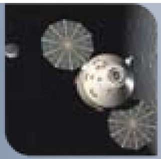
The MathWorks™

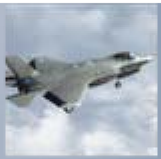
MATLAB® & SIMULINK®

Parallel Computing with MATLAB®

Elwin Chan

© 2008 The MathWorks, Inc.





MathWorks
Aerospace and Defence Conference '08

Solving Big Technical Problems

Difficulties	You could…		Solutions
Long running / Computationally intensive	Wait	⇨	Run similar *tasks* on independent processors in *parallel*
Large data set	Reduce size of problem	⇨	Load *data* onto multiple machines that work together in *parallel*

Parallel Computing

Difficulties

Jobs run in scheduled mode

Hard to debug

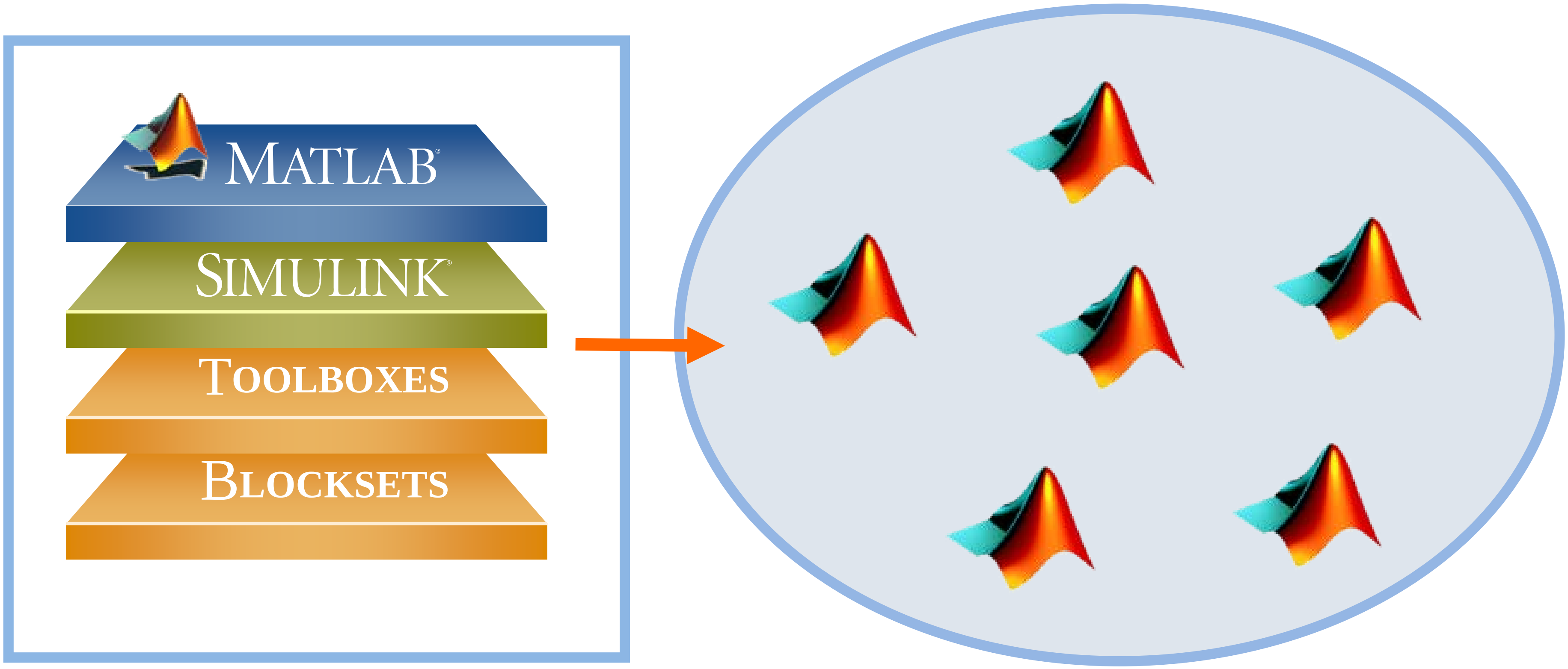
Cannot access intermediate answers

Hard to diagnose bottlenecks in algorithm

Solution

Work *interactively* in parallel

Parallel Computing with MATLAB



Pool of MATLAB Workers

Parallel Computing with MATLAB

No code changes

- Implicit Multithreaded MATLAB
- Toolbox Support:
  - Optimization Toolbox™
  - Genetic Algorithm and Direct Search Toolbox™
  - SystemTest™

Trivial changes

Task Parallel

- `parfor`
- `job` and `tasks`

Data Parallel

- `distributed`
- MATLAB and MPI

Extensive changes

Agenda

Speed up algorithms without code changes

- Develop parallel code interactively
  - Task parallel applications for faster processing
  - Data parallel applications for handling large data sets
- Schedule your programs to run
- Tips on developing parallel code

Parallel Computing with MATLAB

No code changes

- Implicit Multithreaded MATLAB
- Toolbox Support:
  - Optimization Toolbox
  - Genetic Algorithm and Direct Search Toolbox
  - SystemTest

Task Parallel

- `parfor`
- `job` and `tasks`

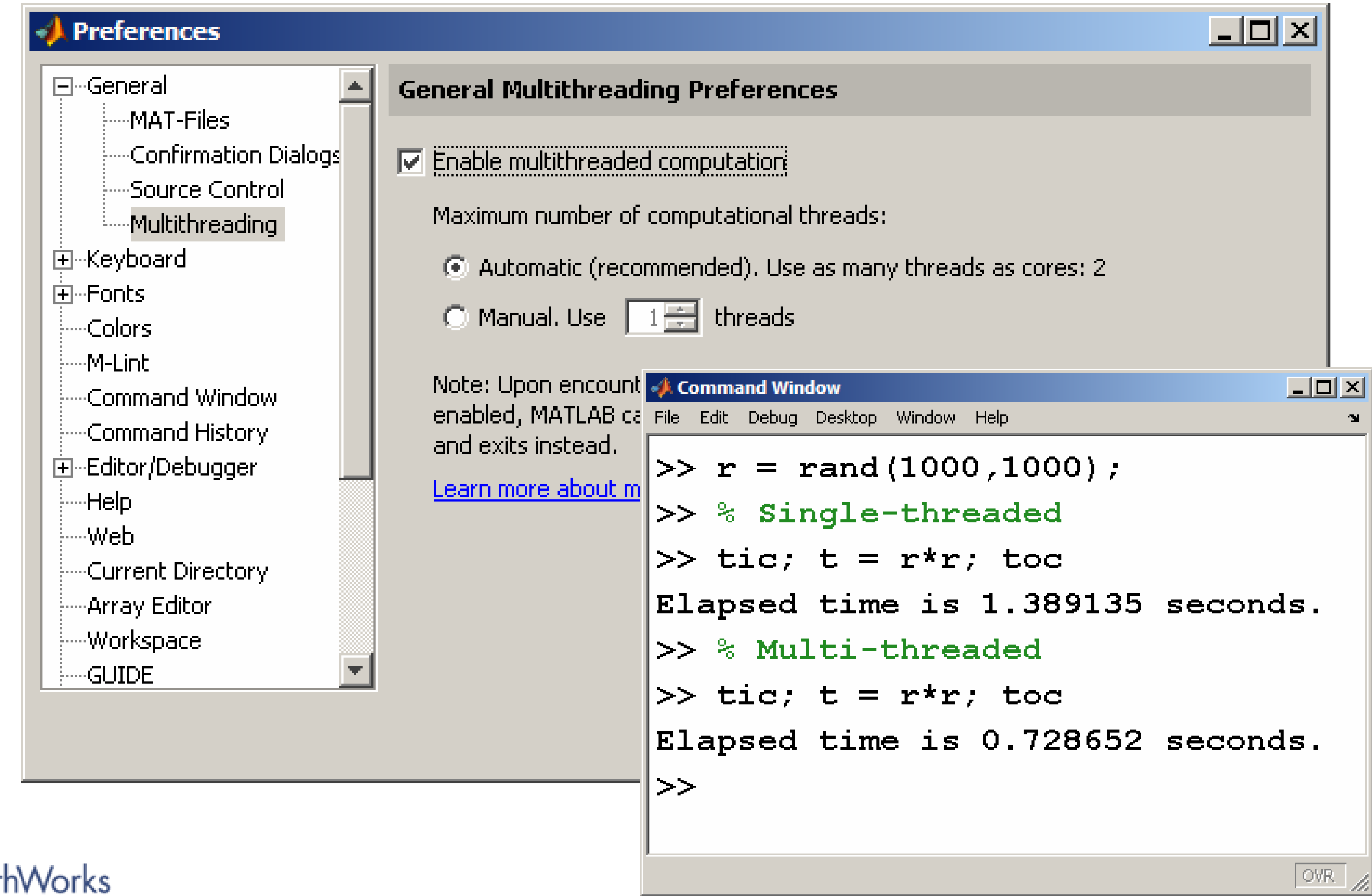
Data Parallel

- `distributed`
- MATLAB and MPI

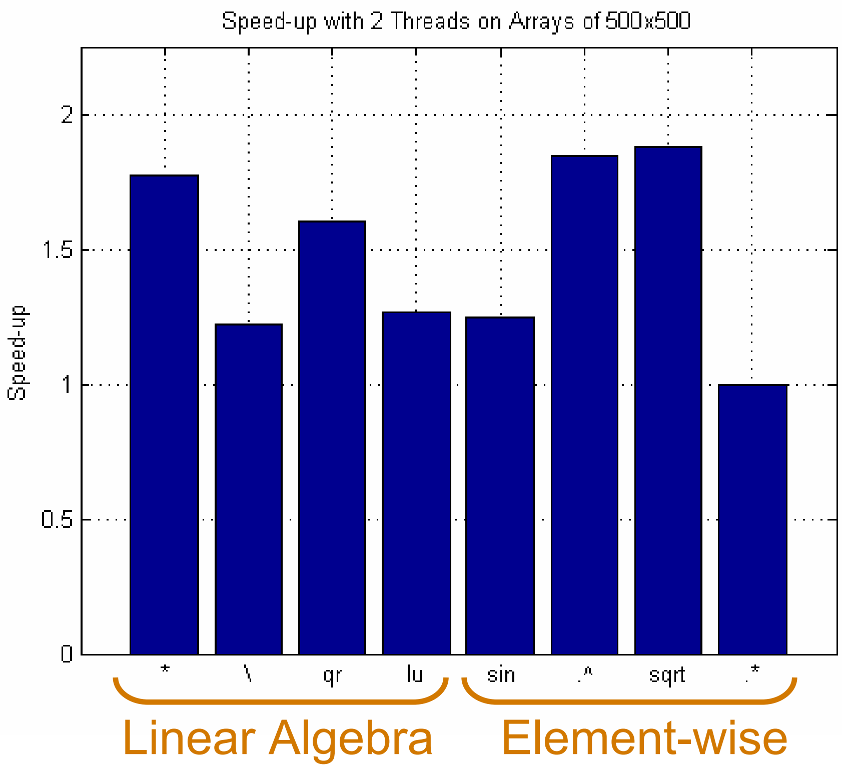
Trivial changes

Extensive changes

Demo: Speed Up Mathematical Operations



Demo: Speed Up for Implicit Multithreaded Computations



- No change required for user code
- Enables multithreading for key mathematical routines
  - Linear algebra operations
  - Element-wise operations

Implicit Multithreaded Computation

- Linear algebra operations
  - Uses multithreaded Basic Linear Algebra Subroutines (BLAS)
  - BLAS are vendor specific
  - Optimized for specific processor
- Element-wise operations
  - Just-in-time acceleration (JIT) generates on-the-fly multithreaded code

Parallel Computing with MATLAB

No code changes

- Implicit Multithreaded MATLAB
- Toolbox Support:
  - Optimization Toolbox
  - Genetic Algorithm and Direct Search Toolbox
  - SystemTest

Trivial changes

Task Parallel

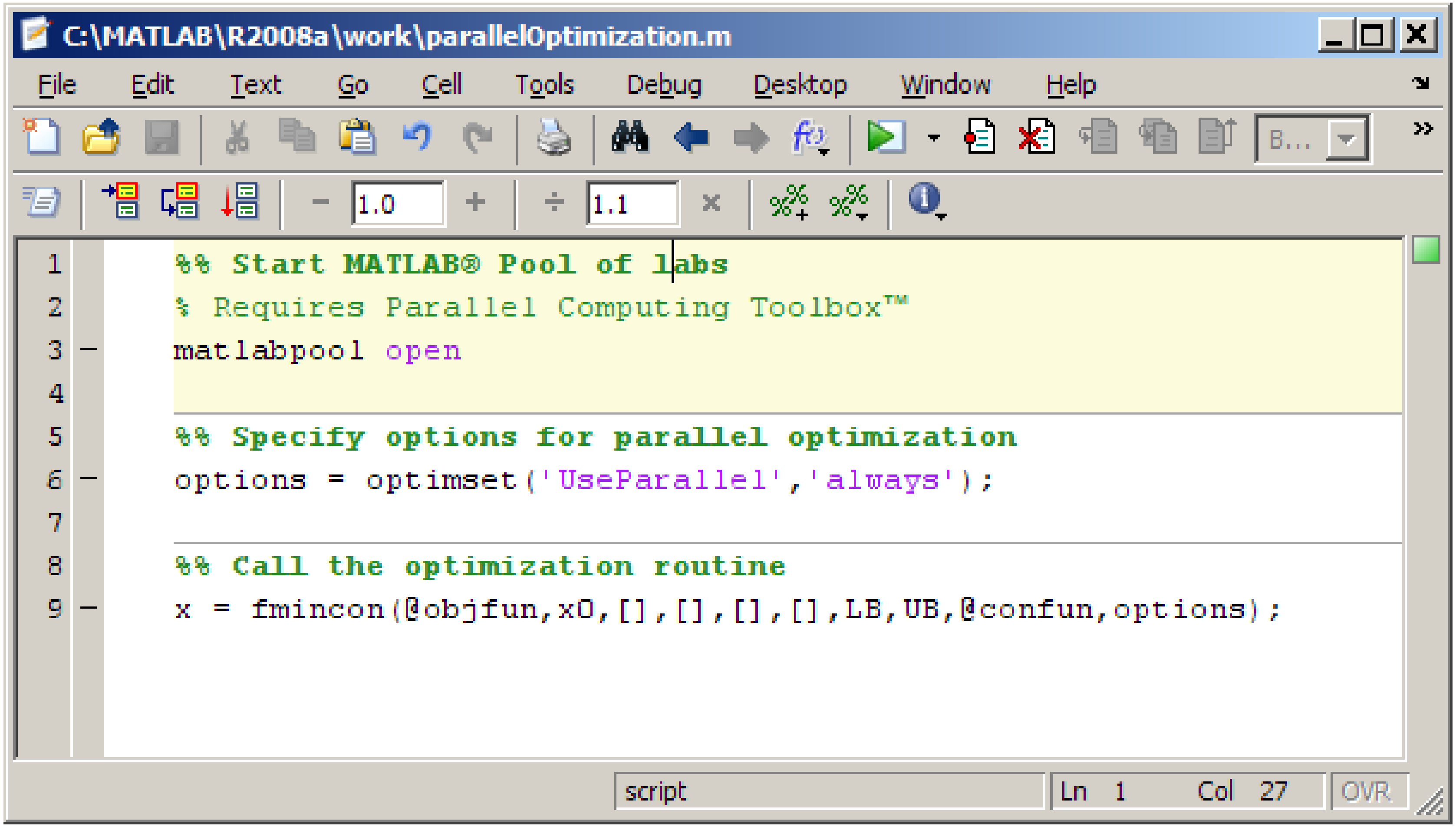
- `parfor`
- `job` and `tasks`

Data Parallel

- `distributed`
- MATLAB and MPI

Extensive changes

Demo: Support in Optimization Toolbox



Parallel Support in Optimization Toolbox

- Functions:
  - **`fmincon`**
    finds a constrained minimum of a function of several variables
  - **`fminimax`**
    finds a minimax solution of a function of several variables
  - **`fgoalattain`**
    solves the multiobjective goal attainment optimization problem

- Functions can take finite differences in parallel in order to speed the estimation of gradients

SystemTest Supports Parallel Computing for MATLAB and Simulink Applications

Distribute MATLAB and Simulink models for execution on a computer cluster or a multiprocessor system

- Run multiple simulations faster
- Use a checkbox to distribute – no additional code required
- Use homogeneous or heterogeneous platforms

Agenda

- Speed up algorithms without code changes
- Develop parallel code interactively
  - Task parallel applications for faster processing
  - Data parallel applications for handling large data sets
- Schedule your programs to run

Parallel Computing with MATLAB

No code changes

- Implicit Multithreaded MATLAB
- Toolbox Support:
  - Optimization Toolbox
  - Genetic Algorithm and Direct Search Toolbox
  - SystemTest

Task Parallel ***Data Parallel***

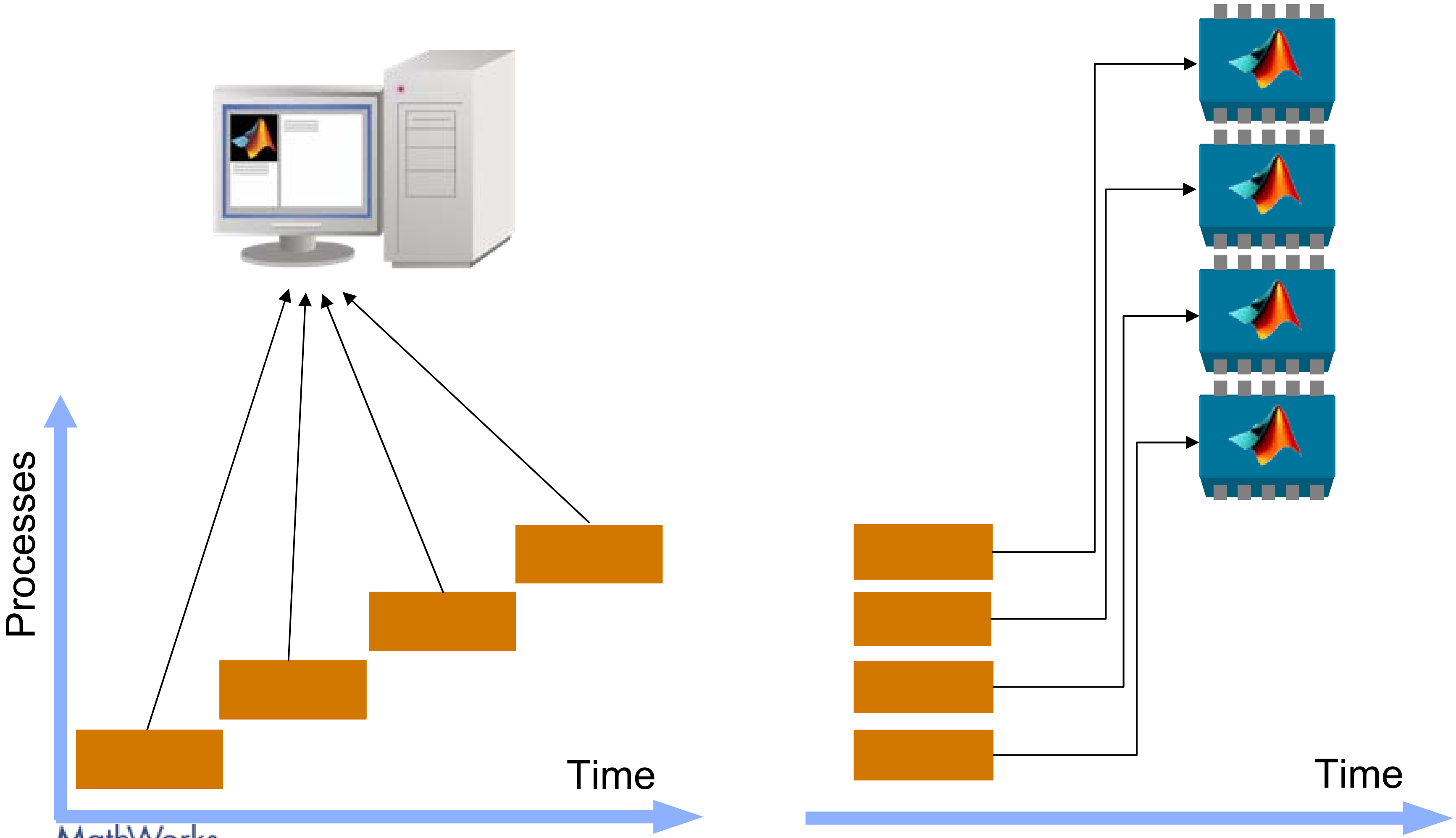
Trivial changes

- `parfor`
- `distributed`
- `job` and `tasks`

Extensive changes

- MATLAB and MPI

Distributing Tasks (Task Parallel)

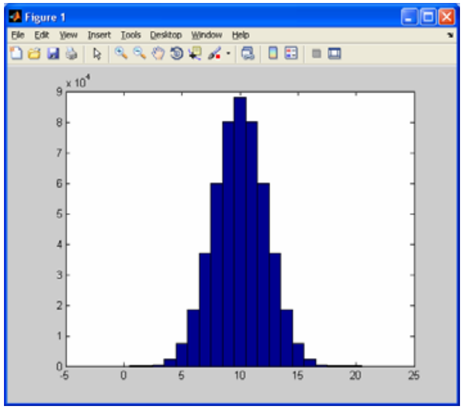


Demo: Monte Carlo Simulation of Coin Tossing

10 Simulations of Flipping 20 Coins at a Time

11	7	12	15	7	9	9	7	8	12

Number of Heads Out of 20



Demo: Monte Carlo Simulation of Coin Tossing

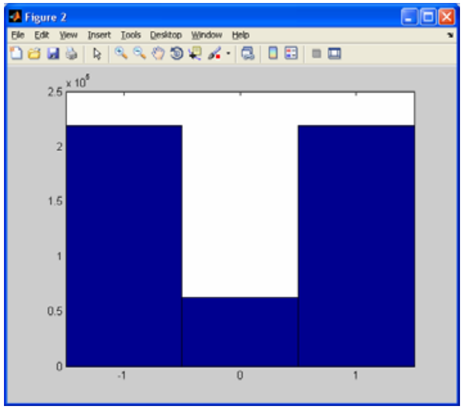
10 Simulations of Flipping 20 Coins at a Time

11	7	12	15	9	9	13	7	8	12

Number of Heads Out of 20

Change in Number of Heads

--	↓	↑	↑	↓	→	↑	↓	↑	↑



Parallel `for`-Loops

```
parfor i = 1 : n
   % do something with i
end
```

- Mix task-parallel and serial code in the same function
- Run loops on a pool of MATLAB resources
- Iterations must be order-independent
- M-Lint analysis helps in converting existing **`for`**-loops into to **`parfor`**-loops

Agenda

- Speed up algorithms without code changes
- Develop parallel code interactively
  - Task parallel applications for faster processing
  - Data parallel applications for handling large data sets
- Schedule your programs to run
- Tips on developing parallel code

Parallel Computing with MATLAB

No code changes

Trivial changes

Extensive changes

- Implicit Multithreaded MATLAB
- Toolbox Support:
  - Optimization Toolbox
  - Genetic Algorithm and Direct Search Toolbox
  - SystemTest

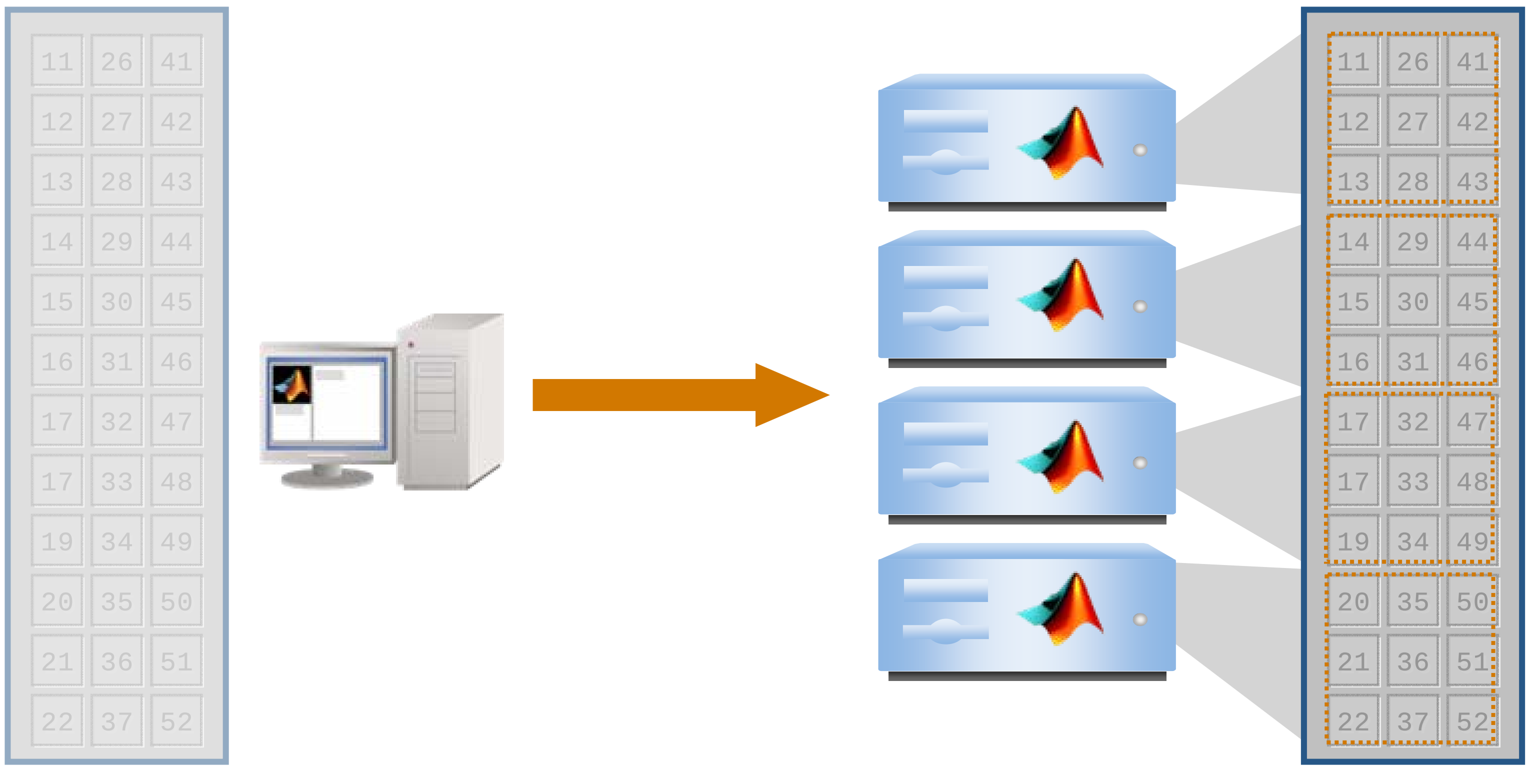
Task Parallel

- `parfor`
- `job` and `tasks`

Data Parallel

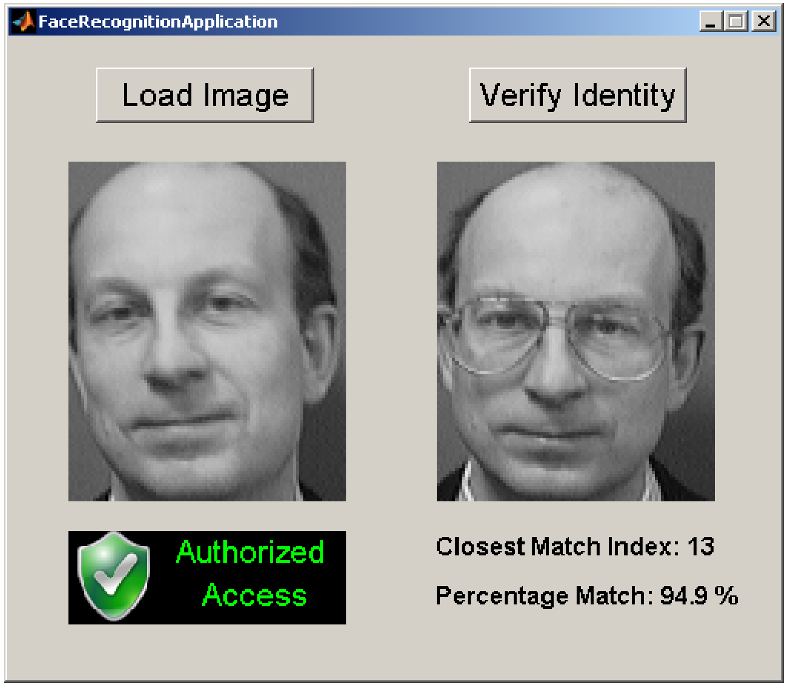
- `distributed`
- MATLAB and MPI

Large Data Sets (Data Parallel)



Demo: Interactive Face Recognition

- Do we recognize this person?
- Compare this image against a database.
- Images in database are represented using six principal eigenfaces (component images).
- Image set must be handled in one bite.



Dataset of Faces

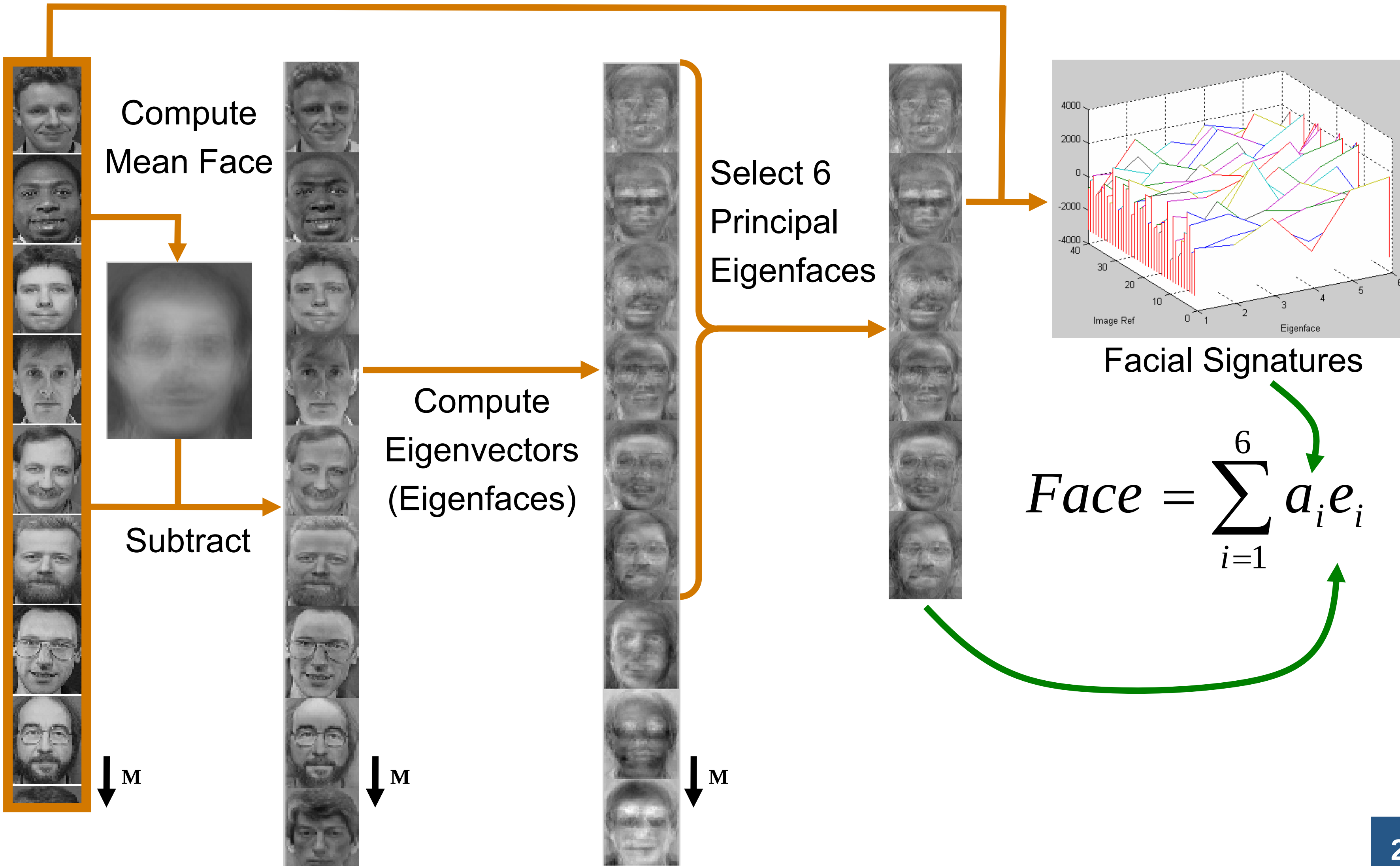
- Single snapshot used to build eigenfaces
- Data set also contains same individuals pulling different expressions – used to test recognition algorithm
- 40 individuals in 10 poses in this dataset

…etc

[Face Data provided courtesy of AT&T Laboratories Cambridge]

Face Recognition Algorithm

- Sample faces processed into eigenface components

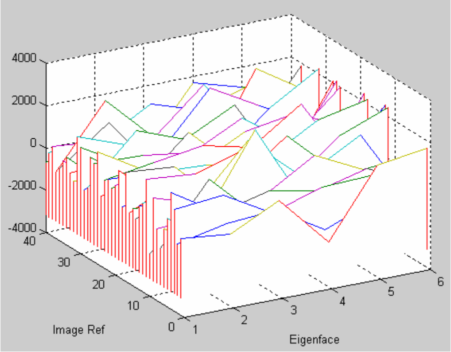


Face Recognition Algorithm

- Sample faces processed into eigenface components

Mean Face

Select 6 Principal Eigenfaces

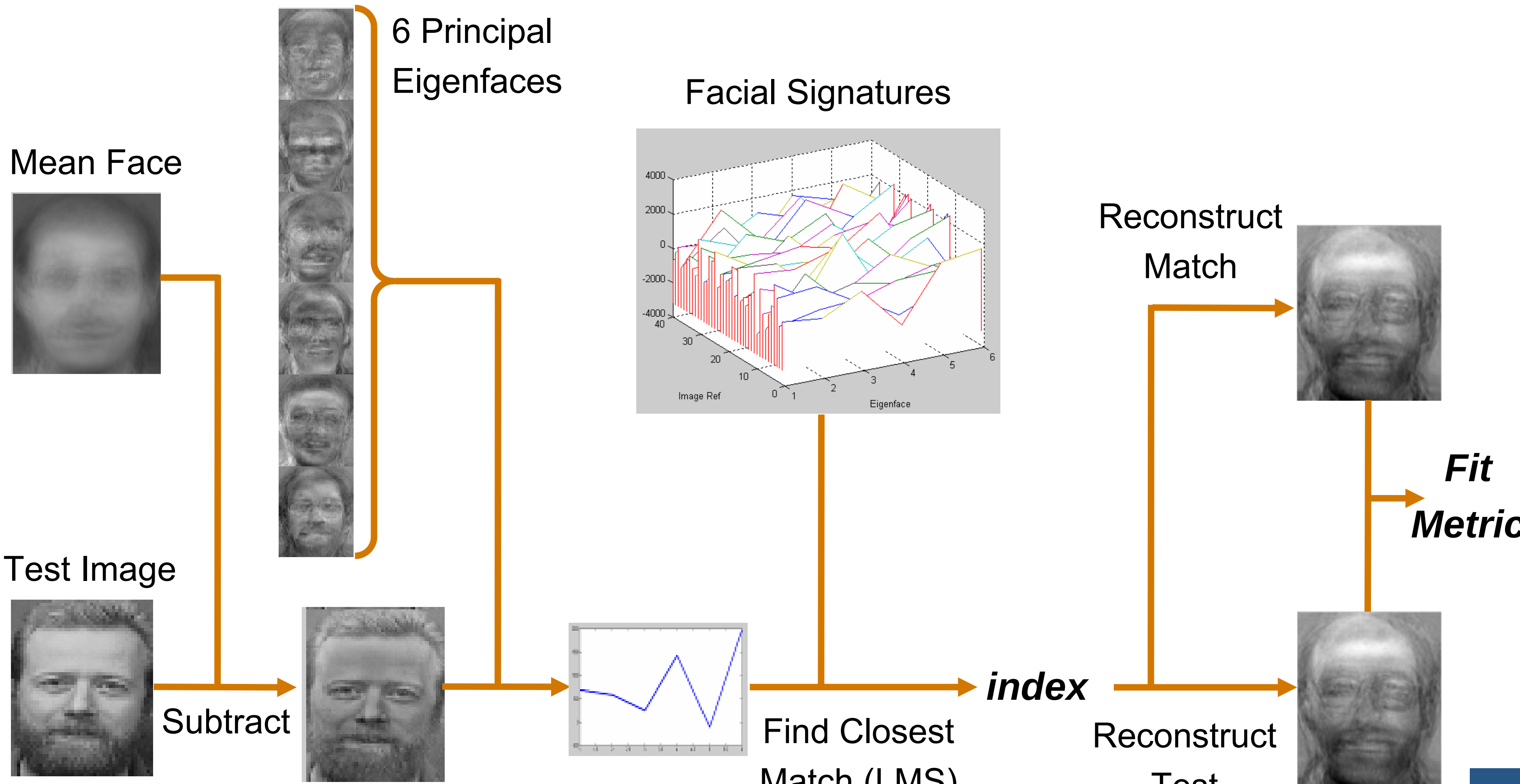


Facial Signatures

Identification requires only this Reduced Dataset!

Face Recognition Algorithm

- Test image broken into eigenface components and compared with existing database



Distributed Arrays and Parallel Algorithms

- Distributed arrays
  - Store segments of data across participating workers
  - Create from any built-in class in MATLAB
    - Examples: doubles, sparse, logicals, cell arrays, and arrays of structs
- Parallel algorithms for distributed arrays
  - Matrix manipulation operations
    - Examples: indexing, data type conversion, and transpose
  - Parallel linear algebra functions, such as **`svd`** and **`lu`**
  - Data distribution
    - Automatic, specify your own, or change at any time

MPI-Based Functions in Parallel Computing Toolbox™

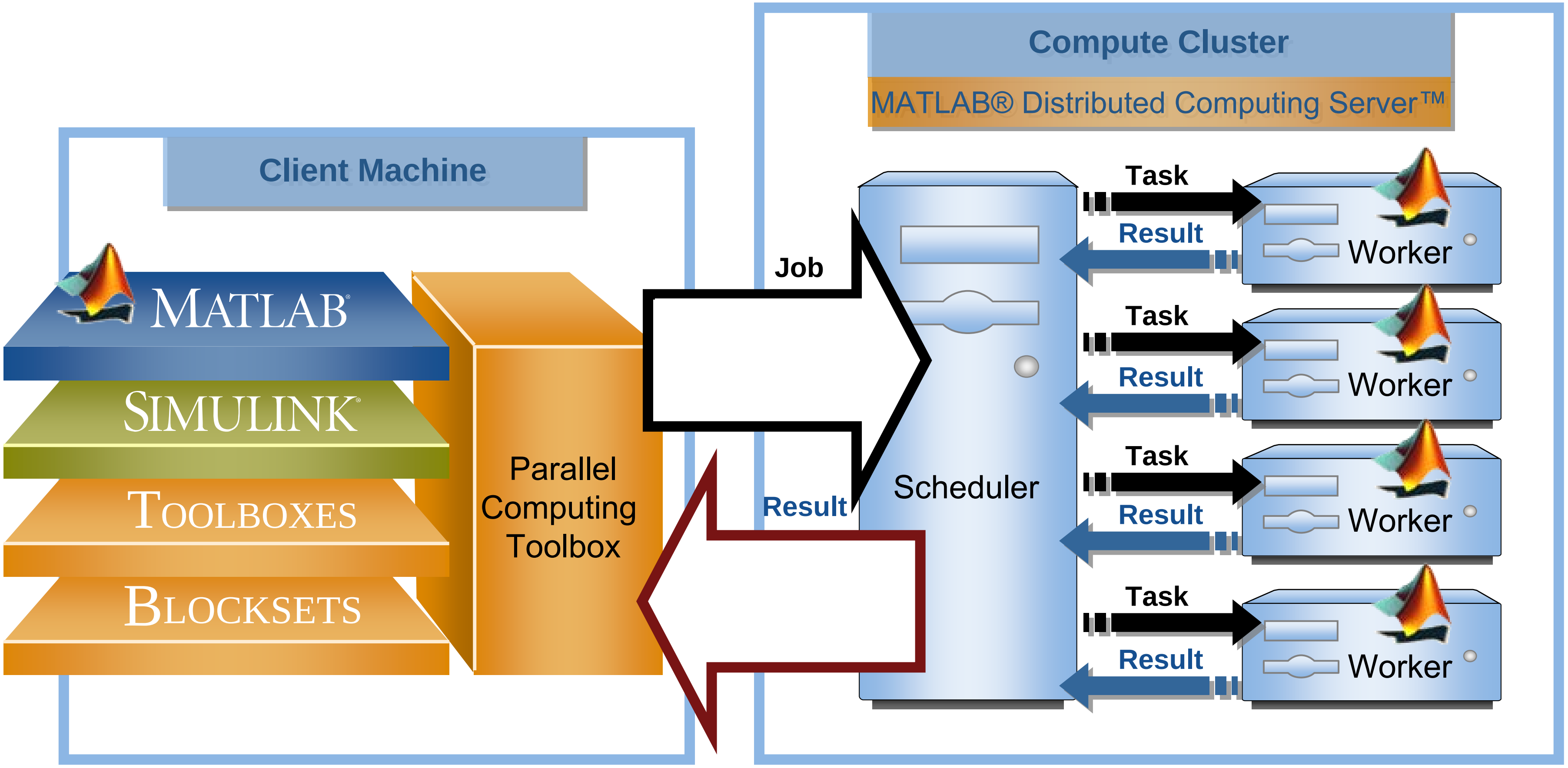
Use when a high degree of control over parallel algorithm is required

- High-level abstractions of MPI functions
  - **`labSendReceive`**, **`labBroadcast`**, and others
  - Send, receive, and broadcast any data type in MATLAB
- Automatic bookkeeping
  - Setup: communication, ranks, etc.
  - Error detection: deadlocks and miscommunications
- Pluggable
  - Use any MPI implementation that is *binary*-compatible with MPICH2

Agenda

- Speed up algorithms without code changes
- Develop parallel code interactively
  - Task parallel applications for faster processing
  - Data parallel applications for handling large data sets
- Schedule your programs to run
- Tips on developing parallel code

Distributed Applications



Demo: Scheduled Monte Carlo Coin

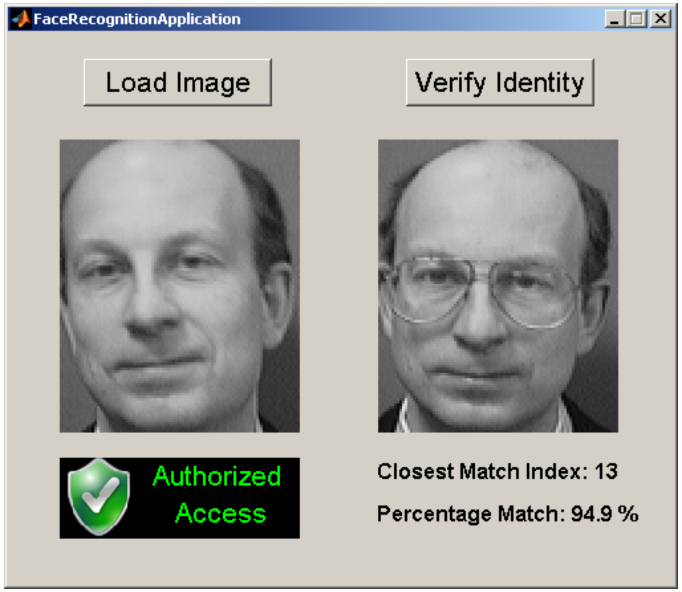
```
>> createJob(…)
>> createTask(…)
```

Demo: Scheduled Monte Carlo Coin using parfor

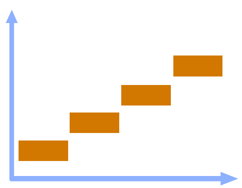
```
>> createMatlabPoolJob
```

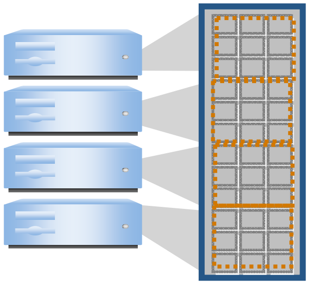
Demo: Scheduled Face Recognition

```
>> createParallelJob
```



Options for Scheduling Jobs





<table>
<tr><th>Task Parallel</th><th>Data Parallel</th></tr>
<tr><td><code>>> createMatlabPoolJob</code>
or
<code>>> batch</code></td><td rowspan="2"><code>>> createParallelJob</code></td></tr>
<tr><td><code>>> createJob(…)</code>
<code>>> createTask(…)</code></td></tr>
</table>

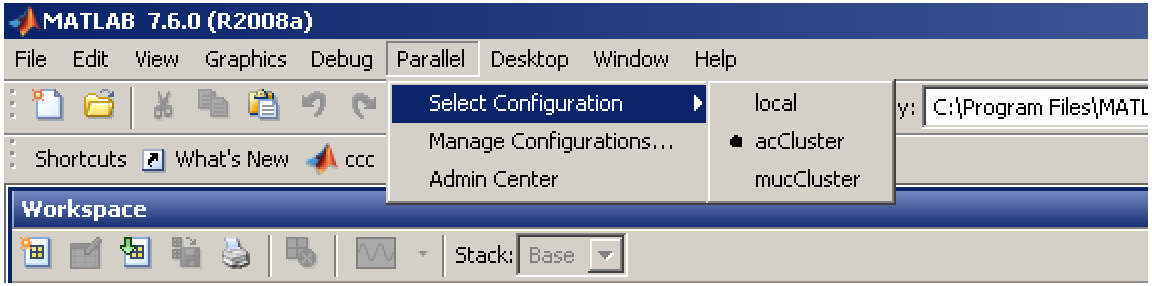
Dependencies

- `job – FileDependencies`
  - Files are copied from client to each worker machine
  - Zip compressed
  - Uncompressed and added to the MATLAB path
  - Convenient for `.m` files, but can be slow for large data files

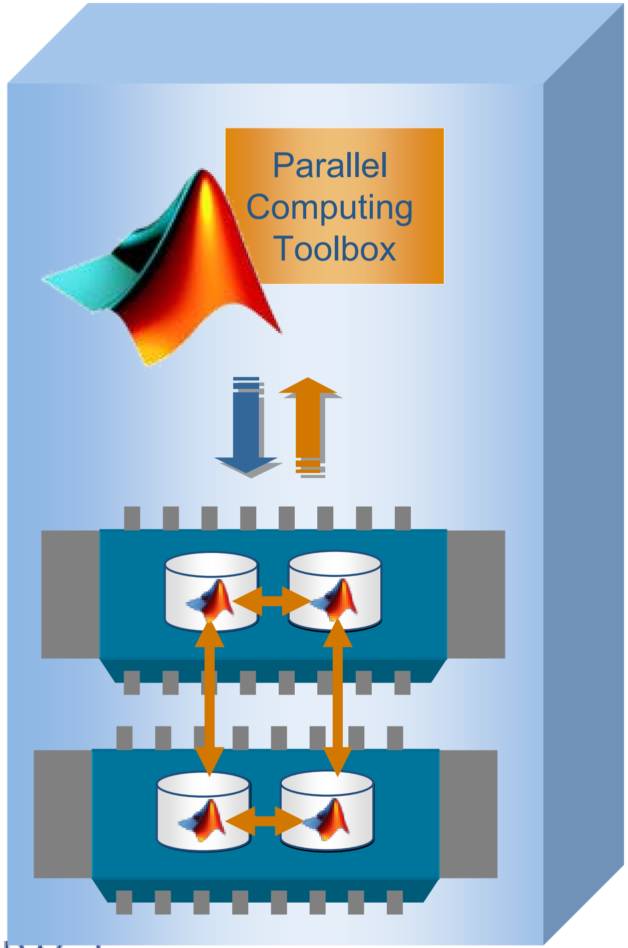
- `job – PathDependencies`
  - Shared directories are added to the MATLAB path
  - Mixing of Windows® and UNIX® paths allowed
  - Reduces the amount of data transfer from client to cluster

Configurations

- Save environment-specific parameters for your cluster
- Benefits
  - Enter cluster information only once
  - Modify configurations without changing MATLAB code
  - Apply multiple configurations when running within same session

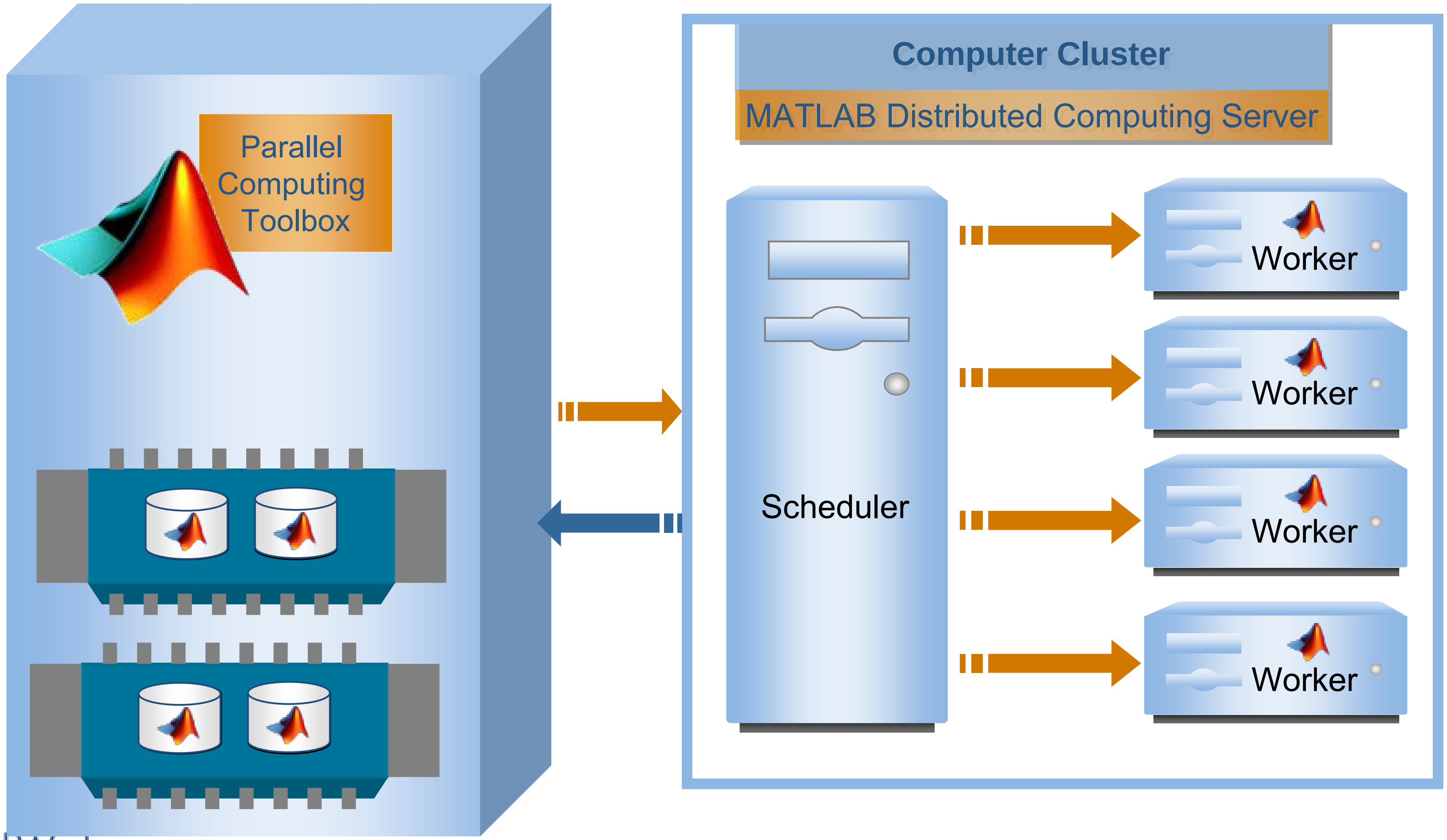


Run *Four Local* Workers with a Parallel Computing Toolbox License



- Easily experiment with explicit parallelism on multicore machines
- Rapidly develop parallel applications on local computer
- Take full advantage of desktop power
- Separate computer cluster not required

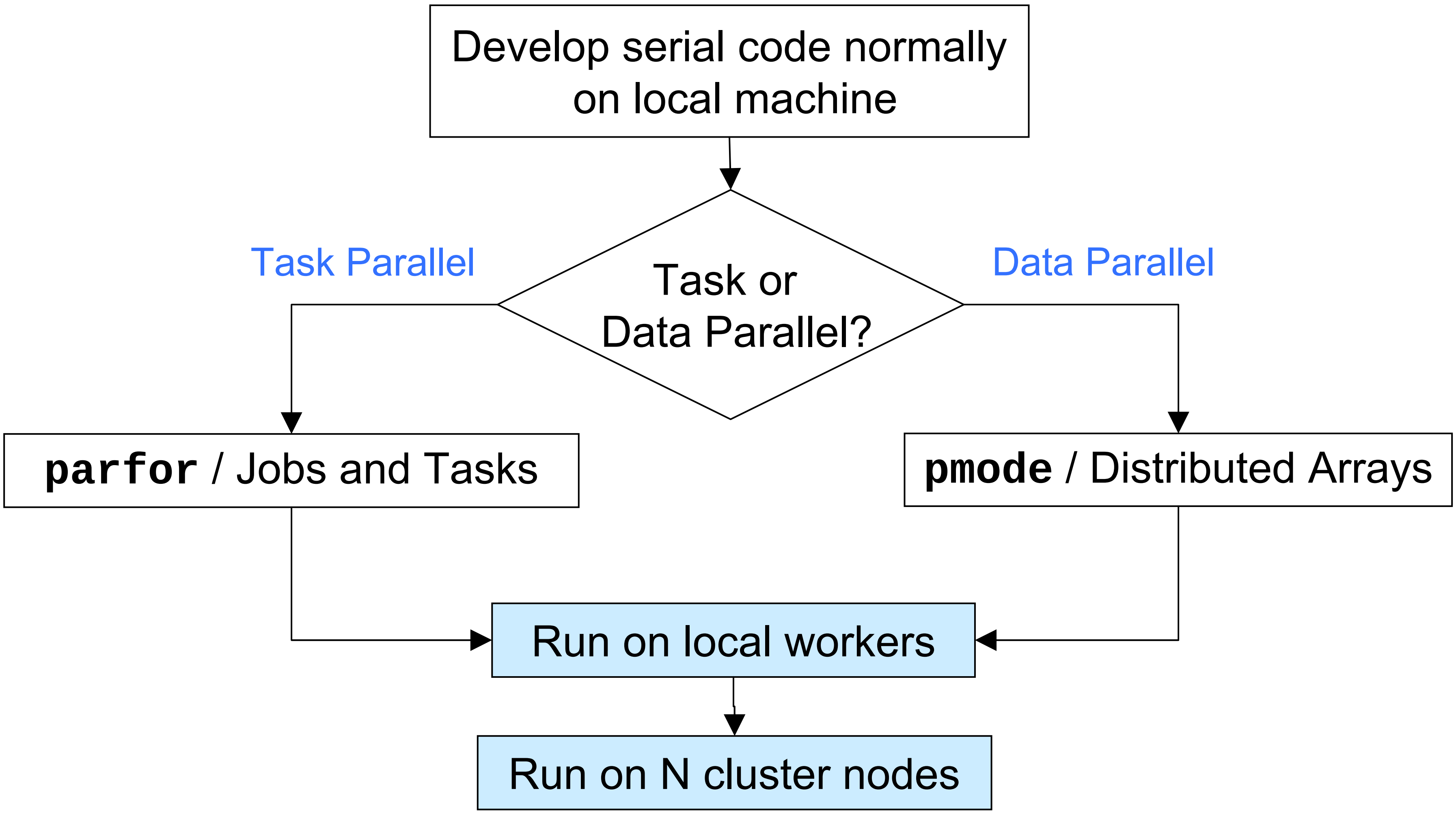
Scale Up to Cluster Configuration with No Code Changes



Agenda

- Speed up algorithms without code changes
- Develop parallel code interactively
  - Task parallel applications for faster processing
  - Data parallel applications for handling large data sets
- Schedule your programs to run
- Tips on developing parallel code

Development and Debugging Process



Parallel Profiler

- Profiles the execution time for a function
  - Similar to the MATLAB profiler
  - Includes information about the communication between labs
    - Time spent in communication
    - Amount of data passed between labs
- Benefits
  - Identify the bottlenecks in your parallel algorithm
  - Understand which operations require communication

Factors to Consider for Speeding Up Your Code

- Share code and data with workers efficiently using **`FileDependencies`** or **`PathDependencies`**
- There is always an overhead to distribution
  - Don't make a task too small
  - Combine small repetitive function calls into one larger one
- Use the M-lint and parallel profiler (**`mpiprofile`**) to identify slow code
- Minimize I/O

Summary

- Speed up algorithms without code changes
- Develop parallel code interactively
  - Task-parallel applications for faster processing
  - Data-parallel applications for handling large data sets
- Schedule your programs to run
- Tips on developing parallel code