Getting Started with Parallel Computing using MATLAB on the LiDO3 HPC Cluster

This document provides the steps to configure MATLAB to submit jobs to a cluster, retrieve results, and debug errors.

CONFIGURATION

After logging into the cluster, configure MATLAB to run parallel jobs on your cluster by calling the shell script `configCluster.sh` This only needs to be called once per version of MATLAB.

```
$ module load matlab
$ configCluster.sh
```

Jobs will now default to the cluster rather than submit to the local machine.

CONFIGURING JOBS

Prior to submitting the job, we can specify various parameters to pass to our jobs, such as queue, e-mail, walltime, etc. *Only MemUsage is required.*

```
>> % Get a handle to the cluster
>> c = parcluster;

[REQUIRED]

>> % Specify memory to use for MATLAB jobs, per core (MB)
>> c.AdditionalProperties.MemUsage = '4000';

[OPTIONAL]

>> % Request to run on nodes with a specific constraint (e.g. GPU)
>> c.AdditionalProperties.Constraint = 'tesla';

>> % Specify e-mail address to receive notifications about your job
>> c.AdditionalProperties.EmailAddress = 'user-id@tu-dortmund.de';

>> % Request 1 Tesla GPU card per node
>> c.AdditionalProperties.GpusPerNode = 1;

>> % Specify a queue to use for MATLAB jobs
>> c.AdditionalProperties.QueueName = 'queue-name';

>> % Request exclusive nodes
>> c.AdditionalProperties.RequireExclusiveNode = true;

>> % Specify the walltime (e.g. 5 hours)
```

```
>> c.AdditionalProperties.WallTime = '05:00:00';
```

Save changes after modifying AdditionalProperties for the above changes to persist between MATLAB sessions.

```
>> c.saveProfile
```

To see the values of the current configuration options, display AdditionalProperties.

```
>> % To view current properties
>> c.AdditionalProperties
```

Unset a value when no longer needed.

```
>> % Turn off email notifications
>> c.AdditionalProperties.EmailAddress = '';
>> c.saveProfile
```

INTERACTIVE JOBS

To run an interactive pool job on the cluster, continue to use `parpool` as you've done before.

```
>> % Get a handle to the cluster
>> c = parcluster;

>> % Open a pool of 64 workers on the cluster
>> p = c.parpool(64);
```

Rather than running local on the local machine, the pool can now run across multiple nodes on the cluster.

```
>> % Run a parfor over 1000 iterations
>> parfor idx = 1:1000
       a(idx) = …
   end
```

Once we're done with the pool, delete it.

TO LEARN MORE

To learn more about the MATLAB Parallel Computing Toolbox, check out these resources:

- Parallel Computing Coding Examples
- Parallel Computing Documentation
- Parallel Computing Overview

- Parallel Computing Tutorials
- Parallel Computing Videos
- Parallel Computing Webinars