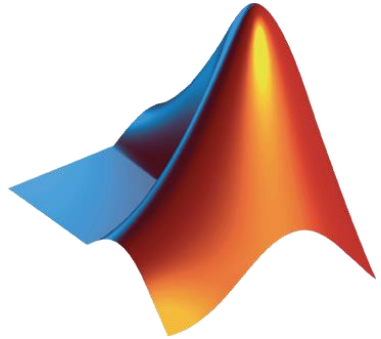
Workshop: Parallel Computing With MATLAB (Part II)

Raymond Norris
Application Engineer, MathWorks
May 2021



Agenda

- Part I – Parallel Computing with MATLAB on the Desktop
  - Parallel Computing Toolbox
  - MATLAB Online
- Part II – Scaling MATLAB to magnitUDE
  - MATLAB Parallel Server
  - MATLAB Desktop

https://www.uni-due.de/css/magnitude_software

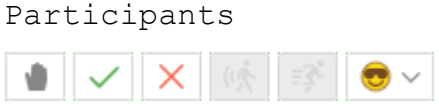
Agenda

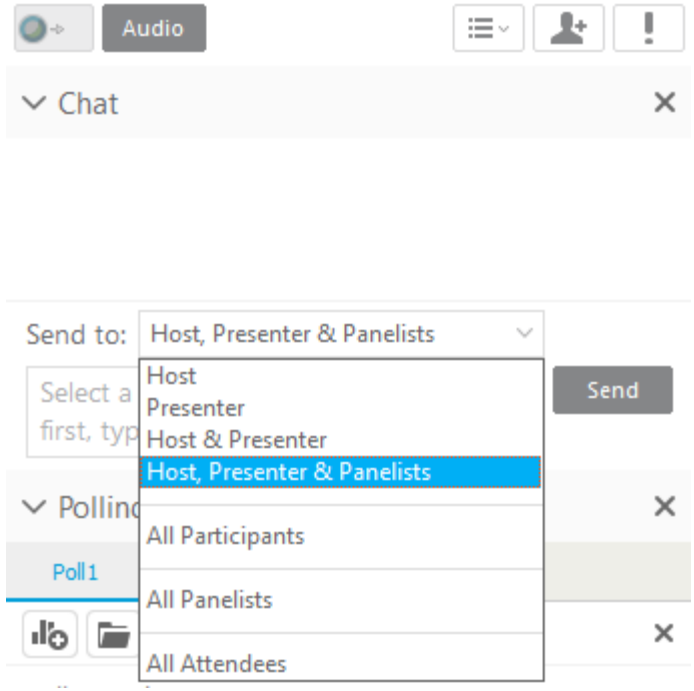
- Part I – Parallel Computing with MATLAB on the Desktop
  - Parallel Computing Toolbox
  - MATLAB Online
- Part II – Scaling MATLAB to magnitUDE
  - MATLAB Parallel Server
  - MATLAB Desktop

https://www.uni-due.de/css/magnitude_software

Chatting

- Send to at least the *Host, Presenter & Panelists*
- Ideally, send to *All Attendees*





Required: MATLAB & Parallel Computing Toolbox

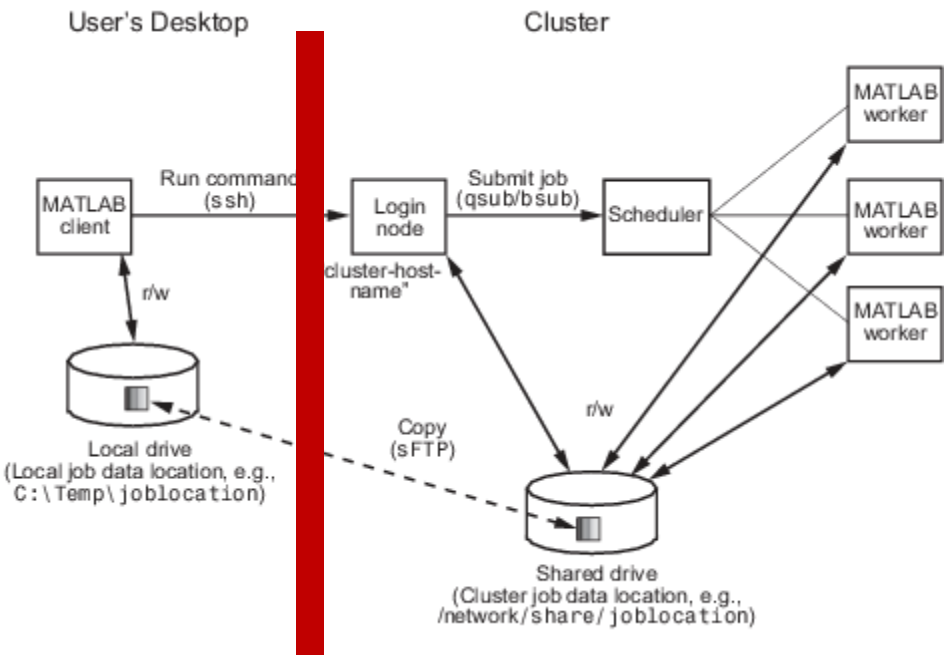
```
>> license('test','distrib_computing_toolbox')
ans =
     1
```

R2021a

R2020b

Scaling MATLAB to magnitUDE (1)

Your machine



Cluster

```
>> job1 = batch( );
>> job2 = batch( );
```

#SBATCH …

module load matlab

matlab …

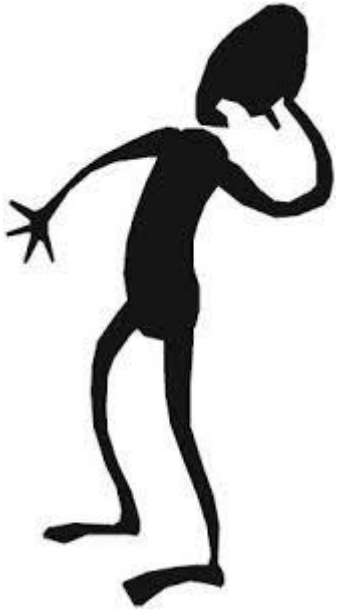
Desktop vs on-cluster submission

- Today's workshop will focus on desktop submission
- For on-cluster submissions, look at Getting Started Guide (not covered today)

https://www.uni-due.de/css/magnitude_software

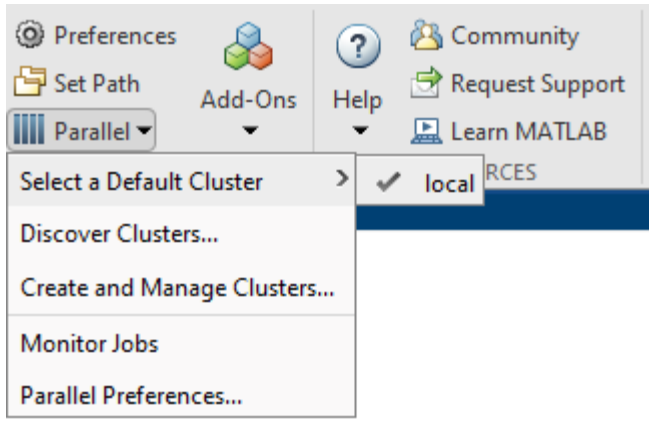
A note about today's workshop…

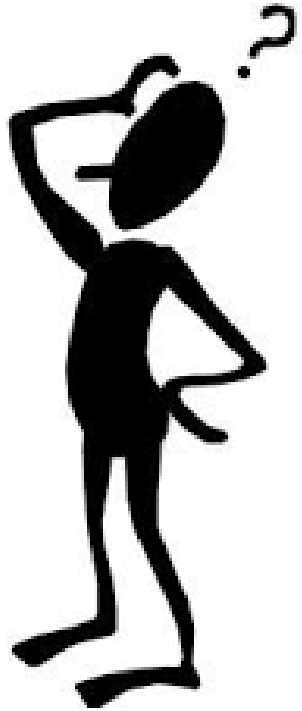
- The workflow and examples are about process, not performance



`local` profile

"How does MATLAB know about magnitUDE?"





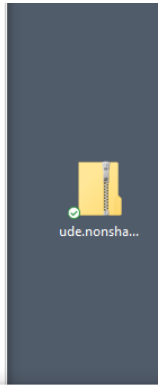
Download the magnitUDE MATLAB support package

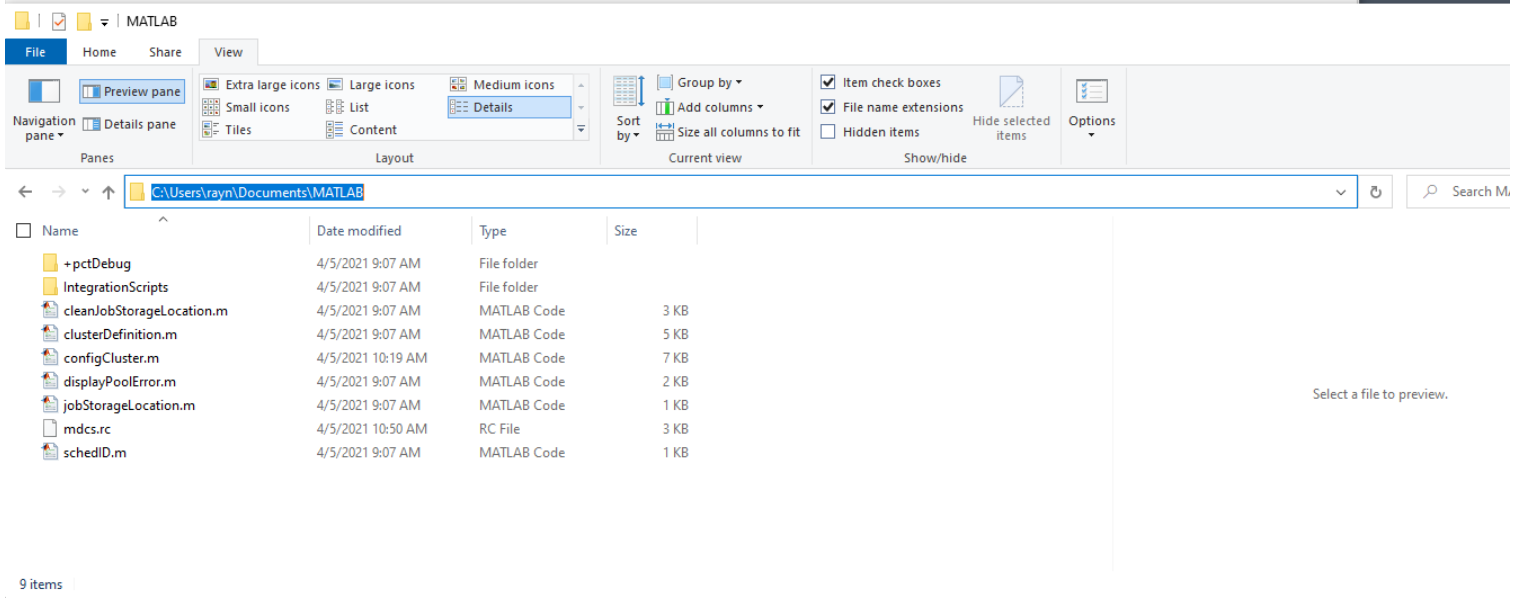
- https://www.uni-due.de/css/magnitude_software

- Provides job hooks for
  - ❑ submission (sbatch)
  - ❑ state (squeue)
  - ❑ deletion (scancel)

Install the support package – Windows

```
>> userpath

ans =

    'c:\Users\rayn\Documents\MATLAB'

>>
>>
```





Install the support package – Linux/macOS

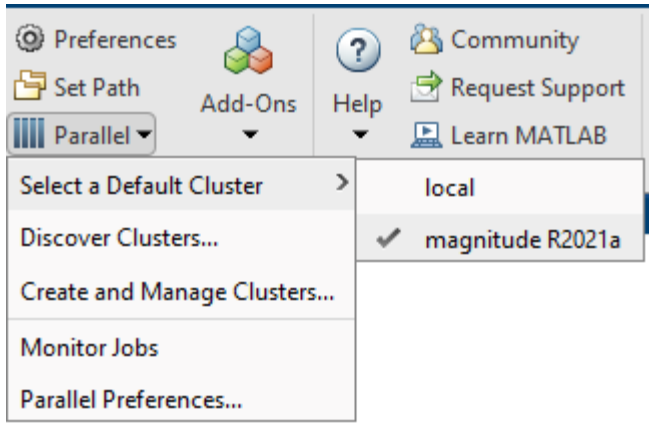
```
-bash4.4
-bash4.4 mkdir -p $HOME/Documents/MATLAB
-bash4.4
-bash4.4 tar xf ~/ude.nonshared.R2021a.tar.gz -C ~/Documents/MATLAB
-bash4.4
-bash4.4 ls -1 ~/Documents/MATLAB
cleanJobStorageLocation.m
clusterDefinition.m
configCluster.m
configHostname.m
displayPoolError.m
IntegrationScripts
jobStorageLocation.m
mdcs.rc
+pctDebug
schedID.m
-bash4.4
```

Configure MATLAB to create magnitUDE profile

Replace with your ZIM user-id

```
>> configCluster
Username on MAGNITUDE (e.g. joe): your-ZIM-user-id
>>
```

New magnitUDE profile



Only call `configCluster` once

MATLAB job submitters

- `parpool`
  - Single session
  - Synchronous execution
  - Seamlessly runs `parfor`, `parfeval`, and `spmd`

https://www.mathworks.com/help/parallel-computing/parpool.html

- `batch`
  - Multiple submissions
  - Non-blocking
  - Calls top-level function or script
  - Requires API to extract results

https://www.mathworks.com/help/parallel-computing/batch.html

Workshop: Add reservation

```
>> c = parcluster;
>> c.AdditionalProperties.Reservation = 'MATLAB';
>>
```

Reservation is only good for today's workshop

Exercise: “Hello, World!”


```
>> % Submit job to magnitUDE to find out where MATLAB is running
>> c = parcluster;
>> j = c.batch(@pwd,1,{}, 'AutoAddClientPath',false, 'CurrentFolder','.');
additionalSubmitArgs =
    '--ntasks=1 --cpus-per-task=1 --ntasks-per-core=1'
>>
```

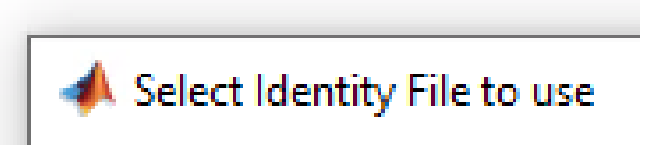
#SBATCH …

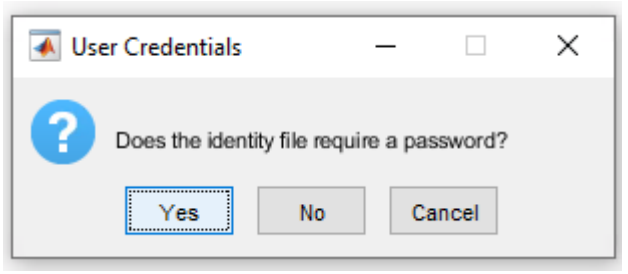
module load matlab

matlab …

If no profile is supplied to `parcluster`, use the default profile

SSH credentials (private key and passphrase)



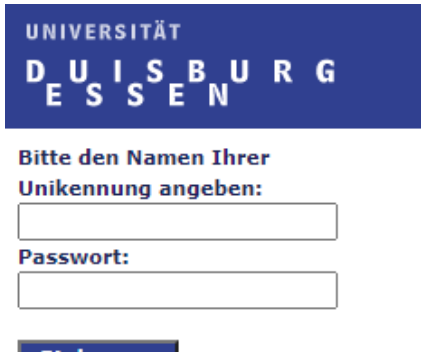


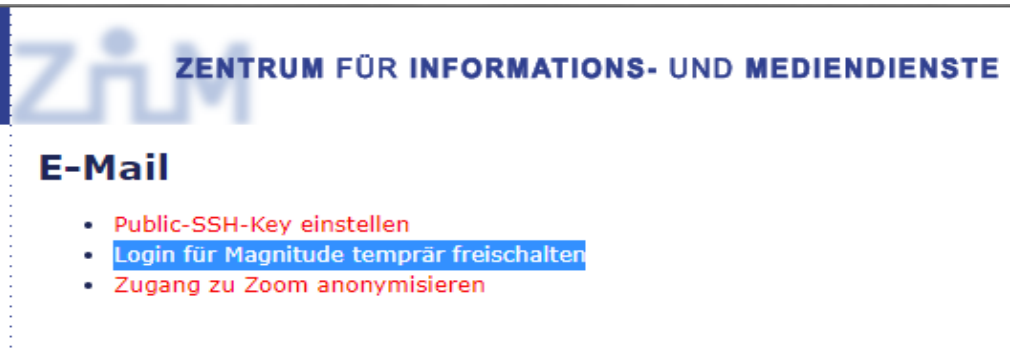
Two factor authentication

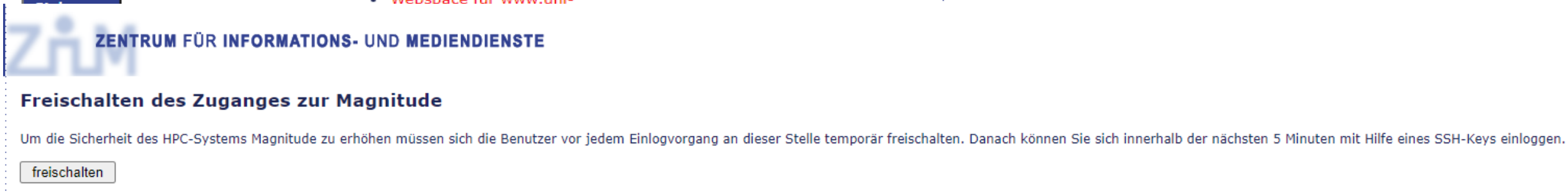
```
>> j = c.batch(@pwd,1,{}, 'AutoAddClientPath',false, 'CurrentFolder','.');
Error using parallel.Cluster/batch (line 158)
Job submission failed because the plugin function 'independentSubmitFcn.m' errored.
Caused by:
    Error using getRemoteConnection (line 137)
    Could not connect to remote host login01.magnitude.uni-due.de.
        Error using parallel.cluster.JavaRemoteClusterAccess/connect (line 332)
        Could not create a leasable connection: Auth fail.
>>
```

SSH connection to magnitUDE

- Request SSH access (within 5 minutes of request)
- https://www.uni-due.de/css/magnitude_nutzung.php
  - https://benutzerverwaltung.uni-duisburg-essen.de/portal
    - Einstellungen
    - Login für Magnitude temprär freischalten







SSH connection to magnitUDE

- Request SSH access (within 5 minutes of request)
- https://www.uni-due.de/css/magnitude_nutzung.php
  - https://benutzerverwaltung.uni-duisburg-essen.de/portal
    - Settings
    - Activate temporary login for magnitude
    - Unlock

“Hello, World!”

```
>> % Submit job to magnitUDE to find out where MATLAB is running
>> c = parcluster;
>> j = c.batch(@pwd,1,{}, 'AutoAddClientPath',false, 'CurrentFolder','.');
additionalSubmitArgs =
    '--ntasks=1 --cpus-per-task=1 --ntasks-per-core=1'
>>
```

Fetching results

```
>> % Submit job to magnitUDE to find out where MATLAB is running
>> c = parcluster;
>> j = c.batch(@pwd,1,{}, 'AutoAddClientPath',false, 'CurrentFolder','.');
additionalSubmitArgs =
    '--ntasks=1 --cpus-per-task=1 --ntasks-per-core=1'
>>
>> % Check the state of the job
>> j.State
ans =
    'finished'
>>
>> % Fetch the results
>> j.fetchOutputs{:}
ans =
    '/homes/adj403w'
>>
```

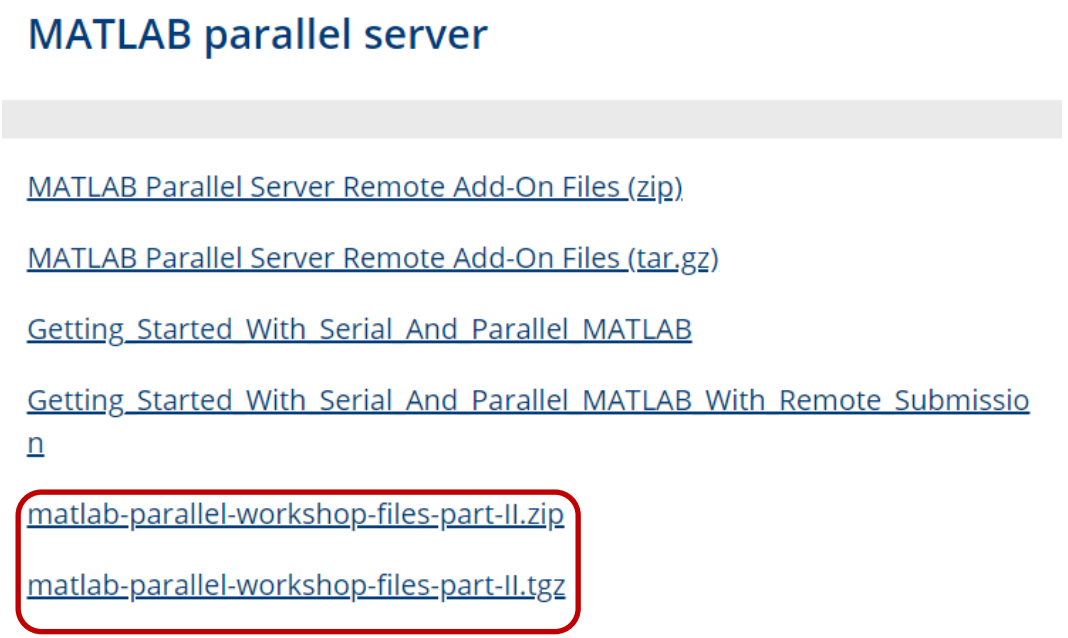
Benign warning if *CurrentFolder* isn't set

```
>> j.fetchOutputs{:}
Warning: The task with ID 1 issued the following warnings:
  Warning: Worker unable to change folder to
  '\\fs-2.uni-due.de\adj403w\Dokumente\MATLAB\matlab-workshop' at the start of the batch job.
  The job will be executed from '/homes/adj403w'. To execute from a different folder use the
  'CurrentFolder' parameter of batch. To suppress this warning, set 'CurrentFolder' to '.'.

ans =
    '/homes/adj403w'
>>
```

Download the workshop files

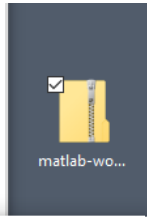
- https://www.uni-due.de/css/magnitude_software

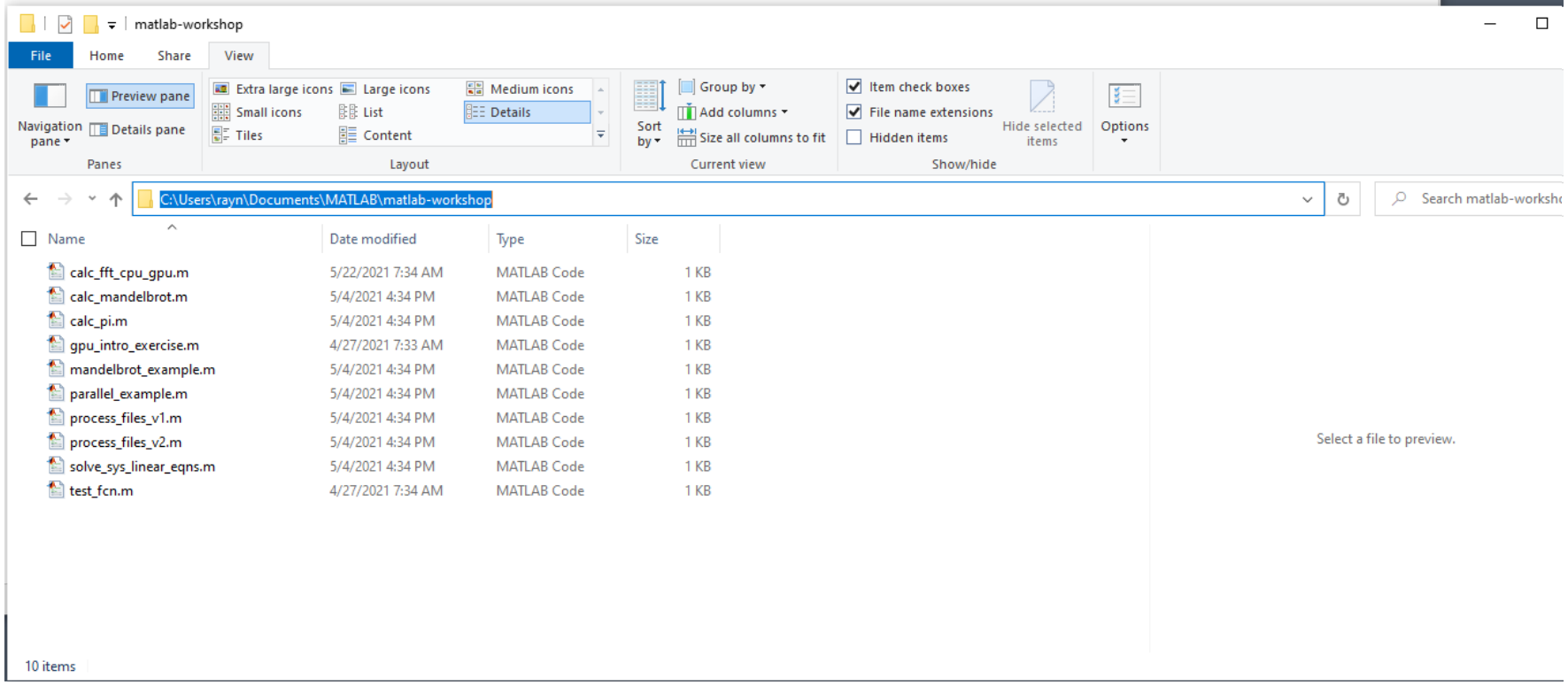
```
calc_fft_cpu_gpu.m
calc_mandelbrot.m
calc_pi.m
mandelbrot_example.m
parallel_example.m
process_files_v1.m
process_files_v2.m
solve_sys_linear_eqns.m
test_fcn.m
```



Install workshop files – Windows

```
>> userpath
ans =
     'c:\Users\rayn\Documents\MATLAB'
>>
```





Install workshop files – Linux/macOS

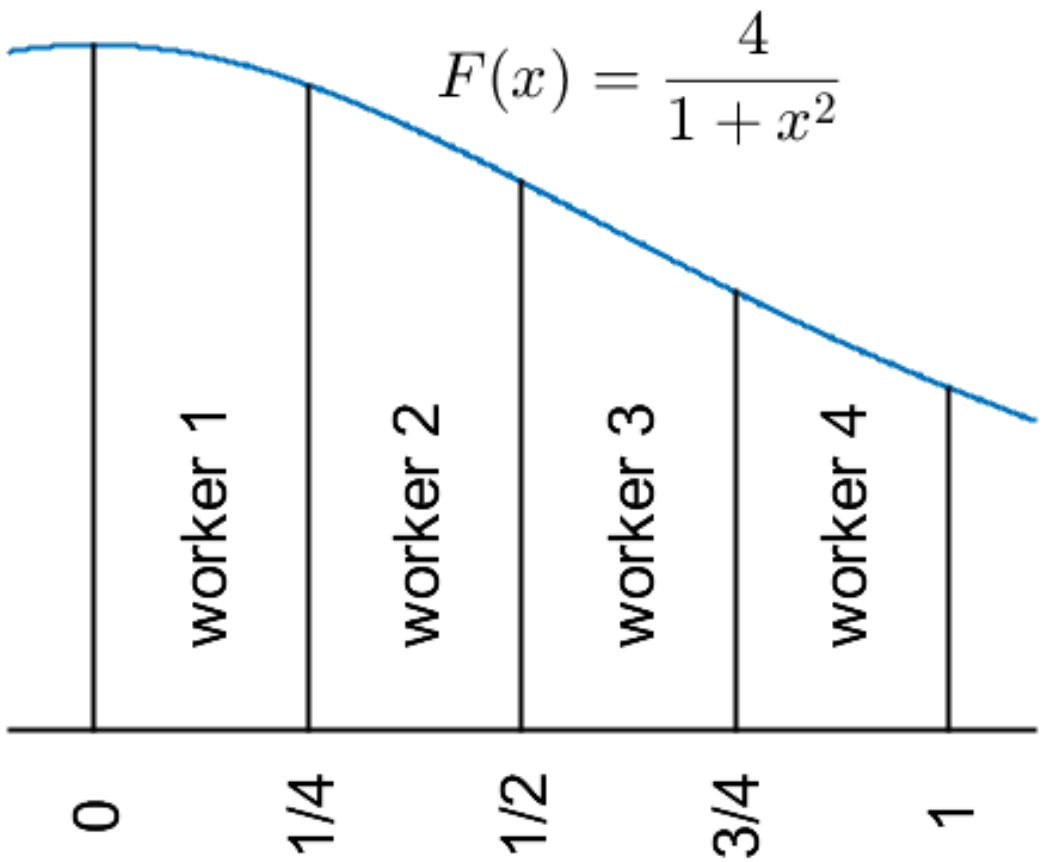
```
-bash4.4 mkdir -p ~/Documents/MATLAB
-bash4.4 tar -xf matlab-parallel-workshop-files-part-II.tgz -C ~/Documents/MATLAB
-bash4.4
-bash4.4 ls -1 ~/Documents/MATLAB/matlab-workshop/
calc_fft_cpu_gpu.m
calc_mandelbrot.m
calc_pi.m
mandelbrot_example.m
parallel_example.m
process_files_v1.m
process_files_v2.m
solve_sys_linear_eqns.m
test_fcn.m
-bash4.4
```

Change directories to workshop

```
>> cd(fullfile(userpath,'matlab-workshop'))
```

Exercise: Calculate π

$$\int_0^1 \frac{4}{1+x^2} dx = 4(atan(1) - atan(0)) = \pi$$



Calculate π

```
function calc_pi

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);
```

Submit calc_pi job

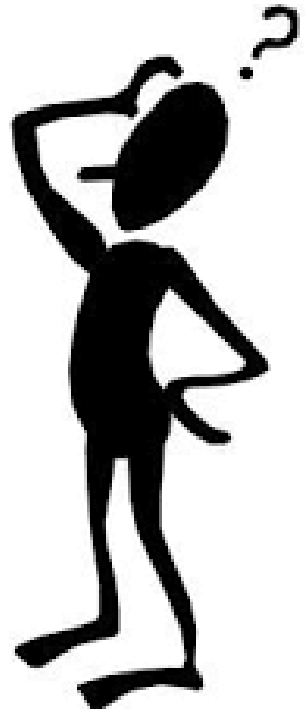
```
>> % Submit calc_pi job
>> c = parcluster;
>>
>> % Request 16 workers
>> j = c.batch(@calc_pi,0,{}, 'AutoAddClientPath',false, 'CurrentFolder','.', 'Pool',16);
additionalSubmitArgs =
    '--ntasks=17 --cpus-per-task=1 --ntasks-per-core=1'
>>
```

“If my Pool is size 16, why am I requesting 17 tasks?”

Fetch the results

```
>> % Submit calc_pi job
>> c = parcluster;
>>
>> % Request 16 workers
>> j = c.batch(@calc_pi,0,{}, 'AutoAddClientPath',false, 'CurrentFolder','.', 'Pool',16);
additionalSubmitArgs =
    '--ntasks=17 --cpus-per-task=1 --ntasks-per-core=1'
>>
>> % Check the state of the job
>> j.State
ans =
    'finished'
>>
>> % Fetch the results
>> j.fetchOutputs{:}
>>
```

"Where's the output?"



Fetch the diary

```
>> j.diary
--- Start Diary ---
Lab  2:
  Subinterval: [0.0625, 0.125]
Lab 11:
  Subinterval: [0.625, 0.6875]
Lab 12:
  Subinterval: [0.6875, 0.75]
Lab 16:
  Subinterval: [0.9375, 1    ]
Lab  1:
  Subinterval: [0    , 0.0625]
Lab  4:
  Subinterval: [0.1875, 0.25]
Lab  6:
  Subinterval: [0.3125, 0.375]
Lab  9:
  Subinterval: [0.5 , 0.5625]
Lab 10:
  Subinterval: [0.5625, 0.625]
Lab  6:
  Subinterval: [0.3125, 0.375]   Integral: 0.223543
Lab  8:
  Subinterval: [0.4375, 0.5 ]   Integral: 0.204949
Lab  9:
  Subinterval: [0.5 , 0.5625]   Integral: 0.194967
Lab 10:
  Subinterval: [0.5625, 0.625]   Integral: 0.184839
Lab 11:
  Subinterval: [0.625, 0.6875]   Integral: 0.174752
Lab 14:
  Subinterval: [0.8125, 0.875]   Integral: 0.146054
Lab 15:
  Subinterval: [0.875, 0.9375]   Integral: 0.137285
Lab 16:
  Subinterval: [0.9375, 1    ]   Integral: 0.128988
pi            : 3.141592653589793116
Approximation: 3.141592653589793116
Error         : 0

--- End Diary ---
>>
```

What gets “returned”?

- Function output
- Diary
- Saved files

Example

"What size Pool am I running?"

```
function [t, A] = test_fcn(sims)

disp('Start sim')

A = nan(sims,1);
t0 = tic;
parfor idx = 1:sims
    A(idx,1) = idx;
    pause(0.5)
    idx
end
t = toc(t0);

disp('Finished')

save RESULTS A
```

Job submission

```
>> j = c.batch(@test_fcn,1,{100}, 'AutoAddClientPath',false, 'CurrentFolder','.', 'Pool',10);
additionalSubmitArgs =
    '--ntasks=11 --cpus-per-task=1 --ntasks-per-core=1'
>>
```

Fetch output

```
function [t, A] = test_fcn(sims)
```

```
c.batch(@test_fcn,1,{100},
```

```
>> % Fetch the results
>> j.fetchOutputs{:}
ans =
    7.5236
```

"Where's *A*?"

Diary

```
% View the diary
j.diary
--- Start Diary ---
Start sim

ans =

     2


ans =

     4

  ...

ans =

   100


ans =

    98

Finished

--- End Diary ---
```

```
function [t, A] = test_fcn(sims)

disp('Start sim')

A = nan(sims,1);
t0 = tic;
parfor idx = 1:sims
    A(idx,1) = idx;
    pause(0.5)
    idx
end
t = toc(t0);

disp('Finished')

save RESULTS A
```

Save files

"Where does **RESULTS** get written to?"

```
function [t, A] = test_fcn(sims)

disp('Start sim')

A = nan(sims,1);
t0 = tic;
parfor idx = 1:sims
    A(idx,1) = idx;
    pause(0.5)
    idx
end
t = toc(t0);

disp('Finished')

save RESULTS A
```

Other settable job properties (1)

```
>> c.AdditionalProperties
ans =
  AdditionalProperties with properties:

                    AccountName: ''
          AdditionalSubmitArgs: ''
                    ClusterHost: 'login01.magnitude.uni-due.de'
                     Constraint: ''
                   EmailAddress: ''
                    EnableDebug: 0
                   IdentityFile: '\\fs-2.uni-due.de\adj403w\Dokumente\MATLAB\ude.ppk'
     IdentityFileHasPassphrase: 0
                       MemUsage: ''
                   ProcsPerNode: 0
                      QueueName: ''
      RemoteJobStorageLocation: '/homes/adj403w/.matlab/3p_cluster_jobs/magnitude/nb-zim-pp002/R2021a/nonshared'
                   Reservation: ''
                      Switches: ''
              UseIdentityFile: 1
                       UseSmpd: 0
                      Username: 'adj403w'
                      WallTime: ''
```

Other settable job properties (2)

- AccountName
- Constraint
- EmailAddress
- MemUsage
- ProcsPerNode
- QueueName
- Reservation
- Switches
- WallTime

Other settable job properties (3)

```
>> c = parcluster;
>>
>> % Memory per core
>> c.AdditionalProperties.MemUsage = '4gb';
>>
>> % Walltime (5 hours)
>> c.AdditionalProperties.WallTime = '05:00:00';
>>
>> % Clear memory requirement
>> c.AdditionalProperties.MemUsage = '';
>>
```

Submitting scripts, instead of functions

```
>> z = 10;
>>
>> % Submit a script (instead of a function)
>> j = c.batch('x = 3; y = 4, z', 'AutoAddClientPath',false, 'CurrentFolder','.');
additionalSubmitArgs =
    '--ntasks=1 --cpus-per-task=1 --ntasks-per-core=1'
>>
```

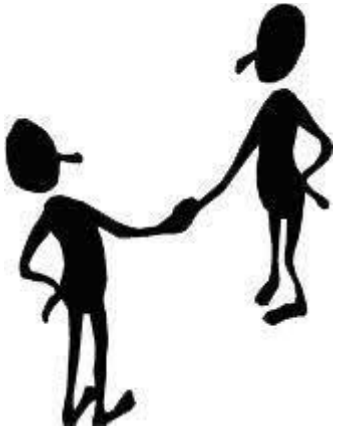
Loading variables to local workspace

```
>> z = 10;
>> % Submit a script (instead of a function)
>> j = c.batch('x = 3; y = 4, z', 'AutoAddClientPath',false, 'CurrentFolder','.');
additionalSubmitArgs =
    '--ntasks=1 --cpus-per-task=1 --ntasks-per-core=1'
>> clear z
>> who

Your variables are:

c  j

>> % Check the state of the job
>> j.State
ans =
    'finished'
>> % Load variables from job
>> j.load
>> who

Your variables are:

ans  c    j    x    y    z

>> j.diary
--- Start Diary ---

y =

     4


z =

    10


--- End Diary ---
```

"If we clear `z`, then why does `who` display it?

"I'll pass all of the variables in my local workspace to all of the workers. Then I'll receive everything the workers generate and load it back to my local workspace."

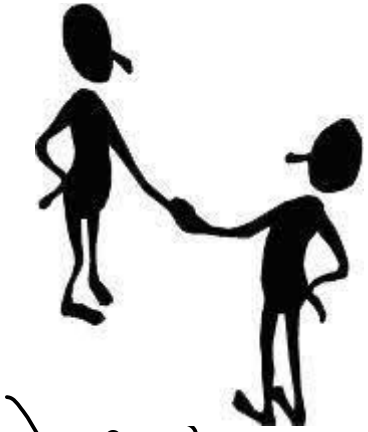
Adding files to the job



- AdditionalPaths
  - List absolute paths on the cluster (include subdirectories if needed)
- AttachedFiles
  - List files not automatically added to the job (e.g., binary files)
- AutoAddClientPath
  - Should be set to false
- AutoAttachFiles
  - Useful for small number, often changing files

Should I send files with the job?

- By default, each job will copy all required files
  - How many jobs are you going to submit?
  - How large, in totality, are your files?
  - Do you files change a lot?



5 - 10 files
50 - 100 KB

Add to job

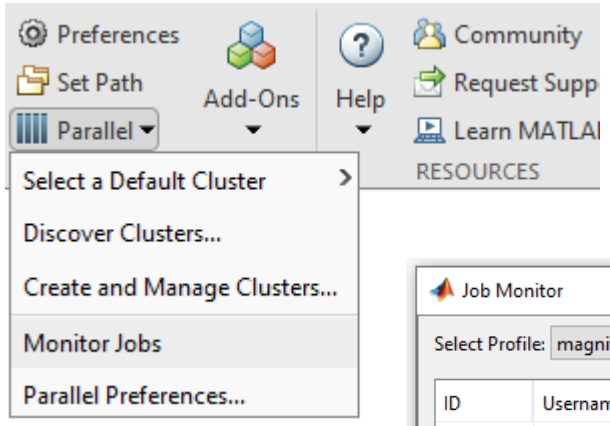
40 - 100 files
250 MB - 5GB

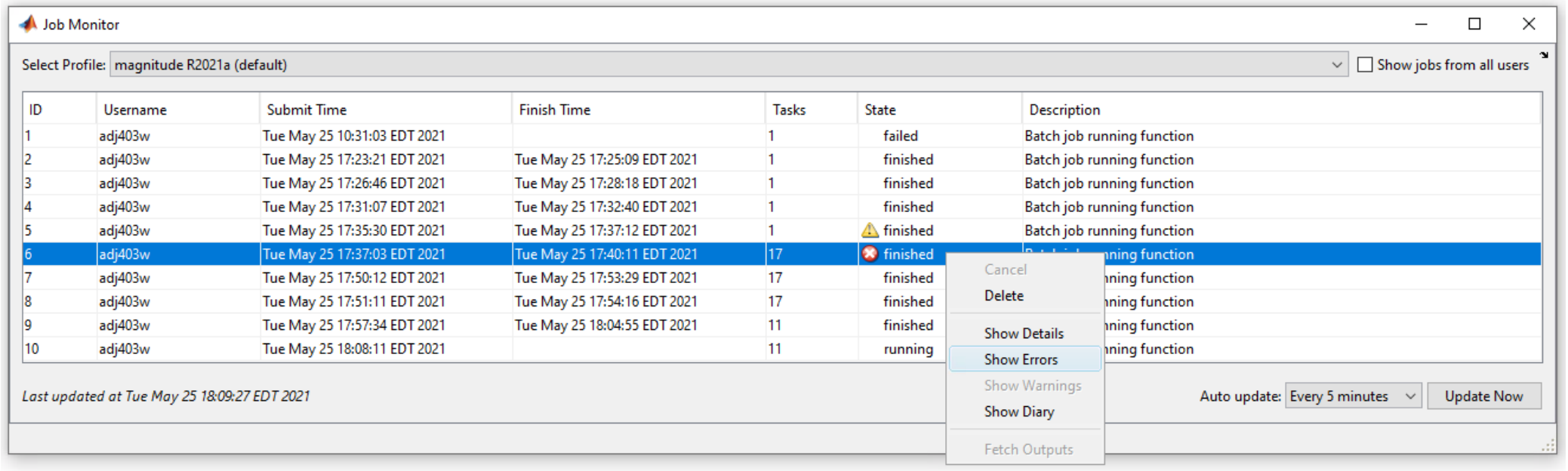
Attach on cluster

When has my job run and finished?

```
>> % Get email notification when my job has finished running
>> c.AdditionalProperties.EmailAddress = 'your-email-alias@uni-due.de';
>>
>> j = c.batch(@test_fcn,1,{100}, 'AutoAddClientPath',false, 'CurrentFolder','.', 'Pool',10);

additionalSubmitArgs =

    '--ntasks=11 --cpus-per-task=1 --ntasks-per-core=1 --mail-type=ALL --mail-user=your-email-alias@uni-due.de'

>>
```

Retrieving past jobs





Keep cluster files minimal: delete jobs

- As a good practice, delete jobs you no longer need

```
>> % Finished with the job, delete it to cleanup list of jobs
>> j.delete
>>
```

Debugging and Troubleshooting

Scheduler ID

```
>> j = c.batch(@pwd,1,{}, 'AutoAddClientPath',false, 'CurrentFolder','.');
additionalSubmitArgs =
    '--ntasks=1 --cpus-per-task=1 --ntasks-per-core=1'
>>
>> % Job ID vs Scheduler Id
>> j.ID
ans =
    41
>>
>> j.getTaskSchedulerIDs{1}
ans =
    '144468'
>>
```

Examples

```
>> % Undefined function
>> j = c.batch(@invalid_fcn,1,{}, 'AutoAddClientPath',false, 'CurrentFolder','.');
additionalSubmitArgs =
    '--ntasks=1 --cpus-per-task=1 --ntasks-per-core=1'
>>
>> % Reference to undefined variable or function
>> j2 = c.batch('y = x', 'AutoAddClientPath',false, 'CurrentFolder','.');
additionalSubmitArgs =
    '--ntasks=1 --cpus-per-task=1 --ntasks-per-core=1'
>>
>> % Incorrect argument list
>> j3 = c.batch(@pwd,1,{'home'}, 'AutoAddClientPath',false, 'CurrentFolder','.');
additionalSubmitArgs =
    '--ntasks=1 --cpus-per-task=1 --ntasks-per-core=1'
>>
```

Errored jobs (1)

```
>> % Undefined function
>> j.State
ans =
     'finished'
>>
>> j.fetchOutputs{:}
Error using parallel.Job/fetchOutputs (line 1286)
An error occurred during execution of Task with ID 1.
Caused by:
    Unrecognized function or variable 'invalid_fcn'.
>>
```

Errored jobs (2)

```
>> % Reference to undefined variable or function
>> j2.State
ans =
    'finished'
>>
>> j2.fetchOutputs{:}
Error using parallel.Job/fetchOutputs (line 1286)
An error occurred during execution of Task with ID 1.
Caused by:
    Unrecognized function or variable 'x'.
>>
```

Errored jobs (3)

```
>> % Incorrect argument list
>> j3.State
ans =
    'finished'
>>
>> j3.fetchOutputs{:}
Error using parallel.Job/fetchOutputs (line 1286)
An error occurred during execution of Task with ID 1.
Caused by:
    Too many input arguments.
>>
```

Logfile: Single core job

```
>> j = c.batch(@pwd,1,{}, 'AutoAddClientPath',false, 'CurrentFolder','.');
additionalSubmitArgs =
    '--ntasks=1 --cpus-per-task=1 --ntasks-per-core=1'
>>
>> % Retrieve log file for single core job
>> c.getDebugLog(j.Tasks(1))
LOG FILE OUTPUT:
================================================
SLURM_JOB_NAME = Job37.1
SLURM_JOB_USER = adj403w
SLURM_JOB_ID = 144464
SLURM_JOB_NUM_NODES = 1
SLURM_JOB_NODELIST = node288
================================================
Executing: /opt/Matlab/R2021a/bin/worker
Exiting with code: 0

>>
```

Logfile: Multi-core job

```
>> j = c.batch(@pwd,1,{}, 'AutoAddClientPath',false, 'CurrentFolder','.', 'Pool',2);
additionalSubmitArgs =
    '--ntasks=3 --cpus-per-task=1 --ntasks-per-core=1'
>>
>> % Retrieve log file for multi-core job
>> c.getDebugLog(j)
LOG FILE OUTPUT:
========================================
SLURM_JOB_NAME = Job38
SLURM_JOB_USER = adj403w
SLURM_JOB_ID = 144465
SLURM_JOB_NUM_NODES = 1
SLURM_JOB_NODELIST = node120
========================================
The scheduler has allocated the following nodes to this job:
node120
"/opt/Matlab/R2021a//bin/mw_mpiexec" -l -n 3 "/opt/Matlab/R2021a/bin/worker" -parallel
[0] Sending a stop signal to all the labs...
[0] Parallel pool is shutting down.[0]
Exiting with code: 0

>>
```

Designing Robust Code

From Coding to Cluster (1)

```
% Notes - From Coding to Cluster
% 1. Using a script, not a function
% 2. Paths are hardcoded
% 3. File separator is hard coded
% 4. Assumes TIF file exists
% 5. TIF files must be on the MATLAB path
% 6. Assumes output folder already exists where ever MATLAB is running
% 7. Results MAT-File will be overwritten next time it's run
% 8. Changes MATLAB working directory

filelist = dir('tif\*.tif');
fileNames = {filelist.name}';

segmentedCellSequence = batchProcessFiles(@detectCells,fileNames);
cd output
save SCS segmentedCellSequence
```

```
function [ofile, segmentedCellSequence] = process_files_v2(rootd,outd)
if nargin==0
    rootd = fullfile(pwd,'tif');
    outd = fullfile(pwd,'output');
end

filelist = dir(fullfile(rootd,'*.tif'));
if isempty(filelist)
    error('Failed to find image files: %s',rootd)
end
fileNames = {filelist.name}';

addpath(rootd)
segmentedCellSequence = batchProcessFiles(@detectCells,fileNames);

% Ensure output directory exists
if exist(outd,'dir')==false
    [FAILED,emsg,eid] = mkdir(outd);
    if FAILED==true
        error(eid,emsg)
    end
end

% Add timestamp for file uniqueness
ts = strrep(strrep(datestr(now),' ','_'),':','-');

% Save dir
old_dir = pwd;
c = onCleanup(@()cd(old_dir));
cd(outd)
ofile = ['SCS_' ts];
save(ofile,'segmentedCellSequence')
```

Run it locally

```
>> % Start local parallel pool
>> parpool(4);
Starting parallel pool (parpool) using the 'local' profile ...
Connected to the parallel pool (number of workers: 4).
>>
>> % Call the function locally
>> ofile = process_files_v2

ofile =

    'S:\sandbox\Workshops\Parallel-Computing-Workshop\matlab-workshop-files'

>>
```

Run it on the cluster

```
>> % Submit job to cluster
>> c = parcluster;
>> j = c.batch(@process_files_v2, 1, {'/work/raymond/proj-tiffs','/home/raymond/output-results'},'Pool',3);
>>
>> % Wait for job to finish
>> j.wait
>>
>> % Fetch the results
>> ofile = j.fetchOutputs{:}

ofile =

    '/home/raymond/output-results/SCS_27-Apr-2021_16-54-28'

>>
```

From Coding to Cluster (2)

```
% Notes - From Coding to Cluster
% 1. Using a script, not a function
%       return status or output directory
% 2. Paths are hardcoded
%       pass in root directory
% 3. File separator is hard coded
%       use fullfile
% 4. Assumes TIF file exists
%       check results when touching the file system
% 5. TIF files must be on the MATLAB path
%       add tif folder to the MATLAB path
% 6. Assumes output folder already exists where ever MATLAB is running
%       supply output directory to write to.  check if folder exists; if
%       not, create it
% 7. Results MAT-File will be overwritten next time it's run
%       add timestamp to filename
% 8. Changes MATLAB working directory
%       Track old directory, change back before leaving
```