Introduction to Modern Controls – with Illustrations in MATLAB and Python

Xu Chen and Masayoshi Tomizuka

September 10, 2023

University of Washington
University of California, Berkeley

Introduction to Modern Controls, with Illustrations in MATLAB and Python

Copyright

License for Code

Publisher

To cite this material: Xu Chen and Masayoshi Tomizuka, “Introduction to Modern Controls – with Illustrations in MATLAB and Python,” 2023, ISBN: 9798854992640.

Preface

This book introduces the theory and practice of modern control systems. The emphasis is on using state-space methods to model, analyze, and control dynamic systems. Topics include state-space modeling and solutions, stability, controllability and observability, state-feedback control, observers, observer state feedback controls, least square estimation, Kalman filter, and Linear Quadratic Gaussian optimal control. These topics are discussed in both continuous- and discrete-time settings throughout the book.

The material in this book is based on many years of teaching experience at the University of Washington and the University of California, Berkeley. The main sources of the material are:

- ME 232 and ME 233 at the University of California, Berkeley, and
- ME 547 at the University of Washington, Seattle.

This book consists of four parts. Part I introduces the basics of dynamic systems modeling, such as Laplace and Z transforms, state-space descriptions and realization theory, and how to solve the state equation. Part II examines the properties of dynamic systems, such as classic and Lyapunov stability theories, controllability, observability, and the decomposition of an uncontrollable and/or unobservable system. After understanding these system properties, in Part III, we cover estimation and controls for state-space systems. Chapter 11 centers on the power of state feedback. Then in Chapter 12, we discuss state observers and observer-state feedback. As a powerful state-feedback control method, Chapter 13 covers the linear quadratic optimal control algorithm. Part IV is dedicated to estimation and control of stochastic systems, where the state-space system equations are subject to input and output stochastic noises. We review first relevant results in probability theory in Chapter 14, building on which we derive the least square estimation in Chapter 15 and then the discrete- and continuous-time Kalman filters in Chapter 16. Chapter 17 integrates linear quadratic optimal control with the Kalman filter, to provide the celebrated Linear Quadratic Gaussian (LQG) Optimal control. At the end of the book, we provide a review of related linear algebra for controls.

Over three hundred examples, figures, table summaries, and exercises distilled from physical systems supplement the learning. Perhaps unique in this book is the modernization of tools we use to illustrate controls. MATLAB and Python are the primary tools for our numerical demonstrations. Whenever MATLAB examples appear, complementary Python codes will follow to provide results as equivalent as possible in the more nascent computation environment.

All the main codes are available for download on the book website https://mcimp-book.github.io/mcimp/. We have also provided accompanying slides and lecture recordings – all accessible from the book website.

All the MATLAB demonstrations were performed in `MATLAB 2022b`, and the Python demonstrations in `Python 3.9.13`, using toolboxes SymPy v1.11.1 and python-control v0.9.2. For simple calculations and graphical illustrations, we use gnuplot, a light-weight command-line driven graphing utility across different operation systems. The coding commands and results are all provided in an "in-line" fashion, directly embedded in the text materials. Appendix "`How to Install and Run Python`" provides a summary of ways to configure Python in different operation systems.

We are grateful to the many teaching assistants and students who helped typeset problems and proofread course contents over the years. In particular, we would like to thank Liting Sun for creating some of the drawings in LaTeX and Lingfeng Sun for typesetting the Berkeley ME 233 course reader. Jonas Beachy, Xiaohai Hu, and Marina Ruediger helped with a few case studies. Their contributions have greatly enhanced the quality of this book.

We hope this book will serve as a useful reference for students and researchers interested in the field of dynamics and control.

September 10, 2023
Seattle, Washington
Berkeley, California

Contents

Preface iii

Contents v

1 Introduction 1

1.1 The Power of Controls . . . 1

1.2 Relevant Terminologies . . . 1

1.3 The Objectives and The Means of Controls . . . 3

1.4 Societies to Learn More about Controls . . . 5

System Description 7

2 Modeling 9

2.1 Methods of Modeling . . . 9

2.2 Continuous-Time Systems . . . 10

2.3 Discrete-Time Systems . . . 11

2.4 Example: Atomic Force Microscopy . . . 11

2.5 Example: Hard Disk Drive and Information Storage . . . 14

2.6 Model Properties . . . 19

2.7 Nonlinear Systems . . . 20

2.7.1 Example: Magnetically Suspended Ball . . . 20

2.7.2 Example: Water Tank . . . 21

2.7.3 Example: Pendulum . . . 21

2.7.4 Example: Vehicle Steering . . . 22

2.8 "All Models are Wrong, but Some are Useful" . . . 24

2.9 Exercise . . . 28

3 Laplace and Z Transforms 29

3.1 The Laplace Transform . . . 29

3.1.1 The Laplace Approach to ODEs . . . 29

3.1.2 Relevant Properties . . . 36

3.2 Inverse Laplace Transform and Partial Fraction Expansion . . . 39

3.3 From Laplace Transform to Transfer Functions . . . 41

3.4 The Z Transform . . . 43

3.4.1 Introduction . . . 43

3.4.2 Relevant Properties . . . 45

3.4.3 Applications to Dynamic Systems . . . 50

3.5 From Difference Equation to Discrete-Time Transfer Functions . . . 51

3.6 Recap . . . 52

3.7 Exercise . . . 54

4 State-Space Description of a Dynamic System 57

4.1 The Concept of States . . . 57

4.2 General State-Space Descriptions . . . 58
4.3 From the State Space to Transfer Functions . . . 60
4.4 Linearization and State-Space Representation of Nonlinear Systems . . . 66
4.4.1 State-Space Representation of General Nonlinear Systems . . . 66
4.4.2 Equilibrium Point and Linearization around an Equilibrium Point . . . 67
4.4.3 Multivariariate Partial Derivative . . . 69
4.4.4 Example: Water Tank . . . 71
4.4.5 Example: Vehicle Steering . . . 73
4.5 Recap . . . 81
4.6 Exercise . . . 81

5 State-space Realizations: The Canonical Forms 83
5.1 Controllable Canonical Form . . . 83
5.2 Observable Canonical Form . . . 86
5.3 Diagonal and Jordan Canonical Forms . . . 90
5.4 Discrete-Time LTI Systems and Their State-Space Canonical Forms . . . 92
5.5 Similar Realizations . . . 94
5.6 Recap . . . 96
5.7 Exercise . . . 97

6 Solution of Time-Invariant State-Space Equations 99
6.1 Continuous-Time State-Space Solutions . . . 99
6.1.1 The Solution to $\dot{x} = ax + bu$ . . . 99
6.1.2 Fundamental Theorem of Differential Equations . . . 102
6.1.3 The Solution to n^{th}-order LTI System . . . 102
6.1.4 The State Transition Matrix e^{At} . . . 103
6.2 Discrete-Time LTI State-Space Solutions . . . 106
6.2.1 General Solution . . . 106
6.2.2 The State Transition Matrix A^k . . . 107
6.3 Explicit Computation of the State Transition Matrix e^{At} . . . 108
6.3.1 Principle Concept . . . 108
6.3.2 Existence of Solutions . . . 108
6.3.3 Computing the Eigenvalues and Eigenvectors to form T . . . 109
6.4 Explicit Computation of the State Transition Matrix A^k . . . 118
6.5 Transition Matrix via Inverse Transformation . . . 118
6.6 Solutions of Time-Varying State Equations . . . 120
6.6.1 Continuous-Time Case . . . 120
6.6.2 Discrete-Time Case . . . 121
6.7 Recap . . . 121
6.8 Exercise . . . 122

7 Discrete-Time Models of Continuous Systems 125
7.1 State-Space Models . . . 125
7.2 Transfer-Function Models . . . 129
7.3 Exercise . . . 133

System Properties 135

8 Stability 137
8.1 Definitions . . . 137
8.1.1 Review of Relevant Functional Analysis . . . 137
8.1.2 Lyapunov's Definition of Stability . . . 138
8.2 Stability of LTI Systems . . . 141
8.2.1 Method of Eigenvalue Locations . . . 141
8.2.2 Routh-Hurwitz Criterion for Continuous-Time LTI Systems . . . 142
8.2.3 Routh-Hurwitz Criterion for Discrete-Time LTI Systems . . . 144
8.3 Lyapunov's Approach to Stability . . . 145
8.3.1 Stability from an Energy Viewpoint . . . 145
8.3.2 Relevant Mathematical Tools to Formalize the Concept . . . 146
8.4 Lyapunov Stability Theorems . . . 155
8.4.1 Lyapunov Stability for Continuous-Time Systems . . . 155
8.4.2 Lyapunov Stability for Discrete-Time Systems . . . 161
8.5 Recap . . . 164
8.6 Exercise . . . 166

9 Controllability and Observability 167
9.1 The Case for Discrete-Time Systems . . . 168
9.1.1 Controllability . . . 168
9.1.2 Observability . . . 173
9.2 The Case for Continuous-Time Systems . . . 177
9.2.1 Controllability . . . 177
9.2.2 Observability . . . 180
9.3 Transforming Controllable Systems Into the Controllable Canonical Form . . . 186
9.4 Transforming Observable Systems Into the Observable Canonical Form . . . 188
9.5 Recap . . . 191
9.6 Exercise . . . 191

10 Kalman Decomposition 195
10.1 Kalman Decomposition of Uncontrollable Systems . . . 196
10.2 Kalman Decomposition of Unobservable Systems . . . 204
10.3 General Kalman Decomposition, Stabilizability, and Detectability . . . 206
10.4 Recap . . . 207
10.5 Exercise . . . 208

Estimation and Control 209

11 State Feedback 211
11.1 State Feedback and Eigenvalue Assignments . . . 211
11.1.1 Dynamic Properties of the Closed-Loop System . . . 212
11.1.2 Eigenvalue Assignment for Single-Input Systems . . . 213
11.1.3 Multi-Input Systems . . . 215
11.2 Output Feedback . . . 216
11.3 Recap . . . 216
11.4 Exercise . . . 217

12 Observers and Observer-State Feedback **219**
12.1 Introduction . . . 219
12.2 Open-Loop Observer . . . 219
12.3 Continuous-Time Luenberger Observer . . . 220
12.4 Observer State Feedback Control . . . 228
12.5 Discrete-Time Observers . . . 229
12.5.1 Discrete-Time Full State Observer . . . 229
12.5.2 Discrete-Time Full State Observer with Predictor . . . 229
12.6 Reduced-Order Observer . . . 232
12.7 Recap . . . 237
12.8 Exercise . . . 238

13 Linear Quadratic Optimal Control **241**
13.1 Problem Formulation . . . 241
13.2 Solution of the Continuous-Time LQ Problem . . . 242
13.3 Stationary Continuous-Time LQ Problem . . . 250
13.4 Application and Practice . . . 256
13.4.1 Choosing Q and R . . . 256
13.4.2 MATLAB and Python Commands . . . 256
13.4.3 Obtaining P_+ . . . 256
13.4.4 Robustness of Stationary LQ Regulators . . . 258
13.4.5 Example: Inverted Pendulum on a Cart . . . 258
13.5 Further Development of the Continuous-Time LQ Regulator . . . 271
13.5.1 Return Difference Equality . . . 271
13.5.2 Stability Margins of Scalar LQR . . . 272
13.5.3 The Closed-Loop Eigenvalues . . . 273
13.5.4 Symmetric Root Locus . . . 275
13.5.5 Steady-State Property . . . 281
13.6 Discrete-Time LQ Optimal Control . . . 283
13.6.1 Introduction of Dynamic Programming . . . 283
13.6.2 Dynamic Programming for General Optimal Control Problems . . . 284
13.6.3 Solving Discrete-Time Finite-Horizon LQ Problem via Dynamic Programming . . . 286
13.6.4 Discrete-Time Stationary LQ Optimal Control . . . 289
13.7 Further Development of the Discrete-Time LQ Regulator . . . 293
13.7.1 Obtaining P_+ . . . 293
13.7.2 Return Difference Equality . . . 293
13.7.3 The Closed-Loop Eigenvalues . . . 295
13.8 Recap . . . 297
13.9 Exercise . . . 302

STOCHASTIC ESTIMATION AND CONTROL **305**

14 Review of Probability Theory **307**
14.1 Introduction . . . 307
14.2 Sample Space, Events, and Probability Axioms . . . 307
14.3 Random Variables, Probability Density, and Moments of Distributions . . . 308
14.4 Example Distributions . . . 309

14.5 Random Vector, Joint Probability and Distribution, Conditional Probability . . . 313
14.6 Discrete-Time Random Process . . . 317
14.6.1 Filtering a Random Process . . . 319
14.6.2 Filtering a Random Process in the State Space . . . 320
14.7 Continuous-Time Random Process . . . 322
14.8 Recap . . . 325
14.9 Exercise . . . 328

15 Least Square Estimation 331
15.1 General Solution . . . 331
15.2 Solution in the Gaussian Case . . . 332
15.3 Properties of Least Square Estimate (Gaussian Case) . . . 334
15.4 Example Application of Least Square Estimation . . . 338
15.5 Recap . . . 343
15.6 Exercise . . . 344

16 Stochastic State Estimation and Kalman Filter 345
16.1 Review of State Observers . . . 345
16.2 Discrete-Time Stochastic State Estimation . . . 345
16.2.1 Problem Definition . . . 345
16.2.2 Solution . . . 347
16.2.3 Discrete-Time Kalman Filter . . . 349
16.2.4 Steady-State Kalman Filter . . . 350
16.2.5 Return Difference Equality for Equation 16.29 . . . 360
16.3 Continuous-Time Stochastic State Estimation . . . 364
16.3.1 General Continuous-Time Kalman Filter . . . 364
16.3.2 Steady-State Kalman Filter . . . 367
16.3.3 Return Difference Equality . . . 367
16.4 Kalman Filter Application: Kinematic Kalman Filter . . . 369
16.5 The Kalman Filter Equations using Other Notation Systems . . . 370
16.6 Recap . . . 373
16.7 Exercise . . . 376

17 Linear Quadratic Gaussian (LQG) Optimal Control 379
17.1 Stochastic Control with Exactly Known State . . . 379
17.1.1 Stochastic Control with Inexactly Known State . . . 382
17.1.2 Stationary LQG Problem . . . 385
17.2 Continuous-Time LQG Problem . . . 386
17.3 Recap . . . 387
17.4 Exercise . . . 390

18 Further Readings 393

Review of Relevant Linear Algebra 395
1 Basic Concepts of Matrices and Vectors . . . 395
1.1 Matrix Addition and Multiplication . . . 396
1.2 Matrix Transposition . . . 397
2 Linear Systems of Equations . . . 397
3 Vector Space, Linear Independence, Basis, and Span . . . 400

4 Matrix Properties . . . 401
4.1 Rank . . . 401
4.2 Range and Null Spaces . . . 401
4.3 Determinants . . . 401
4.4 Matrix and Linear Equations . . . 402
5 Eigenvalues and Eigenvectors . . . 402
5.1 Matrix, Mappings, and Eigenvectors . . . 402
5.2 Computation of Eigenvalue and Eigenvectors . . . 404
5.3 Eigenbase and Diagonalization . . . 405
5.4 Similarity Transformation . . . 407
6 Matrix Inversion . . . 408
6.1 Block Matrix Decomposition and Inversion . . . 410
6.2 LU and Cholesky Decomposition . . . 412
6.3 Determinant and Matrix Inverse Identity . . . 413
7 Spectral Mapping Theorem . . . 415
8 Matrix Exponential . . . 416
9 Inner Product . . . 417
9.1 Inner Product Spaces . . . 417
9.2 Trace (Standard Matrix Inner Product) . . . 418
10 Vector Norms . . . 418
11 Symmetric and Orthogonal Matrices . . . 419
12 Positive-Definite Matrices . . . 421
12.1 Definitions and Basic Properties . . . 421
12.2 General Positive-Definite Matrices . . . 422
12.3 Positive-Definite Functions . . . 422
13 Singular Value Decomposition (SVD) . . . 423
13.1 Motivation . . . 423
13.2 Main Result . . . 423
13.3 Properties of Singular Values . . . 425
14 Induced Matrix Norm . . . 425

How to Install and Run Python 427

Alphabetical Index 431

About the Authors 433

List of Figures

1.1 A general block diagram. 2
1.2 An open-loop control system. 2
1.3 A closed-loop heating control example. 3
1.4 Control system with feedforward control. 3
1.5 Structure of digital control systems. 4
1.6 Design approaches of digital control systems. 5

2.1 Major components in a hard disk drive. 9
2.2 Example frequency response of the voice coil motor in a hard disk drive. 10
2.3 Isaac Newton. 11
2.4 Schematic of the AFM. 12
2.5 Mechanical models of an AFM. 13
2.6 Control-related components in a hard disk drive. 14
2.7 Baseline model of the VCM component in an HDD. 15
2.8 Schematic structure of dual-stage HDDs. 17
2.9 Baseline model of the PZT component in an HDD. 17
2.10 Kinematic modelling of the four-wheel vehicle steering system. 22
2.11 The bicycle model of the vehicle steering system. 23
2.12 Example frequency responses of the voice coil motor stage in a batch of hard disk drives. 24

3.1 The Laplace approach to ODEs. 29
3.2 Pierre-Simon Laplace. 30
3.3 Illustration of piecewise continuous functions. 30
3.4 Illustration of exponential order. 31
3.5 Illustration of the Dirac impulse function. 33
3.6 Approximation of the unit-step function. 34
3.7 Approximation of the Dirac-delta function. 35
3.8 Toward smoother approximations of the unit-step function. 35

5.1 Block diagram of the controllable canonical form for third-order systems. 85
5.2 Block diagram of the observable canonical form for third-order systems. 87
5.3 Block diagram of the diagonal realization for third-order systems with distinct poles. 90
5.4 Block diagram of the Jordan form realization for third-order systems with repeated poles. 91
5.5 Block diagram of the modified canonical form for third-order systems with complex poles. $k_2 = (\beta + \alpha\sigma)/\omega$ and $k_3 = \alpha$. 92

6.1 The concept of the time constant. 101
6.2 The unit-step response of a first-order system. 101
6.3 Example state trajectory for a second-order system under different initial conditions. 112
6.4 Map of eigenvalue and associated response mode (continuous-time case). 116
6.5 Map of eigenvalue and associated response mode (discrete-time case). 117

7.1 A continuous-time state-space system preceded by zero order hold. 125
7.2 Illustration of the input and output of a zero order hold. 126

7.3 Continuous-time transfer function preceded by a zero order hold. 130

8.1 Example continuous functions. 138
8.2 A uniformly continuous function. 138
8.3 A continuous but not uniformly continuous function. 138
8.4 Stability concepts in the sense of Lyapunov. 139
8.5 Time-domain plot of function $te^{\lambda t}$. 141
8.6 The bilinear transform maps the closed left half plane to the closed unit disk. 144
8.7 A positive-definite function $W(x_1, x_2) = x_1^2 + x_2^2$. 154
8.8 A locally positive-definite function $W(x_1, x_2) = x_1^2 + \sin^2 x_2$. 155

9.1 Arthur Cayley. 169
9.2 William Hamilton. 170

11.1 Illustration of state-feedback control. 211

12.1 Concept of a closed-loop observer. 220
12.2 Realization of the Luenberger observer. 221
12.3 Block diagram of observer state feedback control. 228
12.4 Block diagram of the continuous-time reduced-order observer. 234
12.5 Block diagram of reduced-order observer: implementation form. 235

13.1 LQ example with small penalty on control. $P^*(0) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, $R = 0.0001$ 247
13.2 LQ example with medium penalty on control. $P^*(0) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, $R = 1$ 247
13.3 LQ example with large penalty on control. $P^*(0) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, $R = 100$ 247
13.4 LQ example with large penalty on control. $P^*(0) = \begin{bmatrix} 20 & 0 \\ 0 & 2 \end{bmatrix}$, $R = 100$ 248
13.5 State trajectories of an inverted pendulum on a cart under full-state LQ optimal control. 262
13.6 State trajectories of a linearized inverted pendulum on a cart under full-state LQ optimal control. . 264
13.7 State trajectories of a linearized inverted pendulum on a cart under observer-state LQ control. . . 267
13.8 State estimation errors of a linearized inverted pendulum on a cart under observer-state LQ control. 268
13.9 State estimation errors of a linearized inverted pendulum on a cart under observer-state LQ control. 270
13.10 Block diagram of the optimal LQ system. 271
13.11 Frequency response of the continuous-time LQ loop transfer function. 273
13.12 Frequency response of the discrete-time LQ loop transfer function. 295

14.1 Probability density function of a uniform distribution. 309
14.2 Probability density function of a Gaussian distribution. 310
14.3 Distribution of the addition of two uniformly distributed random variables. 312
14.4 Distribution of the addition of three uniformly distributed random variables. 313
14.5 Illustration of a discrete-time random process. 317

15.1 Geometric interpretation of the least square estimator $E[x|y]$. 335
15.2 Geometric interpretation of the least square estimator $E[x|y,z]$ where y and z are uncorrelated. . 336
15.3 Geometric interpretation of the least square estimator $E[x|y,z]$ where y and z are correlated. . . 338

16.1 Block-diagram representation of the corrector and predictor form of the Kalman filter. 349
16.2 State estimation by Kalman filter and finite settling time observer (FSTO) with $r = 2$. 354
16.3 Performance of the Kalman filter and finite settling time observer (FSTO). 355
16.4 A closed-loop return-difference representation of the Kalman filter. 361
16.5 Block diagram of the return difference in a Kalman filter ($u(t) = 0$ or the forced response has been subtracted). 367
16.6 Example eigenvalues of a second-order Kalman filter. 369

17.1 Structure of the LQG control scheme. 385

1 Example relationship between x and Ax. 403
2 Decomposition of x. 403
3 Construction of Ax. 403
4 Example points in vector space 1. 407
5 The same points in vector space 2. 408

List of Tables

3.1 Common Laplace transforms. 39
3.2 Common properties of the Z transform. 48
3.3 Table of Laplace and Z transforms [5]. $x(t) = 0$ for $t < 0$. $x(kT) = x(k) = 0$ for $k < 0$. Unless otherwise noted, $k = 0, 1, 2, 3, \ldots$ 49
3.4 Comparison of continuous- and discrete-time transfer functions. 52

6.1 Common pairs of J and J^k. 118
6.2 Summary of solutions to state equations. 118
6.3 Common state-transition matrices. 122

7.1 Zero order hold equivalents of continuous-time transfer functions. 131

8.1 Stability of the origin for $\dot{x} = Ax$. 142
8.2 Stability of the origin for $x(k+1) = Ax(k)$. 142
8.3 Analogy between symmetric, skew-symmetric, and orthogonal matrices and the complex plane. . . 149
8.4 Summary of Lyapunov equations in continuous- and discrete-time dynamics. 165

13.1 Summary of continuous- and discrete-time LQ optimal-control properties. 301

16.1 Comparison of the estimation error covariances (diagonal entries) between a Kalman filter and a finite settling time observer. 353

Introduction 1

1.1 The Power of Controls 1
1.2 Relevant Terminologies . . . 1
1.3 The Objectives and The Means of Controls 3
1.4 Societies to Learn More about Controls 5

1.1 The Power of Controls

Our internal body temperature is regulated around the normal value of about 37° C or 98.6° F. A part of our brain called the hypothalamus checks our current temperature and compares it with the normal value. In a sauna room where the temperature is too high, sweat is produced to cool the skin,[1] and the blood vessels under our skin get wider to increase the blood flow to the skin.[2] On the other hand, when building a snowman outside, the blood vessels under our skin become narrower to decrease blood flow to the skin, retaining heat near the warm inner body; muscles, organs, and brain produce heat (e.g., muscles can produce heat by shivering); and our thyroid gland releases hormones to increase our metabolism.[3] [4] That is the power of *feedback controls*: it allows us to make a precision device out of a crude one that works well even in changing environments.

We also use *prediction and feedforward controls* in our regulation of body temperature: as kids, we had learned to wear T-shirts in summer, long sleeves and coats in winter. With such predictive and feedforward controls, the burden of feedback control is greatly lifted.

Using these temperature-control activities, fine-tuned naturally as we grow, our body can respond to internal and external stimuli and make adjustments to keep the body within one or two degrees of the normal temperature, whether in summer or winter, at the north pole or in the Sahara Desert!

1.2 Relevant Terminologies

More formally, **feedback** is the use of information of the past or the present to influence behaviors of a system. A **system** is an interconnection of elements and devices for a desired purpose. In your undergraduate control course, you have obtained basic understandings of a control system with concepts such as transfer functions, proportional integral derivative (PID) controllers, and frequency responses in classic control. Below, we provide a brief review of the key concepts and terminologies.

We use a **block diagram** such as the one in Figure 1.1 to visualize the system structure and the interconnection of system components. Here, the **plant** consists of: (i) a process whose output is to be controlled and (ii) an actuator – a physical device capable to influence the controlled variable of the process. The sensor measures the output of the plant and feeds it back to be compared to a reference signal. The error after the comparison then drives the controller to generate the command for the actuator. It is not uncommon to have an input filter that shapes the reference signal:

[1] The middle layer of the skin, or dermis, stores most of the body's water. When the temperature is too high, that water, along with the body's salt, are brought to the surface of the skin as sweat. On the skin surface, the water evaporates and cools the body.

[2] This process is called vasodilatation.

[3] The three mechanisms are called vasoconstriction, thermogenesis, and hormonal thermogenesis, respectively.

[4] We are continuously losing heat: The Basal Metabolic Rate (BMR) is the number of calories we burn as our body performs basic (basal) life-sustaining function. An average man has a BMR of around 7,100 kJ per day, while an average woman has a BMR of around 5,900 kJ per day. See more at https://www.betterhealth.vic.gov.au/health/conditionsandtreatments/metabolism.

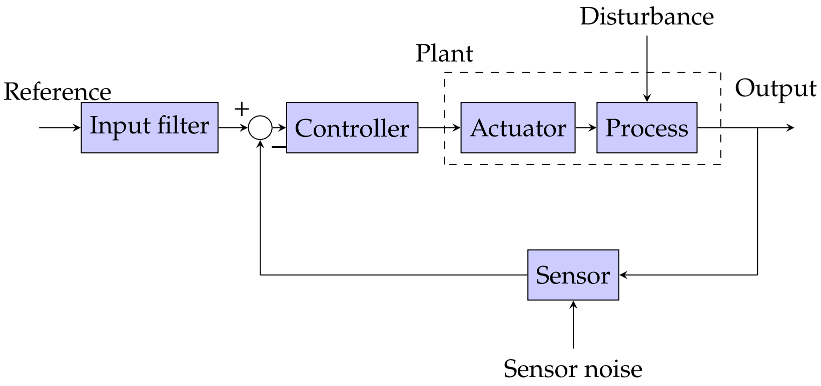


Figure 1.1: A general block diagram.

for example, in controlling the body temperature, we wear clothes of different thickness and warmth that build a buffer between our skin and the environment.

In our introductory example, the body cannot immediately adjust the temperature. Instead, it is an intricate **dynamic process** – blood vessels expand and contract to move blood and heat closer to or further away from the skin, thus releasing or conserving warmth. Dynamic systems do not show the effect of the input immediately but after some **transient** and/or **delays**.

If the mechanism of transient and delay is not correctly taken into account, the feedback controller may over compensate or pump in an excessive amount of control energy, resulting in system instability.

The inputs and outputs in a dynamic system are *signals*, i.e., they are functions of time, e.g., speed of a car, temperature in a room, voltage applied to a motor, price of a stock, and electrical-cardiograph. In practice, there will be disturbances to the system and noises in the sensor measurement signals. These are also signals and will negatively influence the performance of the control system.

An **open-loop** control system (cf. Figure 1.2) is one where the output of the plant does not influence the input to the controller.

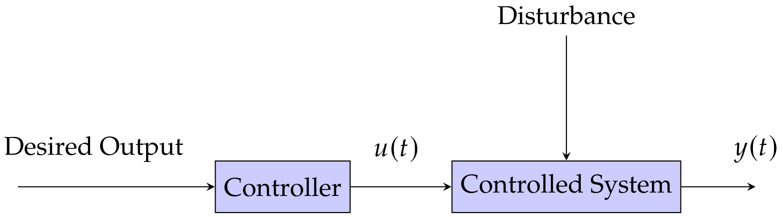


Figure 1.2: An open-loop control system.

A **closed-loop** system is one where multiple components (plant, controller, etc) have a closed interconnection. For exampel, in the heating control system for a house, the thermostat will measure the room temperature, compare it to the set value, and turn on or off the furnace to keep the house warm in a closed loop. There is always feedback in a closed-loop system.

Closed-loop (feedback) controllers provide a more robust performance than open-loop controllers in the presence of disturbances and plant uncertainties. If the reference signal varies fast and its variation is known in advance, then **feedforward** control based on information about the

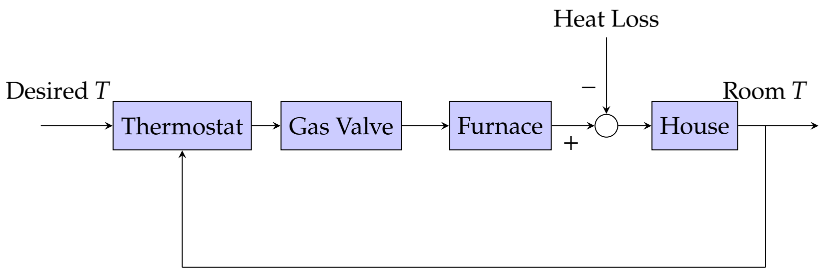


Figure 1.3: A closed-loop heating control example.

reference is useful. This is often the case in machine tool control and robot control. Feedforward control is also effective if the disturbance signal can be measured or estimated. Figure 1.4 shows a control system that integrates these benefits of feedback and feedforward controls.

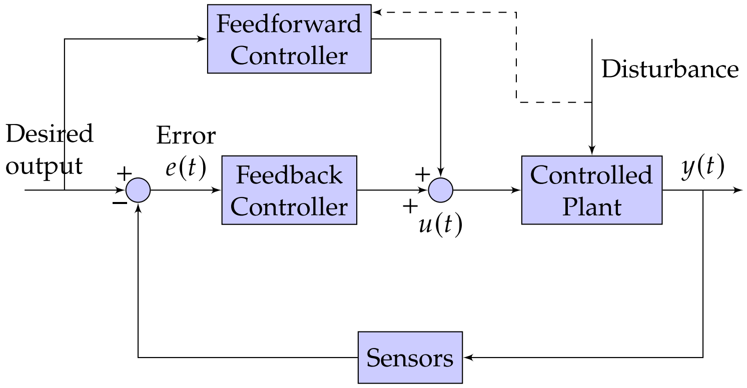


Figure 1.4: Control system with feedforward control.

1.3 The Objectives and The Means of Controls

There are some aspects of control objectives that are universal. For example, we would always want our control system to result in closed-loop dynamics that are stable and insensitive to disturbances. These form the **stabilization** problem and the **disturbance rejection** problem, respectively. When the reference is a fixed point such as the normal temperature of our body, the control objective is the **regulation** of output in the presence of disturbances and noises. When the reference changes and is time-varying, a **tracking** problem is formed.

To achieve the control objectives, the control engineer must *model* the controlled plant, *analyze* the characteristics of the plant, *design* control algorithms (controllers), *analyze* performance and robustness of the control system, and *implement* the controller.

From a hardware viewpoint, control engineers use computers extensively in both (off-line) analysis/design and (real-time) implementation. Computers are inherently discrete devices. If a computer is used in real-time control, it receives the output of the controlled plant as a sensor signal intermittently often at a fixed sampling frequency after an analog to digital (A/D) converter, computes the right control input and send it out to the controlled plant, as depicted in Figure 1.5. The discrete control sequence is then sent to a digital to analog (D/A) converter to form the continuous-time control command to the plant. From the viewpoint of the computer, the plant is a **discrete-time** device that produces a discrete-time output sequence in response to a control sequence provided by the computer.

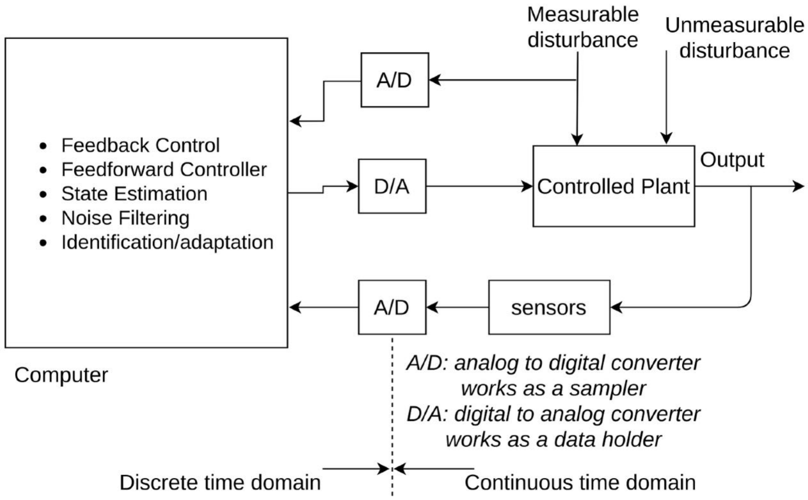


Figure 1.5: Structure of digital control systems.

While computers are used in almost all implementation of control, the controller design, however, may be carried out in the **continuous-time domain** or in the **discrete-time domain**. The choice of the time domain may depend on many factors such as target systems, control methodologies, and personal taste. It is important that the control engineers have a broad knowledge of the analysis and design tools for control systems. For example, there are many design approaches to digital control systems. Figure 1.6 summarizes typical design approaches that we may follow given a physical plant. The continuous-time linear control theory and the discrete-time control theory that we will study will provide us an important and useful set of tools.

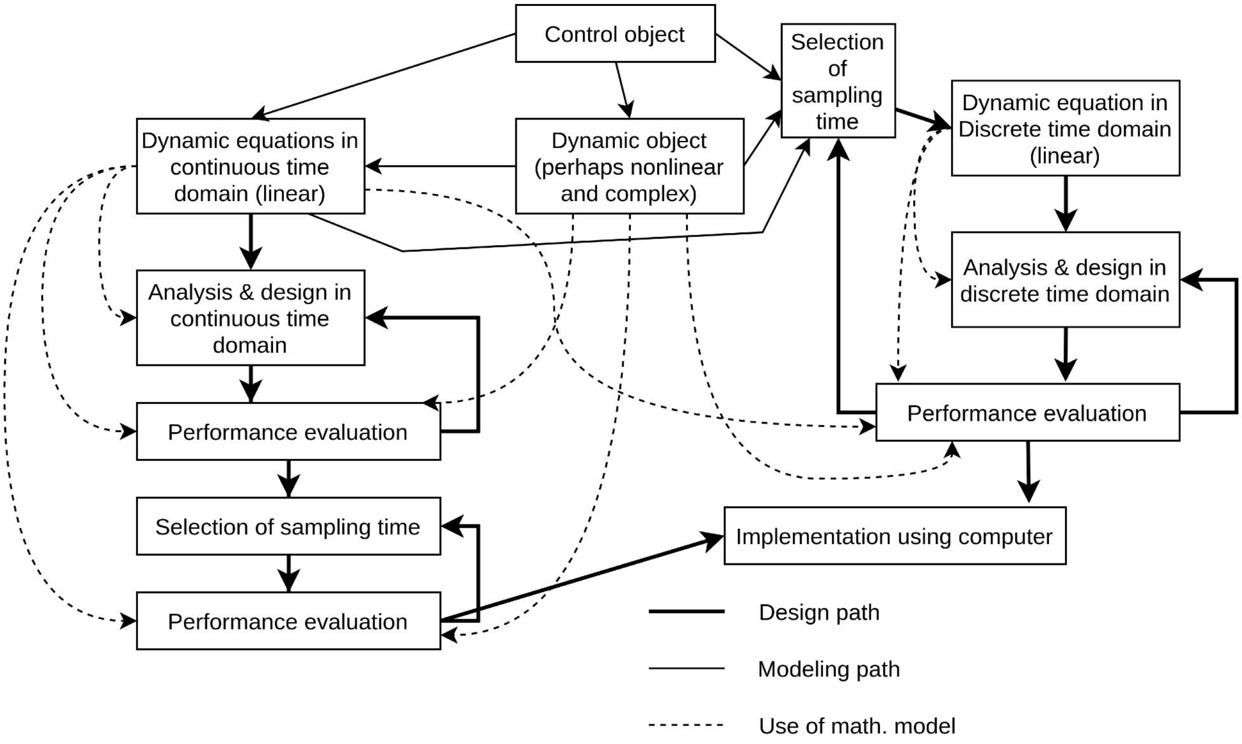

Control object
Selection of sampling time
Dynamic equation in Discrete time domain (linear)
Dynamic equations in continuous time domain (linear)
Dynamic object (perhaps nonlinear and complex)
Analysis & design in continuous time domain
Analysis & design in discrete time domain
Performance evaluation
Performance evaluation
Selection of sampling time
Performance evaluation
Implementation using computer
Design path
Modeling path
Use of math. model

Figure 1.6: Design approaches of digital control systems.

1.4 Societies to Learn More about Controls

Founded in Paris in 1957, the International Federation of Automatic Control (IFAC, website: https://www.ifac-control.org) is the worldwide organization tasked with promoting the science and technology of automatic control in both theory and application. IFAC also disseminate the impact of control technology on society through its conferences, publications, technical committees, and journals. IFAC is well known through the editorship of eight archival journals:

- Automatica,
- Control Engineering Practice,
- Annual Reviews of Control,
- Engineering Applications of Artificial Intelligence,
- Journal of Process Control,
- Mechatronics,
- Nonlinear Analysis: Hybrid Systems, and
- IFAC Journal of Systems and Control.

These are published in partnership with the official IFAC publisher, Elsevier. Control Engineering Practice and Annual Reviews of Control, for example, are good starting points that are rich in examples and reviews of recent advances in controls.

The American Automatic Control Council (AACC, website: www.a2c2.org) represents the United States to the global control community and is the US

National Member Organization (NMO) of IFAC. AACC helps arrange for IFAC events in the U.S. and is an association of nine professional societies:

- American Institute of Aeronautics and Astronautics (AIAA)
- American Institute of Chemical Engineers (AIChE)
- American Society of Civil Engineers (ASCE)
- American Society of Mechanical Engineers (ASME)
- Institute of Electrical and Electronics Engineers (IEEE)
- Institute for Operations Research and the Management Sciences Applied Probability Society (INFORMS APS)
- International Society of Automation (ISA)
- Society for Industrial and Applied Mathematics (SIAM)
- Society for Modeling and Simulation International (SCS)

AIAA, ASME, and IEEE, for example, publish journals such as

- IEEE Control Systems Magazine
- IEEE Transactions on Control Systems Technology
- IEEE Transactions on Automatic Control
- AIAA Journal of Guidance, Control and Navigation
- ASME Journal of Dynamic Systems, Measurement and Control

These are good starting good points at AACC to learn more about controls in both theory and applications.

System Description

2 Modeling

Modeling of physical systems is a vital component of modern engineering. After we understand the governing dynamics of a system, we can simulate and predict system responses, design model-based controllers, and evaluate system properties.

2.1 Methods of Modeling . . . 9
2.2 Continuous-Time Systems 10
2.3 Discrete-Time Systems . . 11
2.4 Example: Atomic Force Microscopy 11
2.5 Example: Hard Disk Drive and Information Storage . 14
2.6 Model Properties 19
2.7 Nonlinear Systems 20
2.8 “All Models are Wrong, but Some are Useful” 24
2.9 Exercise 28

2.1 Methods of Modeling

The dynamics of many systems often consist of complex coupled differential or difference equations. Two general approaches exist to extract these system models. The first and more physics-based approach capitalizes on principles of physics such as Newton’s laws and energy conservation. The second and more data-centric approach integrates input-output responses to extract the system dynamics.[1]

The most successful modeling often integrates both physics- and data-based modeling and analysis techniques. We provide an example below.

In a hard disk drive storage system, the main dynamics of the voice coil motor that rotates the read/write head are governed by Newton’s second law for rotation:

$$\underbrace{\tau}_{\text{net torque}} = \underbrace{J}_{\text{moment of inertia}} \underbrace{\alpha}_{\text{angular acceleration}} .$$

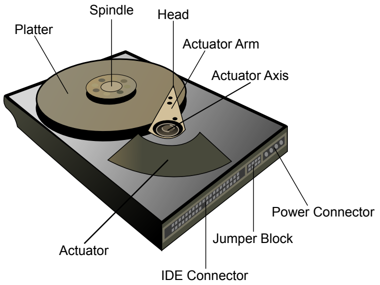


Figure 2.1: Major components in a hard disk drive.
https://en.wikipedia.org/wiki/Hard_disk_drive#/media/File:Hard_drive-en.svg

Let the angular position θ be the output and τ be the input. Then the input-output dynamics follow the formula:

$$\ddot{\theta} = \alpha = \frac{1}{J}\tau.$$

[1] The field of system identification and adaptive control is dedicated to such data-based approach to model and control dynamic systems.

However, hard disk drives are high-speed high-precision (nanometer-scale!) mechatronic systems. In addition to the above fundamental mode, at high angular speeds and frequencies, the rotating disks and actuator arms become no longer rigid, but instead will bend and exhibit high-order vibration modes. Figure 2.2 shows the frequency response of a typical voice coil motor in modern hard disk drives. Many vibration modes appear at high frequencies. The parameters of these modes are not as easy to obtain analytically. Finite element methods and system identification become useful here. However, the analysis from physics is still critical to understand the shapes of these vibration modes.

In section 2.5, we discuss details about how to construct the model in Figure 2.2.

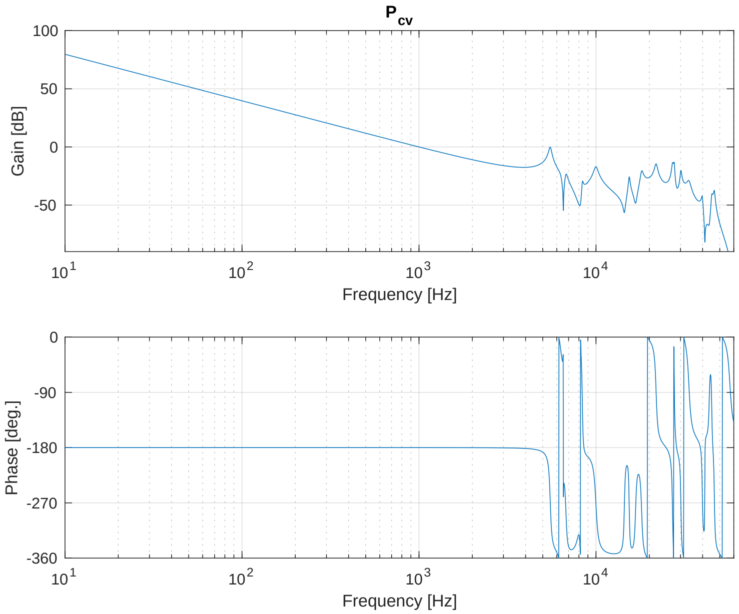


Figure 2.2: Example frequency response of the voice coil motor in a hard disk drive.

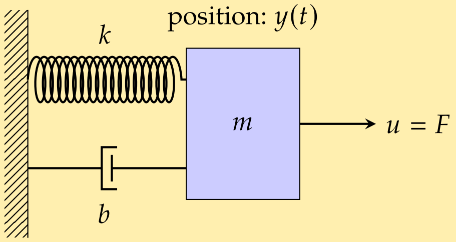
2.2 Continuous-Time Systems

Mathematical models of continuous dynamic systems are differential equations. Here, inputs and outputs of the continuous-time systems are defined for all t, i.e. $u(t)$ and $y(t)$. Continuous linear dynamic systems are described by linear differential equations in the form of

$$\frac{d^n y(t)}{dt^n}+a_{n-1}\frac{d^{n-1}y(t)}{dt^{n-1}}+\cdots+a_0 y(t) = b_m\frac{d^m u(t)}{dt^m}+b_{m-1}\frac{d^{m-1}u(t)}{dt^{m-1}}+\cdots+b_0 u(t), \tag{2.1}$$

with the initial conditions $y(0) = y_0, \ \ldots, y^{(n)}(0) = y_0^{(n)}$, where $y^{(n)}$ is a shorthand of $\frac{d^n y}{dt^n}$.

Example 2.2.1 (Mass spring damper) Consider a mass spring damper system:



Newton's second law gives

$$m\ddot{y}(t) + b\dot{y}(t) + ky(t) = u(t), \ y(0) = y_0, \ \dot{y}(0) = \dot{y}_0. \qquad (2.2)$$

The system is modeled as a second-order ordinary differential equation (ODE) with input $u(t)$ and output $y(t)$.

2.3 Discrete-Time Systems

Inputs and outputs of discrete-time systems are defined at discrete-time points, i.e. $u(k)$ and $y(k)$, where $k = 0, 1, 2, \ldots$. Models of discrete dynamic systems are difference equations in the form of:

$$y(k+1) = f(y(k), y(k-1), \ldots, y(k-n), u(k), u(k-1), \ldots, u(k-n)),$$

where the output at $k+1$ depends on the input and output at $k, k-1, \ldots, k-n$.

Example 2.3.1 (Banking and Interest Rate) In a bank account, let $x(k)$ denote the beginning balance at the k-th month, and let $u(k)$ denote the accumulated deposit/credit or payment/debit during the k-th month). A discrete dynamic model to describe the balance at the beginning of every month is

$$x(k+1) = (1 + a(k))x(k) + u(k),$$

where $a(k)$ is the interest rate of the k-th month. This model is used for a variety of purposes, for example, to predict the balance after 12 months assuming the interest rate and the pattern of deposit during the period.

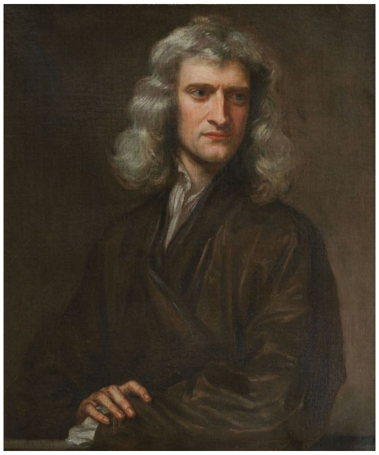


Figure 2.3: Isaac Newton (1642-1726) developed Newton's laws in 1686. He is an extremely brilliant scientist and in the meantime often known to be eccentric. He was described as "...so absorbed in his studies that he forgot to eat". https://en.wikipedia.org/wiki/Isaac_Newton#/media/File:Portrait_of_Sir_Isaac_Newton,_1689.jpg

2.4 Example: Atomic Force Microscopy

One of the most powerful techniques for imaging nanoscale objects is Atomic Force Microscopy (AFM). It can reveal the fine details of surfaces ranging from single molecules to the uneven texture of a glass pane. AFM has many applications in various fields of research, such as cell biology, semiconductors, thin film and coatings, tribology (surface and friction interactions), molecular biology, and energy storage and energy generation (photovoltaic) materials. The key to the high performance of AFM is the feedback control system that enables precise scanning at the nanometer level.

The AFM system has two working modes: the tapping mode and the contact mode. A contact mode AFM system that images a sample surface is shown in Figure 2.4. The system consists of a cantilever with an atomic-point needle that scans the sample's surface. The contact point follows the surface topology by moving up and down. A laser beam is directed at the cantilever and is reflected onto a photodiode that measures the beam's exact location. Based on this information, a control system can adjust the position of the cantilever (or the sample under it). The height of the point is recorded as the surface height at that location. A map of the surface can be created by combining the heights from the whole scan.

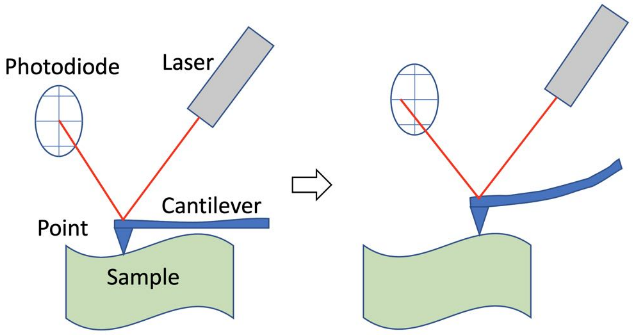


Figure 2.4: Schematic of the AFM.

The control system aims to maintain a constant force on the sample surface by the needle tip. The cantilever deflection changes with the force on the needle, and the photodiode can sense the laser movement. This information is fed into the control system, which adjusts the sample height to keep the cantilever deflection at the desired level.

The sample height is controlled by a piezoelectric stack actuator element (piezo) under the sample. Piezo elements are crystals that deform according to the electric charge they receive. This deformation is used to move the sample up or down in the z-direction, to regulate the force on the needle. Usually, piezoelectric stack actuators are also used to move the sample in the x and y directions, but since scanning is done at a constant speed and pattern, these piezo elements do not require the same precision and bandwidth in their feedback control as the z-direction piezo.

The needle encounters forces on the nanoscale that are not obvious to us. Forces such as the attractive van der Waals force, which pulls molecules together, and the repulsive Pauli force, which pushes molecules away, influence the needle as it moves across the surface, in addition to the reaction force from the surface. These forces are nonlinear. However, the major mode of AFM is a spring-mass-damper system. Reference [1] created two models, one second-order and one fourth-order, to describe the dynamics of the AFM system simply, and then more accurately.

[1]: Schitter et al. (2007), *Design and Modeling of a High-Speed AFM-Scanner*

Figure 2.5 shows the second-order system. Here, M_1 models the sample and the upper part of the piezo element, and has a combined mass of m_1. M_2 models the lower part of the piezo element and the mass of the supporting element beneath, with a combined mass of m_2. The piezo element resizes due to force F, which affects the two masses at the center of the piezo element. The supporting element has a spring constant of k_2 and a damping coefficient of b_2. The input to the system is force F, which is generated by a voltage signal that causes the piezo element to expand and contract.

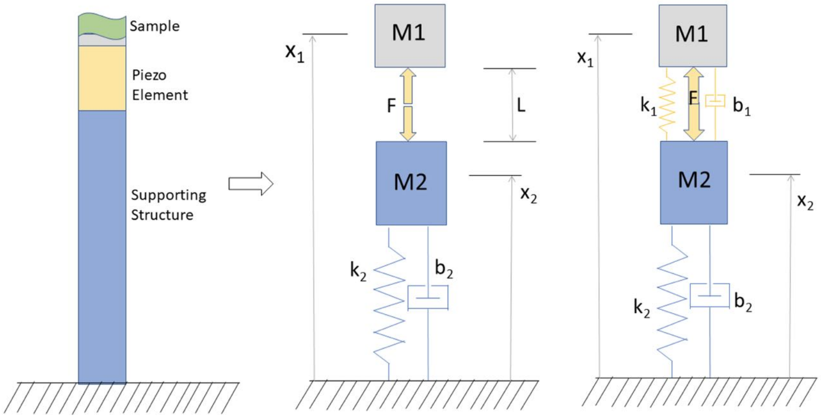


Figure 2.5: Mechanical models of an AFM.

Based on Newton's law, the governing equation of the second-order model of the AFM here is

$$\begin{aligned} m_1\frac{d^2x_1}{dt^2} &= F, \\ m_2\frac{d^2x_2}{dt^2} &= -b_2\frac{dx_2}{dt} - k_2x_2 - F, \\ l &= x_1 - x_2, \end{aligned} \tag{2.3}$$

where l is the distance between x_1 and x_2 as shown in Figure 2.5.

Fourth order is known to be the highest-order model for an AFM system that provides benefits in control design. Models with higher orders do not enhance the precision much, but instead will increase the computational cost considerably.

To account for the piezo element dynamics, we need to modify the system to a fourth-order model. This is done by adding another spring-damper component. The final schematic model in Figure 2.5 shows k_1 as the effective spring constant and b_1 as the effective damping coefficient of the piezo element. Force F still affects both masses. The fourth-order model includes the dynamics of the second-order model and also captures high-order dynamics observed in the frequency response of a real-world AFM system [2].

[2]: Schitter et al. (2001), *High performance feedback for fast scanning atomic force microscopes*

Using first principles in Figure 2.5, when spring and damping effects are considered between the two masses, the governing equations become:

$$\begin{aligned} m_1\frac{d^2x_1}{dt^2} &= -b_1\left(\frac{dx_1}{dt} - \frac{dx_2}{dt}\right) - k_1(x_1 - x_2) + F, \\ m_2\frac{d^2x_2}{dt^2} &= b_1\left(\frac{dx_1}{dt} - \frac{dx_2}{dt}\right) + k_1(x_1 - x_2) - b_2\left(\frac{dx_2}{dt} - 0\right) - k_2x_2 - F. \end{aligned} \tag{2.4}$$

2.5 Example: Hard Disk Drive and Information Storage

Hard disk drives (HDDs) are amazing mechanical systems that store and retrieve digital data using magnetic storage. They consist of one or more rigid rapidly rotating platters coated with magnetic material, and a read-write head that moves across the platter surface to access the data. HDDs were the standard storage system for personal computers for over 30 years and have been the main storage element in data centers. Their history goes back to the 1950s when IBM invented the first HDD. Since then, HDDs have undergone tremendous improvements in terms of capacity, speed, size, power consumption and reliability.

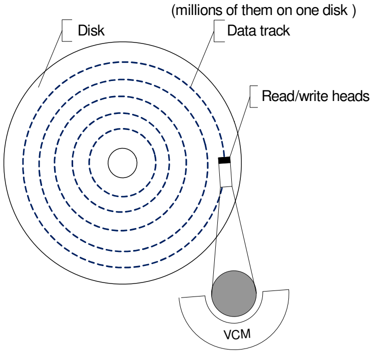


Figure 2.6: Control-related components in a hard disk drive.

HDDs are examples of high-precision engineering, as they operate at nanometer scales and millisecond speeds, while being mass-produced and affordable. HDDs are also versatile, as they can store any kind of digital data, from text and images to audio and video. They are truly remarkable devices that have revolutionized the field of data storage and enabled the development of modern computing.

Read more about the HDD mechatronics at, e.g.,

1. Hard disk drive - Wikipedia. https://en.wikipedia.org/wiki/Hard_disk_drive.
2. Anatomy of a Storage Drive: Hard Disk Drives | TechSpot. https://www.techspot.com/article/1984-anatomy-hard-drive/ Accessed 6/7/2023.
3. Hard disk | Definition & Facts | Britannica. https://www.britannica.com/technology/hard-disk Accessed 6/7/2023.

In a modern HDD, data is stored in circular patterns of magnetization known as data tracks or simply, tracks (Figure 2.6). During reading and writing of the data, the disk spins and the read/write head is controlled to follow the circular tracks. This creates the track-following problem, where the servo system performs regulation control to position the read/write head at the desired track, with as low variance as possible. During track following, the position errors are measured periodically at servo sectors that are embedded uniformly over one period of rotation of the disk. Suppose a disk has a rotational speed of 7200 revolutions per minute (rpm) and the number of servo sectors are 220. Then at every revolution of the disks, 220 measurements are obtained, at a sampling frequency of $220 \times 7200/60\,(= 26,400)$ Hz.

The actuator in a single-stage HDD is powered by a voice coil motor (VCM). The dynamics between the input current to the voice coil motor and the output position error signal of the read/write head include the effects of inertia, damping, spring constant, and resonant modes of the head assembly. In Section 2.1, we have seen that by Newton's law, the dynamics again has a nominal response of a double integrator. Sometimes, the nominal model also considers friction effects and is written as a second-order damped system instead of a pure double integrator. In high precision control, multiple high-frequency modes are typically present due to structural resonances (Figure 2.7) and the full-order model of the VCM system is in the form of:

$$G(s) = K \sum_{i=1}^{n} \frac{\kappa_i}{s^2 + 2\zeta_i \omega_i s + \omega_i^2}. \tag{2.5}$$

The example is based on: Takenori Atsumi (2023), Magnetic-head positioning control system in HDDs (https://www.mathworks.com/matlabcentral/fileexchange/111515-magnetic-head-positioning-\control-system-in-hdds), MATLAB Central File Exchange. Retrieved June 9, 2023.

The following codes establish a VCM model of a 7200 rpm HDD with 420 servo sectors.

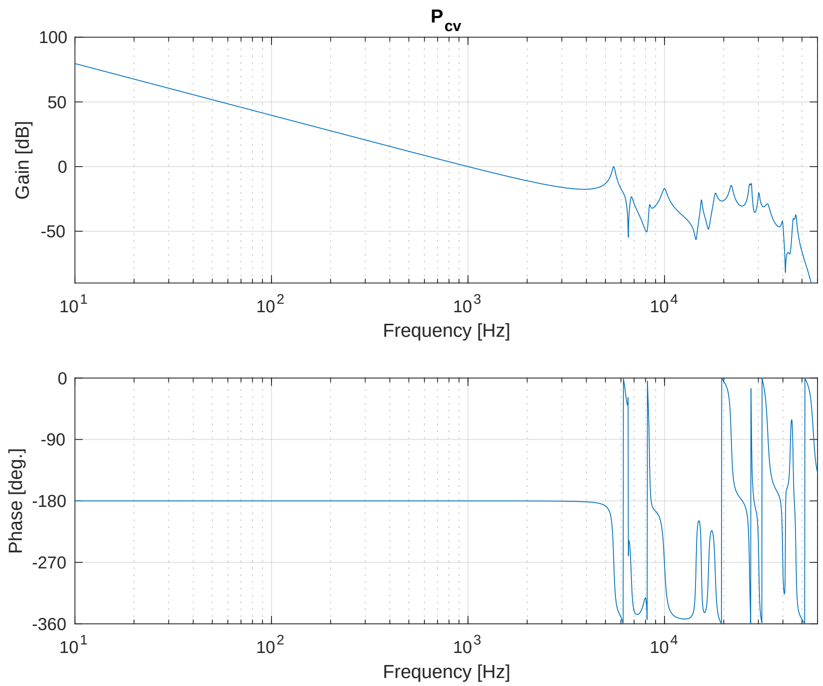


Figure 2.7: Baseline model of the VCM component in an HDD.

```
% modeling/hddvcm.m
% MATLAB code to generate a single-stage HDD model
num_sector=420;                      % Number of sector
num_rpm=7200;                        % Number of RPM
Ts = 1/(num_rpm/60*num_sector);      % Sampling time

% VCM
Kp_vcm=3.7976e+07;                   % VCM gain
omega_vcm=[0, 5300 ,6100 ,6500 ,8050 ,9600 ,14800 ,17400 ,21000 ,26000
↪  ,26600 ,29000 ,32200 ,38300 ,43300 ,44800]*2*pi;
kappa_vcm=[1, -1.0 ,+0.1 ,-0.1 ,0.04 ,-0.7 ,-0.2  ,-1.0  ,+3.0  ,-3.2
↪  ,2.1   ,-1.5  ,+2.0  ,-0.2  ,+0.3  ,-0.5 ];
zeta_vcm =[0, 0.02 ,0.04 ,0.02 ,0.01 ,0.03 ,0.01  ,0.02  ,0.02  ,0.012
↪  ,0.007 ,0.01  ,0.03  ,0.01  ,0.01  ,0.01 ];

Sys_Pc_vcm_c1=0;
for i=1:length(omega_vcm)
        Sys_Pc_vcm_c1=Sys_Pc_vcm_c1+tf([0,0,kappa_vcm(i)]*Kp_vcm,[1,
        ↪  2*zeta_vcm(i)*omega_vcm(i), (omega_vcm(i))^2]);
end

%% Frequency response
f=logspace(1,log10(60e3),3000);
Fr_Pc_vcm_c1=squeeze(freqresp(Sys_Pc_vcm_c1,f*2*pi)).';

figure
subplot(211)
semilogx(f,20*log10(abs(Fr_Pc_vcm_c1)))
title('P_{cv}');xlabel('Frequency [Hz]');ylabel('Gain
↪  [dB]');grid;axis([1e1 f(end) -90 100])
subplot(212)
semilogx(f,mod(angle(Fr_Pc_vcm_c1)*180/pi+360,360)-360)
xlabel('Frequency [Hz]');ylabel('Phase [deg.]');grid;axis([1e1 f(end)
↪  -360 0]);yticks(-360:90:0)
% if you want, you can save the images as follows:
```

```
% saveas(gcf,'images/hdd_pcvm_baseline.pdf')
% saveas(gcf,'images/hdd_pcvm_baseline.png')
```

The Python version of the code is as follows:

```
# modeling/hddvcm.py
import numpy as np
import matplotlib.pyplot as plt
from scipy import signal
import control as ct

num_sector = 420  # Number of sector
num_rpm = 7200  # Number of RPM
Ts = 1 / (num_rpm / 60 * num_sector)  # Sampling time

# VCM
Kp_vcm = 3.7976e+07  # VCM gain
omega_vcm = np.array([0, 5300, 6100, 6500, 8050, 9600, 14800, 17400,
                     21000, 26000, 26600, 29000, 32200, 38300, 43300,
                     ↪  44800]) * 2 * np.pi
kappa_vcm = np.array([1, -1.0, +0.1, -0.1, 0.04, -0.7, -
                    0.2, -1.0, +3.0, -3.2, 2.1, -1.5, +2.0, -0.2, +0.3,
                     ↪  -0.5])
zeta_vcm = np.array([0, 0.02, 0.04, 0.02, 0.01, 0.03, 0.01,
                    0.02, 0.02, 0.012, 0.007, 0.01, 0.03, 0.01, 0.01,
                    ↪  0.01])

Sys_Pc_vcm_c1 = ct.TransferFunction(
    [], [1])  # Create an empty transfer function
for i in range(len(omega_vcm)):
    Sys_Pc_vcm_c1 = Sys_Pc_vcm_c1 + ct.TransferFunction(np.array(
        [0, 0, kappa_vcm[i]]) * Kp_vcm, np.array([1, 2 * zeta_vcm[i] *
        ↪  omega_vcm[i], (omega_vcm[i]) ** 2]))

# Frequency response
f = np.logspace(1, np.log10(60e3), 3000)
w = f * 2 * np.pi
magPc_vcm, phase_Pc_vcm, omega_Pc_vcm = ct.freqresp(
    Sys_Pc_vcm_c1, w)  # Get the frequency response

plt.figure()
plt.subplot(211)
plt.semilogx(f, 20*np.log10(magPc_vcm))
plt.title('$P_{cv}$')
plt.xlabel('Frequency [Hz]')
plt.ylabel('Gain [dB]')
plt.grid()
plt.axis([10, f[-1], -90, 100])
plt.subplot(212)
plt.semilogx(f, np.mod(phase_Pc_vcm*180/np.pi+360, 360)-360)
plt.xlabel('Frequency [Hz]')
plt.ylabel('Phase [deg.]')
plt.grid()
plt.axis([10, f[-1], -360, 0])
plt.yticks(np.arange(-360, 90, 90))
```

With the ever increasing demand of larger capacity in HDDs, dual-stage actuation using both a VCM and a piezoelectric actuator has become an essential technique to break the bottleneck of the servo performance in single-actuator HDDs. A dual-stage HDD applies an additional piezoelectric microactuator (MA) at the end of the VCM actuator, as shown in Figure

2.8. We have discussed how piezoelectric microactuators are useful for high-precision control in AFMs. The MA stage has much smaller moving range but greatly improved positioning speed and accuracy. Its dynamical response is also much easier to control, with the low-frequency dynamics governed simply by a DC gain (Figure 2.9). Compared to the VCM actuator, the MA has enhanced mechanical performance in the high-frequency region, providing the capacity to greatly increase the servo bandwidth and disturbance-attenuation capacity.

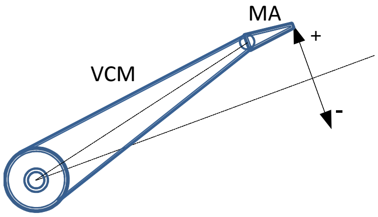


Figure 2.8: Schematic structure of dual-stage HDDs.

Only the position error of the read/write head is measurable in practice. The plant is hence a dual-input-single-output system.

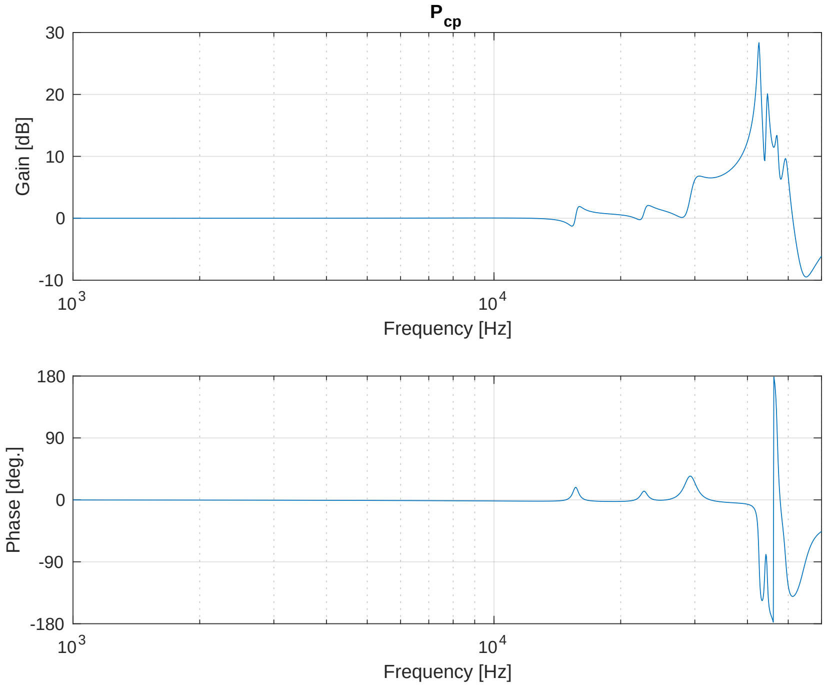


Figure 2.9: Baseline model of the PZT component in an HDD.

The following codes establish a normalized MA model of a 7200 rpm HDD with 420 servo sectors.

```
% modeling/hddpzt.m
% MATLAB code to generate the pzt-stage HDD model
num_sector=420;                    % Number of sector
num_rpm=7200;                      % Number of RPM
Ts = 1/(num_rpm/60*num_sector);    % Sampling time

% PZT
omega_pzt=[14800 ,21500 ,28000 ,40200 ,42050,44400,46500
 ↪  ,100000]*2*pi;
kappa_pzt=[-0.005,-0.01 ,-0.1  ,+0.8  ,0.3  ,-0.25  ,0.3  ,10.0 ];
zeta_pzt =[0.025 ,0.03  ,0.05  ,0.008  ,0.008 ,0.01 ,0.02  ,0.3 ];

Sys_Pc_pzt_c1=0;
for i=1:length(omega_pzt)
        Sys_Pc_pzt_c1=Sys_Pc_pzt_c1+tf([0,0,kappa_pzt(i)],[1,
        ↪  2*zeta_pzt(i)*omega_pzt(i), (omega_pzt(i))^2]);
end
Sys_Pc_pzt_c1=Sys_Pc_pzt_c1/abs(freqresp(Sys_Pc_pzt_c1,0));
```

```
%% Frequency response
f=logspace(1,log10(60e3),3000);
Fr_Pc_pzt_c1=squeeze(freqresp(Sys_Pc_pzt_c1,f*2*pi)).';

figure
subplot(211)
semilogx(f,20*log10(abs(Fr_Pc_pzt_c1)))
title('P_{cp}');xlabel('Frequency [Hz]');ylabel('Gain
↪  [dB]');grid;axis([1e3 f(end) -10 30])
subplot(212)
semilogx(f,angle(Fr_Pc_pzt_c1)*180/pi)
xlabel('Frequency [Hz]');ylabel('Phase [deg.]');grid;axis([1e3 f(end)
↪  -180 180]);yticks(-180:90:180)
```

Here is the model construction in Python:

```
# modeling/hddpzt.py
import numpy as np
import matplotlib.pyplot as plt
import control as ct

num_sector = 420  # Number of sector
num_rpm = 7200  # Number of RPM
Ts = 1 / (num_rpm / 60 * num_sector)  # Sampling time

# PZT
omega_pzt = np.array([14800, 21500, 28000, 40200, 42050,
                     44400, 46500, 100000]) * 2 * np.pi
kappa_pzt = np.array([-0.005, -0.01, -0.1, +0.8, 0.3, -0.25, 0.3,
↪  10.0])
zeta_pzt = np.array([0.025, 0.03, 0.05, 0.008, 0.008, 0.01, 0.02,
↪  0.3])

s = ct.TransferFunction.s  # Create a variable for the differentiation
↪  operator
Sys_Pc_pzt_c1 = 0  # Create an empty transfer function
for i in range(len(omega_pzt)):
    Sys_Pc_pzt_c1 += kappa_pzt[i] / (s**2 + 2 * zeta_pzt[i] *
    ↪  omega_pzt[i]
                                    * s + (omega_pzt[i]) ** 2)  # Add
                                    ↪  the transfer functions
Sys_Pc_pzt_c1 /= Sys_Pc_pzt_c1(0)  # Normalize the gain at zero
↪  frequency

# Frequency response
f = np.logspace(1, np.log10(60e3), 3000)
w = f * 2 * np.pi

magPc_pzt, phase_Pc_pzt, omega_Pc_pzt = ct.freqresp(
    Sys_Pc_pzt_c1, w)  # Get the frequency response

plt.figure()
plt.subplot(211)
plt.semilogx(f, 20*np.log10(magPc_pzt))
plt.title('$P_{cp}$')
plt.xlabel('Frequency [Hz]')
plt.ylabel('Gain [dB]')
plt.grid()
plt.axis([1000, f[-1], - 10, 30])
plt.subplot(212)
plt.semilogx(f, phase_Pc_pzt*180/np.pi)
plt.xlabel('Frequency [Hz]')
plt.ylabel('Phase [deg.]')
plt.grid()
plt.axis([1000, f[-1], - 180, 180])
plt.yticks(np.arange(-180, 270, 90))
```

With the VCM and PZT models, a dual-stage HDD model can be formed as follows.

```
% MATLAB:
Sys_Pc = [Sys_Pc_vcm_c1; Sys_Pc_pzt_c1];
```

```
# Python
Sys_Pc = ct.append(Sys_Pc_vcm_c1, Sys_Pc_pzt_c1)
```

2.6 Model Properties

We now formalize key properties of system models for control purpose. Consider a general system $\mathcal{M}$ with input $u(\sigma)$ ($\sigma = t$ or k depending on whether the signal is in the continuous- or discrete-time domain) and output $y(\sigma)$:

$$u \longrightarrow \boxed{\mathcal{M}} \longrightarrow y$$

$\mathcal{M}$ is said to be

- *memoryless* or *static* if $y(\sigma)$ depends only on $u(\sigma)$,
- *dynamic* (has memory) if y at the current time depends on input values at other times.

For example, $y(t) = \mathcal{M}(u(t)) = \gamma u(t)$ is memoryless; $y(t) = \int_0^t u(\tau)d\tau$ and $y(k) = \sum_{i=0}^k u(i)$ are dynamic. The system $\mathcal{M}$ is *linear* if it satisfies the *superposition* property:

$$\mathcal{M}(\alpha_1 u_1(\sigma) + \alpha_2 u_2(\sigma)) = \alpha_1 \mathcal{M}(u_1(\sigma)) + \alpha_2 \mathcal{M}(u_2(\sigma))$$

for any input signals $u_1(\sigma)$ and $u_2(\sigma)$, and any real numbers α_1 and α_2. If not, the system is nonlinear.

$\mathcal{M}$ is *time-invariant* if its properties do not change with respect to time. Assuming the same initial conditions, if we shift $u(\sigma)$ by a constant time interval, then $\mathcal{M}$ is time-invariant if the output $\mathcal{M}(u(\sigma+\tau_0)) = y(\sigma+\tau_0)$. For example, $\dot{y}(t) = Ay(t) + Bu(t)$ is linear and time-invariant; $\dot{y}(t) = 2y(t) - \sin(y(t))u(t)$ is nonlinear, yet time-invariant; $\dot{y}(t) = 2y(t) - t\sin(y(t))u(t)$ is time-varying. Often, we abbreviate linear time-invariant systems as LTI systems.

In mechanical systems, torque limits of motors, hardening spring, Coulomb friction forces, etc. make systems nonlinear. However, even if physical systems are nonlinear, they can often be linearized or be well represented by linear systems under specific conditions, making linear analysis and design tools powerful.

The system $\mathcal{M}$ is called

- ► *causal* if $y(t)$ or $y(k)$ depends on $u(\tau)$ for $\tau \leq t$ or k,
- ► *strictly causal* if the inequality is strict.

For example, $y(t) = u(t - 10)$ is strictly causal.

A system that is not causal is said to be acausal.

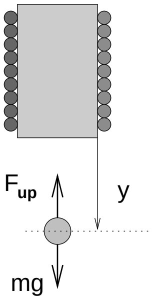
For an LTI continuous-time system to be causal, the order of the denominator must be greater than or equal to the order of the numerator in its transfer function. For example, for the model in Equation 2.1 to be causal, it must be that $n \geq m$. (Check, e.g., the case with $n = 0$ and $m = 1$.)

2.7 Nonlinear Systems

Although we will be mostly focusing on linear systems, in practice, many control systems are nonlinear. Several examples are presented next to show how the nonlinearity plays a role in modeling.

2.7.1 Example: Magnetically Suspended Ball

A magnetically suspended ball consists of a ball of mass m suspended by an electromagnet as shown below.



Let y be the position of the ball, measured down from the base of the electromagnet. If a current u is injected into the coils of the electromagnet, it will produce a magnetic field which in turn exerts an upward force on the ball given by $F_{up} = -\frac{cu^2}{y^2}$. Note that the force decreases as y^2 increases because the effect of the magnet weakens when the ball is further away, and that the force is proportional to u^2 which is representative of the power supplied to the magnet.

Let us assume that we can measure the position y of the ball. This can be arranged optically using light-emitting diodes with photocells.

We can then write a simple model for the motion of the ball from Newton's second law as

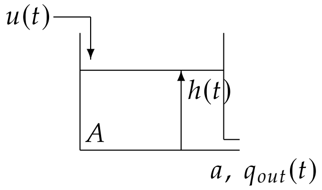
$$m\ddot{y} = mg - \frac{cu^2}{y^2}. \tag{2.6}$$

Note that this is a single-input, single-output nonlinear model. The input and output of this model are:

u	the current injected into the magnet coils
y	the position of the ball

2.7.2 Example: Water Tank

Consider the water tank in the following figure:



Denote by $q_{in}(t)$ the water flow entering the tank and $q_{out}(t)$ the water flow leaving the tank from a hole of area a. Out goal is to obtain a model that describes the evolution of the tank height $h(t)$ as a function of the external input $u(t) = q_{in}(t)$. Let A be the area of the tank. By using the conservation law we can state that

$$A\dot{h}(t) = q_{in}(t) - q_{out}(t).$$

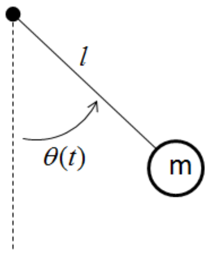
Let $v_{out}(t)$ be the speed of the water at the outlet. Then, $q_{out}(t) = av_{out}(t)$.

Torricelli's law states that the speed of a fluid through a sharp-edged hole under the force of gravity is the same as the speed that a body would acquire in falling freely from a height h, i.e. $v_{out}(t) = \sqrt{2gh(t)}$, where g is the acceleration due to gravity. Considering all the above, our *nonlinear* model is

$$\dot{h} = \frac{1}{A}(u(t) - a\sqrt{2gh(t)}).$$

2.7.3 Example: Pendulum

Consider the pendulum shown in the following figure.



We assume that the pendulum has mass $m = 0.333$ kg which is concentrated at the end point and length $l = 0.75$ meters. The angle θ is measured, and an actuator can supply a torque $u(t) = T_c(t)$ to the pivot point. The moment of inertia about the pivot point is $I = ml^2$. By analyzing the rigid body dynamics and writing Euler's equation for the pendulum, we can readily arrive at a differential equation model for this system:

$$I\ddot{\theta}(t) = T_c - mgl\sin(\theta(t)),$$

or,

$$\ddot{\theta}(t) = \frac{1}{ml^2}T_c - \frac{g}{l}\sin(\theta(t)),$$

which is a second-order nonlinear differential equation.

2.7.4 Example: Vehicle Steering

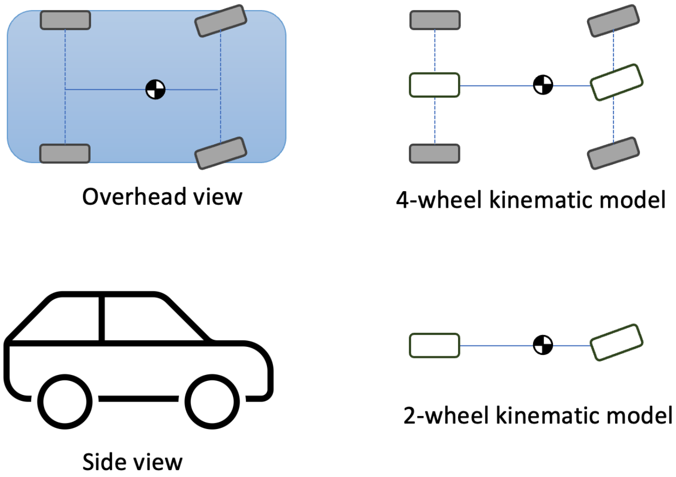


Figure 2.10: Kinematic modelling of the four-wheel vehicle steering system.

Figure 2.10 shows the kinematics of a four-wheeled vehicle. If we assume the front and rear pairs of wheels act the same as a single wheel on each axle at the centerline of the vehicle, we arrive at a two-wheel abstraction known as the bicycle model. The bicycle model does not take into account tilting of the vehicle. A four-wheeled vehicle would experience very different tilting mechanics than those of a two-wheeled vehicle. However, when only the steering of the vehicle is of concern with the assumption that it cannot tilt, such as in vehicles with a low center of gravity and operating at slow enough speeds, the bicycle model is appropriate.

Read more about the bicycle model at, e.g., https://thef1clan.com/2020/09/21/vehicle-dynamics-the-kinematic-bicycle-model/

When the vehicle turns, the inner and outer tires exhibit different steering angles due to their varying distances from point O in the bicycle model in Figure 2.11. Here, δ is the steering angle of the front wheels, b is the wheel base, a is the distance between the rear axle and the center of mass, x and y are the positions of the center of mass, θ is the heading, and α is the angle between the velocity v and the centerline of the vehicle. The point O is the intersection of the centerlines of the of the front and rear wheels. By letting the distance from the center of rotation O to the rear wheel contact point be r_r , we can deduce that $b = r_r \tan\delta$ and $a = r_r \tan\alpha$. This leads to the relationship between α and the steering angle δ:

$$\alpha = \arctan\left(\frac{a\tan\delta}{b}\right). \tag{2.7}$$

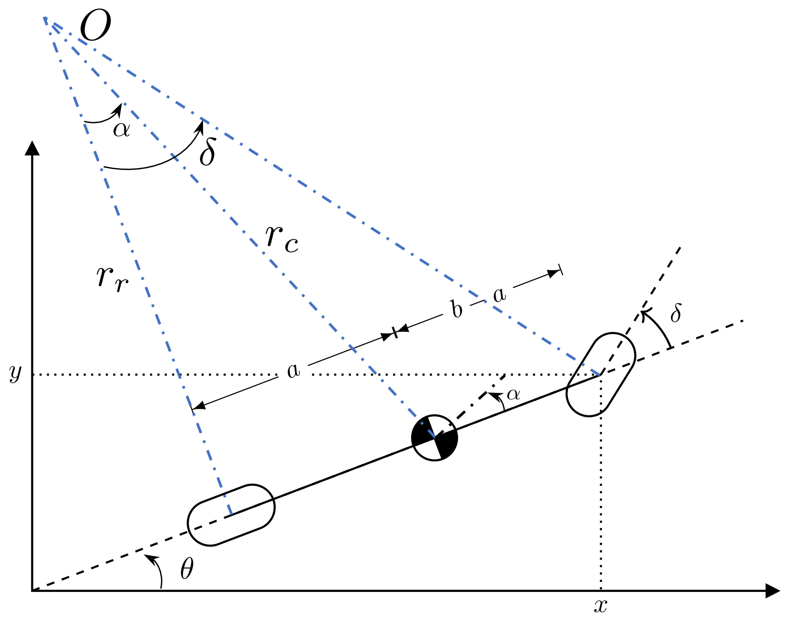


Figure 2.11: The bicycle model of the vehicle steering system.

Given a vehicle speed v at its center of mass, the motion of the center of mass is expressed as:

$$\begin{aligned}\frac{dx}{dt} &= v\cos(\alpha+\theta),\\ \frac{dy}{dt} &= v\sin(\alpha+\theta).\end{aligned} \tag{2.8}$$

To determine the influence of the steering angle on the heading angle θ, we note that the distance from the center of mass to the center of rotation O is $r_c = a/\sin\alpha$. As the vehicle rotates around point O, the angular velocity is given by $v/r_c = (v/a)\sin\alpha$. Therefore,

$$\dot{\theta} = \frac{v}{r_c} = \frac{v\sin\alpha}{a} = \frac{v}{a}\sin\left(\arctan\left(\frac{a\tan\delta}{b}\right)\right). \tag{2.9}$$

When the steering angle δ and the angle α (known as the slip angle) is small, the above equation becomes

$$\dot{\theta} \approx \frac{v}{b}\delta. \tag{2.10}$$

Let the input u be the steering angle δ. The full set of nonlinear equations of motion for the vehicle steering problem is now:

$$\frac{d}{dt}\underbrace{\begin{bmatrix}x\\ y\\ \theta\end{bmatrix}}_{\mathbf{x}} = f(\mathbf{x},u) = \begin{bmatrix}v\cos\left(\alpha\left(u\right)+\theta\right)\\ v\sin\left(\alpha\left(u\right)+\theta\right)\\ \frac{v\sin\alpha(u)}{a}\end{bmatrix},\ \alpha\left(u\right) = \arctan\frac{a\tan u}{b}. \tag{2.11}$$

2.8 "All Models are Wrong, but Some are Useful"

A perfect model is very hard to be obtained in reality. In the Journal of the American Statistical Association [3], British statistician George Box famously wrote in 1976, that "all models are wrong, but some are useful." The aphorism acknowledges that statistical models always fall short of the complexities of reality but can still be useful nonetheless. A dynamic system may simply be too complex (consider the neural system of human brains), or there are inevitable hardware uncertainties such as the fatigue of gears or bearings in a car. Modeling thus involves varying degrees of approximation. For example in modeling a car, we may chose to ignore rolling resistance, aero-dynamic effects, road-tire interactions, etc.

[3]: Box (1976), *Science and statistics*

However, feedback control can empower a system to tolerate model uncertainties. In a 5-year-old car, the same engine control unit (ECU) can maintain smooth performance of the internal combustion engine despite the accumulated mileage and wear, whether they come from highways or mountains. In the hard disk drive example in Section 2.1, the high-frequency vibration modes can change their shapes and locations in a batch of products and when working under different temperatures, leading to the frequency responses in Figure 2.12 [4]. However, thanks to the robustness from feedback controls, when making thousands of hard disk drives per day, the manufactures do not need to tune the controller in each individual drive. Instead, each batch can robustly read and write data using the same servo controller.

Robust control is not tied to complex control. A PID controller can create robust stability too.

[4]: Atsumi (2022), *Magnetic-head Positioning Control System in HDDs*

The subfield of robust control covers details of why and how to handle the model uncertainties.

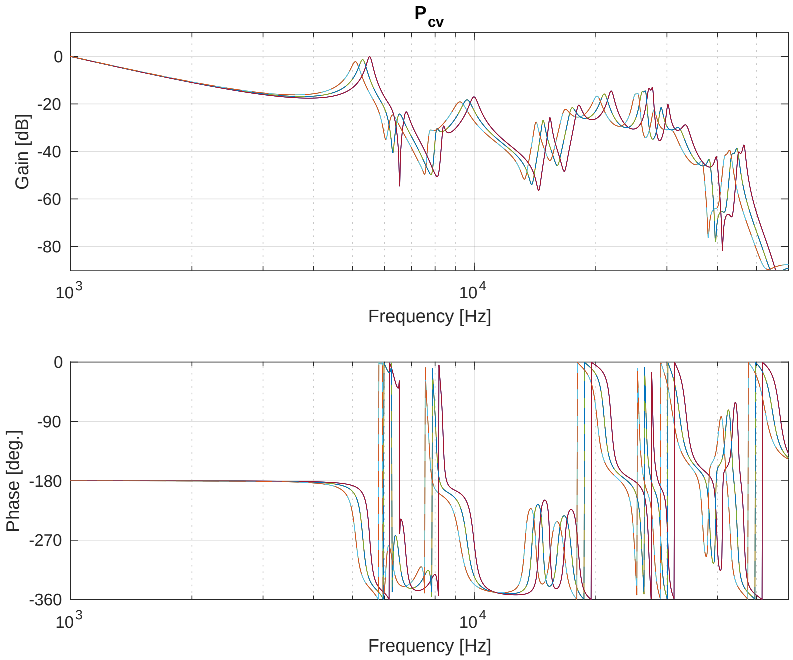


Figure 2.12: Example frequency responses of the voice coil motor stage in a batch of hard disk drives.

The following MATLAB codes generate the perturbed HDD model in Figure 2.12.

```
% modeling/hdddsa.m
% Dual-stage HDD model
num_sector=420;                   % Number of sector
num_rpm=7200;                     % Number of RPM
Ts = 1/(num_rpm/60*num_sector);   % Sampling time

% VCM
Kp_vcm=3.7976e+07;
omega_vcm=[0, 5300 ,6100 ,6500 ,8050 ,9600 ,14800 ,17400 ,21000 ,26000
↪   ,26600 ,29000 ,32200 ,38300 ,43300 ,44800]*2*pi;
kappa_vcm=[1, -1.0 ,+0.1 ,-0.1 ,0.04 ,-0.7 ,-0.2  ,-1.0  ,+3.0  ,-3.2
↪   ,2.1   ,-1.5  ,+2.0  ,-0.2  ,+0.3  ,-0.5 ];
zeta_vcm =[0, 0.02 ,0.04 ,0.02 ,0.01 ,0.03 ,0.01  ,0.02  ,0.02  ,0.012
↪   ,0.007 ,0.01  ,0.03  ,0.01  ,0.01  ,0.01 ];

% PZT
omega_pzt=[14800 ,21500 ,28000 ,40200 ,42050,44400,46500
↪   ,100000]*2*pi;
kappa_pzt=[-0.005,-0.01 ,-0.1  ,+0.8  ,0.3  ,-0.25  ,0.3  ,10.0 ];
zeta_pzt =[0.025 ,0.03  ,0.05  ,0.008  ,0.008 ,0.01 ,0.02  ,0.3 ];

%% LT(Low temp.) model: VCM: +4 % resonance shift from nominal
↪   model, PZT actuator: +6 % resonance shift from nominal
↪   model
% VCM
Sys_Pc_vcm_c1=0;
for i=1:length(omega_vcm)
        Sys_Pc_vcm_c1=Sys_Pc_vcm_c1+ss(tf([0,0,kappa_vcm(i)]*Kp_-
        ↪   vcm,[1, 2*zeta_vcm(i)*0.8*omega_vcm(i)*1.04,
        ↪   (omega_vcm(i)*1.04)^2]));
        Sys_Pc_vcm_c1=ssbal(Sys_Pc_vcm_c1);
end

% PZT
Sys_Pc_pzt_c1=0;
for i=1:length(omega_pzt)
        Sys_Pc_pzt_c1=Sys_Pc_pzt_c1+ss(tf([0,0,kappa_pzt(i)],[1,
        ↪   2*zeta_pzt(i)*0.8*omega_pzt(i)*1.06,
        ↪   (omega_pzt(i)*1.06)^2]));
        Sys_Pc_pzt_c1=ssbal(Sys_Pc_pzt_c1);
end
Sys_Pc_pzt_c1=Sys_Pc_pzt_c1/abs(freqresp(Sys_Pc_pzt_c1,0));

%% RT(Room temp.) model: Same as nominal models
% VCM
Sys_Pc_vcm_c2=0;
for i=1:length(omega_vcm)
        Sys_Pc_vcm_c2=Sys_Pc_vcm_c2+ss(tf([0,0,kappa_vcm(i)]*Kp_-
        ↪   vcm,[1, 2*zeta_vcm(i)*omega_vcm(i),
        ↪   omega_vcm(i)^2]));
        Sys_Pc_vcm_c2=ssbal(Sys_Pc_vcm_c2);
end

% PZT
Sys_Pc_pzt_c2=0;
for i=1:length(omega_pzt)
        Sys_Pc_pzt_c2=Sys_Pc_pzt_c2+ss(tf([0,0,kappa_pzt(i)],[1,
        ↪   2*zeta_pzt(i)*omega_pzt(i), omega_pzt(i)^2]));
        Sys_Pc_pzt_c2=ssbal(Sys_Pc_pzt_c2);
end
Sys_Pc_pzt_c2=Sys_Pc_pzt_c2/abs(freqresp(Sys_Pc_pzt_c2,0));
```

```
%% HT(High temp.) model: VCM: -4 % resonance shift from nominal
↪  model, PZT actuator: -6 % resonance shift from nominal
↪  model
% VCM
Sys_Pc_vcm_c3=0;
for i=1:length(omega_vcm)
        Sys_Pc_vcm_c3=Sys_Pc_vcm_c3+ss(tf([0,0,kappa_vcm(i)]*Kp_-
        ↪  vcm,[1, 2*zeta_vcm(i)*1.2*omega_vcm(i)*0.96,
        ↪  (omega_vcm(i)*0.96)^2]));
        Sys_Pc_vcm_c3=ssbal(Sys_Pc_vcm_c3);
end

% PZT
Sys_Pc_pzt_c3=0;
for i=1:length(omega_pzt)
        Sys_Pc_pzt_c3=Sys_Pc_pzt_c3+ss(tf([0,0,kappa_pzt(i)],[1,
        ↪  2*zeta_pzt(i)*1.2*omega_pzt(i)*0.94,
        ↪  (omega_pzt(i)*0.94)^2]));
        Sys_Pc_pzt_c3=ssbal(Sys_Pc_pzt_c3);
end
Sys_Pc_pzt_c3=Sys_Pc_pzt_c3/abs(freqresp(Sys_Pc_pzt_c3,0));

%% LT / PZT gain +5% (Case 4)
Sys_Pc_vcm_c4=Sys_Pc_vcm_c1;
Sys_Pc_pzt_c4=Sys_Pc_pzt_c1*1.05;

%% RT / PZT gain +5% (Case 5)
Sys_Pc_vcm_c5=Sys_Pc_vcm_c2;
Sys_Pc_pzt_c5=Sys_Pc_pzt_c2*1.05;

%% HT / PZT gain +5% (Case 6)
Sys_Pc_vcm_c6=Sys_Pc_vcm_c3;
Sys_Pc_pzt_c6=Sys_Pc_pzt_c3*1.05;

%% LT / PZT gain -5% (Case 7)
Sys_Pc_vcm_c7=Sys_Pc_vcm_c1;
Sys_Pc_pzt_c7=Sys_Pc_pzt_c1*0.95;

%% RT / PZT gain -5% (Case 8)
Sys_Pc_vcm_c8=Sys_Pc_vcm_c2;
Sys_Pc_pzt_c8=Sys_Pc_pzt_c2*0.95;

%% HT / PZT gain -5% (Case 9)
Sys_Pc_vcm_c9=Sys_Pc_vcm_c3;
Sys_Pc_pzt_c9=Sys_Pc_pzt_c3*0.95;

%% All plant
Sys_Pc_vcm_all=[Sys_Pc_vcm_c1;Sys_Pc_vcm_c2;Sys_Pc_vcm_c3;Sys_Pc_vcm_-
↪  c4;Sys_Pc_vcm_c5;Sys_Pc_vcm_c6;Sys_Pc_vcm_c7;Sys_Pc_vcm_c8;Sys_-
↪  Pc_vcm_c9];
Sys_Pc_pzt_all=[Sys_Pc_pzt_c1;Sys_Pc_pzt_c2;Sys_Pc_pzt_c3;Sys_Pc_pzt_-
↪  c4;Sys_Pc_pzt_c5;Sys_Pc_pzt_c6;Sys_Pc_pzt_c7;Sys_Pc_pzt_c8;Sys_-
↪  Pc_pzt_c9];

%% Cotrolled object (Discrete-time system)
% Case 1
Sys_Pd_vcm_c1=c2d(Sys_Pc_vcm_c1,Ts,'ZOH');
Sys_Pd_pzt_c1=c2d(Sys_Pc_pzt_c1,Ts,'ZOH');

% Case 2
Sys_Pd_vcm_c2=c2d(Sys_Pc_vcm_c2,Ts,'ZOH');
Sys_Pd_pzt_c2=c2d(Sys_Pc_pzt_c2,Ts,'ZOH');
```

```
% Case 3
Sys_Pd_vcm_c3=c2d(Sys_Pc_vcm_c3,Ts,'ZOH');
Sys_Pd_pzt_c3=c2d(Sys_Pc_pzt_c3,Ts,'ZOH');

% Case4
Sys_Pd_vcm_c4=c2d(Sys_Pc_vcm_c4,Ts,'ZOH');
Sys_Pd_pzt_c4=c2d(Sys_Pc_pzt_c4,Ts,'ZOH');

% Case 5
Sys_Pd_vcm_c5=c2d(Sys_Pc_vcm_c5,Ts,'ZOH');
Sys_Pd_pzt_c5=c2d(Sys_Pc_pzt_c5,Ts,'ZOH');

% Case 6
Sys_Pd_vcm_c6=c2d(Sys_Pc_vcm_c6,Ts,'ZOH');
Sys_Pd_pzt_c6=c2d(Sys_Pc_pzt_c6,Ts,'ZOH');

% Case 7
Sys_Pd_vcm_c7=c2d(Sys_Pc_vcm_c7,Ts,'ZOH');
Sys_Pd_pzt_c7=c2d(Sys_Pc_pzt_c7,Ts,'ZOH');

% Case 8
Sys_Pd_vcm_c8=c2d(Sys_Pc_vcm_c8,Ts,'ZOH');
Sys_Pd_pzt_c8=c2d(Sys_Pc_pzt_c8,Ts,'ZOH');

% Case 9
Sys_Pd_vcm_c9=c2d(Sys_Pc_vcm_c9,Ts,'ZOH');
Sys_Pd_pzt_c9=c2d(Sys_Pc_pzt_c9,Ts,'ZOH');

% All
Sys_Pd_vcm_all=[Sys_Pd_vcm_c1;Sys_Pd_vcm_c2;Sys_Pd_vcm_c3;Sys_Pd_vcm_-
↪  c4;Sys_Pd_vcm_c5;Sys_Pd_vcm_c6;Sys_Pd_vcm_c7;Sys_Pd_vcm_c8;Sys_-
↪  Pd_vcm_c9];
Sys_Pd_pzt_all=[Sys_Pd_pzt_c1;Sys_Pd_pzt_c2;Sys_Pd_pzt_c3;Sys_Pd_pzt_-
↪  c4;Sys_Pd_pzt_c5;Sys_Pd_pzt_c6;Sys_Pd_pzt_c7;Sys_Pd_pzt_c8;Sys_-
↪  Pd_pzt_c9];

%% Frequency response
f=logspace(1,log10(60e3),3000);
Fr_Pc_vcm_all=squeeze(freqresp(Sys_Pc_vcm_all,f*2*pi)).';
Fr_Pc_pzt_all=squeeze(freqresp(Sys_Pc_pzt_all,f*2*pi)).';
Fr_Pd_vcm_all=squeeze(freqresp(Sys_Pd_vcm_all,f*2*pi)).';
Fr_Pd_pzt_all=squeeze(freqresp(Sys_Pd_pzt_all,f*2*pi)).';

figure
subplot(211)
semilogx(f,20*log10(abs(Fr_Pc_vcm_all(:,1:7))))
hold on
semilogx(f,20*log10(abs(Fr_Pc_vcm_all(:,8:9))),'--')
hold off
title('P_{cv}');xlabel('Frequency [Hz]');ylabel('Gain
↪  [dB]');grid;axis([1e3 f(end) -90 10])
subplot(212)
semilogx(f,mod(angle(Fr_Pc_vcm_all(:,1:7))*180/pi+360,360)-360)
hold on
semilogx(f,mod(angle(Fr_Pc_vcm_all(:,8:9))*180/pi+360,360)-360,'--')
hold off
xlabel('Frequency [Hz]');ylabel('Phase [deg.]');grid;axis([1e3 f(end)
↪  -360 0]);yticks(-360:90:0)
legend('Case 1','Case 2','Case 3','Case 4','Case 5','Case 6','Case
↪  7','Case 8','Case 9','Location','NorthWest')

figure
subplot(211)
```

```
semilogx(f,20*log10(abs(Fr_Pc_pzt_all(:,1:7))))
hold on
semilogx(f,20*log10(abs(Fr_Pc_pzt_all(:,8:9))),'--')
hold off
title('P_{cp}');xlabel('Frequency [Hz]');ylabel('Gain
↪  [dB]');grid;axis([1e3 f(end) -10 30])
subplot(212)
semilogx(f,angle(Fr_Pc_pzt_all(:,1:7))*180/pi)
hold on
semilogx(f,angle(Fr_Pc_pzt_all(:,8:9))*180/pi,'--')
hold off
xlabel('Frequency [Hz]');ylabel('Phase [deg.]');grid;axis([1e3 f(end)
↪  -180 180]);yticks(-180:90:180)
legend('Case 1','Case 2','Case 3','Case 4','Case 5','Case 6','Case
↪  7','Case 8','Case 9','Location','NorthWest')
```

2.9 Exercise

1. What is the difference between a causal system and an acausal system?
2. Can a static system be acausal?
3. Can a linear system be time-varying?
4. Solve the following ODEs:
   a) $\ddot{y} + \dot{y} - 2y = 0$, $y(0) = 4$, $\dot{y}(0) = -5$,
   b) $\ddot{y} + \dot{y} + 0.25y = 0$, $y(0) = 3$, $\dot{y}(0) = -3.5$,
   c) $\ddot{y} + 0.4\dot{y} + 9.04y = 0$.
5. A quadcopter is a type of unmanned aerial vehicle (UAV) that is controlled by adjusting the speed of its four rotors. Read relevant literature and develop a mathematical model of a quadcopter.
6. A robotic arm is used to move objects in a manufacturing facility. The arm consists of several joints that can be controlled to move the end-effector to a desired position. Read relevant literature. Develop a dynamic model between the torque input to the motors and the joint angles.
7. A power system consists of generators, transformers, transmission lines, and loads. Read relevant literature and develop a simplified model of a power system.
8. A cruise control system in a car maintains a constant speed by adjusting the throttle. The system must be able to handle changes in terrain and driving conditions. Read relevant literature and develop a simplified model of a cruise control system.
9. A temperature control system in a chemical plant consists of a heater, a temperature sensor, and a controller. The controller receives input from the temperature sensor and sends a signal to the heater to adjust the temperature. Develop a block diagram of the control system and a mathematical model of the plant.

3 Laplace and Z Transforms

3.1 The Laplace Transform . . 29
3.2 Inverse Laplace Transform and Partial Fraction Expansion 39
3.3 From Laplace Transform to Transfer Functions 41
3.4 The Z Transform 43
3.5 From Difference Equation to Discrete-Time Transfer Functions 51
3.6 Recap 52
3.7 Exercise 54

3.1 The Laplace Transform

3.1.1 The Laplace Approach to ODEs

The Laplace transform is a powerful tool to solve a wide variety of Ordinary Differential Equations (ODEs). To directly solve an ODE in calculus, we often need to solve a characteristic equation, find the time-domain modes of the solution, and use the method of undetermined coefficients to determine the specific parameters of each mode – a lot of work! With the Laplace transform, the calculus operators in the time domain are replaced by algebraic operations. Algebraic solutions are much easier to obtain, and time-domain ODE solutions can be subsequently obtained via the inverse Laplace transform. Moreover, we will be able to easily manipulate interconnected ODEs and conduct closed-loop analyses – all very useful for dynamic systems and controls.

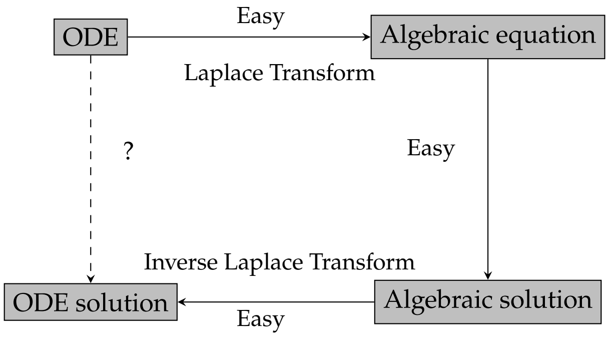


Figure 3.1: The Laplace approach to ODEs.

We start with reviewing a number of relevant mathematical definitions and notations.