Parallel Computing with MATLAB

Jiro Doke, Ph.D.
Senior Application Engineer

Sarah Wait Zaranek, Ph.D.
MATLAB Product Marketing

A Question to Consider

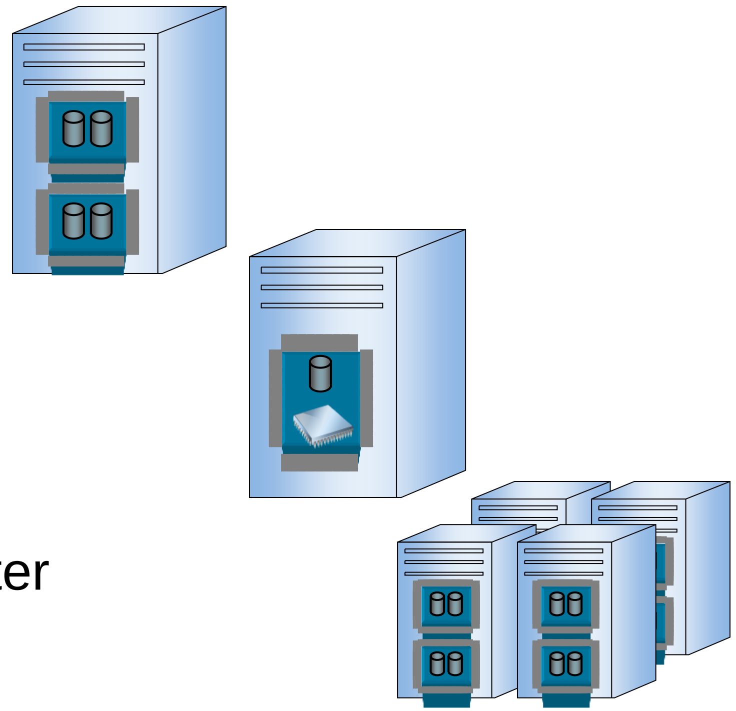
Do you want to speed up your algorithms or deal with large data?

If so…

– Do you have a multi-core or multi-processor computer?

– Do you have a high-end graphics processing unit (GPU)?

– Do you have access to a computer cluster?



Utilizing Additional Processing Power

- Built-in multithreading (implicit)
  - Core MATLAB and Image Processing Toolbox
  - Utility for specific matrix operations (linear algebra, fft, filter, etc)
  - No necessary code change

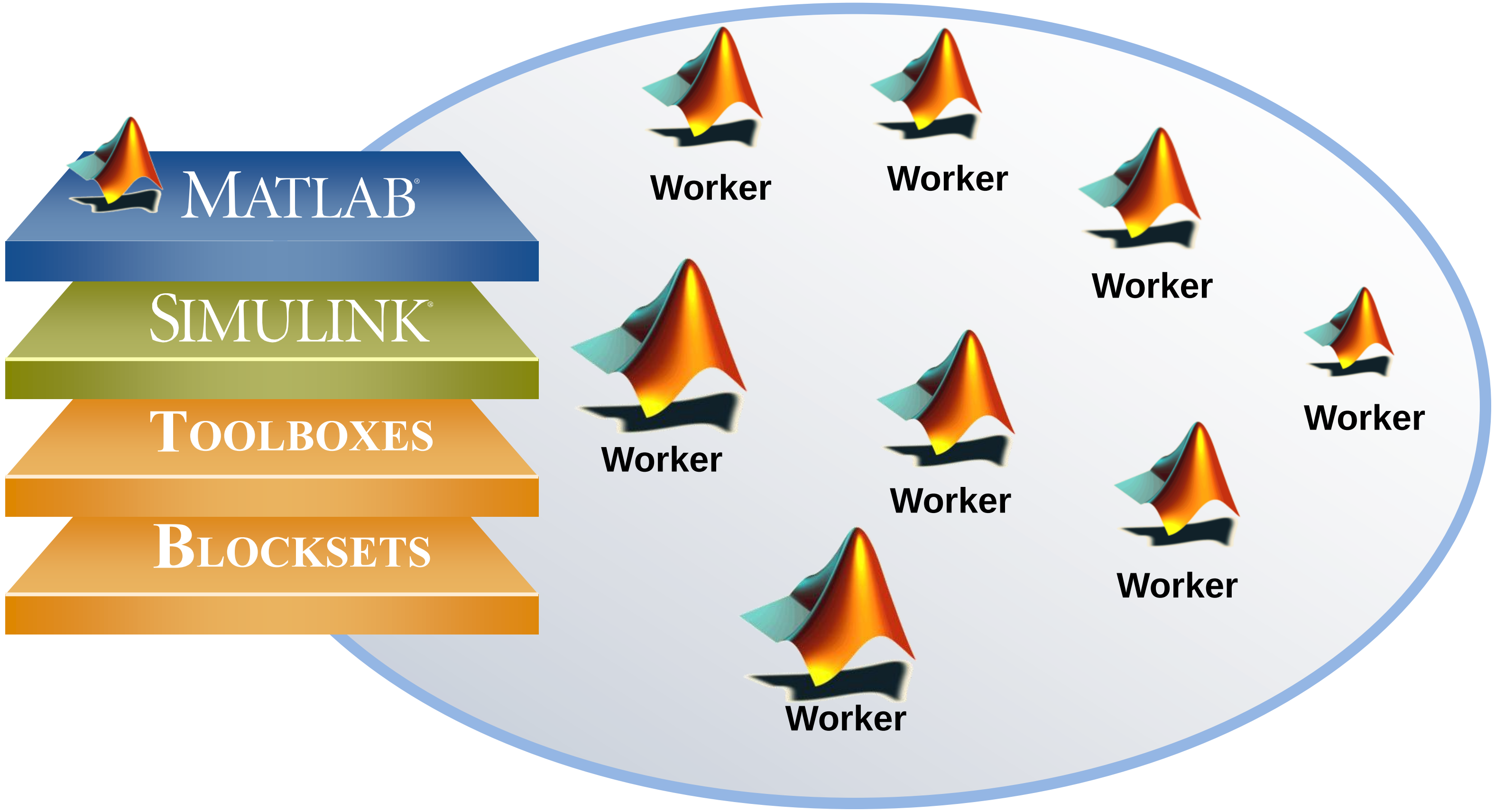
- Parallel computing tools (explicit)
  - Parallel Computing Toolbox
  - MATLAB Distributed Computing Server
  - Broad utility controlled by the MATLAB user

Agenda

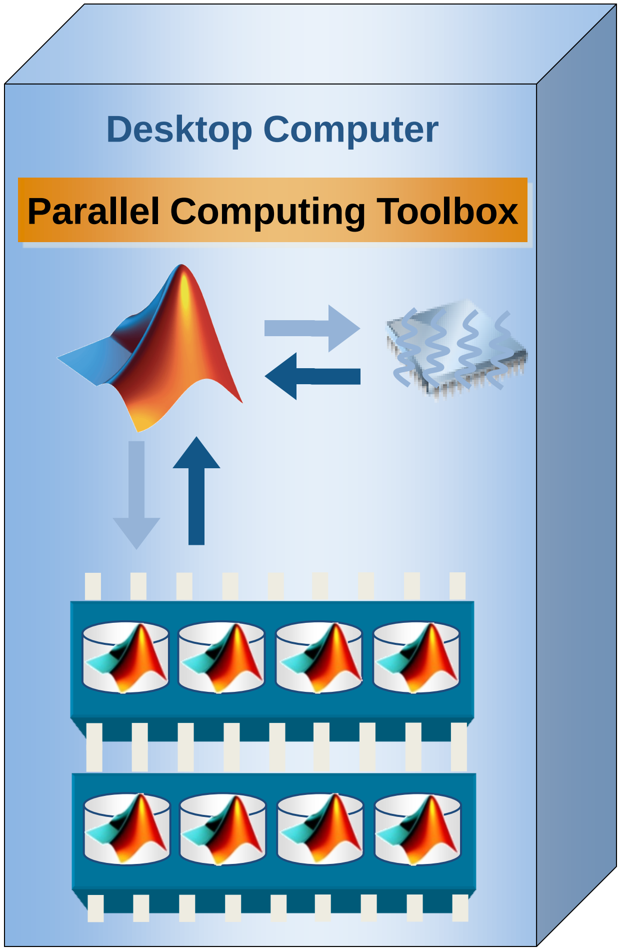
Introduction to Parallel Computing Tools

- Using Multi-core/Multi-processor Machines
- Using Graphics Processing Units (GPUs)
- Scaling Up to a Cluster

Going Beyond Serial MATLAB Applications

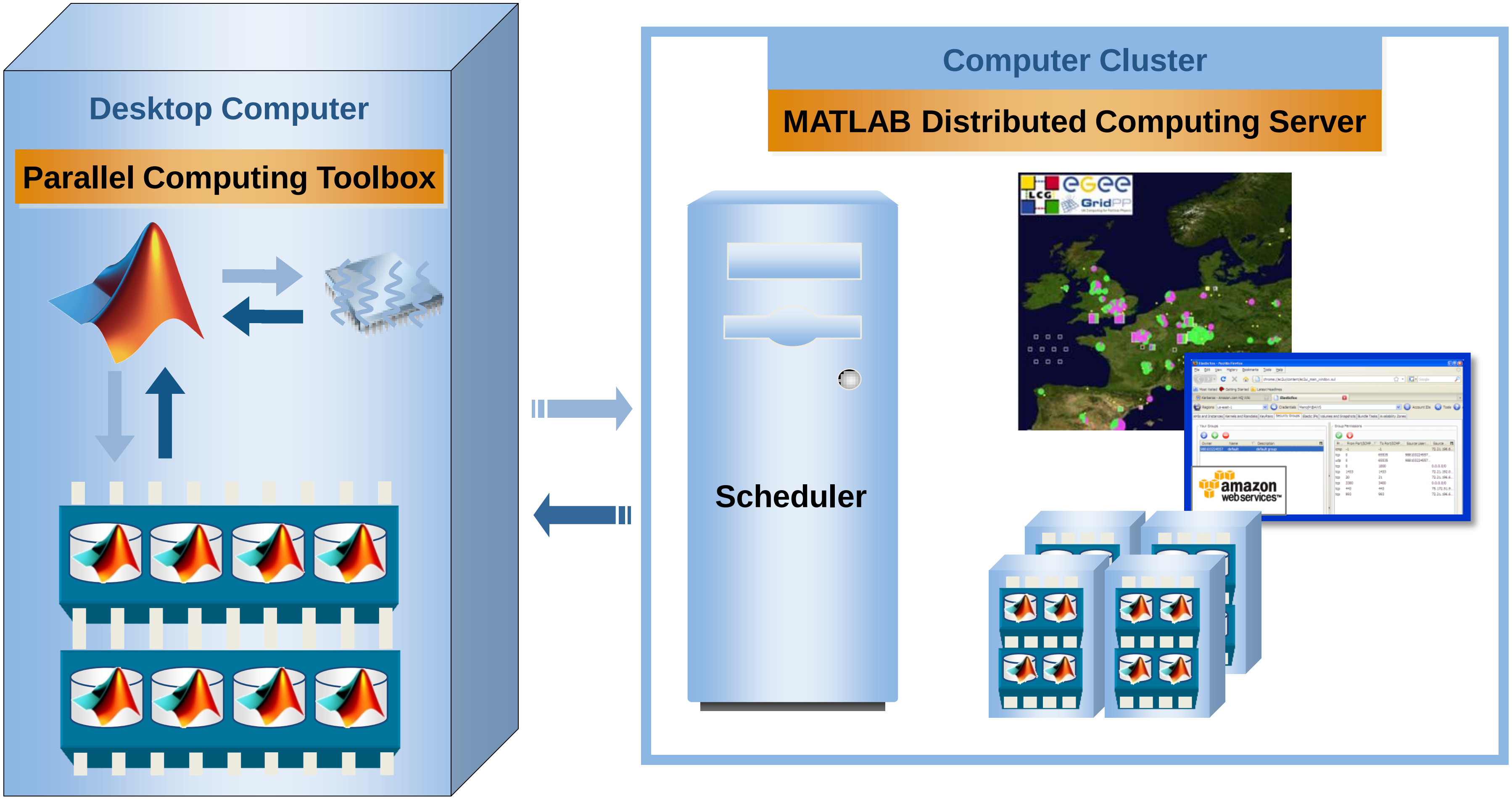


Parallel Computing on the Desktop



- Use Parallel Computing Toolbox
- Speed up parallel applications on local computer
- Take full advantage of desktop power by using CPUs and GPUs (up to 12 workers in R2011b)
- Separate computer cluster not required

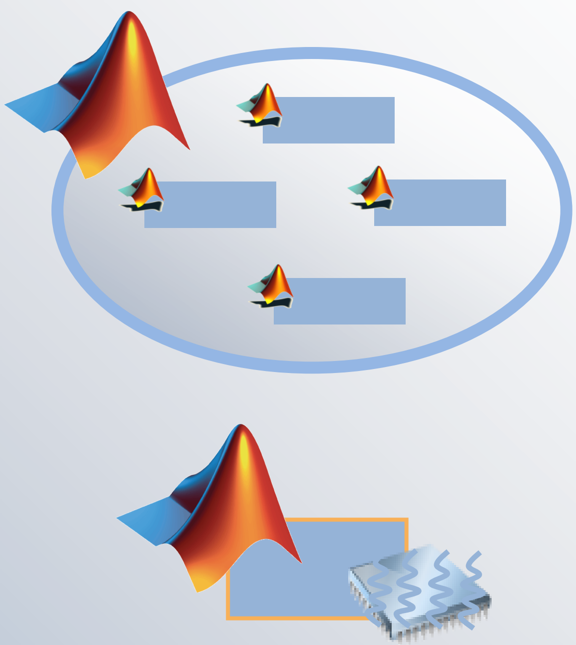
Scale Up to Clusters, Grids and Clouds



Parallel Computing enables you to …

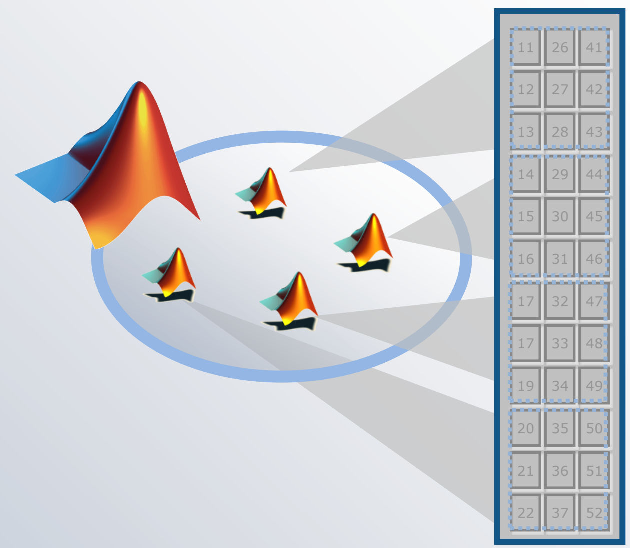
Larger Compute Pool

Speed up Computations



Larger Memory Pool

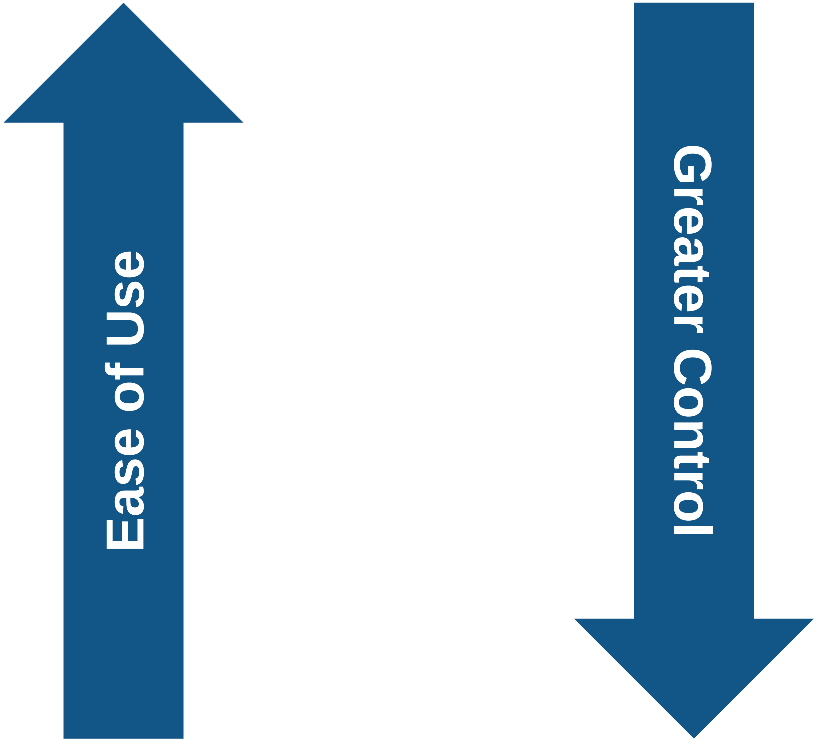
Work with Large Data



Agenda

- Introduction to Parallel Computing Tools
- Using Multi-core/Multi-processor Machines
- Using Graphics Processing Units (GPUs)
- Scaling Up to a Cluster

Programming Parallel Applications



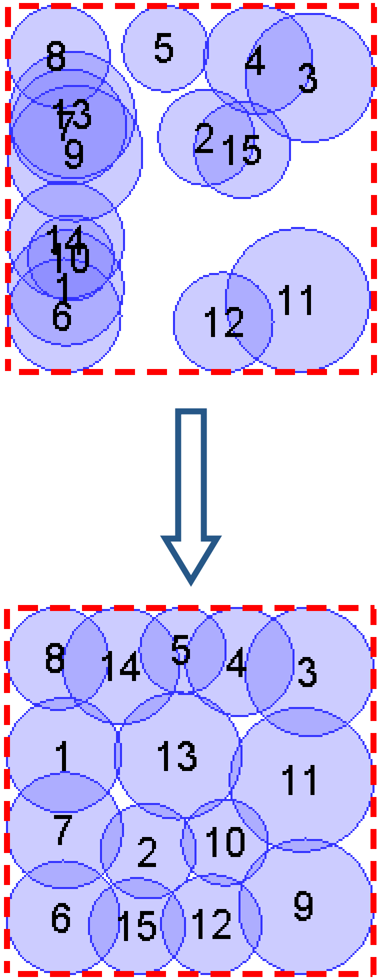
Using Additional Cores/Processors (CPUs)

- Support built into Toolboxes

Example:

Built-in Support for Parallelism in Other Tools

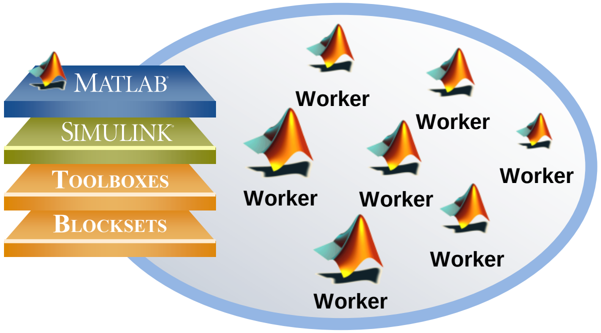
- Use built-in support for Parallel Computing Toolbox in Optimization Toolbox
- Run optimization in parallel
- Use pool of MATLAB workers



Other Tools Providing Parallel Computing Support

- Optimization Toolbox
- Global Optimization Toolbox
- Statistics Toolbox
- Simulink Design Optimization
- Bioinformatics Toolbox
- Model-Based Calibration Toolbox
- …

  http://www.mathworks.com/products/parallel-computing/builtin-parallel-support.html

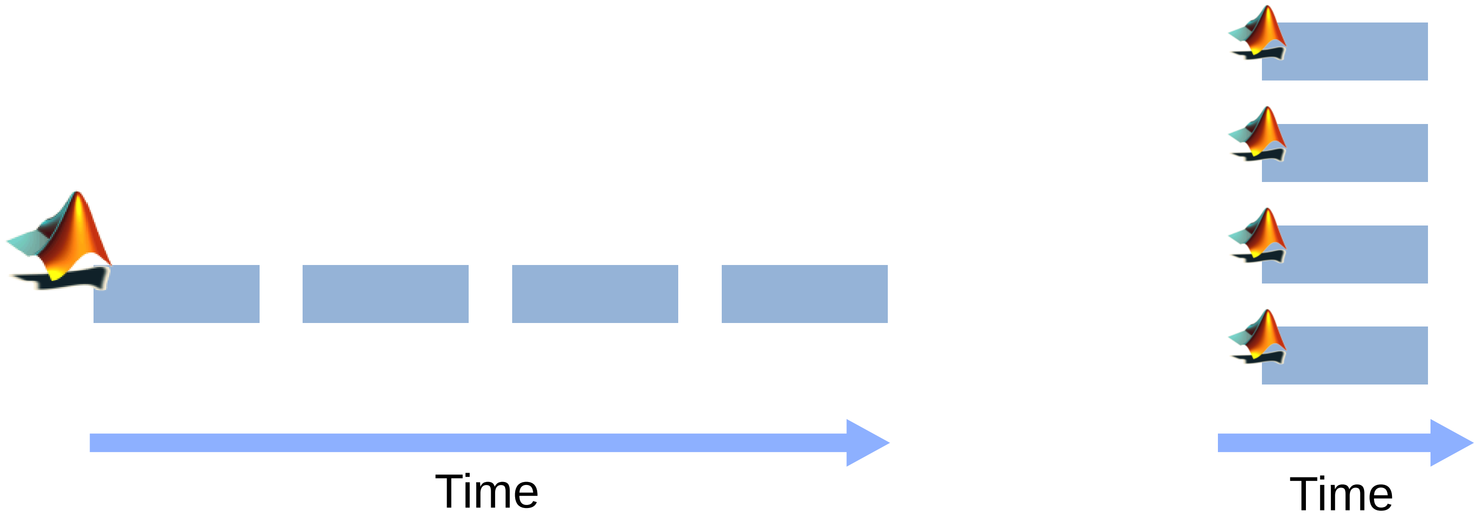


Using Additional Cores/Processors (CPUs)

- Support built into Toolboxes
- Simple programming constructs:
  `parfor`

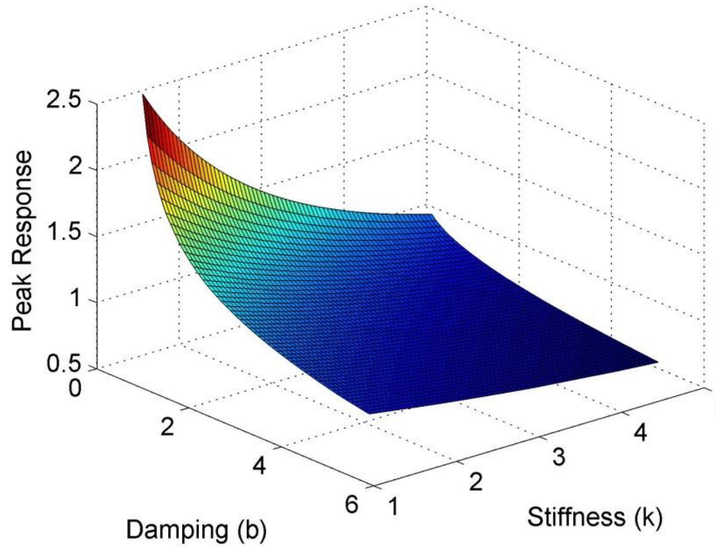
Running Independent Tasks or Iterations

- Ideal problem for parallel computing
- No dependencies or communications between tasks
- Examples include parameter sweeps and Monte Carlo simulations



Example:
Parameter Sweep of ODEs

- Parameter sweep of ODE system
- Use pool of MATLAB workers
- Convert `for` to `parfor`
- Interleave serial and parallel code



Damped spring oscillator

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

- Sweep through different values of ***b*** and ***k***
- Record peak value for each simulation

Tips for using `parfor`

- Requirement: Task and order independence

- Classification of Variables
  - One of the most common type of problems people run into when working with PARFOR.
  - At runtime, MATLAB needs determine how each variable would get treated.
  - Documentation: Parallel Computing Toolbox → User's Guide → Parallel for-Loops → Advanced Topics

- http://blogs.mathworks.com/loren/2009/10/02/using-parfor-loops-getting-up-and-running/

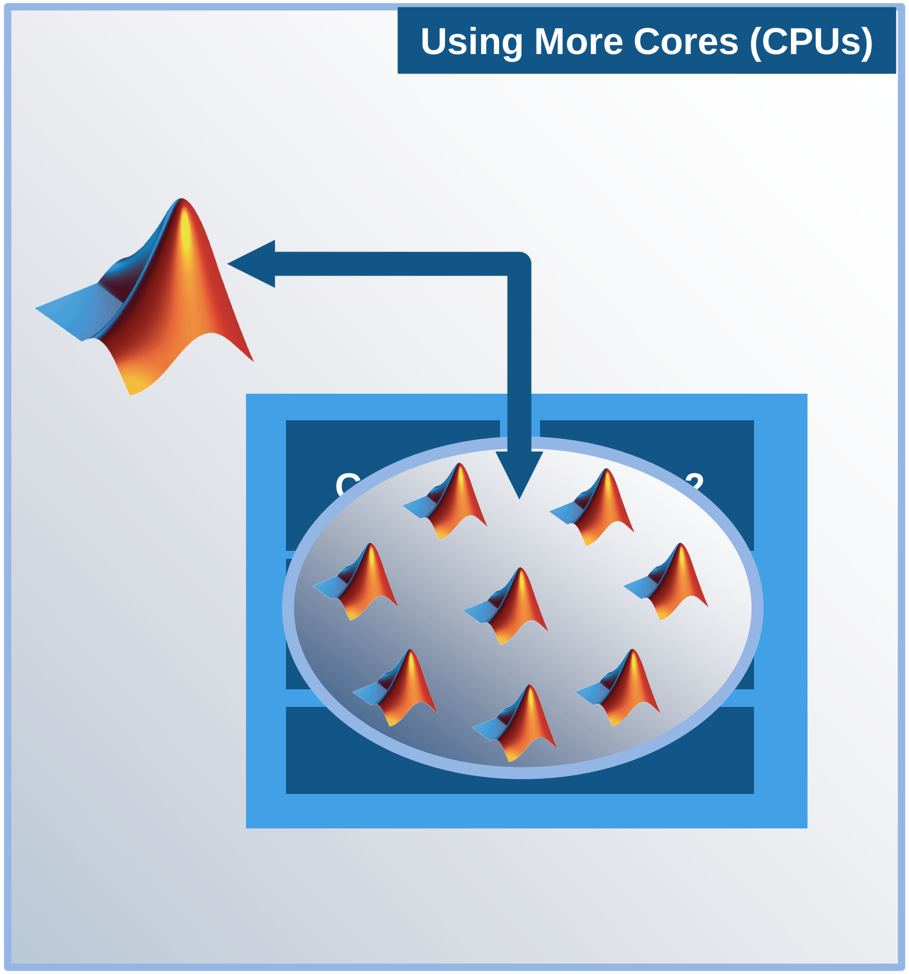
Using Additional Cores/Processors (CPUs)

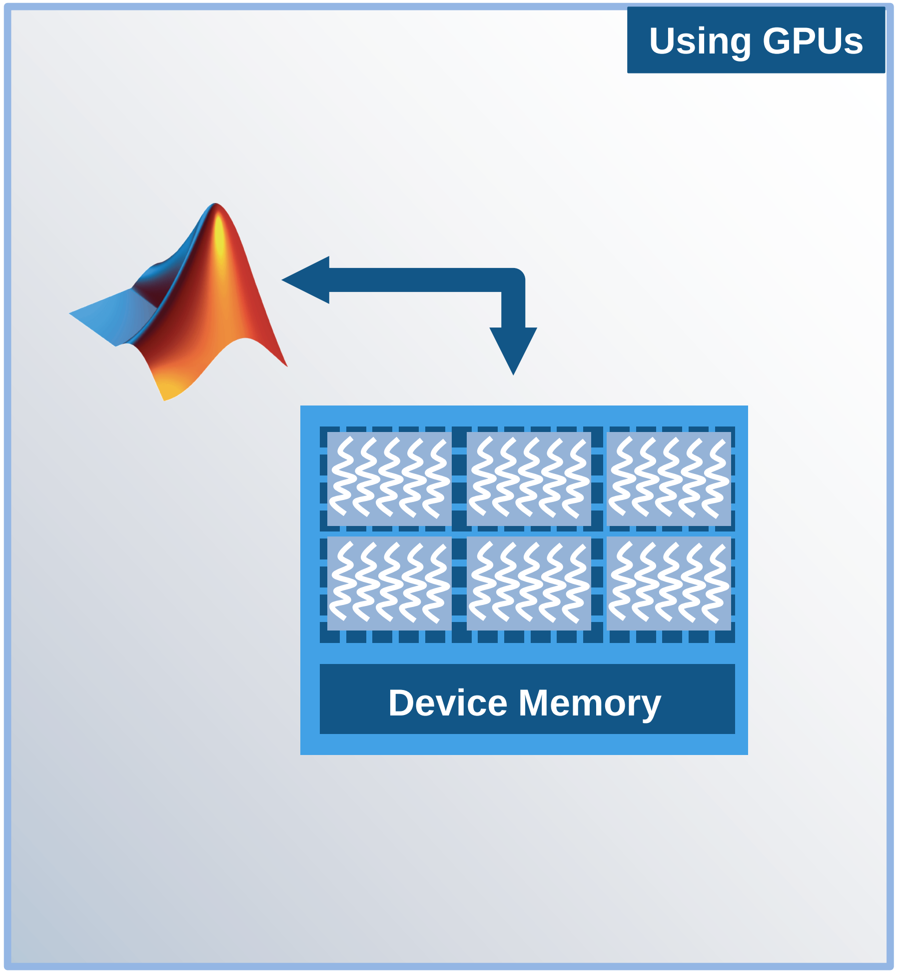
- Support built into Toolboxes
- Simple programming constructs: `parfor`
- Full control of parallelization: jobs and tasks

Agenda

- Introduction to Parallel Computing Tools
- Using Multi-core/Multi-processor Machines
- Using Graphics Processing Units (GPUs)
- Scaling Up to a Cluster

Gaining Performance with More Hardware





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

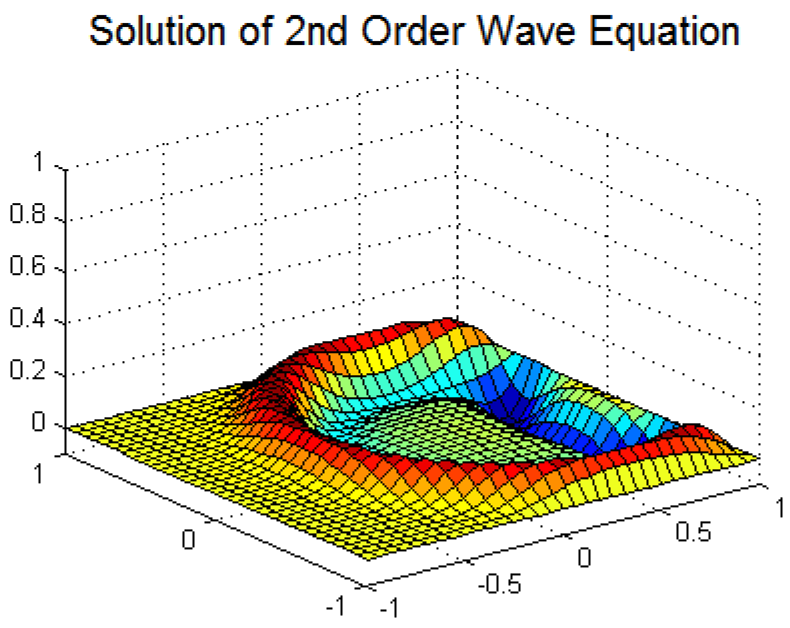
* Parallel Computing Toolbox requires NVIDIA GPUs with Compute Capability 1.3 or greater, including NVIDIA Tesla 10-series and 20-series products. See http://www.nvidia.com/object/cuda_gpus.html for a complete listing

Example:
GPU Computing in the Parallel Computing Toolbox

- Solve 2nd order wave equation:
$$\frac{\partial^2 u}{\partial t^2} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}$$

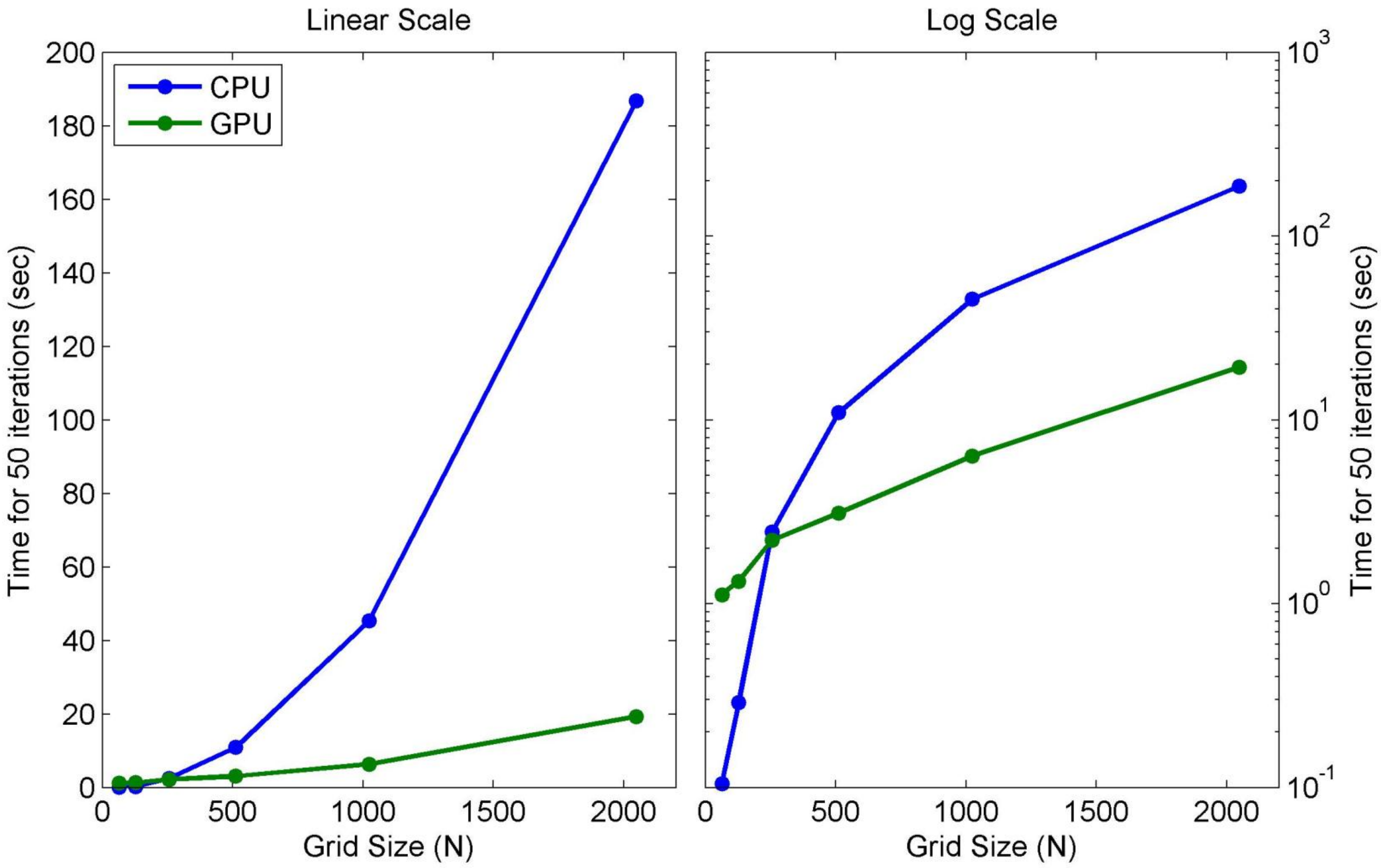
- Send and create data on the GPU

- Run calculations with built-in GPU functions



Benchmark: Solving 2D Wave Equation

CPU vs GPU



Summary of Options for Targeting GPUs

Ease of Use

- Use GPU array interface with MATLAB built-in functions
- Execute custom functions on elements of the GPU array
- Create kernels from existing CUDA code and PTX files

Greater Control

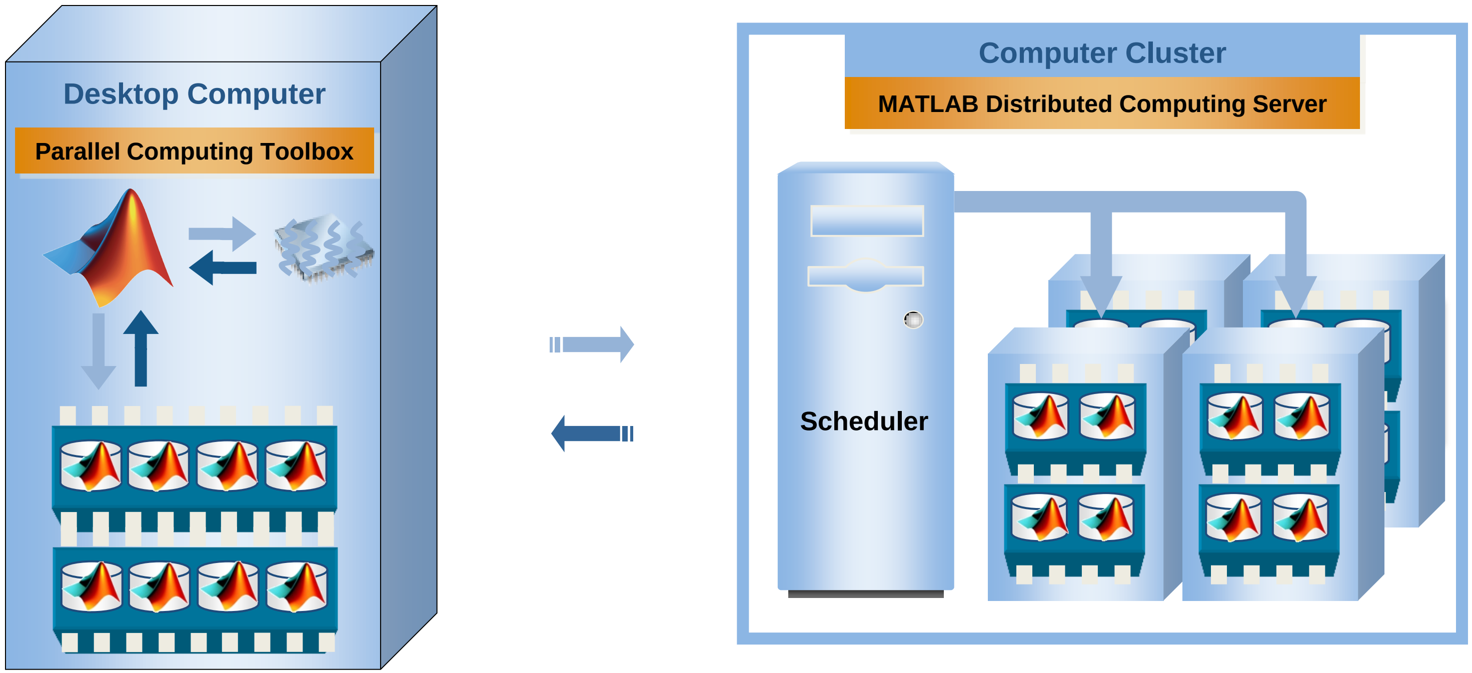
Webinar: “GPU Computing with MATLAB”

http://www.mathworks.com/company/events/webinars/wbnr59816.html

Agenda

- Introduction to Parallel Computing Tools
- Using Multi-core/Multi-processor Machines
- Using Graphics Processing Units (GPUs)
- Scaling Up to a Cluster

Setting Up Cluster Computing (for System Admins)



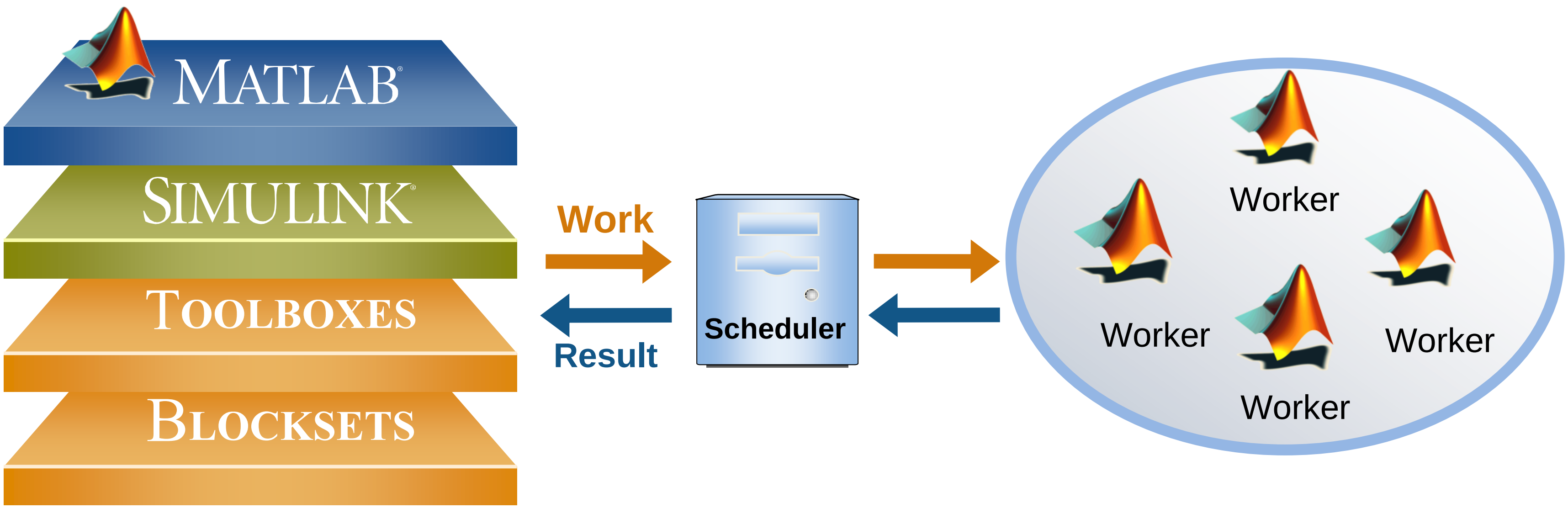
MATLAB Distributed Computing Server

- All-product install
- Worker license per process
- License by packs: 8, 16, 32, 64, etc.
- No additional toolbox licenses needed

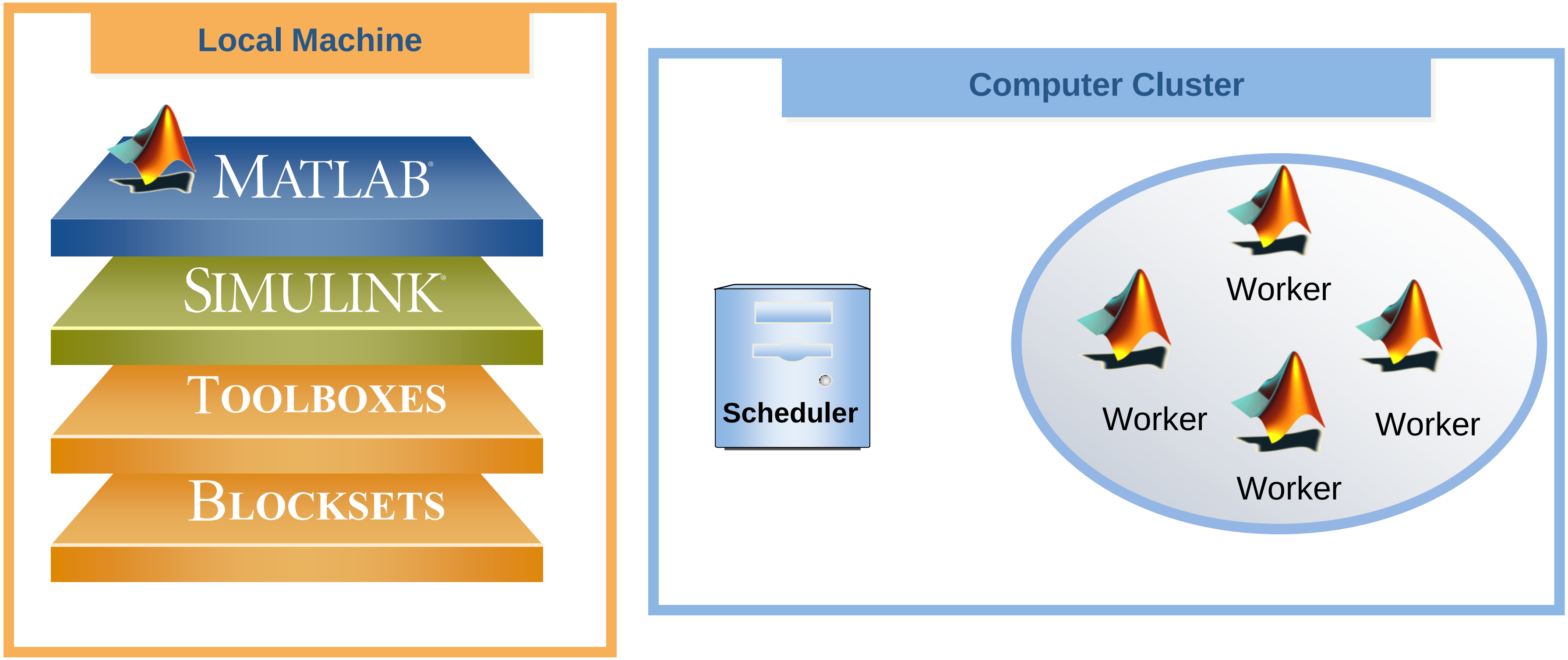
Why scale up to a cluster?

- Solve larger, computationally-intensive problems with more processing power
- Solve memory-intensive problems
- Schedule computations to offload from your local machine

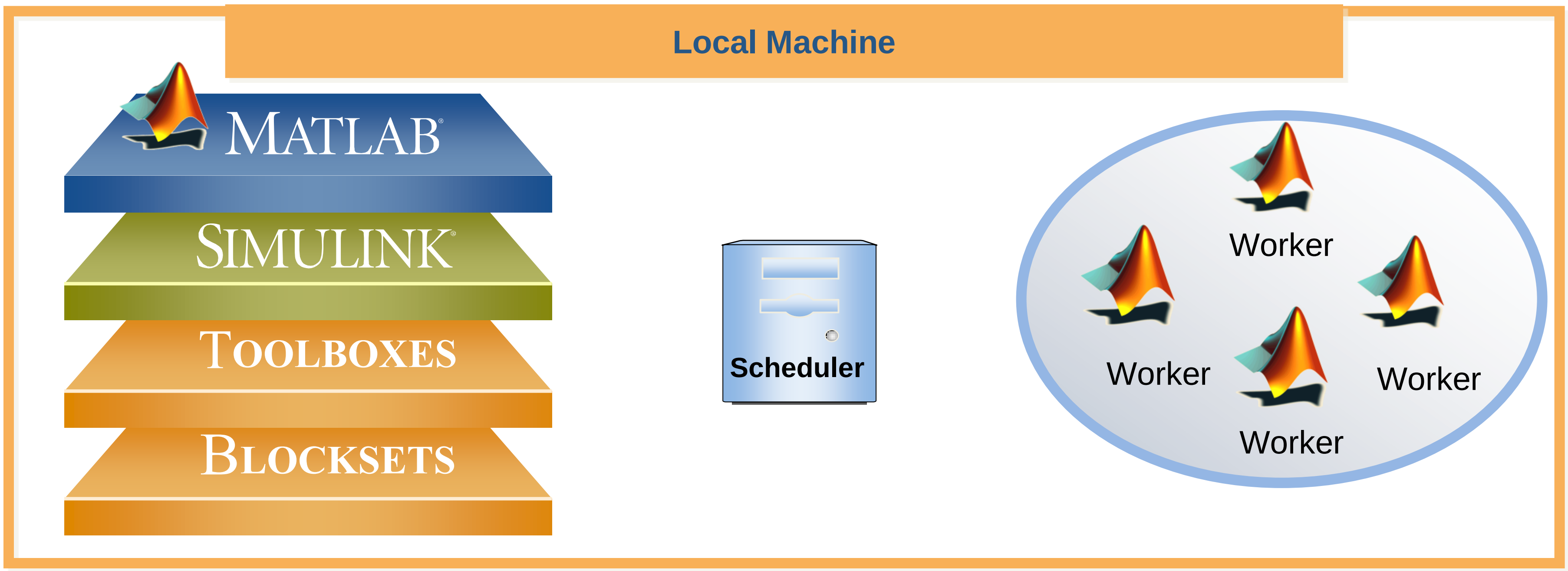
Scheduling Work (batch)



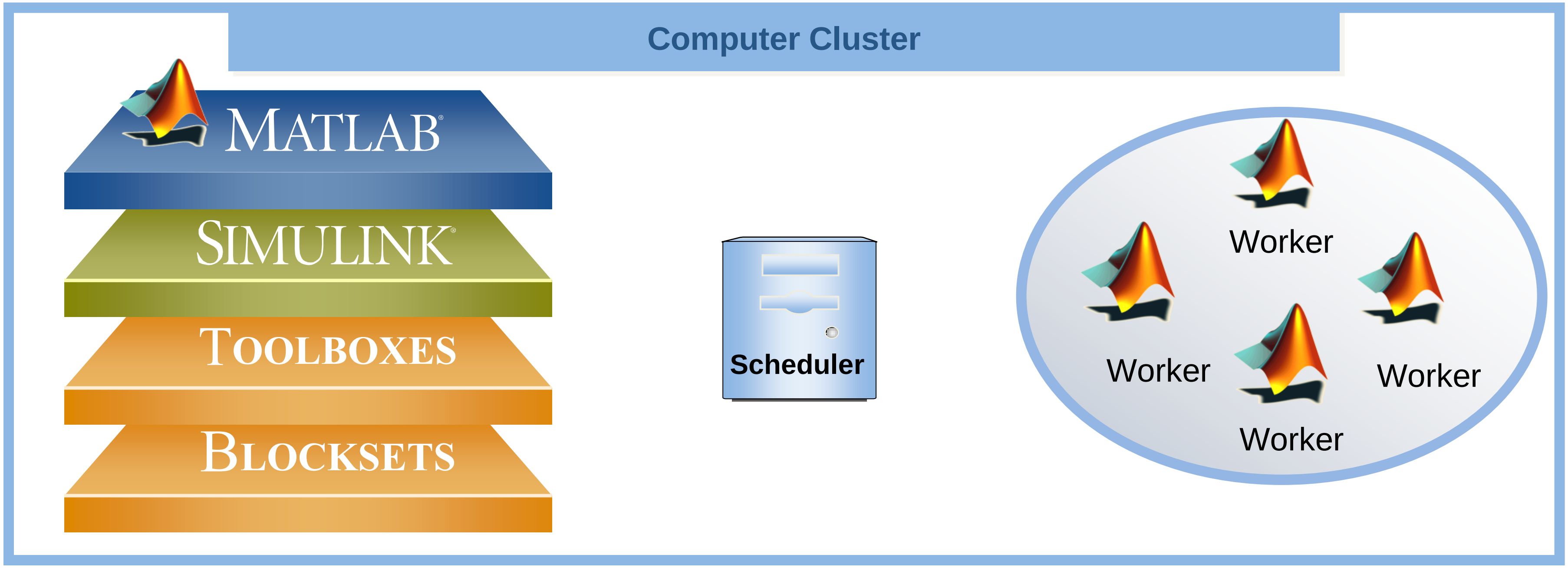
Scheduling Work (batch)



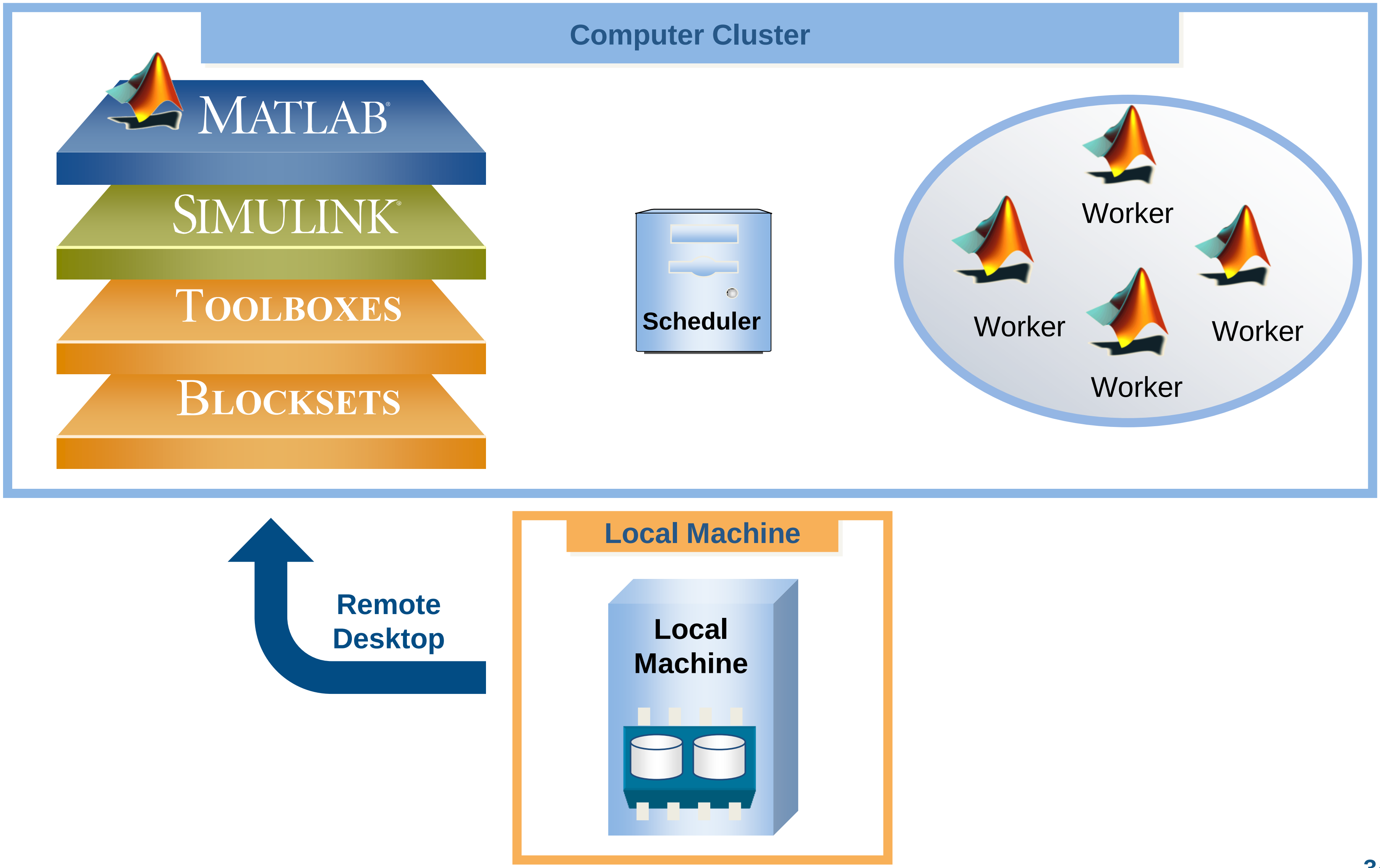
Scheduling Work (batch)



Scheduling Work (batch)

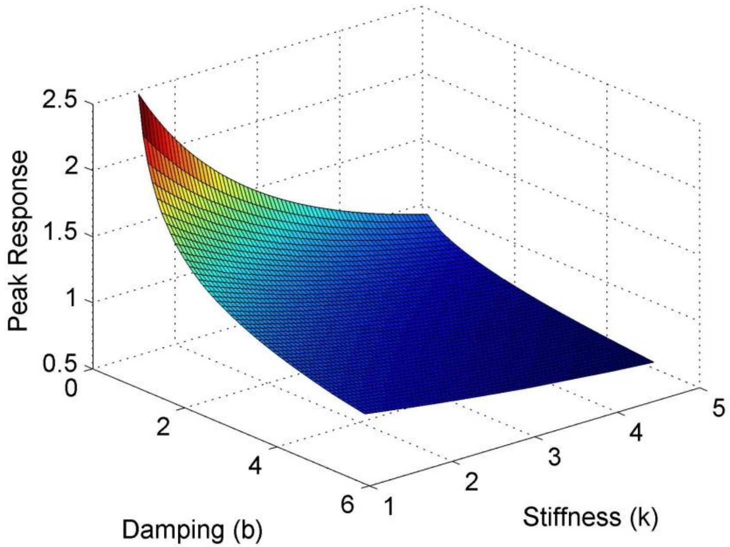


Scheduling Work (batch)



Example:
Scheduled Processing

- Offload processing to workers (local or cluster)
- Regain control of MATLAB after offloading
- Monitor progress of scheduled job
- Retrieve results from job

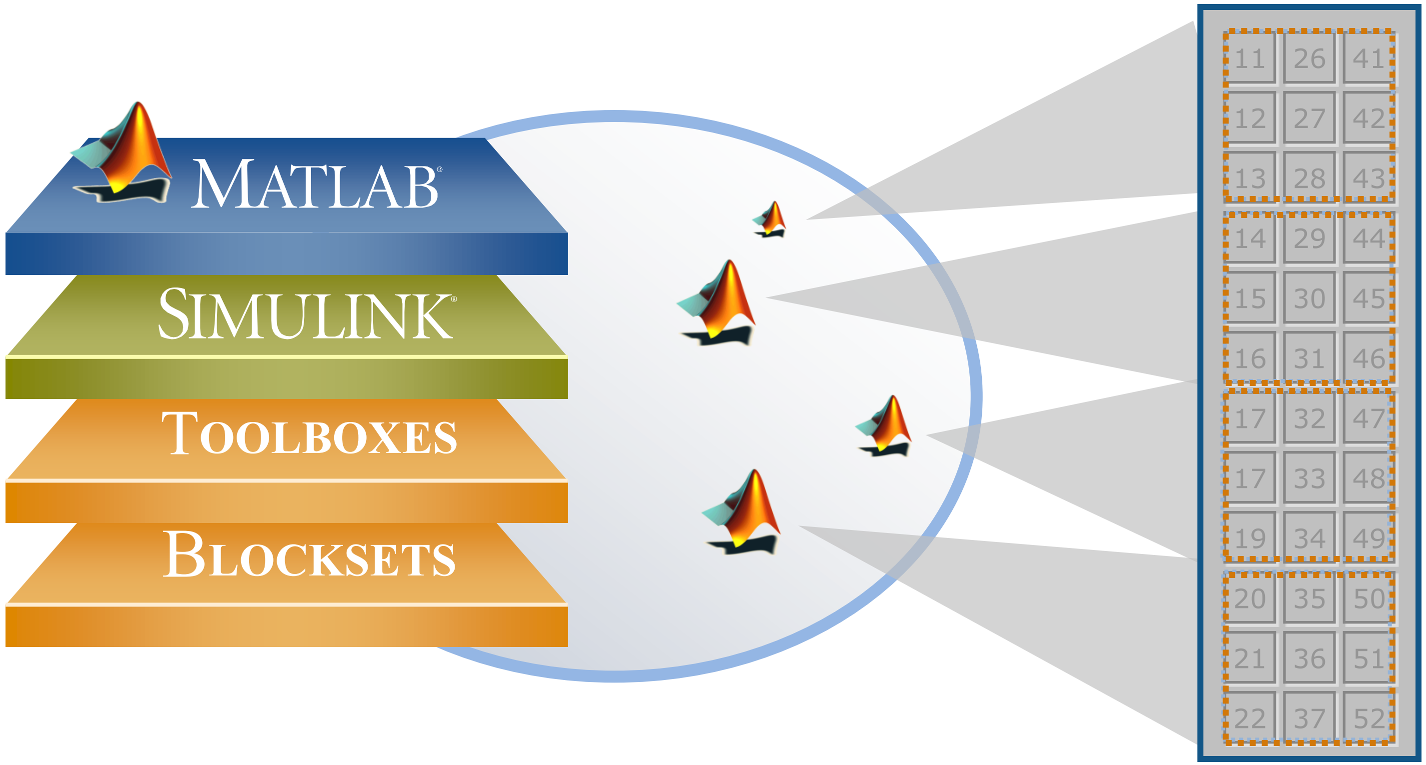


Damped spring oscillator

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

- Sweep through different values of ***b*** and ***k***
- Record peak value for each simulation

Distributing Large Data



Remotely Manipulate Array from Client MATLAB

Distributed Array Lives on the Workers

Client-side Distributed Arrays and SPMD

- Client-side distributed arrays
  - Class `distributed`
  - Can be created and manipulated directly from the client.
  - Simpler access to memory on labs
  - 100s of built-in functions that operate on distributed arrays, including client-side visualization functions

- `spmd`
  - Block of code executed on workers
  - Worker specific commands
  - Explicit communication between workers using MPI
  - Mixture of parallel and serial code

Summary

- Speed up parallel applications on desktop by using *Parallel Computing Toolbox*

- Take full advantage of CPU and GPU hardware

- Use *MATLAB Distributed Computing Server* to
  - Scale up to clusters, grids and clouds
  - Work with data that is too large to fit on to desktop computers

For more information

Visit

www.mathworks.com/products/parallel-computing