in the first column denotes a file. The owner of the file comes next and then the size of the file, a date stamp and then the name.

Lets say we want to prevent others from reading the contents of the file home_off_cover2 but we want to allow people in our group to write to the file we could do this via the commands:

chmod o-r home_off_cover2
chmod g+w home_off_cover2

```
ls -l memos
total 44
-rw-r--r--  1 jon         1990 May 24  2003 Beattie
-rw-r--r--  1 jon         1665 May 30  2005 bepac_lmia
-rw-r--r--  1 jon         3102 Mar 12  2003 home_off_cover
-rw-rw----  1 jon         4078 Feb  5  2004 home_off_cover2
-rw-r--r--  1 jon         4163 Mar 27  2005 home_off_cover3
-rw-r--r--  1 jon         2133 Nov 30  2004 pcs210394
-rw-r--r--  1 jon         1678 Oct 28 14:04 roosvlt_280196
-rw-r--r--  1 jon        22035 Oct 28 14:05 roosvlt_280196.ps
```

The above commands would work in OSX and Linux and Cygwin. If someone else owned the files you would need to preface the command with a 'sudo' in OSX and Linux. In Cygwin you would have to switch users to change the permissions.

A common problem with permissions is for so-called scripts which can automate tasks. If you copy a script or get it via an email transfer the execute permissions may get lost. When this happens the script does not run. You might get a message about permissions and you might get no response at all. To restore the permissions you would need to give a command like:

chmod a+x myscript.sh

Finding executables

Most operating systems include options for searching within a disk drive for specific files but that can take a long time to run. Sometimes it is useful to quickly find out where your computer operating system thinks is the location of an executable (such as one of the ESP-r executables). An IT support person might ask you about this. For Cygwin and OSX and Linux there is the **which** command. Lets look for the ESP-r simulator via the command:

which bps
/usr/esru/esp-r/bin/bps

If you have used the link_to script to set a short-cut to the ESP-r distribution you might get the following:

which bps
/home/fred/bin/bps

which is a link. In which case if you really want to locate the actual simulator you would need to give a further command:

ls -l /home/fred/bin/bps

The **which** command makes use of the PATH information in your account. If there is a response to the command then that indicates that the ESP-r module has been located in one of the standard paths known to your account. If there is **no** response then it could indicated that your environment variables have not been correctly setup.

If **which** did not work then on Cygwin and OSX and Linux you could try an alternative file search technique via the **find** command. The general form of find is:

find . -name <name_of_file> -print

find . -name bps.exe -print

find . -name "*.cfg" -print

In the above the dot after the find command is equivalent to my-current-location (and implicitly all of the sub-folders within my-current-location). The -print is required so that you can see the result of the search. The -name command indicates that the next token in the command line will indicate the name of the file. Use double quotes if there are spaces in the file name or if you are going to use a wild-card. Thus the last command above will report on all files which end in .cfg within all sub-folders of the current location.

Finding what is in files:

Often we are confronted by the need to locate a topic or key word within a file. There is a command line tool named 'grep' which is designed to do this. As an example if I wanted to quickly check the location of the word fred in a text file the command response would be:

grep -ni fred esp_command_line_hints.text
17:the 'Fred sent me a model and I can't get it working' variety.
153:path pattern /home/fred/databases and on OSX will follow
154:the pattern /Users/fred/databases.
171:folders (for a folder namde /home/fred/Src the commands:
173:cd /home/fred/src and cd /home/fred/Src are equivalent.

315: /home/fred/databases/fred-materials.db
319:have a /home/fred on their machine then
321:an alternative location for fred-materials.db.

The option -n requests the line number and the -i says ignore the case (Fred = fred). Grep can be used to search multiple files and folder structures.

Editing files

ESP-r models are largely composed of ASCII text files. We recommend your first choice for manipulating ESP-r models is via the ESP-r interface. There are times when direct editing of files is required so *if you must edit* then do so with tools which are least likely to corrupt your model files.

Firstly, ESP-r model files may be sensitive to formatting rules so if you use an editor that automatically warps text lines then you may corrupt the file. Second ESP-r model files must maintain the so-called line ending characters that they started out with.

Cygwin
: There are several editors which can run within Cygwin via one of the graphic command windows - nedit and nano are examples of text editors that can be installed within Cygwin.
Cywin lives within the Windows file structure and you can also invoke editors from Windows. Avoid Word or Notepad for edit ESP-r model files. A great free editor is Notepad++ (easily found on the web). WordPad can be used with care.

OSX
: If you used the Fink or MacPorts environments when you setup your computer for ESP-r use then you can install one of the standard Linux editors (nedit or medit) for use with ESP-r files. The (not free) BBedit editor works well with ESP-r model files.

Linux
: There are dozens of text editors available on Linux. Almost any of them can be used with ESP-r model files.

Settings within text editors: a) turn off line warping, b) avoid tab characters in ESP-r model files, c) generally stick with Unix line endings on OSX, Linux and Cygwin.

Finding differences in files:

Many people assume that determining the differences between text files (e.g. model files or reports) is a tedius process. We have many options beginning with a command line tool called 'diff'. If there are two model configration files and I want to see the differences:

```
 diff bld_simple.cfg  bld_simple_fzy.cfg
3,5c3,5
< # bld_simple.cfg
< *date Fri Jul 23 16:07:40 2010  # latest file modification
< *root bld_simple
---
> # bld_simple_fzy.cfg
> *date Fri Jul 23 16:10:41 2010  # latest file modification
> *root bld_simple_fzy
28c28
< *ctl  ../ctl/bld_simple.ctl
---
> *ctl  ../ctl/bld_simple_fzy.ctl
63c63,65
< *imdoc no documentation (yet) for this image
---
> *imdoc there are 4 control periods in the day
> *img GIF   FCTL  ../images/fzzy_montg.gif
> *imdoc figures indicate the control via fuzzy logic
73c75
< *sps    1   2   2  10   4   0
---
```

```
> *sps    1   2   1  10   2   0
75c77
< *sblr bld_simple.res
---
> *sblr bld_simple_fzy.res
96c98
< bld_simple.log
---
> bld_simple_fzy.log
98c100
< L-shaped reception, convective heating, ideal control
---
> L-shaped reception, convective heating, fuzzy control
106c108
< *cnn bld_simple.cnn  # connections
---
> *cnn bld_simple_fzy.cnn  # connections
bash-3.2$
```

Lines with < are the first file and > signals the second file in the original command. The tool diff and its variants are available on all the platforms.

OSX
: OSX offers a comparison tool as part of the XCode source code development environment. It is used via: opendiff <file name> <file name>

Linux
: Depending on whether you are using Gnome or KDE there will be a slightly different choice of file differencing utilities to choose from. kompare shows a side-by-side graphic view of file differences. It also is capable of copying specific differences between files.

One of the most valuable uses for file differencing tools is for looking at two different model contents reports!

Removing files:

Sometimes you want to get rid of a file. Unlike other operating systems, when you do this you REALLY delete it. The syntax is:

rm <file_name>

Remove has lots of powerful options which experts occassionally use. The operating system also provides facilities for archiving your work (see next discussion).

Making backups of files and/or directories:

Anyone who has lost half a weeks work by a disk crash or one of "rm's" more powerful options has an interest in archiving their work. Linux and OSX and Cygwin provide a simple way to save your projects similar to how one might use zip to create archives on a PC. It is called "tar". Tar creates an archive file out of the contents of many files or folders and sub-folders. Say I wanted to archive the memos folder I would give the following command:

```
lambda: jon/text% tar  cvf  memos_030297.tar  memos
memos/
memos/pcs210394
memos/Beattie
memos/home_off_cover
memos/home_off_cover2
memos/roosvlt_280199
memos/home_off_cover3
memos/bepac_lmia
memos/roosvlt_280199.ps
lambda: jon/text%
```

This says create verbosely the file memos_030297.tar with the contents of the folder memos. The resulting file:

```
 ls -l memos_030297.tar
-rw-r--r--  1 jon        51200 Feb  4 19:16 memos_030297.tar
```

To find out what it contains I give the command:

```
lambda: jon/text% tar  tvf  memos_030297.tar
drwxr-xr-x1002/10       0 Oct 28 14:05 2009 memos/
-rw-r--r--1002/10    2133 Nov 30 11:14 2004 memos/pcs210394
-rw-r--r--1002/10    1990 May 24 11:57 2003 memos/Beattie
-rw-r--r--1002/10    3102 Mar 12 14:10 2003 memos/home_off_cover
-rw-r--r--1002/10    4078 Feb  5 14:56 2004 memos/home_off_cover2
-rw-r--r--1002/10    1678 Oct 28 14:04 2009 memos/roosvlt_280199
-rw-r--r--1002/10    4163 Mar 27 19:26 2005 memos/home_off_cover3
-rw-r--r--1002/10    1665 May 30 09:39 2005 memos/bepac_lmia
-rw-r--r--1002/10   22035 Oct 28 14:05 2009 memos/roosvlt_280199.pfR
```

which says tell me (verbosely) the contents of memos_030297.tar.

Perhaps I managed to corrupt the file roosvlt_280196.ps. I can recover it via the command:

```
 tar  xvf  memos_030297.tar  memos/roosvlt_280196.ps
```

which say extract (verbosely) from the archive memos_030297.tar the specific file memos/roosvlt_280196.ps.

This tar archive, being a single file is easy to send to others. A typical use is to create a "tar" file of a project, have someone halfway around the world pick it up and then expand the project to recreate exactly the same structure of project that you have.

Compressing files:

Sometimes you have work or simulation results which you will not be needing for a few days. You can compress files with the command gzip:

```
 gzip  <file_name>
```

which adds a ".gz" to the end of the file and makes it 20-80% smaller. To reverse the process use the command:

```
 gzip  -d  <compressed_file_name>
```

For example the "tar" file of the memos folder can be compressed:

```
lambda: jon/text% gzip -9 memos_030297.tar
lambda: jon/text% ls -l memos_030297.tar.gz
-rw-r--r--  1 jon        21290 Feb  4 19:16 memos_030297.tar.gz
```

the -9 directs gzip to take more time for greater compression.

On Cygwin you could also use a native Windows compression utility on files which live within the C:using the 7z utility (it is free). OSX allows you to right-click with a compression option.

Creating archives and compressing them is useful in archiving models. Disks die, files get removed by accident and thus model archives are part of good working procedures. What we include in our archives is a policy decision. For example, simulation results files can take up many hundreds of megabytes even in a compressed form. Some groups prefer to archive results files and others prefer to archive the instructions about how to recreate the result file.

Model transfer

The transfer of models within a team that is geographically dispersed can be made simpler if the procedure is regularized and models are designed for transport. Here are a few suggestions:

a) Ensure that documentation and images associated with the model are located in the doc and images folders of the model.

b) Ensure that project specific databases are included in the model dbs folder and include a readme file which identifies the version of the standard databases that are assumed to exist on the source and recipient

computers.

c) Include model contents reports with the model so that ther recipient can quickly scan the contents of the model without needing to invoke ESP-r modules.

If you are using Cygwin or OSX or Linux the ESP-r project manager includes a facility to 'archive current model' which scans the model configuration file and creates a tar file with the referenced model files. This is particularly useful if a model folder structure includes multiple model variants - only the referenced files are included in the archive.

If the resulting archive file is not large then you might choose to send it to others as-is. If you compress it then make sure that the person you are sending it to has the necessary decompression software.

If someone sends you a model as a 'zip'file then it was probably created on a Windows computer. If you are running Cygwin or Linux or OSX you can deal with this file on the command line via the unzip command:

unzip <the_file_name>

Automation of performance data extraction

One of the secrets of ESP-r experts is their ability to automate tasks. ESP-r modules on Cygwin and OSX and Linux can be invoked in text as well as graphic mode. Text mode opeation allows user commands to be stored in a file and used to drive ESP-r modules.

For example, if you have a dozen completed simulations and you want to extract statistics on a number of topics (zone dry builb temperatures, zone resultant temperatures, zone sensible and latent plant used during the simulation and the sensible capacity during the simulation period) in the same reporting format - there are several approaches.

You could run the results analysis application a dozen times to generate the reports. Or you could run the results analysis tool once and keep a careful record of the keystrokes you used and then use this to quickly generate the remaining reports.

The text mode and X11 graphic mode imvocations of ESP-r listen to the keyboard as well as the mouse position. Below are the key strokes recorded during the initial res session and enbedded into a script file. The 1st line ensures the csh command interpreter is used. The 2nd line assigns the file name used with the script to a local variable RESFILE. The 3rd line invokes the results analysis module with the file, forces it to run in text mode and the **<XXX** tells res to take its input from the following lines until it reaches the characters XXX. The last line of the script has the characters XXX.

Once the ESP-r module starts the initial character of the command is used. For example in the 5th line the characters 'select zone' are a comment.

```
#!/bin/csh -fb
set RESFILE=$1
res -file $RESFILE -mode text<<XXX

4  select zones
b
c
d
e
f
g
h
i
-
d  enquire
>
$1.txt
$1 summary
a  summary stats
b
a  dry bulb
b
e  resultant
```

```
-
f  energy delivered
a  stats
h  heating
e  sensible
-
>  close output
-
-
XXX
```

If the script name is extract.sh then it could be invoked via commands like (the 2nd token is the name of the results library:

```
./extract.sh holmes_aut_dg.res
./extract.sh holmes_aut_dl.res
./extract.sh holmes_aut_gf.res
```

The 1st report generated by the first of the commands is included below to illustrate the types of information that can be recovered.

```
# Lib: holmes_aut.res: Results for holmes_b
# Period: Mon-22-Sep@00h07(1986) to Sun-28-Sep@23h52(1986) : sim@15m, output@15m
# Zone db temperature (degC)
Description Max_value Max_occur Min_value Min_occur Ave_value Std_dev
kiten_util  22.336 25-Sep@17h07  16.147 22-Sep@05h52 19.953    1.4290
bedroom1    21.247 24-Sep@08h22  15.613 24-Sep@05h52 19.822    1.4725
bath        21.541 25-Sep@16h37  15.999 22-Sep@05h52 19.507    1.3050
bedroom2    21.178 24-Sep@08h52  15.826 24-Sep@05h52 19.096    1.2503
living      21.348 24-Sep@08h22  15.000 22-Sep@03h52 19.512    1.9426
hall        21.122 22-Sep@07h22  16.697 24-Sep@05h52 19.957    1.1262
study       22.342 24-Sep@19h37  15.671 22-Sep@05h52 19.974    1.5202
family      21.212 27-Sep@19h22  15.555 22-Sep@05h52 19.709    1.5169

   All      22.342      --       15.000      --      19.691       --

# Lib: holmes_aut.res: Results for holmes_b
# Period: Mon-22-Sep@00h07(1986) to Sun-28-Sep@23h52(1986) : sim@15m, output@15m
# Resultant temperature (degC)
Description Max_value Max_occur Min_value Min_occur Ave_value Std_dev
kiten_util  22.363 25-Sep@17h07 17.317 22-Sep@05h52 20.028    1.0042
bedroom1    21.281 25-Sep@08h52 16.833 24-Sep@05h52 19.925    1.0533
bath        21.359 25-Sep@16h37 16.801 22-Sep@05h52 19.636    0.99441
bedroom2    20.818 25-Sep@08h52 16.853 22-Sep@05h52 19.284    0.79845
living      21.155 25-Sep@08h52 15.781 22-Sep@05h52 19.413    1.4278
hall        20.800 25-Sep@18h37 17.514 22-Sep@05h52 19.892    0.73674
study       21.623 24-Sep@19h37 16.449 22-Sep@05h52 19.764    1.1073
family      20.523 24-Sep@19h37 16.571 22-Sep@05h52 19.505    0.96236

   All      22.363      --      15.781      --      19.681       --
Lib: holmes_aut.res: Results for holmes_b
Period: Mon-22-Sep@00h07(1986) to Sun-28-Sep@23h52(1986) : sim@15m, output@15m

 Zone total sensible and latent plant used (kWhrs)
    Zone       Sensible heating  Sensible cooling  Humidification   Dehumidification
 id name       Energy   No. of   Energy   No. of   Energy   No. of  Energy   No. of
               (kWhrs)  Hr rqd   (kWhrs)  Hr rqd   (kWhrs)  Hr rqd  (kWhrs)  Hr rqd
  2 kiten_util  52.94  105.8       0.00    0.0       0.00    0.0       0.00    0.0
```

```
  3 bedroom1    54.53  104.8       0.00    0.0       0.00    0.0       0.00    0.0
  4 bath         6.49   53.0       0.00    0.0       0.00    0.0       0.00    0.0
  5 bedroom2    19.02   64.8       0.00    0.0       0.00    0.0       0.00    0.0
  6 living      59.07  116.0       0.00    0.0       0.00    0.0       0.00    0.0
  7 hall        27.68  118.0       0.00    0.0       0.00    0.0       0.00    0.0
  8 study       12.67   94.5       0.00    0.0       0.00    0.0       0.00    0.0
  9 family      32.13  124.0       0.00    0.0       0.00    0.0       0.00    0.0

    All        264.53  780.8       0.00    0.0       0.00    0.0       0.00    0.0


# Lib: holmes_aut.res: Results for holmes_b
# Period: Mon-22-Sep@00h07(1986) to Sun-28-Sep@23h52(1986) : sim@15m, output@15m
# Zone sensible load (kW)
Description Max_value Max_occur Min_value Min_occur Ave_value Std_dev
kiten_util  1.3984 24-Sep@06h22  0.00000 22-Sep@00h22  0.31511  0.34329
bedroom1    1.5294 24-Sep@06h22  0.00000 22-Sep@00h22  0.32458  0.36422
bath        0.38427 22-Sep@06h37 0.00000 22-Sep@00h22  0.03861  0.08020
bedroom2    0.82774 24-Sep@06h37 0.00000 22-Sep@00h22  0.11323  0.19984
living      1.6467 22-Sep@06h37  0.00000 22-Sep@00h22  0.35162  0.38454
hall        0.77728 24-Sep@06h37 0.00000 22-Sep@00h22  0.16476  0.17797
study       0.45479 22-Sep@06h37 0.00000 22-Sep@00h22  0.07544  0.10675
family      0.89996 22-Sep@06h37 0.00000 22-Sep@00h22  0.19125  0.22004

   All      7.8464 24-Sep@06h22  0.00000 22-Sep@00h00       --       --
```

Why bother? Well onced the script is working correctly it will carry out the same sequence every time. Users get distracted and might not choose exactly the same options in the same order. Interactive use of the res module takes time and attention. Automated tasks takes the human delay out of the work-flow.

Other examples of scripts can be found in the folder validation/benchmark/QA/model_1.1/cfg. TEST is a high level script that calls the SIMULATE.* scripts and ANAYSE* scripts to run a standard set of over a score of assessments and report generation tasks.

Finding library dependencies

Say you just updated your ESP-r distribution and you can edit details of an ESP-r model file but the simulator does not start. Most frustrating! This might be because of a missing or corrupted library (dll) file. You can use a command line command to investigate this.

Essentially what is needed is to go to the folder where the ESP-r module executable lives on your machine and then use some command line magic to find out more about the executable.

Cygwin

ESP-r executables tend to live (within the Cygwin folder structure) in /usr/esru/esp-r/bin so if you Cygwin includes the utility ldd.exe then you can find out about dependencies via:

cd /usr/esru/esp-r/bin
ldd bps

OSX

ESP-r executables tend to live in /opt/esru and the utility that identifies dependencies is named otool so you can find out about dependencies via:

cd /opt/esru/esp-r/bin
otool -L bps

Linux

ESP-r executables tend to live in either /usr/esru or /opt/esru and the utility that identifies dependencies is named ldd so you can find out about dependencies via:

cd /opt/esru/esp-r/bin
ldd bps