Parallel Computing – Matlab – Part 1

Shelley Knuth
shelley.knuth@colorado.edu

www.rc.colorado.edu

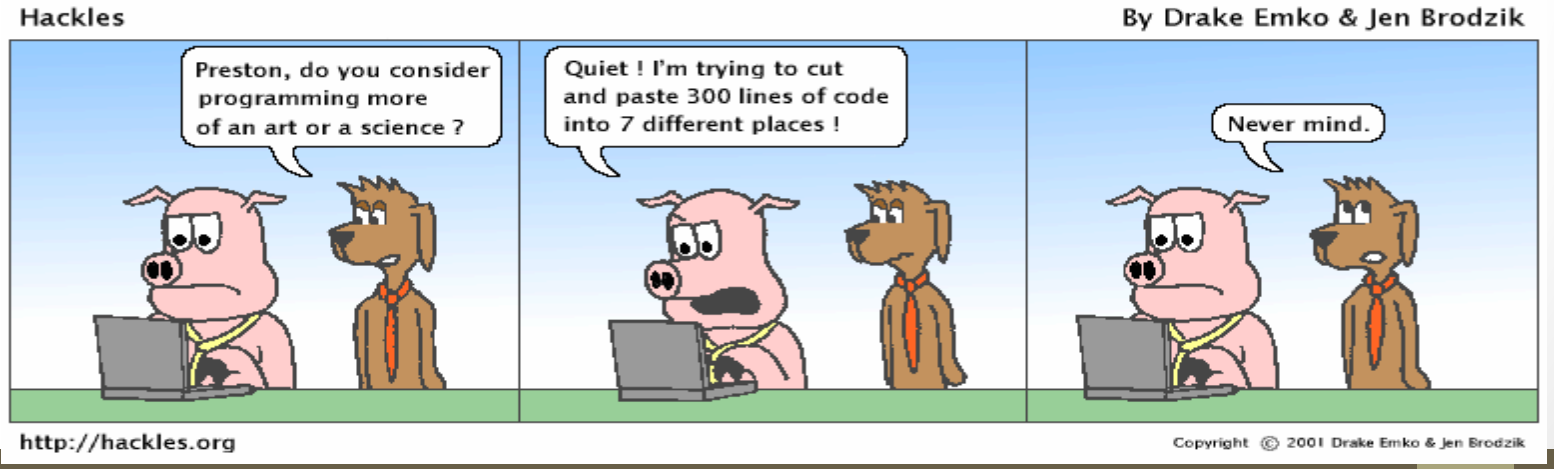
Slides: https://github.com/ResearchComputing/Parallelization_Workshop

Outline

- Profiling
- Vectorization
- Multi-threading
- Parallel Matlab
- Parallel Computing Toolbox
  - Prepping your GUI
  - Submitting jobs
  - Parfor

My code is slooooowwww

- What is the problem I have with my serial code?
  - Is it too slow? Where is it too slow?
- Can I vectorize my code to improve speedup?
- Can I use a function that implicitly uses parallelism?
- Can I use multi-threading?
- Do I have a for loop I am trying to speed up?



Step One…

- Where is my code slowing down?
  - `Tic` and `Toc`
  - Can't tell you exactly what within that code is slowest

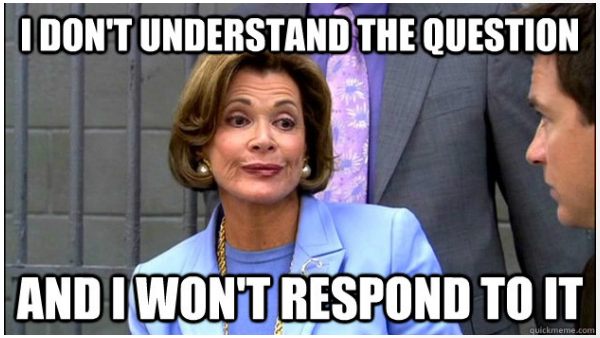
- Let's use the profiler

```
ml slurm/summit
sinteractive –-reservation=parallelD2
–-ntasks=4 –-nodes=1
ml matlab
matlab
Simple_loop.m
```

- Code analyzer is another good tool to use
- Take five minutes and use the profiler on some code you have
- Does it make sense to reduce any loop iterations?

Step Two…

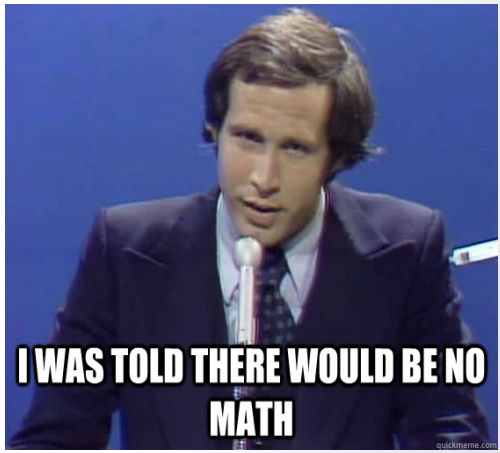
- Are there any places where my code could benefit from vectorization? Pre-allocation?
  - Pre-allocation: initialize arrays ahead of time to avoid dynamic resizing
- And if not, can I implement some type of parallelization?
  - Making use of existing functions
  - Multi-threading
  - The Parallel Computing Toolbox



https://www.mathworks.com/help/matlab/matlab_prog/vectorization.html

Vectorization

- Process of revising loop-based, scalar-oriented code to use MATLAB matrix and vector operations
- Do this because:
  - Appearance: looks like textbook math, so easier to understand
  - Less opportunity for error
  - Usually runs much faster than code that contains loops
- Great option in Matlab because its optimized for operations involving matrices and vectors



Example of Vectorization

- The code below demonstrates how a loop could be vectorized in Matlab

```
% Non-vectorized
a= rand(1,4);
b= rand(1,4);

for k= 1:length(a)
   c(k)= a(k) + b(k);
end
```

```
% Vectorized
a= rand(1,4);
b= rand(1,4);

c= a + b;
```

Existing Functions

- Certain functions in Matlab utilize the programming constructs of parallel computing implicitly to run the functions in parallel on a multi-core system
- These functions are automatically multi-threaded
- Little work, but potentially lots of gain
- No control over the processing
- Examples:
  - Fft, fmincon, ode, and many others

Threads

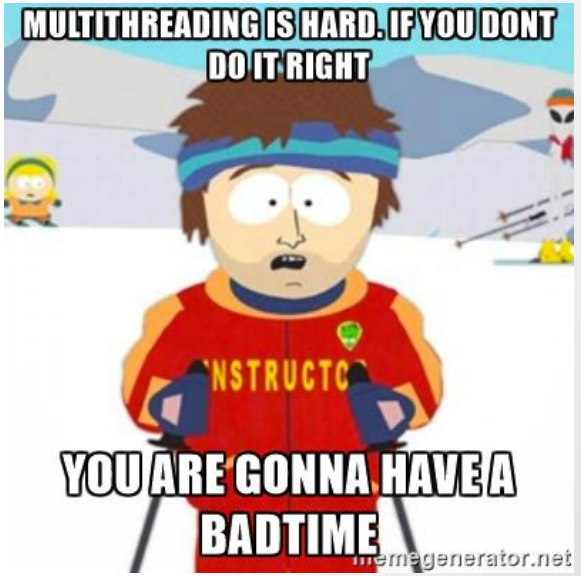
- A thread is a sequence of instructions within a program that can be executed independently of other code
  - It is a component of a process
- Every line in a program is a thread
- This is called your main thread, or your serial thread

Multi-threading

- Multi-threading occurs when you reach a point in your program where these instructions can be executed not only independently but also simultaneously
  - On multiple cores
- Multi-threading is a form of parallelism, but lighter than distributed computing
- Programming constructs such as OpenMP use multi-threading as their basis
- One of the biggest differences between multi-threading and distributed computing or larger-scale parallelism is that multi-threading uses shared, rather than distributed, memory

Multi-threading Real World Example

- Say you have a project: building a subdivision
- There are ten houses that will be built in this subdivision
- There are several components of the project that need to be completed on each of the houses
  - Pouring the foundation
  - Framing the house
  - Putting in the kitchen
  - Laying the carpeting
  - Landscaping
  - Etc, etc



Multi-threading Real World Example – Cont.

- Each component requires individual tasks to complete it
- For example, when landscaping one must pour the driveway, lay the grass, select shrubs, etc to plant, put in lighting, etc.

- There is a general contractor in charge of the entire project
- The general contractor has ten workers

Multi-threading Real World Example - Continued

- The general contractor divides up the tasks in the subdivision based on skill set or time
- The five workers work on each of the tasks

- The general contractor is the client
- The workers are the cores
- The tasks for each individual project are the threads
- When the tasks are worked on at the same time by the workers it is multi-threading

Multi-threading Programming Example

- Say you have a program that does a matrix multiplication of two 2x2 square matrices
- Each part of the calculation, where a row element is multiplied by a column element, can be thought of as a thread

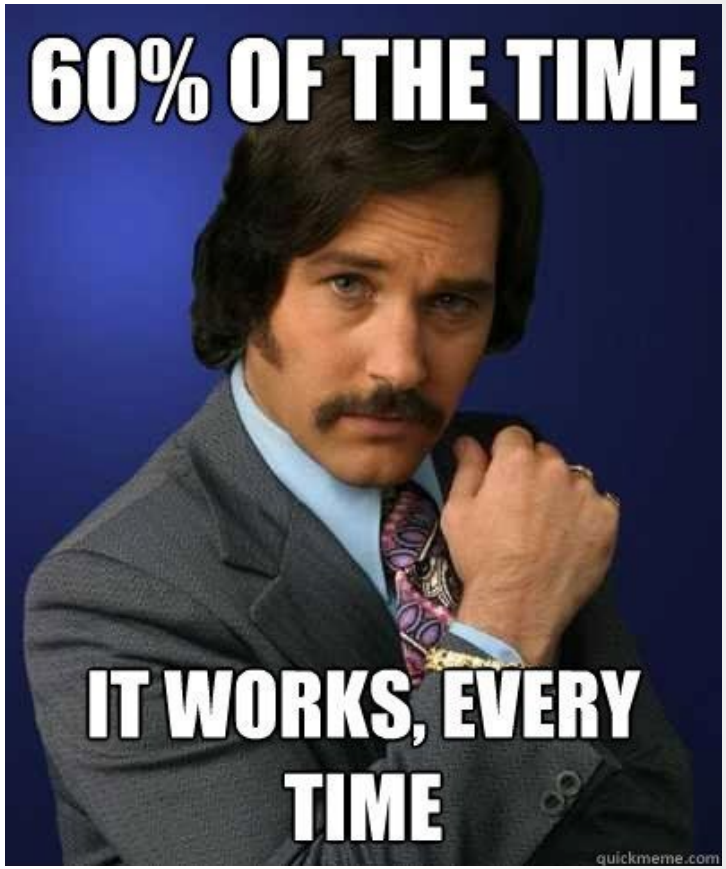
- You can implement multi-threading implicitly or explicitly
  - In Matlab, implicit occurs by using certain functions
  - Explicitly might mean implementing multi-threaded flags in slurm or using the PCT

Multi-threading example

- multi_threading.sh
- multi_threading.m

- Show speed up on 1 vs. 24 cores

- But first….



Before we go further…

- Let's look at running Matlab on the supercomputer
- The best way to do this is to call your Matlab program from a bash script
- Within the bash script will be several flags that help specify your job requirements

Sample (Partial) Bash Script

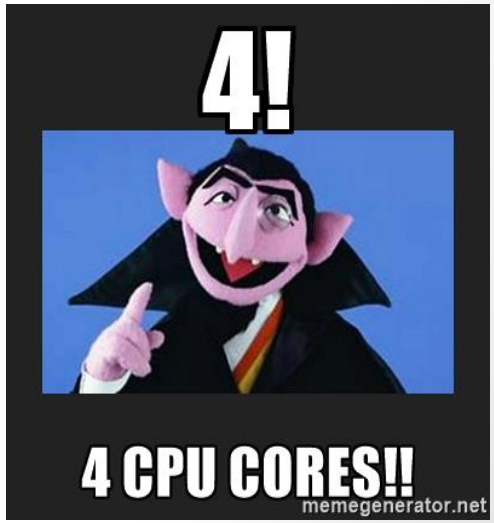
```
#!/bin/bash
#SBATCH --nodes=1
#SBATCH --ntasks=24

matlab -nosplash -nodesktop -r "clear;
num_workers=$SLURM_NTASKS; parallel_std;"
```

- RC's Matlab license does not allow for parallel computation across nodes
- Module load slurm/summit

Notes on Running Matlab on a Cluster

- Request the number of cores on the cluster and also the number of workers that will perform operations on those cores
- In older versions of Matlab you had to manually open a pool of workers
- Now it does in when run PCT command



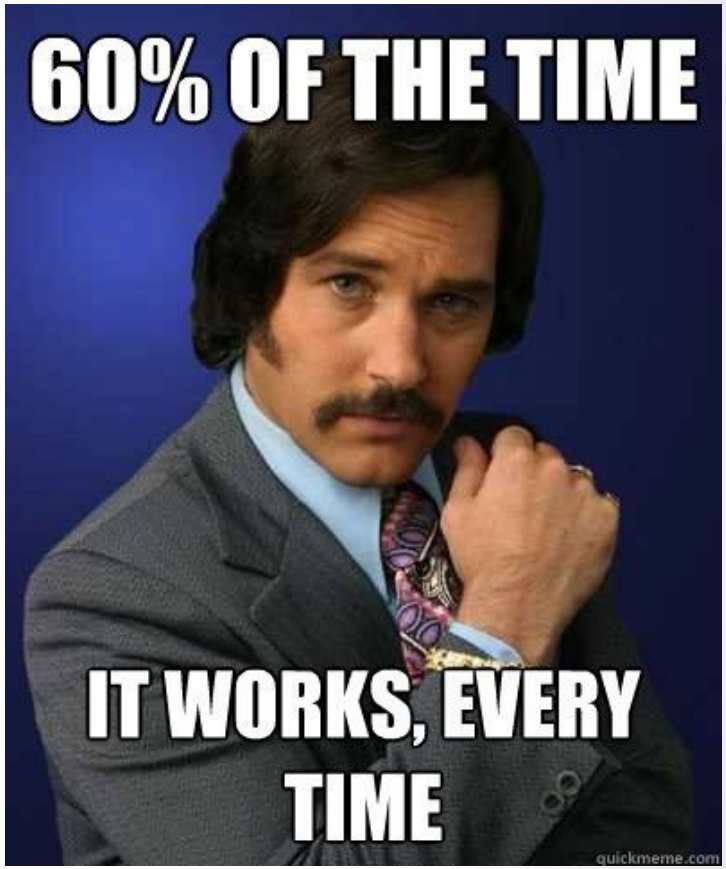
Matlab vs Slurm – who wins?

- You can open a parallel pool of workers in Matlab
- You also request a specific number of cores
- If the two are the same then no issue
- What if you ask for more cores than workers?
  - Each worker will operate on a core and the other cores will be idle
- What if you ask for less cores than workers?
  - The code will fail
  - Will say that the default value for a local cluster is the number of cores available

Multi-threading example

- multi_threading.sh
- multi_threading.m

- Show speed up on 1 vs. 24 cores



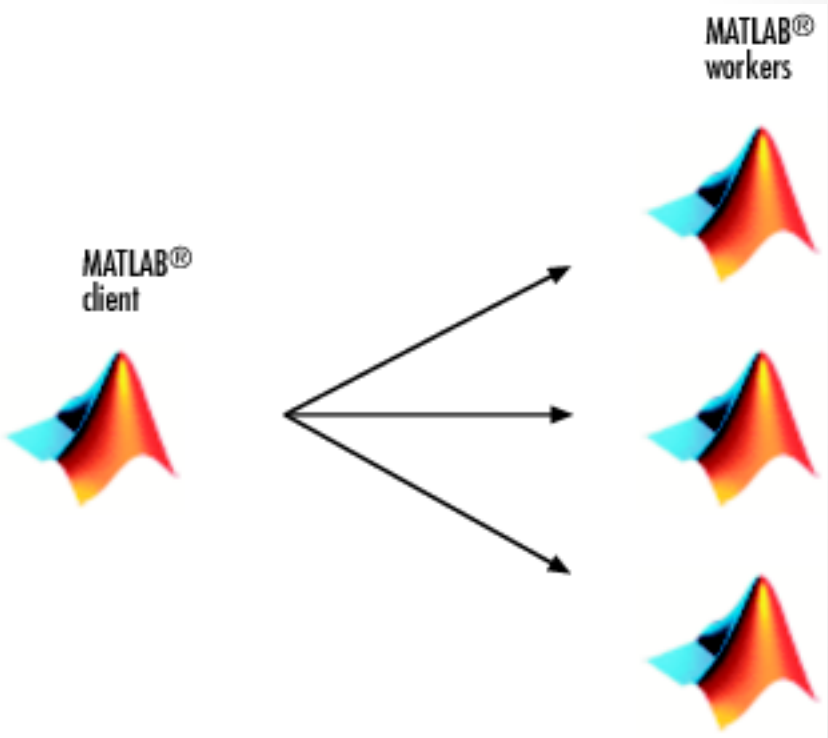
Explicit parallelism

- Several instances of Matlab run on several cores or computers
  - May have shared or distributed memory
- Simultaneously execute a single Matlab command or function
- In explicit parallelism, programmers can create and manage their parallelism to suit their needs
  - Can choose the processors to run on

Apple cores. One type of core.

Parallel Matlab

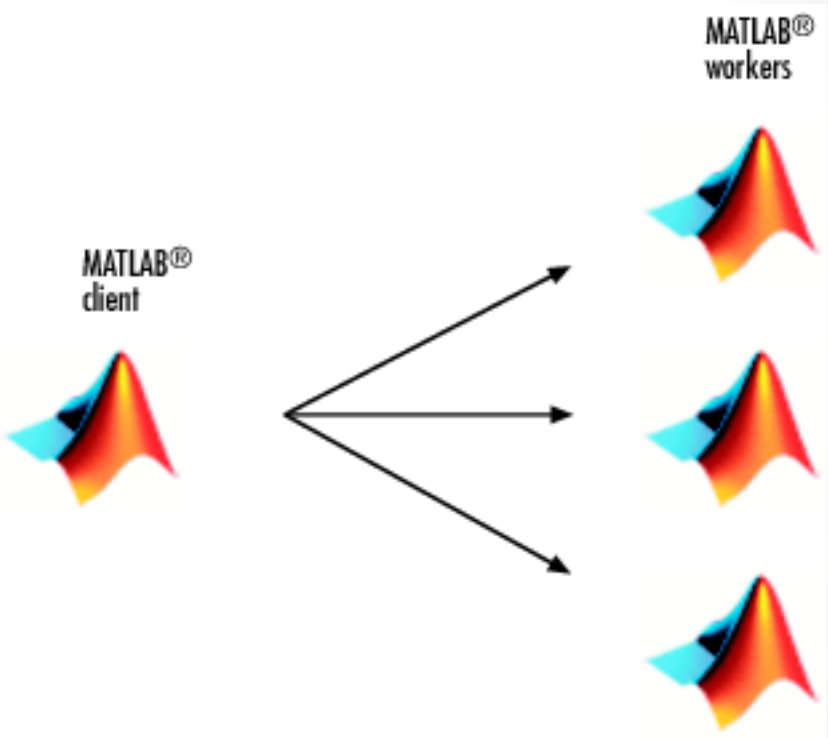
- When you start up a Matlab session by opening Matlab, you are starting a client session
  - Creates workers, receives results, distributes work
- Move to a parallel session requires the client to pass some of its computational work along to Matlab workers



https://www.mathworks.com/help/distcomp/how-parallel-computing-products-run-a-job.html

Parallel Matlab

- A Matlab worker is an individual Matlab session
  - Usually have one worker per core on your system
  - Same as client without front end – just computation
  - Workers can communicate to the client
  - Workers complete tasks to help speed up job completion
  - Must run independently



https://www.mathworks.com/help/distcomp/how-parallel-computing-products-run-a-job.html

Parallel Computing Toolbox (PCT)

- Matlab offers explicit parallelism within the Parallel Computing Toolbox
- Perform parallel computations on multicore computers, GPUs, and computer clusters
- Mimics OpenMP in many ways
- Many Matlab functions work in concert with the PCT
- Simple to utilize with just the use of certain commands

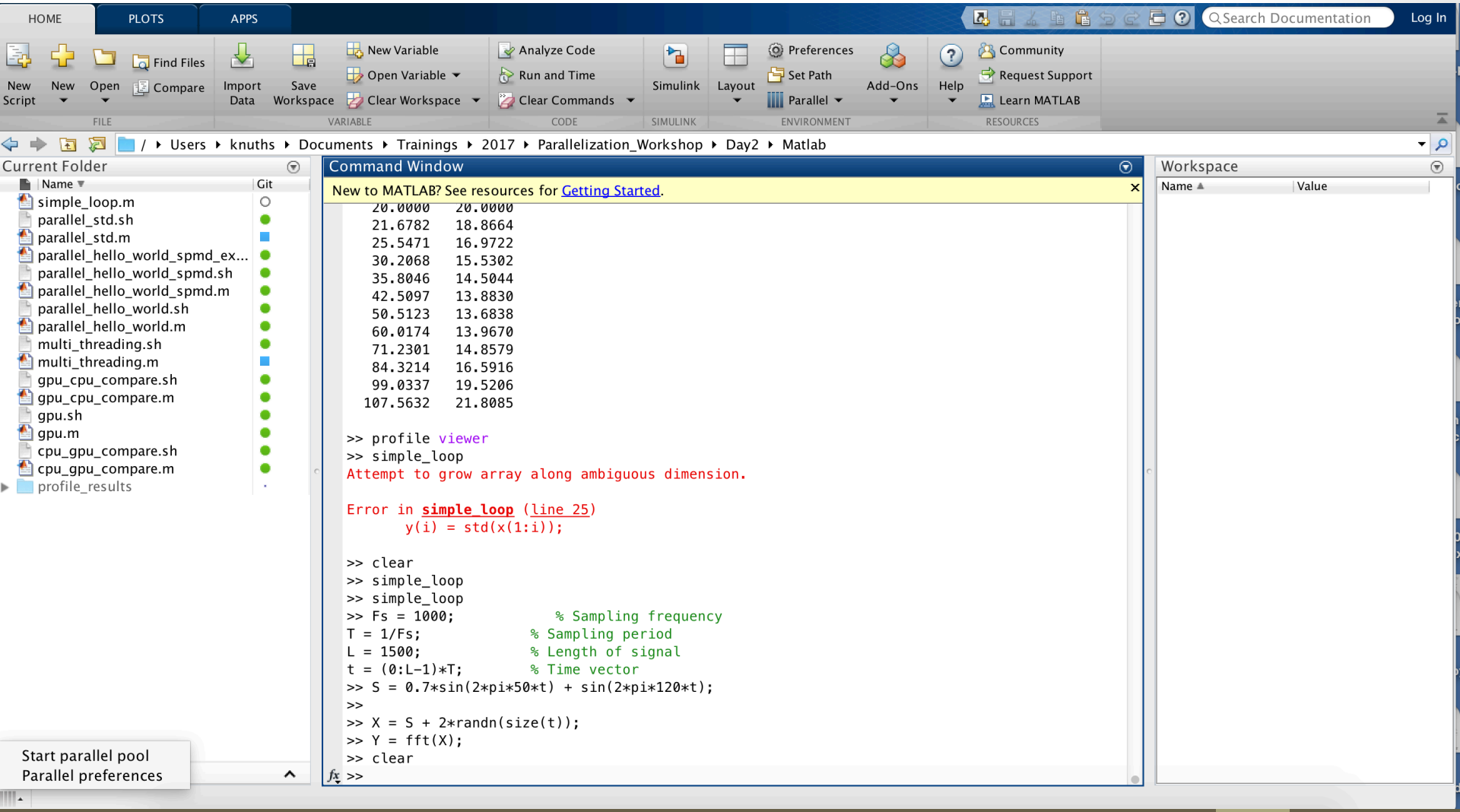
Requirements to Run PCT

- Access to Matlab
- The PCT add-on
  - Prohibitively expensive for individuals
  - Likely accessing through institution
- At CU you have access to a Matlab site license
- Where can I run PCT?
  - On cluster (Summit)
  - On your laptop
  - On your desktop/workstation

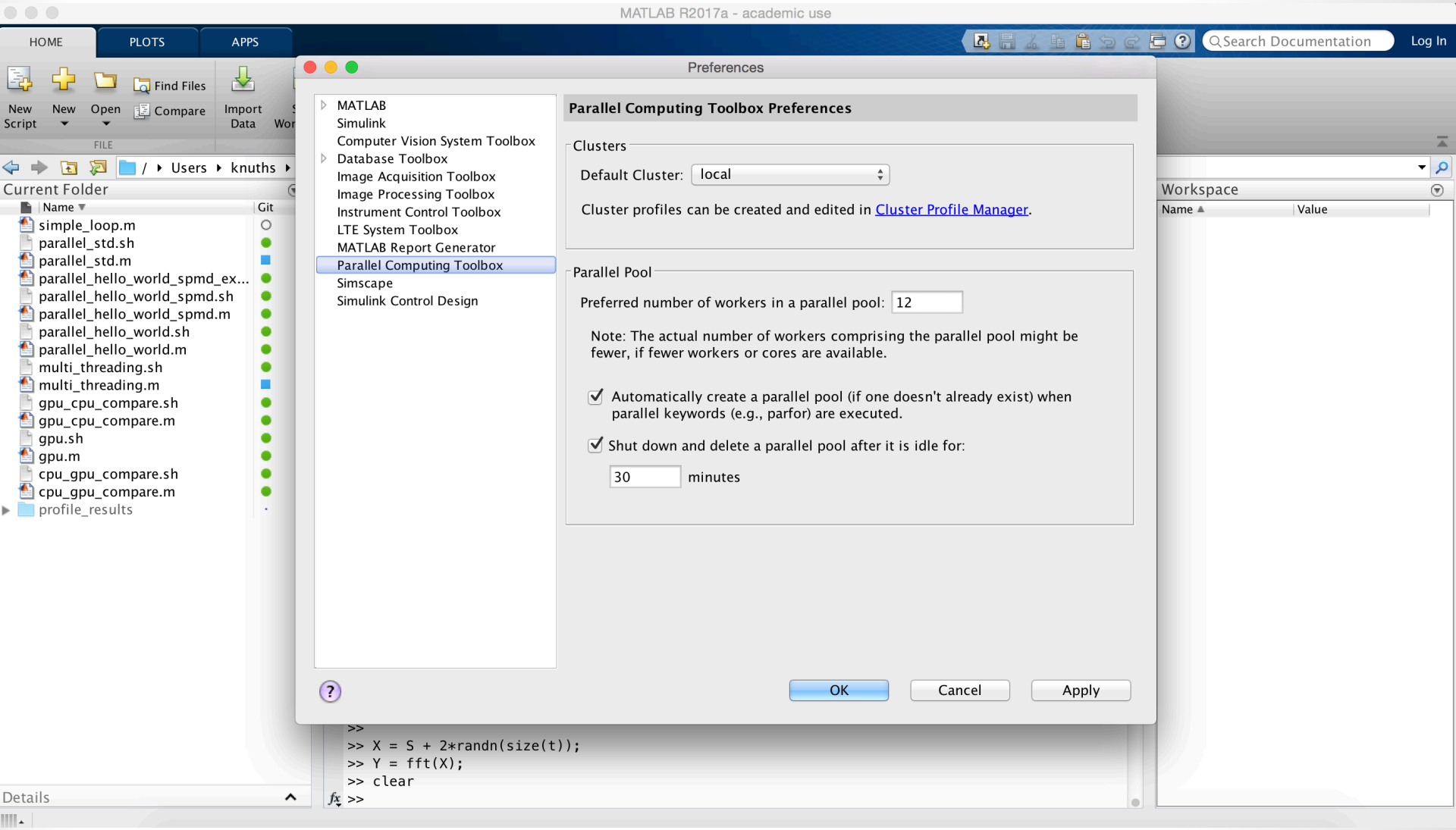
Parallel and the GUI

- You can set parallel constructs either in your script or in the Matlab Desktop
- In order to utilize the PCT, you must open up a pool of parallel workers
- You can do this explicitly by running the command `parpool`
- You can also set this explicitly in the GUI to do automatically

Starting a parallel pool from GUI

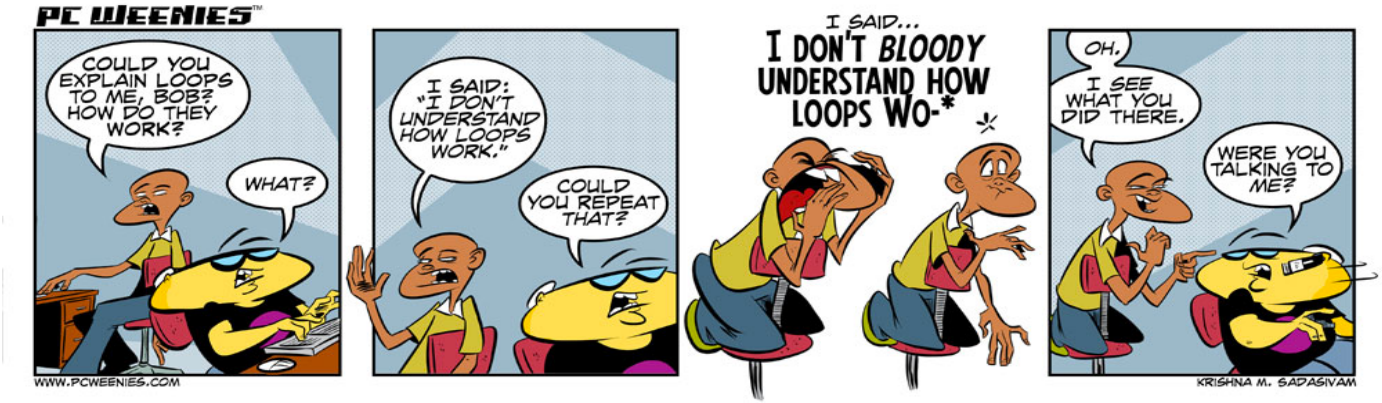


Parallel Preferences



parfor

- Loops are slow – this is a good place to start
- Matlab workers execute loop iterations in parallel on workers in parallel pool simultaneously
- Matlab client issues the parfor command and coordinates with workers
- Must have a parallel pool to use
- Cannot nest parfor loops



Parallel and Not Parallel

Not Parallel:

```
for i=1:10
    x=x(i)+1;
end
```

Parallel:

```
parfor i=1:10
    x=x(i)+1;
end
```

When to Use Parfor

- If you have determined your for loop is causing a bottleneck
- If your loop iterations are completely independent
  - Many iterations of the same calculation
- If you are not making any global variable declarations
- Your loop does not contain any break or return statements
- If you have the PCT

Running Matlab in Parallel

- Let's take a serial for loop and convert it to run in parallel

  parallel_std.sh

  parallel_std.m

Your Turn

- Write up Matlab code that instructs each of the workers to print ”Hello World” each time the parfor loop runs
- Use 24 workers
- Set up your slurm configurations properly

Result

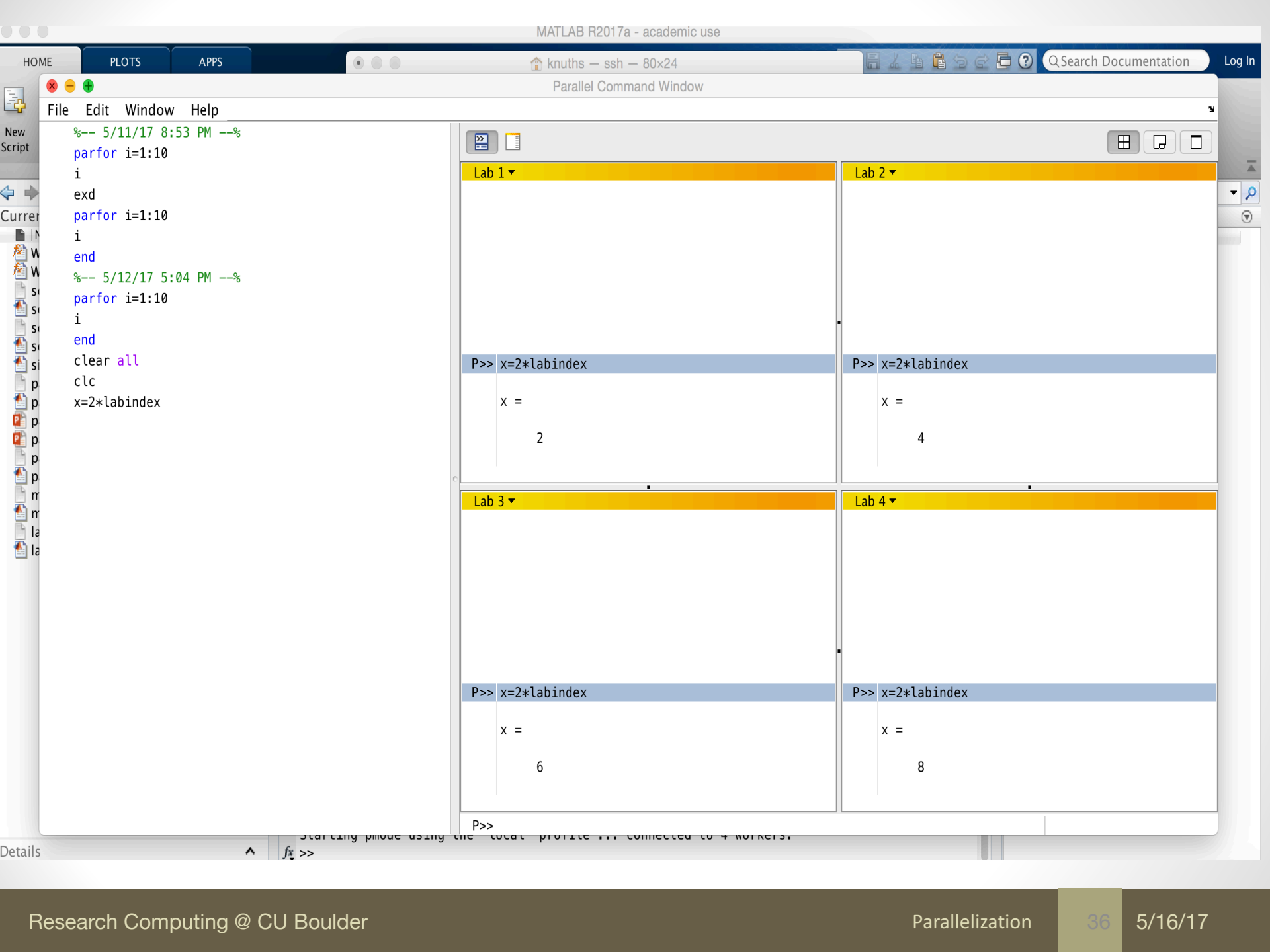
- Parallel_hello_world.sh
- Parallel_hello_world.m

Pmode

- One last thing…
- There's a parallel profiler GUI
- Let's start it up!

```
ml slurm/summit
sinteractive –-reservation=parallelD2
–-ntasks=4 –-nodes=1
ml matlab
matlab
```

- Once Matlab starts, type `pmode start x=2*labindex`
- This will only work if you have X-windows installed on your local system


MATLAB R2017a - academic use
HOME
PLOTS
APPS
knuths — ssh — 80×24
Search Documentation
Log In
Parallel Command Window
File Edit Window Help
%-- 5/11/17 8:53 PM --%
parfor i=1:10
i
exd
parfor i=1:10
i
end
%-- 5/12/17 5:04 PM --%
parfor i=1:10
i
end
clear all
clc
x=2*labindex
Lab 1
P>> x=2*labindex
x =
2
Lab 2
P>> x=2*labindex
x =
4
Lab 3
P>> x=2*labindex
x =
6
Lab 4
P>> x=2*labindex
x =
8
P>>
New Script
Details
>>

Questions?

- Email rc-help@colorado.edu
- Twitter: CUBoulderRC

- Link to survey on this topic: http://tinyurl.com/curc-survey16

- Slides: https://github.com/ResearchComputing/Parallelization_Workshop