Parallel Computing with MATLAB at UVa

Ed Hall [1]

[1]University of Virginia Advanced Computing Services and Engagement
uvacse@virginia.edu

October 6, 2014

Outline

Matlab and Parallelism

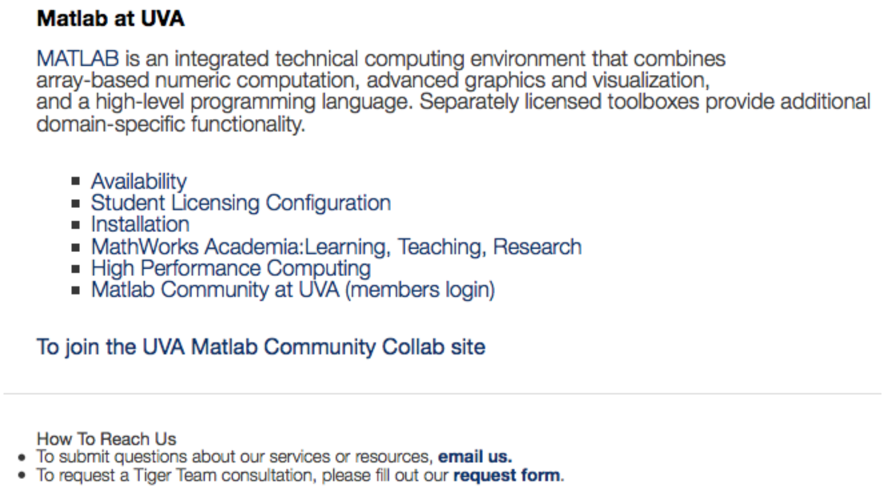
See slide deck presented by the MathWorks on parallel Matlab at local event in the Fall of 2013.

Matlab Parallel Computing Short Course

More generally, Matlab High Performance Computing

http://www.uvacse.virginia.edu/software/matlab-at-uva/high-performance-matlab-at-uva/

Matlab at UVa



Matlab at UVA

MATLAB is an integrated technical computing environment that combines array-based numeric computation, advanced graphics and visualization, and a high-level programming language. Separately licensed toolboxes provide additional domain-specific functionality.

- Availability
- Student Licensing Configuration
- Installation
- MathWorks Academia:Learning, Teaching, Research
- High Performance Computing
- Matlab Community at UVA (members login)

To join the UVA Matlab Community Collab site

How To Reach Us

- To submit questions about our services or resources, **email us.**
- To request a Tiger Team consultation, please fill out our **request form**.

UVACSE link:
http://www.uvacse.virginia.edu/software/matlab-at-uva/

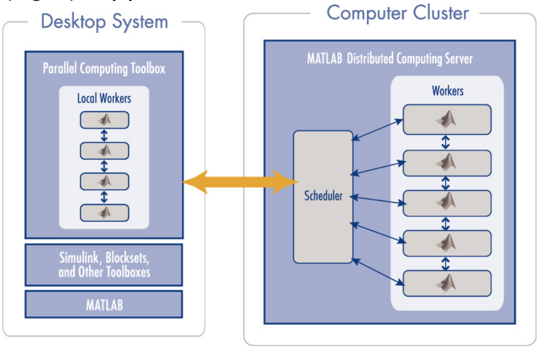
Installing and Configuring NX Client

The NX client provides a Gnome Linux desktop interface to the login node of the fir.itc Linux cluster.

http://uvacse.virginia.edu/resources/itc-linux-cluster-overview/the-nx-client/

Matlab Parallel Workflow

The toolbox enables application prototyping on the desktop with up to 12 local workers (left), and with Matlab Distributed Computing Server (right), applications can be scaled to multiple computers on a cluster.


Desktop System
Parallel Computing Toolbox
Local Workers
Simulink, Blocksets, and Other Toolboxes
MATLAB
Computer Cluster
MATLAB Distributed Computing Server
Workers
Scheduler

Scaling Up from the Desktop

Parallel Computing Toolbox provides the ability to use up to 12 local workers on a multicore or multiprocessor computer using a single toolbox license.

When used together with MATLAB Distributed Computing Server, you can scale up your application to use any number of workers running on any number of computers.

ITS Linux cluster has MDCS licenses for 256 workers.

Alternatively, you can run up to 12 workers on a single multi-core compute node of the cluster.

ITS Cluster documentation:
http://www.uvacse.virginia.edu/itc-clusters/

Using Parallel Configurations with PBS Pro

Parallel Configurations - Where and How the Code is Executed

Maintain named configurations

- Predefine cluster information and environment-specific parameters
- No code changes required
- Set once, use many times

Useful for both interactive and batch workflows

Toolbox provides GUI to manage configurations

Passwordless ssh Connections

You will need to configure your cluster account so that the Matlab PCT can use passwordless ssh to log into the cluster compute nodes using the following procedure:

1. If you don't have a key pair (id_rsa, id_rsa.pub in .ssh), run

```
ssh-keygen
```

2. Give it no passphrase
3. If you don't already have an authorized_keys file, from your home directory run the commands

```
cd  .ssh
cp  id_rsa.pub authorized_keys
```

You can test if this works by logging into a compute node directly from the cluster front-end node (generally discouraged) with the command

```
ssh lc4-compute-2-3
```

Running Matlab on Cluster Front-end Node

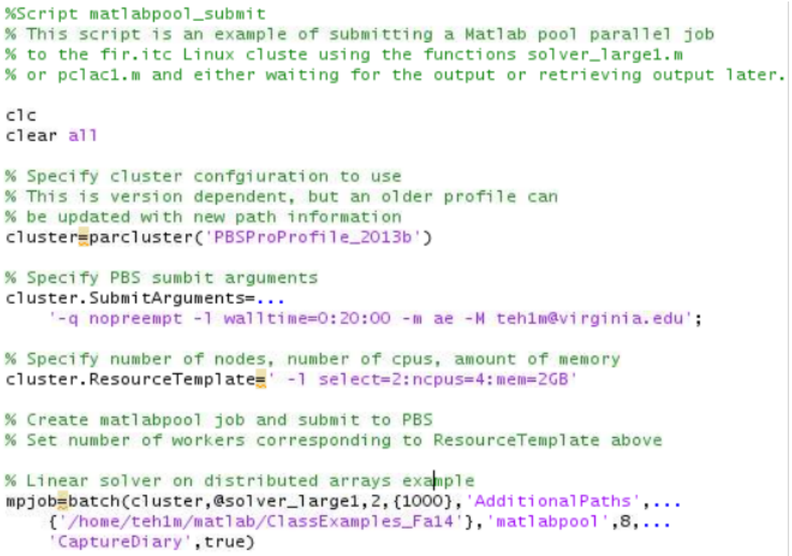
Matlab Parallel Computing jobs can be submitted to the ITC Linux cluster by first logging onto the cluster front-end node fir.itc.virginia.edu using the NX client.

Start up Matlab from a Linux desktop terminal window.

Matlab Parallel Computing Toolbox jobs can be submitted from within Matlab and the example scripts show how to setup and submit the jobs

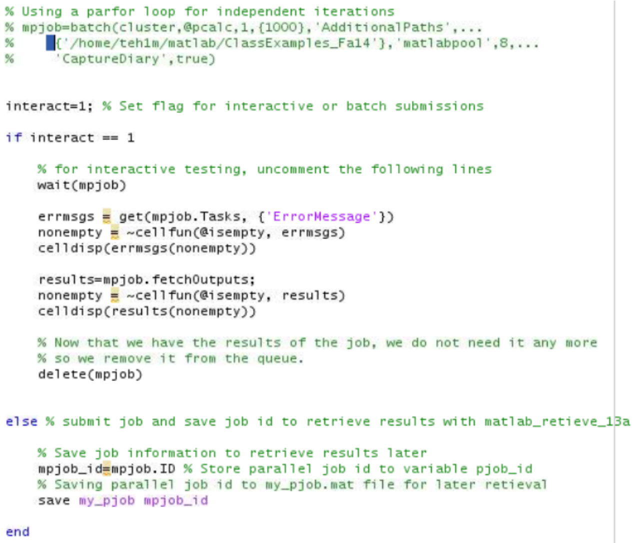
Alternatively, parallel Matlab jobs can be launched to the cluster using PBS shell scripts from the Linux command line in a terminal window.

Matlab Parallel Job Submission



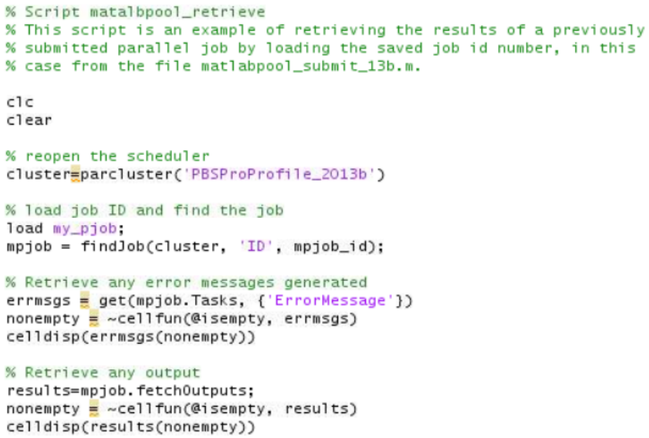
```
%Script matlabpool_submit
% This script is an example of submitting a Matlab pool parallel job
% to the fir.itc Linux cluste using the functions solver_large1.m
% or pclac1.m and either waiting for the output or retrieving output later.

clc
clear all

% Specify cluster confgiuration to use
% This is version dependent, but an older profile can
% be updated with new path information
cluster=parcluster('PBSProProfile_2013b')

% Specify PBS sumbit arguments
cluster.SubmitArguments=...
    '-q nopreempt -l walltime=0:20:00 -m ae -M teh1m@virginia.edu';

% Specify number of nodes, number of cpus, amount of memory
cluster.ResourceTemplate=' -l select=2:ncpus=4:mem=2GB'

% Create matlabpool job and submit to PBS
% Set number of workers corresponding to ResourceTemplate above

% Linear solver on distributed arrays example
mpjob=batch(cluster,@solver_large1,2,{1000},'AdditionalPaths',...
    {'/home/teh1m/matlab/ClassExamples_Fa14'},'matlabpool',8,...
    'CaptureDiary',true)
```

Matlab Parallel Job Submission



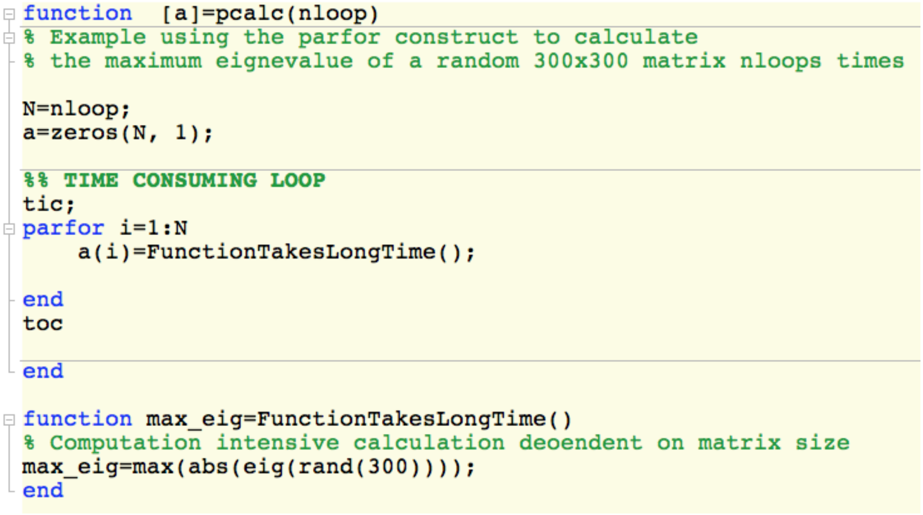
```
% Using a parfor loop for independent iterations
% mpjob=batch(cluster,@pcalc,1,{1000},'AdditionalPaths',...
%     {'/home/teh1m/matlab/ClassExamples_Fa14'},'matlabpool',8,...
%      'CaptureDiary',true)


interact=1; % Set flag for interactive or batch submissions

if interact == 1

    % for interactive testing, uncomment the following lines
    wait(mpjob)

    errmsgs = get(mpjob.Tasks, {'ErrorMessage'})
    nonempty = ~cellfun(@isempty, errmsgs)
    celldisp(errmsgs(nonempty))

    results=mpjob.fetchOutputs;
    nonempty = ~cellfun(@isempty, results)
    celldisp(results(nonempty))

    % Now that we have the results of the job, we do not need it any more
    % so we remove it from the queue.
    delete(mpjob)


else % submit job and save job id to retrieve results with matlab_retieve_13a

    % Save job information to retrieve results later
    mpjob_id=mpjob.ID % Store parallel job id to variable pjob_id
    % Saving parallel job id to my_pjob.mat file for later retieval
    save my_pjob mpjob_id

end
```

Matlab Parallel Job Submission



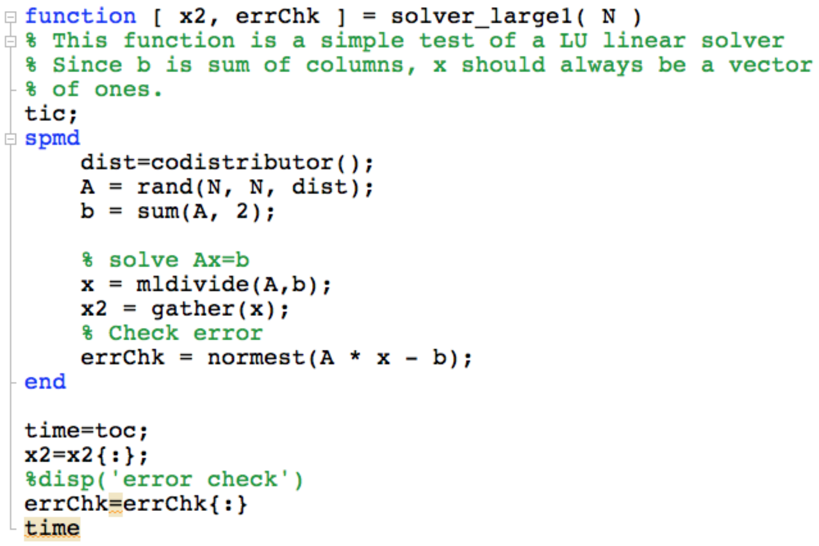
```
% Script matalbpool_retrieve
% This script is an example of retrieving the results of a previously
% submitted parallel job by loading the saved job id number, in this
% case from the file matlabpool_submit_13b.m.

clc
clear

% reopen the scheduler
cluster=parcluster('PBSProProfile_2013b')

% load job ID and find the job
load my_pjob;
mpjob = findJob(cluster, 'ID', mpjob_id);

% Retrieve any error messages generated
errmsgs = get(mpjob.Tasks, {'ErrorMessage'})
nonempty = ~cellfun(@isempty, errmsgs)
celldisp(errmsgs(nonempty))

% Retrieve any output
results=mpjob.fetchOutputs;
nonempty = ~cellfun(@isempty, results)
celldisp(results(nonempty))
```

Matlab Parallel Job Submission



```
function  [a]=pcalc(nloop)
% Example using the parfor construct to calculate
% the maximum eignevalue of a random 300x300 matrix nloops times

N=nloop;
a=zeros(N, 1);

%% TIME CONSUMING LOOP
tic;
parfor i=1:N
    a(i)=FunctionTakesLongTime();

end
toc

end

function max_eig=FunctionTakesLongTime()
% Computation intensive calculation deoendent on matrix size
max_eig=max(abs(eig(rand(300))));
end
```

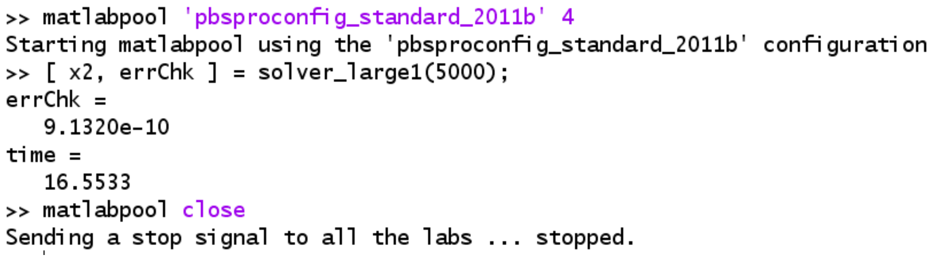
Matlab Parallel Job Submission



```
function [ x2, errChk ] = solver_large1( N )
% This function is a simple test of a LU linear solver
% Since b is sum of columns, x should always be a vector
% of ones.
tic;
spmd
    dist=codistributor();
    A = rand(N, N, dist);
    b = sum(A, 2);

    % solve Ax=b
    x = mldivide(A,b);
    x2 = gather(x);
    % Check error
    errChk = normest(A * x - b);
end

time=toc;
x2=x2{:};
%disp('error check')
errChk=errChk{:}
time
```

Interactive Parallel Matlab Jobs on Cluster

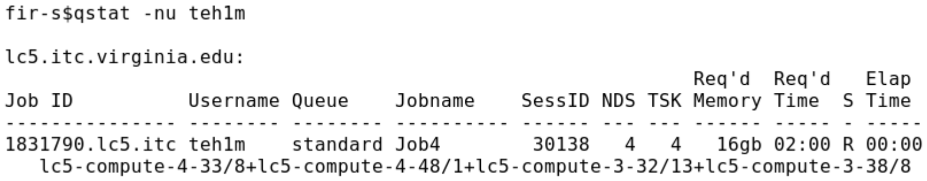
The `matlabpool` command can be used with the cluster configuration file to launch an interactive job to the cluster from within the Desktop interface on the cluster front-end.



```
>> matlabpool 'pbsproconfig_standard_2011b' 4
Starting matlabpool using the 'pbsproconfig_standard_2011b' configuration
>> [ x2, errChk ] = solver_large1(5000);
errChk =
   9.1320e-10
time =
   16.5533
>> matlabpool close
Sending a stop signal to all the labs ... stopped.
```

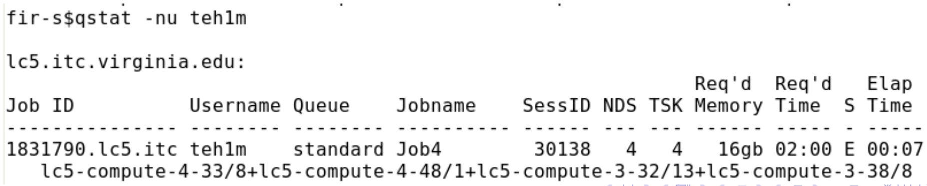
Interactive Parallel Matlab Jobs on Cluster

After execution of the `matlabpool` command with the cluster configuration file, the PBS command `qstat` command shows the compute nodes serving as workers have been allocated.



```
fir-s$qstat -nu tehlm

lc5.itc.virginia.edu:
                                                                  Req'd  Req'd   Elap
Job ID          Username Queue    Jobname    SessID NDS TSK Memory Time  S Time
--------------- -------- -------- ---------- ------ --- --- ------ ----- - -----
1831790.lc5.itc tehlm    standard Job4        30138   4   4   16gb 02:00 R 00:00
   lc5-compute-4-33/8+lc5-compute-4-48/1+lc5-compute-3-32/13+lc5-compute-3-38/8
```

After the `matlabpool close` command, the job is exiting.



```
fir-s$qstat -nu tehlm

lc5.itc.virginia.edu:
                                                                  Req'd  Req'd   Elap
Job ID          Username Queue    Jobname    SessID NDS TSK Memory Time  S Time
--------------- -------- -------- ---------- ------ --- --- ------ ----- - -----
1831790.lc5.itc tehlm    standard Job4        30138   4   4   16gb 02:00 E 00:07
   lc5-compute-4-33/8+lc5-compute-4-48/1+lc5-compute-3-32/13+lc5-compute-3-38/8
```

Parallel Matlab Jobs on Cluster Node

The following command will submit an interctive job to PBS to use 12 cores on one compute node:

```
qsub -I -v DISPLAY="fir.itc.virginia.edu$DISPLAY"
-l select =1:ncpus=12
```

Once you are logged into the compute node, you can start Matlab from the Linux command line.

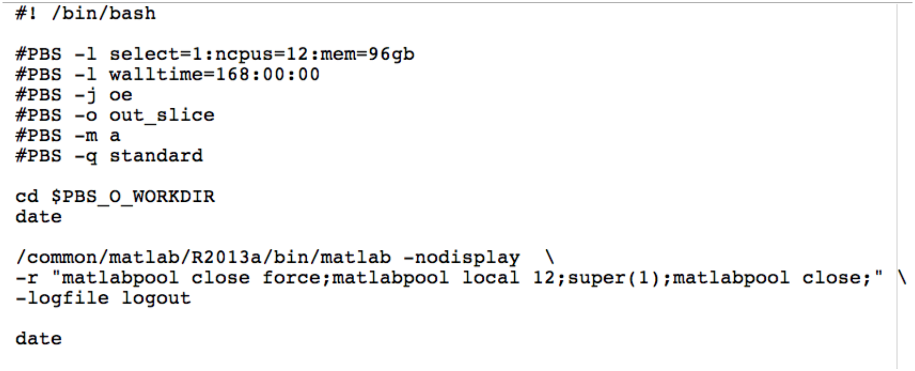
```
matlab
```

and from within Matlab, open a matlabpool of 12 workers to the 12 local cores you have been allocated by PBS with the command:

```
» matlabpool 'local' 12
```

Parallel Matlab Jobs on Cluster Node

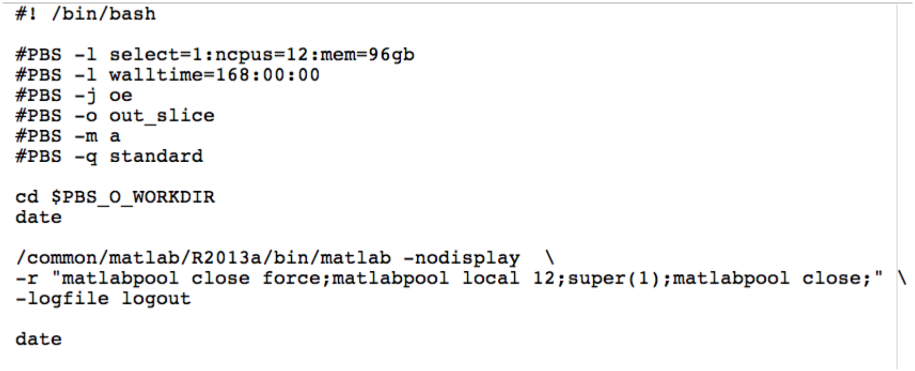
Submitting a batch parallel Matlab job to run on one node.



```
#! /bin/bash

#PBS -l select=1:ncpus=12:mem=96gb
#PBS -l walltime=168:00:00
#PBS -j oe
#PBS -o out_slice
#PBS -m a
#PBS -q standard

cd $PBS_O_WORKDIR
date

/common/matlab/R2013a/bin/matlab -nodisplay  \
-r "matlabpool close force;matlabpool local 12;super(1);matlabpool close;" \
-logfile logout

date
```

Parallel Matlab Jobs on Cluster Node

Submitting a batch parallel Matlab job to run across multiple nodes node.



```
#! /bin/bash

#PBS -l select=1:ncpus=12:mem=96gb
#PBS -l walltime=168:00:00
#PBS -j oe
#PBS -o out_slice
#PBS -m a
#PBS -q standard

cd $PBS_O_WORKDIR
date

/common/matlab/R2013a/bin/matlab -nodisplay  \
-r "matlabpool close force;matlabpool local 12;super(1);matlabpool close;" \
-logfile logout

date
```

Running Matlab in Parallel on the XCG

Cross Campus Grid Project

http://www.uvacse.virginia.edu/cross-campus-grid-xcg/

- Ideal for Monte Carlo Simulations or Parameter Space Studies.
- Up to 1500 cores available.
- For information, email uvacse@virginia.edu

Additional Computational Resources

XSEDE: Extreme Science and Engineering Discovery Environment

https://www.xsede.org/

MATLAB GPU Computing with NVIDIA CUDA-Enabled GPUs

A Tesla GPU card is available for testing Matlab GPU code in the CS department.

References

1. Mathworks Parallel Computing Toolbox Documentation
   http://www.mathworks.com/products/parallel-computing/
2. Mathworks Parallel Computing Toolbox Demos and Webinars
   http://www.mathworks.com/products/parallel-computing/demos.html
3. *Parallel Matlab for Multicore and Multinode Computers*, by Jeremy Kepner, SIAM Press.

Need further help? Email uvacse@virginia.edu.