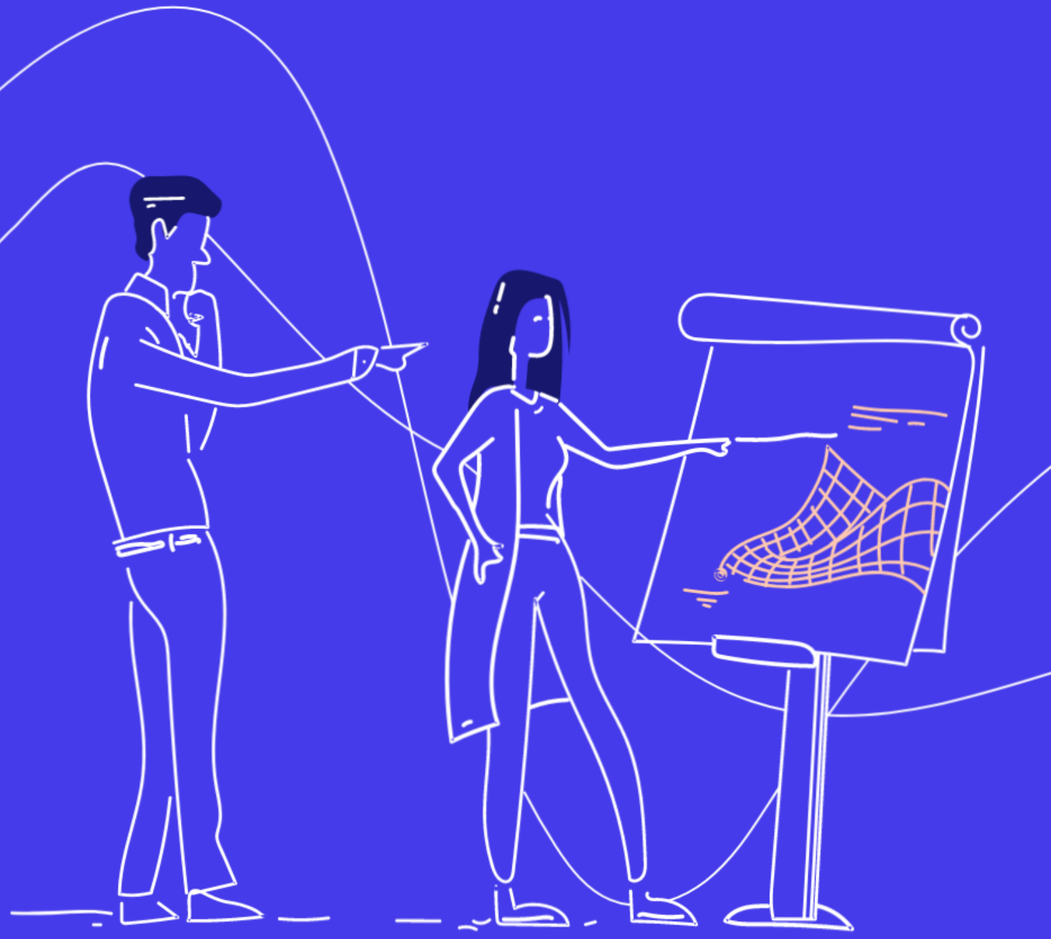
sciengineer

Accelerating and Parallelizing MATLAB Code



SciEngineer's training courses are designed to help organizations and individuals close skills gaps, keep up-to-date with the industry-accepted best practices and achieve the greatest value from MathWorks® and COMSOL® Products.

Accelerating and Parallelizing MATLAB Code

This two-day course covers a variety of techniques for making your MATLAB code run faster. You will identify and remove computational bottlenecks using techniques like pre-allocation and vectorization. In addition, you will compile MATLAB code into MEX-files using MATLAB Coder. On top of that, you will take advantage of multiple cores on your computer by parallelizing for-loops with Parallel Computing Toolbox and scale up across multiple computers using MATLAB Parallel Server.

Prerequisites

MATLAB Fundamentals, or equivalent experience using MATLAB

TOPICS

Day 1

- Improving Performance
- Generating MEX-Files
- Parallelizing Computations

Day 2

- Parallel for-Loops
- Offloading Execution
- Working with Clusters
- GPU Computing

Improving Performance

OBJECTIVE: Analyze code performance and utilize techniques for acceleration within MATLAB.

- Identifying bottle necks
- Pre-allocating arrays
- Vectorizing operations in various ways
- Rewriting algorithms

Generating MEX-Files

OBJECTIVE: Generate compiled code files from MATLAB code for better performance.

- MATLAB Coder overview and workflow
- Generating and verifying MEX-files
- Calling unsupported functions
- Adjusting settings for MEX-file generation

Parallelizing Computations

OBJECTIVE: Parallelize code execution to take advantage of multiple cores.

- Opening additional MATLAB processes
- Running parallel for-loops
- Measuring speedup
- Processing multiple files in parallel

Parallel for-Loops

OBJECTIVE: Explore parallel for-loops in more detail and apply techniques for converting for-loops to parfor-loops.

- Requirements of parallel for-loops
- Parallelizing for-loops
- Retrieving intermediate results

Offloading Execution

OBJECTIVE: Offload computations to another MATLAB process in order to be able to use MATLAB for other tasks in the meantime. This is also a preparation step for working with clusters.

- Processing in batch
- Creating batch jobs
- Retrieving results
- Using the Job Monitor

Working with Clusters

OBJECTIVE: Accelerate computations and realize more extensive simulations by utilizing multiple computers.

- Local and remote clusters
- Dynamic licensing
- Cluster discovery and connection
- File access considerations

GPU Computing

OBJECTIVE: Execute MATLAB code on your computer's graphics card (GPU) as another option for speeding up calculations.

- Overview of GPU architecture and processing
- Applications suitable for GPU processing
- Invoking MATLAB functions on the GPU
- Using pre-existing CUDA code

sciengineer

Expand your knowledge