



Introduction to MATLAB in HPC Environment

Sagnik Singha
*High Performance Computing Center
(on behalf of the HPCC staff)*

November 12th, 2025



- MATLAB in the TTU HPCC environment
- Job submission methods
- Batch Submission
- Graphical Client based Submission
- Parallel computing constructs – CPU, GPU
- Containerized MATLAB (MathWorks account)
- National HPC Resources and their allocations

Logging on to TTU HPCC



TEXAS TECH UNIVERSITY
Information Technology Division™

https://www.depts.ttu.edu/hpcc/userguides/general_guides/login_general.php

Connecting to RedRaider – On Campus

```
ssh <eraider>@login.hpcc.ttu.edu
```

MATLAB in the TTU HPCC - Environment setup



TEXAS TECH UNIVERSITY
Information Technology Division™

Setup MATLAB environment

```
module load matlab
```

Default version loaded (nocona): matlab/matlab/R2024a

check if MATLAB is currently loaded in your environment, please type:

```
module list
```

```
cpu-24-34:$ ml matlab  
cpu-24-34:$ ml list
```

Currently Loaded Modules:

1) nocona/0.15.4 2) matlab/R2024a

```
cpu-1-16:/matlab_shortcourse$ ml list
```

Currently Loaded Modules:

1) matlab/R2024a

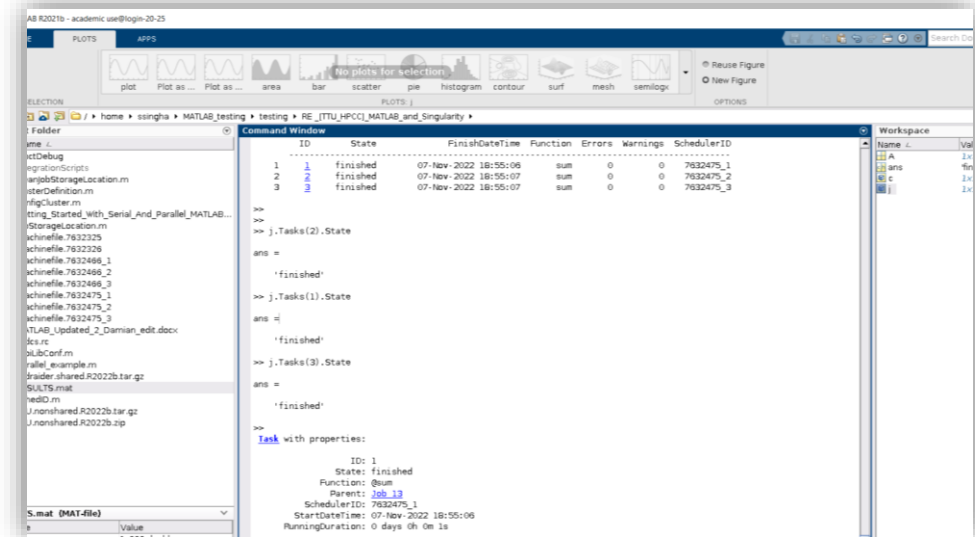
Job submission methods



TEXAS TECH UNIVERSITY
Information Technology Division

- Graphical User Interface –
 - ❖ Interact directly with the job
 - ❖ Prototyping, troubleshooting code - If code is in developmental stages
 - ❖ If direct interaction with output necessary
 - ❖ Worker node
- BATCH submission –
 - ❖ Automate jobs in Linux environment
 - ❖ Production, submitting code - If code is in production stages
 - ❖ If multiple runs needed without direct interaction with output
 - ❖ Login node

<https://slurm.schedmd.com/sbatch.html>



```
#SBATCH --job-name=matlab-test
#SBATCH --output=%x.o%j
#SBATCH --error=%x.e%j
#SBATCH --partition=nocona
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=1
#SBATCH --time=10:00:00      ##10 hour runtime, you may change or remove this op
#SBATCH --cpus-per-task=1      # 1 core per instance

####SBATCH --mem=512000MB

# Add MATLAB to system path
module load matlab

# Run code
matlab -batch calc_pi_multi_node

~
~
~
~
```

Batch submission



TEXAS TECH UNIVERSITY
Information Technology Division™

Example file's location:

```
/lustre/work/examples/nocona/matlab/fall_2025
```

Copy files to own workspace

```
cp -r /lustre/work/examples/nocona/matlab/fall_2025 /home/<eraider_ID>/.
```

Example 1:

Matrix multiplication

Files:

A_matrix.dat

B_matrix.dat

matmulfile.m

matlab.sh

3 X 3 Matrix

$$\begin{bmatrix} 1 & 23 & 456 \\ 54 & 4 & 412 \\ 0 & 99 & 69 \end{bmatrix}$$

Batch submission



TEXAS TECH UNIVERSITY
Information Technology Division™

https://www.depts.ttu.edu/hpcc/userguides/Job_User_Guide.pdf

matlab.sh

```
#!/bin/bash
#SBATCH --job-name=matlab-test
#SBATCH --output=%x.o%j
#SBATCH --error=%x.e%j
#SBATCH --partition nocona
#SBATCH --nodes=1
#SBATCH --ntasks=1
module load matlab
matlab -batch multiplyMatrix
```

matmulfile.m

```
load A.dat;
load B.dat;

n=100
A=rand(n)
B=rand(n)

atimesb = A*B
aeletimesb = A.*B

save -ascii atimesb.txt
atimesb
save -ascii aeletimesb.txt
aeletimesb
quit
```

Batch submission and Monitoring



TEXAS TECH UNIVERSITY
Information Technology Division™

Submission

```
sbatch matlab.sh
```

Monitoring

```
squeue -u <username>
```

Status 1:

```
cpu-26-10:/MATLAB_testing/nocona/matlab/Example_1$ sbatch matlab.sh
Submitted batch job 7599523
cpu-26-10:/MATLAB_testing/nocona/matlab/Example_1$ squeue -u ssingha
      JOBID PARTITION    NAME    USER ST       TIME  NODES NODELIST(REASON)
      7599523     nocona matlab-t  ssingha PD        0:00        1 (None)
```

Status 2:

```
cpu-23-38:/MATLAB_testing/nocona/matlab/Example_1$ squeue -u ssingha
      JOBID PARTITION    NAME    USER ST       TIME  NODES NODELIST(REASON)
      7628458     nocona INTERACT  ssingha R        2:13:20        1 cpu-23-38
      7628776     nocona matlab-t  ssingha R         0:11        1 cpu-23-23
```

Status 3:

```
cpu-23-38:/MATLAB_testing/nocona/matlab/Example_1$ squeue -u ssingha
      JOBID PARTITION    NAME    USER ST       TIME  NODES NODELIST(REASON)
      7628458     nocona INTERACT  ssingha R        2:13:22        1 cpu-23-38
cpu-23-38:/MATLAB_testing/nocona/matlab/Example_1$
```

Batch submission - Monitoring



TEXAS TECH UNIVERSITY
Information Technology Division

Status 2:

Monitoring

`$squeue -u <username>`

```
cpu-23-38:/MATLAB_testing/nocona/matlab/Example_1$ squeue -u ssingha
```

JOBID	PARTITION	NAME	USER	ST	TIME	NODES	NODELIST(Reason)
7628458	nocona	INTERACT	ssingha	R	2:13:20	1	cpu-23-38
7628776	nocona	matlab-t	ssingha	R	0:11	1	cpu-23-23

`$ssh cpu-23-23`

`$top | grep MATLAB`

```
cpu-23-1:$ top | grep MATLAB
```

5318	ssingha	20	0	21.3g	475432	217068	S	133.3	0.1	0:03.12	MATLAB
4541	ssingha	20	0	21.3g	466260	214360	S	105.6	0.1	0:03.11	MATLAB
4537	ssingha	20	0	21.3g	482448	221040	S	100.0	0.1	0:03.14	MATLAB
4544	ssingha	20	0	21.3g	486596	220236	S	94.4	0.1	0:03.15	MATLAB
4539	ssingha	20	0	21.3g	489100	220840	S	88.9	0.1	0:03.19	MATLAB
5074	ssingha	20	0	21.3g	480076	216208	S	83.3	0.1	0:03.08	MATLAB
4534	ssingha	20	0	21.3g	476296	215080	S	77.8	0.1	0:03.11	MATLAB
4531	ssingha	20	0	21.7g	736408	230976	S	55.6	0.1	0:03.75	MATLAB
4534	ssingha	20	0	22.1g	951032	262028	S	105.0	0.2	0:06.29	MATLAB
5074	ssingha	20	0	22.1g	953908	262056	S	103.6	0.2	0:06.22	MATLAB
4539	ssingha	20	0	22.1g	952092	263620	S	101.0	0.2	0:06.25	MATLAB
4537	ssingha	20	0	22.0g	936720	262352	S	97.7	0.2	0:06.10	MATLAB
4544	ssingha	20	0	22.1g	923068	261824	S	97.7	0.2	0:06.11	MATLAB
4541	ssingha	20	0	22.0g	896352	257788	S	93.4	0.2	0:05.94	MATLAB
4531	ssingha	20	0	22.2g	976064	272068	S	89.1	0.2	0:06.45	MATLAB
5318	ssingha	20	0	22.0g	895616	256076	S	87.8	0.2	0:05.78	MATLAB
3698	ssingha	20	0	23.2g	1.2g	312528	S	0.3	0.2	0:13.07	MATLAB
5074	ssingha	20	0	22.8g	1.1g	290956	S	99.3	0.2	0:09.23	MATLAB
4544	ssingha	20	0	22.5g	1.1g	292092	S	99.0	0.2	0:09.11	MATLAB

Batch submission - Monitoring



TEXAS TECH UNIVERSITY
Information Technology Division™

Status 1:

```
cpu-26-10:/MATLAB_testing/nocona/matlab/Example_1$ sbatch matlab.sh
Submitted batch job 7599523
cpu-26-10:/MATLAB_testing/nocona/matlab/Example_1$ squeue -u ssingha
```

JOBID	PARTITION	NAME	USER	ST	TIME	NODES	NODELIST(REASON)
7599523	nocona	matlab-t	ssingha	PD	0:00	1	(None)

Monitoring

`squeue -u <username>`

Status 2:

```
cpu-23-38:/MATLAB_testing/nocona/matlab/Example_1$ squeue -u ssingha
```

JOBID	PARTITION	NAME	USER	ST	TIME	NODES	NODELIST(REASON)
7628458	nocona	INTERACT	ssingha	R	2:13:20	1	cpu-23-38
7628776	nocona	matlab-t	ssingha	R	0:11	1	cpu-23-23

Files

- `matlab-test.e<JOB_ID>`
- `matlab-test.o<JOB_ID>`
- `atimesb.txt`

Status 3:

```
cpu-23-38:/MATLAB_testing/nocona/matlab/Example_1$ squeue -u ssingha
```

JOBID	PARTITION	NAME	USER	ST	TIME	NODES	NODELIST(REASON)
7628458	nocona	INTERACT	ssingha	R	2:13:22	1	cpu-23-38

```
cpu-23-38:/MATLAB_testing/nocona/matlab/Example_1$
```

Batch submission



TEXAS TECH UNIVERSITY
Information Technology Division™

Example 2:

Matrix multiplication

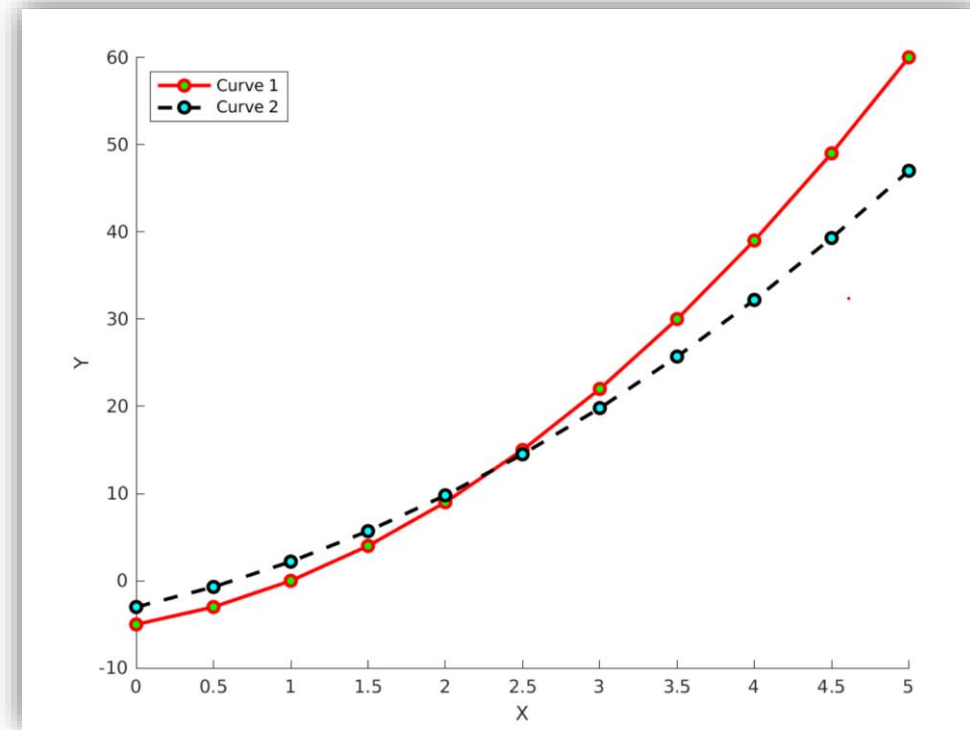
Files:

example_2.m

matlab.sh

Files

- matlab-test.e<JOB_ID>
- matlab-test.o<JOB_ID>
- func_plot.png



Batch submission -

Parallel implementation, Maximum Eigenvalue of a matrix



Information Technology Division

Link: /matlab/matlabPar

Serial version

Code:

```
%Start timing
tic

%Calculates spectral radius of each matrix and
displays results
n = 10;
A = 500;
a = zeros(n);
for i = 1:n*n
    a(i) = max(abs(eig(rand(A)))));
end
disp(a)

%End timing
time = toc;

%Display the time of computation
fprintf('The parallel method is executed in %5.2f
seconds. \n', time);
```

Uniform distribution
of random numbers

Parallel version

Batch submission -

Parallel implementation, Maximum Eigenvalue of a matrix



TEXAS TECH UNIVERSITY

Information Technology Division™

Link: /matlab/matlabPar

Serial version

Code:

```
%Start timing
tic

%Calculates spectral radius of each matrix and
displays results
n = 10;
A = 500;
a = zeros(n);
for i = 1:n*n
    a(i) = max(abs(eig(rand(A)))));
end
disp(a)

%End timing
time = toc;

%Display the time of computation
fprintf('The parallel method is executed in %5.2f
seconds. \n', time);
```

P 1

P 2

Parallel version

Code:

```
c = parcluster('local'); % local is a cluster profile name
sz = str2num([getenv('SLURM_CPUS_PER_TASK')]);
if isempty(sz), sz = maxNumCompThreads; end
if isempty(gcp('nocreate')), c.parpool(sz); end

%Start timing
tic

%Calculates spectral radius of each matrix and displays
results
n = 10;
A = 500;
a = zeros(n);
parfor i = 1:n*n
    a(i) = max(abs(eig(rand(A)))));
end
disp(a)

%End timing
time = toc;

%Display the time of computation
fprintf('The parallel method is executed in %5.2f seconds.
\n', time);
```

Parallel implementation – Components



TEXAS TECH UNIVERSITY
Information Technology Division™

Link: /matlab/matlabPar

P1-

- Creates a cluster object representing the cluster identified by the cluster profile name *local*

P 1

Parallel version

Code:

```
c = parcluster('local'); % local is a cluster profile name
sz = str2num([getenv('SLURM_CPUS_PER_TASK')]);
if isempty(sz), sz = maxNumCompThreads; end
if isempty(gcp('nocreate')), c.parpool(sz); end

%Start timing
tic

%Calculates spectral radius of each matrix and displays
results
n = 10;
A = 500;
a = zeros(n);
parfor i = 1:n*n
    a(i) = max(abs(eig(rand(A)))));
end
disp(a)

%End timing
time = toc;

%Display the time of computation
fprintf('The parallel method is executed in %5.2f seconds.\n', time);
```

Parallel implementation – Components



TEXAS TECH UNIVERSITY
Information Technology Division

Link: /matlab/matlabPar

P1-

- Picks up *--cpus-per-task* from batch submit script
- Assigns to the variable *sz*
- String to number check to ensure correct format

P 1

Parallel version

Code:

```
c = parcluster('local'); % local is a cluster profile name
sz = str2num([getenv('SLURM_CPUS_PER_TASK')]);
if isempty(sz), sz = maxNumCompThreads; end
if isempty(gcp('nocreate')), c.parpool(sz); end

%Start timing
tic

%Calculates spectral radius of each matrix and displays
results
n = 10;
A = 500;
a = zeros(n);
parfor i = 1:n*n
    a(i) = max(abs(eig(rand(A)))));
end
disp(a)

%End timing
time = toc;

%Display the time of computation
fprintf('The parallel method is executed in %5.2f seconds.\n', time);
```

Parallel implementation – Components



TEXAS TECH UNIVERSITY
Information Technology Division™

Link: /matlab/matlabPar

P1-

- If *--cpus-per-task* not defined, maximum number of cores are allotted

P 1

Parallel version

Code:

```
c = parcluster('local'); % local is a cluster profile name
sz = str2num([getenv('SLURM_CPUS_PER_TASK')]);
if isempty(sz), sz = maxNumCompThreads; end
if isempty(gcp('nocreate')), c.parpool(sz); end

%Start timing
tic

%Calculates spectral radius of each matrix and displays
results
n = 10;
A = 500;
a = zeros(n);
parfor i = 1:n*n
    a(i) = max(abs(eig(rand(A)))));
end
disp(a)

%End timing
time = toc;

%Display the time of computation
fprintf('The parallel method is executed in %5.2f seconds.\n', time);
```

Parallel implementation – Components



TEXAS TECH UNIVERSITY
Information Technology Division

Link: /matlab/matlabPar

P1-

- If a parallel pool is not created, creates one and takes the size from *--cpus-per-task*

Takes some extra
time to initialize
the local cluster

P 1

Parallel version

Code:

```
c = parcluster('local'); % local is a cluster profile name
sz = str2num([getenv('SLURM_CPUS_PER_TASK')]);
if isempty(sz), sz = maxNumCompThreads; end
if isempty(gcp('nocreate')), c.parpool(sz); end

%Start timing
tic

%Calculates spectral radius of each matrix and displays
results
n = 10;
A = 500;
a = zeros(n);
parfor i = 1:n*n
    a(i) = max(abs(eig(rand(A)))));
end
disp(a)

%End timing
time = toc;

%Display the time of computation
fprintf('The parallel method is executed in %5.2f seconds.\n', time);
```

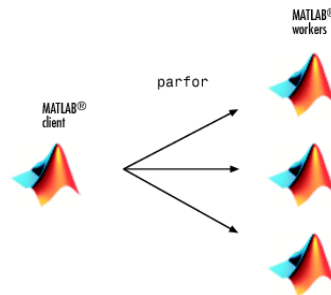
Parallel implementation – Components



TEXAS TECH UNIVERSITY
Information Technology Division™

Link: /matlab/matlabPar

P2- *parfor* splits the execution of for-loop iterations over the workers in a parallel pool



P 2

Parallel version

Code:

```
c = parcluster('local'); % local is a cluster profile name
sz = str2num([getenv('SLURM_CPUS_PER_TASK')]);
if isempty(sz), sz = maxNumCompThreads; end
if isempty(gcp('nocreate')), c.parpool(sz); end

%Start timing
tic

%Calculates spectral radius of each matrix and displays
results
n = 10;
A = 500;
a = zeros(n);
parfor i = 1:n*n
    a(i) = max(abs(eig(rand(A)))));
end
disp(a)

%End timing
time = toc;

%Display the time of computation
fprintf('The parallel method is executed in %5.2f seconds.\n', time);
```

Parallel implementation- Benefits



TEXAS TECH UNIVERSITY
Information Technology Division™

1st Run

Serial version

Parallel version

1 core	10 cores
1304.93 seconds(~21.75 mins)	133.62 seconds (~2.3 mins)

Next Run

Serial version

Parallel version

1 core	10 cores
10.31 seconds	1.89 seconds

Parallel implementation- Multi core parallel implementation



TEXAS TECH UNIVERSITY
Information Technology Division

Nocona – 128 cores/node

<https://www.depts.ttu.edu/hpcc/operations/equipment.php>

Partition:	Nocona	Quanah / XLQuanah	Matador	Toreador	Ivy
Type	CPU	CPU	GPU	GPU	Auxiliary CPU*
Total Nodes	240	467 / 16	20	11	50 / 2
Theoretical	983 TFLOPS	565 TFLOPS	280 TFLOPS	287 TFLOPS	40 TFLOPS
Max		/19 TFLOPS			
Benchmarked	804 TFLOPS	485 TFLOPS	226 TFLOPS		N/A
		/ (N/A)			
OS	CentOS 8.1	CentOS 7.4 /CentOS 8.1	CentOS 8.1	CentOS 8.1	Rocky Linux 8.5 /CentOS 8.1
Manufacturer	Dell	Dell	Dell	Dell	Dell
Node Model	PowerEdge C6525	PowerEdge C6320	PowerEdge R740	Poweredge R7525	PowerEdge C6220 II
Cooling	Liquid Cooled	Air Cooled	Air Cooled	Air Cooled	Air Cooled
Processor Make and Model	AMD EPYC™ 7702	Intel Xeon E5-2695 v4	Intel Xeon Gold 6242	AMD EPYC™ 7262	Intel Xeon E5-2670v2
Family	Rome	Broadwell	Cascade Lake	Rome	Ivy Bridge
Cores/Processor	64	18	20	8	10
Cores/Node	128	36	40 cpu + 1280 tensor + 10,240 CUDA	16 cpu + 1,296 tensor + 20,736 CUDA	20
Total Cores In Partition	30,720	16,812 / 576	800 cpu + 25,600 tensor + 224,800 CUDA	528 cpu + 14,256 tensor + 228,806 CUDA	1,000 / 40

MATLAB Graphical Interface



TEXAS TECH UNIVERSITY
Information Technology Division

Interactive session

```
$ interactive -p nocona -c 8
```

```
$ interactive -h
```

Number of cores

Setting up environment

```
$module load matlab
```

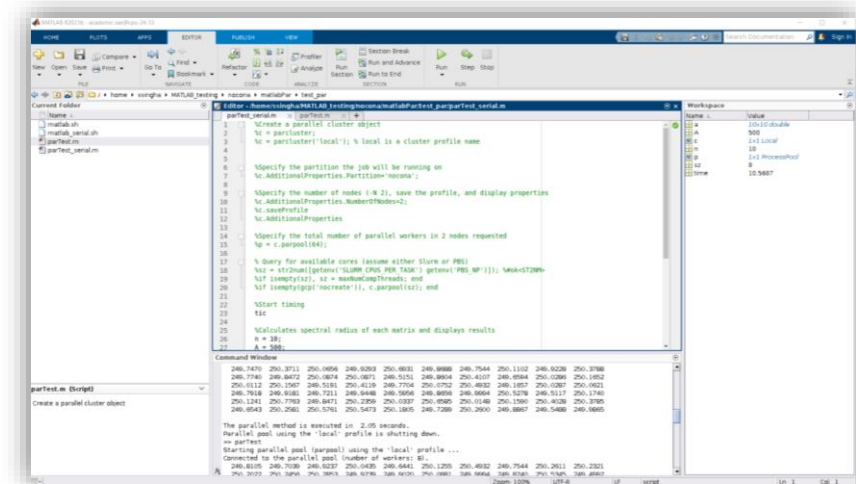
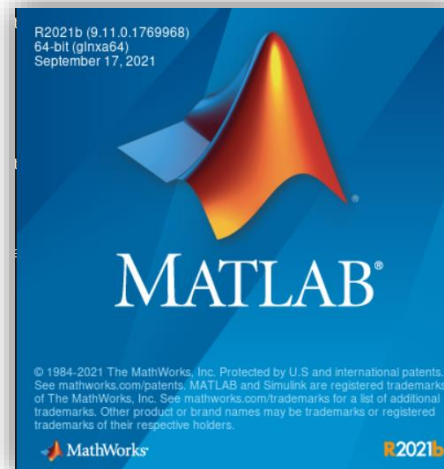
```
$ml list
```

Launching Client

```
$matlab
```

```
cpu-23-14:/matlab_shortcourse$ ml matlab  
cpu-23-14:/matlab_shortcourse$ ml list
```

```
Currently Loaded Modules:  
1) matlab/R2024a
```



Parallel implementation – Maximum Eigenvalue of a matrix



TEXAS TECH UNIVERSITY

Information Technology Division

Link: /matlab/matlabPar/parTest_interactive

Serial version

[parTest_serial.m](#)

Parallel version

[parTest.m](#)

Parallel implementation – Maximum Eigenvalue of a matrix



TEXAS TECH UNIVERSITY
Information Technology Division

Serial version

Code:

```
%Start timing
tic

%Calculates spectral radius of each matrix and
displays results
n = 10;
A = 500;
a = zeros(n);
for i = 1:n*n
    a(i) = max(abs(eig(rand(A)))));
end
disp(a)

%End timing
time = toc;

%Display the time of computation
fprintf('The parallel method is executed in %5.2f
seconds. \n', time);
```

Parallel version

Code:

```
c = parcluster('local'); % local is a cluster profile name

p = c.parpool(10);

%Start timing
tic

%Calculates spectral radius of each matrix and displays
results
n = 10;
A = 500;
a = zeros(n);
parfor i = 1:n*n
    a(i) = max(abs(eig(rand(A)))));
end
disp(a)

%End timing
time = toc;

%Display the time of computation
fprintf('The parallel method is executed in %5.2f seconds.
\n', time);
p.delete
```

Parallel implementation – Maximum Eigenvalue of a matrix



TEXAS TECH UNIVERSITY
Information Technology Division

Parallel version

Object representing cluster identified by the cluster profile name *local*

Starts a parallel pool of 10 workers using the cluster object *c*

- Splits the execution of for-loop iterations over the workers in a parallel pool
- Say $n \times n = 10$, 10 workers available, MATLAB assigns it to each core

- shuts down the parallel pool associated with the object *p*
- destroys the communicating job that comprises the pool

Code:

```
c = parcluster('local'); % local is a cluster profile name

p = c.parpool(10);

%Start timing
tic

%Calculates spectral radius of each matrix and displays
results
n = 10;
A = 500;
a = zeros(n);
parfor i = 1:n*n
    a(i) = max(abs(eig(rand(A)))));
end
disp(a)

%End timing
time = toc;

%Display the time of computation
fprintf('The parallel method is executed in %5.2f seconds.\n', time);
p.delete
```

MATLAB Graphical Interface – Parallel implementation via parpool



TEXAS TECH UNIVERSITY
Information Technology Division

Client outputs

```
>> parTest_serial  
...  
The parallel method is executed in 12.94 seconds.
```

```
>> parTest  
Starting parallel pool (parpool) using the 'local' profile ...  
Connected to the parallel pool (number of workers: 1).  
...  
The parallel method is executed in 10.69 seconds.  
Parallel pool using the 'local' profile is shutting down.
```

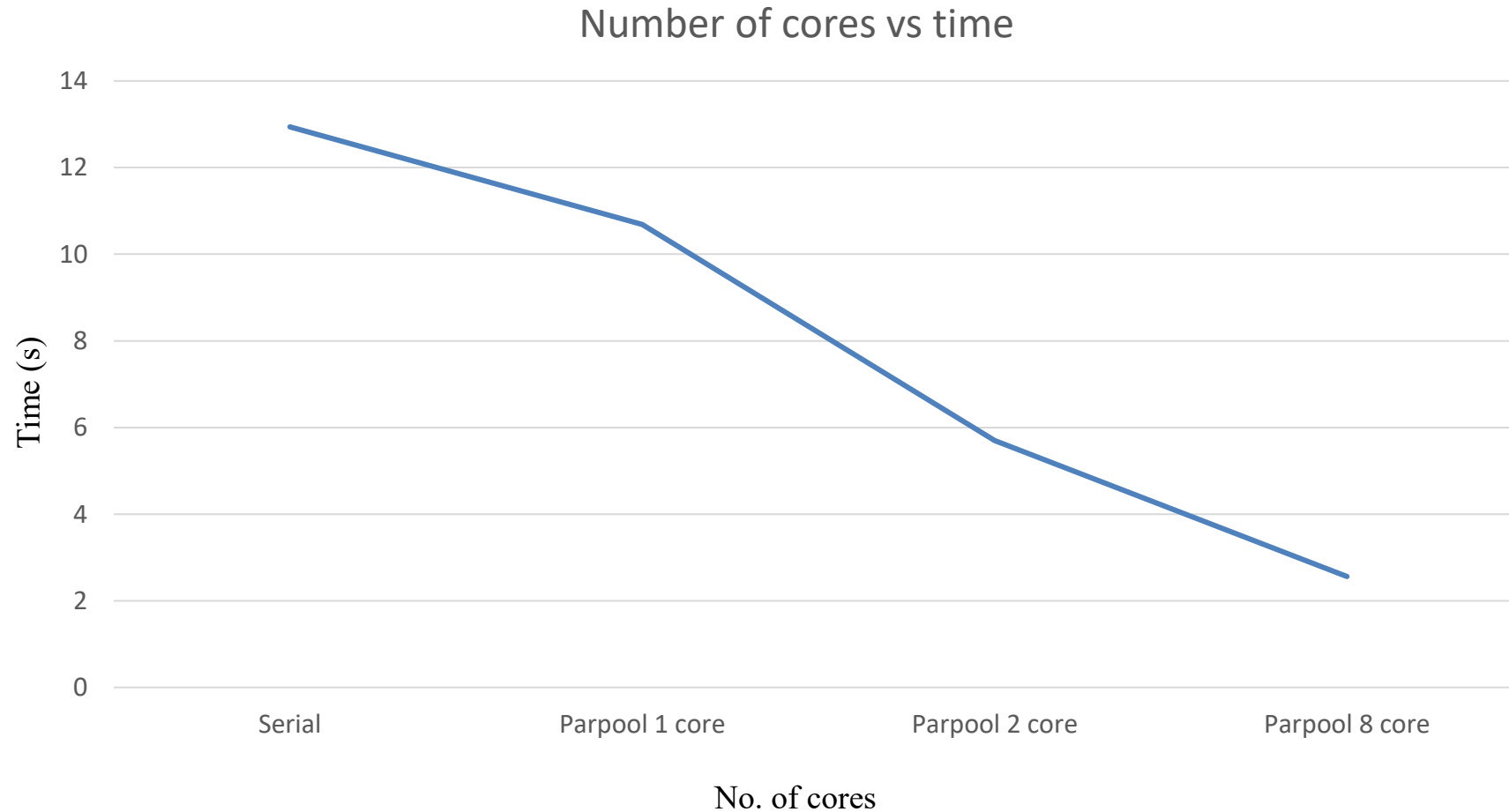
```
>>... Connected to the parallel pool (number of workers: 2).  
The parallel method is executed in 5.70 seconds ...
```

```
>>... Connected to the parallel pool (number of workers: 8).  
The parallel method is executed in 2.56 seconds ...
```

MATLAB Graphical Interface – Parallel implementation via parpool



TEXAS TECH UNIVERSITY
Information Technology Division



Parallel constructs in MATLAB



TEXAS TECH UNIVERSITY

Information Technology Division™

parfor

- Do not allow communication between workers
- Require synchronization

spmd

- To enable you require fine-grained worker-to-worker communication and collaboration between workers
- Require synchronization

parfeval

- Do not allow communication between workers
- Jobs can be run asynchronously

Options for parallel computing in MATLAB



TEXAS TECH UNIVERSITY
Information Technology Division

Serial version

Code:

```
%Start timing
tic

%Calculates spectral radius of each matrix and
displays results
n = 500;
A = 500;
a = zeros(n);
for i = 1:n
    a(i) = max(abs(eig(rand(i)))));
end
disp(a)

%End timing
time = toc;

%Display the time of computation
fprintf('The parallel method is executed in %5.2f
seconds. \n', time);
```

Options for parallel computing in MATLAB



TEXAS TECH UNIVERSITY
Information Technology Division

Parfor version

Code:

```
%Create a parallel cluster object
% local is a cluster profile name
c = parcluster('local');
p = c.parpool(4);

%Start timing
tic

%Calculates spectral radius of each matrix and displays results
%Distributed across several workers
n = 500;
A = 500;
a = zeros(n);
parfor i = 1:n
    a(i) = max(abs(eig(rand(i)))));
end

%End timing
time = toc

%Display the time of computation
fprintf('The parallel method is executed in %5.2f seconds. \n', time);

>Delete parallel workers
p.delete;
delete(p);
```

Options for parallel computing in MATLAB



TEXAS TECH UNIVERSITY
Information Technology Division

Parfeval version

Code:

```
c = parcluster('local');
p = c.parpool(8);

tic
n = 500;
size = 500;
a = zeros(n); % Initialize as a column vector for easier indexing
feig = @(size) max(abs(eig(rand(size))))); % Corrected anonymous function
input

futures = cell(n); % Cell array to store Future objects

for i = 1:n
    futures{i} = parfeval(feig, 1, i); % Store the Future object
end

% Retrieve the results from the Future objects
for i = 1:n
    a(i) = fetchOutputs(futures{i});
end

toc

%Delete parallel workers
p.delete;
delete(p);
```

Options for parallel computing in MATLAB



TEXAS TECH UNIVERSITY
Information Technology Division™

Spmd version

Code:

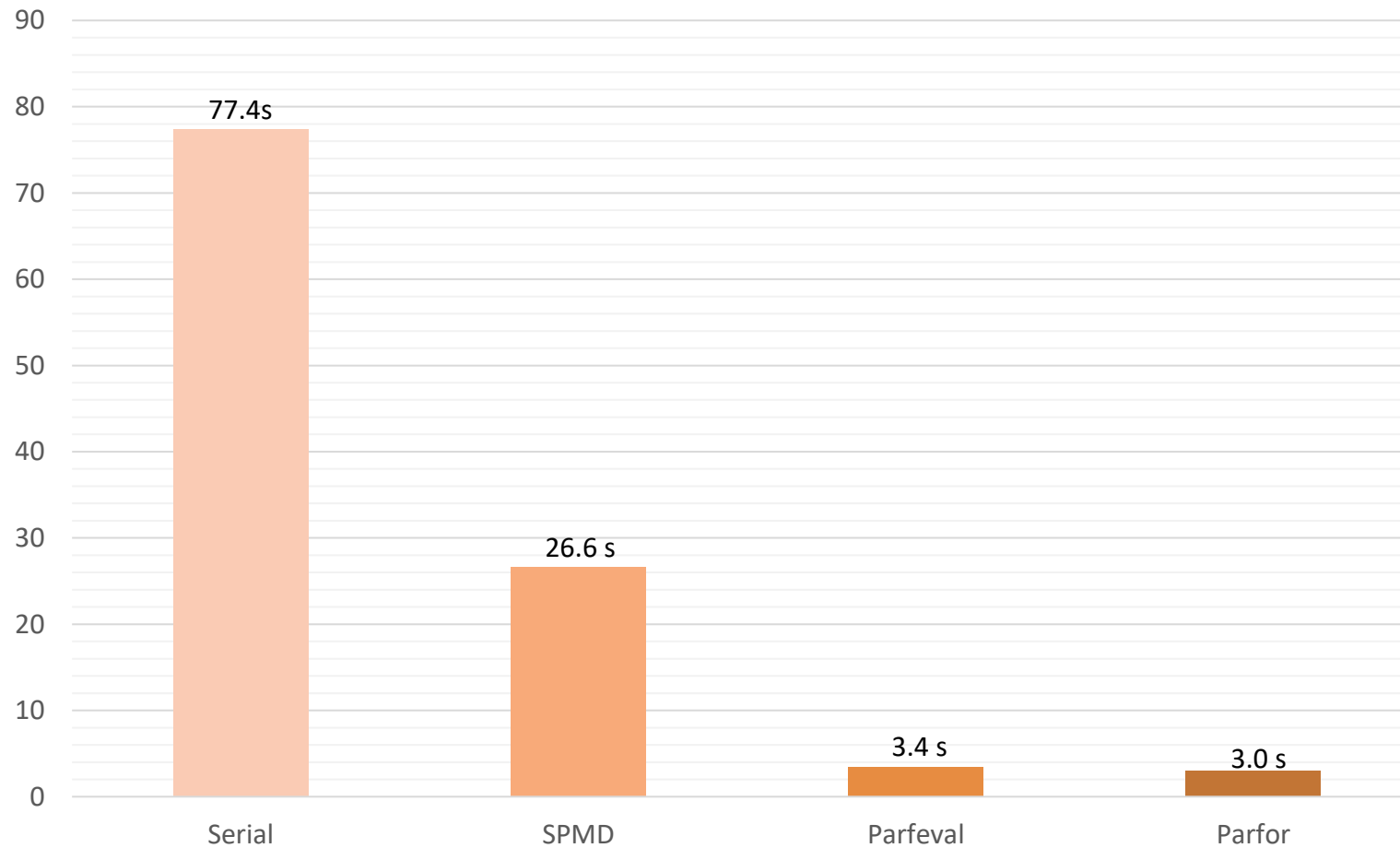
```
c = parcluster('local');  
p = c.parpool(10);  
  
tic  
  
spmd (10)  
    n = 500;  
    A = 500;  
    a = zeros(n);  
    for i = 1:n  
        a(i) = max(abs(eig(rand(i)))));  
    end  
end  
  
toc  
  
p.delete;  
delete(p);
```

Options for parallel computing in MATLAB



TEXAS TECH UNIVERSITY
Information Technology Division

Runtime comparison



MATLAB in the TTU HPCC - GPU Environment setup



TEXAS TECH UNIVERSITY
Information Technology Division

Interactive session

```
$ interactive -p matador -c 6
```

```
$ interactive -h
```

Setup MATLAB environment
module load matlab

Setting up environment

```
$module load matlab
```

```
$ml list
```

Version to load:
matlab/matlab/R2023b

Launching Client

```
$matlab
```

Check loaded modules:
module list

GPU Array



TEXAS TECH UNIVERSITY
Information Technology Division™

Serial version

Code:

```
%Start timing
tic

%Calculates spectral radius of each matrix and
displays results
n = 500;
A = 500;
a = zeros(n);
for i = 1:n
    a(i) = max(abs(eig(rand(A)))));
end
disp(a)

%End timing
time = toc;

%Display the time of computation
fprintf('The parallel method is executed in %5.2f
seconds. \n', time);
```

GPU Array



TEXAS TECH UNIVERSITY
Information Technology Division™

Parallel version

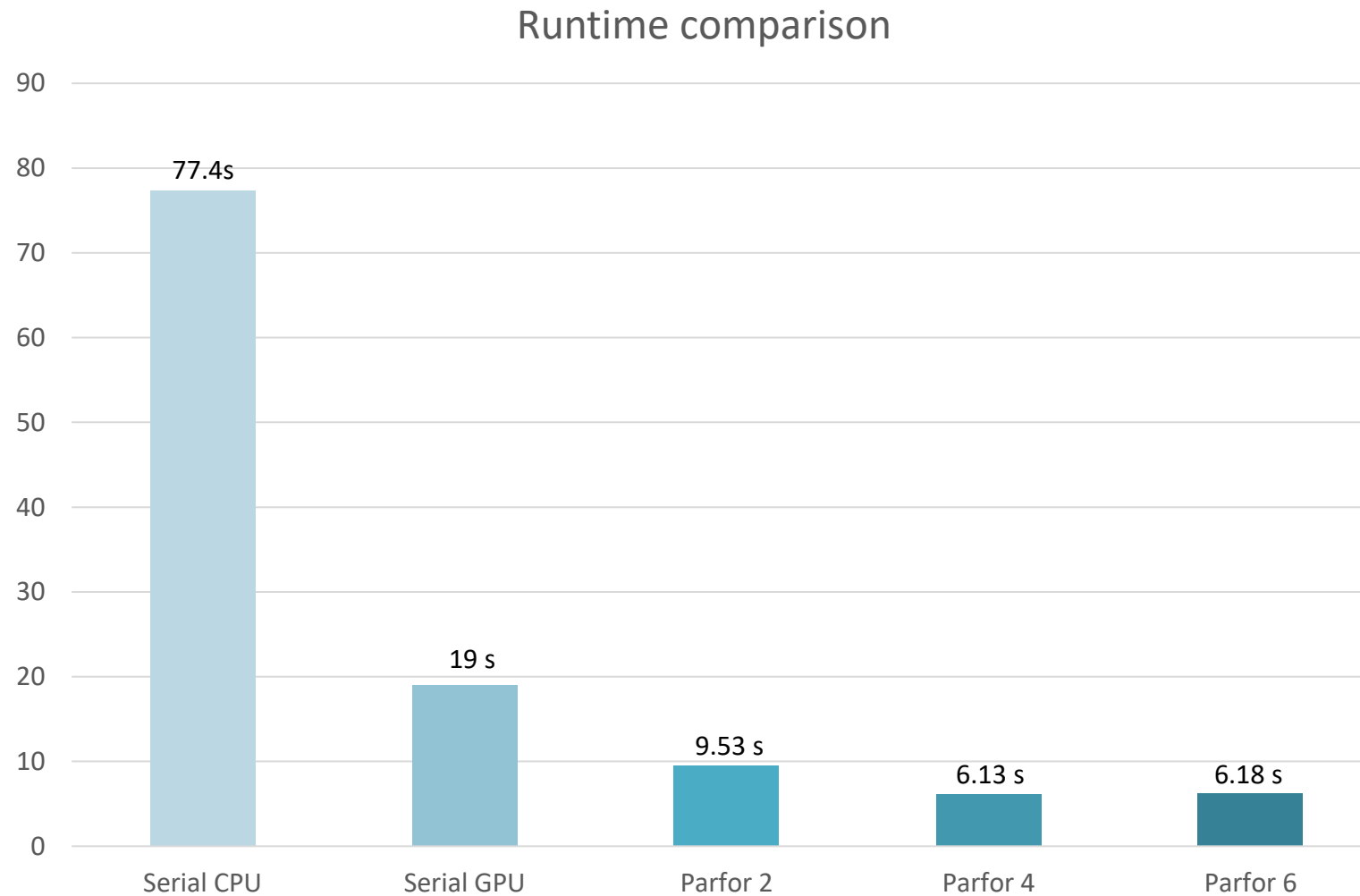
Code:

```
c = parcluster('local');  
p = c.parpool(2);  
  
%Start timing  
tic  
n = 500;  
A = 500;  
%a = zeros(n);  
a = zeros(n,"gpuArray");  
parfor i = 1:n  
    a(i) = max(abs(eig(rand(i)))));  
End  
  
%End timing  
time = toc;  
  
%Delete parallel workers  
p.delete;  
delete(p);
```

GPU Array



TEXAS TECH UNIVERSITY
Information Technology Division™



Introduction to MATLAB in HPC Environment



TEXAS TECH UNIVERSITY
Information Technology Division™

Break
Reconvene at 1 pm

Batch submission -

Prime Factorization – complexity $\sim$ magnitude of number



Information Technology Division™

Serial version

Code:

```
primeNumbers = primes(uint64(2^21));  
compositeNumbers =  
primeNumbers.*primeNumbers(randperm(numel(primeNumbers))));  
factors = zeros(numel(primeNumbers),2);  
  
%Start timing  
tic  
  
%Calculates the prime factors of composite numbers  
for idx = 1:numel(compositeNumbers)  
    factors(idx,:) = factor(compositeNumbers(idx));  
end  
%End timing  
time = toc;  
  
%Display the time of computation  
fprintf('The parallel method is executed in seconds=')  
disp(time)
```

Returns and array of
prime numbers

Batch submission - Prime Factorization



TEXAS TECH UNIVERSITY
Information Technology Division™

Serial version

Code:

```
primeNumbers = primes(uint64(2^21));  
compositeNumbers =  
primeNumbers.*primeNumbers(randperm(numel(primeNumbers))));  
factors = zeros(numel(primeNumbers),2);  
  
%Start timing  
tic  
  
%Calculates the prime factors of composite numbers  
for idx = 1:numel(compositeNumbers)  
    factors(idx,:) = factor(compositeNumbers(idx));  
end  
%End timing  
time = toc;  
  
%Display the time of computation  
fprintf('The parallel method is executed in seconds=')  
disp(time)
```

Returns no. of elements

Batch submission - Prime Factorization



TEXAS TECH UNIVERSITY
Information Technology Division™

Serial version

Code:

```
primeNumbers = primes(uint64(2^21));  
compositeNumbers =  
primeNumbers.*primeNumbers(randperm(numel(primeNumbers))));  
factors = zeros(numel(primeNumbers),2);  
  
%Start timing  
tic  
  
%Calculates the prime factors of composite numbers  
for idx = 1:numel(compositeNumbers)  
    factors(idx,:) = factor(compositeNumbers(idx));  
end  
%End timing  
time = toc;  
  
%Display the time of computation  
fprintf('The parallel method is executed in seconds=')  
disp(time)
```

Returns row matrix of
random permutation

Batch submission - Prime Factorization



TEXAS TECH UNIVERSITY
Information Technology Division™

Serial version

Code:

```
primeNumbers = primes(uint64(2^21));  
compositeNumbers =  
primeNumbers.*primeNumbers(randperm(numel(primeNumbers))));  
factors = zeros(numel(primeNumbers),2);  
  
%Start timing  
tic  
  
%Calculates the prime factors of composite numbers  
for idx = 1:numel(compositeNumbers)  
    factors(idx,:) = factor(compositeNumbers(idx));  
end  
%End timing  
time = toc;  
  
%Display the time of computation  
fprintf('The parallel method is executed in seconds=')  
disp(time)
```

Returns a null matrix

Batch submission - Prime Factorization



TEXAS TECH UNIVERSITY
Information Technology Division™

Serial version

Code:

```
primeNumbers = primes(uint64(2^21));  
compositeNumbers =  
primeNumbers.*primeNumbers(randperm(numel(primeNumbers))));  
factors = zeros(numel(primeNumbers),2);  
  
%Start timing  
tic  
  
%Calculates the prime factors of composite numbers  
for idx = 1:numel(compositeNumbers)  
    factors(idx,:) = factor(compositeNumbers(idx));  
end  
%End timing  
time = toc;  
  
%Display the time of computation  
fprintf('The parallel method is executed in seconds=')  
disp(time)
```

Returns prime factors

Batch submission - Prime Factorization



TEXAS TECH UNIVERSITY
Information Technology Division™

Serial version

Code:

```
primeNumbers = primes(uint64(2^21));  
compositeNumbers =  
primeNumbers.*primeNumbers(randperm(numel(primeNumbers))));  
factors = zeros(numel(primeNumbers),2);  
  
%Start timing  
tic  
  
%Calculates the prime factors of composite numbers  
for idx = 1:numel(compositeNumbers)  
    factors(idx,:) = factor(compositeNumbers(idx));  
end  
%End timing  
time = toc;  
  
%Display the time of computation  
fprintf('The parallel method is executed in seconds=')  
disp(time)
```

Returns and array of
prime numbers

Returns no. of elements

Returns row matrix of
random permutation

Returns a null matrix

Returns prime factors

Batch submission - Prime Factorization



TEXAS TECH UNIVERSITY
Information Technology Division™

The screenshot shows the MATLAB environment. The Editor window displays a script named 'example_4.m' with the following code:

```
1 primeNumbers = primes(uint64(2^21));  
2 compositeNumbers = primeNumbers.*primeNumbers(randperm(numel(primeNumbers))));  
3 factors = zeros(numel(primeNumbers),2);  
4  
5 tic;  
6 for idx = 1:numel(compositeNumbers)  
7     factors(idx,:) = factor(compositeNumbers(idx));  
8 end  
9 toc  
10  
11  
12
```

The Command Window shows the execution of the script:

```
>>  
>>  
>> example_4  
Elapsed time is 724.906721 seconds.  
>>  
>>  
>>  
>>  
>>  
>>
```

A red box highlights the value '724.906721 seconds.' in the Command Window, with a blue arrow pointing to it from the text box on the right.

Time of computation (serial)

Batch submission - Prime Factorization



TEXAS TECH UNIVERSITY
Information Technology Division™

Serial version

Code:

```
primeNumbers = primes(uint64(2^21));
compositeNumbers =
primeNumbers.*primeNumbers(randperm(numel(primeNumbers))));
factors = zeros(numel(primeNumbers),2);

%Start timing
tic

%Calculates the prime factors of composite numbers
for idx = 1:numel(compositeNumbers)
    factors(idx,:) = factor(compositeNumbers(idx));
end

%End timing
time = toc;

%Display the time of computation
fprintf('The serial method is executed in seconds=')
disp(time)
```

Parallel version

Code:

```
%Create a parallel cluster object
c = parcluster('local'); % local is a cluster profile name

p = c.parpool(10);

primeNumbers = primes(uint64(2^11));
compositeNumbers =
primeNumbers.*primeNumbers(randperm(numel(primeNumbers))));
factors = zeros(numel(primeNumbers),2);

%Start timing
tic;

%Calculates the prime factors of composite numbers
parfor idx = 1:numel(compositeNumbers)
    factors(idx,:) = factor(compositeNumbers(idx));
End

%End timing
time=toc;

%Display the time of computation
fprintf('The parallel method is executed in seconds=')
disp(time)
delete(gcp('ncreate'))
```

Batch submission - Benefits



TEXAS TECH UNIVERSITY
Information Technology Division

```
Editor - /home/ssingha/MATLAB_User_Training_Spring_2023/testing/matlab/Example_4/example_4.m
example_4.m
1 primeNumbers = primes(uint64(2^21));
2 compositeNumbers = primeNumbers.*primeNumbers(randperm(numel(primeNumbers))));
3 factors = zeros(numel(primeNumbers),2);
4
5 tic;
6 for idx = 1:numel(compositeNumbers)
7     factors(idx,:) = factor(compositeNumbers(idx));
8 end
9 toc
10
11
12
```

```
Command Window
>>
>> example_4
Elapsed time is 724.906721 seconds.
```

Time of computation (serial)

```
Editor - /home/ssingha/MATLAB_User_Training_Spring_2023/matlab/Example_4/example_4_parallel.m
example_4_parallel.m
1 %Create a parallel cluster object
2 c = parcluster('local'); % local is a cluster profile name
3 %sz = str2num([getenv('SLURM_CPUS_PER_TASK')]);
4 %if isempty(sz), sz = maxNumCompThreads; end
5 %if isempty(gcp('nocreate')), c.parpool(sz); end
6
7 p = c.parpool(10);
8
9 primeNumbers = primes(uint64(2^11));
10 compositeNumbers = primeNumbers.*primeNumbers(randperm(numel(primeNumbers))));
11 factors = zeros(numel(primeNumbers),2);
12
13 tic;
14 parfor idx = 1:numel(compositeNumbers)
15     factors(idx,:) = factor(compositeNumbers(idx));
16 end
17 time=toc;
18
19 %Display the time of computation
20 fprintf('The parallel method is executed in seconds=')
21 disp(time)
22
```

```
Command Window
>>
>>
>>
>>
>> example_4_parallel
Starting parallel pool (parpool) using the 'local' profile ...
Connected to the parallel pool (number of workers: 10).
The parallel method is executed in seconds= 0.3251

Parallel pool using the 'local' profile is shutting down.
>>
>>
```

Time of computation (parallel)

Parallel implementation- Multi core parallel implementation



TEXAS TECH UNIVERSITY
Information Technology Division

Nocona – 128 cores/node

<https://www.depts.ttu.edu/hpcc/operations/equipment.php>

Partition:	Nocona	Quanah / XLQuanah	Matador	Toreador	Ivy
Type	CPU	CPU	GPU	GPU	Auxiliary CPU*
Total Nodes	240	467 / 16	20	11	50 / 2
Theoretical	983 TFLOPS	565 TFLOPS	280 TFLOPS	287 TFLOPS	40 TFLOPS
Max		/19 TFLOPS			
Benchmarked	804 TFLOPS	485 TFLOPS	226 TFLOPS		N/A
		/ (N/A)			
OS	CentOS 8.1	CentOS 7.4 /CentOS 8.1	CentOS 8.1	CentOS 8.1	Rocky Linux 8.5 /CentOS 8.1
Manufacturer	Dell	Dell	Dell	Dell	Dell
Node Model	PowerEdge C6525	PowerEdge C6320	PowerEdge R740	Poweredge R7525	PowerEdge C6220 II
Cooling	Liquid Cooled	Air Cooled	Air Cooled	Air Cooled	Air Cooled
Processor Make and Model	AMD EPYC™ 7702	Intel Xeon E5-2695 v4	Intel Xeon Gold 6242	AMD EPYC™ 7262	Intel Xeon E5-2670v2
Family	Rome	Broadwell	Cascade Lake	Rome	Ivy Bridge
Cores/Processor	64	18	20	8	10
Cores/Node	128	36	40 cpu + 1280 tensor + 10,240 CUDA	16 cpu + 1,296 tensor + 20,736 CUDA	20
Total Cores In Partition	30,720	16,812 / 576	800 cpu + 25,600 tensor + 204,800 CUDA	528 cpu + 14,256 tensor + 222,816 CUDA	1,000 / 40

Job submission – Asynchronous Batch GUI



TEXAS TECH UNIVERSITY
Information Technology Division

- Run once
- Re-runs deletes older profile, creates a new profile
- All saved data deleted

- `>>c=parcluster;`
- Default cluster profile is set as **redraider**
- Always makes a new secondary job submission to Slurm

- `>>c=parcluster('local');`
- Opens cluster profile to utilize present allocation
- Uses just resources on present node, no submission to scheduler

```
Command Window
>> configCluster

Must set Partition and NumberOfNodes before submitting jobs to REDRAIDER. E.g.

>> c = parcluster;
>> c.AdditionalProperties.Partition = 'nocona';
>> c.AdditionalProperties.NumberOfNodes = 1;
>> c.saveProfile

Complete. Default cluster profile set to "redraider R2021b".
>> c=parcluster

c =

Generic Cluster

Properties:

    Profile: redraider R2021b
    Modified: false
    Host: login-20-25.localdomain
    NumWorkers: 100000
    NumThreads: 1

    JobStorageLocation: /home/ssingha/.matlab/3p_cluster_jobs/redraider/R2021b/shared
    ClusterMatlabRoot: /opt/apps/snfs/RedRaider/matlab/R2021b
    OperatingSystem: unix

RequiresOnlineLicensing: false
PluginScriptsLocation: /home/ssingha/MATLAB_testing/testing/RE_[TTU_HPCC]_MATLAB_and_Singularity/IntegrationScr...
AdditionalProperties: List properties

Associated Jobs:

    Number Pending: 0
    Number Queued: 0
    Number Running: 2
    Number Finished: 4

fx >> |
```

Job submission – Asynchronous Batch GUI



TEXAS TECH UNIVERSITY
Information Technology Division™

Profile Properties

- Slurm properties that can be modified
- *AdditionalSubmitArgs* – Manually allows user to pass arguments, flags say constrains of the cluster
- `c.AdditionalProperties.NumberOfNodes = 1;`
- `c.AdditionalProperties.Partitions='nocona';`
- `c.saveProfile`
- Creates a submission script and performs the submission for the user

A screenshot of a Windows Command Window titled "Command Window". The window shows a series of commands and their outputs. The first command is a list of Slurm properties: `AdditionalSubmitArgs: ''`, `Constraint: ''`, `EmailAddress: ''`, `EnableDebug: 0`, `GpusPerNode: 0`, `MemUsage: '4gb'`, `NumberOfNodes: 1`, `Partition: ''`, `ProcsPerNode: 0`, `RequireExclusiveNode: 0`, `Reservation: ''`, `UseSmpd: 0`, and `WallTime: ''`. The next command is `>> t`, which results in an error: `Unrecognized function or variable 't'.`. This is followed by a series of commands: `>>`, `>> c.AdditionalProperties.NumberOfNodes=1;`, `>> c.AdditionalProperties.Partition='nocona';`, `>> c.saveProfile`, `>>`, and `>> c.AdditionalProperties`. The output shows `ans =` followed by a list of properties: `AdditionalProperties` with properties: `AccountName: ''`, `AdditionalSubmitArgs: ''`, `Constraint: ''`, `EmailAddress: ''`, `EnableDebug: 0`, `GpusPerNode: 0`, `MemUsage: '4gb'`, `NumberOfNodes: 1`, `Partition: 'nocona'`, `ProcsPerNode: 0`, `RequireExclusiveNode: 0`, `Reservation: ''`, `UseSmpd: 0`, and `WallTime: ''`. The window ends with a prompt `>>`.

Job submission – Asynchronous Batch GUI



TEXAS TECH UNIVERSITY
Information Technology Division

Job object name Function name Number of outputs

`>>j=batch(c,@pwd,1,{});`

Inputs to function

`>>j.State` – returns status of job

`>>j.fetchOutputs` – returns output of job

`>>j.diary` – returns command line outputs

```
Command Window
>> j=batch(c,@pwd,1,{});
additionalSubmitArgs =
    '--ntasks=1 --cpus-per-task=1 --ntasks-per-core=1 -p nocona --mem-per-cpu=4gb --nodes=1'
>> j.State
ans =
    'finished'
>> j.fetchOutputs
ans =
    1x1 cell array
    {'/home/ssingha/MATLAB_testing/testing/RE_[TTU_HPCC]_MATLAB_and_Singularity'}
>> j=batch(c,@parallel_example,1,{300},'pool',10);
additionalSubmitArgs =
    '--ntasks=11 --cpus-per-task=1 --ntasks-per-core=1 -p nocona --mem-per-cpu=4gb --nodes=1'
>> j.State
ans =
    'running'
>> j.State
ans =
    'finished'
>> j.fetchOutputs
ans =
    1x1 cell array
```



Job submission – Asynchronous Batch GUI

Serial Example

```
function [t, A] = serial_example(iter)
if nargin==0
    iter = 8;
end
disp('Start sim')
t0 = tic;
for idx = 1:iter
    A(idx) = idx;
    pause(2)
    idx
end
t = toc(t0);
disp('Sim completed')
save RESULTS A
end
```

```
>> j=batch(c,@serial_example,1,{8});
```

```
>> j.State
```

```
>> j.fetchOutputs
```

```
ans =
    1×1 cell array
    {[16.0098]}
```

Job submission – Asynchronous Batch GUI



TEXAS TECH UNIVERSITY
Information Technology Division

Parallel Example

```
function [t, A] = parallel_example(iter)
if nargin==0
    iter = 8;
end
disp('Start sim')
t0 = tic;
parfor idx = 1:iter
    A(idx) = idx;
    pause(2)
    idx
end
t = toc(t0);
disp('Sim completed')
save RESULTS A
end
```

```
>> j=batch(c,@serial_example,1,{8});
```

```
{[16.0098]}
```

```
>>j=batch(c,@parallel_example,1,{8},'pool',2);
{[8.3045]}
```

```
>> =batch(c,@parallel_example,1,{8},'pool',4);
{[4.3124]}
```

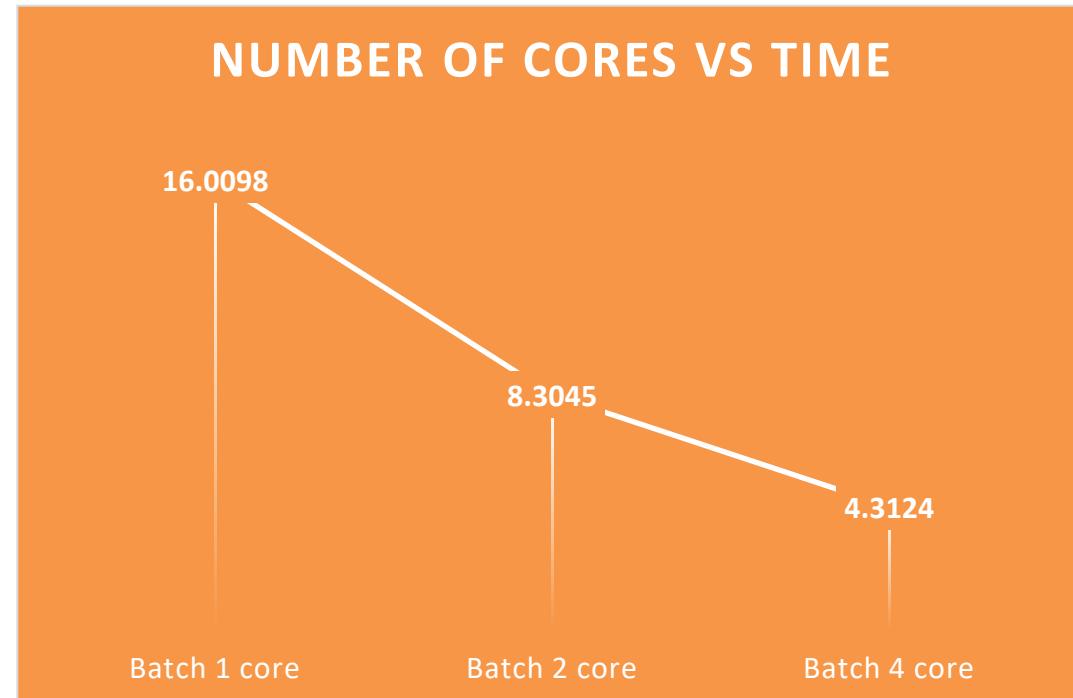
Job submission – Asynchronous Batch GUI



TEXAS TECH UNIVERSITY
Information Technology Division™

Parallel Example

```
function [t, A] = parallel_example(iter)
if nargin==0
    iter = 8;
end
disp('Start sim')
t0 = tic;
parfor idx = 1:iter
    A(idx) = idx;
    pause(2)
    idx
end
t = toc(t0);
disp('Sim completed')
save RESULTS A
end
```



Job submission – Asynchronous Batch GUI



TEXAS TECH UNIVERSITY
Information Technology Division

Job Arrays

```
>> j=batch(c,@sum,1,{[1 1]});
```

```
>> j=createJob(c);
```

```
createTask(j,@sum,1,{[1 1]});
```

```
createTask(j,@sum,1,{[1 1]});
```

```
createTask(j,@sum,1,{[1 1]});
```

```
submit(j);
```

```
>>
>> j=createJob(c);
createTask(j,@sum,1,{[1 1]});
createTask(j,@sum,1,{[1 1]});
createTask(j,@sum,1,{[1 1]});
submit(j);

additionalSubmitArgs =

    '--ntasks=1 --cpus-per-task=1 --ntasks-per-core=1 -p nocona --mem-per-cpu=4gb --nodes=1'

>> j.Tasks

ans =

3x1 Task array:

      ID      State      FinishDateTime  Function  Errors  Warnings  SchedulerID
-----
1      1    running
2      2    pending
3      3    pending
      sum          0          0      7632475_1
      sum          0          0      7632475_2
      sum          0          0      7632475_3

>>
>>
>>
```

Job submission – Asynchronous Batch GUI



TEXAS TECH UNIVERSITY
Information Technology Division

Job Arrays

>> j.Tasks

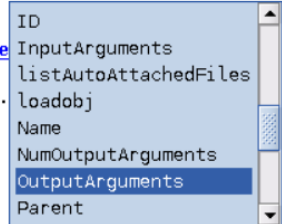
```
>> j.Tasks
ans =
3x1 Task array:
      ID      State      FinishDateTime      Function      Errors      Warnings
-----
1      1      pending
2      2      pending
3      3      pending
      sum      0      0
      sum      0      0
      sum      0      0
```

>> j.Tasks(2).cancel
>> j.Tasks(1).State

```
>> j.Tasks(2).cancel
>> j.Tasks
ans =
3x1 Task array:
      ID      State      FinishDateTime      Function      Errors      Warnings
-----
1      1      pending
2      2      finished      09-Nov-2022 00:11:14      sum      1      0
3      3      pending
      sum      0      0
      sum      0      0
```

Monitoring job parameters

```
ans =
1x0 empty cell array
>> j.Tasks(1).loadobj
ans =
1
>> j.Tasks(1).
```

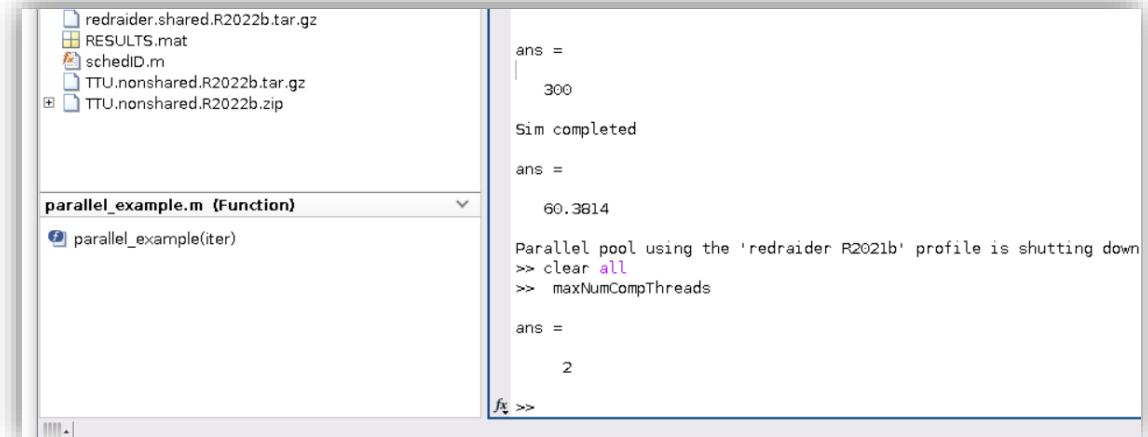
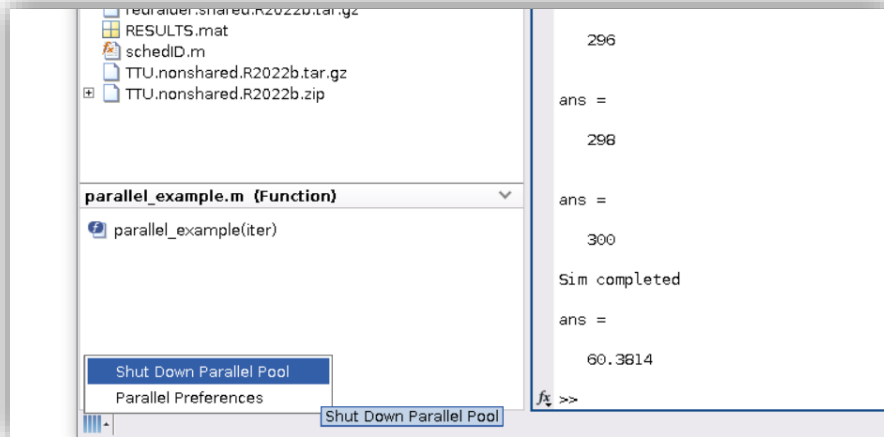


Job submission – Interactive pool submission



TEXAS TECH UNIVERSITY
Information Technology Division

Shutting down Parallel pool



>>clear all

Job submission – Interactive pool submission



TEXAS TECH UNIVERSITY
Information Technology Division

Monitoring

The screenshot displays the MATLAB Parallel Computing Toolbox Job Monitor interface. The top toolbar includes options for HOME, PLOTS, and APPS, along with various file and workspace management tools. The Job Monitor window is open, showing a table of job submissions. The table has columns for ID, Username, Submit Time, Finish Time, and Description. The jobs are listed in descending order of submit time. The status of each job is indicated by a colored circle and the text 'finished', 'pending', or 'queued'. The Command Window shows the execution of the 'parallel_example' function, and the Workspace window displays the variables 'ans', 'c', 'j', and 'p'.

ID	Username	Submit Time	Finish Time	Description
1	ssingha	Fri Oct 28 14:26:40 CDT 2022	Fri Oct 28 14:26:56 CDT 2022	Batch job running function
2	ssingha	Fri Oct 28 14:36:56 CDT 2022	Fri Oct 28 14:38:25 CDT 2022	Batch job running function
3	ssingha	Fri Oct 28 14:46:09 CDT 2022	Fri Oct 28 14:46:24 CDT 2022	Independent job
5	ssingha	Fri Oct 28 15:09:42 CDT 2022	Fri Oct 28 15:14:19 CDT 2022	Batch job running function
6	ssingha	Fri Oct 28 15:14:55 CDT 2022	Fri Oct 28 15:16:21 CDT 2022	Batch job running function
8	ssingha	Mon Nov 07 18:31:45 CST 2022	Mon Nov 07 18:32:01 CST 2022	Batch job running function
9	ssingha	Mon Nov 07 18:33:56 CST 2022	Mon Nov 07 18:35:26 CST 2022	Batch job running function
10	ssingha			Independent job
11	ssingha	Mon Nov 07 18:53:59 CST 2022	Mon Nov 07 18:54:11 CST 2022	Independent job
12	ssingha			Independent job
13	ssingha	Mon Nov 07 18:54:55 CST 2022	Mon Nov 07 18:55:07 CST 2022	Independent job
14	ssingha	Mon Nov 07 20:18:34 CST 2022		Batch job running function

Last updated at Mon Nov 07 20:20:24 CST 2022

Auto update: Every 5 minutes | Update Now

Current Folder: +pctDebug, IntegrationScripts, cleanJobStorageLocation.m, clusterDefinition.m, configCluster.m, Getting_Started_With_Serial_And_Parallel_MATL..., jobStorageLocation.m, machinefile.7632325, machinefile.7632326, parallel_example.m (Function), parallel_example(iter)

Command Window: >> j.State, ans = 'queued', >> j.State, ans = 'queued'

Workspace: ans (1x1 Generic), c (1x1 CJSCommun...), j (1x1 ProcessPool), p (1x1 ProcessPool)

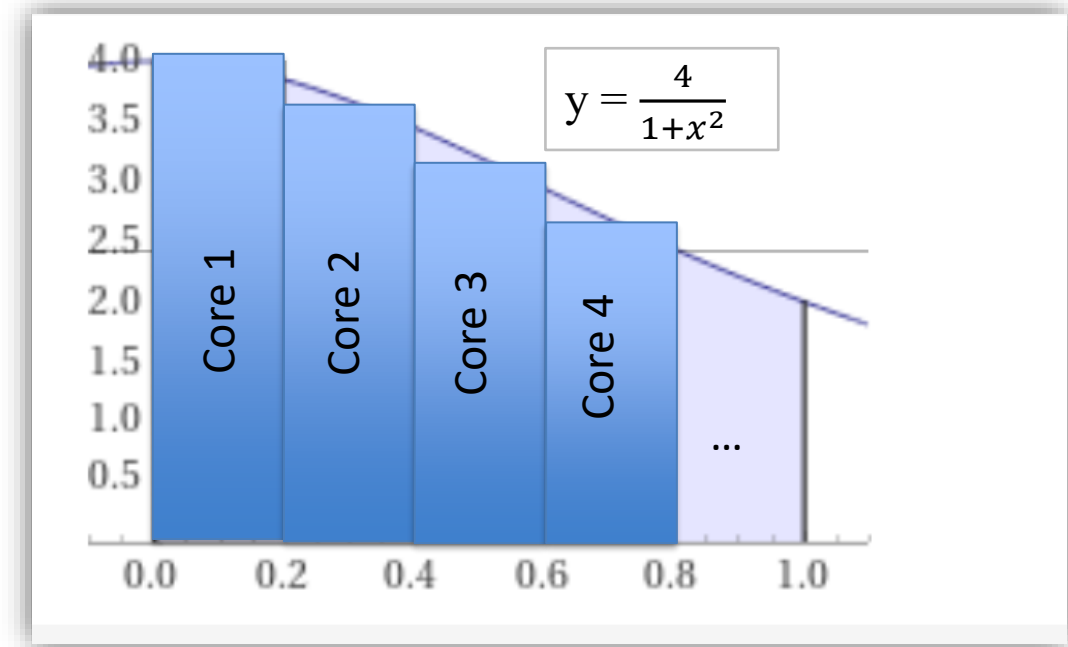
Parallel job implementation example



TEXAS TECH UNIVERSITY
Information Technology Division™

Calculation of π
(based on Leibniz formula)

$$\int_0^1 \frac{4}{1+x^2} dx = 4 \arctan = \pi$$



Plotted in Wolfram Alpha

Parallel job implementation example



TEXAS TECH UNIVERSITY
Information Technology Division

Single node parallel

```
function calc_pi

c = parcluster('local');

%sz = str2num([getenv('SLURM_CPUS_PER_TASK')]); %#ok<ST2NM>
sz = str2num([getenv('SLURM_CPUS_PER_TASK')]); %#ok<ST2NM>
if isempty(sz), sz = maxNumCompThreads; end

if isempty(gcp('ncreate')), c.parpool(sz); end

spmd
    a = (labindex - 1)/numlabs;
    b = labindex/numlabs;
    fprintf('Subinterval: [%-4g, %-4g]\n', a, b)

    myIntegral = integral(@quadpi, a, b);
    fprintf('Subinterval: [%-4g, %-4g]    Integral: %4g\n', a, b, myIntegral)

    piApprox = gplus(myIntegral);
end

approx1 = piApprox{1}; % 1st element holds value on worker 1
fprintf('pi          : %.18f\n', pi)
fprintf('Approximation: %.18f\n', approx1)
fprintf('Error          : %g\n',    abs(pi - approx1))

function y = quadpi(x)
%QUADPI Return data to approximate pi.

% Derivative of 4*atan(x)
y = 4./(1 + x.^2);
```

Multi node parallel

```
function calc_pi_multi_node

c = parcluster;

% Required fields
c.AdditionalProperties.WallTime = '00:20:00';
c.AdditionalProperties.Partition = 'nocona';
c.AdditionalProperties.NumberOfNodes = 1;

if isempty(gcp('ncreate')), c.parpool(20); end

spmd
    a = (labindex - 1)/numlabs;
    b = labindex/numlabs;
    fprintf('Subinterval: [%-4g, %-4g]\n', a, b)

    myIntegral = integral(@quadpi, a, b);
    fprintf('Subinterval: [%-4g, %-4g]    Integral: %4g\n', a, b, myIntegral)

    piApprox = gplus(myIntegral);
end

approx1 = piApprox{1}; % 1st element holds value on worker 1
fprintf('pi          : %.18f\n', pi)
fprintf('Approximation: %.18f\n', approx1)
fprintf('Error          : %g\n',    abs(pi - approx1))

function y = quadpi(x)
%QUADPI Return data to approximate pi.

% Derivative of 4*atan(x)
y = 4./(1 + x.^2);
```

Parallel job implementation example



TEXAS TECH UNIVERSITY
Information Technology Division

Accuracy of numerical calculation

```
>> calc_pi
Starting parallel pool (parpool) using the 'local' profile ...
Connected to the parallel pool (number of workers: 4).
Worker 1:
  Subinterval: [0 , 0.25]
  Subinterval: [0 , 0.25]   Integral: 0.979915
Worker 2:
  Subinterval: [0.25, 0.5 ]
  Subinterval: [0.25, 0.5 ]   Integral: 0.874676
Worker 3:
  Subinterval: [0.5 , 0.75]
  Subinterval: [0.5 , 0.75]   Integral: 0.719414
Worker 4:
  Subinterval: [0.75, 1 ]
  Subinterval: [0.75, 1 ]   Integral: 0.567588
pi          : 3.141592653589793116
Approximation: 3.141592653589793560
Error       : 4.44089e-16
```

Lower

```
>> calc_pi_multi_node
Starting parallel pool (parpool) using the 'teton R2022a' profile ...

additionalSubmitArgs =

    '--ntasks=20 --cpus-per-task=1 --ntasks-per-core=1

Connected to the parallel pool (number of workers: 20).
Worker 1:
  Subinterval: [0 , 0.025]
Worker 2:
  Subinterval: [0.025, 0.05]
Worker 3:
  Subinterval: [0.05, 0.075]
...
Worker 18:
  Subinterval: [0.425, 0.45]   Integral: 0.0839331
Worker 20:
  Subinterval: [0.525, 0.55]   Integral: 0.0775848
pi          : 3.141592653589793116
Approximation: 3.141592653589793116
Error       : 0
```

Lower

Containerized MATLAB



TEXAS TECH UNIVERSITY
Information Technology Division™

1. Login into login.hpcc.ttu.edu using your eRaider account and password.
2. Get an interactive session on a partition (say nocona)
\$ interactive -p nocona
3. Check singularity installation and version
\$ singularity --version
singularity version 1.4.3-1.el9
4. Pull a sample singularity container to get started
\$ singularity pull shub://singularityhub/hello-world

```
cpu-25-27:/singularity_testing/nocona$ singularity pull shub://singularityhub/hello-world
INFO:   Downloading shub image
59.8MiB / 59.8MiB [=====] 100 % 44.5 MiB/s 0s
cpu-25-27:/singularity_testing/nocona$ ls
hello-world_latest.sif
```

Containerized MATLAB



TEXAS TECH UNIVERSITY
Information Technology Division™

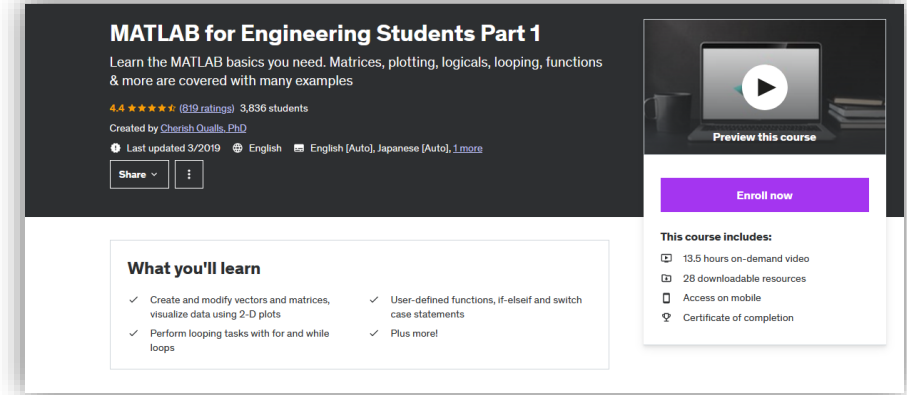
1. In the event of errors, re-runs execute
\$ singularity cache list
to list container files, present in the account memory
\$ singularity cache clean
to remove such files before beginning a fresh pull
2. Execute the singularity image file (*.sif)
\$./hello-world_latest.sif
Tacotacotaco
3. Alternate ways to explore yourself:
Directly run a singularity container
\$ singularity run hello-world_latest.sif
4. Pull latest MATLAB docker container:
\$ singularity pull docker://mathworks/matlab:r2022b

Additional Resources



TEXAS TECH UNIVERSITY
Information Technology Division™

- Udemy –
 - <https://ttu.udemy.com/course/matlab-the-need-to-know-basics/>



MATLAB for Engineering Students Part 1
Learn the MATLAB basics you need. Matrices, plotting, logicals, looping, functions & more are covered with many examples

4.4 ★★★★★ (819 ratings) 3,836 students
Created by [Cherish Qualls, PhD](#)
Last updated 3/2019 English English [Auto], Japanese [Auto], 1 more

Share

What you'll learn

- ✓ Create and modify vectors and matrices, visualize data using 2-D plots
- ✓ Perform looping tasks with for and while loops
- ✓ User-defined functions, if-elseif and switch case statements
- ✓ Plus more!

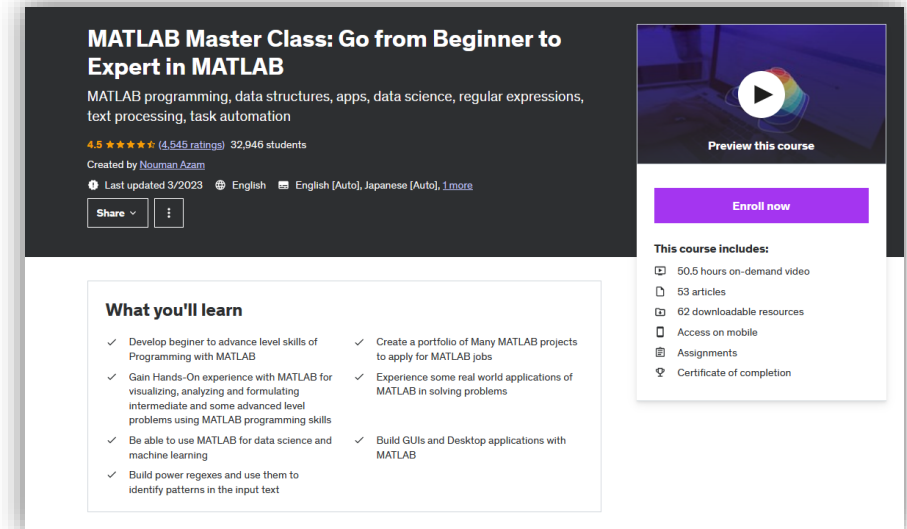
This course includes:

- 13.5 hours on-demand video
- 28 downloadable resources
- Access on mobile
- Certificate of completion

Preview this course

Enroll now

- <https://ttu.udemy.com/course/matlab-essentials-for-engineering-and-science-students/>



MATLAB Master Class: Go from Beginner to Expert in MATLAB
MATLAB programming, data structures, apps, data science, regular expressions, text processing, task automation

4.5 ★★★★★ (4,545 ratings) 32,946 students
Created by [Nouman Azam](#)
Last updated 3/2023 English English [Auto], Japanese [Auto], 1 more

Share

What you'll learn

- ✓ Develop beginner to advance level skills of Programming with MATLAB
- ✓ Gain Hands-On experience with MATLAB for visualizing, analyzing and formulating intermediate and some advanced level problems using MATLAB programming skills
- ✓ Be able to use MATLAB for data science and machine learning
- ✓ Build power regexes and use them to identify patterns in the input text
- ✓ Create a portfolio of Many MATLAB projects to apply for MATLAB jobs
- ✓ Experience some real world applications of MATLAB in solving problems
- ✓ Build GUIs and Desktop applications with MATLAB

This course includes:

- 50.5 hours on-demand video
- 53 articles
- 62 downloadable resources
- Access on mobile
- Assignments
- Certificate of completion

Preview this course

Enroll now

National Testbed Allocations - ACCESS



TEXAS TECH UNIVERSITY

Information Technology Division

ACCESS Resources

CPU Computing

Anvil (Purdue)—1,000 AMD Milan nodes, 128 cores per node, large memory nodes available

Bridges-2 (PSC)—504 AMD Rome nodes, 128 cores per node, large memory nodes available; extreme memory (4 TB) nodes allocated separately

Delta (UIUC)—124 AMD Milan nodes, 128 cores per node

Expanse (SDSC)—728 AMD Rome nodes, 128 cores & 1 TB NVMe per node

KyRIC (U Kentucky)—Large-memory (3 TB, 6 TB) nodes, 300TB storage

Stampede3 (TACC) Intel Skylake, Ice Lake, Sapphire Rapids w/HBM H100 nodes

GPU Computing

Anvil GPU (Purdue)—16 nodes, 4 NVIDIA A100 GPUs each

Bridges-2 GPU (PSC) —10 nodes w/ 8x NVIDIA H100 GPUs, 24 nodes w/ 8x V100-32GB GPUs, 9 nodes w/ 8x NVIDIA V100 16GB GPUs

DeltaAI(UIUC) —152 nodes w/ 4x NVIDIA GH200-96GB

Delta GPU (NCSA)—4 node configs: 100 nodes w/ 4x A100s; 100 w/ 4x A40 GPUs; 5 w/ 8x A100s; 1 w/ 8x AMD MI100 GPUs

Expanse GPU (SDSC)—52 nodes, 4 NVIDIA V100 GPUs each

National Testbed Allocations - ACCESS



TEXAS TECH UNIVERSITY

Information Technology Division

Allocation Types

- **Code characteristics:** Core-hours, Node-hours, GPU-hours
- *Credits*
- Types
- **Explore** 400, 000 credits
 - Getting started, evaluating resources, dissertations, small-scale activities.
 - Grad students with a letter from their PI.
- **Discover** 1.5 million credits
 - 1-page document with science and computational description,
 - In house review 1 day-1 week
 - Grad students with a letter from their PI.
- **Accelerate** 3 million credits
 - 3 pg. doc science and computational methods, research plan, scaling, research team
 - External reviewers few days - few weeks
- **Maximize** 10 million credits
 - 10-page proposal, larger panel of external reviewers twice a year, 100 proposals /year of this scale
 - Detailed doc of code runs, scaling, assure reviewers of funding for using resources!

Note: They try to help. Emphasis on folks who have done good scaling runs, have run on ACCESS machines at a smaller scale and now are sure that they require larger resource pool.

National Testbed Allocations - DOE



TEXAS TECH UNIVERSITY

Information Technology Division

DOE Resources

CPU and GPU Computing

Argonne LCF

- **Polaris** - 560 - 1 x AMD Milan CPU + 4x NVIDIA A100 GPU
560 CPUs + 2,240 GPUs
- **Aurora** - 10,624 - 2 x Intel Xeon SPR +6 x Intel PVC GPU
21,248 CPUs + 63,744 GPUs

Oak Ridge LCF

- **Frontier** - 9,408 - 1 x AMD EYPC CPU + 4 x AMD MI250x GPU
9,408 CPUs + 37,362 GPUs

NERSC

- **Perlmutter** -
 - 3,072 CPU only nodes - 2 AMD Milan per node
 - 1,792 GPU Accelerated Nodes with 4 NVIDIA A100 + 1 AMD Milan per node.6144 CPUs and 1792 CPUs + 7,168 GPUs

National Testbed Allocations - DOE



TEXAS TECH UNIVERSITY

Information Technology Division

Allocation Types

- **Code characteristics:** Core-hours, Node-hours, GPU-hours
- ALCF (and OLCF)
 - **Director's [Discretionary](#)** - Relatively new to High-Performance Computing (HPC).
 - **[INCITE](#)** –Large-scale, computationally intensive projects that address “*grand challenges*” in science and engineering. **Cycle:** Mid-April and mid to late June. **Review:** Two-part peer review by international experts and a computational-readiness review. **Time:** Awarded for one to three years.
 - **[ALCC](#)** –High-risk, high-payoff scientific campaigns in areas directly related to the DOE mission e.g., Biophysics, Energy Efficiency in Aerospace and Combustion technologies, Fusion Energy, Geosciences, High Energy Physics, Materials Sciences, Nuclear Engineering, and Nuclear Physics. Current [awards](#). **Cycle:** July 1 – June 30th of the subsequent year. **Review:** Like INCITE. **Time:** Possibility of renewal for up to 2 years.
- NERSC
 - **Education, Exploratory:** Small allocations (~250 CPU Node Hours) to support research projects to explore using HPC and for educational efforts to train the next-generation workforce. For first-time PIs to try out NERSC resources. Decision within 4 weeks of applying (pending DOE approval for Exploratory) valid till 18 months from award.

National Testbed Allocations - NAIRR



TEXAS TECH UNIVERSITY

Information Technology Division

National Artificial Intelligence Research Resource Pilot

- [Aim](#): Advancing AI methods that enable scientific discovery, accelerating through AI enabled automation, new experimental methods, integrating simulations and AI, developing open-source AI tools, models, datasets, and methods, training AI-savvy workforce.
- Start-Up projects are awarded for three months, May 16, 2025 through the end of the NAIRR Pilot program.
- [Resources](#)
- [Instructions](#) for writing a proposal



Thoughts ?

Feedback

- Course assessment survey.
- Your feedback is incredibly valuable, it helps us enhance your learning experience.
- Your insights guide us in making our training course even better.