# DESCRIPTION & CONTENTS



* **Basic course: must be passed in order to attend any of other available courses.**

## CONTENTS of All Courses:

### 1. Introduction to MATLAB

**Course Description:**

MATLAB is a high-level technical computing language and interactive environment for algorithm development, data visualization, data analysis, and numeric computation. Using MATLAB assists in solving technical computing problems faster than with traditional programming languages, such as C, C++, and FORTRAN. Introduction to MATLAB is primary course guides student to basic capabilities of the software.

### 2. Programming in MATLAB

**Course Description:**

Programming in MATLAB is a rather advanced course that guides the student to the programming features of the software. This course introduces the fundamental ideas for programming in Matlab. It requires no programming experience, but the student should be familiar with Matlab basics. All background needed can be found on the course contents.

The programming structures presented applies to Matlab. However, these structures look very similar in other computer languages, such as C, Java, Pascal, etc., so by understanding how loops, logical operations, etc., work in Matlab, you will be well-prepared for beginning programming in other languages as well.

### 3. Advanced Programming in MATLAB

**Course Description:**

This is an advanced programming course of MATLAB software programming with extra variable classes and more mathematical operations. More advanced programs and subroutines are also included. When writing larger programs, it is necessary split the problem up into functions that can be passed input values and return the necessary output values. This is what we cover in this tutorial.

Note that, it is important to implement the examples of this tutorial, run them, and carefully compare the output with the code.

### 4. MATLAB GUI (Graphical User Interface)

**Course Description:**

A graphical user interface (GUI) is a graphical display in one or more windows containing controls, called components, that enable a user to perform interactive tasks. The user of the GUI does not have to create a script or type commands at the command line to accomplish the tasks. Unlike coding programs to accomplish tasks, the user of a GUI need not understand the details of how the tasks are performed. GUI components can include menus, toolbars, push buttons, radio buttons, list boxes, and sliders—just to name a few.

GUIs created using MATLAB® tools can also perform any type of computation, read and write data files, communicate with other GUIs, and display data as tables or as plots.

### 5. Advanced Graphics and Animations

**Course Description:**

The MATLAB environment offers a variety of data plotting functions plus a set of GUI tools to create, and modify graphic displays. The GUI tools afford most of the control over graphic properties and options that typed commands such as annotate, get, and set provide.

A figure is a MATLAB window that contains graphic displays (usually data plots) and UI components. You create figures explicitly with the figure function, and implicitly whenever you plot graphics and no figure is active. By default, figure windows are resizable and include pull-down menus and toolbars.

A plot is any graphic display you can create within a figure window. Plots can display tabular data, geometric objects, surface and image objects, and annotations such as titles, legends, and colorbars. Figures can contain any number of plots. Each plot is created within a 2-D or a 3-D data space called an axes. You can explicitly create axes with the axes or subplot functions.

A graph is a plot of data within a 2-D or 3-D axes. Most plots made with MATLAB functions and GUIs are therefore graphs. When you graph a one-dimensional variable (e.g., rand(100,1)), the indices of the data vector (in this case 1:100) become assigned as x values, and plots the data vector as y values. Some types of graphs can display more than one variable at a time, others cannot.

**6. Numerical Analysis**

**Course Description:**

This course introduces applied numerical methods for engineering and science students in sophomore to senior levels; it targets the students of today who do not like or do not have time to derive and prove mathematical results. It can also serve as a reference to MATLAB applications for professional engineers and scientists, since many of the MATLAB codes presented after introducing each algorithm's basic ideas can easily be modified to solve similar problems even by those who do not know what is going on inside the MATLAB routines and the algorithms they use.

**7. Statistics (Level 1)**

**Course Description:**

Statistics Toolbox software extends MATLAB® to support a wide range of common statistical tasks. The toolbox contains two categories of tools:

- Building-block statistical functions for use in MATLAB programming.
- Graphical user interfaces (GUIs) for interactive data analysis.

Code for the building-block functions is open and extensible. Use the MATLAB Editor to review, copy, and edit code for any function. Extend the toolbox by copying code to new files or by writing files that call toolbox functions.

Toolbox GUIs allow you to perform statistical visualization and analysis without writing code. You interact with the GUIs using sliders, input fields, push buttons, etc. and the GUIs automatically call building-block functions.

**8. Statistics (Level 2)**

**Course Description:**

This level introduces more advanced features of the statistics toolbox and guides the user through accomplishing advanced programming statistical tasks.

**9. Modern Optimization**

**Course Description:**

Optimization Toolbox extends the capability of the MATLAB® numeric computing environment. The toolbox includes routines for many types of optimization including:

- Unconstrained nonlinear minimization.
- Constrained nonlinear minimization, including goal attainment problems, minimax. problems, and semi-infinite minimization problems.
- Quadratic and linear programming.
- Nonlinear least-squares and curve fitting.
- Nonlinear system of equation solving.
- Constrained linear least squares.
- Sparse and structured large-scale problems.

**10. Simulink**

**Course Description:**

Simulink® software models, simulates, and analyzes dynamic systems. It enables us to pose a question about a system, model the system, and see what happens. With Simulink, we can easily build models from scratch, or modify existing models to meet your needs.

Simulink supports linear and nonlinear systems, modeled in continuous time, sampled time, or a hybrid of the two. Systems can also be multirate — having different parts that are sampled or updated at different rates.

Thousands of scientists and engineers around the world use Simulink to model and solve real problems in a variety of industries, including:

- Aerospace and Defense
- Automotive
- Communications
- Electronics and Signal Processing
- Medical Instrumentation

**11. Stateflow**

**Course Description:**

Stateflow® extends Simulink® with a design environment for developing state charts and flow diagrams. Stateflow software provides the language elements required to describe complex logic in a natural, readable, and understandable form. It is tightly integrated with MATLAB® and Simulink products, providing an efficient environment for designing embedded systems that contain control, supervisory, and mode logic.

**12. Control System (Level 1)**

**Course Description:**

Control System Toolbox provides industry-standard algorithms and tools for systematically analyzing, designing, and tuning linear control systems. You can specify your system as a transfer function, state-space, pole-zero-gain, or frequency-response model. Interactive tools and command-line functions, such as step response plot and Bode plot, let you visualize system behavior in time domain and frequency domain.

You can tune compensator parameters using automatic PID controller tuning, Bode loop shaping, root locus method, LQR/LQG design, and other interactive and automated techniques. You can validate your design by verifying rise time, overshoot, settling time, gain and phase margins, and other requirements.

**13. Control System (Level 2)**

**Course Description:**

This level introduces more advanced features of the control system toolbox and guides the user through accomplishing advanced control applications.

**14. System Identification**

**Course Description:**

System Identification Toolbox software lets you construct mathematical models of dynamic systems from measured input-output data. This data-driven approach helps you describe systems that are not easily modeled from first principles or specifications, such as chemical processes and engine dynamics. It also helps you simplify detailed first-principle models, such as finite-element models of structures and flight dynamics models, by fitting simpler models to their simulated responses.

**15. <u>Vibration Analysis</u>**

**Course Description:**

Solving Engineering Vibration Analysis Problems using MATLAB course is designed as an introductory undergraduate or graduate course for engineering students of all disciplines. Vibration analysis is a multidisciplinary subject and presents a system dynamics methodology based on mathematical fundamentals and stresses physical system modeling. The classical methods of vibration analysis engineering are covered: matrix analysis, Laplace transforms and transfer functions.

This course has two main purposes. The first purpose is to teach the student the various methods of constructing and representing dynamic mechanical models. The second purpose is to help the student develop a strong understanding of the modal analysis technique, where the total response of a system can be constructed by combinations of individual modes of vibration.

**16. <u>Image and Video Processing (Level 1)</u>**

**Course Description:**

The Image Processing Toolbox software is a collection of functions that extend the capability of the MATLAB numeric computing environment. The toolbox supports a wide range of image processing operations, including

- Spatial image transformations
- Morphological operations
- Neighborhood and block operations
- Linear filtering and filter design
- Transforms
- Image analysis and enhancement
- Image registration
- Deblurring
- Region of interest operations

**17. <u>Image and Video Processing (Level 2)</u>**

**Course Description:**

This level introduces more advanced features of the image processing toolbox and guides the user through accomplishing advanced image processing applications.

**18. <u>Fuzzy Logic</u>**

**Course Description:**

Fuzzy Logic Toolbox provides MATLAB® functions, graphical tools, and a Simulink® block for analyzing, designing, and simulating systems based on fuzzy logic. The product guides you through the steps of designing fuzzy inference systems. Functions are provided for many common methods, including fuzzy clustering and adaptive neurofuzzy learning.

The toolbox lets you model complex system behaviors using simple logic rules and then implement these rules in a fuzzy inference system. You can use it as a standalone fuzzy inference engine. Alternatively, you can use fuzzy inference blocks in Simulink and simulate the fuzzy systems within a comprehensive model of the entire dynamic system.

**19. <u>Artificial Neural Networks</u>**

**Course Description:**

Neural Network Toolbox provides tools for designing, implementing, visualizing, and simulating neural networks. Neural networks are used for applications where formal analysis would be difficult or impossible, such as pattern recognition and nonlinear system identification and control. Neural Network Toolbox supports feedforward networks, radial basis networks, dynamic networks, self-organizing maps, and other proven network paradigms.

### 20. <u>Genetic Algorithms</u>

**Course Description:**

The Genetic and Evolutionary Algorithm Toolbox provides global optimization capabilities in Matlab to solve problems not suitable for traditional optimization approaches. Are you looking for a sophisticated way of solving your problem in case it has no derivatives, is discontinuous, stochastic, non-linear or has multiple minima or maxima? The GEATbx should be your method of choice!

Powerful genetic and evolutionary algorithms find solutions to your problems - and it's easy to use! Numerous ready to run examples and demonstrations give you a head start in setting up your problem, selecting the appropriate optimization algorithm and monitoring the state and progress of the optimization. This enables beginners and advanced users to achieve results fast.

In order to solve large and complex problems, the GEATbx contains extensions that are needed especially for the optimization of real world problems.

- visualization of the state and progress of your optimization,
- multi-objective optimization,
- constraint handling,
- problem-specific initialization and visualization.
- multi-strategy and multi-population support.

### 21. <u>Data Acquisition using MATLAB</u>

**Course Description:**

Data Acquisition Toolbox software provides a complete set of tools for analog input, analog output, and digital I/O from a variety of PC-compatible data acquisition hardware. The toolbox lets you configure your external hardware devices, read data into MATLAB® and Simulink® environments for immediate analysis, and send out data.

Data Acquisition Toolbox enables you to customize your acquisitions, access the built-in features of hardware devices, and incorporate the analysis and visualization features of MATLAB and related toolboxes into your design. You can analyze or visualize your data, save it for post-processing, and make iterative updates to your test setup based on your analysis results. Data Acquisition Toolbox allows you to use MATLAB as a single, integrated environment to support the entire data acquisition, data analysis, and application development process.

Data Acquisition Toolbox also supports Simulink with blocks that enable you to incorporate live data or hardware configuration directly into Simulink models. You can then verify and validate your model against live, measured data as part of the system development process.