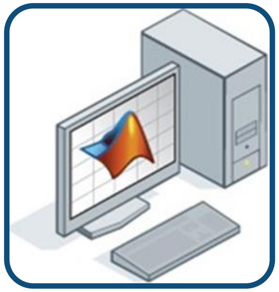
Accelerating MATLAB algorithms

By Dr Jasmina Lazić
Application Engineer
MathWorks

Agenda



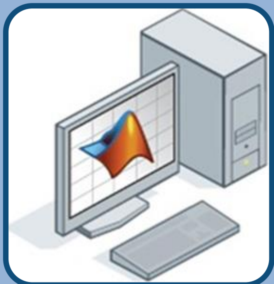
Accelerating MATLAB code with parallel computing

Scaling up: accessing more/better hardware

Big data with MATLAB:

an overview

Agenda



Accelerating MATLAB code with parallel computing

Scaling up: accessing more/better hardware

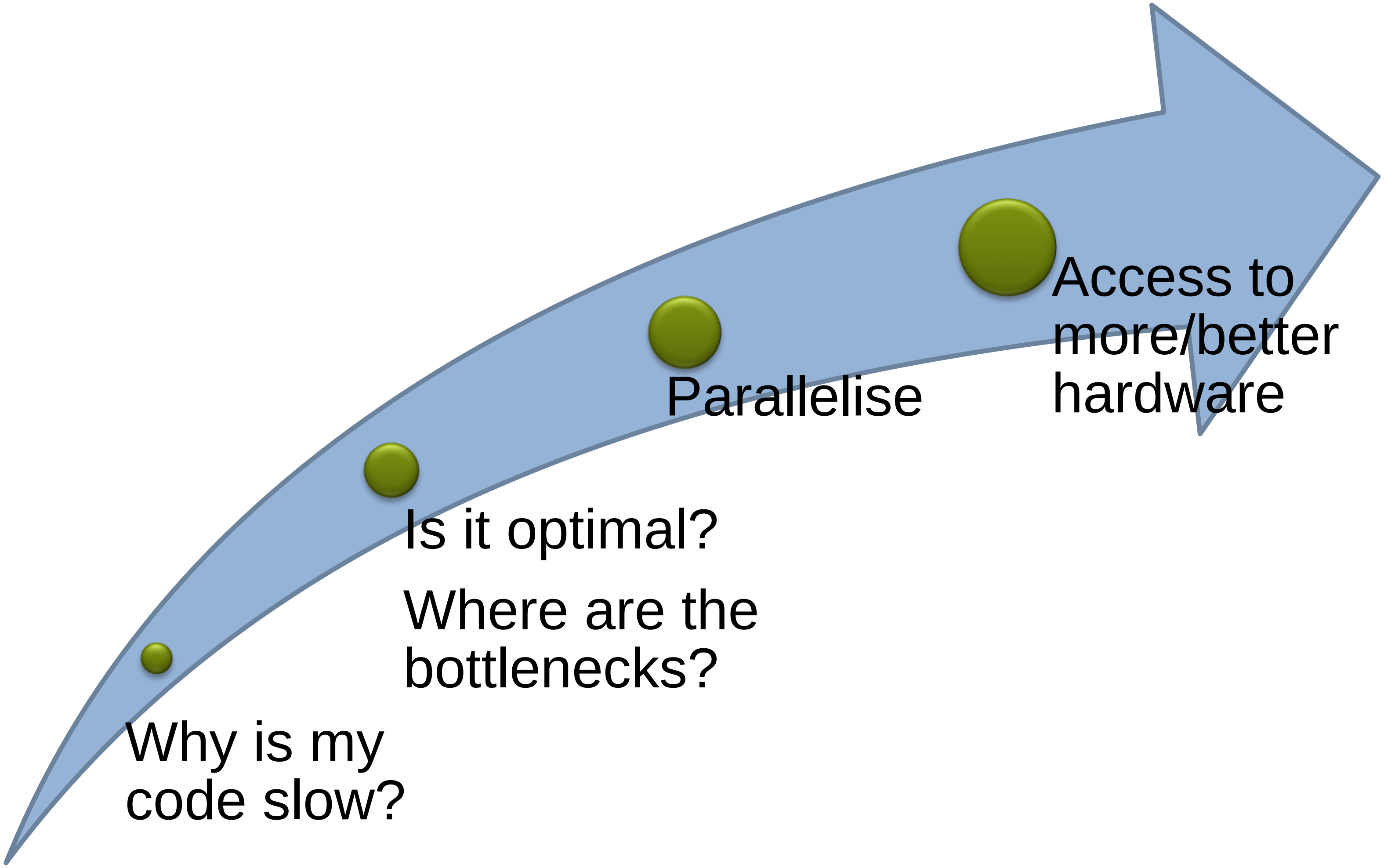
Big data with MATLAB:

an overview

Accelerating MATLAB code

- Profile and optimise your code
- Introduction to parallel computing tools
- Solving big technical problems in MATLAB
  - Programming task parallel applications
  - Programming data parallel applications

Who wants faster code?



Using More Hardware

Built-in multithreading

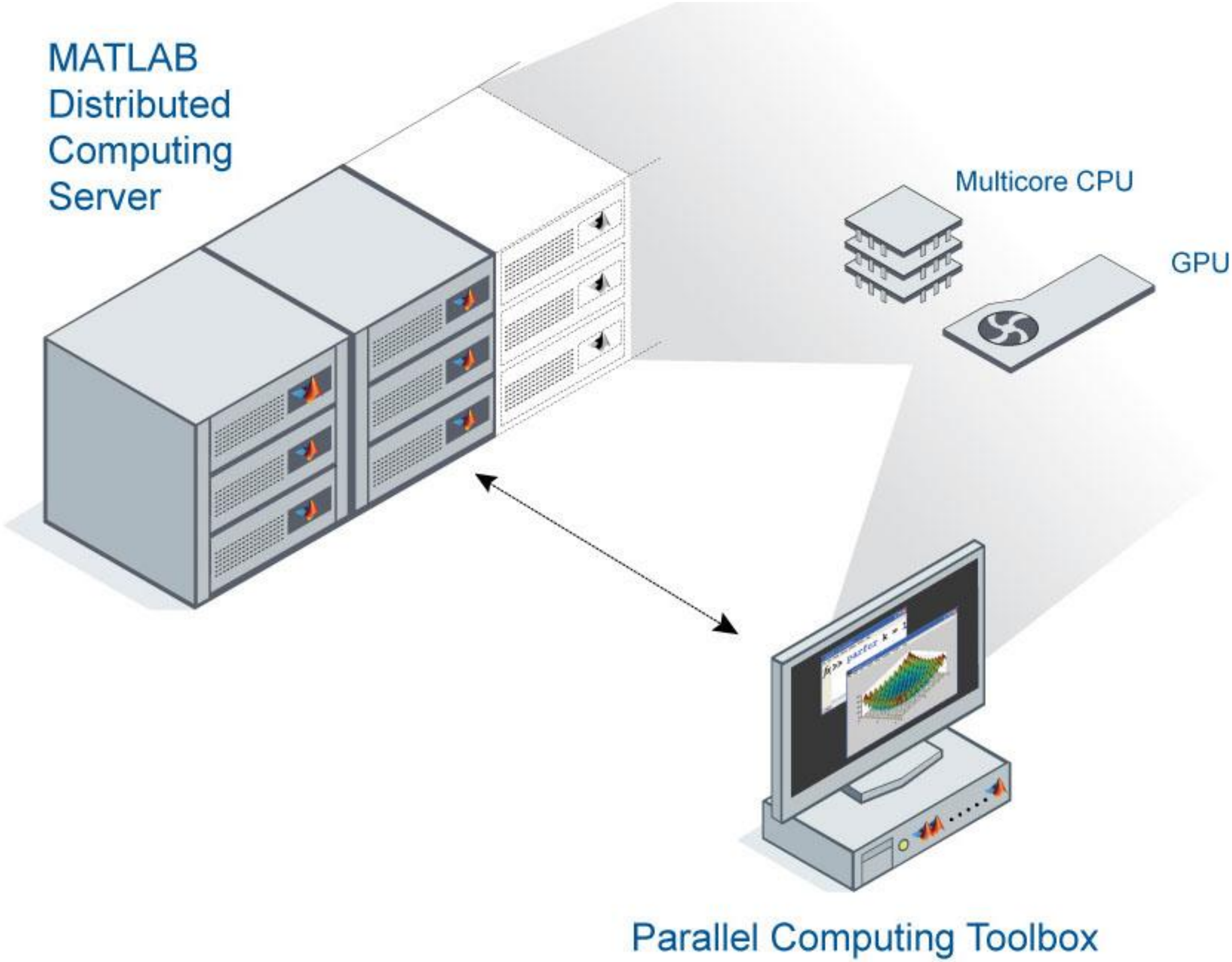
- Automatically enabled in MATLAB since R2008a
- Multiple threads in a single MATLAB computation engine

  www.mathworks.com/discovery/multicore-matlab.html

Parallel computing using explicit techniques

- Multiple computation engines controlled by a single session
- High-level constructs to let you parallelise MATLAB applications
- Perform MATLAB computations on GPUs

Parallel Computing Products



Solving Big Technical Problems

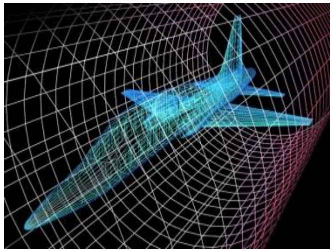
Challenges	You could…		Solutions
Long running / Computationally intensive	Wait	⇨	Larger Compute Pool (e.g. More Processors)
Large data set	Reduce size of problem	⇨	Larger Memory Pool (e.g. More Machines)

[Optimizing JIT Steel Manufacturing Schedule]

Cut simulation time from 1 hour to 5 minutes

[Heart Transplant Studies]

3-4 weeks reduced to 5 days

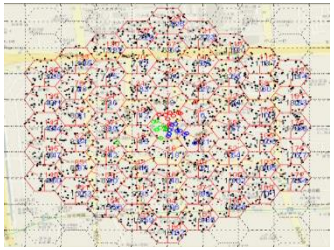


[Flight Test Data Analysis]

16x Faster

[Mobile Communications Technology]

Simulation time reduced from weeks to hours, 5x more scenarios



[Hedge Fund Portfolio Management]

Simulation time reduced from 6 hours to 1.2 hours

Programming Parallel Applications (CPU)

Ease of Use

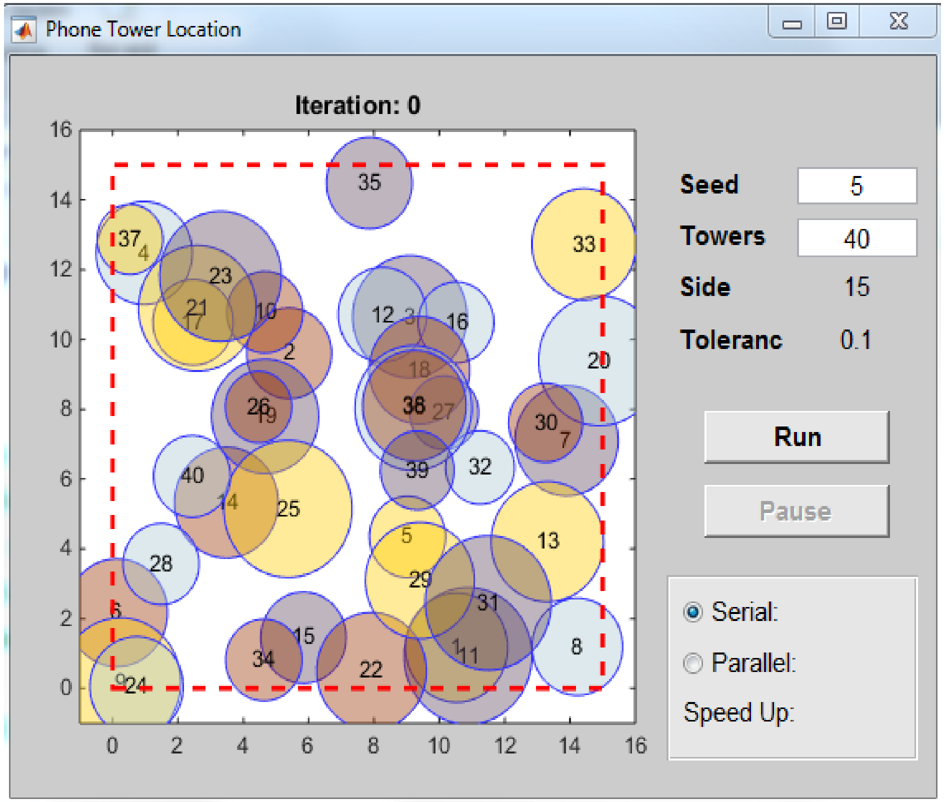
- Parallel enabled functionality in toolboxes

Greater Control

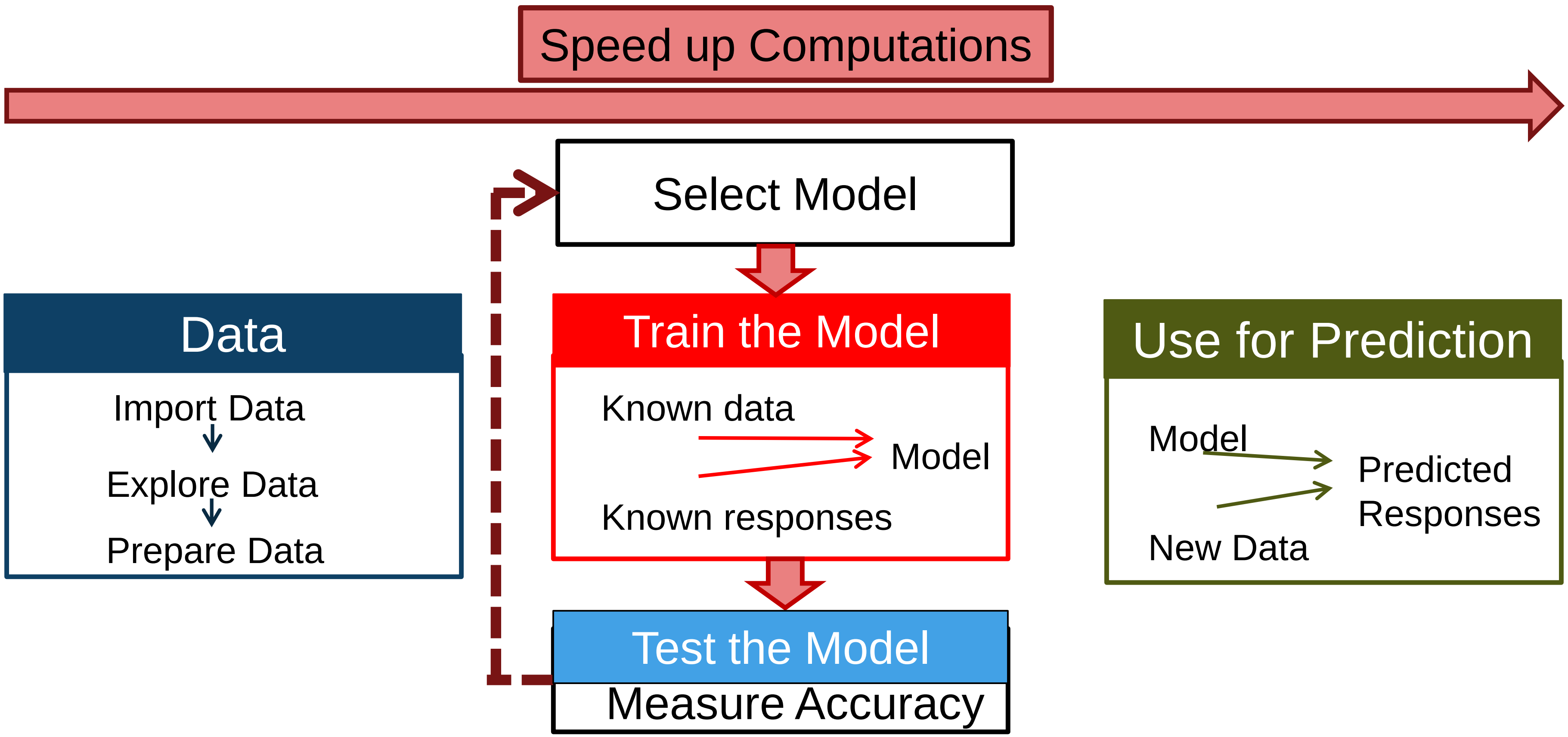
Example: Optimizing Cell Tower Position

Parallel enabled functionality

- With Parallel Computing Toolbox use parallel enabled algorithms in Optimization Toolbox
- Run optimization in parallel
- Use pool of MATLAB workers

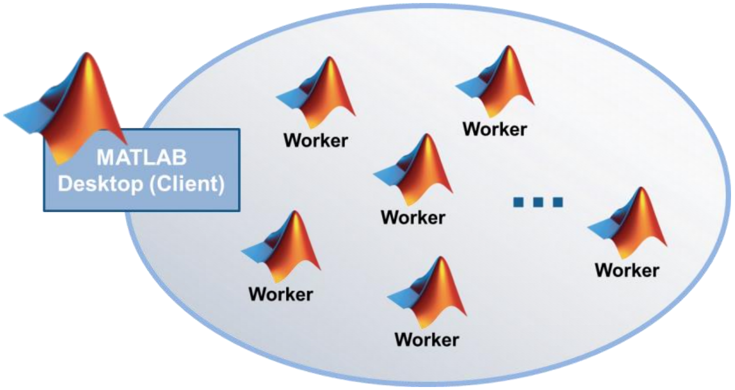


Supervised Learning - Workflow



Tools Providing Parallel Computing Support

- Optimization Toolbox
- Global Optimization Toolbox
- Statistics Toolbox and MachineLearning Toolbox
- Signal Processing Toolbox
- Neural Network Toolbox
- Image Processing Toolbox



'UseParallel', true

options = optimset('UseParallel', true);

Directly leverage functions in Parallel Computing Toolbox

www.mathworks.com/builtin-parallel-support

Programming Parallel Applications (CPU)

Ease of Use

- Parallel enabled functionality in toolboxes

- Simple programming constructs:

  `parfor, batch, distributed`

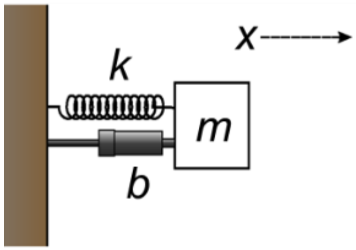
Greater Control

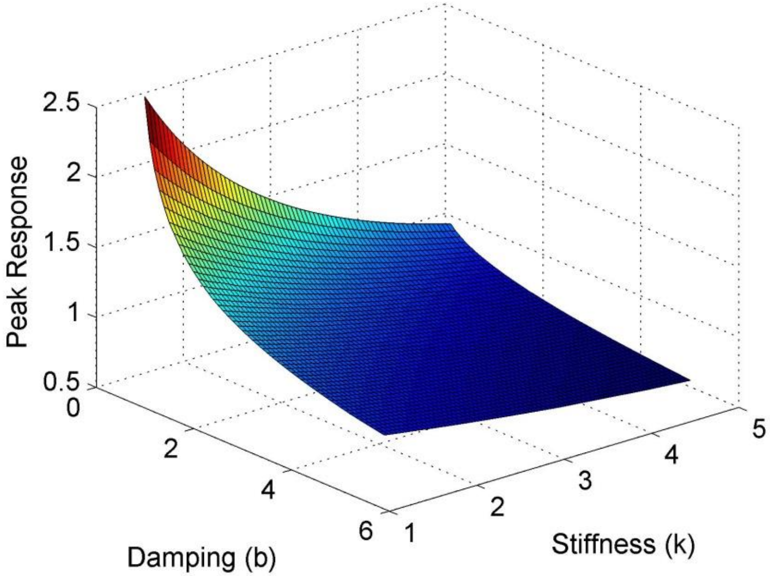
Example: Parameter Sweep of ODEs

Parallel for-loops

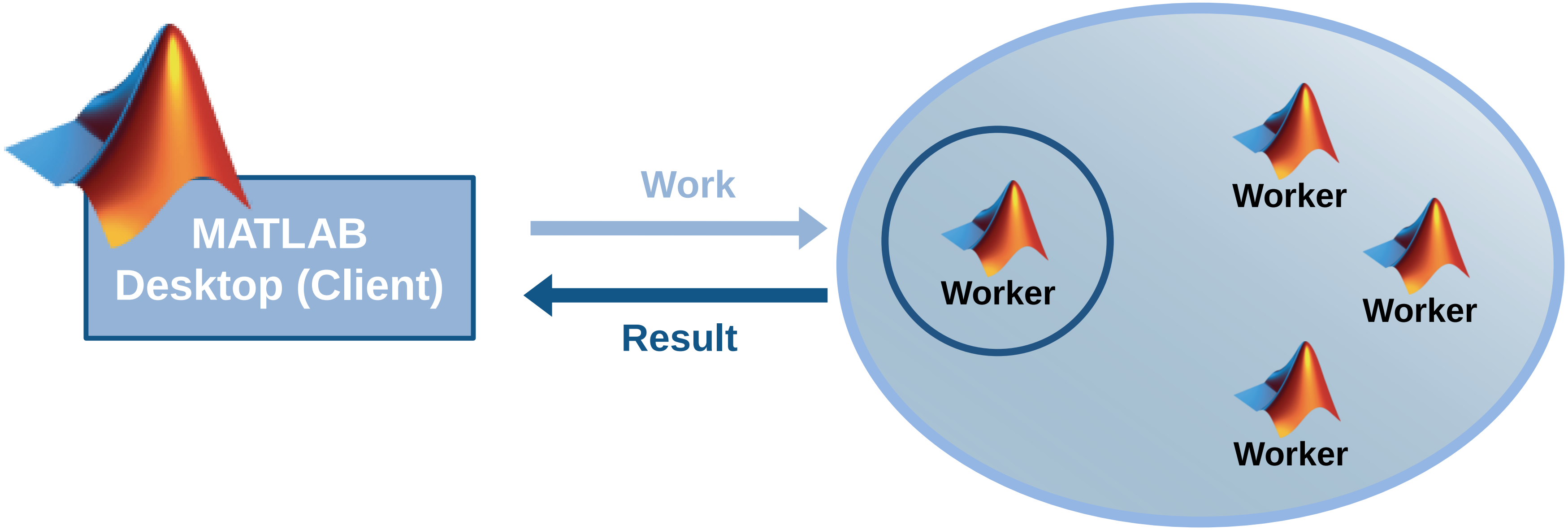
$$\overbrace{m}^{5}\ddot{x} + \underbrace{b}_{1,2,\ldots}\dot{x} + \underbrace{k}_{1,2,\ldots}x = 0$$

- Parameter sweep of ODE system
  - Damped spring oscillator
  - Sweep through different values of damping and stiffness
  - Record peak value for each simulation
- Convert `for` to `parfor`
- Use pool of MATLAB workers



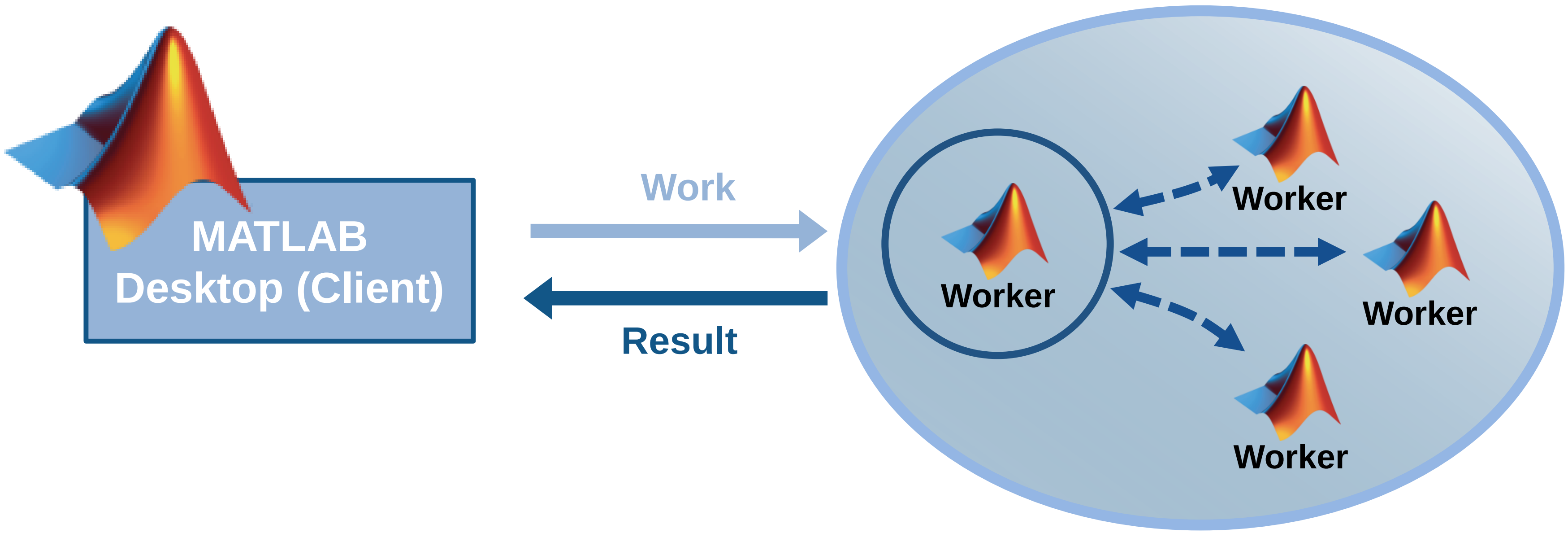


Offload Computations with batch



batch(...)

Offload and Scale Computations with `batch` & `pool`

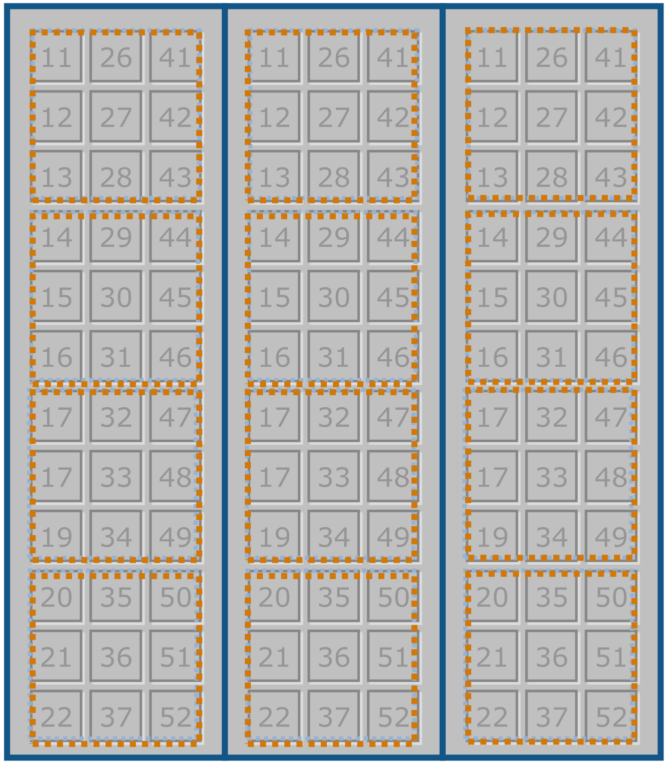



```
batch(…,'Pool',…)
```

distributed

Split data over multiple cores

```
parpool('local')
A = distributed.randn(1000);
[V,D] = eig((A+A')/2);
D = gather(D);
```



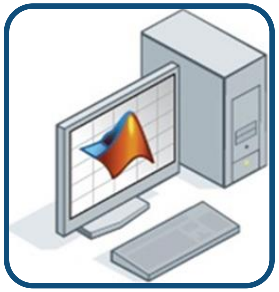
Programming Parallel Applications (CPU)

Ease of Use

- Parallel enabled functionality in toolboxes
- Simple programming constructs: **`parfor, batch, distributed`**
- Advanced programming constructs: **`createJob, spmd, labSend/labReceive`**

Greater Control

Agenda



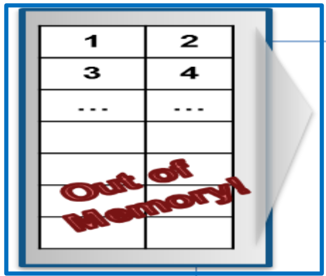
Accelerating MATLAB code with parallel computing

Scaling up: accessing more/better hardware

Big data with MATLAB: an overview

Using more hardware: why and when?

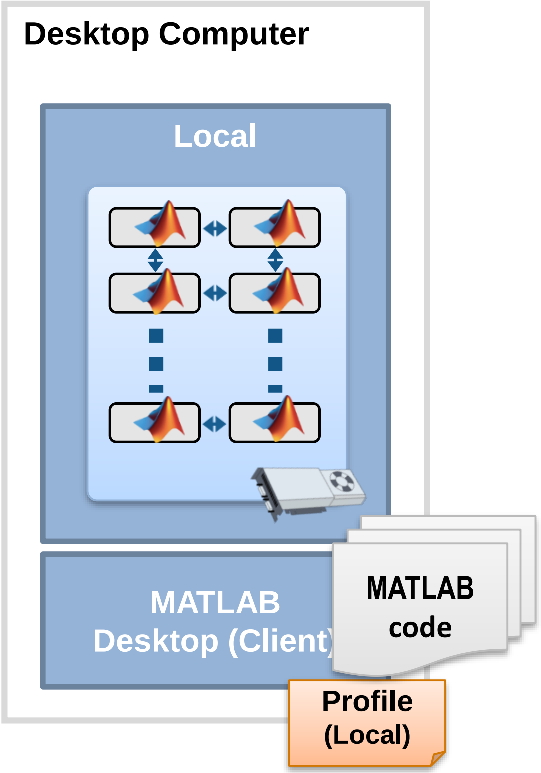
We have seen that with PCT we get some speed up locally and we have access to a powerful server or cluster with MDCS



Our data is too large to fit in the desktop memory

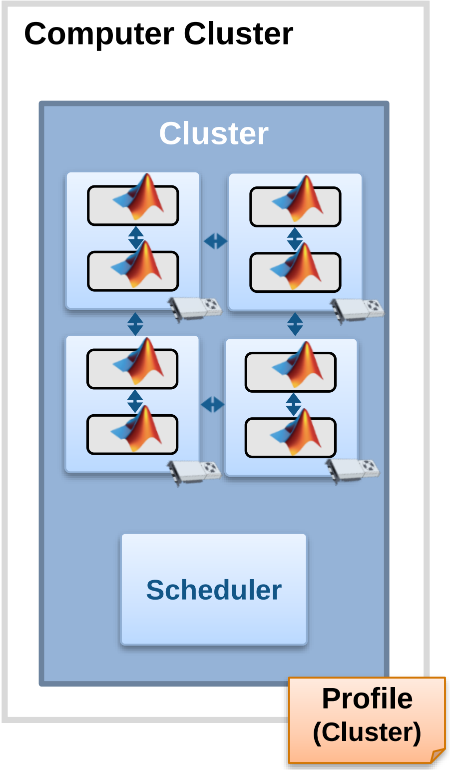
We have access to GPUs

Use MATLAB Distributed Computing Server



1. Prototype code

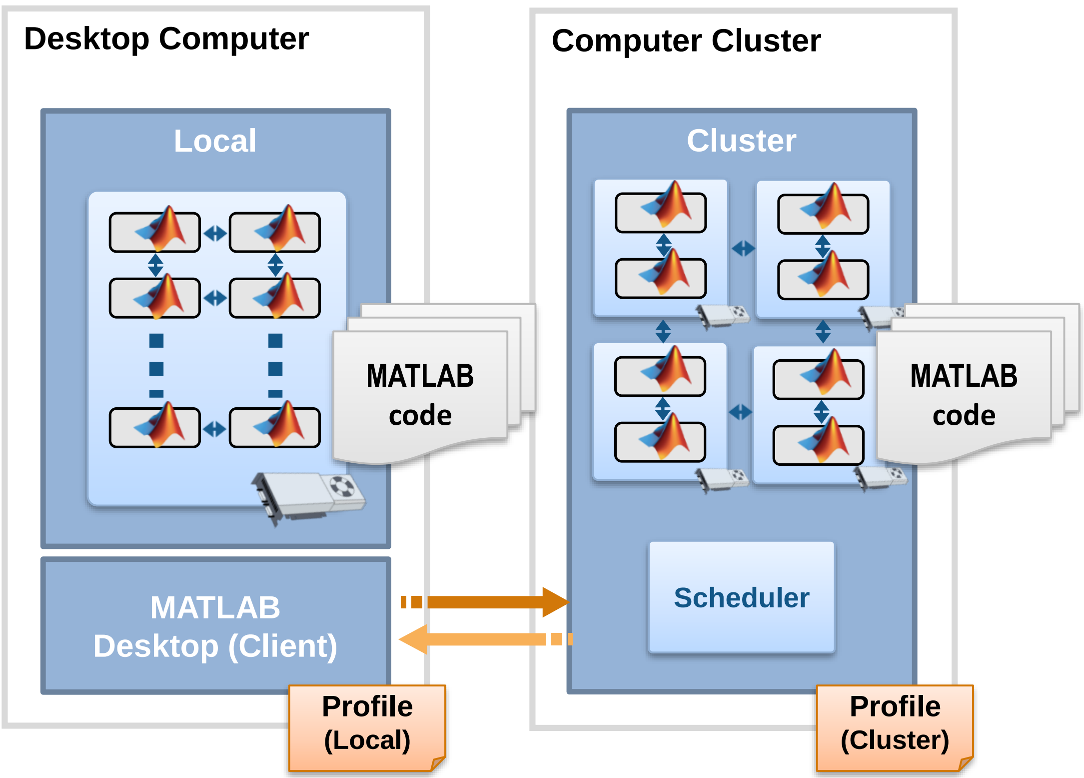
Use MATLAB Distributed Computing Server



1. Prototype code

2. Get access to an enabled cluster

Use MATLAB Distributed Computing Server

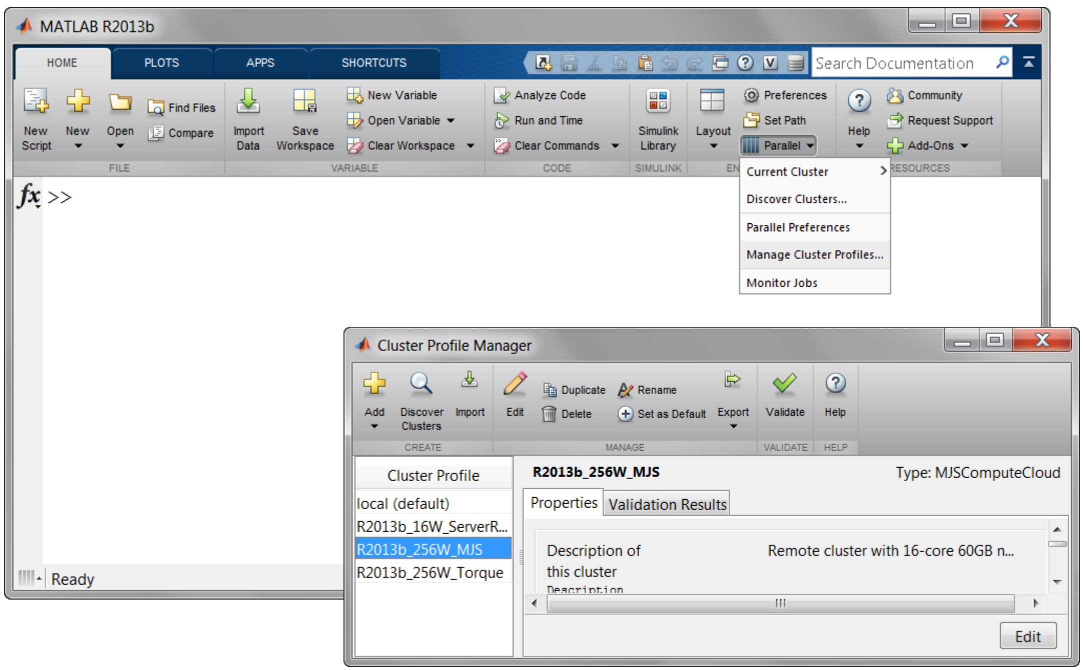


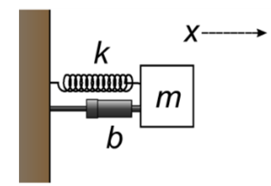
1. Prototype code
2. Get access to an enabled cluster
3. Switch cluster profile to run on cluster resources

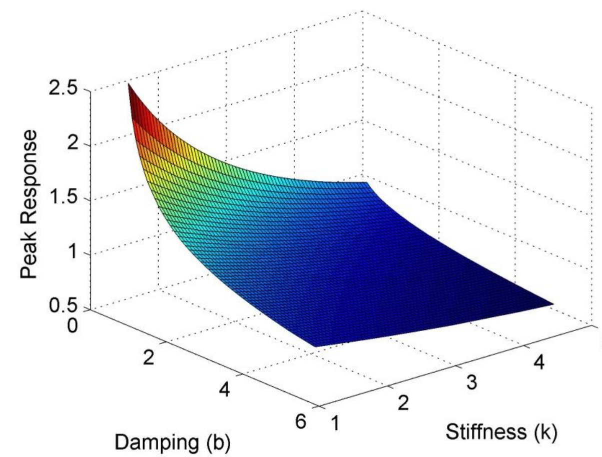
Example: Migrate from Desktop to Cluster

- Change hardware without changing algorithmic code

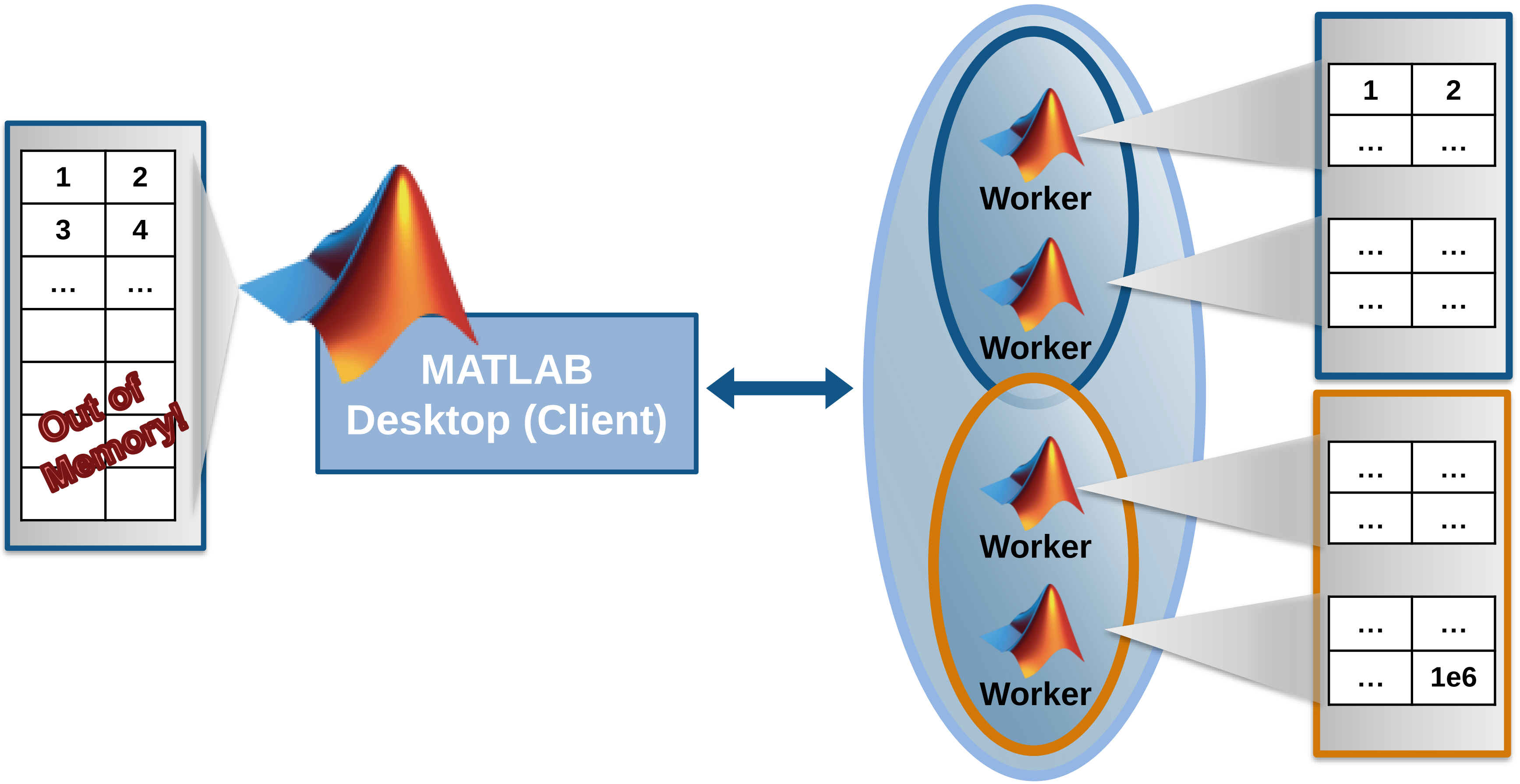
$$\overbrace{m}^{5}\ddot{x} + \underbrace{b}_{1,2,\ldots}\dot{x} + \underbrace{k}_{1,2,\ldots}x = 0$$







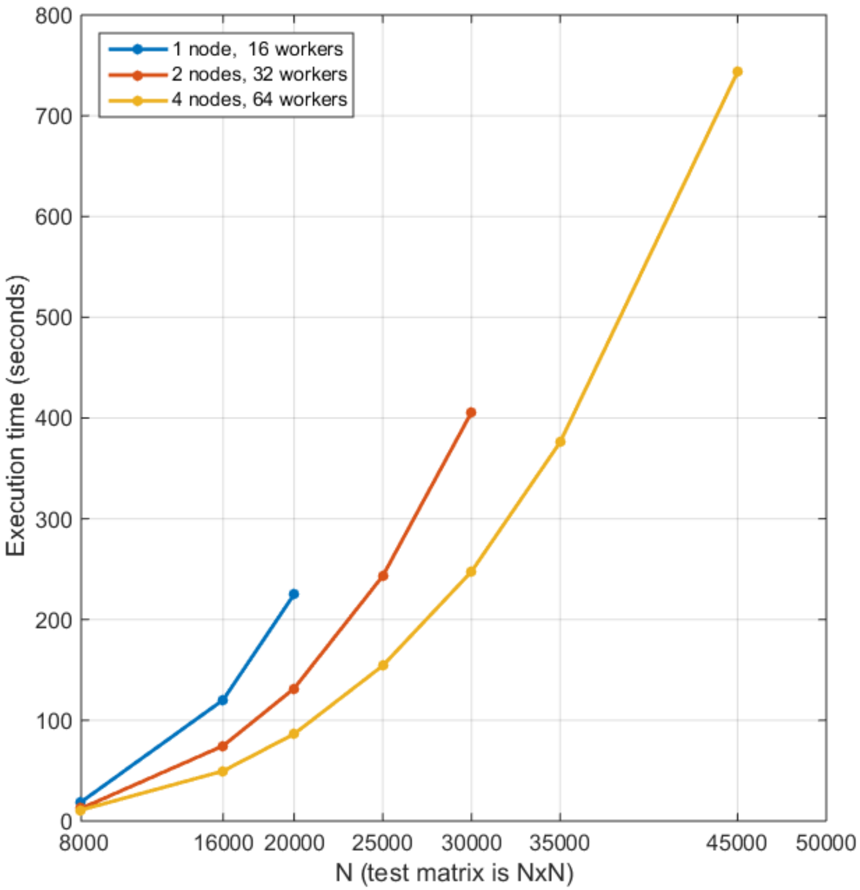
Overcoming Desktop Memory Limitations



www.mathworks.com/help/distcomp/using-matlab-functions-on-codistributed-arrays.html

Distributed Memory Calculations

Multiplication of 2 NxN matrices

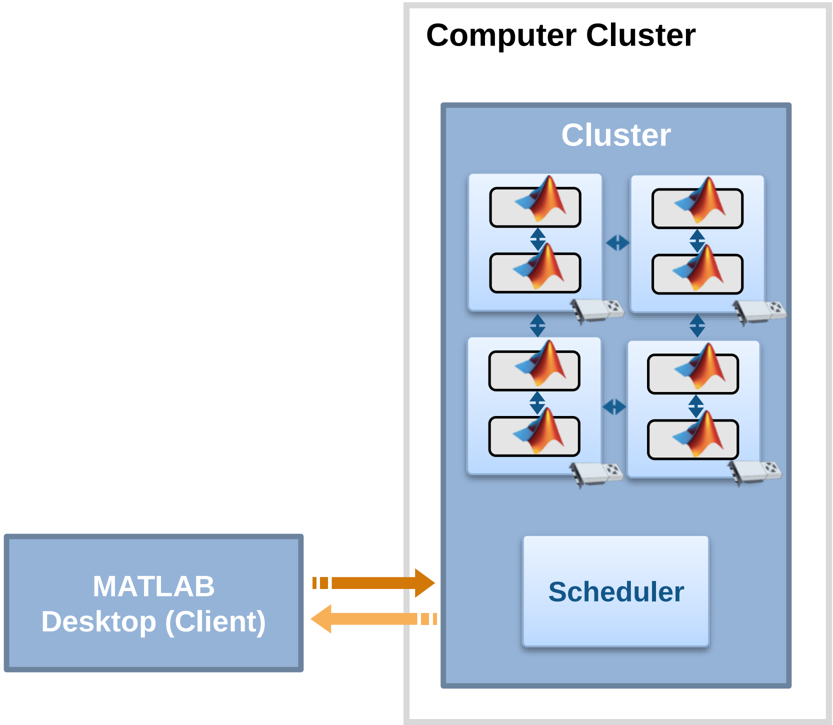


N	Execution time (seconds)		
	1 node, 16 workers	2 nodes, 32 workers	4 nodes, 64 workers
8000	19	13	11
16000	120	75	50
20000	225	132	86
25000	-	243	154
30000	-	406	248
35000	-	-	376
45000	-	-	743
50000	-	-	-

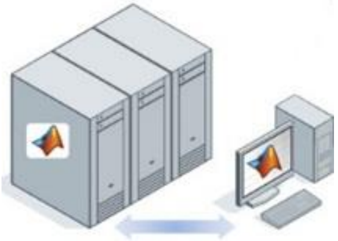
Processor: Intel Xeon E5-class v2
16 cores, 60 GB RAM per compute node, 10 Gb Ethernet

Take Advantage of Cluster Hardware

- Offload computation:
  - Free up desktop
  - Access better computers
- Scale speed-up:
  - Use more cores
  - Go from hours to minutes
- Scale memory:
  - Utilize distributed arrays
  - Solve larger problems without re-coding algorithms
- Leverage supplied infrastructure
  - File transfer / path augmentation
  - Job monitoring



Further Scaling Big Data Capacity



MATLAB supports a number of programming constructs for use with clusters

- General compute clusters
  - *Parallel for loops – embarrassingly parallel algorithms*
  - *SPMD – distributed processing*
- Hadoop clusters
  - *MapReduce – analyze data stored in the Hadoop Distributed File System*

What is a Graphics Processing Unit (GPU)

- Originally for graphics acceleration, now also used for scientific calculations
- Massively parallel array of integer and floating point processors
  - Typically hundreds of processors per card
  - GPU cores complement CPU cores
- Dedicated high-speed memory

* Parallel Computing Toolbox requires NVIDIA GPUs with Compute Capability 1.3 or higher, including NVIDIA Tesla 20-series products. See a complete listing at www.nvidia.com/object/cuda_gpus.html

Programming Parallel Applications (GPU)

Ease of Use

- Built-in support with Toolboxes
- Simple programming constructs:
  `gpuArray, gather`
- Advanced programming constructs:
  `arrayfun, bsxfun, spmd`
- Interface for experts:
  `CUDAKernel, MEX support`

Greater Control

Accessing GPUs (Graphics Processing Units)

```
>> A = someArray(200, 10000);
>> G = gpuArray(A); % Push to GPU memory
>> F = cov(G);
>> B = gather(F) % Bring back into MATLAB
```

To access GPU functionality you will need R2010b or later, and NVIDIA GPUs with Compute Capability 1.3 or greater including NVIDIA Tesla 10-series and 20-series products: www.nvidia.com/object/cuda_gpus.html

CPU/GPU comparison

```
%% CPU
Acpu = rand(2000);

tic
Bcpu = cov(Acpu);
toc
```

Elapsed time is 0.264531 seconds.

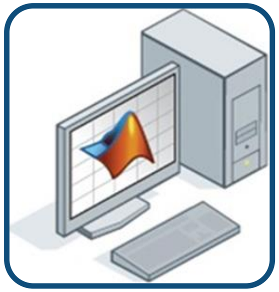
```
%% GPU
Agpu = gpuArray.rand(2000);

tic
Bgpu = cov(Agpu);
wait(gpuDevice);
toc

B = gather(Bgpu);
```

Elapsed time is 0.041266 seconds.

Summary

- Develop parallel MATLAB applications without being a parallel programming expert
- Speed up the execution of your MATLAB applications using additional hardware
- Prototype on your desktop and easily scale to a cluster

Agenda



Accelerating MATLAB code with parallel computing

Scaling up: accessing more/better hardware

Big data with MATLAB:

an overview

Big Data

“Any collection of data sets so large and complex that it becomes difficult to process using… traditional data processing applications.”

(Wikipedia)

FINANCE
Market Risk, Regulatory

AUTO
Fleet Data Analysis

Medical Devices
Patient Outcomes

ENERGY
Asset Optimization

- **Access to Data**
- **Rapid data exploration**
- **Development of scalable algorithms**
- **Ease of deployment**

Considerations: large data analytics

Data Characteristics

1. Size & type of data?
2. Where is your data?
3. What hardware do you have access to?
4. Analysis Characteristics
   - Embarrassingly Parallel
   - Analyse sub-segments of data and aggregate results
   - Operate on entire dataset

Big Data Capabilities in MATLAB

Memory and Data Access

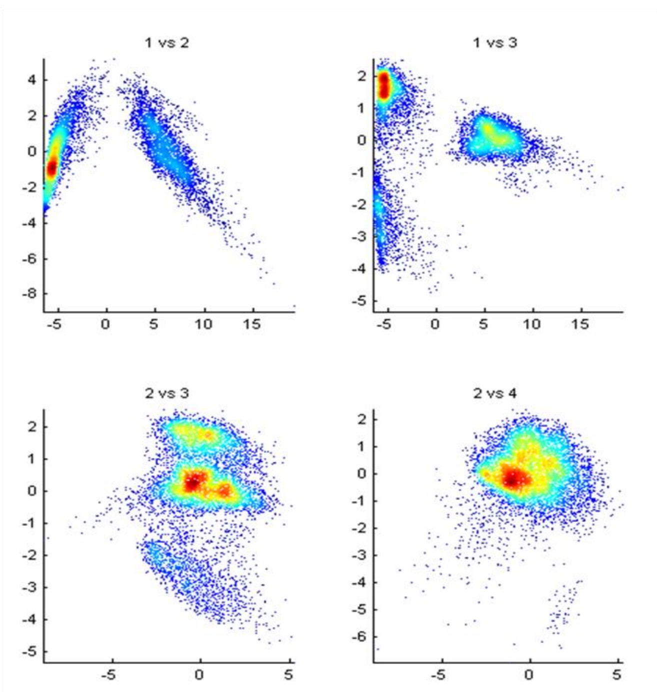
- 64-bit processors
- Memory Mapped Variables
- Disk Variables
- Databases
- Datastores

Programming Constructs

- Streaming
- Block Processing
- Parallel-for loops
- GPU Arrays
- SPMD and Distributed Arrays
- MapReduce

Platforms

- Desktop (Multicore, GPU)
- Clusters
- Cloud Computing (MDCS on EC2)
- Hadoop

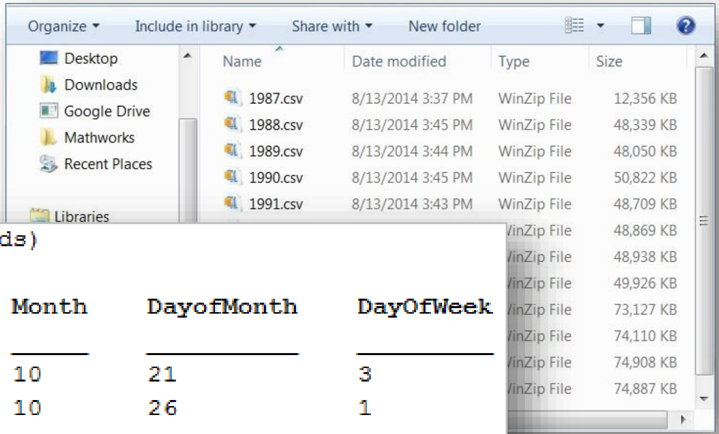


Big Data on the Desktop
MATLAB

datastore

Access files or collections of files that do not fit into memory

- Preview data structure and format
- Import data using column names
- Incremental access to data




```
>> preview(ds)
ans =
    Year      Month     DayofMonth     DayOfWeek
    ____      _____     __________     _________
    1987      10        21             3
    1987      10        26             1
    1987      10        23             5
```

```
airdata = datastore('*.csv');
airdata.SelectedVariables = {'Distance', 'ArrDelay'};

data = read(airdata);
```

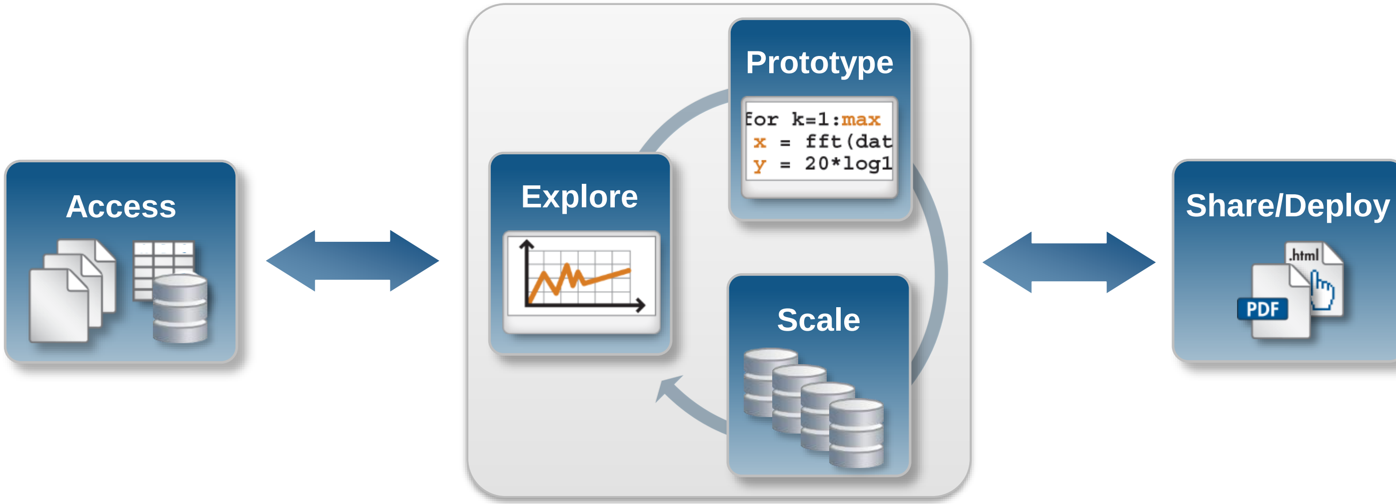
mapreduce

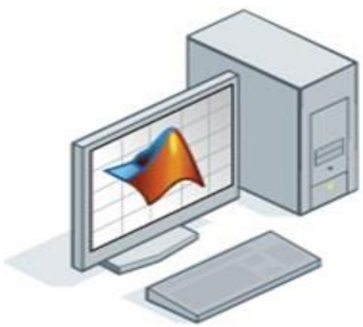
Programming technique for analyzing data sets that do not fit into memory

- Tabular text files or big database tables
- Increase compute capacity with parallel processing
- Access HDFS to develop algorithms for use on Hadoop

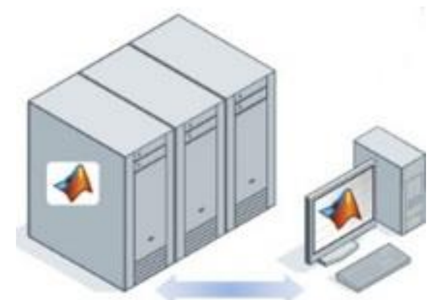
```
**********************************
*      MAPREDUCE PROGRESS        *
**********************************
Map   0%          Reduce   0%
Map  20%          Reduce   0%
Map  40%          Reduce   0%
Map  60%          Reduce   0%
Map  80%          Reduce   0%
Map 100%          Reduce  25%
Map 100%          Reduce  50%
Map 100%          Reduce  75%
Map 100%          Reduce 100%
```

Big Data Analytics with MATLAB





Work on the desktop



Scale capacity as needed