Overview of MSU Compute Servers

The DECS Linux-based compute servers are well suited for programs that are too slow to run on typical desktop computers but do not require the power of supercomputers. The purpose of this handout is to describe how to access a compute server remotely from a Windows PC or MAC.
Additional information about the compute servers that may help you decide which one to use is available on the DECS website (search the DECS support site for 'compute server').

You may wish to run matlab from a PC using DECS remote desktop services (search the DECS support site for 'remote desktop'). That process is NOT the same as described in this handout. The procedure described here connects you directly with matlab on a compute server.

NOTE: From the engineering network, users may login directly to the desired computer server using the SSH connection.

Computer Setup to Run Unix/Linux Programs

Quick Start – for either MAC or PC, **skip the use of the GUI when outside the engineering network**. Most tasks can be done quickly without the GUI. The GUI runs VERY SLOWLY when outside the engineering network.

SSH Connections

MACs – the MAC 'XTerm' application can be used to create an ssh connection, no utility is needed.

PCs – X2Go

Note that X2Go is already installed on all DECS machines. For your own machine, search the DECS support site for X2Go and follow the instructions. The program will provide a GUI or terminal window if you need to use the GUI or if plots need to be displayed.

Connecting to a Linux server

MACS and PCs - It may save time to run your program on a compute server with a relatively low computational load. To determine which compute servers have low loads, once you are logged into a compute server described below, in a terminal window, enter 'computeuse'. The response will be displayed indicating the relative load on the various compute servers (example below). Logout and reconnect to a machine with a low load if desired. NOTE: A common error is to log into the scully gateway. Matlab will not run on scully.

```
<127 [lira@brock]:~/Desktop >computeuse
HOST      VCPU   LOAD   LOAD%   MEMTOT  MEMUSE  MEM%     SWAP    UPTIME
================================================================================
baron       48  17.69    37%     377G     65G    17%       0%     7 days
brock       72  55.14    77%     377G    318G    84%     100%     9 days
byron       88   5.70     6%     755G    211G    28%       0%    12 days
courtney    72  36.19    50%     755G    305G    40%       0%    38 days
monarch     72  69.46    96%     755G    244G    32%       0%     7 days
rusty       48  26.50    55%     377G    183G    49%       0%     7 days
sheila      72  65.94    92%     755G     28G     4%       0%    12 days
triana                      (Unavailable)
```

MACS –

1) Start Applications>Utilities>XTerm.
2) Enter '`ssh -Y <username>@<computername>.egr.msu.edu`' where <username> is your engineering username and <computername> is the machine where you want to connect. Omit the '-Y' if not using GUI or plots.

PCs -

1) Start X2Go and log in. The first time you connect to a given server, you will receive a prompt to accept an RSA security key that will be stored on your computer. If the key ever changes after that, you know that the server has changed or that your connection has been compromised.
2) On X2Go, the menu is obtained by right-clicking on the desktop, and prominent choice is to 'Open Terminal Here'.

Running Matlab

After connecting, there are several ways to run Matlab.

Running without the desktop

The MATLAB GUI can be slow when logged in remotely. Or you may have already debugged your code and have no need for the interface and the associated use of connection bandwidth. Move to the folder with your matlab files and enter

If you will be displaying plots:

```
matlab -nodesktop  -nosplash
```

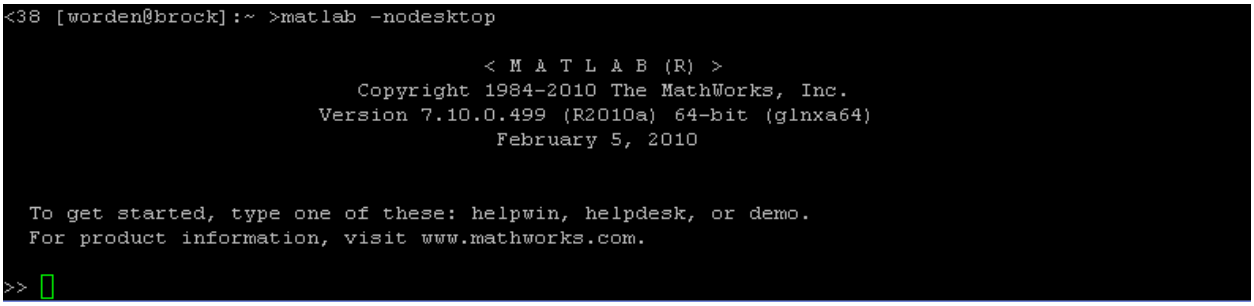
If not displaying plots:

```
matlab -nodesktop  -nosplash -nojvm
```

The GUI will not appear, but MATLAB will prompt you for commands through the terminal window, as shown below. Code without the desktop GUI will still be able to show graphical output if the –nojvm option was not used. To run 'myfile.m', simply type '`run myfile`' at the command window prompt. (The 'run' command will not work if the .m files require inputs or return values just use function syntax: [out1,out2,…] = functionname(arg1, arg2,…)). Note that '`cd`' and '`dir`' commands can be used to navigate/view folders within matlab. To find the present working directory, 'pwd'.

To view a long command window output, it may help to create a 'diary' log. Use the matlab command diary('output.txt') to append a log of the command window into 'output.txt' or whatever filename is used. Use 'diary off' to stop logging.

To view or edit a file the 'bang' (!) command can be used to run a linux system command from within Matlab. (These commands also work from outside Matlab without the !.) Some common linux commands:
!nano myscript.m – runs the 'nano' editor to edit the file myscript.m.
!less -N myscript.m – view myscript.m with line numbers. PgUp, PgDn, scroll. To quit, press q.
Note: within less, type 'ng' to jump to line n, e.g. 50g jumps to line 50.
!head myscript.m – view the first few lines of myscript.m
!rm output.txt – remove file 'output.txt'



Debugging with the command line interface

When running debugging small jobs, the command line interface works very well. Use 'help dbstop' for a summary of commands. Common commands are:
dbstop in <filnename> - will stop at first executable line in <filename> when it is run.
dbstop in <filename> at n – will stop at line 'n' when <filename> is run.
dbstop if error – will stop if an error occurs so that you can inspect variables.
Helpful line commands at a breakpoint:
dbstep – advance one line
dbcont – continue.
dbstatus – list of breakpoints.
dbtype <filename> <optional first line>:<optional last line> - display file with line numbers.
who – list variables.
whos – list variables, type, and size.
Note: the bang command works at a breakpoint which may be helpful to run 'less' to view a file.
Note: enter a variable name to view the contents.

Use 'quit' to exit Matlab.

Running batch job without interactive input

If the job is big, it may be most convenient to avoid running the user interface at all. Suppose we want to run a long job 'myfile.m'. The file 'myfile.m' does not create any graphical output, but returns a lot of results to the workspace.

IMPORTANT: The commands below will start MATLAB and run your file, but MATLAB will stay running unless you include an 'exit;' or 'quit;' statement in your script/function.

Use of 'nohup' will enable you to log out without killing the job.

Create a unix/linux script file using Notepad++ (This is NOT windows notepad) or a unix/linux editor. In Notepad++ be sure to set 'Format>Convert to UNIX format'. This script needs to be created only one time:

```
#! /bin/csh -f
# Clear the DISPLAY.
unsetenv DISPLAY  # unset DISPLAY for some shells
# Call MATLAB with the appropriate input and output,
# make it immune to hangups and quits using ''nohup'',
# and run it in the background.
nohup matlab -nodisplay -nodesktop -nojvm -nosplash -r $1 > $2 &
```

All lines that start with # except the first are comments that can be omitted. The first line is required to specify the script language used. **Be sure to follow whitespace carefully and be sure to put a newline after the last line because the last line will 'hang' without a newline character, just as if you type a command but fail to press the 'enter' key.** Filenames you specify on the command line will be substituted for $1 and $2 when the script is executed. Save the file with a convenient name such as 'runmatlab.sh' and set executable permissions using '`chmod 700 runmatlab.sh`'. The 'sh' extension is commonly used to name script files.

Run the matlab program using
`runmatlab.sh myfile.m myoutput.txt`
where you substitute the .m filename for 'myfile.m' the desired command window output file name for myoutput.txt.

The system will return a 'process id' number. Make a note of the server name and the process id. Command window output will go to 'myoutput.txt' or whatever name you designated. The file myfile.m can contain a command to save the workspace, such as 'save myworkspace.mat' or whatever .mat name you choose. The workspace results can later be restored in a new matlab session by using 'load myworkspace.mat'. The workspace results can then be interactively plotted or examined. There are also various methods to create text-based output files.

The GUI and command-line methods can be combined by establishing both command-line and GUI connections to the compute servers. The time-intensive computations could be run using the command-line connection and then saving the workspace as a myworkspace.mat file. Then, the GUI connection could be used to load the myworkspace.mat file and plot the results at a later time. You can even create the .mat file on linux and then load it into windows.

To check that a hohup command has finished, you must log into the same server used to start the nohup job. Enter '`ps -umyuser`' where myuser is your username. If you recorded the process id or if you submitted only one nohup job, you can verify it the process is finished when it disappears from the list.

Running with the desktop GUI

The first way is to use MATLAB's graphical user interface (GUI) by entering the command 'matlab' at the prompt.

The MATLAB splash screen should show up briefly on the screen, followed by the GUI shown below.

When the GUI window opens, the working directory should be changed to the directory with your files.

When you are done

PCs – Closing the ssh session

Click the x to close the terminal window. To exit X2Go 'nicely', click the green 'running person' icon in the top bar.

MACs- Closing the ssh session

To exit 'nicely' use 'logout' at the ssh prompt to close the session.