

MATLAB: FROM FUNDAMENTALS TO ARTIFICIAL INTELLIGENCE

FILIPPE AZEVEDO

AUTHOR

Filipe Azevedo

TITLE

MATLAB: FROM FUNDAMENTALS TO ARTIFICIAL INTELLIGENCE

PUBLISHER

Quântica Editora – Conteúdos Especializados, Lda.
Tel. +351 220 939 053 · Email: geral@quanticaeditora.pt · www.quanticaeditora.pt
Praça da Corujeira no. 38 · 4300-144 PORTO · Portugal

IMPRINT

Engebook – Conteúdos de Engenharia

DISTRIBUTION

Booki – Conteúdos Especializados
Tel. +351 220 104 872 · Email: info@booki.pt · www.booki.pt

COVER DESIGN

Luciano Carvalho
Delineatura, Design de Comunicação · www.delineatura.pt

PRINTING

July 2025



Illegal copying violates the rights of authors.
We are all harmed by it.

Copyright © 2025 | All rights reserved to Quântica Editora – Conteúdos Especializados, Lda.

Reproduction of this work, in whole or in part, by photocopy or any other means, whether electronic, mechanical, or otherwise, without prior written authorization from the Publisher and the Author, is illegal and may result in legal action against the offender.

This book complies with the new Orthographic Agreement of 1990, following its general guidelines and assuming some specific options.

UDC (Universal Decimal Classification)

004.4	Software. Equipment, programs
004.8	Artificial intelligence
519.6	Computational mathematics. Numerical analysis and mathematical programming.
	Mathematics for engineering
004.45	System software
62	Engineering. Technology in general

ISBN

E-book: 9789899177871

Publication cataloging

Family: Engineering
Subfamily: Programming Languages

Contents

1	Introduction	1
1.1	MATLAB Windows	1
1.2	Variables	2
1.3	Basic Arithmetic Operations in MATLAB	3
1.4	Rounding	4
1.5	Logical Operators	5
1.6	Relational Operators	6
1.7	Trigonometric Functions	6
2	Flow Control	9
2.1	Introduction	9
2.2	<i>IF</i> conditional statement	10
2.2.1	Conditional branching <i>IF...ELSE</i>	10
2.2.2	Nested conditional branching <i>IF...ELSEIF...ELSE</i>	11
2.3	Repetitive statement <i>FOR</i>	11
2.4	Repetitive statement <i>WHILE</i>	12
2.5	Case selection <i>SWITCH</i>	12
2.6	Statement <i>TRY-CATCH</i>	13
2.7	Exercises	14
3	Matrices and Structures	19
3.1	Introduction	19
3.2	Data Types	19
3.3	Creating multidimensional matrices	22
3.4	Size and dimension of matrices	23
3.5	Pre-allocation of matrices	24
3.6	Concatenation of matrices	25

3.7	Data access	26
3.8	Modifying the shape of a matrix	27
3.9	Indexing multidimensional matrices	28
3.10	Manipulation of real or complex number matrices	29
3.11	Exercises	32
4	Plots and Graphs	35
4.1	Introduction	35
4.2	Components of a Graph	35
4.3	2-D Graphs	36
4.3.1	Saving Graphs	39
4.4	3-D Graphs	41
4.4.1	3-D Line Graphs	41
4.5	Surface Graphs	42
4.6	Function <code>subplot</code>	49
4.7	Exercises	50
5	M-Files	53
5.1	<i>Scripts</i>	53
5.1.1	Simple example of a <i>script</i>	54
5.1.2	Executing <i>scripts</i>	55
5.2	Functions	55
5.2.1	Simple example of a Function	57
5.2.2	Executing Functions	58
5.3	Exercises	58
6	Optimization	61
6.1	Introduction	61
6.2	Formulation and Optimization Process	62
6.2.1	General Formulation	62
6.2.2	Steps in the Optimization Process	63
6.3	Classification of Optimization Problems	63
6.3.1	Linear Programming (LP)	63
6.3.2	Quadratic Programming (QP)	63
6.3.3	Nonlinear Optimization (NLO)	63
6.3.4	Integer and Mixed-Integer Optimization	64
6.3.5	Global Optimization	64
6.4	Optimization Tools in MATLAB	64

6.4.1	Native Functions	64
6.4.2	Optimization Toolbox	64
6.4.3	Global Optimization Toolbox	65
6.4.4	Graphical Interface	65
6.5	Practical Examples in MATLAB	65
6.5.1	Linear Programming	65
6.5.2	Nonlinear Optimization	65
6.5.3	Global Optimization with GA	66
6.5.4	Global Optimization with PSO	66
6.6	Comprehensive Practical Example	66
6.6.1	Formulation	66
6.6.2	Solution with <code>intlinprog</code>	66
6.6.3	Solution with GA	67
6.6.4	Solution with PSO	67
6.7	Multiobjective and Advanced Optimization	68
6.7.1	Multiobjective Optimization	68
6.7.2	Optimization Under Uncertainty	68
6.7.3	Hybrid Methodologies and Parallelization	68
6.8	Proposed Exercises	68
7	Neural Networks	71
7.1	Neural Networks in MATLAB	71
7.2	Theoretical Introduction	71
7.2.1	Motivation and History	71
7.2.2	The Artificial Neuron	71
7.2.3	Network Architectures	72
7.3	Data Preparation and Preprocessing	73
7.4	Types of Neural Networks in MATLAB	73
7.4.1	Comparative Table	73
7.5	Mathematical Operation and Training Algorithms	73
7.5.1	Cost Function	73
7.5.2	Backpropagation	74
7.5.3	Optimality Conditions	74
7.6	Practical Example: Classification with Feedforward Network	74
7.6.1	General Steps	74
7.6.2	MATLAB Code	75
7.7	Training, Validation, and Regularization	75

7.7.1	Avoiding Overfitting	75
7.7.2	Cross-Validation	75
7.8	Recurrent Networks and Time Series	75
7.8.1	NARX Example	75
7.9	Visualization and Interpretation of Results	76
7.10	Frequently Asked Questions and Tips	76
7.11	Advanced Examples and Case Studies	76
7.11.1	Transfer Learning with CNN	76
7.11.2	Clustering with Kohonen	77
7.12	Conclusion	77
7.13	Proposed Exercises	77
8	Deep Learning: Advanced Techniques and Applications	79
8.1	Introduction to Advanced Deep Learning in MATLAB	79
8.2	Evolution and State of the Art of Deep Learning	79
8.2.1	From Perceptron to Deep Networks	79
8.2.2	Current Trends	80
8.3	Deep and Modern Architectures	80
8.3.1	Advanced Convolutional Networks (CNNs)	80
8.3.2	Recurrent Networks, LSTM and Transformers	80
8.3.3	Generative Adversarial Networks (GANs)	81
8.3.4	Siamese Networks and Triplet Networks	81
8.4	Advanced Training and Regularization Techniques	81
8.4.1	Advanced Data Augmentation	81
8.4.2	Regularization and Optimization	81
8.4.3	Hyperparameter Optimization	81
8.4.4	Distributed Training and Multi-GPU	81
8.5	Transfer Learning and Fine-Tuning	82
8.5.1	Transfer Learning	82
8.5.2	Fine-Tuning and Layer Freezing	82
8.6	Deployment, Optimization and Edge AI	82
8.6.1	Model Optimization for Production	82
8.6.2	Deployment	83
8.7	Explainability, Ethics and Security in Deep Learning	83
8.7.1	Model Explainability	83
8.7.2	Ethics and Bias	83
8.7.3	Security and Robustness	83

8.8	Integration with Other Languages and Frameworks	84
8.8.1	Python, TensorFlow, PyTorch, ONNX	84
8.8.2	MLOps Pipelines	84
8.9	Current Challenges and Future of Deep Learning	84
8.10	Advanced Case Studies	84
8.10.1	Anomaly Detection in Industrial Signals with Autoen- coders	84
8.10.2	Synthetic Medical Image Generation with GANs	85
8.10.3	Energy Time Series Prediction with LSTM and Trans- formers	85
8.11	Proposed Exercises	85
9	<i>Reinforcement Learning: Concepts, Methods, and Applica- tions</i>	87
9.1	Introduction	87
9.2	Fundamental Concepts	87
9.3	Models and Formalisms	88
9.3.1	Markov Decision Processes (MDP)	88
9.3.2	Episodic and Continuous Problems	89
9.4	Classical RL Algorithms	89
9.4.1	Table-Based Methods	89
9.4.2	Exploration vs. Exploitation	89
9.4.3	Policy Gradient Methods	89
9.5	Deep Reinforcement Learning	89
9.5.1	Motivation	89
9.5.2	Main Algorithms	90
9.5.3	DQN Example	90
9.6	Professional Workflow with MATLAB	90
9.6.1	General Steps	90
9.6.2	Environments in MATLAB and Simulink	90
9.6.3	Custom Environment Example	91
9.6.4	Creating and Training a DQN Agent	91
9.7	Advanced Applications	92
9.7.1	Robotics and Optimal Control	92
9.7.2	Industrial Processes and Energy	92
9.7.3	Finance and Trading	92
9.7.4	Telecommunications and Networks	92

9.7.5	Games and Simulation	92
9.8	Parallelization, Deployment and Integration	92
9.8.1	Training Parallelization	92
9.8.2	Deployment	92
9.8.3	Policy Import/Export	93
9.9	Current Challenges and Future Trends	93
9.10	Practical Tips and Best Practices	93
9.11	Proposed Exercises	93
10	Symbolic Math	95
10.1	Introduction to Symbolic Mathematics in MATLAB	95
10.1.1	Evolution of Symbolic Computation in MATLAB	95
10.1.2	Advantages of Symbolic vs. Numerical Calculation	96
10.1.3	Applications and Areas of Use	97
10.2	Fundamentals of the Symbolic Math Toolbox	97
10.2.1	Symbolic Data Types	97
10.2.2	Creating Symbolic Variables and Expressions	97
10.2.3	Basic Operations with Symbolic Expressions	98
10.2.4	Substitution and Evaluation	99
10.2.5	Conversion between Symbolic and Numerical Formats	99
10.3	Advanced Symbolic Algebra	100
10.3.1	Algebraic Manipulation of Expressions	100
10.3.2	Simplification of Expressions	101
10.3.3	Manipulation of Rational Fractions	101
10.3.4	Symbolic Polynomials	102
10.3.5	Trigonometric Expressions	102
10.4	Symbolic Differential Calculus	103
10.4.1	Simple and Partial Derivatives	103
10.4.2	Implicit Differentiation	103
10.4.3	Gradients, Divergence, and Curl	104
10.4.4	Chain Rule and Applications	104
10.4.5	Jacobian and Hessian Matrix	105
10.5	Symbolic Integral Calculus	105
10.5.1	Indefinite Integration	105
10.5.2	Advanced Integration Techniques	106
10.5.3	Definite Integrals	106
10.5.4	Improper Integrals	107

10.5.5	Multiple Integrals	107
10.5.6	Applications in Physics and Engineering	108
10.6	Limits, Series, and Approximations	108
10.6.1	Calculating Limits	108
10.6.2	Taylor and Maclaurin Series	109
10.6.3	Fourier Series	109
10.6.4	Residues and Contour Integrals	110
10.7	Equations and Systems	110
10.7.1	Solving Algebraic Equations	110
10.7.2	Systems of Linear Equations	111
10.7.3	Nonlinear Systems	112
10.7.4	Ordinary Differential Equations (ODEs)	112
10.7.5	ODEs with Initial Conditions	113
10.7.6	Systems of ODEs	113
10.8	Symbolic Linear Algebra	114
10.8.1	Symbolic Matrices and Vectors	114
10.8.2	Symbolic Matrix Operations	114
10.8.3	Determinants and Inverses	115
10.8.4	Eigenvalues and Eigenvectors	115
10.8.5	Matrix Decompositions (SVD, QR, LU)	116
10.8.6	Null Spaces and Images	116
10.9	Transforms	117
10.9.1	Laplace Transform	117
10.9.2	Inverse Laplace Transform	117
10.9.3	Applications in ODEs	117
10.9.4	Fourier Transform	118
10.9.5	Z Transform	118
10.10	Applications in Physics and Engineering	119
10.10.1	Mechanics	119
10.10.2	Electromagnetism	119
10.10.3	Thermodynamics	120
10.10.4	Control	120
10.11	Applications in Financial Mathematics	121
10.11.1	Compound Interest Formulas	121
10.11.2	Option Pricing	121
10.11.3	Cash Flow Series	121
10.12	Visualization of Symbolic Expressions	122

10.12.1 Plotting Symbolic Functions	122
10.12.2 Surface and Contour Plots	122
10.12.3 Implicit Representation of Curves and Surfaces	123
10.12.4 Generation of Parametric Figures	123
10.13 Advanced Techniques and Optimization	124
10.13.1 Variable Precision Arithmetic	124
10.13.2 Symbolic Optimization	125
10.13.3 Integration with Numerical Methods	125
10.13.4 Using Assumptions and Conditions	126
10.14 Export and Integration	126
10.14.1 MATLAB Code Generation	126
10.14.2 Export to LaTeX	127
10.14.3 Export to C/C++	127
10.14.4 Integration with Simulink	128
10.15 Complete Case Studies	128
10.15.1 Robotics: Forward and Inverse Kinematics	128
10.15.2 Digital Filter Design	129
10.15.3 Control Theory: Inverted Pendulum System	129
10.15.4 Classical Calculus of Variations Problems	130
10.16 Tips, Tricks, and Best Practices	130
10.16.1 Performance Optimization	130
10.16.2 Debugging Symbolic Expressions	131
10.16.3 Effective Simplification	131
10.16.4 Working with Complex Expressions	132
10.17 Proposed Exercises	132
10.18 Appendix: Symbolic Mathematics Functions	135
10.18.1 Summary Table of Important Functions	135
10.18.2 Examples of One-Liner Code	135
11 Signal Processing	137
11.1 Introduction to Signal Processing in MATLAB	137
11.2 Fundamentals of Signals and Systems	137
11.2.1 Definition of Signal and System	137
11.2.2 Classification of Signals	138
11.2.3 Sampling and Quantization	138
11.3 Representation and Visualization of Signals in MATLAB	139
11.3.1 Creation and Importing of Signals	139

11.3.2	Temporal and Spectral Visualization	139
11.4	Transforms in Signal Processing	139
11.4.1	Fourier Transform	139
11.4.2	Short-Time Fourier Transform (STFT)	140
11.4.3	Wavelet Transform	140
11.5	Signal Filtering	140
11.5.1	Filtering Theory	140
11.5.2	FIR and IIR Filters	141
11.5.3	Filter Design and Analysis	141
11.5.4	Adaptive Filters	141
11.6	Spectral and Statistical Analysis of Signals	142
11.6.1	Power Spectral Density	142
11.6.2	Statistical Parameters	142
11.7	Digital Audio Processing	142
11.7.1	Acquisition, Playback, and Recording	142
11.7.2	Effects Processing and Noise Removal	142
11.7.3	Pitch and Formant Analysis	143
11.8	System Identification and Modeling	143
11.8.1	Linear System Identification	143
11.8.2	Impulse and Step Response	143
11.9	Advanced Applications and Real-Time Processing	144
11.9.1	Real-Time Processing	144
11.9.2	Development of Graphical Interfaces for Signals	144
11.10	Proposed Exercises	145
11.10.1	Basic Exercises	145
11.10.2	Intermediate Exercises	145
11.10.3	Advanced Exercises	145
11.11	Best Practices and Optimization in MATLAB	145
12	Image Processing	147
12.1	Introduction to Image Processing in MATLAB	147
12.1.1	History and Evolution of Image Processing	147
12.1.2	The Image Processing Toolbox	147
12.2	Fundamentals of Digital Images	148
12.2.1	Digital Representation of Images	148
12.2.2	Data Types for Images	148
12.3	Basic Operations with Images	149

12.3.1	Reading and Visualizing Images	149
12.3.2	Saving Images	149
12.3.3	Visualizing Multiple Images	149
12.3.4	Extracting Image Information	150
12.4	Geometric Transformations	150
12.4.1	Resizing	150
12.4.2	Rotation	150
12.4.3	Cropping	151
12.5	Color and Contrast Manipulation	151
12.5.1	Conversion Between Color Spaces	151
12.5.2	Contrast Adjustment	151
12.5.3	Histogram Equalization	152
12.5.4	RGB Channel Manipulation	152
12.6	Filtering and Smoothing	153
12.6.1	Mean Filters	153
12.6.2	Gaussian Filters	153
12.6.3	Median Filters	153
12.7	Edge Detection	154
12.7.1	Gradient Operators	154
12.7.2	Laplacian Operator	154
12.7.3	Canny Edge Detector	155
12.7.4	Comparative Visualization	155
12.8	Image Segmentation	155
12.8.1	Binarization and Thresholding	155
12.8.2	Watershed Segmentation	156
12.8.3	K-means for Segmentation	156
12.9	Mathematical Morphology	157
12.9.1	Basic Morphological Operations	157
12.9.2	Extraction of Morphological Features	157
12.9.3	Advanced Morphological Operations	158
12.10	Region and Object Analysis	158
12.10.1	Object Labeling	158
12.10.2	Property Extraction	159
12.10.3	Filtering Objects by Properties	159
12.11	Frequency Domain Transformations	160
12.11.1	Fourier Transform	160
12.11.2	Filtering in the Frequency Domain	160

12.11.3 Discrete Cosine Transform (DCT)	161
12.12 Object Recognition and Detection	162
12.12.1 Circle Detection with Hough Transform	162
12.12.2 Line Detection with Hough Transform	162
12.12.3 Template Matching	163
12.13 Advanced Applications	163
12.13.1 Image Compression	163
12.13.2 Image Registration	164
12.13.3 Image Restoration	165
12.14 Development of User Interfaces (GUIs)	166
12.14.1 Example Interface for Image Processing	166
12.15 Proposed Exercises	169
12.15.1 Basic Exercises	169
12.15.2 Intermediate Exercises	170
12.15.3 Advanced Exercises	170
12.16 Best Practices and Tips	170
12.16.1 Performance Optimization	170
12.16.2 Memory Management	171
12.16.3 Visualization and Presentation	171
13 Numerical Methods and Scientific Computing	173
13.1 Introduction to Numerical Mathematics	173
13.1.1 Application Areas	173
13.2 Numerical Errors and Stability	174
13.2.1 Types of Errors	174
13.2.2 Problem Conditioning	174
13.3 Solving Nonlinear Equations	174
13.3.1 Newton-Raphson Method	174
13.3.2 Secant Method	175
13.4 Systems of Linear Equations	175
13.4.1 Gaussian Elimination with Pivoting	175
13.4.2 Iterative Methods	176
13.5 Interpolation and Approximation	176
13.5.1 Lagrange Polynomial	176
13.5.2 Cubic Splines	177
13.6 Numerical Integration	177
13.6.1 Composite Simpson's Rule	177

13.6.2	Gaussian Quadrature	178
13.7	Ordinary Differential Equations	178
13.7.1	Euler Method	178
13.7.2	4th Order Runge-Kutta Method	178
13.8	Numerical Optimization	179
13.8.1	Gradient Descent Method	179
13.9	Proposed Exercises	179
13.9.1	Basic Exercises	179
13.9.2	Intermediate Exercises	179
13.9.3	Advanced Exercises	180
13.10	Best Practices in Numerical Computation	180
14	Control Systems and Simulink	181
14.1	Introduction to Control Systems with MATLAB and Simulink	181
14.1.1	MATLAB vs Simulink	181
14.2	System Modeling in Simulink	181
14.2.1	Building Dynamic Models	181
14.2.2	Integration with MATLAB	182
14.3	Controller Design in Simulink	183
14.3.1	PID Controller	183
14.3.2	Controller Design and Analysis	183
14.4	Simulation of Non-Linear and Multi-Domain Systems	184
14.4.1	Modeling Non-Linearities	184
14.4.2	Multi-Domain Systems	184
14.5	Real-Time Implementation and Simulation	184
14.5.1	Simulink Real-Time and Hardware-in-the-Loop (HIL) .	184
14.5.2	Real-Time Data Acquisition and Processing	185
14.6	Practical Example: Design of a DC Motor Speed Control System	185
14.7	Proposed Exercises	185
14.7.1	Basic Exercises	185
14.7.2	Intermediate Exercises	186
14.7.3	Advanced Exercises	186
14.8	Best Practices in Modeling and Simulation with Simulink . . .	186
15	Review Exercises	189
15.1	Proposed Exercises	189

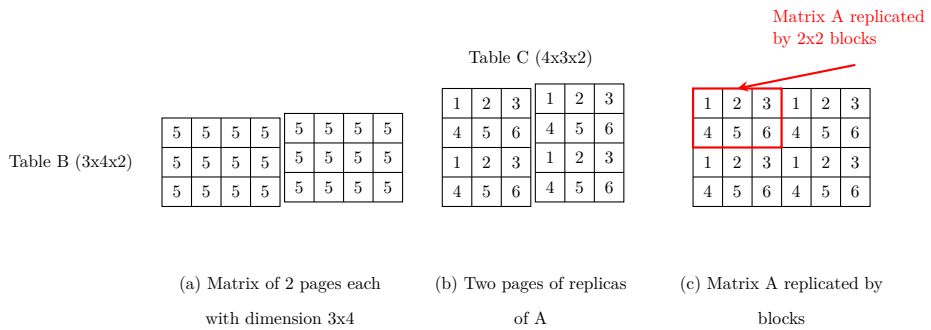


Figure 3.8: Pre-allocation of matrices.

3.6 Concatenation of matrices

a) Using square brackets

Ex: $A = [1 \ 2; \ 3 \ 4];$
 $B = [5 \ 6; \ 7 \ 8];$
 $C = [A, \ B];$ %Horizontal concatenation
 $D = [A; \ B];$ %Vertical concatenation

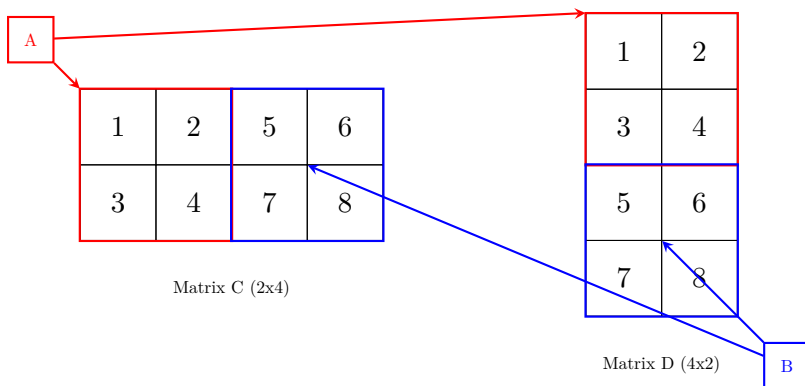


Figure 3.9: Concatenation of matrices.

b) Functions horzcat and vertcat

`horzcat(A,B,C, ...)`

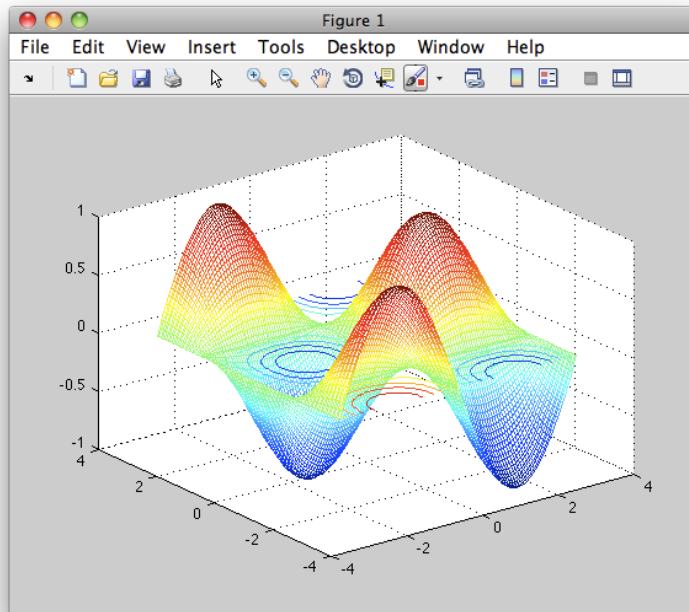


Figure 4.13: Functions `mesh` and `contour` in the same figure

4.6 Function subplot

The `subplot` function allows you to construct multiple graphs in the same window by dividing it into rows and columns. Consider the following example:

```
1 >>figure(1) % Creates the figure window
2 >>subplot(2,1,1) % Divides the figure window into 2 rows and 1 column
3 % and constructs the graph in position (1,1)
4 >>contour(X,Y,Z);
5 >>subplot(2,1,2) % Constructs the graph in position (2,1) of the
6 % figure window
7 >>mesh(X,Y,Z)
```

The output is displayed in Figure 4.14.

11.9 Advanced Applications and Real-Time Processing

11.9.1 Real-Time Processing

MATLAB allows simulating real-time processing, useful for prototyping acquisition and control systems.

```

1 while true
2     audio = getaudiodata(recObj);
3     plot(audio); drawnow;
4     pause(0.1);
5 end

```

11.9.2 Development of Graphical Interfaces for Signals

```

1 function signalViewer
2     f = figure('Name','Signal Viewer');
3     ax1 = subplot(2,1,1); ax2 = subplot(2,1,2);
4     uicontrol('Style','pushbutton','String','Load Audio',...
5         'Position',[20 20 100 30],'Callback',@loadAudio);
6
7     function loadAudio(~,~)
8         [file, path] = uigetfile('*.wav');
9         if file ~= 0
10            [audio, Fs] = audioread(fullfile(path,file));
11            t = (0:length(audio)-1)/Fs;
12            plot(ax1, t, audio); title(ax1, 'Signal in Time');
13            N = length(audio);
14            f = Fs*(0:(N/2))/N;
15            Y = fft(audio);
16            plot(ax2, f, abs(Y(1:N/2+1))); title(ax2, 'Spectrum');
17        end
18    end
19 end

```

```
11     title(axes_processed, 'Processed Image');
12
13     % Buttons for operations
14     uicontrol('Parent', fig, 'Style', 'pushbutton', ...
15             'String', 'Load Image', ...
16             'Position', [50, 200, 120, 30], ...
17             'Callback', @loadImage);
18
19     uicontrol('Parent', fig, 'Style', 'pushbutton', ...
20             'String', 'Grayscale', ...
21             'Position', [50, 150, 120, 30], ...
22             'Callback', @convertGrayscale);
23
24     uicontrol('Parent', fig, 'Style', 'pushbutton', ...
25             'String', 'Edge Detection', ...
26             'Position', [50, 100, 120, 30], ...
27             'Callback', @edgeDetection);
28
29     uicontrol('Parent', fig, 'Style', 'pushbutton', ...
30             'String', 'Equalize Histogram', ...
31             'Position', [180, 150, 120, 30], ...
32             'Callback', @equalizeHistogram);
33
34     uicontrol('Parent', fig, 'Style', 'pushbutton', ...
35             'String', 'Median Filter', ...
36             'Position', [180, 100, 120, 30], ...
37             'Callback', @medianFilter);
38
39     % Variables to store images
40     original_image = [];
41     processed_image = [];
42
43     % Callbacks
44     function loadImage(~, ~)
45         [filename, pathname] = uigetfile({'*.jpg;*.png;*.bmp',
46             'Images'});
47         if filename ~= 0
48             original_image = imread(fullfile(pathname, filename));
49             axes(axes_original);
50             imshow(original_image);
51             title(axes_original, 'Original Image');
```

8. Generate a vector $\mathbf{x}$ with 200 elements using the function `rand`. Extract to another vector all elements of $\mathbf{x}$ greater than the mean. Use the `for` and `if` statements.
9. Solve the previous exercise using the `while` and `if` statements.
10. Calculate the nodal admittance matrix of the network in Figure 15.1.

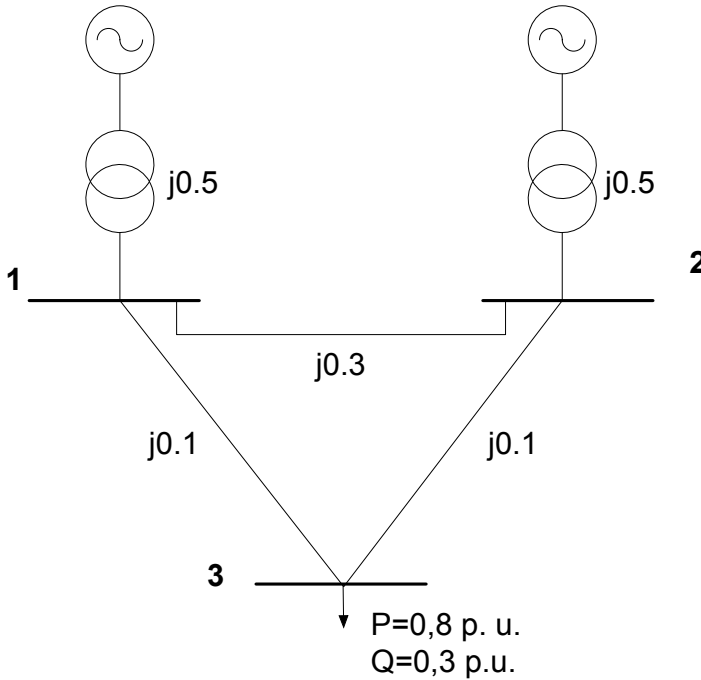


Figure 15.1: Network

11. Calculate the power flow in each line of the network in Figure 15.1 using the DC Model. Consider that:

$$\hat{\theta} = B^{-1} \cdot \hat{P} \quad \text{and} \quad P_{ik} = \frac{\theta_i - \theta_k}{x_{ik}}.$$

12. Write a MATLAB script that constructs the graph of the polynomial $p(x) = x^5 - 2x^4 + 6x^3 - 10x^2 + 15$ with a solid blue line, indicating the

MATLAB: FROM FUNDAMENTALS TO ARTIFICIAL INTELLIGENCE

FILIFE AZEVEDO

About the book

This book is an essential and comprehensive guide for students, professionals, and enthusiasts who want to explore and master MATLAB, a powerful platform for numerical computation, graphical visualization, and computational development.

Starting with the fundamentals of the language, the reader is led through a solid pedagogical progression that includes flow control, matrix manipulation, 2D and 3D graphics, scripts and functions, culminating in advanced applications in optimization, artificial intelligence, neural networks, deep learning, reinforcement learning, and modeling with Simulink.

With a practical and accessible approach, reinforced by real examples and progressive exercises, this work offers an effective learning curve, preparing the reader to face academic and professional challenges in the MATLAB universe.

About the author

Filipe Azevedo is an Associate Professor at ISEP, Polytechnic of Porto, and an integrated researcher at INESC TEC – Institute for Systems and Computer Engineering, Technology and Science. He holds a Bachelor's and Master's degree in Electrical and Computer Engineering from the Faculty of Engineering of the University of Porto, and received his PhD in Electrical and Computer Engineering from the University of Trás-os-Montes e Alto Douro with distinction and honors.

He began his professional career at EFACEC Energia in the technical-commercial area, before dedicating himself to teaching and research in higher education. Since 2000, he has been teaching at ISEP, where he has been developing work in the areas of energy systems, artificial intelligence, and engineering economics.

His research at INESC TEC focuses mainly on modeling and optimization of smart electrical networks, electricity markets, integration of renewable sources, and the application of decision support techniques. His interests also extend to energy security, cryptography, and cybersecurity, with particular attention to the impact of these technologies in strategic contexts.

With more than two decades of academic and scientific experience, he has sought to articulate pedagogical practice with applied research, developing innovative solutions and training engineers with critical thinking and technological vision.

