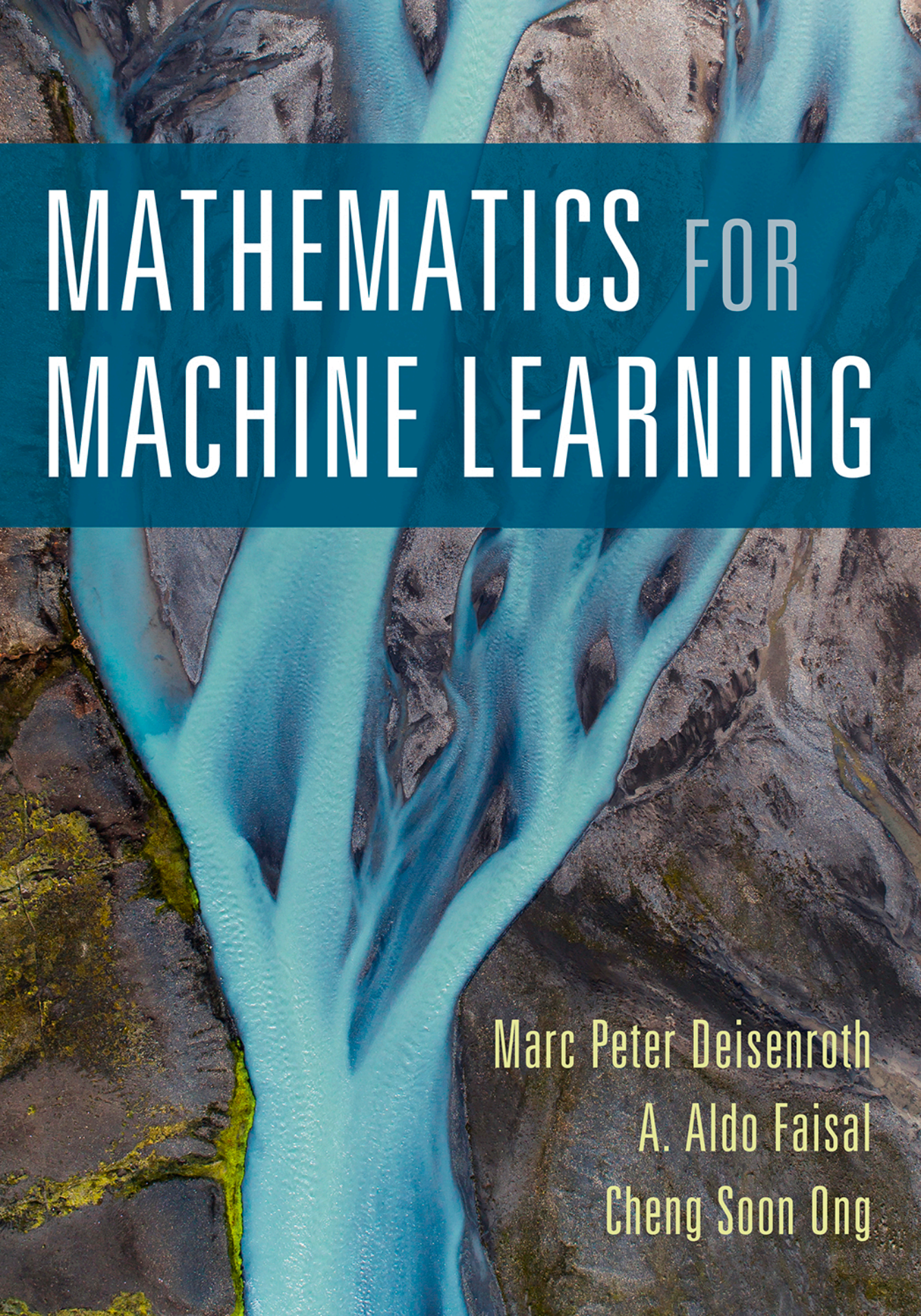

MATHEMATICS FOR
MACHINE LEARNING
Marc Peter Deisenroth
A. Aldo Faisal
Cheng Soon Ong

Mathematics for Machine Learning

The fundamental mathematical tools needed to understand machine learning include linear algebra, analytic geometry, matrix decompositions, vector calculus, optimization, probability, and statistics. These topics are traditionally taught in disparate courses, making it hard for data science or computer science students, or professionals, to efficiently learn the mathematics.

This self-contained textbook bridges the gap between mathematical and machine learning texts, introducing the mathematical concepts with a minimum of prerequisites. It uses these concepts to derive four central machine learning methods: linear regression, principal component analysis, Gaussian mixture models, and support vector machines. For students and others with a mathematical background, these derivations provide a starting point to machine learning texts. For those learning the mathematics for the first time, the methods help build intuition and practical experience with applying mathematical concepts.

Every chapter includes worked examples and exercises to test understanding. Programming tutorials are offered on the book's web site.

MARC PETER DEISENROTH is the DeepMind Chair in Artificial Intelligence at University College London. Prior to this, Marc was a faculty member at Imperial College London. His research areas include data-efficient learning, probabilistic modeling, and autonomous decision making. His research received Best Paper Awards at the ICRA 2014 and the ICCAS 2016. Marc has been awarded the President's Award for Outstanding Early Career Researcher at Imperial College London, a Google Faculty Research Award, and a Microsoft PhD grant.

A. ALDO FAISAL leads the Brain & Behaviour Lab at Imperial College London, where he is faculty at the Departments of Bioengineering and Computing and a Fellow of the Data Science Institute. He is the director of the 20Mio£ United Kingdom Research and Innovation (UKRI) Center for Doctoral Training in AI for Healthcare. He obtained a PhD in computational neuroscience at Cambridge University and became Junior Research Fellow in the Computational and Biological Learning Lab. His research is at the interface of neuroscience and machine learning to understand and reverse engineer brains and behavior.

CHENG SOON ONG is Principal Research Scientist at the Machine Learning Research Group, Data61, CSIRO and Adjunct Associate Professor at the Australian National University. His research focuses on enabling scientific discovery by extending statistical machine learning methods. He received his PhD in computer science at Australian National University in 2005. He has been a lecturer in the Department of Computer Science at ETH Zürich, and has worked in the Diagnostic Genomics Team at NICTA in Melbourne.

Mathematics for Machine Learning

Marc Peter Deisenroth
University College London

A. Aldo Faisal
Imperial College London

Cheng Soon Ong
Data61, CSIRO

CAMBRIDGE
UNIVERSITY PRESS

CAMBRIDGE
UNIVERSITY PRESS

University Printing House, Cambridge CB2 8BS, United Kingdom

One Liberty Plaza, 20th Floor, New York, NY 10006, USA

477 Williamstown Road, Port Melbourne, VIC 3207, Australia

314–321, 3rd Floor, Plot 3, Splendor Forum, Jasola District Centre, New Delhi – 110025, India

79 Anson Road, #06–04/06, Singapore 079906

Cambridge University Press is part of the University of Cambridge.

It furthers the University's mission by disseminating knowledge in the pursuit of education, learning, and research at the highest international levels of excellence.

www.cambridge.org
Information on this title: www.cambridge.org/9781108470049
DOI: 10.1017/9781108679930

First published 2020

Printed in Singapore by Markono Print Media Pte Ltd

A catalogue record for this publication is available from the British Library.

Library of Congress Cataloging-in-Publication Data
Names: Deisenroth, Marc Peter, author. | Faisal, A. Aldo, author. | Ong, Cheng Soon, author.
Title: Mathematics for machine learning / Marc Peter Deisenroth, A. Aldo Faisal, Cheng Soon Ong.
Description: Cambridge ; New York, NY : Cambridge University Press, 2020. |
Includes bibliographical references and index.
Identifiers: LCCN 2019040762 (print) | LCCN 2019040763 (ebook) |
ISBN 9781108470049 (hardback) | ISBN 9781108455145 (paperback) | ISBN 9781108679930 (epub)
Subjects: LCSH: Machine learning–Mathematics.
Classification: LCC Q325.5 .D45 2020 (print) | LCC Q325.5 (ebook) | DDC 006.3/1–dc23
LC record available at https://lccn.loc.gov/2019040762
LC ebook record available at https://lccn.loc.gov/2019040763

ISBN 978-1-108-47004-9 Hardback
ISBN 978-1-108-45514-5 Paperback

Additional resources for this publication at https://mml-book.com.

Contents

List of Symbols

Symbol	Typical meaning
$a, b, c, \alpha, \beta, \gamma$	Scalars are lowercase
$\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$	Vectors are bold lowercase
$\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}$	Matrices are bold uppercase
$\boldsymbol{x}^\top, \boldsymbol{A}^\top$	Transpose of a vector or matrix
$\boldsymbol{A}^{-1}$	Inverse of a matrix
$\langle \boldsymbol{x}, \boldsymbol{y} \rangle$	Inner product of $\boldsymbol{x}$ and $\boldsymbol{y}$
$\boldsymbol{x}^\top \boldsymbol{y}$	Dot product of $\boldsymbol{x}$ and $\boldsymbol{y}$
$B = (\boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}_3)$	(Ordered) tuple
$\boldsymbol{B} = [\boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}_3]$	Matrix of column vectors stacked horizontally
$\mathcal{B} = \{\boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}_3\}$	Set of vectors (unordered)
$\mathbb{Z}, \mathbb{N}$	Integers and natural numbers, respectively
$\mathbb{R}, \mathbb{C}$	Real and complex numbers, respectively
$\mathbb{R}^n$	n-dimensional vector space of real numbers
$\forall x$	Universal quantifier: for all x
$\exists x$	Existential quantifier: there exists x
$a := b$	a is defined as b
$a =: b$	b is defined as a
$a \propto b$	a is proportional to b, i.e., $a = \text{constant} \cdot b$
$g \circ f$	Function composition: "g after f"
$\iff$	If and only if
$\implies$	Implies
$\mathcal{A}, \mathcal{C}$	Sets
$a \in \mathcal{A}$	a is an element of the set $\mathcal{A}$
$\emptyset$	Empty set
D	Number of dimensions; indexed by $d = 1, \ldots, D$
N	Number of data points; indexed by $n = 1, \ldots, N$
$\boldsymbol{I}_m$	Identity matrix of size $m \times m$
$\boldsymbol{0}_{m,n}$	Matrix of zeros of size $m \times n$
$\boldsymbol{1}_{m,n}$	Matrix of ones of size $m \times n$
$\boldsymbol{e}_i$	Standard/canonical vector (where i is the component that is 1)
$\dim(\mathrm{V})$	Dimensionality of vector space V

Symbol	Typical meaning
$\mathrm{rk}(\boldsymbol{A})$	Rank of matrix $\boldsymbol{A}$
$\mathrm{Im}(\Phi)$	Image of linear mapping Φ
$\ker(\Phi)$	Kernel (null space) of a linear mapping Φ
$\mathrm{span}[\boldsymbol{b}_1]$	Span (generating set) of $\boldsymbol{b}_1$
$\mathrm{tr}(\boldsymbol{A})$	Trace of $\boldsymbol{A}$
$\det(\boldsymbol{A})$	Determinant of $\boldsymbol{A}$
$\lvert \cdot \rvert$	Absolute value or determinant (depending on context)
$\lVert \cdot \rVert$	Norm; Euclidean unless specified
λ	Eigenvalue or Lagrange multiplier
E_λ	Eigenspace corresponding to eigenvalue λ
$\boldsymbol{\theta}$	Parameter vector
$\frac{\partial f}{\partial x}$	Partial derivative of f with respect to x
$\frac{\mathrm{d}f}{\mathrm{d}x}$	Total derivative of f with respect to x
∇	Gradient
$\mathfrak{L}$	Lagrangian
$\mathcal{L}$	Negative log-likelihood
$\binom{n}{k}$	Binomial coefficient, n choose k
$\mathbb{V}_X[\boldsymbol{x}]$	Variance of $\boldsymbol{x}$ with respect to the random variable X
$\mathbb{E}_X[\boldsymbol{x}]$	Expectation of $\boldsymbol{x}$ with respect to the random variable X
$\mathrm{Cov}_{X,Y}[\boldsymbol{x}, \boldsymbol{y}]$	Covariance between $\boldsymbol{x}$ and $\boldsymbol{y}$.
$X \perp\!\!\!\perp Y \mid Z$	X is conditionally independent of Y given Z
$X \sim p$	Random variable X is distributed according to p
$\mathcal{N}\left(\boldsymbol{\mu}, \boldsymbol{\Sigma}\right)$	Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$
$\mathrm{Ber}(\mu)$	Bernoulli distribution with parameter μ
$\mathrm{Bin}(N, \mu)$	Binomial distribution with parameters N, μ
$\mathrm{Beta}(\alpha, \beta)$	Beta distribution with parameters α, β

List of Abbreviations and Acronyms

Acronym	Meaning
e.g.	Exempli gratia (Latin: for example)
GMM	Gaussian mixture model
i.e.	Id est (Latin: this means)
i.i.d.	Independent, identically distributed
MAP	Maximum a posteriori
MLE	Maximum likelihood estimation/estimator
ONB	Orthonormal basis
PCA	Principal component analysis
PPCA	Probabilistic principal component analysis
REF	Row-echelon form
SPD	Symmetric, positive definite
SVM	Support vector machine

Preface

Machine learning is the latest in a long line of attempts to distill human knowledge and reasoning into a form that is suitable for constructing machines and engineering automated systems. As machine learning becomes more ubiquitous and its software packages become easier to use, it is natural and desirable that the low-level technical details are abstracted away and hidden from the practitioner. However, this brings with it the danger that a practitioner becomes unaware of the design decisions and, hence, the limits of machine learning algorithms.

The enthusiastic practitioner who is interested to learn more about the magic behind successful machine learning algorithms currently faces a daunting set of prerequisite knowledge:

- Programming languages and data analysis tools
- Large-scale computation and the associated frameworks
- Mathematics and statistics and how machine learning builds on it

At universities, introductory courses on machine learning tend to spend early parts of the course covering some of these prerequisites. For historical reasons, courses in machine learning tend to be taught in the computer science department, where students are often trained in the first two areas of knowledge, but not so much in mathematics and statistics.

Current machine learning textbooks primarily focus on machine learning algorithms and methodologies and assume that the reader is competent in mathematics and statistics. Therefore, these books only spend one or two chapters of background mathematics, either at the beginning of the book or as appendices. We have found many people who want to delve into the foundations of basic machine learning methods who struggle with the mathematical knowledge required to read a machine learning textbook. Having taught undergraduate and graduate courses at universities, we find that the gap between high school mathematics and the mathematics level required to read a standard machine learning textbook is too big for many people.

This book brings the mathematical foundations of basic machine learning concepts to the fore and collects the information in a single place so that this skills gap is narrowed or even closed.

Why Another Book on Machine Learning?

Machine learning builds upon the language of mathematics to express concepts that seem intuitively obvious but that are surprisingly difficult to formalize. Once formalized properly, we can gain insights into the task we want to solve. One common complaint of students of mathematics around the globe is that the topics covered seem to have little relevance to practical problems. We believe that machine learning is an obvious and direct motivation for people to learn mathematics.

"Math is linked in the popular mind with phobia and anxiety. You'd think we're discussing spiders." (Strogatz, 2014, 281)

This book is intended to be a guidebook to the vast mathematical literature that forms the foundations of modern machine learning. We motivate the need for mathematical concepts by directly pointing out their usefulness in the context of fundamental machine learning problems. In the interest of keeping the book short, many details and more advanced concepts have been left out. Equipped with the basic concepts presented here, and how they fit into the larger context of machine learning, the reader can find numerous resources for further study, which we provide at the end of the respective chapters. For readers with a mathematical background, this book provides a brief but precisely stated glimpse of machine learning. In contrast to other books that focus on methods and models of machine learning (MacKay, 2003; Bishop, 2006; Alpaydin, 2010; Murphy, 2012; Barber, 2012; Shalev-Shwartz and Ben-David, 2014; Rogers and Girolami, 2016) or programmatic aspects of machine learning (Müller and Guido, 2016; Raschka and Mirjalili, 2017; Chollet and Allaire, 2018), we provide only four representative examples of machine learning algorithms. Instead, we focus on the mathematical concepts behind the models themselves. We hope that readers will be able to gain a deeper understanding of the basic questions in machine learning and connect practical questions arising from the use of machine learning with fundamental choices in the mathematical model.

We do not aim to write a classical machine learning book. Instead, our intention is to provide the mathematical background, applied to four central machine learning problems, to make it easier to read other machine learning textbooks.

Who Is the Target Audience?

As applications of machine learning become widespread in society, we believe that everybody should have some understanding of its underlying principles. This book is written in an academic mathematical style, which enables us to be precise about the concepts behind machine learning. We encourage readers unfamiliar with this seemingly terse style to persevere and to keep the goals of each topic in mind. We sprinkle comments and remarks throughout the text, in the hope that it provides useful guidance with respect to the big picture.

The book assumes the reader to have mathematical knowledge commonly covered in high school mathematics and physics. For example, the reader should have seen derivatives and integrals before, and geometric vectors in two or three dimensions. Starting from there, we generalize these concepts. Therefore, the

target audience of the book includes undergraduate university students, evening learners and learners participating in online machine learning courses.

In analogy to music, there are three types of interaction that people have with machine learning:

Astute Listener The democratization of machine learning by the provision of open-source software, online tutorials and cloud-based tools allows users to not worry about the specifics of pipelines. Users can focus on extracting insights from data using off-the-shelf tools. This enables non-tech-savvy domain experts to benefit from machine learning. This is similar to listening to music; the user is able to choose and discern between different types of machine learning, and benefits from it. More experienced users are like music critics, asking important questions about the application of machine learning in society such as ethics, fairness and privacy of the individual. We hope that this book provides a foundation for thinking about the certification and risk management of machine learning systems and allows them to use their domain expertise to build better machine learning systems.

Experienced Artist Skilled practitioners of machine learning can plug and play different tools and libraries into an analysis pipeline. The stereotypical practitioner would be a data scientist or engineer who understands machine learning interfaces and their use cases and is able to perform wonderful feats of prediction from data. This is similar to a virtuoso playing music, where highly skilled practitioners can bring existing instruments to life and bring enjoyment to their audience. Using the mathematics presented here as a primer, practitioners would be able to understand the benefits and limits of their favourite method, and to extend and generalize existing machine learning algorithms. We hope that this book provides the impetus for more rigorous and principled development of machine learning methods.

Fledgling Composer As machine learning is applied to new domains, developers of machine learning need to develop new methods and extend existing algorithms. They are often researchers who need to understand the mathematical basis of machine learning and uncover relationships between different tasks. This is similar to composers of music who, within the rules and structure of musical theory, create new and amazing pieces. We hope this book provides a high-level overview of other technical books for people who want to become composers of machine learning. There is a great need in society for new researchers who are able to propose and explore novel approaches for attacking the many challenges of learning from data.

Acknowledgments

We are grateful to many people who looked at early drafts of the book and suffered through painful expositions of concepts. We tried to implement their ideas that we did not vehemently disagree with. We would like to especially acknowledge Christfried Webers for his careful reading of many parts of the book, and his detailed suggestions on structure and presentation. Many friends and colleagues have also been kind enough to provide their time and energy on different versions of each chapter. We have been lucky to benefit from the generosity of the online community, who have suggested improvements via github.com, which greatly improved the book.

The following people have found bugs, proposed clarifications and suggested relevant literature, either via github.com or personal communication. Their names are sorted alphabetically.

Abdul-Ganiy Usman
Adam Gaier
Adele Jackson
Aditya Menon
Alasdair Tran
Aleksandar Krnjaic
Alexander Makrigiorgos
Alfredo Canziani
Ali Shafti
Amr Khalifa
Andrew Tanggara
Angus Gruen
Antal A. Buss
Antoine Toisoul Le Cann
Areg Sarvazyan
Artem Artemev
Artyom Stepanov
Bill Kromydas
Bob Williamson
Boon Ping Lim
Chao Qu
Cheng Li
Chris Sherlock
Christopher Gray
Daniel McNamara
Daniel Wood
Darren Siegel
David Johnston
Dawei Chen
Ellen Broad
Fengkuangtian Zhu
Fiona Condon
Georgios Theodorou
He Xin
Irene Raissa Kameni
Jakub Nabaglo
James Hensman
Jamie Liu
Jean Kaddour
Jean-Paul Ebejer
Jerry Qiang
Jitesh Sindhare
John Lloyd
Jonas Ngnawe
Jon Martin
Justin Hsi

Kai Arulkumaran
Kamil Dreczkowski
Lily Wang
Lionel Tondji Ngoupeyou
Lydia Knüfing
Mahmoud Aslan
Mark Hartenstein
Mark van der Wilk
Markus Hegland
Martin Hewing
Matthew Alger
Matthew Lee
Maximus McCann
Mengyan Zhang
Michael Bennett
Michael Pedersen
Minjeong Shin
Mohammad Malekzadeh
Naveen Kumar
Nico Montali
Oscar Armas
Patrick Henriksen
Patrick Wieschollek
Pattarawat Chormai
Paul Kelly
Petros Christodoulou
Piotr Januszewski
Pranav Subramani
Quyu Kong
Ragib Zaman
Rui Zhang
Ryan-Rhys Griffiths
Salomon Kabongo
Samuel Ogunmola
Sandeep Mavadia
Sarvesh Nikumbh
Sebastian Raschka
Senanayak Sesh Kumar Karri
Seung-Heon Baek
Shahbaz Chaudhary
Shakir Mohamed
Shawn Berry
Sheikh Abdul Raheem Ali
Sheng Xue
Sridhar Thiagarajan
Syed Nouman Hasany
Szymon Brych
Thomas Bühler
Timur Sharapov
Tom Melamed
Vincent Adam
Vincent Dutordoir
Vu Minh
Wasim Aftab
Wen Zhi
Wojciech Stokowiec
Xiaonan Chong
Xiaowei Zhang
Yazhou Hao
Yicheng Luo
Young Lee
Yu Lu
Yun Cheng
Yuxiao Huang
Zac Cranko
Zijian Cao
Zoe Nolan

Contributors through github, whose real names were not listed on their github profile, are the following:

SamDataMad	insad	empet
bumptiousmonkey	HorizonP	victorBigand
idoamihai	cs-maillist	17SKYE
deepakiim	kudo23	jessjing1995

We are also very grateful to Parameswaran Raman and the many anonymous reviewers, organized by Cambridge University Press, who read one or more

chapters of earlier versions of the manuscript, and provided constructive criticism that led to considerable improvements. A special mention goes to Dinesh Singh Negi, our LaTeX support for detailed and prompt advice about LaTeX-related issues. Last but not least, we are very grateful to our editor Lauren Cowles, who has been patiently guiding us through the gestation process of this book.

Part I

Mathematical Foundations

1

Introduction and Motivation

Machine learning is about designing algorithms that automatically extract valuable information from data. The emphasis here is on "automatic," i.e., machine learning is concerned about general-purpose methodologies that can be applied to many datasets, while producing something that is meaningful. There are three concepts that are at the core of machine learning: data, a model, and learning.

Since machine learning is inherently data driven, *data* is at the core of machine learning. The goal of machine learning is to design general-purpose methodologies to extract valuable patterns from data, ideally without much domain-specific expertise. For example, given a large corpus of documents (e.g., books in many libraries), machine learning methods can be used to automatically find relevant topics that are shared across documents (Hoffman et al., 2010). To achieve this goal, we design *models* that are typically related to the process that generates data, similar to the dataset we are given. For example, in a regression setting, the model would describe a function that maps inputs to real-valued outputs. To paraphrase Mitchell (1997): A model is said to learn from data if its performance on a given task improves after the data is taken into account. The goal is to find good models that generalize well to yet unseen data, which we may care about in the future. *Learning* can be understood as a way to automatically find patterns and structure in data by optimizing the parameters of the model.

data

model

learning

While machine learning has seen many success stories, and software is readily available to design and train rich and flexible machine learning systems, we believe that the mathematical foundations of machine learning are important in order to understand fundamental principles upon which more complicated machine learning systems are built. Understanding these principles can facilitate creating new machine learning solutions, understanding and debugging existing approaches, and learning about the inherent assumptions and limitations of the methodologies we are working with.

1.1 Finding Words for Intuitions

A challenge we face regularly in machine learning is that concepts and words are slippery, and a particular component of the machine learning system can be abstracted to different mathematical concepts. For example, the word "algorithm" is used in at least two different senses in the context of machine learning. In the first sense, we use the phrase "machine learning algorithm" to mean a system that makes predictions based on input data. We refer to these algorithms as *predictor*.

predictor

In the second sense, we use the exact same phrase "machine learning algorithm" to mean a system that adapts some internal parameters of the predictor so that it performs well on future unseen input data. Here we refer to this adaptation as *training* a system.

training

This book will not resolve the issue of ambiguity, but we want to highlight upfront that, depending on the context, the same expressions can mean different things. However, we attempt to make the context sufficiently clear to reduce the level of ambiguity.

The first part of this book introduces the mathematical concepts and foundations needed to talk about the three main components of a machine learning system: data, models, and learning. We will briefly outline these components here, and we will revisit them again in Chapter 8 once we have discussed the necessary mathematical concepts.

While not all data is numerical, it is often useful to consider data in a number format. In this book, we assume that *data* has already been appropriately converted into a numerical representation suitable for reading into a computer program. Therefore, we think of data as vectors. As another illustration of how subtle words are, there are (at least) three different ways to think about vectors: a vector as an array of numbers (a computer science view), a vector as an arrow with a direction and magnitude (a physics view), and a vector as an object that obeys addition and scaling (a mathematical view).

data as vectors

model

A *model* is typically used to describe a process for generating data, similar to the dataset at hand. Therefore, good models can also be thought of as simplified versions of the real (unknown) data-generating process, capturing aspects that are relevant for modeling the data and extracting hidden patterns from them. A good model can then be used to predict what would happen in the real world without performing real-world experiments.

learning

We now come to the crux of the matter, the *learning* component of machine learning. Assume we are given a dataset and a suitable model. *Training* the model means to use the data available to optimize some parameters of the model with respect to a utility function that evaluates how well the model predicts the training data. Most training methods can be thought of as an approach analogous to climbing a hill to reach its peak. In this analogy, the peak of the hill corresponds to a maximum of some desired performance measure. However, in practice, we are interested in the model to perform well on unseen data. Performing well on data that we have already seen (training data) may only mean that we found a good way to memorize the data. However, this may not generalize well to unseen data, and, in practical applications, we often need to expose our machine learning system to situations that it has not encountered before.

Let us summarize the main concepts of machine learning that we cover in this book:

- We represent data as vectors.
- We choose an appropriate model, either using the probabilistic or optimization view.
- We learn from available data by using numerical optimization methods with the aim that the model performs well on data not used for training.

1.2 Two Ways to Read This Book

We can consider two strategies for understanding the mathematics for machine learning:

- **Bottom-up:** Building up the concepts from foundational to more advanced. This is often the preferred approach in more technical fields, such as mathematics. This strategy has the advantage that the reader at all times is able to rely on their previously learned concepts. Unfortunately, for a practitioner many of the foundational concepts are not particularly interesting by themselves, and the lack of motivation means that most foundational definitions are quickly forgotten.
- **Top-down:** Drilling down from practical needs to more basic requirements. This goal-driven approach has the advantage that the readers know at all times why they need to work on a particular concept, and there is a clear path of required knowledge. The downside of this strategy is that the knowledge is built on potentially shaky foundations, and the readers have to remember a set of words that they do not have any way of understanding.

We decided to write this book in a modular way to separate foundational (mathematical) concepts from applications so that this book can be read in both ways. The book is split into two parts, where Part I lays the mathematical foundations and Part II applies the concepts from Part I to a set of fundamental machine learning problems, which form four pillars of machine learning as illustrated in Figure 1.1: regression, dimensionality reduction, density estimation, and classification. Chapters in Part I mostly build upon the previous ones, but it is possible to skip a chapter and work backward if necessary. Chapters in Part II are only loosely coupled and can be read in any order. There are many pointers forward and backward between the two parts of the book to link mathematical concepts with machine learning algorithms.

Of course there are more than two ways to read this book. Most readers learn using a combination of top-down and bottom-up approaches, sometimes building up basic mathematical skills before attempting more complex concepts, but also choosing topics based on applications of machine learning.

Part I Is about Mathematics

The four pillars of machine learning we cover in this book (see Figure 1.1) require a solid mathematical foundation, which is laid out in Part I.

We represent numerical data as vectors and represent a table of such data as a matrix. The study of vectors and matrices is called *linear algebra*, which we introduce in Chapter 2. The collection of vectors as a matrix is also described there.

linear algebra

Given two vectors representing two objects in the real world, we want to make statements about their similarity. The idea is that vectors that are similar should be predicted to have similar outputs by our machine learning algorithm (our predictor). To formalize the idea of similarity between vectors, we need to introduce operations that take two vectors as input and return a numerical

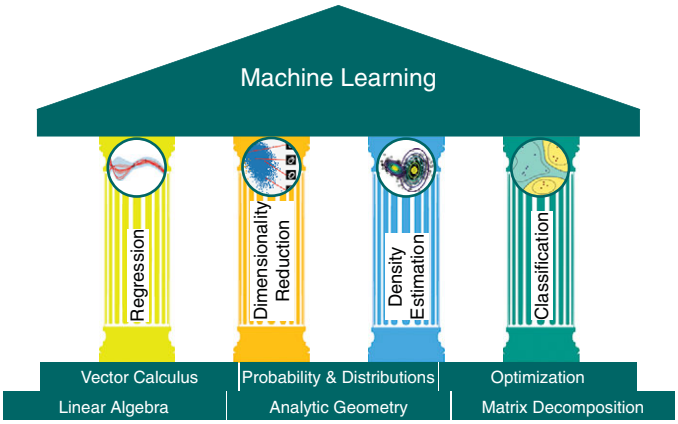


Figure 1.1 The foundations and four pillars of machine learning.

value representing their similarity. The construction of similarity and distances is central to *analytic geometry* and is discussed in Chapter 3.

analytic geometry

In Chapter 4, we introduce some fundamental concepts about matrices and *matrix decomposition*. Some operations on matrices are extremely useful in machine learning, and they allow for an intuitive interpretation of the data and more efficient learning.

matrix decomposition

We often consider data to be noisy observations of some true underlying signal. We hope that by applying machine learning we can identify the signal from the noise. This requires us to have a language for quantifying what "noise" means. We often would also like to have predictors that allow us to express some sort of uncertainty, e.g., to quantify the confidence we have about the value of the prediction at a particular test data point. Quantification of uncertainty is the realm of *probability theory* and is covered in Chapter 6.

probability theory

To train machine learning models, we typically find parameters that maximize some performance measure. Many optimization techniques require the concept of a gradient, which tells us the direction in which to search for a solution. Chapter 5 is about *vector calculus* and details the concept of gradients, which we subsequently use in Chapter 7, where we talk about *optimization* to find maxima/minima of functions.

vector calculus

optimization

Part II Is about Machine Learning

The second part of the book introduces *four pillars of machine learning* as shown in Figure 1.1. We illustrate how the mathematical concepts introduced in the first part of the book are the foundation for each pillar. Broadly speaking, chapters are ordered by difficulty (in ascending order).

In Chapter 8, we restate the three components of machine learning (data, models, and parameter estimation) in a mathematical fashion. In addition, we provide some guidelines for building experimental setups that guard against overly optimistic evaluations of machine learning systems. Recall that the goal is to build a predictor that performs well on unseen data.

In Chapter 9, we will have a close look at *linear regression*, where our objective is to find functions that map inputs $\boldsymbol{x} \in \mathbb{R}^D$ to corresponding observed

linear regression

function values $y \in \mathbb{R}$, which we can interpret as the labels of their respective inputs. We will discuss classical model fitting (parameter estimation) via maximum likelihood and maximum a posteriori estimation, as well as Bayesian linear regression, where we integrate the parameters out instead of optimizing them.

Chapter 10 focuses on *dimensionality reduction*, the second pillar in Figure 1.1, using principal component analysis. The key objective of dimensionality reduction is to find a compact, lower-dimensional representation of high-dimensional data $\boldsymbol{x} \in \mathbb{R}^D$, which is often easier to analyze than the original data. Unlike regression, dimensionality reduction is only concerned about modeling the data – there are no labels associated with a data point $\boldsymbol{x}$.

dimensionality reduction

In Chapter 11, we will move to our third pillar: *density estimation*. The objective of density estimation is to find a probability distribution that describes a given dataset. We will focus on Gaussian mixture models for this purpose, and we will discuss an iterative scheme to find the parameters of this model. As in dimensionality reduction, there are no labels associated with the data points $\boldsymbol{x} \in \mathbb{R}^D$. However, we do not seek a low-dimensional representation of the data. Instead, we are interested in a density model that describes the data.

density estimation

Chapter 12 concludes the book with an in-depth discussion of the fourth pillar: *classification*. We will discuss classification in the context of support vector machines. Similar to regression (Chapter 9), we have inputs $\boldsymbol{x}$ and corresponding labels y. However, unlike regression, where the labels were real-valued, the labels in classification are integers, which requires special care.

classification

1.3 Exercises and Feedback

We provide some exercises in Part I, which can be done mostly by pen and paper. For Part II, we provide programming tutorials (jupyter notebooks) to explore some properties of the machine learning algorithms we discuss in this book.

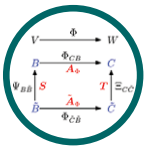
We appreciate that Cambridge University Press strongly supports our aim to democratize education and learning by making this book freely available for download at

https://mml-book.com

where tutorials, errata, and additional materials can be found. Mistakes can be reported and feedback provided using the preceding URL.

2

Linear Algebra

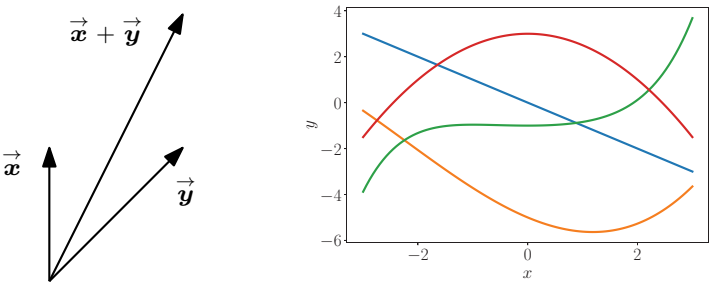


algebra

When formalizing intuitive concepts, a common approach is to construct a set of objects (symbols) and a set of rules to manipulate these objects. This is known as an *algebra*. Linear algebra is the study of vectors and certain rules to manipulate vectors. The vectors many of us know from school are called "geometric vectors," which are usually denoted by a small arrow above the letter, e.g., $\vec{x}$ and $\vec{y}$. In this book, we discuss more general concepts of vectors and use a bold letter to represent them, e.g., $\boldsymbol{x}$ and $\boldsymbol{y}$.

In general, vectors are special objects that can be added together and multiplied by scalars to produce another object of the same kind. From an abstract mathematical viewpoint, any object that satisfies these two properties can be considered a vector. Here are some examples of such vector objects:

1. Geometric vectors. This example of a vector may be familiar from high school mathematics and physics. Geometric vectors – see Figure 2.1(a) – are directed segments, which can be drawn (at least in two dimensions). Two geometric vectors $\vec{\boldsymbol{x}}, \vec{\boldsymbol{y}}$ can be added, such that $\vec{\boldsymbol{x}} + \vec{\boldsymbol{y}} = \vec{\boldsymbol{z}}$ is another geometric vector. Furthermore, multiplication by a scalar $\lambda \vec{\boldsymbol{x}}, \lambda \in \mathbb{R}$, is also a geometric vector. In fact, it is the original vector scaled by λ. Therefore, geometric vectors are instances of the vector concepts introduced previously. Interpreting vectors as geometric vectors enables us to use our intuitions about direction and magnitude to reason about mathematical operations.
2. Polynomials are also vectors; see Figure 2.1(b): Two polynomials can be added together, which results in another polynomial; and they can be multiplied by a scalar $\lambda \in \mathbb{R}$, and the result is a polynomial as well. Therefore, polynomials are (rather unusual) instances of vectors. Note that polynomials



(a) Geometric vectors

(b) Polynomials

Figure 2.1 Different types of vectors. Vectors can be surprising objects, including (a) geometric vectors and (b) polynomials.

are very different from geometric vectors. While geometric vectors are concrete "drawings," polynomials are abstract concepts. However, they are both vectors in the sense previously described.

3. Audio signals are vectors. Audio signals are represented as a series of numbers. We can add audio signals together, and their sum is a new audio signal. If we scale an audio signal, we also obtain an audio signal. Therefore, audio signals are a type of vector, too.
4. Elements of $\mathbb{R}^n$ (tuples of n real numbers) are vectors. $\mathbb{R}^n$ is more abstract than polynomials, and it is the concept we focus on in this book. For instance,

$$\boldsymbol{a} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \in \mathbb{R}^3 \tag{2.1}$$

is an example of a triplet of numbers. Adding two vectors $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^n$ componentwise results in another vector: $\boldsymbol{a} + \boldsymbol{b} = \boldsymbol{c} \in \mathbb{R}^n$. Moreover, multiplying $\boldsymbol{a} \in \mathbb{R}^n$ by $\lambda \in \mathbb{R}$ results in a scaled vector $\lambda\boldsymbol{a} \in \mathbb{R}^n$. Considering vectors as elements of $\mathbb{R}^n$ has an additional benefit that it loosely corresponds to arrays of real numbers on a computer. Many programming languages support array operations, which allow for convenient implementation of algorithms that involve vector operations.

Be careful to check whether array operations actually perform vector operations when implementing on a computer.

Linear algebra focuses on the similarities between these vector concepts. We can add them together and multiply them by scalars. We will largely focus on vectors in $\mathbb{R}^n$ since most algorithms in linear algebra are formulated in $\mathbb{R}^n$. We will see in Chapter 8 that we often consider data to be represented as vectors in $\mathbb{R}^n$. In this book, we will focus on finite-dimensional vector spaces, in which case there is a 1:1 correspondence between any kind of vector and $\mathbb{R}^n$. When it is convenient, we will use intuitions about geometric vectors and consider array-based algorithms.

One major idea in mathematics is the idea of "closure." This is the question: What is the set of all things that can result from my proposed operations? In the case of vectors: What is the set of vectors that can result by starting with a small set of vectors, and adding them to each other and scaling them? This results in a vector space (Section 2.4). The concept of a vector space and its properties underlie much of machine learning. The concepts introduced in this chapter are summarized in Figure 2.2.

Pavel Grinfeld's series on linear algebra: http://tinyurl.com/nahclwm
Gilbert Strang's course on linear algebra: http://tinyurl.com/29p5q8j
3Blue1Brown series on linear algebra: https://tinyurl.com/h5g4kps

This chapter is mostly based on the lecture notes and books by Drumm and Weil (2001), Strang (2003), Hogben (2013), Liesen and Mehrmann (2015), as well as Pavel Grinfeld's Linear Algebra series. Other excellent resources are Gilbert Strang's Linear Algebra course at MIT and the Linear Algebra Series by 3Blue1Brown.

Linear algebra plays an important role in machine learning and general mathematics. The concepts introduced in this chapter are further expanded to include the idea of geometry in Chapter 3. In Chapter 5, we will discuss vector calculus, where a principled knowledge of matrix operations is essential. In Chapter 10,

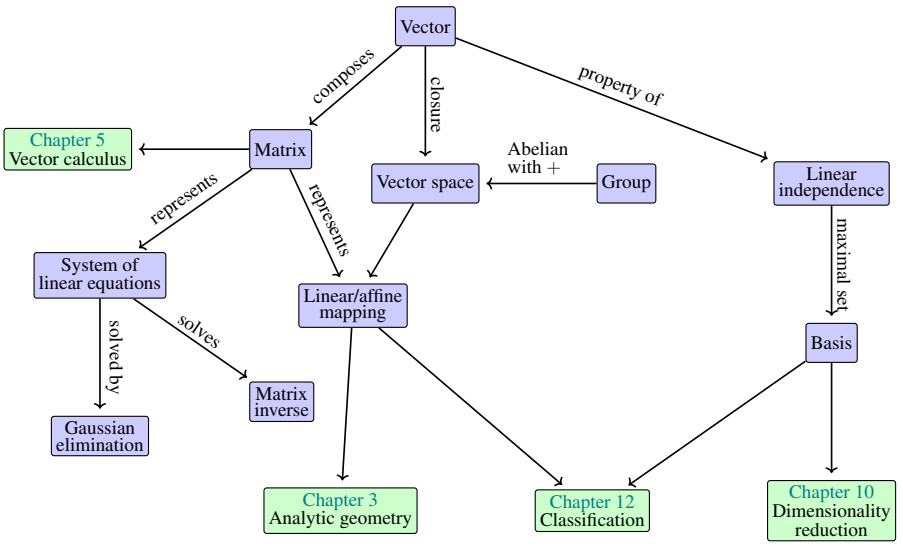


Figure 2.2 A mind map of the concepts introduced in this chapter, along with where they are used in other parts of the book.

we will use projections (to be introduced in Section 3.8) for dimensionality reduction with principal component analysis (PCA). In Chapter 9, we will discuss linear regression, where linear algebra plays a central role for solving least-squares problems.

2.1 Systems of Linear Equations

Systems of linear equations play a central part of linear algebra. Many problems can be formulated as systems of linear equations, and linear algebra gives us the tools for solving them.

Example 2.1

A company produces products $N_1, \ldots, N_n$ for which resources $R_1, \ldots, R_m$ are required. To produce a unit of product N_j, a_{ij} units of resource R_i are needed, where $i = 1, \ldots, m$ and $j = 1, \ldots, n$.

The objective is to find an optimal production plan, i.e., a plan of how many units x_j of product N_j should be produced if a total of b_i units of resource R_i are available and (ideally) no resources are left over.

If we produce $x_1, \ldots, x_n$ units of the corresponding products, we need a total of

$$a_{i1}x_1 + \cdots + a_{in}x_n \tag{2.2}$$

many units of resource R_i. An optimal production plan $(x_1, \ldots, x_n) \in \mathbb{R}^n$, therefore, has to satisfy the following system of equations:

$$\begin{aligned} a_{11}x_1 + \cdots + a_{1n}x_n &= b_1 \\ &\vdots \\ a_{m1}x_1 + \cdots + a_{mn}x_n &= b_m \end{aligned}, \tag{2.3}$$

where $a_{ij} \in \mathbb{R}$ and $b_i \in \mathbb{R}$.

Equation (2.3) is the general form of a *system of linear equations*, and $x_1, \ldots, x_n$ are the *unknowns* of this system. Every n-tuple $(x_1, \ldots, x_n) \in \mathbb{R}^n$ that satisfies (2.3) is a *solution* of the linear equation system.

system of linear equations

solution

Example 2.2

The system of linear equations

$$\begin{array}{rcrcrcll} x_1 & + & x_2 & + & x_3 & = & 3 & (1) \\ x_1 & - & x_2 & + & 2x_3 & = & 2 & (2) \\ 2x_1 & & & + & 3x_3 & = & 1 & (3) \end{array} \tag{2.4}$$

has *no solution:* Adding the first two equations yields $2x_1 + 3x_3 = 5$, which contradicts the third equation (3).

Let us have a look at the system of linear equations

$$\begin{array}{rcrcrcll} x_1 & + & x_2 & + & x_3 & = & 3 & (1) \\ x_1 & - & x_2 & + & 2x_3 & = & 2 & (2) \\ & & x_2 & + & x_3 & = & 2 & (3) \end{array} \; . \tag{2.5}$$

From the first and third equation, it follows that $x_1 = 1$. From (1) + (2), we get $2x_1 + 3x_3 = 5$, i.e., $x_3 = 1$. From (3), we then get that $x_2 = 1$. Therefore, $(1, 1, 1)$ is the only possible and *unique solution* (verify that $(1, 1, 1)$ is a solution by plugging in).

As a third example, we consider

$$\begin{array}{rcrcrcll} x_1 & + & x_2 & + & x_3 & = & 3 & (1) \\ x_1 & - & x_2 & + & 2x_3 & = & 2 & (2) \\ 2x_1 & & & + & 3x_3 & = & 5 & (3) \end{array} \; . \tag{2.6}$$

Since $(1) + (2) = (3)$, we can omit the third equation (redundancy). From (1) and (2), we get $2x_1 = 5 - 3x_3$ and $2x_2 = 1 + x_3$. We define $x_3 = a \in \mathbb{R}$ as a free variable, such that any triplet

$$\left(\frac{5}{2} - \frac{3}{2}a, \frac{1}{2} + \frac{1}{2}a, a \right), \quad a \in \mathbb{R} \tag{2.7}$$

is a solution of the system of linear equations, i.e., we obtain a solution set that contains *infinitely many* solutions.

In general, for a real-valued system of linear equations we obtain either no, exactly one, or infinitely many solutions. Linear regression (Chapter 9) solves a version of Example 2.1 when we cannot solve the system of linear equations.

Remark (Geometric Interpretation of Systems of Linear Equations). In a system of linear equations with two variables x_1, x_2, each linear equation defines a line on the x_1x_2-plane. Since a solution to a system of linear equations must satisfy all equations simultaneously, the solution set is the intersection of these lines. This intersection set can be a line (if the linear equations describe the same line), a point, or empty (when the lines are parallel). An illustration is given in Figure 2.3 for the system

$$\begin{aligned} 4x_1 + 4x_2 &= 5 \\ 2x_1 - 4x_2 &= 1 \end{aligned} \tag{2.8}$$

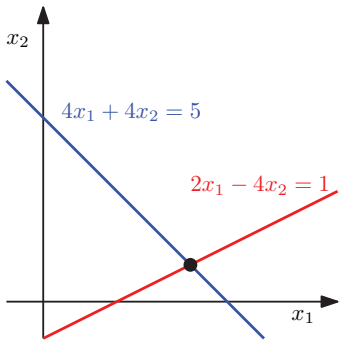


Figure 2.3 The solution space of a system of two linear equations with two variables can be geometrically interpreted as the intersection of two lines. Every linear equation represents a line.

where the solution space is the point $(x_1, x_2) = (1, \frac{1}{4})$. Similarly, for three variables, each linear equation determines a plane in three-dimensional space. When we intersect these planes, i.e., satisfy all linear equations at the same time, we can obtain a solution set that is a plane, a line, a point, or empty (when the planes have no common intersection). ♢

For a systematic approach to solving systems of linear equations, we will introduce a useful compact notation. We collect the coefficients a_{ij} into vectors and collect the vectors into matrices. In other words, we write the system from (2.3) in the following form:

$$x_1 \begin{bmatrix} a_{11} \\ \vdots \\ a_{m1} \end{bmatrix} + x_2 \begin{bmatrix} a_{12} \\ \vdots \\ a_{m2} \end{bmatrix} + \cdots + x_n \begin{bmatrix} a_{1n} \\ \vdots \\ a_{mn} \end{bmatrix} = \begin{bmatrix} b_1 \\ \vdots \\ b_m \end{bmatrix} \tag{2.9}$$

$$\iff \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ \vdots \\ b_m \end{bmatrix}. \tag{2.10}$$

In the following, we will have a close look at these *matrices* and define computation rules. We will return to solving linear equations in Section 2.3.

2.2 Matrices

Matrices play a central role in linear algebra. They can be used to compactly represent systems of linear equations, but they also represent linear functions (linear mappings), as we will see later in Section 2.7. Before we discuss some of these interesting topics, let us first define what a matrix is and what kind of operations we can do with matrices. We will see more properties of matrices in Chapter 4.

matrix

Definition 2.1 (Matrix)**.** With $m, n \in \mathbb{N}$ a real-valued (m, n) *matrix* $\boldsymbol{A}$ is an $m \cdot n$-tuple of elements a_{ij}, $i = 1, \ldots, m$, $j = 1, \ldots, n$, which is ordered according to a rectangular scheme consisting of m rows and n columns:

$$\boldsymbol{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}, \quad a_{ij} \in \mathbb{R}. \tag{2.11}$$

By convention $(1, n)$-matrices are called *rows*, and $(m, 1)$-matrices are called *columns*. These special matrices are also called *row/column vectors*.

row
column
row vector
column vector

$\mathbb{R}^{m\times n}$ is the set of all real-valued (m, n)-matrices. $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ can be equivalently represented as $\boldsymbol{a} \in \mathbb{R}^{mn}$ by stacking all n columns of the matrix into a long vector; see Figure 2.4.

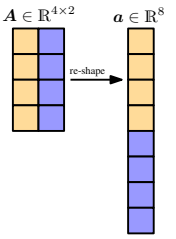


Figure 2.4 By stacking its columns, a matrix $\boldsymbol{A}$ can be represented as a long vector $\boldsymbol{a}$.

2.2.1 Matrix Addition and Multiplication

The sum of two matrices $\boldsymbol{A} \in \mathbb{R}^{m\times n}, \boldsymbol{B} \in \mathbb{R}^{m\times n}$ is defined as the element wise sum, i.e.,

$$\boldsymbol{A} + \boldsymbol{B} := \begin{bmatrix} a_{11} + b_{11} & \cdots & a_{1n} + b_{1n} \\ \vdots & & \vdots \\ a_{m1} + b_{m1} & \cdots & a_{mn} + b_{mn} \end{bmatrix} \in \mathbb{R}^{m\times n}. \tag{2.12}$$

For matrices $\boldsymbol{A} \in \mathbb{R}^{m\times n}, \boldsymbol{B} \in \mathbb{R}^{n\times k}$ the elements c_{ij} of the product $\boldsymbol{C} = \boldsymbol{A}\boldsymbol{B} \in \mathbb{R}^{m\times k}$ are computed as

$$c_{ij} = \sum_{l=1}^{n} a_{il} b_{lj}, \qquad i = 1, \ldots, m, \quad j = 1, \ldots, k. \tag{2.13}$$

Note the size of the matrices.

```
C = np.einsum('il, lj', A, B)
```

This means, to compute element c_{ij} we multiply the elements of the ith row of $\boldsymbol{A}$ with the jth column of $\boldsymbol{B}$ and sum them up. Later in Section 3.2, we will call this the *dot product* of the corresponding row and column. In cases where we need to be explicit that we are performing multiplication, we use the notation $\boldsymbol{A} \cdot \boldsymbol{B}$ to denote multiplication (explicitly showing "$\cdot$").

There are n columns in $\boldsymbol{A}$ and n rows in $\boldsymbol{B}$ so that we can compute $a_{il}b_{lj}$ for $l = 1, \ldots, n$.

Remark. Matrices can only be multiplied if their "neighboring" dimensions match. For instance, an $n \times k$-matrix $\boldsymbol{A}$ can be multiplied with a $k \times m$-matrix $\boldsymbol{B}$, but only from the left side:

$$\underbrace{\boldsymbol{A}}_{n\times k}\underbrace{\boldsymbol{B}}_{k\times m} = \underbrace{\boldsymbol{C}}_{n\times m} \tag{2.14}$$

The product $\boldsymbol{B}\boldsymbol{A}$ is not defined if $m \neq n$ since the neighboring dimensions do not match. ♢

Commonly, the dot product between two vectors $\boldsymbol{a}, \boldsymbol{b}$ is denoted by $\boldsymbol{a}^\top \boldsymbol{b}$ or $\langle \boldsymbol{a}, \boldsymbol{b} \rangle$.

Remark. Matrix multiplication is *not* defined as an elementwise operation on matrix elements, i.e., $c_{ij} \neq a_{ij}b_{ij}$ (even if the size of $\boldsymbol{A}, \boldsymbol{B}$ was chosen appropriately). This kind of elementwise multiplication often appears in programming languages when we multiply (multidimensional) arrays with each other, and is called a *Hadamard product*. ♢

Hadamard product

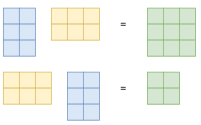
Example 2.3

For $\boldsymbol{A} = \begin{bmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \end{bmatrix} \in \mathbb{R}^{2\times 3}$, $\boldsymbol{B} = \begin{bmatrix} 0 & 2 \\ 1 & -1 \\ 0 & 1 \end{bmatrix} \in \mathbb{R}^{3\times 2}$, we obtain

$$\boldsymbol{A}\boldsymbol{B} = \begin{bmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \end{bmatrix} \begin{bmatrix} 0 & 2 \\ 1 & -1 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 2 & 3 \\ 2 & 5 \end{bmatrix} \in \mathbb{R}^{2\times 2}, \tag{2.15}$$

$$\boldsymbol{B}\boldsymbol{A} = \begin{bmatrix} 0 & 2 \\ 1 & -1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \end{bmatrix} = \begin{bmatrix} 6 & 4 & 2 \\ -2 & 0 & 2 \\ 3 & 2 & 1 \end{bmatrix} \in \mathbb{R}^{3\times 3}\,. \tag{2.16}$$

Figure 2.5 Even if both matrix multiplications $\boldsymbol{A}\boldsymbol{B}$ and $\boldsymbol{B}\boldsymbol{A}$ are defined, the dimensions of the results can be different.



From this example, we can already see that matrix multiplication is not commutative, i.e., $\boldsymbol{A}\boldsymbol{B} \neq \boldsymbol{B}\boldsymbol{A}$; see also Figure 2.5 for an illustration.

Definition 2.2 (Identity Matrix)**.** In $\mathbb{R}^{n\times n}$, we define the *identity matrix*

$$\boldsymbol{I}_n := \begin{bmatrix} 1 & 0 & \cdots & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & \cdots & 1 \end{bmatrix} \in \mathbb{R}^{n\times n} \tag{2.17}$$

identity matrix

as the $n \times n$-matrix containing 1 on the diagonal and 0 everywhere else.

Now that we defined matrix multiplication, matrix addition, and the identity matrix, let us have a look at some properties of matrices:

associativity

- *Associativity:*

$$\forall \boldsymbol{A} \in \mathbb{R}^{m\times n}, \boldsymbol{B} \in \mathbb{R}^{n\times p}, \boldsymbol{C} \in \mathbb{R}^{p\times q} : (\boldsymbol{A}\boldsymbol{B})\boldsymbol{C} = \boldsymbol{A}(\boldsymbol{B}\boldsymbol{C}) \tag{2.18}$$

distributivity

- *Distributivity:*

$$\forall \boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{m\times n}, \boldsymbol{C}, \boldsymbol{D} \in \mathbb{R}^{n\times p} : (\boldsymbol{A} + \boldsymbol{B})\boldsymbol{C} = \boldsymbol{A}\boldsymbol{C} + \boldsymbol{B}\boldsymbol{C} \tag{2.19a}$$

$$\boldsymbol{A}(\boldsymbol{C} + \boldsymbol{D}) = \boldsymbol{A}\boldsymbol{C} + \boldsymbol{A}\boldsymbol{D} \tag{2.19b}$$

- Multiplication with the identity matrix:

$$\forall \boldsymbol{A} \in \mathbb{R}^{m\times n} : \boldsymbol{I}_m \boldsymbol{A} = \boldsymbol{A}\boldsymbol{I}_n = \boldsymbol{A} \tag{2.20}$$

Note that $\boldsymbol{I}_m \neq \boldsymbol{I}_n$ for $m \neq n$.

2.2.2 Inverse and Transpose

A square matrix possesses the same number of columns and rows.

inverse

Definition 2.3 (Inverse)**.** Consider a square matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$. Let matrix $\boldsymbol{B} \in \mathbb{R}^{n\times n}$ have the property that $\boldsymbol{A}\boldsymbol{B} = \boldsymbol{I}_n = \boldsymbol{B}\boldsymbol{A}$. $\boldsymbol{B}$ is called the *inverse* of $\boldsymbol{A}$ and denoted by $\boldsymbol{A}^{-1}$.

regular
invertible
nonsingular
singular
noninvertible

Unfortunately, not every matrix $\boldsymbol{A}$ possesses an inverse $\boldsymbol{A}^{-1}$. If this inverse does exist, $\boldsymbol{A}$ is called *regular/invertible/nonsingular*, otherwise *singular/noninvertible*. When the matrix inverse exists, it is unique. In Section 2.3, we will discuss a general way to compute the inverse of a matrix by solving a system of linear equations.

Remark (Existence of the Inverse of a 2×2-matrix). Consider a matrix

$$\boldsymbol{A} := \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \in \mathbb{R}^{2\times 2}\,. \tag{2.21}$$

If we multiply $\boldsymbol{A}$ with

$$\boldsymbol{B} := \begin{bmatrix} a_{22} & -a_{12} \\ -a_{21} & a_{11} \end{bmatrix} \tag{2.22}$$

we obtain

$$\boldsymbol{AB} = \begin{bmatrix} a_{11}a_{22} - a_{12}a_{21} & 0 \\ 0 & a_{11}a_{22} - a_{12}a_{21} \end{bmatrix} = (a_{11}a_{22} - a_{12}a_{21})\boldsymbol{I}. \tag{2.23}$$

Therefore,

$$\boldsymbol{A}^{-1} = \frac{1}{a_{11}a_{22} - a_{12}a_{21}} \begin{bmatrix} a_{22} & -a_{12} \\ -a_{21} & a_{11} \end{bmatrix} \tag{2.24}$$

if and only if $a_{11}a_{22} - a_{12}a_{21} \neq 0$. In Section 4.1, we will see that $a_{11}a_{22} - a_{12}a_{21}$ is the determinant of a 2×2-matrix. Furthermore, we can generally use the determinant to check whether a matrix is invertible. ♢

Example 2.4 (Inverse Matrix)
The matrices

$$\boldsymbol{A} = \begin{bmatrix} 1 & 2 & 1 \\ 4 & 4 & 5 \\ 6 & 7 & 7 \end{bmatrix}, \quad \boldsymbol{B} = \begin{bmatrix} -7 & -7 & 6 \\ 2 & 1 & -1 \\ 4 & 5 & -4 \end{bmatrix} \tag{2.25}$$

are inverse to each other since $\boldsymbol{AB} = \boldsymbol{I} = \boldsymbol{BA}$.

Definition 2.4 (Transpose)**.** For $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ the matrix $\boldsymbol{B} \in \mathbb{R}^{n \times m}$ with $b_{ij} = a_{ji}$ is called the *transpose* of $\boldsymbol{A}$. We write $\boldsymbol{B} = \boldsymbol{A}^\top$.

transpose

The main diagonal (sometimes called "principal diagonal," "primary diagonal," "leading diagonal," or "major diagonal") of a matrix $\boldsymbol{A}$ is the collection of entries A_{ij} where $i = j$.

In general, $\boldsymbol{A}^\top$ can be obtained by writing the columns of $\boldsymbol{A}$ as the rows of $\boldsymbol{A}^\top$. The following are some important properties of inverses and transposes:

$$\boldsymbol{AA}^{-1} = \boldsymbol{I} = \boldsymbol{A}^{-1}\boldsymbol{A} \tag{2.26}$$

$$(\boldsymbol{AB})^{-1} = \boldsymbol{B}^{-1}\boldsymbol{A}^{-1} \tag{2.27}$$

$$(\boldsymbol{A} + \boldsymbol{B})^{-1} \neq \boldsymbol{A}^{-1} + \boldsymbol{B}^{-1} \tag{2.28}$$

$$(\boldsymbol{A}^\top)^\top = \boldsymbol{A} \tag{2.29}$$

$$(\boldsymbol{A} + \boldsymbol{B})^\top = \boldsymbol{A}^\top + \boldsymbol{B}^\top \tag{2.30}$$

$$(\boldsymbol{AB})^\top = \boldsymbol{B}^\top\boldsymbol{A}^\top \tag{2.31}$$

The scalar case of (2.28) is $\frac{1}{2+4} = \frac{1}{6} \neq \frac{1}{2} + \frac{1}{4}$.

Definition 2.5 (Symmetric Matrix)**.** A matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ is *symmetric* if $\boldsymbol{A} = \boldsymbol{A}^\top$.

symmetric matrix

Note that only (n, n)-matrices can be symmetric. Generally, we call (n, n)-matrices also *square matrices* because they possess the same number of rows and columns. Moreover, if $\boldsymbol{A}$ is invertible, then so is $\boldsymbol{A}^\top$, and $(\boldsymbol{A}^{-1})^\top = (\boldsymbol{A}^\top)^{-1} =: \boldsymbol{A}^{-\top}$.

square matrix

Remark (Sum and Product of Symmetric Matrices). The sum of symmetric matrices $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{n \times n}$ is always symmetric. However, although their product is always defined, it is generally not symmetric:

$$\begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix}. \tag{2.32}$$

♢

2.2.3 Multiplication by a Scalar

Let us look at what happens to matrices when they are multiplied by a scalar $\lambda \in \mathbb{R}$. Let $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ and $\lambda \in \mathbb{R}$. Then $\lambda \boldsymbol{A} = \boldsymbol{K}$, $K_{ij} = \lambda a_{ij}$. Practically, λ scales each element of $\boldsymbol{A}$. For $\lambda, \psi \in \mathbb{R}$, the following holds:

associativity

- *Associativity:*
  $(\lambda\psi)\boldsymbol{C} = \lambda(\psi\boldsymbol{C}), \quad \boldsymbol{C} \in \mathbb{R}^{m \times n}$
- $\lambda(\boldsymbol{B}\boldsymbol{C}) = (\lambda\boldsymbol{B})\boldsymbol{C} = \boldsymbol{B}(\lambda\boldsymbol{C}) = (\boldsymbol{B}\boldsymbol{C})\lambda, \quad \boldsymbol{B} \in \mathbb{R}^{m \times n}, \boldsymbol{C} \in \mathbb{R}^{n \times k}$.
  Note that this allows us to move scalar values around.

distributivity

- $(\lambda\boldsymbol{C})^\top = \boldsymbol{C}^\top\lambda^\top = \boldsymbol{C}^\top\lambda = \lambda\boldsymbol{C}^\top$ since $\lambda = \lambda^\top$ for all $\lambda \in \mathbb{R}$.
- *Distributivity:*
  $(\lambda + \psi)\boldsymbol{C} = \lambda\boldsymbol{C} + \psi\boldsymbol{C}, \quad \boldsymbol{C} \in \mathbb{R}^{m \times n}$
  $\lambda(\boldsymbol{B} + \boldsymbol{C}) = \lambda\boldsymbol{B} + \lambda\boldsymbol{C}, \quad \boldsymbol{B}, \boldsymbol{C} \in \mathbb{R}^{m \times n}$

Example 2.5 (Distributivity)
If we define

$$\boldsymbol{C} := \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, \tag{2.33}$$

then for any $\lambda, \psi \in \mathbb{R}$ we obtain

$$(\lambda + \psi)\boldsymbol{C} = \begin{bmatrix} (\lambda+\psi)1 & (\lambda+\psi)2 \\ (\lambda+\psi)3 & (\lambda+\psi)4 \end{bmatrix} = \begin{bmatrix} \lambda+\psi & 2\lambda+2\psi \\ 3\lambda+3\psi & 4\lambda+4\psi \end{bmatrix} \tag{2.34a}$$

$$= \begin{bmatrix} \lambda & 2\lambda \\ 3\lambda & 4\lambda \end{bmatrix} + \begin{bmatrix} \psi & 2\psi \\ 3\psi & 4\psi \end{bmatrix} = \lambda\boldsymbol{C} + \psi\boldsymbol{C}\,. \tag{2.34b}$$

2.2.4 Compact Representations of Systems of Linear Equations

If we consider the system of linear equations

$$\begin{aligned} 2x_1 + 3x_2 + 5x_3 &= 1 \\ 4x_1 - 2x_2 - 7x_3 &= 8 \\ 9x_1 + 5x_2 - 3x_3 &= 2 \end{aligned} \tag{2.35}$$

and use the rules for matrix multiplication, we can write this equation system in a more compact form as

$$\begin{bmatrix} 2 & 3 & 5 \\ 4 & -2 & -7 \\ 9 & 5 & -3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 8 \\ 2 \end{bmatrix}. \tag{2.36}$$

Note that x_1 scales the first column, x_2 the second one, and x_3 the third one.

Generally, a system of linear equations can be compactly represented in their matrix form as $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$; see (2.3), and the product $\boldsymbol{A}\boldsymbol{x}$ is a (linear) combination of the columns of $\boldsymbol{A}$. We will discuss linear combinations in more detail in Section 2.5.

2.3 Solving Systems of Linear Equations

In (2.3), we introduced the general form of an equation system, i.e.,

$$\begin{aligned} a_{11}x_1 + \cdots + a_{1n}x_n &= b_1 \\ &\vdots \\ a_{m1}x_1 + \cdots + a_{mn}x_n &= b_m \,, \end{aligned} \tag{2.37}$$

where $a_{ij} \in \mathbb{R}$ and $b_i \in \mathbb{R}$ are known constants and x_j are unknowns, $i = 1, \ldots, m$, $j = 1, \ldots, n$. Thus far, we saw that matrices can be used as a compact way of formulating systems of linear equations so that we can write $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$; see (2.10). Moreover, we defined basic matrix operations, such as addition and multiplication of matrices. In the following, we will focus on solving systems of linear equations and provide an algorithm for finding the inverse of a matrix.

2.3.1 Particular and General Solution

Before discussing how to generally solve systems of linear equations, let us have a look at an example. Consider the system of equations

$$\begin{bmatrix} 1 & 0 & 8 & -4 \\ 0 & 1 & 2 & 12 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 42 \\ 8 \end{bmatrix} . \tag{2.38}$$

The system has two equations and four unknowns. Therefore, in general we would expect infinitely many solutions. This system of equations is in a particularly easy form, where the first two columns consist of a 1 and a 0. Remember that we want to find scalars $x_1, \ldots, x_4$, such that $\sum_{i=1}^{4} x_i \boldsymbol{c}_i = \boldsymbol{b}$, where we define $\boldsymbol{c}_i$ to be the ith column of the matrix and $\boldsymbol{b}$ the right-hand side of (2.38). A solution to the problem in (2.38) can be found immediately by taking 42 times the first column and 8 times the second column so that

$$\boldsymbol{b} = \begin{bmatrix} 42 \\ 8 \end{bmatrix} = 42 \begin{bmatrix} 1 \\ 0 \end{bmatrix} + 8 \begin{bmatrix} 0 \\ 1 \end{bmatrix} . \tag{2.39}$$

Therefore, a solution is $[42, 8, 0, 0]^\top$. This solution is called a *particular solution* or *special solution*. However, this is not the only solution of this system of linear equations. To capture all the other solutions, we need to be creative in generating $\mathbf{0}$ in a nontrivial way using the columns of the matrix: Adding $\mathbf{0}$ to our special solution does not change the special solution. To do so, we express the third column using the first two columns (which are of this very simple form)

particular solution
special solution

$$\begin{bmatrix} 8 \\ 2 \end{bmatrix} = 8 \begin{bmatrix} 1 \\ 0 \end{bmatrix} + 2 \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{2.40}$$

so that $\mathbf{0} = 8\boldsymbol{c}_1 + 2\boldsymbol{c}_2 - 1\boldsymbol{c}_3 + 0\boldsymbol{c}_4$ and $(x_1, x_2, x_3, x_4) = (8, 2, -1, 0)$. In fact, any scaling of this solution by $\lambda_1 \in \mathbb{R}$ produces the $\mathbf{0}$ vector, i.e.,

$$\begin{bmatrix} 1 & 0 & 8 & -4 \\ 0 & 1 & 2 & 12 \end{bmatrix} \left(\lambda_1 \begin{bmatrix} 8 \\ 2 \\ -1 \\ 0 \end{bmatrix} \right) = \lambda_1 (8\boldsymbol{c}_1 + 2\boldsymbol{c}_2 - \boldsymbol{c}_3) = \mathbf{0} \,. \tag{2.41}$$

Following the same line of reasoning, we express the fourth column of the matrix in (2.38) using the first two columns and generate another set of nontrivial versions of $\mathbf{0}$ as

$$\begin{bmatrix}1 & 0 & 8 & -4\\ 0 & 1 & 2 & 12\end{bmatrix}\left(\lambda_2\begin{bmatrix}-4\\12\\0\\-1\end{bmatrix}\right) = \lambda_2(-4\boldsymbol{c}_1 + 12\boldsymbol{c}_2 - \boldsymbol{c}_4) = \mathbf{0} \tag{2.42}$$

for any $\lambda_2 \in \mathbb{R}$. Putting everything together, we obtain all solutions of the equation system in (2.38), which is called the *general solution*, as the set

general solution

$$\left\{\boldsymbol{x} \in \mathbb{R}^4 : \boldsymbol{x} = \begin{bmatrix}42\\8\\0\\0\end{bmatrix} + \lambda_1\begin{bmatrix}8\\2\\-1\\0\end{bmatrix} + \lambda_2\begin{bmatrix}-4\\12\\0\\-1\end{bmatrix}, \lambda_1, \lambda_2 \in \mathbb{R}\right\}. \tag{2.43}$$

Remark. The general approach we followed consisted of the following three steps:

1. Find a particular solution to $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$.
2. Find all solutions to $\boldsymbol{A}\boldsymbol{x} = \mathbf{0}$.
3. Combine the solutions from steps 1 and 2 to the general solution.

Neither the general nor the particular solution is unique. ♢

The system of linear equations in the preceding example was easy to solve because the matrix in (2.38) has this particularly convenient form, which allowed us to find the particular and the general solution by inspection. However, general equation systems are not of this simple form. Fortunately, there exists a constructive algorithmic way of transforming any system of linear equations into this particularly simple form: Gaussian elimination. Key to Gaussian elimination are elementary transformations of systems of linear equations, which transform the equation system into a simple form. Then we can apply the three steps to the simple form that we just discussed in the context of the example in (2.38).

2.3.2 Elementary Transformations

elementary transformations

Key to solving a system of linear equations are *elementary transformations* that keep the solution set the same, but that transform the equation system into a simpler form:

- Exchange of two equations (rows in the matrix representing the system of equations)
- Multiplication of an equation (row) with a constant $\lambda \in \mathbb{R}\backslash\{0\}$
- Addition of two equations (rows)

Example 2.6

For $a \in \mathbb{R}$, we seek all solutions of the following system of equations:

$$\begin{array}{rcrcrcrcrcr} -2x_1 & + & 4x_2 & - & 2x_3 & - & x_4 & + & 4x_5 & = & -3\\ 4x_1 & - & 8x_2 & + & 3x_3 & - & 3x_4 & + & x_5 & = & 2\\ x_1 & - & 2x_2 & + & x_3 & - & x_4 & + & x_5 & = & 0\\ x_1 & - & 2x_2 & & & - & 3x_4 & + & 4x_5 & = & a \end{array}. \tag{2.44}$$

We start by converting this system of equations into the compact matrix notation $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$. We no longer mention the variables $\boldsymbol{x}$ explicitly and build the *augmented matrix* (in the form $\left[\boldsymbol{A} \,|\, \boldsymbol{b}\right]$)

augmented matrix

$$\left[\begin{array}{rrrrr|r} -2 & 4 & -2 & -1 & 4 & -3 \\ 4 & -8 & 3 & -3 & 1 & 2 \\ 1 & -2 & 1 & -1 & 1 & 0 \\ 1 & -2 & 0 & -3 & 4 & a \end{array}\right] \begin{array}{l} \text{Swap with } R_3 \\ \\ \text{Swap with } R_1 \\ \\ \end{array}$$

where we used the vertical line to separate the left-hand side from the right-hand side in (2.44). We use $\rightsquigarrow$ to indicate a transformation of the augmented matrix using elementary transformations.

The augmented matrix $\left[\boldsymbol{A} \,|\, \boldsymbol{b}\right]$ compactly represents the system of linear equations $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$.

Swapping Rows 1 and 3 leads to

$$\left[\begin{array}{rrrrr|r} 1 & -2 & 1 & -1 & 1 & 0 \\ 4 & -8 & 3 & -3 & 1 & 2 \\ -2 & 4 & -2 & -1 & 4 & -3 \\ 1 & -2 & 0 & -3 & 4 & a \end{array}\right] \begin{array}{l} \\ -4R_1 \\ +2R_1 \\ -R_1 \end{array}$$

When we now apply the indicated transformations (e.g., subtract Row 1 four times from Row 2), we obtain

$$\left[\begin{array}{rrrrr|r} 1 & -2 & 1 & -1 & 1 & 0 \\ 0 & 0 & -1 & 1 & -3 & 2 \\ 0 & 0 & 0 & -3 & 6 & -3 \\ 0 & 0 & -1 & -2 & 3 & a \end{array}\right] -R_2 - R_3$$

$$\rightsquigarrow \left[\begin{array}{rrrrr|r} 1 & -2 & 1 & -1 & 1 & 0 \\ 0 & 0 & -1 & 1 & -3 & 2 \\ 0 & 0 & 0 & -3 & 6 & -3 \\ 0 & 0 & 0 & 0 & 0 & a+1 \end{array}\right] \begin{array}{l} \\ \cdot(-1) \\ \cdot(-\frac{1}{3}) \\ \end{array}$$

$$\rightsquigarrow \left[\begin{array}{rrrrr|r} 1 & -2 & 1 & -1 & 1 & 0 \\ 0 & 0 & 1 & -1 & 3 & -2 \\ 0 & 0 & 0 & 1 & -2 & 1 \\ 0 & 0 & 0 & 0 & 0 & a+1 \end{array}\right]$$

This (augmented) matrix is in a convenient form, the *row-echelon form* (REF). Reverting this compact notation back into the explicit notation with the variables we seek, we obtain

row-echelon form

$$\begin{array}{rcrcrcrcr} x_1 & - & 2x_2 & + & x_3 & - & x_4 & + & x_5 & = & 0 \\ & & & & x_3 & - & x_4 & + & 3x_5 & = & -2 \\ & & & & & & x_4 & - & 2x_5 & = & 1 \\ & & & & & & & & 0 & = & a+1 \end{array} \quad . \tag{2.45}$$

Only for $a = -1$ this system can be solved. A *particular solution* is

particular solution

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} 2 \\ 0 \\ -1 \\ 1 \\ 0 \end{bmatrix} . \tag{2.46}$$

general solution

The *general solution*, which captures the set of all possible solutions, is

$$\left\{ \boldsymbol{x} \in \mathbb{R}^5 : \boldsymbol{x} = \begin{bmatrix} 2 \\ 0 \\ -1 \\ 1 \\ 0 \end{bmatrix} + \lambda_1 \begin{bmatrix} 2 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \lambda_2 \begin{bmatrix} 2 \\ 0 \\ -1 \\ 2 \\ 1 \end{bmatrix}, \quad \lambda_1, \lambda_2 \in \mathbb{R} \right\}. \quad (2.47)$$

In the following, we will detail a constructive way to obtain a particular and general solution of a system of linear equations.

pivot

Remark (Pivots and Staircase Structure). The leading coefficient of a row (first nonzero number from the left) is called the *pivot* and is always strictly to the right of the pivot of the row above it. Therefore, any equation system in row-echelon form always has a "staircase" structure. ♢

row-echelon form

Definition 2.6 (Row-Echelon Form)**.** A matrix is in *row-echelon form* if

- All rows that contain only zeros are at the bottom of the matrix; correspondingly, all rows that contain at least one nonzero element are on top of rows that contain only zeros.
- Looking at nonzero rows only, the first nonzero number from the left (also called the *pivot* or the *leading coefficient*) is always strictly to the right of the pivot of the row above it.

pivot
leading coefficient
In other texts, it is sometimes required that the pivot is 1.

Remark (Basic and Free Variables). The variables corresponding to the pivots in the row-echelon form are called *basic variable*, and the other variables are *free variable*. For example, in (2.45), x_1, x_3, x_4 are basic variables, whereas x_2, x_5 are free variables. ♢

basic variable
free variable

Remark (Obtaining a Particular Solution). The row-echelon form makes our lives easier when we need to determine a particular solution. To do this, we express the right-hand side of the equation system using the pivot columns, such that $\boldsymbol{b} = \sum_{i=1}^{P} \lambda_i \boldsymbol{p}_i$, where $\boldsymbol{p}_i,\ i = 1, \ldots, P$, are the pivot columns. The λ_i are determined easiest if we start with the rightmost pivot column and work our way to the left.

In the previous example, we would try to find $\lambda_1, \lambda_2, \lambda_3$ so that

$$\lambda_1 \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \lambda_2 \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} + \lambda_3 \begin{bmatrix} -1 \\ -1 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ -2 \\ 1 \\ 0 \end{bmatrix}. \quad (2.48)$$

From here, we find relatively directly that $\lambda_3 = 1, \lambda_2 = -1, \lambda_1 = 2$. When we put everything together, we must not forget the nonpivot columns for which we set the coefficients implicitly to 0. Therefore, we get the particular solution $\boldsymbol{x} = [2, 0, -1, 1, 0]^\top$. ♢