Beyond Limits

Infinitesimals, Abstract Algebra, Category Theory, Synthetic Differential Geometry, and Automatic Differentiation

by **Stephen Fratini**

Table of Contents

List of Figures 8
List of Tables 9
Preface 10
Legal 11
1 Introduction 14
1.1 Purpose 14
1.2 Intended Audience 14
1.3 Prerequisites 15
1.4 Outline 15
2 Introduction to Infinitesimals and Differentiation 17
2.1 The Allure of Infinitesimals 17
2.2 Hyperreal Numbers: Infinitesimals in Non-Standard Analysis 17
2.3 Dual Numbers: Algebraic Infinitesimals 19
2.4 Category Theory: An Abstract Framework for Smoothness 19
2.5 Automatic Differentiation: Computational Derivatives 21
2.6 Synergies and Motivations 23
3 Abstract Algebra 25
3.1 Overview 25
3.2 Groups 26
3.2.1 Definition 26
3.2.2 Examples 27
3.2.3 Some Basic Theorems 31
3.2.4 Subgroups 33
3.2.5 Group Structure 35
3.2.6 Classification of Finite Simple Groups 40
3.3 Rings 41
3.3.1 Definitions and Basic Concepts 41
3.3.2 Examples 43
3.3.3 Ideals 46
3.3.4 Quotient Rings 48
3.4 Fields 50
3.4.1 Definitions and Basic Concepts 50

3.4.2 Construction of Non-prime Finite Fields....52
3.5 Vector Spaces....56
3.5.1 Definition....56
3.5.2 Examples....57
3.5.3 Concepts....58
3.6 Algebras....61
3.6.1 Comparison to Fields and Vector Spaces....61
3.6.2 Simple Real-World Example: The Complex Numbers....62
3.6.3 Some Other Examples....63
3.6.4 Summary....63
4 Extending Number Systems....64
4.1 Dedekind Cuts....64
4.2 Complex Numbers....66
4.3 Quaternions....68
4.3.1 Definition of Quaternions....68
4.3.2 Background and History....68
4.3.3 Derivation and View as an Extension of Complex Numbers....69
4.3.4 Summary and Modern Usage....70
5 Hyperreal Numbers: Foundations of Non-Standard Analysis....71
5.1 Key Definitions....71
5.2 Ultrapower Construction....74
5.3 The Standard Part Function....75
5.4 Summary: The Hyperreal Universe from the Ultrapower....76
5.5 Key Properties....78
5.6 Rationale for the Ultrapower Construction....79
5.6.1 The Dream vs. The Reality....79
5.6.2 Why the Ultrapower is Necessary....80
5.6.3 Analogy: Real Numbers....81
5.6.4 Conclusion....81
6 Dual Numbers: Algebra and Geometry....82
6.1 Introduction....82
6.2 Definition and Algebraic Structure....82
6.3 Properties and Operations....83

6.4 Geometric Interpretations 84
6.5 Applications in Differentiation 84
6.6 Extension to Vectors and Matrices 86
6.7 Extension to Hyper-Dual Numbers 86
7 Comparing Hyperreals and Duals: Infinitesimal Approaches 89
7.1 Introduction 89
7.2 Structural Similarities and Differences 90
7.3 Infinitesimals: Invertible vs. Nilpotent 91
7.4 Applications in Differentiation: Comparisons 91
7.5 Examples 92
8 Surreal Numbers 94
8.1 What are Surreal Numbers? An Intuitive Overview 94
8.2 Formal Definition and Construction 94
8.3 Order and Equality 95
8.4 Fundamental Theorems 97
8.5 The Real Numbers as a Subset of the Surreals 99
8.6 Relation to Hyperreal and Dual Numbers 101
8.7 Key Properties and Remarkable Features 102
8.8 Summary and Conclusion 103
9 Weil Algebras: The Algebra of Infinitesimals 104
9.1 Overview 104
9.2 Step-by-Step Analogy 104
9.2.1 Example 1: The Dual Numbers (The simplest Weil algebra) 104
9.2.2 Example 2: A Slightly More Complex Weil Algebra 105
9.3 The Formal Definition in Simple Terms 105
9.4 What is it Used For? 106
9.5 Summary in a Nutshell 106
10 Basics of Category Theory 107
10.1 Introduction 107
10.2 What is a Category? 107
10.3 Morphisms 108
10.3.1 Types of Morphisms 109
10.3.2 Other Notable Morphism Types 110

10.3.3 Why Morphisms Matter 110
10.4 Functors: Maps Between Categories 110
10.5 Natural Transformations: Maps Between Functors 115
10.5.1 Basic Definition 116
10.5.2 The Naturality Condition: What Makes It "Natural"? 116
10.5.3 Why Natural Transformations Matter 117
10.5.4 Example 1: The Diagonal Transformation 118
10.5.5 Example 2: Inclusion of Constants 118
10.5.6 Common Pitfalls and Tips for Beginners 119
10.6 Cartesian Closed Categories: Prelude to Synthetic Differential Geometry 119
10.6.1 Introduction to Cartesian Closed Categories 119
10.6.2 Some Topological Background 120
10.6.3 Finite Products: Combining Objects 121
10.6.4 Exponentials: Modeling Function Spaces 122
10.6.5 Full Definition of Cartesian Closed Categories 124
10.6.6 Examples of Cartesian Closed Categories 125
10.6.7 Prelude to Synthetic Differential Geometry 125
10.6.8 Exercises 126
11 Synthetic Differential Geometry: A Categorical Lens 129
11.1 Introduction 129
11.2 Subobjects and Colimits in Category Theory 129
11.2.1 Subobjects: Generalized Subsets 129
11.2.2 Limits: Combining Objects via Universal Properties 130
11.2.3 Colimits: Gluing Objects Together 132
11.3 Classic Real Numbers versus The Smooth Real Line 133
11.4 Toposes: The Categorical Foundation 133
11.5 The Kock-Lawvere Axiom 135
11.5.1 The Basic Kock-Lawvere Axiom 135
11.5.2 Generalized Axiom 136
11.5.3 Implications and Proof Sketch 138
11.6 Infinitesimals in SDG 139
11.7 Applications to Differentiation and Geometry 140
11.7.1 Synthetic Derivatives 140

11.7.2 Geometric Structures: Tangent Bundles and Vector Fields....141
11.7.3 Connections and Curvature....141
11.8 Connections to Hyperreals and Duals....142
11.9 Examples....143
11.9.1 Example 1: The Fundamental Theorem of Calculus (FTC)....143
11.9.2 Example 2: Vector Fields and Flow on the Circle....144
11.9.3 Example 3: The Symmetry of Second Derivatives (Clairaut's Theorem)....145
12 Automatic Differentiation: Principles and Modes....147
12.1 Introduction....147
12.2 Principles of Automatic Differentiation....147
12.2.1 Computational Graphs....147
12.2.2 Decomposition into Elementary Operations....148
12.3 Forward-Mode Automatic Differentiation....149
12.3.1 Theory: Tangents and the Connection to Dual Numbers....149
12.3.2 The Forward-Mode AD Algorithm....150
12.3.3 Step-by-Step Example....150
12.3.4 Complexity and Use Cases....152
12.4 Reverse-Mode Automatic Differentiation (Backpropagation)....152
12.4.1 Theory: Adjoints and the Reverse Pass....152
12.4.2 Why the Backward Pass Works....153
12.4.3 The Reverse-Mode AD Algorithm....155
12.4.4 Step-by-Step Example....155
12.4.5 Complexity and Use Cases....156
12.5 Practical Guidance: Choosing Between Forward and Reverse Mode....157
12.5.1 Case Study 1: The Jacobian-Vector Product (Forward-Mode Domain)....157
12.5.2 Case Study 2: The Vector-Jacobian Product (Reverse-Mode Domain)....158
12.5.3 Summary and Decision Matrix....159
12.6 Implementations and Efficiency....159
12.6.1 Implementation Techniques....159
12.6.2 Memory and Computational Trade-offs....162
12.7 A Categorical Perspective on Automatic Differentiation....163
12.7.1 Forward-Mode AD as a Functor....163
12.7.2 Reverse-Mode AD and the Reverse Derivative Category....164

12.7.3 Unification and Generalization 166
12.8 Bayesian Inference with Automatic Differentiation 166
12.8.1 The Core Computational Problem in Bayesian Inference 166
12.8.2 The Role of Gradients: Hamiltonian Monte Carlo 167
12.8.3 How Automatic Differentiation Enables Modern Bayesian Inference 167
12.9 Exercises 168
12.9.1 Fundamental Practice 168
12.9.2 Implementation and Design 169
12.9.3 Theoretical and Conceptual 169
12.9.4 Challenge Problem 170
12.9.5 Hints and Solutions 171
Acronyms and Symbols 187
References 189
Index of Terms 193

List of Figures

Figure 1. The hyperreals near the reals 18
Figure 2. Filters and their relationships 73
Figure 3. Summary of Ultrapower Construction 76
Figure 4. Adjunction between F and U 113
Figure 5. Adjunction between F and U (detailed version) 114
Figure 6. Natural Transformation 116
Figure 7. Commutative diagram for the naturality condition 117
Figure 8. Diagram for finite product 121
Figure 9. Commutative diagram concerning exponentials 122
Figure 10. Limit diagram example 131
Figure 11. Pullback diagram 131
Figure 12. Colimit diagram example 132
Figure 13. Computational graph for Example 1 148
Figure 14. Computation graph for f(x) = sin(x) + x^2 151

List of Tables

Table 1. Cayley table for $\boldsymbol{D3}$ 31
Table 2. Addition table for the finite field of order 4 54
Table 3. Multiplication table for the finite field of order 4 54
Table 4. Addition table for the finite field of order 8 55
Table 5. Multiplication table for the finite field of order 8 55
Table 6. Comparison of Ring, Field, Vector Space and Algebra 63
Table 7. Comparison of Complex Numbers and Quaternions 70
Table 8. Comparison of dual, hyperreal and surreal numbers 101
Table 9. Summary of Weil algebra 106
Table 10. Infinitesimals Comparison 140
Table 11. Forward-mode AD trace 152
Table 12. Forward Pass (Compute and Store Primal Values) 155
Table 13. Backward Pass (Compute Adjoints $\boldsymbol{v = \partial f \partial v}$) 156

Preface

For centuries, mathematicians have been captivated by a deceptively simple, almost magical idea: the infinitesimal. These are quantities so small that they defy measurement—smaller than any positive number you can name, yet somehow not quite zero. They were the hidden engine of calculus, wielded with brilliant intuition by pioneers like Newton and Leibniz to unravel the mysteries of motion, change, and the curves of nature itself.

Though the quest for rigor in the 19th century seemingly banished infinitesimals in favor of the formal precision of limits, their spirit refused to fade. It lingered in the background, a ghost in the mathematical machine, waiting for its return.

This book is the story of that return.

We will trace the remarkable legacy of the infinitesimal as it re-emerges, transformed and more powerful than ever, across four distinct yet deeply interconnected mathematical landscapes:

- **Hyperreal Numbers**: Here, we give the infinitesimals of old a modern, rigorous home. In the world of non-standard analysis, we can once again manipulate these elusive quantities with confidence, performing calculus with an intuitive clarity that feels both ancient and revolutionary.
- **Dual Numbers**: Next, we encounter an elegant algebraic structure where computing derivatives become almost effortless. These "shadow numbers" act as a computational powerhouse, turning the abstract process of differentiation into simple arithmetic.
- **Synthetic Differential Geometry (SDG)**: From there, we ascend into the abstract universe of category theory. In this rarefied realm, we will discover a vision of geometry built from the ground up on an infinitesimal foundation – a perspective that is breathtaking in its simplicity and power.
- **Automatic Differentiation (AD)**: Finally, we descend from abstraction into the intensely practical world of computation. AD is the workhorse that translates these theoretical ideas into raw speed and accuracy, powering everything from machine learning and scientific simulation to financial modeling.

This journey is a testament to the enduring power of a profound intuition. It reveals how a concept, once deemed too elusive for formal mathematics, could not only be resurrected and rigorously defined but could also spawn powerful new fields of thought and become the bedrock of the computational tools that shape our modern world.

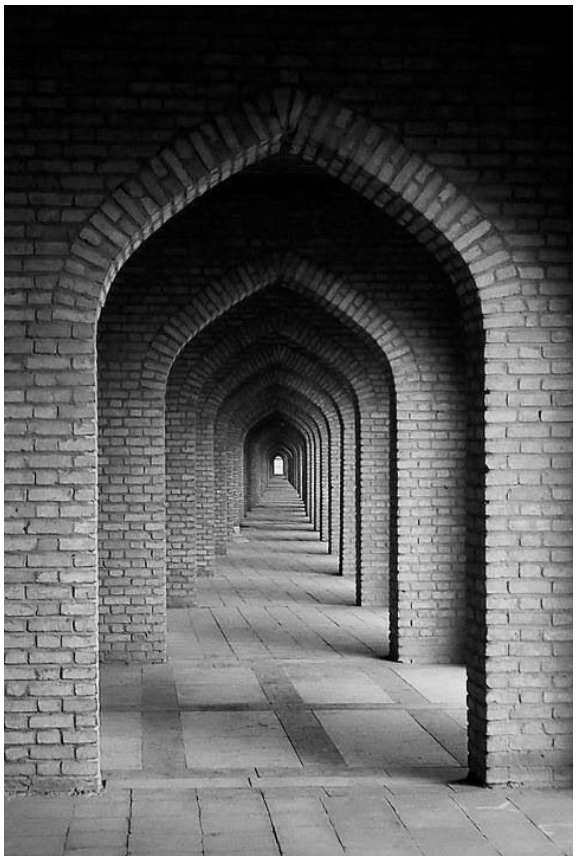
To equip you for this journey, the book includes a foundational section on abstract algebra, as its language and structures are essential for understanding the later chapters. A section on surreal numbers is also included for completeness, as this astonishing number system provides a unifying framework that encompasses the reals, duals, and hyperreals. However, the surreal numbers are not a prerequisite for grasping the core ideas of Synthetic Differential Geometry and Automatic Differentiation.

We invite you to turn the page and begin an exploration that stretches from the intuitive beginnings of calculus to the very frontiers of modern mathematics and computation.

Legal

Everything old is new again.

Originates from the song of the same name, written by Peter Allen and Carole Bayer Sager



Stephen Fratini
Sole Proprietor of The Art of Managing Things
Eatontown, New Jersey (USA)
Email: sfratini@artofmanagingthings.com or sfratini@outlook.com
LinkedIn: www.linkedin.com/in/stephenfratini

Other books by the author:

- *The Art of Managing Things (2nd edition)*, self-published on Amazon, https://www.amazon.com/Art-Managing-Things-Stephen-Fratini-ebook/dp/B07N4H4YWH/, January 2019.
- *Mathematical Thinking: Exercises for the Mind (2nd Edition)*, self-published on Amazon, https://www.amazon.com/dp/B0CL34FRP1, October 2023.
- *Financial Mathematics (2nd Edition)*, self-published on Barnes and Noble, https://www.barnesandnoble.com/w/financial-mathematics-stephen-fratini/1145166826, March 2023.
- *Math in Art, and Art in Math*, self-published on Amazon, https://www.amazon.com/dp/B091D1F8MB, March 2021.
- *Algebra through Discovery and Experimentation*, self-published on Amazon, https://www.amazon.com/dp/B09B5L9WL5, July 2021.
- *The Struggle Against Chaos*, self-published on Amazon, https://www.amazon.com/dp/B09BLPQ86Q, July 2021.
- *Mathematical Vignettes: Number theory, stochastic processes, game theory, cryptography, linear programming and more*, self-published on Amazon, https://www.amazon.com/Mathematical-Vignettes-stochastic-cryptography-programming-ebook/dp/B0BBP1PBJQ/, August 2022.
- *Learning Math through Puzzles: Number properties, counting, sequences and series, algebra, functions, and mathematical reasoning*, self-published on Amazon, https://www.amazon.com/dp/B0BZFRZP5B, March 2023.
- *Mathematical Vignettes: Volume II: Topics from combinatorial design, magic squares, finite geometry, abstract algebra, error correcting codes, geometric packing problems and much more,* self-published on Amazon, *https://www.amazon.com/dp/B0CM1CLSK8, October 2023.*
- *Shape Up and Solve It!: Learn Geometry Through Puzzles*, self-published on Amazon, https://www.amazon.com/dp/B0CRS7DRWF, January 2024.
- *Mathematical Vignettes III (2nd edition): Introductions to non-Euclidean geometry, topology and complex analysis*, self-published on Amazon, https://www.amazon.com/Mathematical-Vignettes-III-Introductions-non-Euclidean/dp/B0F1N6522B, March 2025.
- The Shape of Space: A Guided Tour of Vectors, Matrices, Tensors and Markov Chains, self-published on Amazon, https://www.amazon.com/dp/B0FQCGRNW1, September 2025.

Electronic versions of my books are available (free of charge) at

https://github.com/sfratini33/art-of-managing-things-external/tree/master/free_books

My Video Series:

- *Linear Algebra*, YouTube videos by the author that follows Section 2 of this book, https://www.youtube.com/playlist?list=PLSalWQAk7_UniISLwX82IFFqN1OsloWPK
- *Matrices - Some Less Common Properties,* YouTube videos by the author that follow Section 3 of this book, https://www.youtube.com/playlist?list=PLSalWQAk7_Umu0emwEcVtjDQs7O4ioZgr
- *Vector and Matrix Calculus*, YouTube videos by the author that follow Section 4 of this book, https://www.youtube.com/playlist?list=PLSalWQAk7_Ull8xLnrTOQaNe-KX6v0EDp
- *Tensors*, YouTube videos by the author that follow Sections 5 and 6 of this book, https://www.youtube.com/playlist?list=PLSalWQAk7_UltOlPJUX7nRHxIZ1RMYJXN
- *Non-negative matrices, Markov chains and Least Squares*, YouTube videos by the author that follow Section 7, 8 and 9 of this book, https://www.youtube.com/playlist?list=PLSalWQAk7_UlvuQ49JQUmV6GZGU7U1voD
- *Puzzles to Exercise the Mind*, https://www.youtube.com/playlist?list=PLSalWQAk7_UmCBhNtgSpmgss0XAJq392j
- *Non-Euclidean Geometry*, https://www.youtube.com/playlist?list=PLSalWQAk7_UnXtByWwyjg9L-0JcA81HK8
- *Topology*, https://www.youtube.com/playlist?list=PLSalWQAk7_Uk62dBW9a6myMpmXrOllR90
- *Complex Analysis*, https://www.youtube.com/playlist?list=PLSalWQAk7_UkA9q_2Ptfz9mK9yzmngTRe
- *Abstract Algebra, https://www.youtube.com/playlist?list=PLSalWQAk7_UmjFOzc39_O5U5o953dJ22N*

If you are working from a paper copy of my book, it may be easier to visit my YouTube channel rather than typing in the long URLs above.

www.youtube.com/@sfratini

1 Introduction

What has been, that will be; what has been done, that will be done. Nothing is new under the sun!

Ecclesiastes 1:9

1.1 Purpose

This book aims to:

- Unify four mathematical frameworks — hyperreal numbers, dual numbers, category theory (via synthetic differential geometry), and automatic differentiation — under the common theme of infinitesimals and their modern applications.
- Bridge intuition and rigor by showing how the intuitive infinitesimal reasoning of Newton and Leibniz can be made rigorous through modern mathematical structures.
- Connect pure mathematics with practical computation by demonstrating how abstract concepts (like hyperreals and categories) lead to powerful computational tools (like automatic differentiation) used in machine learning, physics, and engineering.
- Provide a self-contained introduction to the necessary algebraic and categorical foundations, making advanced topics accessible to motivated readers without prior exposure to non-standard analysis or category theory.
- Illustrate the synergy between different mathematical perspectives — algebraic, analytic, categorical, and computational — in understanding and working with differentiation and smoothness.

1.2 Intended Audience

This book is written for:

- Advanced undergraduate and graduate students in mathematics, computer science, physics, or engineering who want to explore the deep connections between calculus, algebra, and computation.
- Researchers and practitioners in machine learning, scientific computing, or computational physics who want to understand the mathematical foundations of tools like automatic differentiation.
- Mathematically curious readers with a background in calculus and linear algebra who are interested in modern extensions of calculus, including non-standard analysis and synthetic differential geometry.
- Educators who are looking for a resource that connects historical ideas (infinitesimals) with contemporary topics, e.g., automatic differentiation, synthetic differential geometry, in a coherent narrative.
- Self-learners who enjoy exploring abstract mathematics with tangible real-world applications.

1.3 Prerequisites

To fully engage with this book, readers should have:

- A solid understanding of single-variable calculus, including limits, derivatives, and the basic ideas of integration.
- Familiarity with linear algebra, including vectors, matrices, and basic operations.
- Some exposure to abstract algebra (groups, rings, fields) is helpful but not required; Chapter 3 provides a self-contained overview of the necessary concepts.
- Comfort with mathematical reasoning and proofs, including basic set theory and logic.
- No prior knowledge of non-standard analysis, category theory, or automatic differentiation is assumed — these topics are developed from the ground up.

While the book is self-contained in its presentation of advanced topics, readers will benefit most if they have completed a standard undergraduate curriculum in mathematics, physics, computer science, or a related discipline. The later chapters (especially those on synthetic differential geometry and automatic differentiation) build on earlier foundations, so a sequential reading is recommended.

1.4 Outline

Section 1 – Introduction
This section introduces the book's purpose, audience, and structure, and sets the stage for the journey ahead.

Section 2 – Introduction to Infinitesimals and Differentiation
A high-level overview of the four pillars of the book: hyperreal numbers, dual numbers, category theory (via synthetic differential geometry), and automatic differentiation, with a focus on their historical and conceptual connections.

Section 3 – Abstract Algebra
A concise primer on groups, rings, fields, vector spaces, and algebras, providing the algebraic vocabulary needed for the rest of the book.

Section 4 – Extending Number Systems
Explores foundational methods for extending number systems, including Dedekind cuts, complex numbers, and quaternions, as a prelude to hyperreals and duals.

Section 5 – Hyperreal Numbers: Foundations of Non-Standard Analysis
Constructs the hyperreal numbers via the ultrapower method, introduces key properties such as the transfer principle and standard part function, and shows how calculus can be done with infinitesimals.

Section 6 – Dual Numbers: Algebra and Geometry
Defines dual numbers and explores their algebraic structure, geometric interpretation, and role in automatic differentiation.

Section 7 – Comparing Hyperreals and Duals
Highlights the structural similarities and differences between hyperreal and dual number approaches to infinitesimals, with a focus on invertibility vs. nilpotency.

Section 8 – Surreal Numbers (Optional)
An introductory tour of surreal numbers, included for completeness and their remarkable unifying properties, though not required for later chapters.

Section 9 – Weil Algebras
Introduces Weil algebras as a generalization of dual numbers, preparing the ground for synthetic differential geometry.

Section 10 – Basics of Category Theory
A focused introduction to categories, functors, natural transformations, and cartesian closed categories — just enough to understand synthetic differential geometry.

Section 11 – Synthetic Differential Geometry: A Categorical Lens
Presents SDG as an axiomatic approach to calculus using infinitesimals, built on categorical foundations and the Kock–Lawvere axiom.

Section 12 – Automatic Differentiation: Principles and Modes
Covers forward-mode and reverse-mode automatic differentiation, their implementation, categorical interpretations, and applications in machine learning and Bayesian inference.

Appendices – Include acronyms, references, and an index of terms for quick reference.

2 Introduction to Infinitesimals and Differentiation

History repeats itself, but in such cunning disguise that we never detect the resemblance until the damage is done. — Sydney J. Harris

2.1 The Allure of Infinitesimals

Infinitesimals, quantities smaller than any positive real number yet not zero, have fascinated mathematicians for centuries. From their intuitive but imprecise use in the early calculus of Newton and Leibniz to their modern formalizations, infinitesimals offer a powerful lens for understanding limits, derivatives, and smooth structures (like curves and surfaces without sharp corners). This book explores four interconnected frameworks that leverage infinitesimals or related concepts: hyperreal numbers, dual numbers, category theory (via synthetic differential geometry), and automatic differentiation. Together, they form a cohesive narrative bridging pure mathematics, abstract structures, and computational applications.

Historically, infinitesimals were central to the development of calculus. Leibniz's dx and dy were treated as infinitesimally small quantities, allowing derivatives to be computed as ratios. However, 19th century mathematicians like Cauchy[1] and Weierstrass[2] replaced infinitesimals with the $\epsilon - \delta$ definition of limits to achieve rigor, thereby sidelining infinitesimals. In the 20th century, infinitesimals were revived in the form of non-standard analysis (using hyperreal numbers) and formalized in algebraic structures such as dual numbers. Meanwhile, category theory provided an abstract framework to generalize smoothness and derivatives, and automatic differentiation harnessed these ideas for computational efficiency in fields such as machine learning and physics.

This section introduces these four pillars, outlines their synergies, and sets the stage for a unified exploration of infinitesimal-based differentiation.

2.2 Hyperreal Numbers: Infinitesimals in Non-Standard Analysis

Hyperreal numbers, introduced by Abraham Robinson in the 1960s [1], provide a rigorous foundation for infinitesimals via non-standard analysis. The hyperreal field (denoted $\mathbb{R}^*$) extends the real numbers $\mathbb{R}$ by including infinite and infinitesimal quantities. For example, an infinitesimal δ satisfies $0 < |\delta| < r$ for any positive real r, and an infinite number H satisfies $|H| > n$ for any natural number n. These numbers obey the transfer principle (to be discussed in Section 5.5), which ensures that many statements about the real numbers extend to the hyperreal numbers in a consistent way.

[1] Augustin-Louis Cauchy was a French mathematician, engineer, and physicist. He was one of the first to rigorously state and prove the key theorems of calculus (thereby creating real analysis), pioneered the field complex analysis, and the study of permutation groups in abstract algebra.

[2] Karl Weierstrass is renowned as the father of modern mathematical analysis for his rigorous foundational work in calculus, including the epsilon-delta definition of limits and continuity, as well as the development of uniform convergence and tests for series convergence.

Hyperreals allow derivatives to be defined intuitively: for a function $f(x)$, the derivative at x is the standard part of the ratio

$$\frac{f(x+\delta)-f(x)}{\delta}$$

where δ is an infinitesimal. This approach mirrors Leibniz's original intuition but is fully rigorous. This is made rigorous using the standard part function, $st(x)$, which maps a finite hyperreal number to the unique real number infinitely close to it.

For example, consider $f(x)=x^2$. Using infinitesimals, we can compute $f'(x)$ as shown below. Note that st stands for "standard part" (i.e., the part of the expression not involving δ).

$$\frac{df(x,\delta)}{\delta} \equiv st\left(\frac{f(x+\delta)-f(x)}{\delta}\right) = st\left(\frac{x^2+2x\delta+\delta^2-x^2}{\delta}\right)$$

$$= st\left(\frac{2x\delta+\delta^2}{\delta}\right) = st\left(\frac{2x\delta}{\delta}+\frac{\delta^2}{\delta}\right) = st(2x+\delta) = 2x$$

Hyperreals are particularly useful in analysis, offering simplified proofs for theorems in calculus, topology, and even physics.

Figure 1 depicts the hyperreal numbers on either side of the real numbers 0 and 100. The figure also indicates the infinite hyperreals $-\frac{1}{\varepsilon}$ and $\frac{1}{\varepsilon}$. The figure comes from a presentation entitled "Introduction to Hyperreals" [2]. The author of the figure uses ε (as is common). However, we will use δ to distinguish the hyperreal infinitesimal from the dual number ε when the possibility of confusion exists.

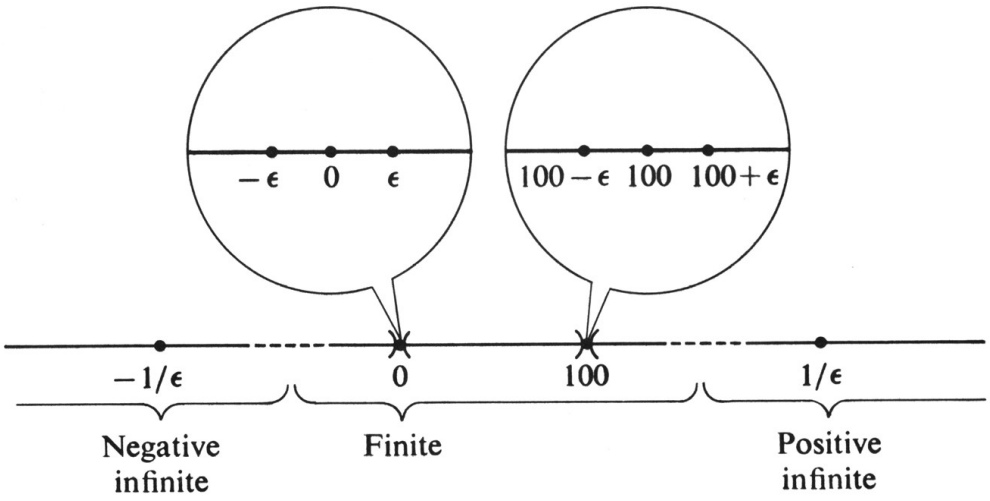


Figure 1. The hyperreals near the reals

2.3 Dual Numbers: Algebraic Infinitesimals

Dual numbers, introduced by William Clifford in the 19th century [3], offer a different approach to infinitesimals. A dual number is of the form $a + b\varepsilon$, where a, b are real numbers and ε is a nilpotent element satisfying $\varepsilon^2 = 0$. The dual numbers form a ring (not a field, due to zero divisors), and their algebraic structure makes them ideal for computing derivatives.

[**Author's Remark**: Concepts such as rings and fields are from a part of mathematics known as abstract algebra. For the reader not familiar with these topics, an introduction to abstract algebra is provided in Section 3 of this book.]

For a function $f(x)$, evaluating $f(x + \varepsilon)$ yields $f(x) + f'(x)\varepsilon$, directly giving the derivative $f'(x)$ as the coefficient of ε. This follows from the Taylor series for $f(x + \varepsilon)$ and the fact that $\varepsilon^2 = 0$.

For example, consider $f(x) = x^3$.

$$f(x + \varepsilon) = (x + \varepsilon)^3 = x^3 + 3x^2\varepsilon + 3x\varepsilon^2 + \varepsilon^3 = x^3 + 3x^2\varepsilon = f(x) + f'(x)\varepsilon$$

So, we can read off the derivative as $3x^2$. (In the above, note that $\varepsilon^3 = \varepsilon\varepsilon^2 = \varepsilon 0 = 0$.)

Dual numbers are computationally efficient and underpin forward-mode automatic differentiation (Section 12.3), where derivatives are calculated alongside function evaluations. Their simplicity and algebraic nature make them a bridge between pure mathematics and applied computation.

2.4 Category Theory: An Abstract Framework for Smoothness

Thus far, we have explored infinitesimals by extending the real numbers into new number systems such as the hyperreals and duals. Now, we will take a more abstract approach using category theory. Think of category theory not as a theory about specific mathematical objects (such as numbers or shapes), but as a "theory of theories", i.e., a universal language for describing mathematical structures and the relationships between them.

Developed by Samuel Eilenberg and Saunders Mac Lane in the 1940s [4], category theory provides a bird's-eye view of mathematics. It allows us to see the deep connections between seemingly different fields, like algebra and geometry, by focusing on the patterns of relationships rather than the internal details of the objects themselves.

The Core Idea: What is a Category?

At its heart, a category is built from two simple concepts:

1. **Objects:** These are the things we are studying. They could be sets, groups, vector spaces, topological spaces, or even types in a programming language.
2. **Morphisms (or Arrows):** These are the relationships or processes between objects. For every two objects A and B, there is a collection of morphisms that go from A to B.

The key is that these morphisms can be composed. If you have a morphism f from A to B, and a morphism g from B to C, then there must be a morphism $g \circ f$ (read "g after f") from A to C. This composition must be associative (the order of grouping doesn't matter), and every object must have a special identity morphism that acts as a do-nothing process.

Analogy**:** Imagine a category where the objects are cities, and the morphisms are possible flight routes. A morphism from New York to London is a flight. You can compose this with a morphism from London to Berlin. The identity morphism for New York is like staying put at the airport in New

York. Category theory is interested in the entire network of flights and connections, rather than the internal geography of each city.

Synthetic Differential Geometry (SDG): A New Foundation for Calculus

This is where category theory meets our quest for infinitesimals. Synthetic Differential Geometry (SDG), pioneered by mathematicians such as Anders Kock and William Lawvere, uses category theory to construct a new universe for doing differential geometry, i.e., a universe where infinitesimals are built directly into the fabric of reality.

In this synthetic universe, we work within a specially chosen category (called a smooth topos) where the objects behave like smooth spaces (generalizations of curves and surfaces) and the morphisms are smooth maps between them.

The revolutionary concept in SDG is to postulate the existence of infinitesimals as fundamental entities. It introduces a special object, often denoted D, which is the set of "nilsquare infinitesimals." These are elements d that satisfy $d^2 = 0$, just like the ε in dual numbers. However, in SDG, D is not defined within a larger number system; it is a basic, axiomatic part of the universe.

The Kock-Lawvere Axiom: The Engine of SDG

The entire power of SDG rests on a simple but profound axiom. It states, roughly, that for any function $f: D \to R$ (from the infinitesimal line D to the real line R in this universe), there exist unique real numbers a and b such that for all infinitesimals d in D:

$$f(d) = a + b \cdot d$$

Let's unpack what this means:

- Any function defined on D is automatically linear. There are no complicated, wiggly functions at the infinitesimal scale. The graph of f over D is always a straight line.
- The slope of this line, b, is the derivative of f at the point a.

In other words, **the derivative is no longer defined as a limit**. It falls out directly from the geometry of this synthetic world. To find the derivative of a function f at a point x, you simply look at how f acts on the infinitesimals around x. The coefficient b that appears is $f'(x)$.

While this behavior is reminiscent of dual numbers, the key difference is foundational: in SDG, the set D and its properties are *axiomatic*, defining the very nature of the smooth world, whereas dual numbers are a specific algebraic *construction* within classical set theory.

Comparison to Dual Numbers: This should feel familiar! In dual numbers, we evaluated $f(x + \varepsilon)$ and found $f(x) + f'(x)\varepsilon$. SDG makes this idea foundational. It's as if the entire universe of SDG is built so that every function behaves like a dual number when you zoom in infinitesimally.

Why Does This Matter?

1. Conceptual Clarity: SDG recaptures the intuitive "infinitesimal" reasoning of Leibniz in a completely rigorous way, free from the cumbersome $\epsilon - \delta$ language of limits. Proofs of fundamental theorems in calculus often become strikingly simple and geometric.
2. A Unifying Language: Category theory provides a powerful framework to connect SDG with other approaches. For instance, the algebra of dual numbers can be seen as a concrete model that fits into the abstract framework of SDG.

3. Connections to Computation: The categorical perspective of SDG has inspired new ways of thinking about Automatic Differentiation (AD). We can view differentiation itself as a functor (a map between categories), providing a deep, structural understanding of why AD algorithms work so well. This will be explored further in Section 12.7.

In summary, Category Theory and Synthetic Differential Geometry offer a radical shift in perspective. They don't just add infinitesimals to an existing system; they build a new mathematical world from the ground up, one where the concepts of smoothness and change are primitive and intuitive. This abstract framework not only justifies the use of infinitesimals but also reveals the beautiful, universal structures that underpin calculus and geometry.

2.5 Automatic Differentiation: Computational Derivatives

We have seen how infinitesimals provide elegant theoretical frameworks for calculus. But how do we actually compute derivatives in the real world, especially for the complex, multi-step functions that arise in modern science and engineering? Symbolic differentiation, as done by hand or with systems like Mathematica, can lead to expression swell and is often inefficient. Numerical differentiation, using finite differences such as

$$\frac{f(x+h)-f(x)}{h}$$

is simple but prone to rounding and truncation errors.

This is where Automatic Differentiation (AD), also called “Auto Diff,” comes in. AD is a computational technique that evaluates the derivative of a function specified by a computer program exactly and with high efficiency, to the limits of machine precision. It is neither a symbolic manipulation nor a finite difference approximation; instead, it cleverly applies the chain rule from calculus in a systematic way.

The Core Principle: The Chain Rule on a Computer

At its heart, AD recognizes that any complex function – from a simple polynomial to a massive neural network – is ultimately a composition of elementary operations (like +, -, *, /, sin, cos, exp, etc.), whose derivatives are known. By breaking the function down into a sequence of these elementary operations, AD can compute the derivative by systematically applying the chain rule through the entire sequence.

Think of it as a computational recipe. To compute $y = f(x)$, a computer must evaluate it step-by-step, creating intermediate variables (e.g., $v_1 = x \cdot x$, $v_2 = \sin(v_1)$, $y = 2 \cdot v_2$). AD simply augments this recipe to simultaneously compute the derivative of each intermediate variable with respect to the input.

The Two Main Modes: Forward and Reverse

AD can be implemented in two primary ways, which are essentially two different directions for traversing the computational recipe.

Forward-Mode AD:

- How it works: We start with the input variable x and its derivative $\dot{x}$ (which is typically 1 if x is the independent variable). Then, as we evaluate each elementary operation in the function, we simultaneously compute the operation on the primary values and its derivative, propagating the derivatives forward.
- The Dual Number Connection: This is where our previous discussion becomes practical. Forward-mode AD is mathematically equivalent to evaluating the function using dual numbers! The primal value is the real part a, and the tangent value is the dual part b in $a + b\varepsilon$. Evaluating $f(x + \varepsilon)$ automatically computes $f(x) + f'(x)\varepsilon$, giving both the function value and its derivative in one pass.
- Best for: Functions with few inputs and many outputs (e.g., computing the Jacobian of a function $f: \mathbb{R}^n \to \mathbb{R}^m$ where n is small).

Reverse-Mode AD:

- How it works: This mode works in two passes. First, a forward pass evaluates the function and records the sequence of operations (the computational graph) and all intermediate values. Then, a backward pass starts from the final output and propagates derivatives backward through the graph to the inputs, applying the chain rule in reverse.
- The Famous Example: This is the algorithm known as backpropagation, which is the engine that powers the training of neural networks. It efficiently computes the gradient of a loss function with respect to millions or even billions of parameters.
- Best for: Functions with many inputs and few outputs (e.g., $f: \mathbb{R}^n \to \mathbb{R}$, which is the common case for machine learning loss functions).

Why AD is a Pillar of This Book

Automatic Differentiation is not just a handy computational trick; it is a profound idea that connects directly to our other pillars:

- It makes Dual Numbers practical. AD provides the computational machinery that brings the algebraic elegance of dual numbers to life in real-world code.
- It has a deep connection to Category Theory. As we will explore in Section 12.7, AD can be viewed through a categorical lens. Forward-mode and reverse-mode can be seen as different functors, providing a unifying, mathematical structure to the process of differentiation itself. This perspective helps in designing new, more efficient AD systems.
- It is the practical realization of infinitesimal reasoning. While hyperreals provide a logical foundation and SDG an axiomatic one, AD delivers a working, efficient implementation of derivative-as-infinitesimal-ratio for the functions that run our world.

In summary, Automatic Differentiation is the bridge from the abstract world of infinitesimals to the concrete world of high-performance computation. It is the workhorse that powers modern machine learning, scientific computing, and optimization, proving that the centuries-old ideas of Leibniz and Newton are more relevant than ever in the algorithmic age. The following chapters will provide the foundational knowledge to understand its inner workings in detail.

2.6 Synergies and Motivations

Having now introduced our four pillars, i.e., Hyperreals, Dual Numbers, Category Theory (via SDG), and Automatic Differentiation, it might seem that they are separate, parallel paths. In reality, they form a deeply interconnected ecosystem. They are not competing frameworks, but rather complementary perspectives that, when viewed together, reveal a much richer picture of how to formalize and compute with infinitesimals. This subsection illuminates the powerful synergies between them, motivating our journey through the rest of this book.

The Theoretical and the Practical: Hyperreals and Dual Numbers

At first glance, hyperreals and dual numbers both offer a home for infinitesimals. However, their core differences are precisely what make their comparison so illuminating:

- Hyperreals: The Analytical Powerhouse. The hyperreal field $\mathbb{R}^*$ is a *theoretical* tool of immense power. Its transfer principle allows us to prove classical theorems with intuitive, infinitesimal arguments and then transfer the result back to the standard real numbers, guaranteeing rigor. It is the framework that most faithfully resurrects the original Leibnizian vision for all of analysis.
- Dual Numbers: The Algebraic Workhorse. The dual number ring $\mathbb{D}$ is a *practical* and *algebraic* tool. Its nilpotency ($\varepsilon^2 = 0$) makes it inherently computational, as it automatically truncates Taylor series after the first-order term. It is not designed to rebuild all of calculus, but to compute derivatives with perfect accuracy and efficiency.

The Synergy: Comparing these two (as we will do in detail in Section 7) highlights a fundamental trade-off: the richness and invertibility of hyperreal infinitesimals make them ideal for theoretical analysis, while the simplicity and nilpotency (i.e., $\varepsilon^2 = 0$) of dual numbers make them ideal for algebraic computation. One provides a universe for doing analysis, the other an algebra for calculating within it.

The Unifying Language: Category Theory and SDG

Category theory acts as the grand unifier in this landscape. It provides the abstract language to see hyperreals, dual numbers, and differentiation itself as specific instances of more general patterns.

- SDG as a "Synthetic" Compromise: Synthetic Differential Geometry (SDG) creates a new, self-contained mathematical universe where infinitesimals are fundamental. The infinitesimals in SDG (d with $d^2 = 0$) behave much like dual numbers. In this sense, SDG can be seen as an axiomatic foundation that justifies the use of dual-like infinitesimals for all of geometry and calculus, providing a third, structural path that is distinct from both the non-standard analysis of hyperreals and the standard limit-based analysis of calculus.
- A Categorical Perspective on AD**:** Category theory doesn't just unify the static structures; it also unifies the dynamic processes. As we will explore in Section 12.7, we can view Automatic Differentiation through a categorical lens. Forward-mode AD can be described as a functor (mapping) from a category of functions to a category of their tangents (a formalization of the dual number idea). Reverse-mode AD fits into a different categorical structure, explaining its unique efficiency for gradients. This abstract perspective is not just academically elegant; it helps computer scientists design better, more general AD systems.

The Computational Realization: Automatic Differentiation

Automatic Differentiation is the point where all these abstract ideas are translated into concrete, transformative computation.

- It is the practical implementation of the dual number approach, made efficient and scalable for complex software.
- It can be modeled and generalized using the categorical frameworks inspired by SDG and category theory.
- While it doesn't use the full power of hyperreals, it operationalizes their core intuition (that derivatives can be computed as exact ratios of infinitesimal changes) in a computationally feasible way.

The Narrative Arc of This Book

This book is structured to follow this interconnected narrative:

1. Foundation (Abstract Algebra): We begin by building the necessary mathematical vocabulary, e.g., groups, rings, fields, vector spaces, to understand the structures we are discussing.
2. The Number Systems (Hyperreals & Duals): We then explore the two primary extensions of the real numbers that incorporate infinitesimals, understanding their construction, properties, and key differences.
3. The Unifying Framework (Category Theory & SDG): Ascending to a higher level of abstraction, we see how category theory and SDG provide a powerful, unified language for smoothness and infinitesimals.
4. The Computational Tool (Automatic Differentiation): Finally, we descend back to the practical, showing how these ideas culminate in the algorithms that power modern machine learning and scientific computing.

This journey, from concrete numbers to abstract categories and back to practical algorithms, demonstrates the remarkable vitality of the infinitesimal concept. It shows how an intuitive but once deprecated idea can be rehabilitated through rigorous mathematics, generalized through abstract structures, and ultimately become the engine of our most advanced computational technologies. This is the powerful synergy that motivates our journey.

3 Abstract Algebra

History doesn't repeat itself, but it does sometimes rhyme.

Attributed to Mark Twain (though unverified)

In the subsequent sections of this book, various concepts from abstract algebra (e.g., structures such as groups, quotient groups, rings and fields) are necessary for understanding the discussion. This section provides a brief overview of such concepts. There is more here that you need to understand the other sections of this book but it is hard to just extract various concepts without losing context.

3.1 Overview

As the name suggests "abstract algebra" is an abstraction (and extension) of concepts from elementary algebra. Some definitions from various sources:

- (Wolfram MathWorld) "Abstract algebra is the set of advanced topics of algebra that deal with abstract algebraic structures rather than the usual number systems. The most important of these structures are groups, rings, and fields."
- (dictionary.com) "Abstract algebra: the branch of mathematics that deals with the extension of algebraic concepts usually associated with the real number system to other, more general systems."

From Google's AI application, when asked for the definition of "abstract algebra":

"Abstract algebra is the branch of mathematics that studies the properties of algebraic structures, such as groups, rings, and fields. ***These structures are abstract in the sense that they are not tied to any particular set of numbers or objects. Instead, they are defined by their properties****, which can be studied independently of any specific implementation.*

Abstract algebra is a vast and complex subject, but it is also one of the most important branches of mathematics. It has applications in many other areas of mathematics, as well as in physics, chemistry, computer science, and other fields.

Some of the most important concepts in abstract algebra include:

- *Groups: A group is a set of elements together with an operation that combines any two elements of the set to produce another element of the set. The operation must satisfy certain properties, such as associativity, commutativity, and the existence of an identity element.*
- *Rings: A ring is a set of elements together with two operations, addition and multiplication, that satisfy certain properties. The properties of a ring are similar to those of a group, but with the additional requirement that multiplication is associative and distributes over addition.*
- *Fields: A field is a ring in which every nonzero element has an inverse under multiplication. Fields are the most important algebraic structures, and they have many applications in mathematics and other areas.*

Abstract algebra is a challenging but rewarding subject. It requires a high level of mathematical maturity and abstract thinking, but it can provide deep insights into the nature of mathematics."

In terms of preliminaries, it is assumed that the reader is familiar with the basics of set theory, number theory and permutations. Much of the assumed background material can be found, for example, in the textbook by Judson [6].

Group theory is a vast topic. This section is a very brief summary of some key concepts. A short but good video (from 1992, using old presentation technology) is recommended as a starting point before reading this section, see Part 1 of Neumann [7].

3.2 Groups

3.2.1 Definition

Consider the set of integers, i.e., $\{\dots, -3, -2, -1, 0, 1, 2, 3, \dots\}$.

If we add two integers, we necessarily get another integer, e.g., $(-3) + 7 = 4$. In general, a set having this property is said to be closed under addition. **Warning**: there is only one operation being considered here, i.e., addition. Subtraction is not an operation but rather a sign appended to a number.

The order of addition does not matter, e.g.,

$$2 + (-4) = (-4) + 2 = -2$$

This is known as the commutative property.

With respect to the addition of integers, grouping does not change the result, e.g.,

$$2 + (5 + (-11)) = (2 + 5) + (-11) = -4$$

This is known as the associative property.

The sum of any integer with 0 is the integer itself, e.g., $(-5) + 0 = -5$. With respect to integer addition, 0 is known as the identity element.

For every integer, there exists another number such that when the two numbers are added together, the result is 0, e.g., $2 + (-2) = 0$. This is the property of each number in a set having an inverse.

Many other sets (with an associated binary operation) follow the properties listed above for the integers with the binary operation $+$. In general, a **binary operation** * on a set G, is an operation that takes any two elements $a, b \in G$ and yields another element of G, i.e., $a * b \in G$. This property is known as **closure**.

A **group** is a set G and binary operation *, with the following properties:

- the associative law holds, i.e., for every $a, b, c \in G$, $(a * b) * c = a * (b * c)$
- there exists a special element e, called the identity, such that $e * a = a$ for every $a \in G$.
    - Sometimes 0 in the case of additive groups, and 1 in the case of multiplicative groups are used in lieu of e.

- for every $a \in G$, there exist an inverse element $a' \in G$ such that $a' * a = e$.
    - The alternate notation a^{-1} is often used to represent the inverse of a. In what follows, we sometimes use this alternate notation.
    - It is also true for groups that $a * a' = e$, and we prove this in Theorem 3.

Further, if $a * b = b * a$ for every $a, b \in G$, then G is said to be a **commutative** (or abelian) group.

A group is represented as the pair $(G,*)$.

A group with a finite number of elements is said to be a finite group. In this case, the number of elements in such a group is known as its **order**. The order of a finite group G is written as $|G|$.

3.2.2 Examples

As we saw, the set of integers (represented by the symbol $\mathbb{Z}$) under addition is a commutative group. The set of real numbers (represented by the symbol $\mathbb{R}$) and the set of rational numbers (represented by the symbol $\mathbb{Q}$) are also commutative groups under addition.

These sets (with 0 removed) are written as $\mathbb{R}\backslash\{0\}$ (reads as the set of real number with 0 removed), $\mathbb{Q}\backslash\{0\}$ and $\mathbb{Z}\{0\}$. The backslash (in this context) should be interpreted as set subtraction.

If we consider the operation of multiplication, we have an issue with 0 since there is no multiplicative inverse of 0. However, if we remove 0, then the rational numbers and real numbers are commutative groups under the operation of multiplication. In these cases, the identity element is the number 1.

Another example comes from modular arithmetic under addition [8]. The integers modulo n (represented by the symbol $\mathbb{Z}_n$) under modulo n addition form a commutative group. $\mathbb{Z}_n = \{\overline{0}, \overline{1}, \overline{2}, \dots, \overline{n-1}\}$ where addition on $\mathbb{Z}_n$ is defined as the remainder upon division by n. For example, consider $\mathbb{Z}_5 = \{\overline{0}, \overline{1}, \overline{2}, \overline{3}, \overline{4}\}$. If we add $\overline{3}$ and $\overline{4}$, the result is $\overline{2}$ since the remainder of $3 + 4 = 7$ is 2 when dividing by 5. In $\mathbb{Z}_5$, the inverse of $\overline{0}$ is itself, the inverse of $\overline{1}$ is $\overline{4}$, and the inverse of $\overline{2}$ is $\overline{3}$.

Modular arithmetic under multiplication works in a similar manner to modular arithmetic under addition, i.e., multiply two numbers and then take the remainder. For example, consider $\mathbb{Z}_6 = \{\overline{0}, \overline{1}, \overline{2}, \overline{3}, \overline{4}, \overline{5}\}$. If we multiply $\overline{2}$ times $\overline{4}$ we get $\overline{2}$ since the remainder of 8 when divided by 6 is 2. Is $\mathbb{Z}_6$ a multiplicative group? It is closed under multiplication and we do have a multiplicative identity, i.e., $\overline{1}$. The associative and commutative laws also hold true. However, there is a problem with inverses, i.e., $\overline{0}, \overline{2}, \overline{3}$, and $\overline{4}$ do not have multiplicative inverses. The general solution to this problem is to restrict $\mathbb{Z}_n$ to only those elements that are relatively prime to n. Under such a restriction $(\mathbb{Z}_n,\times)$ is known as the **multiplicative group of integers modulo n** [9]. (The symbol $\times$ is used to represent modular multiplication, e.g., we write $\overline{2} \times \overline{5} = \overline{4}$ in $\mathbb{Z}_6$.) For example, the multiplicative group $(\mathbb{Z}_{10},\times)$ has elements $\{\overline{1}, \overline{3}, \overline{7}, \overline{9}\}$. The inverse of $\overline{1}$ is itself and the inverse of $\overline{9}$ is also itself. $\overline{3}$ and $\overline{7}$ are inverses of each other. If we take consecutive powers of $\overline{3}$, we get $\overline{3}, \overline{9}, \overline{7}, \overline{1}$. An element of a group whose powers cover every element in the group is known as a **generator** of the group. For $(\mathbb{Z}_{10},\times)$, $\overline{3}$ is the generator.

As a shorthand, we sometime use the symbol $\mathbb{Z}_n^+$ to represent the additive group modulo n, and $\mathbb{Z}_n^\times$ to represent the multiplicative group modulo n.

The **symmetric group** [10], defined over any set, is the group whose elements are all the one-one functions from the given set onto itself (i.e., a bijective mapping [11]), and whose group operation is the composition of functions. In particular, the finite symmetric group S_n is defined as all permutations on the set $\{1, 2, \dots, n\}$. So, S_n has $n!$ elements.

All finite groups can be mapped to a subset of S_n for some positive integer n.

To check that the symmetric group on n elements is in fact a group, we need to verify the group axioms, i.e., closure, associativity, identity, and existence of an inverse for every element.

- The operation of function composition is closed in the set of permutations of the given set $\{1, 2, \dots, n\}$.
- Function composition (not just for permutations, but in general) is associative.
- The permutation that assigns each element to itself is the identity permutation.
- Every bijection has an inverse function that undoes its action, and thus each element of a symmetric group does have an inverse which is also permutation.

As an example, consider S_5. The following is one of the 120 elements in S_5

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 3 & 4 & 1 & 5 & 2 \end{pmatrix} = (1,3)(2,4,5)$$

The permutation is written in two different (by equivalent) ways in the above example. On the left, we explicitly list the mapping for each element, i.e., 1 to 3, 2 to 4, 3 to 1, 4 to 5 and 5 to 2. On the right, the permutation is divided into component cycles, i.e., $1 \to 3 \to 1$ and $2 \to 4 \to 5 \to 2$.

When written in the cyclic notation on the right, it is understood that one "wraps-around" at the end of the cycle. For example, $(2,4,5)$ represents the mapping $2 \to 4 \to 5 \to 2$.

The identity element for S_5 is

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 1 & 2 & 3 & 4 & 5 \end{pmatrix} = (1)(2)(3)(4)(5)$$

The inverse of an element is the reverse of an element, e.g.,

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 3 & 4 & 1 & 5 & 2 \end{pmatrix} \circ \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 3 & 5 & 1 & 2 & 4 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 1 & 2 & 3 & 4 & 5 \end{pmatrix}$$

Note: we go from left to right when multiplying two permutations. Some books and articles on this topic do multiplication from right to left.

The following shows the multiplication of two elements of S_5, with the order of multiplication reversed in the second line. Since the result of multiplication depends on the order of the terms, S_5 is not a commutative group.

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 3 & 4 & 1 & 5 & 2 \end{pmatrix} \circ \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 4 & 3 & 5 & 1 & 2 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 5 & 1 & 4 & 2 & 3 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 4 & 3 & 5 & 1 & 2 \end{pmatrix} \circ \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 3 & 4 & 1 & 5 & 2 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 5 & 1 & 2 & 3 & 4 \end{pmatrix}$$

For $n \geq 2$, S_n is a non-abelian (i.e., non-commutative) group.

Every permutation can be broken down into the product of one or more cycles. For example, consider the following permutation

$$\begin{pmatrix}1&2&3&4&5&6&7&8&9&10&11\\9&7&4&1&2&5&6&8&11&3&10\end{pmatrix} = (1,9,11,10,3,4)(2,7,6,5)(8)$$

If the cycles are independent of each other (i.e., don't share any numbers), then the order does not matter. So, for example, the above permutation also equals $(2,7,6,5)(8)(1,9,11,10,3,4)$.

It is always possible to decompose a permutation into a product of two-cycles, and one-cycles. For example, the permutation $\sigma = (1,3,2,5,4,6)$ can be written as

$$(1,3)(1,2)(1,5)(1,4)(1,6)$$

This may be hard to process upon seeing for the first time. So, let's expand the above multiplications in detail.

$$(1,3)(1,2) = \begin{pmatrix}1&2&3&4&5&6\\3&2&1&4&5&6\end{pmatrix} \circ \begin{pmatrix}1&2&3&4&5&6\\2&1&3&4&5&6\end{pmatrix} = \begin{pmatrix}1&2&3&4&5&6\\3&1&2&4&5&6\end{pmatrix}$$

$$(1,3)(1,2)(1,5) = \begin{pmatrix}1&2&3&4&5&6\\3&1&2&4&5&6\end{pmatrix} \circ \begin{pmatrix}1&2&3&4&5&6\\5&2&3&4&1&6\end{pmatrix} = \begin{pmatrix}1&2&3&4&5&6\\3&5&2&4&1&6\end{pmatrix}$$

$$(1,3)(1,2)(1,5)(1,4) = \begin{pmatrix}1&2&3&4&5&6\\3&5&2&4&1&6\end{pmatrix} \circ \begin{pmatrix}1&2&3&4&5&6\\4&2&3&1&5&6\end{pmatrix} = \begin{pmatrix}1&2&3&4&5&6\\3&5&2&1&4&6\end{pmatrix}$$

$$(1,3)(1,2)(1,5)(1,4)(1,6) = \begin{pmatrix}1&2&3&4&5&6\\3&5&2&1&4&6\end{pmatrix} \circ \begin{pmatrix}1&2&3&4&5&6\\6&2&3&4&5&1\end{pmatrix} = \begin{pmatrix}1&2&3&4&5&6\\3&5&2&6&4&1\end{pmatrix} = (1,3,2,5,4,6)$$

In general, any cycle can be decomposed as follows (assuming left to right order of composition):

$$(a_1, a_2, a_3, \dots, a_n) = (a_1, a_2)(a_1, a_3) \dots (a_1, a_n) = (a_1, a_2)(a_2, a_3) \dots (a_{n-1}, a_n)$$

Other 2-cycle decompositions are possible, e.g., $(a_1, a_n)(a_2, a_n) \dots (a_{n-1}, a_n)$.

So, every permutation of a finite set can be expressed as the product of 2-cycles (known as **transpositions**). While several such expressions for a given permutation may exist, either all contain an even number of transpositions or they all contain an odd number of transpositions (for a proof of this fact, see Theorem 2.40 of Rotman [17]). Thus, all permutations can be classified as even or odd depending on the number of transpositions in its cyclic decomposition.

Some examples

- $(1,3,2,5,4,6) = (1,3)(1,2)(1,5)(1,4)(1,6)$ is an odd permutation.
- $\begin{pmatrix}1&2&3&4&5&6&7&8&9&10&11\\9&7&4&1&2&5&6&8&11&3&10\end{pmatrix} = (1,9,11,10,3,4)(2,7,6,5)(8) =$ $(1,9)(1,11)(1,10)(1,3)(1,4)(2,7)(2,6)(2,5)(8)$ is an even permutation. Cycles of length 1 (8 in this case) are not counted in the parity calculation.

The subset of S_n containing all the even permutations over a set of n elements is known as the **alternating group** of degree n, and is denoted by A_n [12].

Theorem 1. The number of even permutations in S_n is equal to the number of odd permutations and so, A_n has $\frac{n!}{2}$ elements.

Proof: See Proposition 5.17 of Judson [6]. ■

A_n is a group.

- If we multiple two even permutations, we get an even number of transpositions and so, closure holds true.
- The associative law is inherited from S_n.
- The identity permutation has 0 transpositions (i.e., is an even permutation) and therefore, is in A_n.
- We know that each element of A_n has an inverse, since A_n is a subset of S_n and S_n is a group. The question is whether the inverse of an element in A_n is also an even permutation. To see this, we decompose an element $\alpha \in A_n$ into a product of disjoint cycles and single permutations. Let $\alpha = \sigma_1\sigma_2 \dots \sigma_k(x_1)(x_2) \dots (x_h)$ where the σ_i terms are cycles and the x_i terms are single permutations (e.g., $4 \to 4$). There is no need to invert the single permutations. Let's look at a general cycle, i.e., $\beta = (a_1, a_2, a_3, \dots, a_n)$. Its inverse is $\beta' = (a_n, a_{n-1}, a_{n-2}, \dots, a_3, a_2, a_1)$. β' can be decomposed into the same number of transpositions as β. So, the inverse of a cycle has the same parity as the original permutation. The inverse of α is $\sigma_1'\sigma_2' \dots \sigma_k'(x_1)(x_2) \dots (x_h)$ and each of the σ_i' terms has the same parity as the corresponding σ_i terms. Thus, the parity of α' is the same as that of α (which as even since $\alpha \in A_n$).

. . .

A **dihedral group** [13] is the group of symmetries of a regular polygon, including rotations and reflections. A regular polygon with n sides (i.e., an n-gon) has $2n$ different symmetries: n rotational symmetries (in increments of $\frac{360}{n}$ degrees) and n reflection symmetries. The set of rotations and reflections make up the dihedral group D_n. (**Warning**: Since the group of symmetries on a regular n-gon is of order $2n$, some sources call this group D_{2n}.)

- If n is odd, each axis of symmetry (with respect to reflection) connects the midpoint of one side to the opposite vertex (for a total of n axes of symmetry).
- If n is even, there are $\frac{n}{2}$ axes of symmetry (with respect to reflection) connecting the midpoints of opposite sides, and $\frac{n}{2}$ axes of symmetry connecting opposite vertices.

In either case, there are n axes of symmetry (with respect to reflection).

The dihedral group D_3 consists of the following operations on an equilateral triangle:

- The 3 rotations of an equilateral triangle, i.e., 0 degrees rotation (the identity element e), 120 degrees clockwise rotation (denoted by a) and 240 degrees clockwise rotation (denoted by a^2; applying a twice to the triangle).
- The three reflections about the axis between a vertex and midpoint of the opposite side. If we let b be one of the 180 degrees reflections of the triangle, the other two reflections can be gotten by a combination of rotations and a reflection, i.e., ab and a^2b.

With the above conventions, we have 6 group elements, i.e., e, a, a^2, b, ab, a^2b. The various combinations of multiplying the elements (basically function composition) are shown in Table 1 (which is an example of what is called a Cayley table). The elements of the group are listed in the top row and left column. It works just like a multiplication table for numbers. The following identities are used in the table: $a^3 = e, a^4 = a, b^2 = e, ba = a^2b, ba^2 = ab$.

D_3 is a non-abelian group, e.g., $ab \neq ba$.

Table 1. Cayley table for D_3

$\circ$	$\boldsymbol{e}$	$\boldsymbol{a}$	$\boldsymbol{a^2}$	$\boldsymbol{b}$	$\boldsymbol{ab}$	$\boldsymbol{a^2b}$
$\boldsymbol{e}$	e	a	a^2	b	ab	a^2b
$\boldsymbol{a}$	a	a^2	e	ab	a^2b	b
$\boldsymbol{a^2}$	a^2	e	a	a^2b	b	ab
$\boldsymbol{b}$	b	a^2b	ab	e	a^2	a
$\boldsymbol{ab}$	ab	b	a^2b	a	e	a^2
$\boldsymbol{a^2b}$	a^2b	ab	b	a^2	a	e

. . .

The set of all invertible matrices of size $n \times n$ with elements from the real numbers form a non-abelian group. This group is known as the **general linear group** [14] of degree n over the real numbers, and is denote by $GL_n(\mathbb{R})$. We can also define a general linear group over the rational numbers $GL_n(\mathbb{Q})$ or the complex numbers $GL_n(\mathbb{C})$. In general, one can define general linear groups over any field F. Such groups are denoted $GL_n(F)$. We will discuss fields in Section 3.4. The real numbers, rational numbers and complex numbers are examples of infinite fields. We have already seen an example of a finite field, i.e., $\mathbb{Z}_p$ the integers modulo p where p is a prime. Regardless of the underlying field F, all matrices in $GL_n(F)$ must have non-zero determinant (which implies the inverse of the matrix exists) [15]. For those familiar with linear algebra, the term "linear" in "general linear group" refers to the rows (columns) being linearly independent.

The **special linear group** [16], written $SL_n(F)$, is the set of matrices (with a determinant of 1) over a field F. $SL_n(F)$ is a subgroup within $GL_n(F)$.

3.2.3 Some Basic Theorems

The proof of the following theorem is included to illustrate usage of the properties of a group.

Theorem 2. If $\boldsymbol{a * a = a}$ in a group $(\boldsymbol{G}, *)$, then $\boldsymbol{a = e}$.

Proof: Apply a' (i.e., the inverse of a) to both sides of the given equation to get

$$a' * (a * a) = a' * a$$

Apply the associative law to the left-side of the previous equation and the inverse property to the right side of the equation to get

$$(a' * a) * a = e$$

Applying the inverse property to the left-side of the equation, we get $e * a = e$. Finally, apply the identity property to the left-side of the equation to get the desired result of $a = e$. ■

Even in non-abelian groups, inverses commute, and the identity element commutes with all elements.

Theorem 3. In a group $(G,*)$***,*** $a * a' = e$***, and*** $a * e = a$ ***for every*** $a \in G$***.***

Proof: From the inverse property, we have $a' * a = e$. Multiply both sides on the left by a to get

$$a * (a' * a) = a * e$$

Applying the associative law to the left-side of the above equation yields

$$(a * a') * a = a * e$$

Multiply both sides on the right by a' to get

$$(a * a') * (a * a') = a * (e * a') = (a * a')$$

Applying Theorem 2 to the above, we have that $a * a' = e$.

We use the first part of the theorem to prove the second part, i.e.,

$$a * e = a * (a' * a) = (a * a') * a = e * a = a$$

■

In the next theorem, we show that the identity element is unique for a given group, and that there is only one inverse for each element of a group.

Theorem 4. For a group $(G,*)$***, the following statements hold true***

i. ***If*** $f \in G$ ***satisfies*** $f * a = a$ ***for every*** $a \in G$***, then*** $f = e$***.***

ii. ***For a given*** $a \in G$***, if*** $b * a = e$***,*** $b = a'$***.***

Proof:

For the first part of the theorem, the given condition implies that $f * f = f$ and by Theorem 2, we have that $f = e$.

For the second part of the theorem, we use Theorem 3 and the associative law as follows:

$$b = b * e = b * (a * a')$$
$$= (b * a) * a' = e * a' = a'$$

■

Theorem 5. For a group $(G,*)$***, the following statements hold true***

i. ***If either*** $x * a = x * b$ ***or*** $a * x = b * x$***, then*** $a = b$ ***(cancellation law).***

ii. $(a')' = a$ ***(inverse of the inverse returns the original element)***

iii. $(a_1 * a_2 * \ldots * a_n)' = a_n' * \ldots * a_2' * a_1'$

Proof:

(i) We have the following

$$a = e * a = (x' * x) * a = x' * (x * a)$$
$$= x' * (x * b) = (x' * x) * b = e * b = b$$

The proof for the other case follows in a similar manner.

(ii) By Theorem 4(ii), inverses are unique and so $(a')'$ is the unique inverse of a', i.e., $(a')'$ is the unique element of G such that $(a')' * a' = e$. By Theorem 3, $a * a' = e$ and thus, it must be that $(a')' = a$.

(iii) We show the result for $n = 2$. The general result follows by mathematical induction.

By application of the associative law, we have

$$(a * b) * (b' * a') = \big(a * (b * b')\big) * a' = (a * e) * a' = a * a' = e$$

By Theorem 4(ii), $(a * b)' = b' * a'$. ■

[**Author's Remark**: The preceding theorems may seem a bit tedious and unnecessary (i.e., why not just assume these results as part of the definition of a group?). The point, however, is to define a group with the minimal set of properties. This approach (i.e., minimal definitions) is a general goal in mathematics.]

If one has a string of operations in a group, e.g., $a_1 * a_2 * a_3 * \ldots * a_{n-1} * a_n$, parentheses are not needed. One can group the terms in any manner and the result will be the same. This property is known as **generalized associativity**. For a proof of this fact, see Theorem 2.49 in Rotman [17].

. . .

We can define **exponentiation for groups** in a very similar way to basic algebra for real numbers. For a group $(G,*)$ and $a \in G$, the n^{th} power of a is defined as $a^n = a * a * \ldots * a$ where a is repeated n times. The following laws of exponentiation hold:

- $a^{m+n} = a^m * a^n$
- $(a^m)^n = a^{mn}$
- If a and b commute, then $(ab)^n = a^n * b^n$

If we write the inverse of $a \in G$ as a^{-1} in lieu of our previous notation a', then the above expressions hold for negative as well as positive integers. For example,

$$(a^{-3})^2 = (a^{-1} * a^{-1} * a^{-1})^2 = a^{-1} * a^{-1} * a^{-1} * a^{-1} * a^{-1} * a^{-1} = a^{-6}$$

When talking about groups and exponents, it is more natural to use the alternate notation for the identity element, i.e., use 1 instead of e. For example, we can write $a^1 a^{-1} = a^0 = 1$.

The **order** of $a \in G$ is defined to be the smallest integer k such that $a^k = 1$.

3.2.4 Subgroups

As the name suggests, a **subgroup** H of a group G (written as $H \leq G$) is a subset of G that fulfills the group axioms. Since the associative law is inherited by all subsets of a group, we only need to show the closure, identity and inverse axioms.

A group always has at least two trivial subgroups, i.e., the group itself, and the subgroup consisting of only the identity element. Not very interesting, but this is important to note with regard to various proofs.

We have already seen a non-trivial example, i.e., the alternating group A_n is a subgroup of the symmetric group S_n.

Consider the group $\mathbb{Z}_8^+$ (integers modulo 8 under addition). The powers of $\overline{3}$, which generate the set $\{\overline{1}, \overline{3}\}$, form a subgroup. Also, the powers of $\overline{5}$ form the subgroup $\{\overline{1}, \overline{5}\}$.

The following theorem gives us another way of establishing that a given subset of a group is a subgroup.

Theorem 6. A subset H of a group G is a subgroup if and only if H is non-empty and, whenever $x, y \in H$, then $xy^{-1} \in H$.

Proof: If $H \leq G$, and $x, y \in H$, then $y^{-1} \in H$ and by the closure property, $xy^{-1} \in H$. Further, since $H \leq G$, $1 \in H$ and thus, H is non-empty.

Going in the other direction, assume $x, y \in H$ implies $xy^{-1} \in H$, and that H is non-empty.

Since H is non-empty, there exist $x \in H$. The assumption implies that $xx^{-1} = 1 \in H$. So, the identity property holds.

The associative law is inherited by H.

Since $1, x \in H$, our assumption implies $1x^{-1} = x^{-1} \in H$. Thus, an inverse exists for each element in H.

If $x, y \in H$, then $x, y^{-1} \in H$ (since we just showed inverses exist). Using the assumption, we have that $x(y^{-1})^{-1} = xy \in H$, which establishes the closure property. ■

For finite groups, we have an even easier way of determining if a subset is a subgroup, but first we need the following intermediate (but important) result.

Theorem 7. If G is a finite group with element a, then $a^k = 1$ for some integer $k \geq 1$.

Proof: Consider the sequence

$$1, a, a^2, a^3, \ldots$$

Since G is finite, the terms in the above (infinite) sequence cannot all be unique. So, there must exist integers m and n such that $m > n$ and $a^m = a^n$. Using the laws of exponents, we have

$$1 = a^{-n}a^n = a^{-n}a^m = a^{m-n}$$

So, $k = m - n$ fulfills the theorem. ■

Theorem 8. A non-empty subset of a finite group is a subgroup if it is closed under the binary operation of the group.

Proof: Let H be a subset of a finite group G.

H inherits associativity from G.

Since H is not empty, it has at least one element (call it a). Since H is closed, all powers of a are in H. From Theorem 7, there exists a positive integer k such that $a^k = 1$. Thus, $1 \in H$.

For any $h \in H$, we know by Theorem 7 that there exists a positive integer n such that $h^n = 1$. Since G is a group and h is also in G, we know that h^{-1} exists. Multiple h^{-1} to both sides of the equation $h^n = 1$ to get $h^{-1}h^n = h^{n-1} = h^{-1}$. So, $h^{-1} \in H$. ■

. . .

For a group G and $a \in G$, the set of all powers of a is written as

$$\langle a \rangle = \{a^n : n \in \mathbb{Z}\}$$

Since $a^0 = 1$, the identity is in $\langle a \rangle$. Since positive and negative powers of a are included, we have inverses. Closure and associativity clearly hold true. So, $\langle a \rangle \leq G$. In particular, $\langle a \rangle$ is referred to as the **cyclic subgroup** of G generated by a.

A group G is a **cyclic group** if there exists $a \in G$ such that $G = \langle a \rangle$. In this case, a is known as a generator of G (there could be more than one generator).

The next theorem characterizes the generators of a finite cyclic group. The expression $\gcd(a, b)$ means the greatest common divisor of a and b, e.g., $\gcd(14,49) = 7$.

Theorem 9. If $G = \langle a \rangle$ is a cyclic group of order n, then a^k is a generator of G if and only if $gcd(n, k) = 1$, i.e., n and k are relatively prime.

Proof: See Proposition 2.69 of Rotman [17]. ■

Theorem 10. All cyclic groups are abelian (i.e., commutative).

Proof: Let a be the generator of group G. Take any $x, y \in G$. There exist integers m and n such that $x = a^m$ and $y = a^n$. Using the exponentiation rules for groups, we have $xy = a^m a^n = a^{m+n} = a^n a^m = yx$. ■

. . .

We have the following very powerful (and fundamental) theorem of group theory.

Theorem 11 (Lagrange's Theorem) If $H \leq G$, the $|H|$ divides $|G|$.

Clearly, this only applies to groups of finite order.

Proof: See Lagrange's theorem (group theory) [18].

Lagrange's theorem makes the proof of the following theorem rather simple.

Theorem 12. All groups of prime order are cyclic groups.

Proof: Assume that $|G| = p$ is a prime number. Choose $a \in G$, such that $a \neq 1$, and let $H = \langle a \rangle$ be the cyclic subgroup of G generated by a. By Lagrange's theorem, $|H|$ is a divisor of $|G| = p$. Since p is a prime and $|H| > 1$, it must be that $|H| = p$, and thus, $H = G$, i.e., G is a cyclic group generated by a. ■

3.2.5 Group Structure

A basic concept used in the study of group structure is that of a coset.

Given a subgroup H of a group G and $a \in G$, then the (left) **coset** aH is defined as

$$aH = \{ah : h \in H\}$$

It is common to use juxtaposition (i.e., aH) to indicate a coset rather than $a * H$. However, when the group operation is addition, the notation $a + H$ is more common. The coset aH is clearly a subset of G. Further, one can also define a right coset Ha in an analogous manner.

For example, consider the group of integers under addition and the subgroup consisting of multiples of three, i.e., $H = \{\dots, -9, -6, -3, 0, 3, 6, 9, \dots\}$. There are only three distinct cosets of H, i.e.,

$$0 + H = \{\dots, -9, -6, -3, 0, 3, 6, 9, \dots\}$$
$$1 + H = \{\dots, -8, -5, -2, 1, 4, 7, 10, \dots\}$$
$$2 + H = \{\dots, -7, -4, -1, 2, 5, 8, 11, \dots\}$$

Notice that the union of the cosets equals all of G. As we shall see, this is always true.

Our second example involves the group $S_3 = \{e, (1,2), (1,3), (2,3), (1,2,3), (1,3,2)\}$, and the subgroup generated by the powers of $(1,3)$, i.e., $H = \{e, (1,3)\}$. [Note that e is the identity permutation. In terms of single cycles, it is $(1)(2)(3)$.]

The left cosets of H are

$$eH = (1,3)H = \{e, (1,3)\}$$
$$(1,2)H = (1,2,3)H = \{(1,2), (1,2,3)\}$$
$$(2,3)H = (1,3,2)H = \{(2,3), (1,3,2)\}$$

The right cosets of H are

$$He = H(1,3) = \{e, (1,3)\}$$
$$H(1,2) = H(1,3,2) = \{(1,2), (1,3,2)\}$$
$$H(1,2,3) = H(2,3) = \{(2,3), (1,2,3)\}$$

So, we see that left and right cosets using the same element from G are not necessarily equal.

For our third example, consider the subset $H = \{e, b\}$ of the dihedral group D_3 (see Table 1). The left cosets of H are

$$eH = bH = \{e, b\}$$
$$aH = abH = \{a, ab\}$$
$$a^2H = a^2bH = \{a^2, a^2b\}$$

The right cosets of H are

$$He = Hb = \{e, b\}$$
$$Ha = Ha^2b = \{a, ba\} = \{a, a^2b\}$$
$$Ha^2 = Hab = \{a^2, ba^2\} = \{a^2, ab\}$$

Again, the left and right cosets are not equal in all cases, $aH \neq Ha$ and $a^2bH \neq Ha^2b$.

On the other hand, all the right and left cosets of $H = \{e, a, a^2\}$ are equal, i.e.,

$$eH = aH = a^2H = \{e, a, a^2\} = Ha^2 = Ha = He$$
$$bH = abH = a^2bH = \{b, a^2b, ab\} = Ha^2b = Hab = Hb$$

. . .

The index of a subgroup $H \leq G$, denoted $[G:H]$, is defined to be the number of cosets of H in G.

Theorem 13. If G is a finite group and $H \leq G$, then $[G:H] = \frac{|G|}{|H|}$.

Proof: See Corollary 2.82 of Rotman [17].

. . .

The case where all right and left cosets are equal for a given subgroup of a group is critical in the study of group structure.

A subgroup N of a group G is a **normal subgroup** of G if and only if for every $g \in G$ the corresponding left and right cosets are equal, i.e., $gN = Ng$. In terms of notation, we write $N \triangleleft G$ when N is a normal subgroup of G.

An equivalent definition states that N is a normal subgroup of G if for every $g \in G$ and $x \in N$, implies $gxg^{-1} \in N$. This is sometimes stated as $N = gNg^{-1}$ for every $g \in G$. If G is an abelian group, then every subgroup N is normal, since if $x \in N$ and $g \in G$, then $gxg^{-1} = xgg^{-1} = x \in N$.

For example, $H = \{e, a, a^2\}$ is a normal subgroup of D_3 (as we saw in the previous example).

The special linear group over field F of degree n, i.e., **$SL_n(F)$ is a normal subgroup of $GL_n(F)$**.

Proof (for those familiar with linear algebra): For any $A \in SL_n(F)$ and $B \in GL_n(F)$, we have that

$$\det(BAB^{-1}) = \det(B)\det(A)\det(B^{-1}) = \det(A) = 1$$

since the determinant of the product of matrices is the product of the determinant of each matrix, and $\det(C^{-1}) = \frac{1}{\det(c)}$, when $\det(C)$ is non-zero, i.e., when C is invertible [19]. So, $BAB^{-1} \in SL_n(F)$ and thus, $SL_n(F)$ is a normal subgroup of $GL_n(F)$. ■

A_n is a normal subgroup of S_n.

Proof: The product of two odd or two even permutations is even. The product of an odd permutation and an even permutation is odd. By definition, A_n consists of all the even permutations.

- If $g \in A_n$, then g^{-1} (being an even permutation) is also in A_n. For $x \in A_n$, gxg^{-1} is also even and thus, in A_n.
- If $g \in S_n \backslash A_n$ (i.e., in S_n but not in A_n, or in other words, g is an odd permutation), the g^{-1} is also an odd permutation. For $x \in A_n$, gxg^{-1} (being the product of an odd times an even times an odd permutation) is an even permutation and thus, in A_n. ■

Theorem 14. If $H \leq G$ and $[G:H] = 2$, then $H \triangleleft G$.

Proof: Since H is a subgroup, it is closed under the group operation. Thus, $xH = H$ for every $x \in H$. This gives us one coset. We are given that $[G:H] = 2$, and so, there is one other coset with respect to H. This coset must be of the form aH where a is any element in G but not in H. Further, since $[G:H] = 2$, H is both a left and right coset, $aH = Ha$. We have two cases:

- For $g \notin H$, we have $gH = aH = Ha = Hg$ (since $aH = Ha$ and $g \in aH$), and so, $gHg^{-1} = H$.
- For $g \in H$, we have $gH = H = Hg$ and so, $gHg^{-1} = H$. ■

Theorem 14 gives us another way of showing that $A_n \lhd S_n$ since we know that $[S_n : A_n] = 2$ by Theorem 1.

. . .

The set of cosets of N in G (where $N \lhd G$) form a group, referred to as a **quotient group** or factor group (written as $\frac{G}{N}$). For quotient groups, the group operation $*$ is defined as $aN * bN = abN$. The identity is eN. Since an inverse exists for every $a \in G$, we have $aN * a^{-1}N = aa^{-1}N = eN$. If N is abelian (commutative), so is $\frac{G}{N}$.

Theorem 15. If G is a finite group and $N \lhd G$, then $|G/N| = |G|/|N|$.

Proof: This follows from Theorem 13 since the elements of $\frac{G}{N}$ are the cosets of G with respect to N. ■

One can also form quotient groups from infinite groups. For example, consider the additive group of integers $\mathbb{Z}$, and the normal subgroup $n\mathbb{Z} = \{\dots, -3n, -2n, -n, 0, n, 2n, 3n, \dots\}$. The cosets are the collection $\{n\mathbb{Z}, 1 + n\mathbb{Z}, 2 + n\mathbb{Z}, \dots, (n-2) + n\mathbb{Z}, (n-1) + n\mathbb{Z}\}$. The quotient group $\frac{\mathbb{Z}}{n\mathbb{Z}}$ can be thought of as equivalent to the group of remainders modulo n, i.e., $\mathbb{Z}_n^+$.

The idea of equivalent groups is formalized by the concepts of group homomorphism and isomorphism.

Two groups $(G,*)$ and $(H,\star)$ are **homomorphic** if there exists a function $f: G \to H$ such that for all $x, y \in G$ the following holds true

$$f(x * y) = f(x) \star f(y)$$

The purpose of defining a group homomorphism is to create functions that preserve algebraic structure between groups.

An equivalent definition of group homomorphism is as follows:

> The function $f: G \to H$ is a group homomorphism if whenever $x * y = z$ for $x, y, z \in G$, we also have $f(x) \star f(y) = f(z)$.

Theorem 16. Let $f: G \to H$ be a homomorphism between groups $(G,*)$ and $(H,\star)$. Denote the identity of G by e_G, and the identity of H by e_H. The following is true

- ***f maps the identity of G to the identity of H***
- ***f maps inverses to inverses***

Proof: For any $g \in G$, $e_G * g = g$. By the alternative definition of group homomorphism, we have

$$f(e_G) \star f(g) = f(e_G * g) = f(g) = e_H \star f(g)$$

Thus, $f(e_G) = e_H$.

For any $g \in G$, $g * g^{-1} = e_G$. By the alternative definition of group homomorphism, we have

$$f(g) \star f(g^{-1}) = f(g * g^{-1}) = f(e_G) = e_H$$

Since $f(g) \star f(g^{-1}) = e_H$ and inverses are unique, it follows that $f(g^{-1}) = f(g)^{-1}$. ■

A key point regarding homomorphisms (from G to H) is that they are only required to map into a subgroup of H. For example, the following mapping (call it f) defines a homomorphism from $\mathbb{Z}_3^+$ to a subgroup of the dihedral group D_3 (see Table 1):

$$\overline{0} \to e$$

$$\overline{1} \to a$$

$$\overline{2} \to a^2$$

To prove that f is a homomorphism we need to show $f(x+y) = f(x)g(y)$ for every $x, y \in \mathbb{Z}_3^+$. We show three of the verifications below and leave the others to the reader.

$$a^2 = f(\overline{2}) = f(\overline{1} + \overline{1}) = f(\overline{1})f(\overline{1}) = aa = a^2$$

$$a = f(\overline{1}) = f(\overline{2} + \overline{2}) = f(\overline{2})f(\overline{2}) = a^2a^2 = a$$

$$e = f(\overline{0}) = f(\overline{1} + \overline{2}) = f(\overline{1})f(\overline{2}) = aa^2 = e$$

We note that D_3 has three additional elements for which there is no mapping from $\mathbb{Z}_3^+$. This is fine for a homomorphism.

A homomorphism $f: G \to H$ that is one-to-one (i.e., injective) is called an embedding, i.e., the group G "embeds" into H as a subgroup.

If $f(G) = H$, then f is onto (i.e., surjective).

A homomorphism that is both injective and surjective is an **isomorphism**.

The names of elements in two isomorphic groups may differ. They may also look different in terms of their Cayley diagrams, but the isomorphism guarantees that they have the same algebraic structure.

When two groups G and H are isomorphic, we write $G \cong H$.

. . .

A **simple group** is a group whose only normal subgroups are the trivial group (i.e., the subgroup containing only the identity element) and the group itself. A group that is not simple can be broken into two smaller groups, namely a nontrivial normal subgroup N and the corresponding quotient group $\frac{G}{N}$. This process can be repeated, and for finite groups one can determine a cascade of normal subgroups (known as a **composition series**). Stated more formally: If G is a group, then one can construct a series of the form

$$e = N_0 \lhd N_1 \lhd N_2 \lhd \cdots \lhd N_k = G$$

with strict inclusion (i.e., N_i is a proper subset of N_{i+1}) and each N_i is a maximal normal subgroup of N_{i+1}. Further, each quotient group $\frac{N_{i+1}}{N_i}$ is a simple group.

The following theorem tells us that composition series for a finite group are equivalent.

Theorem 17 (Jordan–Hölder theorem). The composition quotient groups belonging to two composition series of a finite group G are, apart from their sequence, isomorphic in pairs.

In other words, if we have the following two composition series for finite group G

$$e \lhd N_1 \lhd N_2 \lhd \cdots \lhd N_s = G$$

$$e \lhd M_1 \lhd M_2 \lhd \cdots \lhd M_t = G$$

then $t = s$, and all the corresponding quotient groups are isomorphic, i.e.,

$$N_{i+1}/N_i \cong M_{i+1}/M_i$$

Proof: See Baumslag [20].

3.2.6 Classification of Finite Simple Groups

The search for the classification of finite simple groups started (as best as one can tell) in 1892 with the now famous quote from Otto Hölder (translated from German to English) in Mathematische Annalen [21]:

> It would be of the greatest interest if it were possible to give an overview of the entire collection of finite simple groups.

The first paper classifying an infinite family of finite simple groups, starting from a hypothesis on the structure of certain proper subgroups, was published by Burnside in 1899 [22].

The comprehensive classification of finite simple groups is attributed to Daniel Gorenstein in 1983 [23] [24]. However, the classification was not declared complete until corrections of the proof were made by Aschbacher and Smith in 2004 [25]. A detailed history of this work can be found in the Wikipedia article entitled “Classification of finite simple groups” [26].

The classification is divided among abelian and non-abelian finite simple groups, with the latter being vastly more complex.

3.2.6.1 Finite abelian groups

The following theorem describes the structure of all finite simple abelian groups.

Theorem 18. A finite abelian group is simple if and only if it has prime order p. In this case, it is isomorphic to the cyclic group $\mathbb{Z}_p$.

Proof: See “Abelian Group is Simple iff Prime” [27]. ■

The **direct product** is an operation that takes two groups $(G,*)$ and $(H,\star)$ and constructs a new group, denoted $G \times H$.

- The underlying set of the new group $G \times H$ is simply all the ordered pairs (g, h) with $g \in G$ and $h \in H$.
- The binary operation (denoted with a dot) is $(g_1, h_1) \cdot (g_2, h_2) = (g_1 * g_2, h_1 \star h_2)$.

This definition can be extended to more than two groups.

Theorem 19. Every finite abelian group is isomorphic to the direct product of cyclic groups of prime power order, i.e., a group of the form

$$\mathbb{Z}_{p_1^{a_1}} \times \mathbb{Z}_{p_2^{a_2}} \times \ldots \times \mathbb{Z}_{p_n^{a_n}}$$

Proof: See "Fundamental Theorem of Finite Abelian Groups" [28].

3.2.6.2 Finite simple non-abelian groups

The finite simple non-abelian groups are of one of the following types:

- Alternating group $A_n \geq 5$
- Groups of the finite simple Lie type: these are an assortment of matrix groups with elements from a finite field such as $\mathbb{Z}_p$ for prime number p. We discuss fields further in Section 3.4.
- Sporadic groups: these are groups that don't fit into the first two types. There are only 26 sporadic groups; the largest of which (called the Monster or just M) has $808017424794512875886459904961710757005754368000000000$ elements.

A list of all the finite simple groups (abelian and non-abelian) along with some additional information such as the group order is available in the Wikipedia article entitled "List of finite simple groups" [29].

The finite simple groups can also be organized in a sort of periodic table, see https://irandrus.wordpress.com/2012/06/17/the-periodic-table-of-finite-simple-groups/.

3.3 Rings

3.3.1 Definitions and Basic Concepts

A ring is an algebraic structure with two binary operations (usually referred to as "addition" and "multiplication" even though the operations are not necessarily what we think of as addition and multiplication of real numbers). Ring elements may be numbers such as integers, real or complex numbers, but they may also be other types of objects such as polynomials, square matrices, functions, and power series.

More precisely, a **ring** R is a non-empty set, closed under two binary operations referred to as addition (+) and multiplication ($*$), having the following properties:

- With respect to addition, R is an abelian group. The additive identity is labeled as 0.
- The multiplicative operation is associative, and there is a multiplicative identity 1. There are no requirements for inverses, or for commutativity under multiplication.
- The following distributive laws must hold:
  - $a * (b + c) = (a * b) + (a * c)$ for all $a, b, c \in R$ (left distributivity).
  - $(b + c) * a = (b * a) + (c * a)$ for all $a, b, c \in R$ (right distributivity).

If $ab = ba$, for every $a, b \in R$, then R is said to be a **commutative ring**. [Note that juxtaposition is often used as a shorthand for multiplication within rings, i.e., ab is shorthand for $a * b$.]

A **unit element of a ring** is an element that has a multiplicative inverse. So, an element u of a ring R is a unit if there exists $v \in R$ such that $uv = vu = 1$.

The following theorem establishes some basic properties of rings.

Theorem 20. Let R be a ring with additive identity 0 and multiplicative identity 1.

- ***i.*** $\mathbf{0} * \boldsymbol{a} = \mathbf{0}$ ***for every*** $\boldsymbol{a} \in \boldsymbol{R}$
- ***ii.*** ***If*** $-\boldsymbol{a}$ ***is the additive inverse of*** $\boldsymbol{a} \in \boldsymbol{R}$, ***then*** $(-\mathbf{1}) * (-\boldsymbol{a}) = \boldsymbol{a}$
- ***iii.*** $(-\mathbf{1}) * \boldsymbol{a} = -\boldsymbol{a}$ ***for every*** $\boldsymbol{a} \in \boldsymbol{R}$.

Proof:

i) Since $0 + 0 = 0$, we have from the distributive law

$$0 * a = (0 + 0) * a = (0 * a) + (0 * a)$$

Since $0 * a \in R$, it has an additive inverse, i.e., $-(0 * a)$. Adding $-(0 * a)$ to both sides of the above gives us $0 = 0 * a$.

ii) Using the distributive law and part i) of this theorem, we have

$$0 = 0 * (-a) = (-1 + 1) * (-a) = (-1) * (-a) + (-a)$$

Adding a to both sides of the above yields $a = (-1) * (-a)$.

iii) Multiple both sides of the identity $(-1) * (-a) = a$ by -1 to get

$$(-1) * (-1) * (-a) = (-1) * a$$

Applying part ii) of this theorem, we have $(-1) * (-1) = 1$, and so, the above equation reduces to $-a = (-1) * a$. ■

The integers $\mathbb{Z}$ are a commutative ring. The integer 0 is the additive identity, and the integer 1 is the multiplicative identity. Other than 1, no number has a multiplicative inverse. The multiplicative inverse of $a \in \mathbb{Z}$ is $\frac{1}{a} \in \mathbb{Q}$ but $\frac{1}{a} \notin \mathbb{Z}$ for $a \neq 1$. However, notice that if $ca = cb$ and $c \neq 0$, then $a = b$. In general, this property is known as the **multiplicative cancellation law**.

The integers modulo a prime number p under both addition and multiplication is a commutative ring (actually a field). We denote this entity as $\mathbb{F}_p$.

If we take the integers modulo n (with n not being a prime number) under addition and multiplication, we still have a commutative ring. For example, $\mathbb{Z}_6$ is a commutative ring. In this case, we have what are called **zero divisors** (i.e., two non-zero elements that multiple to 0). For example, $\overline{2} \cdot \overline{3} \equiv 0 \bmod (6)$. In this case, the multiplicative cancellation law does not hold. For example, $\overline{2} \cdot \overline{3} \equiv \overline{0} \equiv \overline{2} \cdot \overline{0}$ but if we try to cancel the $\overline{2}$ on both sides of the equation, we get $\overline{3} \equiv \overline{0}$ which is false.

A nonzero commutative ring with no zero divisors is known as an **integral domain**. There are several equivalent definitions, see "Integral domain" [42]. For example, the integers $\mathbb{Z}$ are an integral domain.

Theorem 21. The property of not having zero divisors and the multiplicative cancellation law are equivalent.

Proof: Assume the cancellation law holds for a ring R. Suppose (by way of contradiction) that there are nonzero elements $a, b \in R$ such that $a * b = 0$. By Theorem 20(i), we have $0 * b = 0$ which implies $a * b = 0 = 0 * b$. Applying the cancellation law, we have $a = 0$ (a contradiction).

Conversely, assume R does not have any zero divisors. If $c * a = c * b$ with $c \neq 0$, then $0 = c * a - c * b = c * (a - b)$. Since $c \neq 0$ and we have assumed no zero divisors, it must be that $a - b = 0$ and thus, $a = b$ and the cancellation law holds. ■

. . .

Isomorphic rings are rings that have the same structural properties, even if their elements may be different. Formally, the rings $(R, +, *)$ and $(S, \oplus, \times)$ are **isomorphic** if there is a bijective function $f: R \to S$ such that

- $f(a + b) = f(a) \oplus f(b) for\ all\ a, b \in R$
- $f(a * b) = f(a) \otimes f(b) for\ all\ a, b \in R$
- Unit (multiplicative identity) preserving, i.e., $f(1_R) = 1_S$.

On the other hand, a ring homomorphism only needs to satisfy the above three properties and be a function, but it does not need to be a bijection.

Several properties follow immediately from these assumptions and Theorem 16, e.g., $f(0_R) = 0_S$ and $f(-a) = -f(a)\ \forall a \in R$. For a comprehensive list of properties concerning ring homomorphisms, see "Ring homomorphism" [32].

The ring $\mathbb{R}$ (real numbers) is isomorphic to the ring of all 1×1 real matrices $[r]$ under matrix addition and multiplication. The isomorphism is $f(r) = [r]$.

Another example of isomorphic rings is the Cartesian product $\mathbb{R} \times \mathbb{R}$ and the set of 2×2 diagonal matrices with entries from $\mathbb{R}$, i.e., the set $\left\{\begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix} : a, b \in \mathbb{R}\right\}$. The isomorphism takes $(a, b) \in \mathbb{R} \times \mathbb{R}$ and maps it to the matric $\begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}$.

As an example of non-isomorphic rings consider the ring of polynomials with real coefficients $\mathbb{R}[X]$ and the ring of polynomials with complex coefficients $\mathbb{C}[X]$. To show that they are not isomorphic, assume there is an isomorphism $f: \mathbb{C}[X] \to \mathbb{R}[X]$. Then $f(i) = a$ for some real number a. By the multiplicative property of an isomorphism, we have $a^2 = f(i)^2 = f(i^2) = f(-1)$. From the properties of ring homomorphisms, $f(1) = 1$ and $f(-1) = -1$ as the additive inverse of 1. So, $a^2 = -1$, but no real number when squared equals -1. Thus, we have a contraction, and our initial assumption must be false, i.e., there is no isomorphism between $\mathbb{R}[X]$ and $\mathbb{C}[X]$.

3.3.2 Examples

The set of polynomials with coefficients from the real numbers, denoted $\mathbb{R}[X]$, is a commutative ring. For example, the following is an element of $\mathbb{R}[X]$

$$f(X) = a_0 + a_1 X + a_2 X^2 + \cdots + a_k X^k$$

where each coefficient a_i is an element of $\mathbb{R}$.

Two polynomials are equal if their corresponding coefficients are equal.

The additive identity is the real number 0, and the multiplicative identity is the real number 1.

Addition is pairwise, i.e., add each coefficient of like terms.

The distributive laws for polynomials are defined as in the definition of a ring. For example, given $f(X), g(X), h(X) \in \mathbb{R}[X]$

$$f(X) * \big(g(X) + h(X)\big) = \big(f(X) * g(X)\big) + \big(f(X) * h(X)\big)$$

Multiplication involves extensive use of the distributive law, e.g.,

$$(2 + 3X + X^2)(1 - 2X + 2X^2)$$

$$= 2 + (-4 + 3)X + (-6 + 4 + 1)X^2 + (-2 + 6)X^3 + 2X^4$$

$$= 2 - X - X^2 + 4X^3 + 2X^4$$

Additive inverses are formed by taking the negative of each coefficient.

Multiplicative inverses only exist for non-zero constant polynomials. For example, the multiplicative inverse of the polynomial $X^3 - 5X + 2$ is $\frac{1}{X^3 - 5X + 2}$ which is not an element of $\mathbb{R}[X]$.

Polynomials can also be defined over fields other than the real numbers, see the Wikipedia article "Polynomial ring" [31].

. . .

Take any set S and form the power set $\mathcal{P}(S)$ of S (i.e., the set of all possible subsets including the empty set and the set S itself). Over the elements of $\mathcal{P}(S)$, define addition to be the symmetric difference of sets (aka exclusive OR, or just XOR), and multiplication to be intersection.

- The symmetric difference of sets A and B is the collection of elements in either A or B, but not in both. It is written as $A \triangle B$. The symmetric difference can also be expressed as the union of the two sets, minus their intersection, i.e., $A \triangle B = (A \cup B) - (A \cap B)$.
- The intersection of sets A and B is the collection of elements in both A and B. It is written $A \cap B$.

$\mathcal{P}(S)$ with the above operations is a commutative ring.

- Clearly, intersection and symmetric difference result in another subset of S, and thus, we have additive and multiplicative closure.
- The additive identity is the empty set $\phi = \{\}$.
- The additive inverse of set A is itself.
- Symmetric difference is commutative and associative.
- The multiplicative identity is S. (Only S has a multiplicative inverse.)
- Intersection is commutative and associative.
- Intersection distributes over symmetric difference [33].

Since the cancellation law does not hold for intersection, $\mathcal{P}(S)$ under the stated binary operations is not an integral domain.

. . .

Consider the set $R = \{a + b\sqrt{D} : a, b \in \mathbb{Z}\}$ with D a fixed element of the positive integers. This set is denoted by $\mathbb{Z}[\sqrt{D}]$. In order to reduce the elements of R to their simplest form, we require that D be square free, i.e., none of its factors is a square. For example, 24 is square free but $90 = 2 \cdot 5 \cdot 3^2$ is not.

R is a commutative ring.

- $(a + b\sqrt{D}) + (a_1 + b_1\sqrt{D}) = (a + a_1) + (b + b_1)\sqrt{D}$ (additive closure)
- The additive identity is 0.
- Additive associativity and commutativity are inherited from the real numbers.
- $-a - b\sqrt{D}$ is the additive inverse of $a + b\sqrt{D}$
- $(a + b\sqrt{D})(a_1 + b_1\sqrt{D}) = (aa_1 + bb_1D) + (a_1b + ab_1)\sqrt{D} \in R$ (multiplicative closure)
- Multiplicative associativity and commutativity are inherited from the real numbers.
- The multiplicative identity is $1 = 1 + 0\sqrt{D}$.
- The distributive laws carry over from the real numbers.

The multiplicative inverse of $a + b\sqrt{D}$ is

$$\frac{1}{a + b\sqrt{D}} = \frac{1}{a + b\sqrt{D}} \cdot \frac{a - b\sqrt{D}}{a - b\sqrt{D}} = \frac{a - b\sqrt{D}}{a^2 - b^2D}$$

which is not an element $\mathbb{Z}[\sqrt{D}]$ unless $\frac{a}{a^2 - b^2D}$ and $\frac{b}{a^2 - b^2D}$ are integers.

Without further restrictions on D, there is an issue with our definition of $\mathbb{Z}[\sqrt{D}]$, i.e., we don't have unique factorization. For example, in $\mathbb{Z}[\sqrt{5}]$

$$(1 + \sqrt{5})(1 - \sqrt{5}) = 2 \cdot (-2)$$

. . .

If R is a ring, the following sets of matrices are also rings:

- The set of all $n \times n$ matrices over R, denoted $M_n(R)$. This is sometimes referred to as the "full ring of $n \times n$ matrices".
- The set of all upper triangular matrices over R, e.g., if $R = \mathbb{Z}$ then the following set is a ring

$$\left\{ \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ 0 & \cdots & a_{nn} \end{bmatrix} : a_{ij} = 0, i > j \right\}$$

- The set of all lower triangular matrices over R.
- The set of all diagonal matrices over R.

3.3.3 Ideals

An **ideal** I is a subset of a ring R with special properties. Ideals generalize certain subsets of the integers, such as the even numbers or the multiples of a positive integer n.

An ideal has two defining properties:

- Subgroup property: An ideal is closed under addition and has additive inverses, and thus, an ideal is an additive subgroup of the associated ring.
- Ideal property (sometimes referred to as the "absorbing property"): An ideal is closed under multiplication by any element of the ring.

An ideal can be used to construct a quotient ring in a way similar to how, in group theory, a normal subgroup can be used to construct a quotient group (discussed in Section 3.3.4).

More formally, a subset I of a ring R is called a left ideal of R if it satisfies the following conditions:

- I is an additive subgroup of R
- For every $x \in I$ and $r \in R$, $rx \in I$. Stated differently: $rI \subseteq I$ for every $r \in R$.

A right ideal of R is defined in a similar manner. If I is both a left and right ideal of a ring R, then we just say that I is a "two-sided ideal" or just "an ideal" of R. In a commutative ring, all ideals are two-sided.

Keep in mind that an ideal is not required to have a multiplicative identity and thus, is not necessarily a subring.

For a given ring R, the ring itself is an ideal (known as the **unit ideal**). The subset of R consisting of only the 0 element is an ideal (this follows from Theorem 20i). This is referred to as the **zero ideal**. An ideal of a ring R this is neither the unit nor zero ideal is referred to as a **proper ideal** of R.

Theorem 22. If a left ideal contains a unit (i.e., invertible) element, then it cannot be a proper ideal.

A similar theorem holds for right ideals.

Proof: Let I be a left ideal in a ring R. Assume that $u \in I$ is a unit, i.e., u^{-1} exists. Take any $r \in R$. Since R (as a ring) is closed under multiplication, we have that $ru^{-1} \in R$. Since I is a left ideal, $r = (ru^{-1})u \in I$. Thus, $I = R$ and I is not a proper ideal. ■

. . .

Principal ideals can be defined for any ring. [39]

In a commutative ring R, the principal ideal generated by a is defined as the multiples of a by every element in R, i.e., $(a) = \{ra : r \in R\}$.

In a non-commutative ring, we must distinguish between the principal left ideal, i.e., $(a)_l = \{ra : r \in R\}$, the principal right ideal, i.e., $(a)_r = \{ar : r \in R\}$, and the principal two-sided ideal, i.e., the set of all finite sums $\sum_{i=1}^{n} r_i\, a s_i$ where $r_i, s_i \in R$ and $n \geq 1$. In the commutative case, all three of these definitions are equivalent.

We don't need the summation for one-sided principal ideals. For example, consider left ideals. If $r_1 a, r_2 a \in (a)_l$, then $r_1 a + r_2 a = (r_1 + r_2)a \in (a)_l$, because addition in R distributes over multiplication from the left, i.e., $(r_1 + r_2)a = r_1 a + r_2 a$.

In the case of two-sided ideals, the summation is needed. If we naively tried to mimic the one-sided definitions, we might try $S = \{ ras \mid r, s \in R \}$ as the definition of a two-sided principal ideal.

Is S a two-sided ideal?

- Is it a left ideal? We need to check if for any $t \in R$ and $x \in S$, tx is in S.
    - $x = ras$ for some $r, s \in R$.
    - $tx = t(ras) = (tr)as$.
    - Since $tr \in R$, $(tr)as$ is indeed of the form $r'as'$ (with $r' = tr$ and $s' = s$). So yes, it's closed under left multiplication.
- Is it a right ideal? Similarly, $xt = (ras)t = ra(st)$. This is of the form $r'as'$ (with $r' = r$ and $s' = st$). So yes, it's closed under right multiplication.
- Is it closed under addition? Here is the critical problem. Consider two elements from S, i.e., r_1as_1 and r_2as_2. Their sum is $r_1as_1 + r_2as_2$.
    - There is no way to write this sum as a single term of the form ras for some $r, s \in R$, unless the ring is very special (e.g., a ring with a multiplicative identity, and we take $a = 1$ which, in this case, implies $S = R$).

Since S is not closed under addition, it fails the very definition of an ideal. The set S is just the set of single-term products.

. . .

The even integers form a principal ideal (denoted $2\mathbb{Z}$) in the ring of all integers $\mathbb{Z}$. The generator is 2. Further, we have that

- $2\mathbb{Z}$ is an additive subgroup of $\mathbb{Z}$
- For every $x \in 2\mathbb{Z}$ and $r \in \mathbb{Z}$, we have an even number times an integer (even or odd) which results in another even number, and so, $rx \in I = 2\mathbb{Z}$. Thus, I is a left ideal of $R = \mathbb{Z}$. Similarly, I is a right ideal of R.

The set $n\mathbb{Z} = \{\dots, -3n, -2n, -n, 0, n, 2n, 3n, \dots\}$ is an additive subgroup of $\mathbb{Z}$. Take any $kn \in n\mathbb{Z}$ and any $r \in \mathbb{Z}$. We have that $(kn)r = (kr)n \in n\mathbb{Z}$ and $r(kn) = (kr)n \in n\mathbb{Z}$, and similar when mutlplying by r on the left. Thus, $n\mathbb{Z}$ is a principal ideal of $\mathbb{Z}$ with generator n.

. . .

The set of all polynomials with real coefficients, divisible by $X^2 + 1$ (with no remainder), is an ideal (call it I) in the ring of all real-coefficient polynomials $\mathbb{R}[X]$. If $f(X), g(X) \in I$, the sum $f(X) + g(X)$ is also divisible by $X^2 + 1$, and so, $f(X) + g(X) \in I$ (closure). The polynomial 0 is divisible by $X^2 + 1$ and so, $0 \in I$. If $f(X) \in I$, then so is $-f(X) \in I$ (inverses). Associativity and commutativity in I are inherited from $\mathbb{R}[X]$. So, I is an abelian subgroup within $\mathbb{R}[X]$.

For any $f(X) \in I$ and $r(X) \in \mathbb{R}[X]$, $f(X)r(X)$ is divisible by $X^2 + 1$ which implies $f(X)r(X) \in I$ and so, I is a right ideal. Similarly, we can show that I is a left ideal.

The following are some additional examples of ideals. We leave the verification to the reader.

- In the ring of continuous function f from $\mathbb{R}$ to $\mathbb{R}$ under pointwise multiplication [35], the subset of function f such that $f(1) = 0$ is an ideal.
- Another ideal in the ring of continuous function f from $\mathbb{R}$ to $\mathbb{R}$ is the set of function that vanish for sufficiently large values of x, i.e., continuous functions f for which there exists a value L such that $f(x) = 0$ for all $|x| > L$.
- Consider the ring $M_n(R)$ of all $n \times n$ matrices over any ring R. The subset $I_{ro\ \ i}$ comprised of all matrices in $M_n(R)$ whose i^{th} row consists of zeros is a right ideal of $M_n(R)$. Note that for any $A \in I_{ro\ \ i}$ and $M \in M_n(R)$, the product AM has all zeros in row i and thus, $AM \in I_{ro\ \ i}$. However, I_{row_i} is not a left ideal. On the other hand, the subset I_{col_j} comprised of all matrices in $M_n(R)$ whose j^{th} column consists of zeros is a left ideal of $M_n(R)$ but not a right ideal.

For even more examples of ideals, see "Ideal (ring theory)" [36].

3.3.4 Quotient Rings

A **quotient ring**, also known as factor ring, is a structure similar to a quotient group. Given a ring R and a two-sided ideal $I \subseteq R$, we can construct a quotient ring, denoted R/I, whose elements are the cosets of I in R, where the cosets of I in R are defined in a similar manner to the cosets of a subgroup within a group. Given any $r \in R$, $r + I = \{r + x : x \in I\}$ is a coset of I in R. The set of all cosets of I in R comprise the quotient ring of I with respect to the ring R.

A quotient ring is a ring that is constructed from another ring, called the parent ring, by dividing out by an ideal. Formally, the elements of R/I are the equivalence classes of R under the equivalence relation $\sim$ defined by $x \sim y$ if and only if $x + (-y) \in I$. The addition and multiplication operations on R/I are defined by

- $(x + I) + (y + I) = (x + y) + I$
- $(x + I)(y + I) = (xy) + I$

It can be shown that these operations are well-defined, and that R/I is a ring under these operations.

The key to understanding a particular quotient ring is to identify the cosets, i.e., the equivalence classes.

. . .

Given a ring R, the quotient ring $R/\{0\}$ is equivalent to R, since each coset $r + \{0\} = \{r + x : x \in \{0\}\}$ corresponds to an element of R. On the other hand, the quotient ring R/R has but one coset, i.e., $r + R = \{r + x : x \in R\}$ which generates all of R, regardless of the r we select. This one coset is basically the 0 element of R/R. In general, the larger the ideal I, the smaller the quotient ring R/I. If I is a proper ideal of R, i.e., $I \neq R$, then R/I is not the zero ring.

. . .

Consider the ring of integers $\mathbb{Z}$ and the ideal of even numbers, i.e., $2\mathbb{Z}$. The quotient ring $\mathbb{Z}/2\mathbb{Z}$ has only two elements, the coset $0 + 2\mathbb{Z}$ consisting of the even numbers and the coset $1 + 2\mathbb{Z}$ consisting of the odd numbers. $\mathbb{Z}/2\mathbb{Z}$ is isomorphic $\mathbb{Z}_2$ with addition and multiplication modulo 2. In fact, some textbooks will write $\mathbb{Z}/2\mathbb{Z}$ rather than $\mathbb{Z}_2$ (even before they introduce quotient rings).

Consider the polynomial ring $\mathbb{R}[X]$ and the quotient ring $\mathbb{R}[X]/I$, where I is the principal ideal generated by (X^2+1), i.e., $I = \{a(X)\cdot(X^2+1): a(X) \in \mathbb{R}[X]\}$. A coset of $\mathbb{R}[X]/I$ is of the form

$$g(X) + I = \{g(X) + a(X)\cdot(X^2+1)\}$$

The zero element of R/I is $(X^2+1)+I$. So, the coset $(X^2+1)+I$ consists of all multiples of (X^2+1). Further, we have that $X^2+1=0$ with respect to the quotient ring, which implies $X^2 = -1$ or $X = \sqrt{-1} = i$ (with some abuse of notation).

Using the **division algorithm** for polynomials (also known as "polynomial long division") [37], any $g(X) \in \mathbb{R}[X]$ can be written in the form

$$g(X) = q(X)\cdot(X^2+1) + r(x)$$

where the remainder $r(X)$ is linear, i.e., of the form $aX+b$ for constants $a, b \in \mathbb{R}$, and $q(X) \in \mathbb{R}(X)$.

So, an arbitrary element of $\mathbb{R}[X]/I$ is of the form $aX+b$. The operations of the quotient ring are exactly the same as those for the complex numbers $\mathbb{C}$. More formally, $\mathbb{R}[X]/(X^2+1)$ is isomorphic to $\mathbb{C}$ under the mapping $f(aX+b) = ai+b$. To prove this using the definition of isomorphism, take two elements in $\mathbb{R}[X]/I$, i.e., $aX+b$ and $cX+d$.

For multiplication, we have $f\big((aX+b)(cX+d)\big) = f(acX^2 + (ad+bc)X + bd)$

and since $X^2 = -1$, the above reduces to

$$f\big((ad+bc)X - (bd-ac)\big) = (ad+bc)i + (bd-ac)$$

Further, we have that $f(aX+b)f(cX+d) = (ai+b)(ci+d) = (ad+bc)i + (bd-ac)$

and so, $f\big((aX+b)(cX+d)\big) = f(aX+b)f(cX+d)$.

For addition, we have

$$f\big((aX+b)+(cX+d)\big) = f\big((a+c)X+(b+d)\big) = (a+c)i + (b+d)$$

$$f(aX+b) + f(cX+d) = (ai+b)+(ci+d) = (a+c)i + (b+d)$$

Thus, $\mathbb{R}[X]/(X^2+1)$ and $\mathbb{C}$ are isomorphic rings.

. . .

In a generalization of the previous example, let R be any commutative ring and let $R[X]$ be the polynomial ring defined over R. Take an n^{th} degree polynomial in $R[X]$, i.e.,

$$f(X) = a_n X^n + \cdots + a_1 X + a_0$$

and form the principal ideal generated by $f(X)$, i.e.,

$$I = \{a(X)f(X): a(X) \in R[X]\}$$

A coset of the quotient ring $R[X]/I$ is of the form

$$g(X) + I = \{g(X) + a(X)f(X) : a(X) \in R[X]\}$$

However, by the division algorithm for polynomials, we have that any $g(X) \in R[X]$ can be written in the form

$$g(X) = q(X)f(X) + r(X)$$

where $\deg\big(r(X)\big) < \deg\big(f(X)\big)$, i.e., $r(X)$ is of the form $b_{n-1}X^{n-1} + \cdots + b_1X + b_0$.

So, the cosets of the quotient ring $R[X]/I$ are polynomials of the form

$$\begin{aligned} g(X) + I &= \{g(X) + a(X)f(X) : a(X) \in R[X]\} \\ &= \{q(X)f(X) + r(X) + a(X)f(X) : a(X) \in R[X]\} \\ &= \big\{r(X) + \big(q(X) + a(X)\big)f(X) : a(X) \in R[X]\big\} \\ &= r(X) + I \end{aligned}$$

with $r(X)$ of the form noted above, i.e., polynomial over R of degree $n - 1$ or less.

3.4 Fields

3.4.1 Definitions and Basic Concepts

A **field** is a commutative ring (under both addition and multiplication) such that $0 \neq 1$ and all nonzero elements are invertible under multiplication. The Wikipedia article on fields offers the following (equivalent) definitions of a field [38]:

> Formally, a field is a set F together with two binary operations on F called addition and multiplication. A binary operation on F is a mapping $F \times F \to F$, that is, a correspondence that associates with each ordered pair of elements of F a uniquely determined element of F.
>
> The result of the addition of a and b is called the sum of a and b, and is denoted $a + b$. Similarly, the result of the multiplication of a and b is called the product of a and b, and is denoted $a \cdot b$. These operations are required to satisfy the following properties, referred to as field axioms.
>
> These axioms are required to hold for all elements a, b, c of the field F:
>
> - Associativity of addition and multiplication: $a + (b + c) = (a + b) + c$, and $a \cdot (b \cdot c) = (a \cdot b) \cdot c$.
> - Commutativity of addition and multiplication: $a + b = b + a$, and $a \cdot b = b \cdot a$.
> - Additive and multiplicative identity: there exist two distinct elements 0 and 1 in F such that $a + 0 = a$ and $a \cdot 1 = a$.
> - Additive inverses: for every a in F, there exists an element in F, denoted $-a$, called the additive inverse of a, such that $a + (-a) = 0$.

- Multiplicative inverses: for every $a \neq 0$ in F, there exists an element in F, denoted by a^{-1} or $\frac{1}{a}$, called the multiplicative inverse of a, such that $a \cdot a^{-1} = 1$.
- Distributivity of multiplication over addition: $a \cdot (b + c) = (a \cdot b) + (a \cdot c)$.

An equivalent, and more succinct, definition is as follows: a field has two commutative operations, called addition and multiplication; it is a group under addition with 0 as the additive identity; the nonzero elements form a group under multiplication with 1 as the multiplicative identity; and multiplication distributes over addition.

Even more succinctly: a field is a commutative ring where $0 \neq 1$ and all nonzero elements are invertible under multiplication.

In terms of notation, we also write ab as a shorthand for $a \cdot b$.

If multiplicative commutativity is dropped (while keeping the other axioms), the structure is instead called a **division ring** (or skew field). An example is the set of quaternions $\mathbb{H}$, where multiplication is non-commutative. On the other hand, finite fields like $\mathbb{F}_p$ (for prime p) and the real numbers $\mathbb{R}$ are commutative fields.

[A **quaternion** is an expression of the form

$$a + b\mathbf{i} + c\mathbf{j} + d\mathbf{k}$$

where a, b, c, d, are real numbers, and $\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k}$, are symbols that can be interpreted as unit-vectors pointing along the three spatial axes.]

Wedderburn's Little Theorem states that all finite division rings are commutative (i.e., finite fields). So, there are no finite noncommutative division rings

The following theorem relates rings and ideals to fields.

Theorem 23. A nonzero commutative ring R is a field with multiplicative identity $\mathbf{1}$ if and only if its only ideals are $\{\mathbf{0}\}$ and the ring itself.

Proof: Assume that R is a field with multiplicative identity 1. Take any ideal I in R. If $I = \{0\}$, we are done. If $I \neq \{0\}$, it contains some nonzero element, and every nonzero element in a field is a unit. It follows by Theorem 22 that $I = R$.

Going the other way, assume that the only ideals of the ring R are $\{0\}$ and the ring itself. Since we are given that R is nonzero, it contains an element $a \neq 0$. The principal ideal (a) must equal R, since we assumed the only ideals of R are $\{0\}$ and R, and $a \neq 0$. Since $R = (a)$, there exists an $r \in R$ such that $1 = ra$ which implies that a (which we picked arbitrarily) has an inverse. Thus, R is a field. ■

. . .

The rational numbers $\mathbb{Q}$, real numbers $\mathbb{R}$ and complex numbers $\mathbb{C}$ are examples of fields with an infinite number of elements.

For any prime number p, the integers modulo p form a finite field of order p, denoted $\mathbb{F}_p$ or $GF(p)$ where GF stands for “Galois field.” These are the simplest finite fields. In $\mathbb{F}_p$:

- Addition and multiplication are done modulo p.
- The additive identity is 0.

- The multiplicative identity is 1.
- The additive inverse of a is $p - a$.
- The multiplicative inverse of a non-zero element a can be found using the Extended Euclidean Algorithm since $\gcd(a, p) = 1$.

Example: $\mathbb{F}_5$ = {0, 1, 2, 3, 4}

- $2 + 4 = 6 \equiv 1 \pmod 5$
- $2 \times 3 = 6 \equiv 1 \pmod 5$, and so, the multiplicative inverse of 2 is 3.
- $4 \times 4 = 16 \equiv 1 \pmod 5$, and so, the multiplicative inverse of 4 is 4.
- Of course, the multiplicative inverse of 1 is 1 for all rings. So, we have established multiplicative inverses for each non-zero element in $\mathbb{F}_5$.

. . .

A **field homomorphism** is a function $\varphi: F \to K$ between two fields F and K such that for all a, b in F:

- $\varphi(a + b) = \varphi(a) + \varphi(b)$ (Preserves addition)
- $\varphi(a \times b) = \varphi(a) \times \varphi(b)$ (Preserves multiplication)
- $\varphi(1_F) = 1_K$ (Preserves the multiplicative identity)

A **field isomorphism** is a bijective homomorphism. If an isomorphism exists between two fields F and K, they are said to be **isomorphic**, denoted $F \cong K$. This means they are structurally identical as fields.

Crucial Point: The definition of a field isomorphism is fundamentally about the preservation of both field operations. It is a necessary consequence that such a map is also an isomorphism of the underlying additive and multiplicative groups. However, this group-level isomorphism is a byproduct, not the defining property. One cannot define a field isomorphism solely by the condition that the additive and multiplicative groups are isomorphic.

Finite fields (also called **Galois fields**) are fields with a finite number of elements.

Theorem 24. The order of a finite field (i.e., its number of elements) is either a prime number or a prime power. For every prime number p and every positive integer k there are fields of order p^k, all of which are isomorphic.

Proof: See the Existence and Uniqueness section of the Wikipedia article on Finite Fields [40]. ■

3.4.2 Construction of Non-prime Finite Fields

As noted, finite fields are either of prime order p (isomorphic to $\mathbb{F}_p$) or of prime power order p^n. In this section, we describe how to construct fields of prime power order.

Given a prime power $q = p^n$ with p prime and $n > 1$, the field $\mathbb{F}_q$ (also written as $GF(q)$) may be constructed as follows:

1. Choose an irreducible polynomial $P(X)$ in $\mathbb{F}_p[X]$ of degree n (such an irreducible polynomial always exists). By "irreducible polynomial", we mean a polynomial that cannot be factored into the product of two non-constant polynomials in $\mathbb{F}_p[X]$.

2. Construction the quotient ring $\mathbb{F}_p(X)/P(X)$. The quotient ring is the field of order q that we seek, i.e., $\mathbb{F}_q \cong \mathbb{F}_p[X]/P(X)$.
    a. $\big(P(X)\big)$ is an ideal that consists of all multiples of $P(X)$, i.e., $\big(P(X)\big) = \{f(X) \cdot P(X) : f(X) \in \mathbb{F}_p[X]\}$.
    b. $P(X)$ is the additive zero of $\mathbb{F}_p[X]/P(X)$, since the remainder when dividing $P(X)$ by itself is 0.
    c. All multiples of $P(X)$ are in the same equivalence class as $P(X)$.
    d. Elements of $\mathbb{F}_p[X]/P(X)$ are of the form $f(X) \cdot P(X) + r(X)$ where $r(X), f(X) \in \mathbb{F}_p[X]$ and $\deg\big(r(X)\big) < \deg\big(P(X)\big) = n$. Basically, one takes an element of $\mathbb{F}_p[X]$ divides by $P(X)$ and gets the remainder $r(X)$.

The elements of $\mathbb{F}_q$ are the polynomials over $\mathbb{F}_p[X]$ whose degree is strictly less than n.

- Addition and subtraction in $\mathbb{F}_q$ are defined by addition and subtraction of polynomials in $\mathbb{F}_p[X]$.
- Multiplication of two elements in $\mathbb{F}_q$ entails the product of the two elements and then taking the remainder upon long division by $P(X)$.
- The multiplicative inverse of a non-zero element may be computed with the extended Euclidean algorithm.

. . .

As an example, we construct $\mathbb{F}_4$. With respect to the above procedure, we have that $q = 2^2$ and so, $n = 2$. In $\mathbb{F}_2[X]$, there is only one irreducible polynomial of degree 2, i.e., $P(X) = X^2 + X + 1$. In $\mathbb{F}_2$, we have only two elements, i.e., $\bar{0}$ and $\bar{1}$, and so, $P(\bar{0}) = \bar{1}$ and $P(\bar{1}) = \bar{1}$ (noting that $3 \equiv 1 \pmod 2$). Thus, $P(X)$ is irreducible.

[**Note**: To prove that $P(X) = X^2 + X + 1$ is the only irreducible polynomial in $\mathbb{F}_2[X]$, one needs to consider all the polynomials of the form $aX^2 + bX + c$ where $a, b, c \in \mathbb{F}_2$ (a total of 8 cases) and then test to see which polynomial has no roots in $\mathbb{F}_2$.]

Next, we determine the elements in $\mathbb{F}_4 = \mathbb{F}_2[X]/(X^2 + X + 1)$, each of which is a remainder of a polynomial in $\mathbb{F}_2[X]$ when divided by $X^2 + X + 1$. This means the elements of $\mathbb{F}_2[X]/(X^2 + X + 1)$ are of the form $aX + b$, where $a, b \in \mathbb{F}_2$. Since $a, b \in \{\bar{0}, \bar{1}\}$, there are only four possible outcomes for $aX + b$.

[**Author's Remark**: Letting $X = \alpha$ appears to be a convention in some textbooks and online articles. This is not necessary, but I'll go along with the de facto convention.]

Working with the cosets of $\mathbb{F}_2[X]/(X^2 + X + 1)$, we have

- $\bar{0} + (X^2 + X + 1) \in \mathbb{F}_2(X)/(X^2 + X + 1)$ which is the additive identity of $\mathbb{F}_4$. We will call this element 0.
- $\bar{1} + (X^2 + X + 1) \in \mathbb{F}_2[X]/(X^2 + X + 1)$ which is the multiplicative identity of $\mathbb{F}_4$. We will call this element 1.
- $X + (X^2 + X + 1) \in \mathbb{F}_2[X]/(X^2 + X + 1)$ which we will call $\alpha \in \mathbb{F}_4$.
- $(X + 1) + (X^2 + X + 1) \in \mathbb{F}_2[X]/(X^2 + X + 1)$ which we will call $\alpha + 1 \in \mathbb{F}_4$.

The addition table for $\mathbb{F}_4$ is shown in Table 2. Keep in mind that we are working with polynomials of the form $a + b\alpha$ where $a, b \in \mathbb{F}_2$ (integers modulo 2). So, for example, $(1 + \alpha) + (1 + \alpha) = 2 + 2\alpha = 0$ since $2 \equiv 0 \pmod 2$.

Table 2. Addition table for the finite field of order 4

$+$	$\mathbf{0}$	$\mathbf{1}$	$\boldsymbol{\alpha}$	$\mathbf{1} + \boldsymbol{\alpha}$
$\mathbf{0}$	0	1	α	$1 + \alpha$
$\mathbf{1}$	1	0	$1 + \alpha$	α
$\boldsymbol{\alpha}$	α	$1 + \alpha$	0	1
$\mathbf{1} + \boldsymbol{\alpha}$	$1 + \alpha$	α	1	0

We need one other fact to compute the multiplicative group tables for $\mathbb{F}_4$. Since $X^2 + X + 1$ is the 0 element, we have $X^2 = -X - 1 = X + 1$ since $1 = -1$ in $\mathbb{F}_2$. So, within the quotient ring $\mathbb{F}_2[X]/(X^2 + X + 1)$, $\alpha^2 = X^2 = X + 1 = \alpha + 1$.

The multiplication table for $\mathbb{F}_4$ is shown in Table 3. For example, $(1 + \alpha)(1 + \alpha) = 1 + 2\alpha + \alpha^2 = 1 + 0 + \alpha^2 = 1 + (1 + \alpha) = 2 + \alpha = \alpha$.

Table 3. Multiplication table for the finite field of order 4

$\times$	$\mathbf{0}$	$\mathbf{1}$	$\boldsymbol{\alpha}$	$\boldsymbol{\alpha + 1}$
$\mathbf{0}$	0	0	0	0
$\mathbf{1}$	0	1	α	$\alpha + 1$
$\boldsymbol{\alpha}$	0	α	$\alpha + 1$	1
$\boldsymbol{\alpha + 1}$	0	$\alpha + 1$	1	α

. . .

As a second example, we construct $\mathbb{F}_8$. In this case, we have that $q = 2^3$ and so, $n = 3$. In $\mathbb{F}_2[X]$, there are several irreducible polynomials of degree 3, e.g., $P(X) = X^3 - X - 1$ and $Q(X) = X^3 + X^2 + 1$. We will use $P(x)$ in the following calculations.

Now, let us determine the elements in $\mathbb{F}_8 = \mathbb{F}_2[X]/(X^3 - X - 1)$ each of which is a remainder of a polynomial in $\mathbb{F}_2(X)$ when divided by $X^3 - X - 1$. Thus, elements of $\mathbb{F}_2[X]/(X^3 - X - 1)$ are of the form $aX^2 + bX + c$, where $a, b, c \in \mathbb{F}_2$. Since $a, b, c \in \{\overline{0}, \overline{1}\}$, there are eight possible outcomes for $aX^2 + bX + c$. Letting $\alpha = X$, we have the following cosets of $\mathbb{F}_2[X]/(X^3 - X - 1)$:

$$0, 1, \alpha, \alpha + 1, \alpha^2, \alpha^2 + 1, \alpha^2 + \alpha, \alpha^2 + \alpha + 1$$

The addition table for $\mathbb{F}_8$ is shown in Table 4. Note that the addition table for $\mathbb{F}_4$ is embedded in the first 4 rows and columns (shown in gray).

Table 4. Addition table for the finite field of order 8

$+$	$\mathbf{0}$	$\mathbf{1}$	$\boldsymbol{\alpha}$	$\boldsymbol{\alpha+1}$	$\boldsymbol{\alpha^2}$	$\boldsymbol{\alpha^2+1}$	$\boldsymbol{\alpha^2+\alpha}$	$\boldsymbol{\alpha^2+\alpha+1}$
$\mathbf{0}$	0	1	α	$\alpha+1$	α^2	α^2+1	$\alpha^2+\alpha$	$\alpha^2+\alpha+1$
$\mathbf{1}$	1	0	$\alpha+1$	α	α^2+1	α^2	$\alpha^2+\alpha+1$	$\alpha^2+\alpha$
$\boldsymbol{\alpha}$	α	$\alpha+1$	0	1	$\alpha^2+\alpha$	$\alpha^2+\alpha+1$	α^2	α^2+1
$\boldsymbol{\alpha+1}$	$\alpha+1$	α	1	0	$\alpha^2+\alpha+1$	$\alpha^2+\alpha$	α^2+1	α^2
$\boldsymbol{\alpha^2}$	α^2	α^2+1	$\alpha^2+\alpha$	$\alpha^2+\alpha+1$	0	1	α	$\alpha+1$
$\boldsymbol{\alpha^2+1}$	α^2+1	α^2	$\alpha^2+\alpha+1$	$\alpha^2+\alpha$	1	0	$\alpha+1$	α
$\boldsymbol{\alpha^2+\alpha}$	$\alpha^2+\alpha$	$\alpha^2+\alpha+1$	α^2	α^2+1	α	$\alpha+1$	0	1
$\boldsymbol{\alpha^2+\alpha+1}$	$\alpha^2+\alpha+1$	$\alpha^2+\alpha$	α^2+1	α^2	$\alpha+1$	α	1	0

Since $X^3 - X - 1$ is the 0 element, we have $X^3 = X + 1$. So, within the quotient ring $\mathbb{F}_2[X]/(X^3 - X - 1)$, $\alpha^3 = X^3 = X + 1 = \alpha + 1$. With this information, we can compute the powers of α.

- $\alpha^3 = \alpha + 1$
- $\alpha^4 = \alpha^3\alpha = (\alpha + 1)\alpha = \alpha^2 + \alpha$
- $\alpha^5 = \alpha^4\alpha = (\alpha^2 + \alpha)\alpha = \alpha^3 + \alpha^2 = \alpha^2 + \alpha + 1$
- $\alpha^6 = \alpha^3\alpha^3 = (\alpha + 1)^2 = \alpha^2 + 2\alpha + 1 = \alpha^2 + 1$, since $2 \equiv 0 \pmod 2$
- $\alpha^7 = \alpha^3\alpha^4 = (\alpha + 1)(\alpha^2 + \alpha) = \alpha^3 + 2\alpha^2 + \alpha = \alpha^3 + \alpha = (\alpha + 1) + \alpha = 1$

So, the powers of α generate $\mathbb{F}_8$, i.e., $\mathbb{F}_8 = \langle\alpha\rangle$ is a cyclic group with respect to multiplication.

The multiplication table for $\mathbb{F}_8$ is shown in Table 5.

Table 5. Multiplication table for the finite field of order 8

$\times$	$\mathbf{0}$	$\mathbf{1}$	$\boldsymbol{\alpha}$	$\boldsymbol{\alpha+1}$	$\boldsymbol{\alpha^2}$	$\boldsymbol{\alpha^2+1}$	$\boldsymbol{\alpha^2+\alpha}$	$\boldsymbol{\alpha^2+\alpha+1}$
$\mathbf{0}$	0	0	0	0	0	0	0	0
$\mathbf{1}$	0	1	α	$\alpha+1$	α^2	α^2+1	$\alpha^2+\alpha$	$\alpha^2+\alpha+1$
$\boldsymbol{\alpha}$	0	α	α^2	$\alpha^2+\alpha$	$\alpha+1$	1	$\alpha^2+\alpha+1$	α^2+1
$\boldsymbol{\alpha+1}$	0	$\alpha+1$	$\alpha^2+\alpha$	α^2+1	$\alpha^2+\alpha+1$	α^2	1	α
$\boldsymbol{\alpha^2}$	0	α^2	$\alpha+1$	$\alpha^2+\alpha+1$	$\alpha^2+\alpha$	α	α^2+1	1
$\boldsymbol{\alpha^2+1}$	0	α^2+1	1	α^2	α	$\alpha^2+\alpha+1$	$\alpha+1$	$\alpha^2+\alpha$
$\boldsymbol{\alpha^2+\alpha}$	0	$\alpha^2+\alpha$	$\alpha^2+\alpha+1$	1	α^2+1	$\alpha+1$	α	α^2
$\boldsymbol{\alpha^2+\alpha+1}$	0	$\alpha^2+\alpha+1$	α^2+1	α	1	$\alpha^2+\alpha$	α^2	$\alpha+1$

. . .

The previous two examples relied on the fact that a second or third degree polynomial over $\mathbb{F}_2$ is either irreducible or has a linear factor with associated zero.

For higher degree polynomials (degree 4 and greater), checking for zeros is not sufficient. For example, the polynomial $x^4 + x^2 + 1$ over $\mathbb{F}_2$ has no zeros but it is not irreducible. This polynomial

factors as $(x^2+x+1)^2$, where x^2+x+1 is irreducible over $\mathbb{F}_2$. To verify the factorization, note that

$$(x^2+x+1)^2 = x^4+2x^3+3x^2+2x+1$$
$$\equiv x^4+0\cdot x^3+1\cdot x^2+0\cdot x+1 = x^4+x^2+1(\mathrm{mod}2).$$
$$\ldots$$

In general, we have the following result.

Theorem 25. The multiplicative group of all nonzero elements of a finite field is cyclic.

Proof: See Theorem 22.10 and Corollary 22.11 in Section 22.1 of Judson [6]. ■

3.5 Vector Spaces

3.5.1 Definition

A **vector space** over a field F is a non-empty set V together with two binary operations defined below. The elements of V are called vectors, and the elements of F are called scalars. Vectors are usually represented either in bold (e.g., $\boldsymbol{v}$) or with an arrow above, e.g., $\vec{v}$.

- One of the binary operations is vector addition. This operation assigns to any two vectors $\boldsymbol{u}, \boldsymbol{v} \in V$ a third vector in V which is written as $\boldsymbol{u}+\boldsymbol{v}$, and called the sum of these two vectors.
- The other operation is scalar multiplication. This operation assigns to any scalar $a \in F$ and any vector $\boldsymbol{v} \in V$ another vector in V, denoted $a\boldsymbol{v}$.

In addition to the two closure axioms above, a vector space must satisfy the following axioms.

- The set V must be an abelian group under the vector addition operation. The group identity is represented by the symbol $\mathbf{0}$. [**Caution**: the field F also has an additive identity which is represented by 0 (not bold).]
- Compatibility of scalar multiplication with field multiplication, i.e., $a(b\boldsymbol{v}) = (ab)\boldsymbol{v}$ for $a, b \in F$ and $\boldsymbol{v} \in V$.
- Identity element of scalar multiplication, i.e., there exists $1 \in F$ such that $1\boldsymbol{v} = \boldsymbol{v}$ for every $\boldsymbol{v} \in V$ where 1 is the multiplicative identity in the field F.
- Distributivity of scalar multiplication with respect to vector addition, i.e., $a(\boldsymbol{u}+\boldsymbol{v}) = a\boldsymbol{u} + a\boldsymbol{v}$ for $a, b \in F$ and $\boldsymbol{u}, \boldsymbol{v} \in V$.
- Distributivity of scalar multiplication with respect to field addition, i.e., $(a+b)\boldsymbol{v} = a\boldsymbol{v} + b\boldsymbol{v}$ for $a, b \in F$ and $\boldsymbol{v} \in V$.

Some basic results for vector spaces are presented in the following theorem.

Theorem 26. The following relationships hold true:

i. $0\boldsymbol{v} = \mathbf{0}$
ii. $a\mathbf{0} = \mathbf{0}$
iii. $(-1)\boldsymbol{v} = -\boldsymbol{v}$

iv. $a\boldsymbol{v} = \mathbf{0}$ implies either $a = 0$ or $\boldsymbol{v} = \mathbf{0}$.

Proof:

i. Using the distributive rules for vector spaces, we have

$$0\boldsymbol{v} = (0 + 0)\boldsymbol{v} = 0\boldsymbol{v} + 0\boldsymbol{v}$$

Adding the additive inverse of $0\boldsymbol{v}$ to both sides of the above equation, we get

$$\mathbf{0} = 0\boldsymbol{v} - 0\boldsymbol{v} = (0\boldsymbol{v} + 0\boldsymbol{v}) - 0\boldsymbol{v} = 0\boldsymbol{v}$$

ii. Using the distributive rules for vector spaces, we have

$$a\mathbf{0} = a(\mathbf{0} + \mathbf{0}) = a\mathbf{0} + a\mathbf{0}$$

Adding the additive inverse of $a\mathbf{0}$ to both sides of the above equation, we get

$$\mathbf{0} = a\mathbf{0} - a\mathbf{0} = (a\mathbf{0} + a\mathbf{0}) - a\mathbf{0} = a\mathbf{0}$$

iii. For any $\boldsymbol{v} \in V$, we have

$$\boldsymbol{v} + (-1)\boldsymbol{v} = 1\boldsymbol{v} + (-1)\boldsymbol{v} = \big(1 + (-1)\big)\boldsymbol{v} = 0\boldsymbol{v} = \mathbf{0}$$

and thus, $(-1)\boldsymbol{v} = -\boldsymbol{v}$.

iv. If $a = 0$, we are done. If $a \neq 0$ then a (being an non-zero element of a field) has an inverse, i.e., a^{-1} exists. Multiplying both sides of $a\boldsymbol{v} = \mathbf{0}$ by a^{-1} and using part ii of this theorem, we have that $a^{-1}a\boldsymbol{v} = 1\boldsymbol{v} = \boldsymbol{v} = a^{-1}\mathbf{0} = \mathbf{0}$. ■

3.5.2 Examples

The following are examples of vector spaces:

- The set of all real numbers $\mathbb{R}$ over the real numbers $\mathbb{R}$ is a vector space under the usual addition and multiplication of real numbers. In this case, $V = F = \mathbb{R}$.
- The set of all ordered pairs of real numbers is a vector space over the field of real numbers. Addition is pairwise, i.e., if $(x, y), (s, t) \in V$ then $(x, y) + (s, t) = (x + s, y + t)$. Scalar multiplication is as follows: for $(x, y) \in V$ and $a \in \mathbb{R}$, $a(x, y) = (ax, ay)$.
- The set of ordered n-tuples of real numbers is a vector space over the field of real numbers. Addition and scalar multiplication are defined in a similar manner to the previous case.
- The set of all polynomials with real coefficients is a vector space over the field of real numbers. Vector addition is accomplished by adding coefficients of like terms (i.e., terms of the same power) and scalar multiplication entails the multiplication of the given scalar times each coefficient in the given polynomial.
- The set of all matrices with real coefficients over the real numbers is a vector space. Standard matrix addition is used here, and multiplication of a matrix by a scalar results in every element of the given matrix being multiplied by the scalar.
- The set of all continuous functions from the real numbers to the real numbers is a vector space under the usual addition of functions and the pointwise multiplication of functions by real numbers. The associated field is $\mathbb{R}$.
- The set of all solutions to a system of linear equations with real coefficients is a vector space over the field of real numbers.

3.5.3 Concepts

In what follows (unless stated otherwise), we assume V is a vector space over field F.

A **linear combination of vectors** results in a new vector that is created by adding together scalar multiples of a set of vectors. For example, take vectors $\boldsymbol{u}, \boldsymbol{v} \in V$ and scalars $a, b \in F$ where V is a vector space over the field F. The vector $a\boldsymbol{u} + b\boldsymbol{v} \in V$ is a linear combination of vectors $\boldsymbol{u}$ and $\boldsymbol{v}$. Linear combinations can be created for any number of vectors.

A set of vectors is **linearly independent** if no vector in the set can be written as a linear combination of the other vectors in the set. Equivalent definitions:

- A set of vectors is linearly independent if a linear combination results in the zero vector if and only if all its coefficients (i.e., all the scalars in the linear combination) are zero.
- A set of vectors is linearly independent if two linear combinations of the set define the same element of V if and only if they have the same coefficients.

For example, in 3-dimensional Euclidean space $\mathbb{R}^3$, the vectors $(1,0,0), (0,1,0), (0,0,1)$ are linearly independent since there is no way of writing any one of the vectors as a linear combination of the other two.

A set of vectors that is not linearly independent is said to be **linearly dependent**. For example, in 3-dimensional Euclidean space $\mathbb{R}^3$, the vectors $(1,1,0), (0,1,1), (2,-1,-3)$ are linearly dependent since we can write $(2,-1,-3) = 2 \cdot (1,1,0) - 3 \cdot (0,1,1)$.

A **linear subspace of a vector space** is a subset of the vector space that is also a vector space. In other words, a linear subspace is a set of vectors that is closed under vector addition and scalar multiplication. For example, 2-dimensional Euclidean space $\mathbb{R}^2$ is a linear subspace of $\mathbb{R}^3$.

Theorem 27. A subset U of a vector space V over a field F is a subspace of V if and only if $a\boldsymbol{x} + b\boldsymbol{y} \in U$ for all $\boldsymbol{x}, \boldsymbol{y} \in U$ and all $a, b \in F$.

Proof: If U is a subspace (and thus a vector space in its own right), then closure under vector addition and scalar multiplication implies $a\boldsymbol{x} + b\boldsymbol{y} \in U$ for all $\boldsymbol{x}, \boldsymbol{y} \in U$ and all $a, b \in F$.

If $a\boldsymbol{x} + b\boldsymbol{y} \in$ U for all $\boldsymbol{x}, \boldsymbol{y} \in U$ and all $a, b \in F$, then

- $\boldsymbol{x} + \boldsymbol{y} = 1\boldsymbol{x} + 1\boldsymbol{y} \in U$ for every $\boldsymbol{x}, \boldsymbol{y} \in U$ (closure under vector addition)
- $\boldsymbol{0} = 0\boldsymbol{x} + 0\boldsymbol{y} \in U$
- For $\boldsymbol{x} \in U$, we have that $-\boldsymbol{x} = (-1)\boldsymbol{x} + 0\boldsymbol{y} \in U$
- The additive associative and commutative laws are inherited by U from V.
- The above points prove that U is an abelian subgroup of V.
- $a\boldsymbol{x} = a\boldsymbol{x} + 0\boldsymbol{y} \in U$ for every $\boldsymbol{x} \in U$ and $a \in F$ (closure under scalar multiplication)
- The other properties for a vector space (i.e., compatibility of scalar multiplication and the two distributive rules) are inherited by U from V.

Thus, U is a subspace of V. ■

The **linear span** of a set of vectors is the smallest linear subspace that contains the set of vectors. In other words, the linear span of a set of vectors is the set of all vectors that can be created as linear combinations of the vectors in the set.

More formally, the **linear span** (or simply **span**) of vector $v_1, v_2, \dots, v_n \in V$ is defined as

$$span(\boldsymbol{v_1}, \boldsymbol{v_2}, \dots, \boldsymbol{v_n}) = \{\, a_1\boldsymbol{v_1} + a_2\boldsymbol{v_2} + \cdots + a_n\boldsymbol{v_n} \mid a_1, a_2 \dots, a_n \in F \,\}$$

A **basis of a vector space** is a set of vectors that spans the vector space and is linearly independent. In other words, a basis for a vector space is a set of vectors that can be used to create any vector in the space, and no two vectors in the set are linearly dependent.

For example, the vectors $(1,0,0), (0,1,0), (0,0,1)$ form a basis for $\mathbb{R}^3$. There are an infinite number of bases for $\mathbb{R}^3$.

In general, there are many bases for any vector space. However, it can be shown that the number of vectors in a basis for a given vector space is the same for all bases.

The **dimension of a vector space** is the number of vectors in a basis for the space. It is also possible to have infinite dimensional vector spaces. For example, the set of all polynomials with real coefficients, the set of all continuous functions on the real line, and the set of all sequences of real numbers are infinite dimensional vector spaces.

A vector space V is said to be finite dimensional if there a finite set $S = \{\boldsymbol{v_1}, \boldsymbol{v_2}, \dots, \boldsymbol{v_n}\}$ such that $V = span\{\boldsymbol{v_1}, \boldsymbol{v_2}, \dots, \boldsymbol{v_n}\}$.

The above concepts are tightly related as evidenced by the following theorems.

Theorem 28. Let V be a vector space with $v_1, v_2, \dots, v_n \in V$, then the following holds true

i. $\boldsymbol{v_j} \in span(\boldsymbol{v_1}, \boldsymbol{v_2}, \dots, \boldsymbol{v_n})$ for $j = 1,2, \dots, n$

ii. $span(\boldsymbol{v_1}, \boldsymbol{v_2}, \dots, \boldsymbol{v_n})$ is a subspace of V

iii. If $U \subset V$ is a subspace such that $\boldsymbol{v_1}, \boldsymbol{v_2}, \dots, \boldsymbol{v_n} \in U$, then $span(\boldsymbol{v_1}, \boldsymbol{v_2}, \dots, \boldsymbol{v_n}) \subset U$.

Proof:

i. $\boldsymbol{v_j} = 0\boldsymbol{v_1} + 0\boldsymbol{v_2} + \cdots + 1\boldsymbol{v_j} + \cdots + 0\boldsymbol{v_n}$

ii. By the definition of linear span, $span(\boldsymbol{v_1}, \boldsymbol{v_2}, \dots, \boldsymbol{v_n})$ is closed under vector addition and scalar multiplication and thus, is a subspace of V (by Theorem 27).

iii. Since U is a subspace of V, it is closed under vector addition and scalar multiplication which implies $span(\boldsymbol{v_1}, \boldsymbol{v_2}, \dots, \boldsymbol{v_n}) \subset U$. ■

Theorem 29. If $V = span(v_1, v_2, \dots, v_n)$, then either $S = \{v_1, v_2, \dots, v_n\}$ is a basis for V or some subset of S is a basis for V.

Proof: If some $\boldsymbol{v} \in S$ can be written as a linear combination of the other vectors in S, then $S - \{\boldsymbol{v}\}$ spans V. If some $\boldsymbol{u} \in S - \{v\}$ can be written as a linear combination of the other vectors in $S - \{\boldsymbol{v}\}$,

then $S - \{\boldsymbol{v}, \boldsymbol{u}\}$ spans V. We continue in this manner until the remaining vectors are linearly independent while still spanning V (i.e., until we have found a basis). ■

Theorem 30. Every finite-dimensional vector space V has a basis.

Proof: Since V is finite-dimensional, it has a finite spanning set $S = \{\boldsymbol{v_1}, \boldsymbol{v_2}, \dots, \boldsymbol{v_n}\}$ such that $V = span(\boldsymbol{v_1}, \boldsymbol{v_2}, \dots, \boldsymbol{v_n})$. It then follows from Theorem 29 that V has a basis. ■

Theorem 31. If V is a finite-dimensional vector space, then any two bases of V have the same number of elements.

Proof: See Theorem 5.4.2 in the book "Linear Algebra" [41]. ■

. . .

Take vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ in vector space V over a field F with basis $\boldsymbol{v_1}, \boldsymbol{v_2}, \dots, \boldsymbol{v_n}$. We can uniquely represent $\boldsymbol{x}$ and $\boldsymbol{y}$ in terms of the basis vectors as follows:

$$\boldsymbol{x} = x_1\boldsymbol{v_1} + x_2\boldsymbol{v_2} + \cdots + x_n\boldsymbol{v_n}, \quad x_1, x_2, \dots, x_n \in F$$

$$\boldsymbol{y} = y_1\boldsymbol{v_1} + y_2\boldsymbol{v_2} + \cdots + y_n\boldsymbol{v_n}, \quad y_1, y_2, \dots, y_n \in F$$

The scalars (i.e., the x_i and y_i terms in the above equations) are called the coordinates of $\boldsymbol{x}$ and $\boldsymbol{y}$ over the given basis. They are also said to be the coefficients of the decomposition of $\boldsymbol{x}$ and $\boldsymbol{y}$ over the given of the basis. We can also write $\boldsymbol{x}$ and $\boldsymbol{y}$ in terms of their coordinates as follows:

$$\boldsymbol{x} = (x_1, x_2, \dots, x_n)$$

$$\boldsymbol{y} = (y_1, y_2, \dots, y_n)$$

A **coordinate vector** is a representation of a vector as an ordered list of numbers (a tuple) that describes the vector in terms of a particular ordered basis.

The set of n-tuples with elements from F is a vector space (denoted F^n) with addition and scalar multiplication defined component-wise, i.e.,

$$\boldsymbol{x} + \boldsymbol{y} = (x_1 + y_1, x_2 + y_2, \dots, x_n + y_n)$$

$$a\boldsymbol{x} = (ax_1, ax_2, \dots, ax_n), \qquad a \in F$$

. . .

Take vectors $\boldsymbol{x} = (x_1, x_2, \dots, x_n)$ and $\boldsymbol{y} = (y_1, y_2, \dots, y_n)$, the **inner product** of $\boldsymbol{x}$ and $\boldsymbol{y}$ is defined as

$$\boldsymbol{x} \cdot \boldsymbol{y} = x_1y_1 + x_2y_2 + \cdots + x_ny_n$$

For example, take vectors $\boldsymbol{x} = (1,2,0,1,1)$ and $\boldsymbol{y} = (1,1,1,0,0)$ where the elements of $\boldsymbol{x}$ and $\boldsymbol{y}$ come from $\mathbb{F}_3$, then

$$\boldsymbol{x} \cdot \boldsymbol{y} = 1 + 2 + 0 + 0 + 0 = 0$$

If $\boldsymbol{x} \cdot \boldsymbol{y} = 0$, then $\boldsymbol{x}$ and $\boldsymbol{y}$ are said to be **orthogonal vectors**.

When a vector space is defined over a finite field, it is possible for a vector to be orthogonal to itself. For example, if $\boldsymbol{x} = (1,1,1,1)$ over $\mathbb{F}_2$, then

$$\boldsymbol{x} \cdot \boldsymbol{x} = 1 + 1 + 1 + 1 \equiv 0 (\text{mod } 2)$$

It is also possible for a vector to be orthogonal to itself when defined over an infinite field. For example, consider the vector space $\mathbb{C}^2$ over the field of complex numbers $\mathbb{C}$. The vector $\boldsymbol{v} = (1, i)$ is orthogonal to itself since

$$v \cdot v = 1^2 + i^2 = 1 + (-1) = 0$$

However, a non-zero vector in the vector space $\mathbb{R}^n$ over the field of real numbers cannot be orthogonal to itself.

Theorem 32. For any coordinate vectors $\boldsymbol{x}, \boldsymbol{y}$ and $\boldsymbol{z}$ (with respect to a given basis) in an n-dimensional vector space $\boldsymbol{V}$ over a field $\boldsymbol{F}$, and scalars $\boldsymbol{a}, \boldsymbol{b} \in \boldsymbol{F}$, the following holds true

i. $\boldsymbol{x} \cdot \boldsymbol{y} = \boldsymbol{y} \cdot \boldsymbol{x}$

ii. $(a\boldsymbol{x} + b\boldsymbol{y}) \cdot \boldsymbol{z} = a(\boldsymbol{x} \cdot \boldsymbol{z}) + b(\boldsymbol{y} \cdot \boldsymbol{z})$

Proof: The proof of part i follows directly from the definition of inner product.

For part ii, let $\boldsymbol{x} = (x_1, x_2, \dots, x_n), \boldsymbol{y} = (y_1, y_2, \dots, y_n)$ and $\boldsymbol{z} = (z_1, z_2, \dots, z_n)$. We then have

$$(a\boldsymbol{x} + b\boldsymbol{y}) \cdot \boldsymbol{z} = (ax_1 + by_1, \dots, ax_n + by_n) \cdot (z_1, z_2, \dots, z_n)$$

$$= (ax_1 + by_1)z_1 + \cdots + (ax_n + by_n)z_n$$

$$= (ax_1z_1 + by_1z_1) + \cdots + (ax_nz_n + by_nz_n)$$

$$= (ax_1z_1 + \cdots + ax_nz_n) + (by_1z_1 + \cdots + by_nz_n)$$

$$= a(x_1z_1 + \cdots + x_nz_n) + b(y_1z_1 + \cdots + y_nz_n)$$

$$= a(\boldsymbol{x} \cdot \boldsymbol{z}) + b(\boldsymbol{y} \cdot \boldsymbol{z})$$

. . .

A basis for an n-dimensional vector space V over a field F is a linearly independent subset that spans V. For the specific case $V = F^n$ (the space of n-tuples over F), the standard basis is the ordered set

$$B = \{(1,0,0, \dots,0), (0,1,0, \dots,0), \dots, (0,0,0, \dots,1)\}$$

where the tuples denote coordinates with respect to B itself, 1 is the multiplicative identity in F, and 0 is the additive identity in F. Clearly, the vectors in B are linearly independent, span F^n and, when F^n is equipped with a standard inner product (e.g., the dot product over $\mathbb{R}$ or $\mathbb{C}$), form an orthogonal set of unit vectors.

3.6 Algebras

In simple terms, an **algebra** is a vector space where you can also multiply vectors together in a "nice" way. An algebra provides a multiplication rule that defines how to get another vector when multiplying two vectors.

3.6.1 Comparison to Fields and Vector Spaces

In a Field (e.g., Real Numbers):

- You have a set of objects (like all numbers 1, 2, 3.14, -5, etc.).

- You can add them: $a + b$
- You can multiply them: $a * b$
- Everything works as you'd expect: multiplication is distributive over addition, i.e., $a * (b + c) = a * b + a * c$, and every non-zero number has a multiplicative inverse.

In a Vector Space:

- You have a set of objects called vectors (like arrows pointing in space, or a list of numbers such as $(1, 2, 3)$).
- You can add vectors together: $\boldsymbol{u} + \boldsymbol{v}$
- You can scale a vector by a number (a scalar) from a field: $a\boldsymbol{v}$ (e.g., stretching a vector by 2).
- But you can't multiply two vectors together. You can only scale or add them.

In an Algebra (The Bridge Between Them)**:**

- You start with a vector space (so you can add and scale vectors).
- But then, you add a multiplication rule that lets you combine any two vectors to get a third vector: $\boldsymbol{u} * \boldsymbol{v} = \boldsymbol{w}$
- This new multiplication rule is "nice" in the sense that it plays well with the existing addition and scalar multiplication. This "niceness" is expressed as the following distributive properties**:**
    - $\boldsymbol{u} * (\boldsymbol{v} + \boldsymbol{w}) = (\boldsymbol{u} * \boldsymbol{v}) + (\boldsymbol{u} * \boldsymbol{w})$
    - $(\boldsymbol{u} + \boldsymbol{v}) * \boldsymbol{w} = (\boldsymbol{u} * \boldsymbol{w}) + (\boldsymbol{v} * \boldsymbol{w})$
    - For any scalar k, we have $k(\boldsymbol{u} * \boldsymbol{v}) = (k\boldsymbol{u}) * \boldsymbol{v} = \boldsymbol{u} * (k\boldsymbol{v})$.

3.6.2 Simple Real-World Example: The Complex Numbers

The Complex Numbers $\mathbb{C}$ are a perfect example of an algebra.

1. They are a Vector Space:
    - Any complex number $a + bi$ can be thought of as a vector in a 2D plane (the complex plane).
    - You can add them: $(a + bi) + (c + di) = (a + c) + (b + d)i$
    - You can scale them by a real number.
2. They have a Multiplication Rule:
    - You can multiply them: $(a + bi) * (c + di) = (ac - bd) + (ad + bc)i$
    - The result is another complex number (another vector in the same 2D plane).
    - This multiplication is distributive and works nicely with scalar multiplication.

So, the complex numbers are a 2-dimensional algebra over the real numbers.

3.6.3 Some Other Examples

- **Matrices**: The set of all $n \times n$ matrices forms an algebra. You can add them, scale them, and crucially, multiply them together. The result of multiplying two matrices is another matrix of the same size.
- **Polynomials**: The set of all polynomials with real coefficients is an algebra. You can add, scale, and multiply polynomials, and the result is another polynomial.

3.6.4 Summary

Table 6. Comparison of Ring, Field, Vector Space and Algebra

Structure	**What you can do**
Group	Add (or combine) elements.
Ring/Field	Ring: Add, subtract, and multiply elements. A field is a ring that also lets you divide.
Vector Space	Add vectors and scale them by numbers (scalars).
Algebra	Add vectors, scale them by numbers, **and** multiply vectors together in a consistent way.

In the simplest terms: An algebra is a vector space with a “compatible multiplication” for its vectors.

4 Extending Number Systems

In mathematics, you don't understand things. You just get used to them. — John von Neumann

In Section 5, we will extend the real numbers to the set of hyperreal numbers. In this section, we discuss two approaches for extending a number system, i.e., (1) using Dedekind cuts to extend the rational numbers to the reals, and (2) using an unsolvable quadratic equation to extend the reals to the complex numbers. Although neither approach applies directly to our development of the hyperreals, these examples illuminate the diverse strategies (topological, algebraic, and beyond) that mathematicians have employed to transcend the limitations of existing fields.

4.1 Dedekind Cuts

A **Dedekind cut** is a way to define real numbers by partitioning the set of rational numbers into two non-empty subsets, A and B, such that:

1. $A \cup B = \mathbb{Q}$ (the union of A and B is the set of all rational numbers).
2. $A \cap B = \emptyset$ (the intersection of A and B is empty).
3. Every element of A is less than every element of B, i.e., if $a \in A$ and $b \in B$, then $a < b$.
4. A has no greatest element (there is no largest rational number in A).

This construction, introduced by Richard Dedekind, is used to formally define the real numbers by associating each real number with a unique cut. The idea is that a real number, which may or may not be rational, corresponds to the boundary between A and B. If the cut corresponds to a rational number, B has a smallest element; if it corresponds to an irrational number, B has no smallest element.

Dedekind cuts address the gaps (i.e., irrational numbers) in the rational numbers. For example, there is no rational number whose square is exactly 2, but we can define the real number $\sqrt{2}$ using a Dedekind cut.

Think of a Dedekind cut as splitting the number line of rational numbers into two parts: one containing all rational numbers less than a certain value (real or rational), and the other containing all rational numbers greater than or equal to that value (if the value is rational) or just greater (if the value is irrational).

Dedekind cuts provide a rigorous way to construct the real numbers from the rational numbers, ensuring the real numbers are complete (i.e., every bounded set has a least upper bound).

Example 1: Dedekind Cut for $\sqrt{2}$

Let's define the real number $\sqrt{2}$ (an irrational number) using a Dedekind cut.

Define $A = \{q \in \mathbb{Q}: q^2 < 2 \text{ or } q < 0\}$.

- This includes all negative rational numbers, and all positive rational numbers q such that $q^2 < 2$.
- Examples of numbers in A: $-1, 0, 1, \frac{4}{3}, \frac{99}{70}$ since, for example, $\left(\frac{99}{70}\right)^2 = \frac{9801}{4900} \approx 1.9996 < 2$.

Define $B = \{q \in \mathbb{Q} \mid q^2 \geq 2 \text{ and } q > 0\}$.

- This includes all positive rational numbers q such that $q^2 \geq 2$.
- Examples of numbers in B: $\frac{3}{2}, \frac{99}{70}$ since, for example, $\left(\frac{3}{2}\right)^2 = \frac{9}{4} = 2.25 \geq 2$.

Verification:

1. Union: Every rational number q is either in A if $q < 0$ or $q^2 < 2$, or in B if $q^2 \geq 2$ and $q > 0$, so $A \cup B = \mathbb{Q}$.
2. Disjoint: No rational number can satisfy both $q^2 < 2$ and $q^2 \geq 2$ (for $q > 0$), and negative numbers are only in A, so $A \cap B = \emptyset$.
3. Ordering: If $a \in A$ and $b \in B$, then either $a < 0 \leq b$, or $a > 0$ and $a^2 < 2 \leq b^2$, implying $a < b$ (since the square root function is increasing for positive numbers).
4. No greatest element in A:
   - For any $a \in A$, if $a \geq 0$, then $a^2 < 2$. We can find another rational $a_1 > a$ such that $a_1^2 < 2$. For example, take $a_1 = a + \frac{2-a^2}{k}$ for a sufficiently large rational k, ensuring $a_1^2 < 2$. (Note: The choice $a_1 = a + \frac{2-a^2}{k}$ ensures $a_1 > a$ and $a_1^2 < 2$ for large enough k because the increment is small enough relative to the remaining gap $2 - a^2$, and scaling by $\frac{1}{k}$ allows us to control the quadratic increase in a_1^2.)
   - If $a < 0$, we can take $a_1 = 0$ or a small positive rational. Thus, A has no greatest element.
5. For any $b \in B$, $b > 0$ and $b^2 \geq 2$. We can find another rational b_1 with $0 < b_1 < b$ and $b_1^2 \geq 2$. For example, take $b_1 = b - \frac{b^2-2}{k}$ for a sufficiently large rational k. This ensures $b_1 > 0$ and $b_1^2 \geq 2$, because the decrement is small enough to keep b_1^2 above 2. Thus B has no least element.

This cut defines $\sqrt{2}$, as the boundary between A and B corresponds to the real number whose square is 2. This shows $\sqrt{2}$ is irrational.

Example 2: Dedekind Cut for a Rational Number (e.g., 2)

For a rational number such as 2, the cut is slightly different:

Define $A = \{q \in \mathbb{Q}: q < 2\}$.

Examples: $1, \frac{3}{2}, \frac{199}{100}$

Define $B = \{q \in \mathbb{Q}: q \geq 2\}$.

Examples: $2, \frac{5}{2}, 3$

Verification:

- $A \cup B = \mathbb{Q}$ and $A \cap B = \emptyset$
- If $a \in A$ and $b \in B$, then $a < 2 \leq b$, and so, $a < b$.
- A has no greatest element: for any $a < 2$, there exists $a_1 = a + \frac{2-a}{2} < 2$ with $a_1 > a$.

- B has a smallest element, i.e., 2, indicating the cut corresponds to a rational number.

Key Difference for the Two Cases:

- For irrational numbers (such as $\sqrt{2}$), B has no smallest element.
- For rational numbers (such as 2), B has a smallest element (the rational number itself).

The above construction ensures every real number can be represented by a unique Dedekind cut, completing the number line by filling in the gaps left by the rational numbers.

. . .

The **dyadic numbers** (rational numbers of the form $\frac{m}{2^n}$ where m is an integer and n is a non-negative integer) are dense in the real number line, meaning between any two real numbers, no matter how close, lies a dyadic number. However, the dyadics themselves are not complete; they are riddled with gaps corresponding to rational numbers such as $\frac{1}{3}$ and irrational numbers such as $\sqrt{2}$. We can fill these gaps using Dedekind cuts. As we saw with the irrational numbers, the fundamental idea is to define a real number not as a single value, but as a *separation* of the dyadic numbers into two sets: a left set containing all dyadics less than the intended real number, and a right set containing all those greater. For example, the irrational number $\sqrt{2}$ is uniquely defined by the cut where the left set contains every dyadic number whose square is less than 2, and the right set contains every dyadic whose square is greater than 2. Since every conceivable cut of the dyadics corresponds to a real number, and every real number creates such a cut, this construction seamlessly extends the discrete, "jumpy" world of the dyadics into the continuous, gapless continuum of the real numbers.

More generally, Dedekind cuts can be used to construct the real numbers starting from *any* set that is dense in the reals, such as the Cantor set[3] or the rationals. The crucial property is density, which ensures that for any real number x, the set of elements in your dense set that are less than x is non-empty and distinct from the set of elements greater than x. We then define a real number as a partition of this dense set into a left set L and a right set R, where every element of L is less than every element of R, and L has no greatest element. For instance, using the Cantor set (which, despite being large in terms of cardinality but small in terms of length, is still dense between 0 and 1), a number like $\frac{1}{3}$ is defined by the cut where L contains all points of the Cantor set less than $\frac{1}{3}$, and R contains all points greater than $\frac{1}{3}$. This cut would be different from one defining, say, $\frac{1}{2}$. The resulting set of all such Dedekind cuts, defined on the dense subset, is order-isomorphic to the complete continuum of real numbers, effectively filling all the gaps and achieving completeness.

4.2 Complex Numbers

The extension from the real numbers $\mathbb{R}$ to the complex numbers $\mathbb{C}$ is fundamentally different from the construction of $\mathbb{R}$ from the rational numbers $\mathbb{Q}$ via Dedekind cuts. Dedekind cuts address the issue of completeness in an ordered field [5], ensuring every non-empty subset of real numbers (bounded above) has a least upper bound, which fills the gaps in $\mathbb{Q}$. In contrast, the motivation for

[3] The Cantor set consists of all real numbers of the unit interval $[0, 1]$ that do not require the digit 1 in order to be expressed as a ternary (base 3) fraction. The Cantor set has measure 0 and yet they are dense in the real numbers.

complex numbers is algebraic closure, i.e., ensuring every non-constant polynomial has a root, particularly to solve equations like $x^2 + 1 = 0$ that have no real solutions.

There isn't a direct analog to Dedekind cuts for this extension, as $\mathbb{C}$ is not an ordered field (you can't define a total order on complex numbers that respects addition and multiplication in a way compatible with the reals). Instead, the standard rigorous construction of $\mathbb{C}$ from $\mathbb{R}$ is algebraic and treats complex numbers as ordered pairs of reals with custom operations. This makes $\mathbb{C}$ a field extension of $\mathbb{R}$ of degree 2.

Construction of Complex Numbers from Reals:

We define $\mathbb{C}$ as the set of all ordered pairs (a, b) where $a, b \in \mathbb{R}$, equipped with the following operations:

- Addition: $(a, b) + (c, d) = (a + c, b + d)$.
- Multiplication: $(a, b) \cdot (c, d) = (ac - bd, ad + bc)$.

This structure forms a field (it satisfies all field axioms: closure, associativity, commutativity, identities, inverses, and distributivity). The real numbers embed into $\mathbb{C}$ via $a \to (a, 0)$, and the imaginary unit $i \equiv \sqrt{-1}$ is identified with $(0,1)$, since:

$$(0, 1) \cdot (0, 1) = (0 \cdot 0 - 1 \cdot 1, 0 \cdot 1 + 1 \cdot 0) = (-1, 0)$$

which corresponds to -1 in the reals.

Verification as a Field Extension

- Additive identity: $(0, 0)$
- Multiplicative identity: $(1, 0)$
- Additive inverse: The inverse of (a, b) is $(-a, -b)$.
- Multiplicative inverse: the complex number $(a, b) \neq (0,0)$ has inverse $\left(\frac{a}{a^2+b^2}, \frac{-b}{a^2+b^2}\right)$ since multiplying it by (a, b) yields $(1, 0)$.
- Every non-zero element has an inverse, and the operations make it a division ring (actually a field, as it's commutative).

This construction ensures $\mathbb{C}$ is algebraically closed (by the Fundamental Theorem of Algebra) and contains $\mathbb{R}$ as a subfield. In terms of algebraic closure, consider (for example) the equation $x^2 + 1 = 0$. We now have a solution $i \in \mathbb{C}$ since $i^2 + 1 = -1 + 1 = 0$.

Alternative Perspective, using the concept of quotient rings (as discussed in Section 3.3.4 and summarized below):

Equivalently, $\mathbb{C}$ can be constructed as the quotient ring $\mathbb{R}[x]/\langle x^2 + 1\rangle$, where $\mathbb{R}[x]$ is the ring of polynomials with real coefficients, and $\langle x^2 + 1\rangle$ is the ideal generated by the irreducible polynomial $x^2 + 1$. Elements are cosets like $a + bx + \langle x^2 + 1\rangle$, with $x^2 = -1$, mirroring the pair construction.

Why No “Cut-Like” Analog?

Dedekind cuts rely on the order structure of $\mathbb{Q}$, but $\mathbb{C}$ doesn't admit a compatible total order (e.g., you can't consistently decide if $i > 0$ or $i < 0$ without breaking field properties). Extensions beyond

$\mathbb{C}$, like quaternions or octonions, use similar pair-based constructions (e.g., Cayley-Dickson process), but they're not commutative or associative in higher dimensions and serve different purposes (e.g., rotations in 3D/4D space).

4.3 Quaternions

4.3.1 Definition of Quaternions

Quaternions are a number system that extends the complex numbers. They form what is known as a 4-dimensional associative normed division algebra over the real numbers.

In simpler terms, a quaternion is an expression of the form:

$$\boldsymbol{q = a + bi + cj + dk}$$

where:

- a, b, c, d are real numbers.
- a is the scalar part.
- $bi + cj + dk$ is the vector part.
- i, j, k are the fundamental quaternion units. They are analogous to the imaginary unit i in complex numbers, but there are three of them, and they obey the following defining relations:

The Hamilton Product Rules:

$$i^2 = j^2 = k^2 = ijk = -1$$

From these fundamental rules, the multiplication rules for the units can be derived:

- $ij = k$ but $ji = -k$ (so multiplication is not commutative)
- $jk = i$ but $kj = -i$
- $ki = j$ but $ik = -j$

The non-commutativity ($ij \neq ji$) is the single most important and revolutionary property of quaternions.

4.3.2 Background and History

The story of the quaternion's discovery is one of the most famous in mathematics.

- Inventor: Sir William Rowan Hamilton (1805-1865), an Irish mathematician.
- The “Eureka Moment”: For years, Hamilton was trying to find a way to extend the complex numbers (which represent rotations and dilations in 2D) into three dimensions. He was looking for a 3D number system with a well-defined multiplication. He struggled because such a system is impossible.
- The Breakthrough: On October 16, 1843, while walking with his wife along the Royal Canal in Dublin, the solution came to him. He couldn't create a 3D system, but he could create a 4D one. So struck by this revelation, he famously carved the fundamental formula $i^2 = j^2 = k^2 = ijk = -1$ into the stone of the Broom Bridge.

This discovery was monumental because it was the first example of a consistent number system where the commutative property of multiplication did not hold. It opened the door to the development of modern abstract algebra, including vectors and linear algebra.

4.3.3 Derivation and View as an Extension of Complex Numbers

The journey from real numbers to quaternions is a story of successive extensions, each one sacrificing a property to gain a new capability.

Step 1: Real Numbers $\mathbb{R}$ to Complex Numbers $\mathbb{C}$

- Real Numbers ($a \in \mathbb{R}$): Represent points on a 1D line. You can scale and translate.
- Problem: The equation $x^2 = -1$ has no solution in the reals.
- Extension: Introduce a new imaginary unit i where $i^2 = -1$.
- Complex Number ($a + bi \in \mathbb{C}$): Represents a point in a 2D plane. The a is the real part, and the bi is the imaginary part.
- New Power: Complex numbers provide an elegant way to represent and perform rotations and dilations in the 2D plane.
- Properties Kept: Addition and multiplication are still commutative and associative.

Step 2: Complex Numbers $\mathbb{C}$ to Quaternions $\mathbb{H}$

- Complex Numbers ($a + bi \in \mathbb{C}$): Work perfectly for 2D.
- Problem: Hamilton wanted a similar system to handle rotations in 3D space. He tried to add a second imaginary unit j to create numbers of the form $a + bi + cj$.
- The Obstacle: He couldn't define a consistent multiplication for this 3D system. The product of two such numbers would often escape the system, requiring a fourth dimension.
- The Extension: Hamilton realized he needed *three* imaginary units, i, j, and k, leading to the 4D number $a + bi + cj + dk$.
- The Sacrifice: To make this 4D system consistent and useful (specifically, to preserve the "norm" or length), he had to abandon the commutative property of multiplication. This was a radical but necessary step.
- New Power: Quaternions provide an incredibly efficient and robust way to represent and compose rotations in 3D space. They avoid the problem of "gimbal lock" that plagues other methods like Euler angles. (Gimbal lock is when a spinning object gets stuck because two of its rotation axes accidentally line up, making it lose the ability to turn in one direction.)

How to View a Quaternion as a "Hyper-Complex" Number:

You can think of a quaternion as being built from *pairs* of complex numbers.

Consider a quaternion $q = a + bi + cj + dk$. We can group this as:

$$q = (a + bi) + (c + di)j$$

Notice that $a + bi$ and $c + di$ are both standard complex numbers. So, we can write:
$q = z + wj$ where $z = a + bi$ and $w = c + di$ are complex numbers.

The multiplication rule $ji = -k$ is now built into the system. If you multiply two quaternions written in this form, $z_1 + w^1 j$ and $z^2 + w^2 j$, and enforce the rules $j^2 = -1$ and that j multiplies with the complex numbers z and w by "scrambling" them (e.g., $j * z = \bar{z} * j$), you will get the correct quaternion product.

Let's multiply two quaternions, $q_1 = z_1 + w_1 j$ and $q_2 = z_2 + w_2 j$, using the rules above:

$$\begin{aligned}(z_1 + w_1 j)(z_2 + w_2 j) &= z_1 z_2 + z_1 w_2 j + w_1 j z_2 + w_1 j w_2 j \quad \text{(Distributive property)}\\ &= z_1 z_2 + z_1 w_2 j + w_1(\bar{z}_2 j) + w_1(\bar{w}_2 j) j \quad \text{(Apply } jz = \bar{z}j\text{)}\\ &= z_1 z_2 + z_1 w_2 j + w_1 \bar{z}_2 j + w_1 \bar{w}_2 j^2 \quad \text{(Group terms)}\\ &= z_1 z_2 + z_1 w_2 j + w_1 \bar{z}_2 j + w_1 \bar{w}_2(-1) \quad \text{(Apply } j^2 = -1\text{)}\\ &= (z_1 z_2 - w_1 \bar{w}_2) + (z_1 w_2 + w_1 \bar{z}_2) j \quad \text{(Group the } j \text{ terms)}\end{aligned}$$

This view solidifies the idea that quaternions are a natural, if non-commutative, extension of the complex numbers into a higher dimension.

4.3.4 Summary and Modern Usage

Table 7 shows a comparison of the various properties of complex numbers and quaternions.

Table 7. Comparison of Complex Numbers and Quaternions

Feature	Complex Numbers	Quaternions
Dimension	2D	4D
Form	$a + bi$	$a + bi + cj + dk$
Key Property	$i^2 = -1$	$i^2 = j^2 = k^2 = ijk = -1$
Multiplication	commutative	non-commutative
Primary Use	2D rotations and analysis	3D rotations

Today, quaternions are fundamental in:

- Computer Graphics: The standard way to represent and interpolate 3D rotations.
- Robotics: For controlling the orientation of robotic arms and vehicles.
- Flight Dynamics: Calculating the attitude of aircraft and spacecraft.
- Video Games: Storing and smoothly interpolating the rotation of 3D objects.

They are the perfect tool for the job because they are computationally more efficient and numerically more stable than their main alternatives (rotation matrices or Euler angles).

5 Hyperreal Numbers: Foundations of Non-Standard Analysis

We choose to go to the moon, not because it is easy, but because it is hard.

John F. Kennedy

In this section, we provide a streamlined summary of the construction of the hyperreals via filters and something known as the ultrapower method. The discussion is interspersed with concrete examples at key points to illustrate the concepts, with the intent of making the discussion more accessible while keeping the overall structure concise. For a more detailed construction of the hyperreal numbers, the reader is referred to the book by Goldblatt [44].

5.1 Key Definitions

In what follows, let $I = \mathbb{N}$ (the natural numbers, i.e., 1,2,3,…). Other discrete sets are possible but we will use the natural numbers here.

A **filter** $\mathcal{F}$ on I is a collection of subsets from I satisfying the following properties:

- The empty set $\emptyset$ is **not** in $\mathcal{F}$.
- If $A \in \mathcal{F}$ and $A \subseteq B \subseteq I$, then $B \in \mathcal{F}$ (closed under supersets).
- If $A, B \in \mathcal{F}$, then $A \cap B \in \mathcal{F}$ (closed under finite intersections).

A filter is a subset of the powerset of I, i.e., $\mathcal{F} \subseteq \mathcal{P}(I)$.

A set $A \subseteq I$ is said to be **cofinite** if its complement $I - A$ has a finite number of members.

Example 1: The **Fréchet filter** $\mathcal{F}_F$ consists of all cofinite subsets of I.

- For instance, the set of all natural numbers greater than 10 is in the Fréchet filter since its complement $\{1,2, \dots ,10\}$ is finite.
- However, $\{1, 3, 5,7, \dots\}$ is not in the Fréchet filter since its complement is infinite, i.e., $\{2,4,6,8, \dots\}$. Similarly, $\{2,4,6,8, \dots\}$ is not in the Fréchet filter.

The Fréchet filter captures the idea of "almost all" natural numbers.

An **ultrafilter** $\mathcal{U}$ on I is a filter such that for every $A \subseteq I$, either $A \in \mathcal{U}$ or $I - A \in \mathcal{U}$ (but not both). If both A and $I - A$ were in $\mathcal{U}$, then their intersection, $A \cap (I - A) = \emptyset$, would also be in $\mathcal{U}$ (by the intersection property), but a filter cannot contain the empty set. Thus, the "but not both" condition is needed.

Equivalently, an ultrafilter is a maximal filter, meaning it is not properly contained in any other filter.

Example 2: A **principal ultrafilter** is an ultrafilter that is generated by a single element, say $\{n\}$: it includes all sets containing n. For example, $\{5,6,7, \dots\}$ is in the principal ultrafilter generated by $\{5\}$ since it contains 5 and is a superset of $\{5\}$. A principal ultrafilter can also contain finite sets, e.g., the principal ultrafilter generated by $\{5\}$ contains $\{5,10,15\}$.

A **non-principal (or free) ultrafilter** is defined to be an ultrafilter with no finite sets (which is equivalent to saying it is not generated by any single element $\{n\}$).

- Consider any cofinite set A. By definition, its complement $I - A$ is finite. Since a non-principal ultrafilter $\mathcal{U}$ has no finite sets (by definition), and either A or $I - A$ must be in $\mathcal{U}$ (by the definition of ultrafilter), it must be that $A \in \mathcal{U}$. Thus, $\mathcal{U}$ contains all cofinite subsets of I, and so, the Fréchet filter $\mathcal{F}_F$ is a subset of $\mathcal{U}$. Further, if we consider (for example) the sets $E = \{2,4,6, \dots\}$ and $O = \{1,3,5, \dots\}$, then neither is in $\mathcal{F}_F$ since neither is cofinite. However, being complements of each other, either $E \in \mathcal{U}$ or $O \in \mathcal{U}$. So, $\mathcal{U}$ contains a set not in $\mathcal{F}_F$, and thus, $\mathcal{F}_F$ is a proper subset of $\mathcal{U}$, i.e., $\mathcal{F}_F \subset \mathcal{U}$.
- Intuitively, sets in a non-principal ultrafilter $\mathcal{U}$ are "large" (measure 1 in a finitely additive sense), while their complements are "small" (measure 0). For example, in a non-principal ultrafilter, the even numbers might be "large" (if included in $\mathcal{U}$) or "small" (if their complement is in $\mathcal{U}$), but not both. However, finite sets like $\{1,2,3\}$ are always "small."
- For our construction of the hyperreals in Section 5.2, we need non-principal ultrafilters. As we shall see, principal ultrafilters yield trivial extensions. On the other hand, non-principal ultrafilters ensure a proper extension of $\mathbb{R}$ to $\mathbb{R}^*$ (the hyperreal numbers) that includes infinitesimals and infinites.
- Non-principal ultrafilters exist (by Zorn's lemma [43] applied to the partial order of filters extending the Fréchet filter). Why do we need Zorn's lemma? Any attempt to build a non-principal ultrafilter iteratively (e.g., deciding for each subset whether to include it or its complement while preserving filter properties) leads to a transfinite process that requires choices at uncountably many steps, which can't be justified without the Axiom of Choice or equivalently, Zorn's lemma.

The various types of filters and their relationships are depicted in Figure 2.

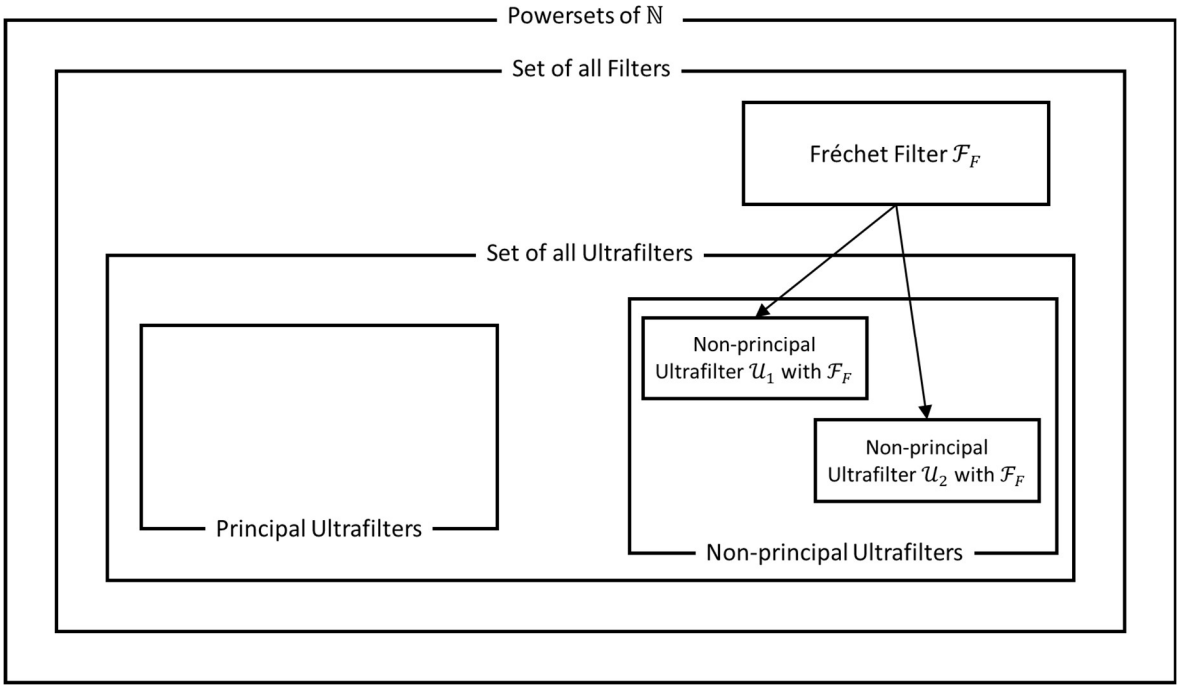


Figure 2. Filters and their relationships

The hierarchy implied in the figure is as follows:

- Powerset of $\mathbb{N}$ (the universe of all subsets)
    - All Filters (collections satisfying filter axioms)
        - Fréchet Filter (a specific, non-ultrafilter example)
        - All Ultrafilters (maximal filters - cannot be extended)
            - Principal Ultrafilters (generated by singletons)
            - Non-principal Ultrafilters (each of which contains the Fréchet filter as proper subset)

Key relationships expressed in the figure

- Fréchet Filter $\subset$ Every Non-principal Ultrafilter, i.e., the Fréchet filter is contained in each non-principal ultrafilter
- Principal Ultrafilters $\cap$ Fréchet Filter $= \emptyset$ because a principal ultrafilter $\mathcal{U}$ contains the finite set $\{n\}$, while the Fréchet filter contains *only* cofinite sets (whose complements are finite). Since $\{n\}$ is not cofinite, $\mathcal{U}$ and the Fréchet filter share no common sets.
- All Ultrafilters $=$ Principal Ultrafilters $\cup$ Non-principal Ultrafilters
- Ultrafilters $\subset$ Filters $\subset$ Powerset of $\mathbb{N}$

5.2 Ultrapower Construction

Fix a non-principal ultrafilter $\mathcal{U}$ on I. Typically, we take $I = \mathbb{N}$, the natural numbers, to ensure countably infinite support for sequences.

A non-principal ultrafilter $\mathcal{U}$ on $I = \mathbb{N}$ contains no finite sets, ensuring that equivalence classes capture "almost everywhere" agreement in a non-trivial way.

1. Consider the set of all sequences of real numbers, $\mathbb{R}^I = \{f: I \to \mathbb{R}\}$.

   For example, take the sequence $f(n) = e^n$ or $g(n) = \sqrt{n}$ for $n \in I$, or constant sequences like $h(n) = \pi$ for all $n \in I$.

2. Define an relation $\sim$ on $\mathbb{R}^I$ as follows: for sequences $f, g \in \mathbb{R}^I$, $f \sim g \Leftrightarrow \{n \in I: f(n) = g(n)\} \in \mathcal{U}$. This relation is reflexive, symmetric and transitive and is therefore, by definition, an equivalence relationship [45].

   For example, suppose $f(n) = g(n) = \sqrt[3]{n}$ for n even, but $g(n) = 7$ for n odd. If the set of even numbers is in $\mathcal{U}$ (a possible choice in a non-principal ultrafilter), then $f \sim g$ because $\{n: f(n) = g(n)\}$ contains the even numbers and is therefore in $\mathcal{U}$.

   If two sequences differ on only finitely many points (e.g., first 10 terms), they are always equivalent since the agreement set is cofinite, and hence in $\mathcal{U}$.

3. The set $J = \{h \in \mathbb{R}^I: \{n \in I: h(n) = 0\} \in U\}$ forms an ideal in the ring $\mathbb{R}^I$. (This J is precisely the equivalence class of the zero sequence, $[0]$). The **hyperreals**, denoted $\mathbb{R}^*$, are constructed as the quotient ring $\mathbb{R}^I/J$. The elements of this quotient are the equivalence classes $[f] = f + J = \{g \in \mathbb{R}^I: f - g \in J\}$, which is precisely the set $\{g \in \mathbb{R}^I: g \sim f\}$ under the equivalence relation $f \sim g \iff \{n \in I: f(n) = g(n)\} \in \mathcal{U}$. We say that $\mathbb{R}^I/J$ is the **ultrapower** of $\mathbb{R}$ with respect to $\mathcal{U}$**.**

   For example, the class $[f]$ where $f(n) = n$ represents an infinite hyperreal, while $[g]$ with $g(n) = \frac{1}{n}$ represents an infinitesimal.

4. Addition and multiplication are defined component-wise on representatives, i.e., $[f] + [g] = [f + g]$ and $[f \cdot g] = [f] \cdot [g]$ where $(f + g)(n) = f(n) + g(n)$ and $[f \cdot g](n) = f(n) \cdot g(n)$. These operations are well-defined because if $f \sim f_1$ and $g \sim g_1$, then $f + g \sim f_1 + g_1$ and $f \cdot g \sim f_1 \cdot g_1$ (since $\mathcal{U}$ is closed under finite intersections and preserves "large" sets). So, $\mathbb{R}^* = \mathbb{R}^I/J$ is a commutative ring [46] with the zero element being the constant 0 sequence, and unity 1 being the constant 1 sequence. Moreover, every nonzero element has a multiplicative inverse, so $\mathbb{R}^*$ is a field [5].

   For example, let $\alpha = [f]$ with $f(n) = n$, and $\beta = [g]$ with $g(n) = \frac{1}{n}$. Then $\alpha + \beta = [h]$ where $h(n) = n + \frac{1}{n}$. Since n dominates for large n (and "large" sets matter via $\mathcal{U}$), $\alpha + \beta$ behaves like an infinite number. For multiplication, $\alpha \cdot \beta = [\gamma]$ with $\gamma(n) = n \cdot \frac{1}{n} = 1$, so $\alpha \cdot \beta = [1] = 1$ (the real number 1).

5. Next, we embed the real numbers via the method of **diagonal embedding** [47]. For $r \in \mathbb{R}$, let $r = [c_r]$, where $c_r(n) = r$ for all n (constant sequence). This is a field embedding, identifying $\mathbb{R}$ as a subfield of $\mathbb{R}^*$.

   For example, the real number 2 is embedded as $[c_2]$ where $c_2(n) = 2$ for all $n \in \mathbb{N}$. Adding it to $\beta = \left[\frac{1}{n}\right]$ yields $\left[2 + \frac{1}{n}\right]$, which is infinitesimally close to 2.

6. Define a total order [48] on $\mathbb{R}^*$ as follows: For $[f], [g] \in \mathbb{R}^*$, $[f] < [g] \Leftrightarrow \{n \in I : f(n) < g(n)\} \in \mathcal{U}$. (Equality holds if and only if $\{n : f(n) = g(n)\} \in \mathcal{U}$, which is already defined via the equivalence relation $\sim$.) This is well-defined, a total order, and compatible with the field operations (making $\mathbb{R}^*$ an ordered field extension of $\mathbb{R}$).

   For $\alpha = [n]$ and the embedded real number $100 = [100\ \forall n \in \mathbb{N}]$, $\alpha > 100$ because $\{n : n > 100\}$ is cofinite (all $n > 100$), hence in $\mathcal{U}$. For $\beta = \left[\frac{1}{n}\right]$ and $0 = [0]$, $\beta > 0$ since $\{n \mid \frac{1}{n} > 0\} = \mathbb{N}$, which is in $\mathcal{U}$; but $\beta < 0.01$ because $\left\{n : \frac{1}{n} < 0.01\right\} = \{n > 100\}$, which again is cofinite and thus in $\mathcal{U}$.

The construction provided here is unique up to isomorphism under certain set-theoretic assumptions, e.g., continuum hypothesis [49], but multiple non-isomorphic hyperreal fields exist in general. Under the continuum hypothesis , all hyperreal fields of cardinality $2^{\aleph_0}$ (the continuum) constructed via ultrapowers over $\mathbb{N}$ are isomorphic. Without the continuum hypothesis, multiple non-isomorphic hyperreal fields exist, differing in saturation properties or the "richness" of their infinitesimal structure.

This structure enables the Transfer Principle: First-order statements true in $\mathbb{R}$ hold in $\mathbb{R}^*$, allowing seamless extension of calculus theorems (as discussed further in Section 5.5).

5.3 The Standard Part Function

For a hyperreal number $x \in \mathbb{R}^*$, if x is finite (i.e., $|x| < n$ for some standard natural number n), then there exists a unique standard real number $r \in \mathbb{R}$ such that $x - r$ is infinitesimal. This r is called the **standard part** of x, denoted $\text{st}(x)$.

The standard part map $\text{st} : \{\textit{finite hyperreals}\} \to \mathbb{R}$ is a surjective ring homomorphism with kernel equal to the set of infinitesimals. Intuitively, it "rounds" each finite hyperreal to the unique real number that is infinitely close to it.

Example: If ω is a positive infinite hyperreal and $\varepsilon = \frac{1}{\omega}$, then $\alpha = \omega + \varepsilon$ is infinite. So, $\text{st}(\alpha)$ is undefined. However, for $\beta = 2 + \varepsilon$, we have $\text{st}(\beta) = 2$.

In nonstandard analysis, the derivative of a function f at a point x can be defined as

$$f'(x) = \text{st}\left(\frac{f(x+\delta) - f(x)}{\delta}\right)$$

for any nonzero infinitesimal δ, provided the quotient is finite and the standard part is independent of the choice of δ. This aligns with Leibniz's intuition of derivatives as ratios of infinitesimals (as shown in Section 2.2).

5.4 Summary: The Hyperreal Universe from the Ultrapower

Figure 3 summarizes the ultrapower construction and the mappings to the real numbers, infinite hyperreals and finite hyperreals.

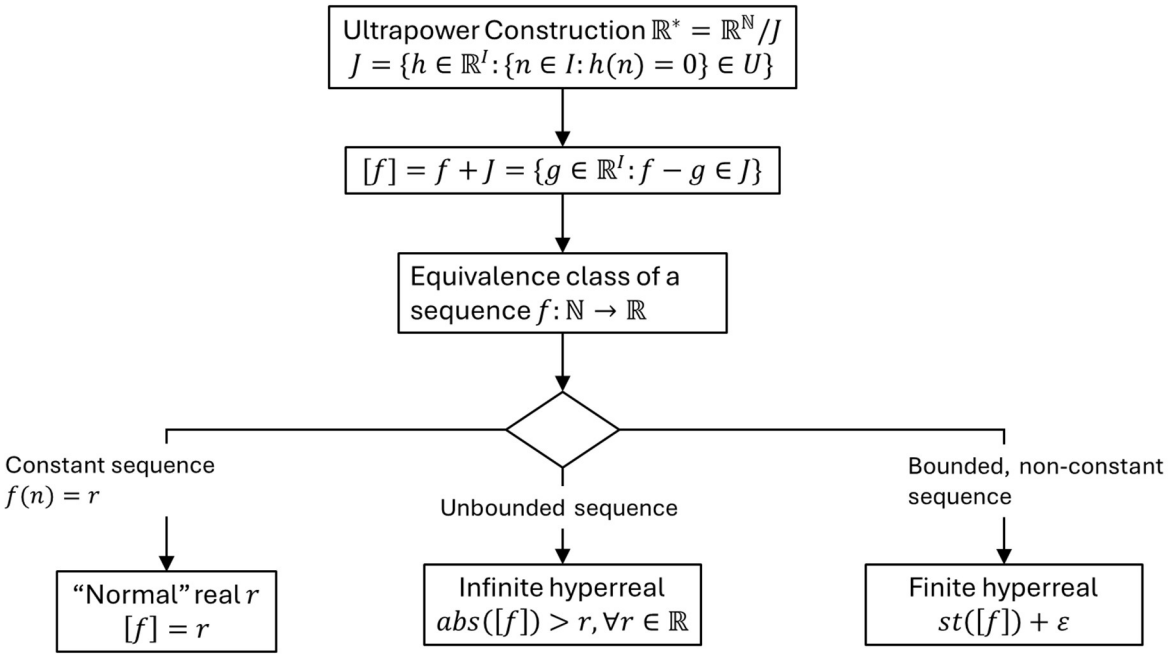


Figure 3. Summary of Ultrapower Construction

The Mapping of Standard Real Numbers ($\mathbb{R} \to \mathbb{R}^*$)

- Mechanism: Each standard real number $r \in \mathbb{R}$ is mapped to the corresponding constant sequence.
    - $r \to [c_r]$, where $c_r(n) = r$ for all $n \in \mathbb{N}$.
- Properties:
    - This mapping is a field embedding. It preserves all algebraic operations $(+, \times, -, \div)$ and the order relation $(<)$.
    - From inside $\mathbb{R}^*$, these elements are indistinguishable from their standard counterparts. We simply identify r with $[c_r]$.
- Example:
    - The real number 5 is the hyperreal $[\langle 5,5,5,5, \ldots \rangle]$.
    - The real number π is the hyperreal $[\langle \pi, \pi, \pi, \pi, \ldots \rangle]$.

The New, Non-Real Numbers in $\mathbb{R}^*$

These are the equivalence classes of **non-constant** sequences. There are two cases, i.e., finite hyperreals and infinite hyperreals.

A. Finite (Limited) Hyperreals

- Definition: A hyperreal $[f]$ is finite if there exists some "normal" real $r \in \mathbb{R}$ such that $-r < [f] < r$.
- Structure: Every finite hyperreal is infinitely close to exactly one standard real number. This unique real number is called its standard part.
    - $[f] = \text{st}([f]) + \varepsilon$, where ε is an infinitesimal.
- Example A.1 (Infinitesimal):
    - Let $f(n) = \frac{1}{n}$. The hyperreal $[f]$ is a positive number smaller than every positive real number but greater than zero. It is an infinitesimal.
    - Its standard part is 0. So, $\left[\left\langle 1, \frac{1}{2}, \frac{1}{3}, \dots \right\rangle\right] = 0 + \varepsilon$.
- Example A.2 (Finite, non-real):
    - Let $f(n) = \pi + \frac{1}{n}$. The hyperreal $[f]$ is greater than π but less than $\pi + \delta$ for any real $\delta > 0$.
    - Its standard part is π. So, $[f] = \left[\left\langle \pi + 1, \pi + \frac{1}{2}, \pi + \frac{1}{3}, \dots \right\rangle\right] = \pi + \varepsilon$.

B. Infinite Hyperreals

- Definition: A hyperreal $[f]$ is infinite if it is larger than every standard real or smaller than every standard real. Formally, $\text{abs}([f]) > r$ for all $r \in \mathbb{R}$.
- Mechanism: These are the equivalence classes of sequences that diverge to infinity.
- Example B.1 (Positive Infinite):
    - Let $f(n) = n$. The hyperreal $[f]$ is greater than $1,10,100,1000$, etc. It is an infinitely large number.
    - We write $[\langle 1,2,3,4, \dots \rangle] = \omega$ (or some other symbol that indicates infinity).
- Example B.2 (Negative Infinite):
    - Let $g(n) = -n^2$. The hyperreal $[g]$ is less than every standard real number.
- Example B.3 (Larger Infinite):
    - Let $h(n) = e^n$. The hyperreal $[h]$ is a different infinite number, and it is greater than $[f]$ from Example 1, because the set $\{n: e^n > n\} = \mathbb{N}$ is cofinite and hence in $\mathcal{U}$. This hints at a hierarchy of infinities in the hyperreals.

Hierarchy of Infinities

Not all infinite hyperreals are equal. Just like some infinities in mathematics are larger than others (e.g., the cardinality of the reals is greater than the cardinality of the natural numbers), some infinite hyperreals are numerically larger than others.

We have an ordered field of infinite numbers. We can perform arithmetic with them and compare their sizes.

1. $\omega = [n]$ (infinite)
2. $\omega + 5 = [n + 5]$ (also infinite, and slightly larger than ω)
3. $2\omega = [2n]$ (infinite, and greater than ω)
4. $\omega^2 = [n^2]$ (infinite, and much greater than ω, since $\{n: n^2 > n\}$is cofinite)
5. $\Omega = [e^n]$ (infinite, and dramatically greater than ω, as we've shown in Example B.3)
6. $\omega - 1{,}000{,}000 = [n - 10^6]$ (still infinite, as it's eventually greater than any real number)

The size of an infinite number is determined by the growth rate of the sequences that represent it. The ultrafilter $\mathcal{U}$ acts as a referee that consistently decides which of two growth rates is "larger almost everywhere."

This structure is perfectly analogous to the world of infinitesimals, where not all infinitesimals are equal (e.g., $\left[\frac{1}{n}\right]$is larger than $\left[\frac{1}{n^2}\right]$), but extended to the infinitely large. It creates a vast, ordered, and coherent number system that extends the real line in both the infinitely small and the infinitely large directions.

Key Takeaways

- Reals in $\mathbb{R}^*$ are the “constant” worlds.
- Infinitesimals in $\mathbb{R}^*$ are the “vanishingly small” worlds, like the ghosts of sequences converging to zero.
- Finite Hyperreals are “shadows” of real numbers, surrounded by a cloud of infinitesimals.
- Infinite Hyperreals are the “divergent” worlds, providing numerical counterparts to the concept of limits going to infinity.
- There is a hierarchy of infinities.

This structure, where every standard real is accompanied by a “halo” of infinitesimally close hyperreals and is followed by infinitely large numbers, is what makes nonstandard analysis work. The Transfer Principle (discussed in the following subsection) guarantees that this new structure behaves consistently with standard real analysis.

5.5 Key Properties

The following are some important properties of the hyperreals (stated without proof).

Definition: A first-order sentence is a statement in first-order logic, a formal language used to express properties and relationships in mathematical structures like the real numbers $\mathbb{R}$ or the hyperreals $\mathbb{R}^*$. First-order logic is restricted to quantifying over individual elements of a structure (e.g., numbers), not over sets of elements, functions, or other higher-order objects.

Transfer Principle: Any first-order sentence in the language of ordered fields that is true in $\mathbb{R}$ is also true in $\mathbb{R}^*$. This ensures $\mathbb{R}^*$ inherits all first-order properties from $\mathbb{R}$, such as algebraic laws and order properties.

Conversely, if a first-order sentence in the language of ordered fields is true in $\mathbb{R}^*$, then it is true in $\mathbb{R}$ as well. This mutual transfer applies because $\mathbb{R}$ and $\mathbb{R}^*$ are elementarily equivalent as ordered fields.

For example, the statement "for all $x \geq 0 \in \mathbb{R}^*$, there exists $y \in \mathbb{R}^*$ such that $y^2 = x$" (existence of square roots) holds in $\mathbb{R}$, so by transfer, it holds in $\mathbb{R}^*$, even for infinite or infinitesimal x.

On the other hand, the non-first-order statement "every nonempty subset bounded above has a least upper bound" holds in $\mathbb{R}$ but not in $\mathbb{R}^*$. This involves quantifying over sets, so it is second order. It's true in $\mathbb{R}$ (by the completeness axiom) but doesn't transfer to $\mathbb{R}^*$, where certain sets of positive infinitesimals lack a least upper bound. For example, the set of all positive infinitesimals has no least upper bound in $\mathbb{R}^*$.

Infinitesimals and Infinites: $\mathbb{R}^*$ properly extends $\mathbb{R}$. For example:

- Let $\varepsilon = \frac{1}{\omega}$, where ω is a positive infinite hyperreal; then $\varepsilon < \frac{1}{n}$ for all positive integers n. Specifically, $\varepsilon < \frac{1}{100}$ because $\omega > 100$. So, ε is a positive infinitesimal hyperreal.
- Let $H = \omega$; then $H > n$ for all standard positive integers n. For instance, $H > 100$ since $\omega > 100$. So, ω is an infinite hyperreal.
- If we let $H_1 = \omega$ and $H_2 = \omega^2$, then both H_1 and H_2 are larger than any real number, and thus both are infinite hyperreals (with $H_2 \gg H_1$).

5.6 Rationale for the Ultrapower Construction

While the ultimate goal is to work with the equivalence classes and their properties, the reason we don't just start with a definition of them is that the ultrapower construction isn't just a way to build the hyperreals – it's the only known way to explicitly construct a workable model that satisfies all the required properties, most critically the Transfer Principle.

Let's break down why the complex machinery is necessary.

5.6.1 The Dream vs. The Reality

The Dream: We want a ordered field $\mathbb{R}^*$ that

1. Contains $\mathbb{R}$ as a subfield.
2. Contains infinitesimals (numbers $\varepsilon > 0$ such that $\varepsilon < r$ for all positive $r \in \mathbb{R}$).
3. Satisfies the Transfer Principle: Every first-order statement (a statement about basic arithmetic and order, using logical quantifiers like ∀ and ∃) is true in $\mathbb{R}$ if and only if it is true in $\mathbb{R}^*$.

The Problem (reality):

How do we prove that such a magical object exists? We can't just declare it into existence. We must construct it from known mathematical objects (like sets, functions, and the standard reals) using the standard rules of set theory, i.e., Zermelo–Fraenkel set theory with the axiom of choice included (aka ZFC). The ultrapower is that construction.

5.6.2 Why the Ultrapower is Necessary

It Guarantees the Field and Order Properties are Well-Defined.

If we just said, “let there be a number ω bigger than all reals,” we immediately run into problems. What is $\omega - 1$? Is it less than ω? Sure. What is $\frac{1}{\omega}$? It should be an infinitesimal. This seems intuitive, but how do we define multiplication and addition for these new elements in a way that is consistent and doesn't lead to contradictions?

The ultrapower construction builds the operations from the ground up using component-wise operations on sequences. We then *prove* that these operations are well-defined on the equivalence classes (they don't depend on the representative you pick) and that they satisfy all the axioms of an ordered field. This is non-trivial, and the properties of the non-principal ultrafilter $\mathcal{U}$ are essential for these proofs.

It is the Engine of the Transfer Principle.

This is the most important reason. The Transfer Principle isn’t a magic wand; it's a theorem (Łoś’s Theorem) that must be *proven*. The ultrapower construction provides the framework to prove it.

How it works:

- A first-order statement P (e.g., "every number has a square root") can be broken down into its logical components.
- Łoś's Theorem states: P is true in $\mathbb{R}^*$ if and only if the set of indices n for which P is true for the sequence component $f(n)$ in the standard model $\mathbb{R}$ is a “large” set, i.e., it belongs to the non-principal ultrafilter $\mathcal{U}$.
- P being true in $\mathbb{R}$ means it's true for *all* standard reals. For a constant sequence, the set of indices where it's true is the entire index set $\mathbb{N}$, which is definitely in $\mathcal{U}$. So P holds for all standard reals inside $\mathbb{R}^*$.
- The magic is that this logic extends perfectly to statements involving non-standard elements.

Without the precise definition of “large set” (i.e., the non-principal ultrafilter $\mathcal{U}$) and the construction of the model, we would have no way to state, let alone prove, this theorem. The Transfer Principle would just be an unproven wish.

It Controls the "Almost Everywhere" Logic.

The non-principal ultrafilter $\mathcal{U}$ formalizes the idea of “for almost all n” or “almost everywhere.” This is crucial for making the intuitive idea of “agreement on a large set” into a rigorous, mathematical concept.

- Why is $[f] \neq 0$ if $f(n)$ is only zero for a finite set of n? Because finite sets are “small” and not in $\mathcal{U}$.
- Why is the order $[f] < [g]$ a total order? Because for any two sequences f and g, the set $\{n: f(n) < g(n)\}$ is either in $\mathcal{U}$ or its complement $\{n: f(n) \geq g(n)\}$ is in $\mathcal{U}$. This is a defining property of an ultrafilter.

If we tried to use a simpler equivalence class definition without this machinery, we would likely end up with contradictions or a structure that isn't a field (e.g., it might have zero divisors).

5.6.3 Analogy: Real Numbers

Think about how we construct the real numbers from the rationals.

- The Dream: We want a complete, ordered field with no gaps.
- The Reality: We can't just declare it. We have to construct it using, for example, Dedekind cuts or equivalence classes of Cauchy sequences.
- The construction with Cauchy sequences is very analogous to the ultrapower:
    - Objects: Sequences of rationals $\{q_n\}$.
    - Equivalence Relation: $\{q_n\} \sim \{p_n\}$ if $\lim_{n\to\infty}(q_n - p_n) = 0$. (This is like saying "the sequence $q_n - p_n$ is zero almost everywhere").
    - Result: The set of equivalence classes $\mathbb{R}$ is a complete ordered field.

The complexity of the Cauchy sequence construction is the price we pay to *prove* that the real numbers exist as we imagine them. The ultrapower is the analogous price for the hyperreals.

5.6.4 Conclusion

You don't define the hyperreals in terms of the ultrapower because it's the only definition. You construct them via the ultrapower to prove that a structure with the desired properties (especially Transfer) actually exists. Once the construction is complete and the properties are proven, mathematicians often black box the ultrapower and work directly with the intuitive properties of the hyperreals and the Transfer Principle. But the rigorous foundation is indispensable.

6 Dual Numbers: Algebra and Geometry

These vanishing quantities… are not zero, yet they are less than any assignable quantity.

Gottfried Wilhelm Leibniz

6.1 Introduction

Dual numbers, introduced by William Clifford in 1873, represent a simple yet powerful extension of the real numbers that incorporates a nilpotent element ε satisfying $\varepsilon^2 = 0$. This algebraic structure allows dual numbers to model infinitesimal perturbations, making them particularly useful in automatic differentiation, kinematics, and geometric applications. Unlike hyperreal numbers, which form a field with infinitesimals that are non-zero and invertible, dual numbers form a ring with zero divisors, emphasizing their algebraic rather than analytical nature. This section explores the definition, properties, and operations of dual numbers, their geometric interpretations, and their role in computing derivatives. We also extend the discussion to hyper-dual numbers for higher-order derivatives, bridging to automatic differentiation (discussed later in this book).

Dual numbers have historical roots in the work of Clifford and Eduard Study, who applied them to represent dual angles for skew lines in space, i.e., pairs of lines that do not intersect and are not parallel, existing in three-dimensional space or higher. Their simplicity, requiring only basic algebra, makes them accessible for computational implementations in fields like game programming and mechanics.

6.2 Definition and Algebraic Structure

A dual number is expressed as $z = a + b\varepsilon$, where a and b are real numbers (the real and dual parts, respectively), and ε is a nilpotent element with $\varepsilon^2 = 0$ and $\varepsilon \neq 0$. [Note that $\varepsilon^2 = 0$ implies that $\varepsilon^n = 0, n = 2,3,4 \ldots$] The set of dual numbers, denoted $\mathbb{D}$, forms a commutative algebra [50] of dimension two over the reals and is an Artinian local ring [52] with maximal ideal generated by ε. This means you can add, subtract, and multiply dual numbers like real numbers, and every dual number can be built from the two basis elements 1 and ε.

Algebraically, $\mathbb{D}$ can be viewed as the quotient ring $\mathbb{R}[\varepsilon]/\varepsilon^2$, where polynomials in ε are truncated beyond linear terms due to the nilpotency of ε and (ε^2) is the ideal generated by ε^2. This structure introduces zero divisors: for instance, $\varepsilon \cdot \varepsilon = 0$, but $\varepsilon \neq 0$, so $\mathbb{D}$ is not a field.

[**Author's Remark**: If you are not familiar with abstract algebra, consider reading Section 3.]

Dual numbers can also be represented as 2×2 matrices: $a + b\varepsilon \rightarrow \begin{bmatrix} a & b \\ 0 & a \end{bmatrix}$, where matrix multiplication preserves the algebra, and ε corresponds to $\begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$, which squares to the zero matrix.

Example: Consider $z_1 = 2 + 3\varepsilon$ and $z_2 = 4 + 5\varepsilon$. Their product is

$$z_1 z_2 = (2 \cdot 4) + (2 \cdot 5 + 3 \cdot 4)\varepsilon + 15\varepsilon^2 = 8 + 22\varepsilon$$

noting that the ε^2 is 0.

6.3 Properties and Operations

Addition and multiplication are defined component-wise, respecting $\varepsilon^2 = 0$:

- Addition: $(a + b\varepsilon) + (c + d\varepsilon) = (a + c) + (b + d)\varepsilon$
- Multiplication: $(a + b\varepsilon)(c + d\varepsilon) = ac + (ad + bc)\varepsilon$

The conjugate of $z = a + b\varepsilon$ is $\bar{z} = a - b\varepsilon$, and the norm is $z\bar{z} \equiv a^2$, which is real-valued. Division is possible if the real part of the denominator is non-zero:

$$\frac{a + b\varepsilon}{c + d\varepsilon} = \frac{a + b\varepsilon}{c + d\varepsilon} \cdot \frac{c - d\varepsilon}{c - d\varepsilon} = \frac{ac + (bc - ad)\varepsilon}{c^2} = \frac{a}{c} + \frac{bc - ad}{c^2}\varepsilon, \qquad c \neq 0$$

If the real part of the denominator is zero, the number is a zero divisor and not invertible.

Example 1: The inverse of $1 + \varepsilon$ is $1 - \varepsilon$, since $(1 + \varepsilon)(1 - \varepsilon) = 1 - \varepsilon^2 = 1$.

Example 2: Show that $e^{a+b\varepsilon} = e^a(1 + b\varepsilon)$ where e is Euler's number.

For a dual number $a + b\varepsilon$, we can define $e^{a+b\varepsilon}$ using the usual power series for e^x, i.e.,

$$e^{a+b\varepsilon} = \sum_{n=0}^{\infty} \frac{(a + b\varepsilon)^n}{n!}$$

We use the binomial theorem:

$$(a + b\varepsilon)^n = a^n + na^{n-1}(b\varepsilon) + \frac{n(n-1)}{2}a^{n-2}(b\varepsilon)^2 + \cdots$$

But $\varepsilon^2 = 0$, so all terms with ε^2 or higher powers vanish.

Thus, we have

$$(a + b\varepsilon)^n = a^n + na^{n-1}b\varepsilon$$

Plug the above into the series equation, we get

The first sum is

$$\sum_{n=0}^{\infty} \frac{a^n}{n!} = e^a$$

The second sum: for $n = 0$, the term is 0 because $na^{n-1} = 0$ when $n = 0$ (careful: a^{-1} is undefined, but $\frac{n}{n!} = 0$ for $n = 0$ anyway). For $n \geq 1$:

$$b\varepsilon \sum_{n=1}^{\infty} \frac{na^{n-1}}{n!} = b\varepsilon \sum_{n=1}^{\infty} \frac{a^{n-1}}{(n-1)!} = b\varepsilon \sum_{m=0}^{\infty} \frac{a^m}{m!} = b\varepsilon e^a$$

So, we have

$$e^{a+b} = e^a + b\varepsilon e^a = e^a(1 + b\varepsilon)$$

6.4 Geometric Interpretations

Geometrically, dual numbers extend the real numbers similarly to complex numbers, but their geometry is parabolic rather than circular. This is evident when examining the "unit circle" – the set of dual numbers $a + b\varepsilon$ where the norm is equal to 1. Since the norm of the dual number $z = a + b\varepsilon$ is defined to be a^2, the "unit circle" consists of points $a = \pm 1$. This condition is independent of the dual component b, resulting in two vertical lines in the dual plane at $a = 1$ and $a = -1$.

[The mention of "parabolic" in the above paragraph refers to a classification in algebra and geometry where number systems like this are grouped by the type of conic section their "unit circle" resembles under a quadratic form. Complex numbers produce a circular (elliptic) unit circle via $x^2 + y^2 = 1$. In contrast, dual numbers produce a parabolic (degenerate) form, where the equation simplifies to $x^2 = 1$ (independent of y), resulting in two parallel lines, i.e., a degenerate conic that's classified as parabolic in this context. This analogy extends to broader geometric structures: dual numbers model "parabolic geometry" (infinitesimal or affine-like transformations), similar to how complexes model circular/Euclidean geometry. The term "parabolic numbers" is sometimes used interchangeably with dual numbers in the literature to emphasize this.]

Multiplication by a dual number of the form $1 + b\varepsilon$ acts as a shear transformation[4] parallel to the dual axis (i.e., the axis corresponding to ε). For a dual number $x + y\varepsilon$, multiplying by $1 + b\varepsilon$ gives:

$$(x + y\varepsilon)(1 + b\varepsilon) = x + xb\varepsilon + y\varepsilon + yb\varepsilon^2 = x + (xb + y)\varepsilon$$

This transformation preserves the real component x but shifts the dual component y by an amount proportional to x. Consequently, a "cycle" of constant distance from the origin is not a circle but a parabola in the dual plane.

These properties make dual numbers foundational for representing spatial transformations.

In three-dimensional space, dual numbers are used to construct dual quaternions, a powerful tool for representing rigid body motions (combinations of rotation and translation). A key concept here is the dual angle, denoted $\hat{\theta} = \theta + d\varepsilon$. This single quantity elegantly encodes the relationship between two skew lines (non-parallel, non-intersecting lines): θ is the angle between the lines, and d is the shortest distance between them. This application is crucial in fields like robotics and computer graphics for describing screw motions and the kinematics of rigid bodies.

6.5 Applications in Differentiation

Consider a differentiable function $f: \mathbb{R} \to \mathbb{R}$. We want to evaluate f at the dual number $x + b\varepsilon$. The Taylor series of f around a point x for a perturbation h is given by

$$f(x + h) = f(x) + f'(x)h + \frac{f''(x)}{2!}h^2 + \frac{f'''(x)}{3!}h^3 + \cdots$$

Here, the series assumes f is analytic or at least C^∞ (infinitely differentiable), but the dual number approach works for once-differentiable f since only the first-order term survives.

[4] Shear transformation is a type of geometric transformation that distorts the shape of an object by shifting its points in a specific direction, while keeping the area the same.

Next, substitute $h = b\varepsilon$ and note that since $\varepsilon^2 = 0$, it follows the $\varepsilon^k = 0, k \geq 2$. Thus, all terms beyond the linear one vanish exactly, yielding a closed-form expression without infinite series summation.

$$f(x + b\varepsilon) = f(x) + f'(x)b\varepsilon + \frac{f''(x)}{2!}(b\varepsilon)^2 + \frac{f'''(x)}{3!}(b\varepsilon)^3 + \cdots = f(x) + f'(x)b\varepsilon$$

Key conceptual point: In standard calculus, dx is often thought of as an infinitesimal quantity where $(dx)^2$ is infinitesimally small but not precisely zero. The dual number ε, in contrast, is a purely algebraic entity defined by the *exact* property $\varepsilon^2 = 0$. This algebraic nilpotency is what forces the Taylor series to truncate *exactly* after the linear term, making the derivative computation not just an approximation but an exact, symbolic result within the dual number system.

If we let $b = 1$ in the above equation, we obtain

$$f(x + \varepsilon) = f(x) + f'(x)\varepsilon$$

This is analogous to the differential $df = f'(x)dx \approx f(x + dx) - f(x)$ from calculus. Here, ε serves as an algebraic representation of an infinitesimal perturbation, similar to dx, but within the rigorous framework of dual numbers. The nilpotency of ε ensures that the Taylor series truncates exactly after the linear term, providing an exact expression for the derivative without higher-order terms.

In practice, this enables forward-mode automatic differentiation: propagating dual numbers through a function computes both the value and derivative in a single pass, with applications in optimization, physics simulations, and machine learning gradients.

Example 1: For $f(x) = x^2$, $f(2 + \varepsilon) = (2 + \varepsilon)^2 = 4 + 4\varepsilon + \varepsilon^2 = 4 + 4\varepsilon$, so $f'(2) = 4$.

Example 2: For $f(x) = e^x$, $f(x + \varepsilon) = e^{x+\varepsilon}$. By Example 2 in Section 6.2, we have that $e^{a+b\varepsilon} = e^a(1 + b\varepsilon)$ and so, $e^{x+\varepsilon} = e^x + \varepsilon e^x$ which implies that $f'(x) = e^x$.

Example 3: For $f(x) = e^x$, we compute $f(a + \varepsilon b) = e^{a+\varepsilon b} = e^a e^{\varepsilon b}$. Since

$$e^{\varepsilon b} = 1 + \varepsilon b + \frac{1}{2}(\varepsilon b)^2 + \cdots = 1 + \varepsilon b,$$

it follows that $e^{a+\varepsilon b} = e^a + \varepsilon b e^a$, matching $f(a) + \varepsilon b f'(a)$ where $f'(x) = e^x$. This illustrates how the method handles transcendental functions seamlessly.

For multivariate functions $f: \mathbb{R}^n \to \mathbb{R}$, dual numbers extend to $\mathbb{R}^n[\varepsilon]$, allowing computation of partial derivatives or the full Jacobian via component-wise propagation. Limitations include scalability for high dimensions (better suited to adjoint methods via reverse-mode AD) and the need for analytic expressions, though symbolic tools like SymPy (a Python package) can automate implementations.

This topic will be discussed in more detail later in Section 12. Also, see the discussion in Section 4 of Fratini [53].

6.6 Extension to Vectors and Matrices

The formalism of dual numbers extends naturally to vector and matrix spaces, enabling the differentiation of functions with vector inputs or outputs.

We define dual vectors as follows:

$$v = a + b\varepsilon,\ \ u = c + d\varepsilon,\ \ a, b, c, d \in \mathbb{R}^n,\ \ \varepsilon^2 = 0$$

Addition/subtraction and multiplication (dot product) are defined as follows:

$$v \pm u = (a \pm c) + (b \pm d)\varepsilon$$
$$v \cdot u = (a \cdot c) + (a \cdot d + b \cdot c)\varepsilon$$

A similar extension can be made for matrices, i.e.,

$A = A_0 + A_1\varepsilon$ where A_0 and A_1 are matrices of the same size andε is a formal symbol satisfying $\varepsilon^2 = 0$. A_0 is sometimes called the "real part" (or standard part) and A_1 is known as the "dual part" (or infinitesimal perturbation).

For dual matrices $A = A_0 + A_1\varepsilon$ and $B = B_0 + B_1\varepsilon$:

Addition: $A + B = (A_0 + B_0) + (A_1 + B_1)\varepsilon$

Scalar multiplication : $cA = cA_0 + cA_1\varepsilon$

Matrix multiplication (if dimensions allow): $AB = A_0B_0 + (A_0B_1 + A_1B_0)\varepsilon$

6.7 Extension to Hyper-Dual Numbers

To compute higher-order derivatives exactly, we need an algebraic structure that preserves higher-order perturbation terms. This leads to the concept of hyper-dual numbers. Hyper-dual numbers extend dual numbers to higher dimensions for computing second-order or higher-order derivatives without truncation errors.

A hyper-dual number is of the form

$$a + b_1\varepsilon_1 + b_2\varepsilon_2 + c\varepsilon_1\varepsilon_2, \text{ where } \varepsilon_1^2 = \varepsilon_2^2 = 0 \text{ but } \varepsilon_1\varepsilon_2 \neq 0$$

If we consider the ring $\mathbb{R}[\varepsilon_1, \varepsilon_2]$ and the ideal I generated by $\{\varepsilon_1^2, \varepsilon_2^2\}$, then the above is essentially the quotient ring $\mathbb{R}[\varepsilon_1, \varepsilon_2\ ]/I$.

This allows exact computation of Hessians[5] and higher derivatives in applications like optimization and eigensystem analysis.

For second derivatives, we have

$$f(x + \varepsilon_1 + \varepsilon_2) = f(x) + f'(x)(\varepsilon_1 + \varepsilon_2) + \frac{1}{2}f''(x)(\varepsilon_1 + \varepsilon_2)^2 + \cdots$$

but with nilpotency, it yields

$$f(x) + f'(x)(\varepsilon_1 + \varepsilon_2) + f''(x)\varepsilon_1\varepsilon_2$$

[5] The Hessian is a square matrix of second-order partial derivatives of a scalar-valued function. It describes the local curvature of a function of many variables.

For example, find the second derivative of $f(x) = x^3$ using the above formula.

We have that

$$f(x + \varepsilon_1 + \varepsilon_2) = (x + \varepsilon_1 + \varepsilon_2)^3$$

$$= x^3 + 3x^2\varepsilon_1 + 3x^2\varepsilon_2 + 3x\varepsilon_1^2 + 6x\varepsilon_1\varepsilon_2 + 3x\varepsilon_2^2 + \varepsilon_1^3 + 3\varepsilon_1^2\varepsilon_2 + 3\varepsilon_1\varepsilon_2^2 + \varepsilon_2^3$$

$$= \boldsymbol{x^3 + 3x^2(\varepsilon_1 + \varepsilon_2) + 6x\varepsilon_1\varepsilon_2}$$

So, the coefficient of $(\varepsilon_1 + \varepsilon_2)$ is $3x^2$, which is $f'(x)$, and the coefficient of $\varepsilon_1\varepsilon_2$ is $6x$, which is $f''(x)$. This matches the known derivatives from basic calculus.

. . .

Generalizations to arbitrary orders use multiple nilpotents, relating back to dual numbers for first-order cases. For exact computation of all k^{th} order partial derivatives, we use k distinct nilpotents, i.e.,

$$\mathbb{R}[\varepsilon_1, \varepsilon_2, \dots, \varepsilon_k\ ]/(\varepsilon_1^2, \varepsilon_2^2, \dots, \varepsilon_k^2\ )$$

This implies the following:

- Each $\varepsilon_i^2 = 0$ (individually nilpotent)
- But products of distinct ε_i are non-zero: $\varepsilon_1\varepsilon_2 \neq 0$, $\varepsilon_1\varepsilon_2\varepsilon_3 \neq 0$, etc.
- The ideal kills (makes 0) any term containing ε_i^2 for any i

Example: Third order derivatives:

Use 3 nilpotents: $\mathbb{R}[\varepsilon_1, \varepsilon_2, \varepsilon_3\ ]/(\varepsilon_1^2, \varepsilon_2^2, \varepsilon_3^2\ )$

Evaluate: $f(x + \varepsilon_1 + \varepsilon_2 + \varepsilon_3)$

The expansion gives:

- Coefficient of $\sum \varepsilon_i \rightarrow$ first derivatives
- Coefficient of $\sum \varepsilon_i\varepsilon_j \rightarrow$ second derivatives
- Coefficient of $\sum \varepsilon_i\varepsilon_j\varepsilon_k \rightarrow$ third derivative

For example, consider $f(x) = x^4$**:**

$$f(x + \varepsilon_1 + \varepsilon_2 + \varepsilon_3) = x^4 + 4x^3(\varepsilon_1 + \varepsilon_2 + \varepsilon_3) + 12x^2 \sum_{i<j} \varepsilon_i\varepsilon_j + 24x\ \varepsilon_1\varepsilon_2\varepsilon_3$$

This gives us the following:

- Coefficient of $\varepsilon_1 + \varepsilon_2 + \varepsilon_3$ implies $4x^3 = f'(x)$
- Coefficient of $\varepsilon_i\varepsilon_j$ implies $12x^2 = \frac{f''(x)}{2!} \times 2! = f''(x)$
- Coefficient of $\varepsilon_1\varepsilon_2\varepsilon_3$ implies $24x = \frac{f'''(x)}{3!} \times 3! = f'''(x)$

Another approach is to use a single nilpotent with higher powers. In particular, we consider the principal ideal generated by ε^{n+1} and associated quotient ring $\mathbb{R}[\varepsilon]/\varepsilon^{n+1}$ where $\varepsilon^{n+1} = 0$ but $\varepsilon^k \neq 0$ for $k \leq n$. Elements of the quotient ring are of the form

$$a_0 + a_1\varepsilon + a_2\varepsilon^2 + \cdots + a_n\varepsilon^n, a_i \in \mathbb{R}$$

For example, let's consider derivatives up to order 3. In this case we use the quotient ring $\mathbb{R}[\varepsilon]/\varepsilon^4$.

The Taylor series truncates naturally as follows:

$$f(x+\varepsilon) = f(x) + f'(x)\varepsilon + \frac{f''(x)}{2!}\varepsilon^2 + \frac{f'''(x)}{3!}\varepsilon^3$$

All derivatives appear directly in the coefficients!

For $f(x) = x^4$ in $\mathbb{R}[\varepsilon]/\varepsilon^4$, we have that

$$f(x+\varepsilon) = (x+\varepsilon)^4 = x^4 + 4x^3\varepsilon + 6x^2\varepsilon^2 + 4x\varepsilon^3$$

Thus, $f'(x) = 4x^3$, $\frac{f''(x)}{2!} = 6x^2$ and $\frac{f'''(x)}{3!} = 4x$.

. . .

For multivariate functions with higher-order derivatives, we combine both ideas and use of the following quotient ring

$$\mathbb{R}[\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_m]/(\varepsilon_1^{n_1+1}, \varepsilon_2^{n_2+1}, \ldots, \varepsilon_m^{n_m+1})$$

This allows computing of all partial derivatives:

$$\frac{\partial^{k_1+k_2+\cdots+k_m} f}{\partial x_1^{k_1}\, \partial x_2^{k_2} \cdots \partial x_m^{k_m}}$$

where $k_i \leq n_i$ for each variable.

7 Comparing Hyperreals and Duals: Infinitesimal Approaches

The infinitely small is a necessary concept in differential calculus… it is a useful fiction."

Augustin-Louis Cauchy

7.1 Introduction

Hyperreal numbers and dual numbers both extend the real numbers to incorporate infinitesimal quantities, enabling intuitive approaches to calculus and differentiation. Hyperreals, rooted in non-standard analysis, provide a field extension with infinite and infinitesimal elements for rigorous analytical work, as discussed in Section 5. Dual numbers, introduced in Section 6, form a ring with a nilpotent infinitesimal for algebraic computations, particularly in automatic differentiation (Section 12). While both frameworks revive Leibnizian infinitesimals, they differ fundamentally in structure, properties of their infinitesimals, and applications. This section examines these similarities and differences, highlighting how hyperreals emphasize theoretical rigor in analysis, whereas dual numbers prioritize computational efficiency. These contrasts lay the groundwork for their integration with category theory and automatic differentiation (to be discussed in subsequent chapters).

Similarities include their use of infinitesimals to simplify derivative definitions and their extension of real arithmetic. Differences arise in algebraic structure (field vs. ring), the nature of infinitesimals (invertible vs. nilpotent), and scope (analytical vs. algebraic/computational).

[**Ring versus Field**: A **ring** is a set with two binary operations, typically called addition (+) and multiplication (·). The set, along with these operations, must satisfy several axioms:

- It forms an abelian (i.e., commutative) group under addition. This means addition is associative and commutative, there's an additive identity 0, and every element has an additive inverse (a negative).
- Multiplication is associative, and there is a multiplicative identity 1
- The multiplication operation distributes over addition.

A key point is that a ring does not require every non-zero element to have a multiplicative inverse.

A **field** is a special type of ring. It satisfies all the axioms of a ring, plus a few more crucial conditions that make it more like the number systems we're used to, such as the rational numbers, real numbers, or complex numbers.

A field has these additional properties:

- The set of all non-zero elements forms an abelian group under multiplication. This means multiplication is associative and commutative, there's a multiplicative identity (one, 1), and every non-zero element has a multiplicative inverse. This last point is the key distinction.
- The multiplicative identity, 1, is not equal to the additive identity, 0.

Because of the multiplicative inverse property, you can perform all four arithmetic operations in a field, i.e., addition, subtraction, multiplication, and division (by any non-zero element) . The presence of division is what gives a field a richer structure than a ring. The other difference is commutativity over multiplication.

These concepts are explained further in Sections 3.3 and 3.4.]

7.2 Structural Similarities and Differences

Both hyperreals and dual numbers extend the real numbers, but their algebraic structures diverge significantly.

Similarities:

- Both embed the real numbers $\mathbb{R}$ as a substructure: In the hyperreals $\mathbb{R}^*$, real numbers are the finite elements without infinitesimal parts. In dual numbers $\mathbb{D}$, real numbers form the standard (or real) part of a dual number with a value of 0 for the dual part.
- Arithmetic operations (addition, multiplication) extend naturally from $\mathbb{R}$, allowing infinitesimal perturbations.
- Both support infinitesimal approximations:
    - Dual numbers implicitly approximate via truncation at $\varepsilon^2 = 0$.
    - In the hyperreal number system $\mathbb{R}^*$ the relation $x \sim y$ is a way to express that two hyperreal numbers are "infinitesimally close" to each other. Specifically, for two hyperreal numbers $x, y \in \mathbb{R}^*$, we say $x \sim y$ if their difference $|x - y|$ is an infinitesimal. An infinitesimal in the hyperreals is a number $\delta \in \mathbb{R}^*$ such that $\delta \neq 0$ and $|\delta| < r$ for every positive real number $r \in \mathbb{R}$.

A classic and concrete example of two hyperreal numbers that are infinitesimally close is as follows:

> Let δ be a positive infinitesimal hyperreal number (so $\delta \neq 0$, but $|\delta| < r$ for every positive real number r, and thus $\delta \sim 0$). Consider $x = 1 + \delta$ and $y = 1$ (where 1 is the standard real number embedded in the hyperreals). The difference is $x - y = \delta$, which is infinitesimal by definition. Therefore, $x \sim y$, meaning $1 + \delta$ is “infinitely close” to 1, but not equal to it.
>
> This illustrates how hyperreals extend the reals by allowing such tiny deviations that are smaller than any positive real, yet nonzero.

Differences:

- Field vs. Ring: Hyperreals form an ordered field, satisfying all field axioms (including multiplicative inverses for non-zero elements) and the transfer principle for first-order properties. Dual numbers form a commutative ring but not a field due to zero divisors (e.g., $\varepsilon \cdot \varepsilon = 0$, but $\varepsilon \neq 0$), preventing division by elements with zero real part.
- Archimedean Property[6]: Hyperreals are non-Archimedean, containing infinite elements larger than any real (e.g., $H > n$ for all natural n). Dual numbers retain an Archimedean-like behavior in their real parts but lack true infinites due to their nilpotent (dual) part.

[6] The **Archimedean property**, named after the ancient Greek mathematician Archimedes of Syracuse, is a property held by some algebraic structures, such as ordered or normed groups, and fields. The property, as typically construed, states that given two positive numbers x and y, there is an integer n such that $nx > y$. It also means that the set of natural numbers is not bounded above. Roughly speaking, it is the property of having no *infinitely large* or *infinitely small* elements.

- Construction Complexity: Hyperreals require advanced set theory (e.g., ultrafilters and ultrapowers), making them theoretically rich but computationally intensive. Dual numbers are simpler, constructed as a quotient ring $\mathbb{R}[\varepsilon]/\varepsilon^2$, ideal for programming (no pun intended).

These structural distinctions influence their handling of infinitesimals and applications.

7.3 Infinitesimals: Invertible vs. Nilpotent

Infinitesimals are central to both systems, but their properties differ markedly.

Similarities:

- Both formalize "infinitely small" quantities: Hyperreal infinitesimals δ satisfy $0 < |\delta| < r$ for any positive real r, while dual nilpotents ε satisfy $\varepsilon \neq 0$ but $\varepsilon^2 = 0$.
- They enable derivative-like computations via infinitesimal increments, mimicking Leibniz's notation $\frac{dx}{dy}$.

Differences:

- Invertibility: Hyperreal infinitesimals are invertible ($\frac{1}{\delta}$ is infinite if $\delta \neq 0$), allowing full field operations and comparisons between different infinitesimal scales. Dual infinitesimals are nilpotent ($\varepsilon^2 = 0$), non-invertible, and lead to truncation in higher powers, resembling first-order Taylor approximations.
- Higher Powers:
    - In hyperreals, $\delta^n \neq 0$ for any finite n, enabling infinite series and non-standard models. This makes hyperreals suitable for non-standard analysis, where full transfer of real analysis theorems is required, and duals ideal for automatic differentiation, where computational efficiency and exact first derivatives are key.
    - In duals, ε^n for $n \geq 2$, limiting infinite series to linear approximations but simplifying computations.
- Approximation: Hyperreals allow precise multi-scale infinitesimals (e.g., δ and δ^2 are distinct orders), while duals collapse higher orders, making them less flexible for advanced analysis but efficient for derivatives.

7.4 Applications in Differentiation: Comparisons

Both systems facilitate differentiation without explicit limits, but their approaches suit different contexts.

Similarities:

- Derivatives as infinitesimal ratios: In hyperreals, $f'(x) = st\left(\frac{f(x+\delta)-f(x)}{\delta}\right)$; in duals, $f(x+\varepsilon) = f(x) + f'(x)\varepsilon$.
- In both cases, they avoid numerical instability in finite differences and symbolic overhead.

Differences:

- Theoretical vs. Computational: Hyperreals excel in proofs (e.g., intermediate value theorem via the Transfer Principle), but require the handling standard parts. Duals are ideal for automatic differentiation, directly yielding exact derivatives in forward mode without approximation.
- Higher-Order Derivatives: Basic duals handle first-order, with extensions such as hyper-duals to manage higher orders (see Section 6.7). Hyperreals naturally support higher derivatives via iterated infinitesimals.
- Scope: Hyperreals apply to broader analysis (e.g., integrals, topology); duals focus on algebraic functions and automatic differentiation in machine learning.

7.5 Examples

Example 1: Find the first derivative of $f(x) = x^2$ at $x = 3$ using the hyperreal and dual number approach.

- **Hyperreal Approach**: Let δ be an infinitesimal. $f(3+\delta) = (3+\delta)^2 = 9 + 6\delta + \delta^2$. Then $\frac{f(3+\delta)-f(3)}{\delta} = 6 + \delta$. The standard part $st(6 + \delta) = 6$, so $f'(3) = 6$. Note $\frac{\delta^2}{\delta} = \delta \approx 0$, but δ is invertible.
- **Dual Approach**: Let ε be nilpotent. $f(3+\varepsilon) = (3+\varepsilon)^2 = 9 + 6\varepsilon + \varepsilon^2 = 9 + 6\varepsilon$ (since $\varepsilon^2 = 0$). The dual part directly gives $f'(3) = 6$. No standard part needed, but ε is not invertible.

This shows duals' simplicity for computation vs. hyperreals' flexibility.

Example 2: Higher Powers and Approximation

This example illustrates how hyperreals and dual numbers handle higher powers in function approximations. For example, we will make use of the Maclaurin series for $\sin x$ around 0, i.e.,

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \cdots$$

We approximate $\sin x$ for "small" x, where "small" means an infinitesimal input in each approach. The key difference is that hyperreal infinitesimals allow higher powers to be non-zero (but infinitesimally smaller), while dual nilpotents force truncation at order 2 and above.

Hyperreal Approach: Consider $\sin \delta$, where δ is a non-zero hyperreal infinitesimal ($0 < |\delta| < r$ for any positive real r). By the transfer principle, the Maclaurin series applies to hyperreals:

$$\sin(\delta) = \delta - \frac{\delta^3}{3!} + \frac{\delta^5}{5!} - \frac{\delta^7}{7!} + \cdots$$

Here, δ^3 is a higher-order infinitesimal (still non-zero, but $\delta^3 \ll \delta$ in the sense that $\frac{\delta^3}{\delta} = \delta^2 \approx 0$). Similarly, δ^5, δ^7, etc., are even smaller infinitesimals. Thus, $\sin \delta \approx \delta - \frac{\delta^3}{6}$, where the approximation $\approx$ means the difference (from higher terms) is infinitesimal relative to the leading

terms. For the basic linear approximation $\sin\delta \approx \delta$, and the error term $-\frac{\delta^3}{6} + \cdots$ is infinitesimal, so $st(\sin(\delta) - \delta) = 0$ (standard part is zero), confirming the approximation holds in the hyperreal sense. However, since δ is invertible ($\frac{1}{\delta}$ is infinite), we can analyze different infinitesimal scales (e.g., δ^3 is distinctly smaller than δ but usable in further computations). This flexibility is useful in non-standard analysis for precise error bounds in limits or series convergence.

Dual Approach: Now consider $\sin\varepsilon$, where ε is the nilpotent element of the dual numbers. Extending the sine function to dual numbers via its MacLaurin series (which works because dual numbers support polynomial extensions, and sine is analytic), we have

$$\sin\varepsilon = \varepsilon - \frac{\varepsilon^3}{3!} + \frac{\varepsilon^5}{5!} - \frac{\varepsilon^7}{7!} + \cdots$$

But since $\varepsilon^2 = 0$, all higher powers vanish exactly. Thus, the series truncates automatically to the first-order term, and so, $\sin\varepsilon = \varepsilon + 0 = \varepsilon$. This gives the exact linear approximation $\sin\varepsilon = \varepsilon$, with no higher terms surviving due to nilpotency.

For a more general case, we evaluate $\sin(a + \varepsilon)$ for a real number a. Using the sum of angles formula for the sine function, we have

$$\sin(a+\varepsilon) = \sin(a)\cos(\varepsilon) + \cos(a)\sin(\varepsilon)$$

Using the Maclaurin series for the cosine and sine, we have that $\cos\varepsilon = 1 - \frac{\varepsilon^2}{2!} + \frac{\varepsilon^4}{4!} \pm \cdots = 1$, and $\sin\varepsilon = \varepsilon$, and so,

$$\sin(a+\varepsilon) = \sin(a)\cdot 1 + \cos(a)\cdot\varepsilon = \sin(a) + \cos(a)\,\varepsilon.$$

This directly encodes the derivative, i.e., the dual part $\cos(a)$ is the derivative of $\sin(a)$. The truncation ensures exact first-order accuracy without residual terms, making duals efficient for computational differentiation (as in forward-mode automatic differentiation – to be discussed later in this document), but it limits analysis of higher-order errors – unlike hyperreals, where such terms persist as smaller infinitesimals.

8 Surreal Numbers

Do not worry about your difficulties in mathematics; I can assure you that mine are still greater.

— Albert Einstein

This section is here for completeness relative to the real numbers, dual numbers and hyperreal numbers. Surreal numbers are not used in the subsequent sections of this book.

8.1 What are Surreal Numbers? An Intuitive Overview

The **surreal number** system, discovered by John Horton Conway and popularized by Donald Knuth in his book *Surreal Numbers*, is a remarkable class of numbers that is both incredibly simple in its construction and immensely powerful in its scope.

The core idea is that every number is defined by two sets of numbers that precede it**:** a "Left set" and a "Right set." A surreal number is written in the form $x = \{L \mid R\}$, where every element in L is less than every element in R.

The most astonishing property of the surreals is that they form a totally ordered field that is:

- Vast**:** It contains not only all real numbers but also all ordinal numbers[7] and an immense variety of new numbers like infinitesimals and infinite numbers.
- Universal**:** It is, in a specific sense, the "largest possible" ordered field.

Think of it this way:

- Dedekind cuts are like a master craftsman using a specific tool to solve one problem perfectly: building the perfect, continuous, one-dimensional number line from the rationals.
- The surreal number construction is like discovering a new universe of numbers. It's a generative process that starts from nothing and recursively builds a structure so rich that the real numbers, along with infinite and infinitesimal numbers, all appear naturally within it as special cases.

Dedekind cuts are a targeted technique for defining a specific system ($\mathbb{R}$), while the surreal number process is a grand, generative framework that reveals $\mathbb{R}$ to be just one interesting neighborhood within a much larger numerical universe.

8.2 Formal Definition and Construction

The construction is recursive, meaning we define numbers in terms of numbers we have already created.

Axiomatic Definition

A surreal number is an ordered pair $x = (L, R)$, written as $\{L \mid R\}$, where L and R are sets of surreal numbers, and no element of L is greater than or equal to any element of R.

This condition, $\forall l \in L, \forall r \in R: l < r$, is the fundamental law of surreal numbers.

[7] Ordinal numbers are numbers that indicate the position or rank of an object in a sequence, such as first, second, third, and so on. They differ from cardinal numbers, which represent quantity, and are often written with suffixes like 1st, 2nd, 3rd, etc.

The Recursive Construction (Birthdays)

The surreal numbers are generated in a day-by-day process.

- **Day 0**: We start with the number 0. We define it using the empty set: $0 = \{ \mid \}$. There is nothing on the left and nothing on the right, so the condition is trivially true.

Now we have 0. We can create new numbers using the sets we have available, i.e., $\{0\}$ and the empty set $\{ \}$.

- **Day 1**: We can form four possible pairs:
    1. $\{ \mid \} = 0$ (This already exists)
    2. $\{0 \mid \}$ We call this number 1.
    3. $\{ \mid 0\}$ We call this number -1.
    4. $\{0 \mid 0\}$ This is *not* a number because it violates the fundamental law, i.e., 0 is not less than 0.

So, on Day 1, two new surreal numbers are "born", i.e., $-1 = \{ \mid 0\}$ and $1 = \{0 \mid \}$.

- **Day 2**: We now have the numbers: $-1, 0, 1$. We can use any of their subsets that satisfy the fundamental law.
    - $\{1 \mid \} \rightarrow$ This is defined to be the number 2.
    - $\{0 \mid \} = 1$ (already exists)
    - $\{0, 1 \mid \} \rightarrow$ This is also 2, as we will see with the Simplicity Theorem.
    - $\{-1 \mid \} \rightarrow$ This is 0, because $\{ -1 \mid \}$ is between -1 and 1, and the simplest such number is 0.
    - $\{ \mid -1\} \rightarrow$ This is defined to be -2.
    - $\{ -1 \mid 0\} \rightarrow$ This is defined to be $-\frac{1}{2}$.
    - $\{ 0 \mid 1\} \rightarrow$ This is defined to be $\frac{1}{2}$.
    - $\{ -1 \mid 1\} \rightarrow$ This is yet another representation of 0.

And so on. This process continues transfinitely. By Day ω (the first infinite ordinal), we can create numbers like $\{0, 1, 2, 3, \ldots \mid \}$ which is the infinite ordinal ω, and $\{ \mid 0, -1, -2, -3, \ldots \}$ which is $-\omega$.

8.3 Order and Equality

How do we compare two surreal numbers?

Let $x = \{X_L \mid X_R\}$ and $y = \{Y_L \mid Y_R\}$.

[**Author's Remark**: The following definition of ordering is wrong but it illustrates a common confusion with surreal numbers. We will follow this with a correct definition.]

Definition of Order $(x \geq y)$:

We say $x \geq y$ if and only if the following conditions hold true:

1. There is no $x_L \in X_L$ such that $y \leq x_L$, AND
2. There is no $y_R \in Y_R$ such that $y_R \leq x$.

This is also a recursive definition. From this, we define:

- $x \leq y$ if and only if $y \geq x$.
- $x = y$ if and only if $x \geq y$ and $y \geq x$.
- $x > y$ if and only if $x \geq y$ and $y \ngeq x$.

Example of issue with the above definition: Is $1 \geq 0$?

Recall $1 = \{0 \mid \}$ and $0 = \{ \mid \}$.

First, we establish that $0 \geq 0$ by our definition of order above.

Check $0 \geq 0$:

- $X_L = \emptyset$, so "no x_L such that $0 \leq x_L$" is vacuously true.
- $Y_R = \emptyset$, so "no y_R such that $y_R \leq 0$" is vacuously true.

So, $0 \geq 0$ holds true. Good.

Check $1 \geq 0$:

- We have that $X_L = \{0\}$. Is there $x_L = 0$ such that $0 \leq x_L$? Yes, we just showed that $0 \leq 0$ is true. So, there exists $x_L \in X_L$ with $y \leq x_L$ (since $y = 0$, $x_L = 0$, and $0 \leq 0$).

So, condition 1 fails: there **is** an x_L (namely 0) such that $0 \leq x_L$.

Thus, $1 \geq 0$ is **false** by this definition.

Check $0 \geq 1$:

- $X_L = \emptyset$, and so, condition 1 vacuously true.
- $Y_R = \emptyset$, and so, condition 2 vacuously true.

Thus, $0 \geq 1$ is **true** by this definition.

So, we have that $0 \geq 1$ is true and $1 \geq 0$ is false, so $0 > 1$ which absurd and not what we want.

The resolution is that we must first define the relation $\leq$ independently. A more standard approach is as follows:

Correct definition of less than or equal to:

$x \leq y$ if and only if (1) there exists no $x_L \geq y$ AND (2) there exists no $y_R \leq x$).

Let's use this definition. We want to know if $0 \leq 1$?

$0 = \{ \mid \}, 1 = \{0 \mid \}$.

- Condition 1: Is there an $x_L \in \{\ \}$ (empty set) such that $x_L \geq 1$? No, the empty set has no elements.
- Condition 2: Is there a $y_R \in \{\ \}$ such that $y_R \leq 0$? No.

Both conditions hold, and so, $0 \leq 1$ is true.

Now check $1 \leq 0$?

$1 = \{0 \mid \}, 0 = \{ \mid \}$.

- Condition 1: Is there an $x_L \in \{0\}$ such that $x_L \geq 0$? Yes, $0 \geq 0$ is true. Condition fails. So $1 \not\leq 0$.

Thus, $0 \leq 1$ and $1 \not\leq 0$, so $0 < 1$, which is the desired result.

8.4 Fundamental Theorems

The Simplicity Theorem (or Genetic Theorem)

This is the most important theorem for determining the "value" of a surreal number.

If x is a surreal number of the form $x = \{L \mid R\}$, and some surreal s satisfies $L < s < R$ (i.e., $s >$ every element of L, and $s <$ every element of R), and if s is the simplest such surreal (born earliest), then $x = s$.

Examples:

- $\{0, 1 \mid \}$: The numbers between $\{0, 1\}$ and nothing. The simplest number greater than 0 and 1 is 2 (born on Day 2). So $\{0, 1 \mid \} = 2$.
- $\{0 \mid 1\}$: The numbers between 0 and 1. The simplest such number is $\frac{1}{2}$ (born on Day 2). So, $\{0 \mid 1\} = \frac{1}{2}$.
- $\{-1 \mid 1\}$: The numbers between -1 and 1. The simplest such number is 0 (born on Day 0). So $\{-1 \mid 1\} = 0$.

Arithmetic Operations

Arithmetic on surreal numbers is also defined recursively.

Let $x = \{X_L \mid X_R\}$ and $y = \{Y_L \mid Y_R\}$.

Negation: $-x = \{-X_R \mid -X_L\}$ (You swap the left and right sets and negate every element).

Addition: $x + y = \{X_L + y, x + Y_L \mid X_R + y, x + Y_R\}$

(Note: $X_L + y$ means the set of all sums of an element from X_L with y).

Multiplication: This is more complex. We first make the following assignments:

$$A = \{x_L y + x y_L - x_L y_L\}_{x_L \in X_L, y_L \in Y_L}$$

$$B = \{x_R y + x y_R - x_R y_R\}_{x_R \in X_R, y_R \in Y_R}$$

$$C = \{x_L y + x y_R - x_L y_R\}_{x_L \in X_L, y_R \in Y_R}$$

$$D = \{x_R y + x y_L - x_R y_L\}_{x_R \in X_R, y_L \in Y_L}$$

Multiplication of surreal numbers is defined as follows:

$$x * y = \{ A \cup B \mid C \cup D \}$$

The various juxtapositions of variables in the definitions of A, B, C, D are in fact surreal multiplications, e.g., $x_L y$ is shorthand for $x_L * y$. Thus, multiplication is defined recursively for surreal numbers.

Equivalently,

$$x * y = \{ X_L y + xY_L - X_L Y_L, X_R y + xY_R - X_R Y_R \mid X_L y + xY_R - X_L Y_R, X_R y + xY_L - X_R Y_L \}$$

where expressions like $X_L y$ mean the set of all products $x_L y$ for $x_L \in X_L$.

For example, one can prove that $\{0 \mid 1\} * \{0 \mid 1\} = \{0 \mid \frac{1}{2}\}$, confirming that $\left(\frac{1}{2}\right) * \left(\frac{1}{2}\right) = \frac{1}{4}$. We will use the first definition of surreal number multiplication for this example.

Step 1: Identify the components

For $x = \{0 \mid 1\}$, we have:

- $X_L = \{0\}$ (left set of x) and $X_R = \{1\}$ (right set of x)
- Since we're computing $x \times x$, we also have $y = x$, and so, $Y_L = \{0\}$ (left set of y) and $Y_R = \{1\}$ (right set of y)

Step 2: Compute the left set of $x \times x$

We need to compute $A \cup B$.

For A (with $x_L = 0, y_L = 0$), we have $0 \cdot x + x \cdot 0 - 0 \cdot 0 = 0 + 0 - 0 = 0$

For B (with $x_R = 1, y_R = 1$), we have $1 \cdot x + x \cdot 1 - 1 \cdot 1 = x + x - 1 = 2x - 1$

So, the left set is $\{0, 2x - 1\}$.

Step 3: Compute the right set of $x \times x$

We need to compute $C \cup D$:

For C (with $x_L = 0, y_R = 1$), we have $0 \cdot x + x \cdot 1 - 0 \cdot 1 = 0 + x - 0 = x$

For D (with $x_R = 1, y_L = 0$), we have $1 \cdot x + x \cdot 0 - 1 \cdot 0 = x + 0 - 0 = x$

So, the right set is $\{x, x\} = \{x\}$ (since sets don't contain duplicates).

Step 4: Putting Steps 2 and 3 together

We have that $x \times x = \{0, 2x - 1 \mid x\}$.

Step 5: Simplify using what we already know

We need to determine what number $\{0, 2x - 1 \mid x\}$ represents.

First, note that by the Simplicity Theorem, if we can show that $x \times x$ lies between 0 and x, and is the simplest such number, then $x \times x = \{0 \mid x\}$.

We know $x = \{0 \mid 1\}$ is positive (since $0 < x$). Also, $x < 1$ because $1 = \{0 \mid \}$ and by the order definition, no element of x's right set (which contains 1) can be ≤ 0. Therefore, $2x - 1 < x$ (since $2x - 1 < x$ simplifies to $x < 1$).Thus, the left set elements are both less than x.

The right set only has x, which is greater than both 0 and $2x - 1$.

Therefore, $2x - 1 < x$ (since substituting $x = \frac{1}{2}$ gives $0 < \frac{1}{2}$).

Step 6: Apply the Simplicity Theorem

The simplest number between 0 and x is $(0|x\}$. We need to check if our calculated product, $\{0, 2x - 1 \mid x\}$, is equal to this target.

First, observe that the Right Sets are already identical: both contain the single element x. Therefore, we only need to verify that the Left Sets are equivalent.

The calculated Left Set is $\{0, 2x - 1 \mid x\}$. We can evaluate the term $2x - 1$ directly:

- We established previously that $x = \frac{1}{2}$.
- Therefore, the expression $2x - 1$ evaluates to $2\left(\frac{1}{2}\right) - 1 = 0$.

This means the calculated Left Set is numerically equivalent to $\{0,0\}$. Since sets do not contain duplicate elements, this simplifies to $\{0\}$.

Since both the Left and Right sets match the target form, we conclude that $x \times x = \{0|x\}$.

Step 7: Evaluate $\{\mathbf{0} \mid \boldsymbol{x}\}$ ***when*** $\boldsymbol{x = \frac{1}{2}}$

We know from earlier in the construction that $\{0|1\} = \frac{1}{2}$. Therefore, $x \times x = \{0 \mid \frac{1}{2}\}$. From the dyadic rational construction (Day 3 creates numbers like $\frac{1}{4}, \frac{3}{4}$, etc.), we established that $\{0 \mid \frac{1}{2}\}$ is the number $\frac{1}{4}$. Thus, $\frac{1}{2} \times \frac{1}{2} = \frac{1}{4}$.

Wasn't that easy?

8.5 The Real Numbers as a Subset of the Surreals

The real numbers are not just like the surreals; they are a genuine, canonical subset. Every real number has a unique surreal representation. The process of building the surreals day-by-day naturally constructs all real numbers.

How are real numbers generated?

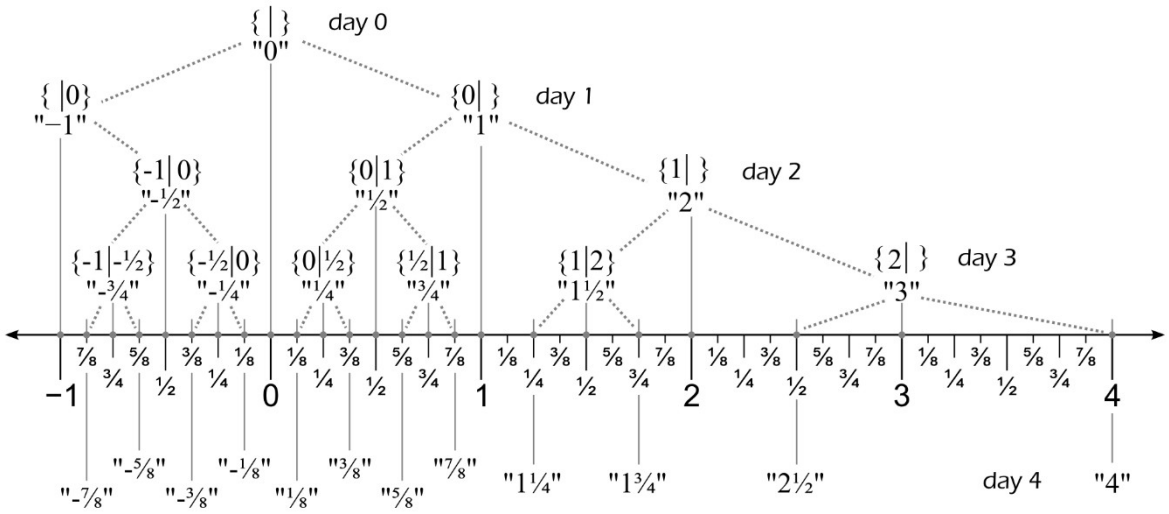
The process mirrors the Dedekind cut construction of the reals, but with a crucial "simplicity" constraint.

Dyadic Rationals, i.e., fractions with denominator a power of 2: These are born on finite days.

- Day 0: $0 = \{ \mid \}$
- Day 1: $1 = \{0 \mid \}$, $-1 = \{ \mid 0\}$
- Day 2: $2 = \{1 \mid \}$, $\frac{1}{2} = \{0 \mid 1\}$, $-2 = \{ \mid -1\}$, $-\frac{1}{2} = \{-1 \mid 0\}$
- Day 3: $3 = \{2 \mid \}$, $\frac{3}{2} = \{1 \mid 2\}$, $\frac{3}{4} = \{\frac{1}{2} \mid 1\}$, $\frac{1}{4} = \{0 \mid \frac{1}{2}\}$, and their negatives.
- On day n, all dyadic rationals of the form $\frac{m}{2^k}$ (where m is an integer and $\frac{m}{2^k}$ is in lowest terms) with "birthday" n are created.

The figure below depicts the generation of surreal numbers between -1 and 4 up to Day 4.

Source: *Surreal number/The dyadics*, Wikiversity, https://en.wikiversity.org/wiki/Surreal_number/The_dyadics



All Other Real Numbers (Irrationals and non-dyadic rationals): These are born on day ω (the first infinite ordinal). They are defined as the simplest number lying between two sets of previously created numbers (the dyadic rationals).

Example: $\frac{1}{3}$ There is no finite-day surreal that equals $\frac{1}{3}$. On day ω, we can define it as the simplest number between all smaller and all larger dyadic rationals:

$$\frac{1}{3} = \left\{\frac{1}{4}, \frac{5}{16}, \frac{21}{64}, \ldots \mid \frac{1}{2}, \frac{3}{8}, \frac{11}{32}, \ldots\right\}$$

The sequence on the left has general form shown below where $\lim_{n\to\infty} S_n = \frac{1}{3}$

$$S_n = \frac{1 + 4 + 4^2 + \cdots + 4^{n-1}}{4^n}$$

The sequence on the right has general form shown below where $\lim_{n\to\infty} T_n = \frac{1}{3}$

$$T_n = \frac{1 + 2 + 2^3 + 2^5 + \cdots + 2^{2n-3}}{2^{2n-1}}$$

The left set contains all dyadic rationals $< \frac{1}{3}$, and the right set contains all dyadic rationals $> \frac{1}{3}$. The Simplicity Theorem dictates that the number born in this gap is $\frac{1}{3}$.

Example: π The canonical representation of π uses **all** dyadic rationals less than $\pi$for L and **all** dyadic rationals greater than π for R, but listing them all is impossible, so we just indicate the pattern, as shown below

$$\pi = \{\, 3, 3.125, 3.140625, 3.1416015625, \cdots \mid 4, 3.25, 3.15625, 3.142578125, \ldots \,\}$$

In this way, the set of all real numbers corresponds precisely to the set of all surreal numbers born on or before day ω. The real numbers are the "foundation" of the surreal universe upon which the infinite and infinitesimal numbers are built.

8.6 Relation to Hyperreal and Dual Numbers

All three number systems—surreal, hyperreal, and dual—extend the real numbers to include infinitesimals, but they do so with different goals and structures.

[**Author's Remark**: Apparently, the de facto notation for the field of hyperreal is **No**. I find this notation confusing for obvious reasons but will stick with the de facto standard (using bold to hopefully avoid any confusion).]

Table 8. Comparison of dual, hyperreal and surreal numbers

Feature	Surreal Numbers (No)	Hyperreal Numbers	Dual Numbers
Purpose/ Motivation	Foundational, Universal. To create the "largest" ordered field.	Analytical. To provide a rigorous foundation for Calculus in the form of Non-Standard Analysis.	Algebraic. To automate differentiation and study nilpotent elements.
Structure	A proper class, not a set. Immensely rich, containing all ordinals.	A set (though a very large one). Multiple non-isomorphic models exist.	A commutative ring. A 2-dimensional vector space over $\mathbb{R}$.
Infinitesimals	A vast, ordered hierarchy ($\varepsilon, \frac{\varepsilon}{2}, \varepsilon^2, \sqrt{\varepsilon}$, etc.).	A hierarchy of infinitesimals, but the structure depends on the chosen ultrafilter.	A single infinitesimal ε with the defining property $\varepsilon^2 = 0$.
Relation	The Surreal Universe contains them all, in spirit.	Every hyperreal field is isomorphic to a subfield of **No**.	The dual numbers can be embedded into **No**, but they are a tiny, specialized part.

Detailed Comparison:

- **Surreals vs. Hyperreals:**
    - The hyperreal numbers are designed to be an elementary extension of the reals. This means any statement that can be written in first-order logic (the language of calculus with limits) is true in the reals if and only if it is true in the hyperreals. This is the core of Non-Standard Analysis.
    - The surreal numbers are far richer. They contain objects like $\omega, \sqrt{\omega}$, and ω^{ω} which have no analogue in a standard hyperreal system designed for analysis. However, a profound theorem states that every maximal (i.e., non-isomorphic to a smaller) hyperreal field is isomorphic to a subfield of the surreal numbers. Think of the hyperreals as a "continent" of numbers suitable for analysis, while the surreals are the entire "planet."

- **Surreals vs. Dual Numbers:**
    - The dual numbers are a much simpler and more limited structure. They are the quotient ring $\mathbb{R}[\varepsilon]/\varepsilon^2$. This means the infinitesimal ε is nilpotent—its square is exactly zero. This property makes them perfect for automatic differentiation (e.g., calculating derivatives in machine learning).
    - In the surreals, no non-zero infinitesimal is nilpotent. If ε is a surreal infinitesimal, then ε^2 is a smaller, non-zero infinitesimal.
    - Can we find the dual numbers inside the surreals? Yes, but it's an "unnatural" embedding. You can't just take a surreal ε and have $\varepsilon^2 = 0$. However, you can define a map that sends a dual number $a + b\varepsilon$ to the surreal pair $\{a \mid a + b\omega\}$, leveraging the infinitely large number ω and its reciprocal $\varepsilon = \frac{1}{\omega}$. While ε^2 isn't zero, it behaves like zero for many algebraic purposes within this specific, identified copy. Thus, the dual number system can be seen as a specific, simplified algebraic model that captures a slice of the surreal structure.

Conclusion

The surreal numbers provide a grand, unified framework:

1. They contain the real numbers as their "finite" and "first infinite" layer (days 0 to ω).
2. They encompass the spirit of the hyperreal numbers, with every hyperreal field finding a home as a subfield of **No**.
3. They vastly exceed the structure of the dual numbers, but contain algebraic substructures that can be mapped to them for specific computational purposes.

In essence, the surreal number field **No** is the "room of all rooms" for ordered number systems, with the real numbers as its foundation and the hyperreals as a major continent within its expansive geography.

8.7 Key Properties and Remarkable Features

1. The Real Numbers are Contained Within: All real numbers appear within the surreals. Integers, rationals, and irrationals like π and e (Euler's number) all have surreal definitions.
2. Infinitesimals: These are numbers greater than 0 but smaller than any positive real number. The simplest infinitesimal is $\varepsilon = \{0 \mid 1, \frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \ldots\}$. You can also define $\varepsilon, 2\varepsilon, \frac{\varepsilon}{2}, \varepsilon^2$, etc.
3. Infinite Numbers: The surreals contain infinitely large numbers. For example, $\omega = \{0, 1, 2, 3, \ldots \mid \}$. You can also define $\omega - 1, \frac{\omega}{2}, \sqrt{\omega}$, and even ω^ω .
4. Totally Ordered Field: The surreal numbers form a totally ordered field, meaning the standard rules of arithmetic (associativity, commutativity, distributivity) and order all hold.
5. Universal Property: The Field of surreal numbers, No, is unique (up to isomorphism) in that it is the largest ordered field. Every other ordered field can be embedded into No.

8.8 Summary and Conclusion

The surreal numbers provide a grand, unified framework for ordered number systems:

- **Construction and Theorems:** Built recursively as $\{L \mid R\}$, their order and arithmetic are defined recursively, governed by the fundamental Simplicity Theorem.
- **Vastness:** They form a totally ordered field containing:
    1. All real numbers as the numbers born by day ω.
    2. Infinitesimals like $\varepsilon = \{0 \mid 1, \frac{1}{2}, \frac{1}{4}, \ldots\}$.
    3. Infinite numbers like $\omega = \{0, 1, 2, \ldots \mid \}$ and beyond.
- **Universality:** The surreal field **No** is the "room of all rooms." It naturally contains the real numbers as its foundation, encompasses the spirit and structure of the hyperreal numbers, and has room for algebraic constructs such as the dual numbers.

In essence, the surreal number system is the most comprehensive and inclusive number system, revealing the real numbers to be just the first layer of a much richer and more expansive mathematical universe.

9 Weil Algebras: The Algebra of Infinitesimals

Like a circle, where beginning and end are one. — Buddhist saying

9.1 Overview

We will need the concept of a Weil algebra in Section 11.5.2. We introduce the concept here (as opposed to the earlier section on algebras) since the concept relies on an understanding of infinitesimals.

Weil algebras are the algebraic engines that power a modern approach to differential geometry. They allow us to define derivatives and tangents not as limits, but through simple algebraic multiplication, providing a rigorous foundation for infinitesimals.

A **Weil algebra** is a finite-dimensional, commutative, associative algebra over the real numbers that has a unique maximal ideal consisting of nilpotent elements (the infinitesimals). This means every element can be uniquely written as a real number (the point) plus nilpotent components (the infinitesimal or derivative data). Think of it as a number system designed to package a value along with its derivatives in a very compact, algebraic way.

In what follows, there is some repetition of the dual number concept that we discussed previously, but the context is different.

9.2 Step-by-Step Analogy

Let's start with a familiar algebra: the real numbers $\mathbb{R}$.

- It's 1-dimensional. Every number is just $a \in \mathbb{R}$.
- It represents a point on a line.

Now, what if we wanted an algebra that represents not just a point, but also the slope (the first derivative) at that point? We need an extra component for that derivative information. This is the simplest Weil algebra, often called the algebra of dual numbers.

9.2.1 Example 1: The Dual Numbers (The simplest Weil algebra)

- **What it is:** Numbers of the form $a + b\varepsilon$
- **The Rule:** The new element ε is so tiny that its square is zero: $\varepsilon^2 = 0$
- **Dimension:** 2-dimensional (spanned by $1 \in \mathbb{R}$ and ε).

How it works:

Think of a as the value of a function at a point, and b as its derivative. Multiplying to such numbers, we have $(a + b\varepsilon) * (c + d\varepsilon) = ac + (ad + bc)\varepsilon + bd\varepsilon^2$.
Since $\varepsilon^2 = 0$, this simplifies to $ac + (ad + bc)\varepsilon$.

The new value ac is just the product of the original values, and the new derivative part $ad + bc$ is exactly the product rule from calculus! So, if $a + b\varepsilon$ represents a function f and $c + d\varepsilon$ represents a function g, then their product $ac + (ad + bc)\varepsilon$ correctly represents the value and derivative of the product function $f \cdot g$ at a point, all computed algebraically.

9.2.2 Example 2: A Slightly More Complex Weil Algebra

Consider the Weil algebra $\mathcal{W} = \mathbb{R}[\varepsilon_1, \varepsilon_2]/(\varepsilon_1^2, \varepsilon_2^2)$. A general element of $\mathcal{W}$ has the form $a + b\varepsilon_1 + c\varepsilon_2 + d\varepsilon_1\varepsilon_2$ where $a, b, c, d \in \mathbb{R}$. The basis for $\mathcal{W}$ is $\{1, \varepsilon_1, \varepsilon_2, \varepsilon_1\varepsilon_2\}$ and so, $\mathcal{W}$ is 4-dimensional.

Interpretation as Truncated Taylor Series:

For any smooth function $f(x, y)$, its behaviour near $(0,0)$ up to first order in each variable separately is captured by the element

$$\tau(f) = f(0,0) + f_x(0,0)\varepsilon_1 + f_y(0,0)\varepsilon_2 + f_{xy}(0,0)\varepsilon_1\varepsilon_2 \in W$$

Notice how the rules $\varepsilon_1^2 = 0$ and $\varepsilon_2^2 = 0$ correctly truncate the Taylor series.

Algebraic Computation of Derivatives:

The power of $\mathcal{W}$ is that it computes derivatives for products algebraically. As an illustration, consider two elements of $\mathcal{W}$ that happen to have their d coefficient equal to zero:

$$f \leftrightarrow a + b\varepsilon_1 + c\varepsilon_2 \text{ since by assumption } f_{xy}(0,0) = 0$$

$$g \leftrightarrow p + q\varepsilon_1 + r\varepsilon_2 \text{ since by assumption } g_{xy}(0,0) = 0$$

Their product in $\mathcal{W}$ is:

$$(a + b\varepsilon_1 + c\varepsilon_2)(p + q\varepsilon_1 + r\varepsilon_2) = ap + (aq + bp)\varepsilon_1 + (ar + cp)\varepsilon_2 + (br + cq)\varepsilon_1\varepsilon_2$$

Observe that the result now has a non-zero $\varepsilon_1\varepsilon_2$ term! This term is not an error; it encodes the mixed partial derivative of the product function $h = f \cdot g$. Let's verify:

- $h(0,0) = f(0,0) \cdot g(0,0) = ap$ (the real part).
- $h_x = f_x\, g + f\, g_x$. At $(0,0)$, this is $bp + aq$, matching the coefficient of ε_1.
- $h_y = f_y\, g + f\, g_y$. At $(0,0)$, this is $cp + ar$, matching the coefficient of ε_2.
- $h_{xy} = \frac{\partial}{\partial y} h_x = f_{xy}g + f_x\, g_y + f_y\, g_x + f\, g_{xy}$. Since we assumed $f_{xy} = 0$ and $g_{xy} = 0$ for our chosen f and g, this simplifies to $br + cq$ which is exactly the coefficient of $\varepsilon_1\varepsilon_2$ generated by the multiplication.

Thus, the Weil algebra $\mathcal{W}$ automatically applies the product rule for both first and mixed partial derivatives through simple multiplication.

9.3 The Formal Definition in Simple Terms

For an algebra to be a Weil algebra, it must be:

1. **Finite-Dimensional:** As a vector space, it has a finite basis. (Our examples were 2D and 4D). This means the amount of information it can hold is limited.
2. **Commutative and Associative:** Multiplication works in the usual, predictable way, i.e., $a * b = b * a$ and $(a * b) * c = a * (b * c)$.
3. **Local:** This is the key technical part. It means the algebra has a unique maximal ideal I. This ideal is the set of all purely infinitesimal elements (those without a real number part).
    a. For dual numbers, I is all numbers $b\varepsilon$, i.e., $I = (\varepsilon)$

   b. In the 4D algebra example , I is all elements $b\varepsilon^1 + c\varepsilon^2 + d\varepsilon^1\varepsilon^2$, i.e., $I = (\varepsilon_1, \varepsilon_2)$.

4. **Nilpotent Ideal:** The infinitesimal elements are so insignificant that if you multiply enough of them, you get zero. Formally, there exists an integer n such that any product of n elements from the ideal I is zero.
   a. For dual numbers ($n = 2$), $(b\varepsilon)(d\varepsilon) = 0$.
   b. In the 4D algebra example ($n = 4$), any product of the three basis elements is zero.

The most important consequence of these properties is that every element in a Weil algebra can be uniquely written in the form

$$\text{real number} + \text{nilpotent component}$$

9.4 What is it Used For?

Weil algebras are the fundamental building blocks in a powerful area of geometry called Synthetic Differential Geometry (SDG), see Section 11.

- In SDG, we imagine a world where every curve has a “tiny, straight” part. This “tiny part” is modeled by the spectrum of a Weil algebra. (The spectrum of a Weil algebra is a single point, whose surrounding infinitesimal structure is encoded by the algebra itself.)
- They provide a purely algebraic and rigorous way to talk about derivatives, tangent vectors, and jets (higher-order derivatives) without using the limits of classical calculus.
- A Weil functor on a manifold is a generalization of the tangent bundle functor. (Functors are discussed in Section 10.4). The tangent bundle uses the dual numbers $a + b\varepsilon$. A more complex Weil algebra gives you a bundle of higher-order jet information as we saw in Example 2.

9.5 Summary in a Nutshell

Table 9. Summary of Weil algebra

Concept	**Simple Analogy**
Algebra	The infinitesimal elements are so insignificant that if you multiply enough of them, you get zero. Formally, there exists an integer n such that any product of n elements from the ideal I is zero.
Weil Algebra	A small, finite-dimensional algebra where every element is a real number (the “value”) plus “infinitesimal” bits (the “derivative information”).
Purpose	To package a value and its derivatives into a single algebraic object for doing differential geometry with rigorous infinitesimals.

In the simplest terms: A Weil algebra is a number system with extra, “tiny” dimensions that are used to store derivative information. The simplest example is the dual numbers $a + b\varepsilon$, where $\varepsilon^2 = 0$, as we saw in Example 1.

10 Basics of Category Theory

Category theory is the mathematics of mathematics. — Anonymous

10.1 Introduction

Category theory, developed by Samuel Eilenberg and Saunders Mac Lane in the 1940s, is a branch of mathematics that studies mathematical structures and their relationships in an abstract way. Often described as "general abstract nonsense" in jest, it provides a unifying language for concepts across algebra, topology, logic, and more. In the context of this book, category theory serves as a bridge to synthetic differential geometry (discussed in the following section), which formalizes infinitesimals in a categorical framework, while connecting to hyperreal numbers, dual numbers, and automatic differentiation.

This section starts from the very basics, assuming no prior knowledge. We'll define categories, their building blocks (objects and morphisms), and key concepts such as functors and natural transformations. We use simple examples to illustrate ideas, while building toward Cartesian Closed Categories (CCCs), which are essential for modeling smooth structures in Synthetic Differential Geometry (SDG)[8]. By the end, you'll see how category theory abstracts the infinitesimal approaches discussed earlier.

10.2 What is a Category?

At its core, a category is a collection of "things" (objects) and "ways to go between them" (morphisms or arrows), with rules for combining these ways.

Definition: A **category** $\mathcal{C}$ consists of the following:

- A collection of **objects** denoted by $Ob(\mathcal{C})$. Objects can be anything – sets, numbers, vector spaces. The focus is on the relationships among the objects and not the objects.
- For every pair of objects $A, B \in Ob(\mathcal{C})$, there is a collection of **morphisms** (or arrows) from A to B, denoted $\hom(A, B)$ or $\mathcal{C}(A, B)$. A morphism $f: A \to B$ represents a structure-preserving arrow from A to B.
- **Composition operation**: If $f: A \to B$ and $g: B \to C$, then there exists a composite morphism $g \circ f: A \to C$.
- For each object A, an **identity morphism** exists, denoted $id_A: A \to A$ (or 1_A).

The following two axioms must hold true for a category:

- Associativity with respect to composition: For morphisms, given $f: A \to B$, $g: B \to C$, $h: C \to D$, then $(h \circ g) \circ f = h \circ (g \circ f)$.
- Identity laws: For any $f: A \to B$, $f \circ id_A = f = id_B \circ f$.

[8] Imagine a version of geometry where the concept of an "infinitely small line segment" is a fundamental, well-defined object. SDG builds a consistent mathematical universe around this idea, providing a powerful and intuitive language for differential geometry where the derivative becomes a simple algebraic operation.

Think of categories as graphs where nodes are objects, edges are morphisms, and composition is path concatenation, with identities as "do nothing" loops.

Example 1: The Category of Sets (Set)

- Objects: All sets, e.g., $\mathbb{R}$, $\{1,2\}$, letters of the alphabet, street names in Paris.
- Morphisms: Functions between sets, e.g., $f: \mathbb{R} \to \mathbb{R}$, $f(x) = x^2$.
- Composition: Standard function composition, e.g., if $g(x) = \sin(x)$ and $f(x) = x^3$, then $g \circ f(x) = \sin(x^3)$.
- Identity: The identity function $id(x) = x$.

The Set category models basic structures without additional constraints.

Example 2: The Category of Groups (Grp)

- Objects: Groups (e.g., the integers $\mathbb{Z}$ under addition).
- Morphisms: Group homomorphisms (functions preserving the group operation, e.g., $f: \mathbb{Z} \to \mathbb{Z}$, $f(n) = 2n$).
- Composition and identities: As in the Set category, but restricted to preserve group structure.

Here, morphisms respect the "group-ness," showing how categories capture specific mathematical domains.

Example 3: The Category of Vector Spaces (Vect)

- **Objects**: Vector spaces over a fixed field K (e.g., $\mathbb{R}$ or $\mathbb{C}$).
- **Morphisms**: Linear transformations (or linear maps) between vector spaces.
- **Composition**: Standard function composition of linear maps, which is associative and preserves linearity.

Example 4: A Small Category (Preorder)

- Consider numbers $1, 2, 3$ as objects, with morphisms $1 \to 2$, $2 \to 3$, $1 \to 3$ (and the identity morphisms, e.g., $1 \to 1$).
- Composition: $(1 \to 3) = (2 \to 3) \circ (1 \to 2)$. This is like a partial order (the $\leq$ relation), illustrating categories as generalized posets (i.e., partially ordered sets).

10.3 Morphisms

Morphisms are at the heart of category theory. They describe how objects relate or transform, abstracting the notion of functions between structures. While objects give us the nouns of mathematics (sets, groups, spaces), morphisms give us the verbs, i.e., the ways in which these objects interact or can be mapped to one another.

Every morphism has a domain (its source object) and a codomain (its target object). For a morphism $f: A \to B$, the object A is the domain and B is the codomain. Composition, which is denoted by $g \circ f$, represents performing f first and then g.

10.3.1 Types of Morphisms

Monomorphism (Mono)

A morphism $f: A \to B$ is monic (or a monomorphism) if it is left-cancellative:
$f \circ g = f \circ h \Rightarrow g = h$ for all morphisms $g, h: C \to A$.

In the category Set, this means f is injective (or one-to-one), i.e., it never maps two distinct elements of A to the same element in B.

Intuitively, monos can be thought of as embeddings: they preserve distinctness of information and correspond to inclusions of substructures (subsets in Set, subgroups in Grp). It's crucial to remember that while this intuition holds in many common categories, the formal definition is the cancellation property, which is what makes it a general categorical concept.

Example: Consider the inclusion map $i: \{1,2\} \to \{1,2,3\}$, defined by $i(1) = 1$ and $i(2) = 2$.

Suppose we have two functions $g, h: C \to \{1,2\}$. If the compositions $i \circ g$ and $i \circ h$ are equal, then for every element $c \in C$, $i\big(g(c)\big) = i\big(h(c)\big)$. Since i preserves distinctness (it never merges elements), this means $g(c) = h(c)$ for all $c \in C$, and therefore $g = h$. Hence, i is a monomorphism in Set.

Intuitively, the inclusion map does not lose information—it faithfully embeds one set into another.

Epimorphism (Epi)

A morphism $f: A \to B$ is epic (or an epimorphism) if it is right-cancellative:
$g \circ f = h \circ f \Rightarrow g = h$ for all morphisms $g, h: B \to C$.

In Set, this is equivalent to f being surjective (or onto). Epis can be thought of as quotients or projections that cover their codomain. However, the correspondence with surjectivity is not universal. “Epimorphism” is defined by the cancellation property, and while it matches surjectivity in Set, Grp, and Vect, this is not true in every category. For example, in the category of rings, the inclusion of the integers into the rationals is an epimorphism that is not surjective.

Epis are quotients or projections: they cover their codomain completely.

Example: The natural projection $p: \mathbb{Z} \to \mathbb{Z}/2\mathbb{Z}$ sending $n \to n \bmod 2$ is an epimorphism in the category Grp, since it maps the integers onto the quotient group.

Isomorphism (Iso)

A morphism $f: A \to B$ is an isomorphism if there exists a morphism $g: B \to A$ such that $g \circ f = id_A$ and $f \circ g = id_B$.

In Set, isomorphisms are bijections, i.e., functions that are both one-to-one and onto.

Two objects A and B are isomorphic (written $A \cong B$) if such an isomorphism exists. Isomorphic objects are considered essentially the same within the category; any categorical property true of one holds for the other.

Example: In the category Vect, the vector spaces $\mathbb{R}^2$ and the space of 2×1 real column matrices are isomorphic—each vector corresponds to exactly one column matrix, and vice versa.

10.3.2 Other Notable Morphism Types

While the above are the most common, other morphism types generalize important mathematical concepts:

- Endomorphism: A morphism from an object to itself, $f: A \to A$. Example: A linear operator on a vector space.
- Automorphism: An isomorphism from an object to itself. Example: A rotation of a square is an automorphism in the category of geometric transformations.
- Zero Morphism: In categories with additive structure (like Ab, the category of abelian groups), a morphism that sends every element to the identity element.

10.3.3 Why Morphisms Matter

Morphisms express structure preservation:

- In Set, morphisms preserve membership (functions).
- In Grp, they preserve group operations.
- In Top (topological spaces), continuous maps preserve open set structure.
- In Vect, linear maps preserve addition and scalar multiplication.

By studying morphisms, we study not the objects themselves, but how structures interact. They also enable powerful categorical notions such as composition, identity morphisms, and isomorphisms, which form the backbone of category theory.

10.4 Functors: Maps Between Categories

A **functor** is a single morphism between two categories. The internal structure of a functor consists of:

- A mapping that sends objects in the source category to objects in the target category
- A mapping that sends morphisms in the source category to morphisms in the target category These mappings must preserve identity morphisms and composition.

More formally, we define a functor as follows:

For categories $\mathcal{C}$ and $\mathcal{D}$, the functor $F: \mathcal{C} \to \mathcal{D}$ has the following properties:

- Assigns to each object $A \in \mathcal{C}$, an object $F(A) \in \mathcal{D}$.
- Assigns to each morphism $f: A \to B$ in $\mathcal{C}$, a morphism $F(f): F(A) \to F(B)$ in $\mathcal{D}$.
- Preserves composition: $F(g \circ f) = F(g) \circ F(f)$.
- Preserves identities: $F(id_A) = id_{F(A)}$.

Example 1: Forgetful Functor

A forgetful functor (also called an “underlying functor”) is a type of functor between categories that forgets or discards some of the structure or properties of the objects and morphisms in the source category, mapping them to a simpler target category. It doesn’t add new information; instead, it reveals the bare bones underneath.

- Why "Forgetful"? The name reflects that the functor intentionally omits certain aspects of the source category's structure. For instance, if the source category has objects with operations (like addition or multiplication), the forgetful functor might ignore those operations, treating the objects as mere collections (sets).
- Formal Properties: It's typically a functor $U: \mathcal{C} \to \mathcal{D}$ where $\mathcal{C}$ has more structure than $\mathcal{D}$. It preserves compositions and identities but drops extra axioms or components.
- **Adjointness**[9]: Forgetful functors are often right adjoints to free functors (as described in Example 2 below), which add back the forgotten structure. For example, a free functor might construct a group from a set by adding generators freely. This adjoint pair creates a balance between forgetting and freely generating, which is useful in abstract algebra.

Forgetful functors are ubiquitous because they help reduce problems from structured categories (e.g., groups) to simpler ones (e.g., sets), allowing us to apply tools from the target category.

The Classic Example: From Groups (Grp) to Sets (Set)

- Source Category: Grp (Category of Groups):
  - Objects: Groups, which are sets equipped with a binary operation (e.g., multiplication or addition), an identity element, and inverses for every element. Examples: $(\mathbb{Z}, +)$, i.e., integers under addition, or $GL_n(\mathbb{R})$, i.e., invertible $n \times n$ matrices under multiplication over the real numbers.
  - Morphisms: Group homomorphisms, functions that preserve the group operation, identity, and inverses. For instance, $\phi: \mathbb{Z} \to \mathbb{Z}$, $\phi(n) = 2n$, which doubles integers while preserving addition: $\phi(m + n) = \phi(m) + \phi(n)$.
- Target Category: Set (Category of Sets):
  - Objects: Plain sets, without any additional structure (no required operations).
  - Morphisms: Arbitrary functions between sets (no preservation requirements beyond domain/codomain).
- The Forgetful Functor $U: \mathrm{Grp} \to \mathrm{Set}$:
  - On Objects: $U(G)$ takes a group G and forgets the group operation, identity, and inverses, leaving just the underlying set of elements. For example:
    - $U\big((\mathbb{Z}, +)\big) = \{\dots, -2, -1, 0, 1, 2, \dots\}$, i.e., maps the group of integers to the set of integers, ignoring addition.
    - $U\big(GL_n(\mathbb{R})\big)$ is mapped to the set of all invertible $n \times n$ matrices over reals (ignoring matrix multiplication).

[9] Adjunction is a relationship that two functors may exhibit, intuitively corresponding to a weak form of equivalence between two related categories. Two functors that stand in this relationship are known as adjoint functors, one being the left adjoint and the other the right adjoint. By definition, an adjunction between categories $\mathcal{C}$ and $\mathcal{D}$ is a pair of functors (assumed to be covariant) $F: \mathcal{D} \to \mathcal{C}$ and $G: \mathcal{C} \to \mathcal{D}$ and, for all objects c in $\mathcal{C}$ and d in $\mathcal{D}$, a bijection between the respective morphism sets $\hom_{\mathcal{C}}(Fd, c) \cong \hom_{\mathcal{D}}(d, Gc)$ such that this family of bijections is natural in c and d. The functor F is called a left adjoint functor or left adjoint to G, while G is called a right adjoint functor or right adjoint to F. We write $F \dashv G$.

- On Morphisms: $U(\phi)$ takes a group homomorphism $\phi: G \to H$ and treats it as a plain function between the underlying sets. For example:
  - The homomorphism $\phi(n) = 2n$ becomes just the set function $f: \mathbb{Z} \to \mathbb{Z}$, where $f(n) = 2n$ (forgetting that it preserves addition).
- Preservation: Composition and identities morphisms are preserved because they hold in Grp and thus in Set, e.g., $U(\psi \circ \phi) = U(\psi) \circ U(\phi)$.

This functor is forgetful because it discards the algebraic structure (operation, identity, inverses) but keeps the set-theoretic foundation. It's like viewing a group as just a bunch of elements without their interactions.

Intuition with Analogy: Imagine a group as a team (set of people with roles/rules for collaboration). The forgetful functor forgets the roles and rules, seeing it as just a group of people (plain set). Homomorphisms become relocations without enforcing team dynamics.

Additional Examples of Forgetful Functors

From Rings (Ring) to Abelian Groups (AbGrp):

- Objects in Ring: Rings (sets with addition and multiplication, such as $\mathbb{Z}$ or polynomials $\mathbb{R}[x]$).
- Forgetful Functor $U: \mathrm{Ring} \to \mathrm{AbGrp}$: Forgets multiplication, keeping only the additive group structure. E.g., $U(\mathbb{Z}) = (\mathbb{Z}, +)$.
- Morphisms: Ring homomorphisms become group homomorphisms for addition.
- Why Useful: Allows studying additive properties separately from multiplicative ones.

From Vector Spaces ($Vect_K$) over Field K to Abelian Groups (AbGrp):

- Objects in $Vect_K$: Vector spaces, e.g., $\mathbb{R}^n$ with scalar multiplication and addition.
- Forgetful Functor $U: \mathrm{Vect_K} \to \mathrm{AbGrp}$: Forgets scalar multiplication by K, leaving the additive abelian group, e.g., $U(Vect_{\mathbb{R}^2}) = (\mathbb{R}^2, +)$.
- Morphisms: Linear maps become group homomorphisms.

From Topological Spaces (Top) to Sets (Set):

- Objects in Top: Sets with topologies (open sets defining continuity).
- Forgetful Functor $U: \mathrm{Top} \to \mathrm{Set}$: Forgets the topology, leaving the underlying set, e.g., $U(\mathbb{R}$ with standard topology$) = \mathbb{R}$ as a set.
- Morphisms: Continuous maps become plain functions.
- Useful for separating topological properties from set-theoretic ones.

Example 2: Free Functor

In category theory, a free functor is a functor $F: \mathcal{C} \to \mathcal{D}$ that is **left adjoint** to a forgetful functor $U: \mathcal{D} \to \mathcal{C}$. This means F equips objects from $\mathcal{C}$ with the minimal structure needed to live in $\mathcal{D}$, "freely generating" the necessary structure without imposing extra relations. The Free and Forgetful functors complement each other.

Key Properties and Motivation:

The adjunction between F and U (denoted $F \dashv U$) is defined by a natural bijection:

$$\hom_{\mathcal{D}}(F(c), d) \cong \hom_{\mathcal{C}}\big(c, U(d)\big), \text{ for all objects } c \in \mathcal{C} \text{ and } d \in \mathcal{D}$$

This bijection is the core idea behind the concept of adjunction:

- U is the forgetful functor, which strips structure from objects in $\mathcal{D}$ to get objects in $\mathcal{C}$.
- $F(c)$ is the *free object* on c in $\mathcal{D}$.
- The bijection says: A morphism from the free object $F(c)$ to any other object d in $\mathcal{D}$ corresponds uniquely to a morphism from the original object c to the underlying object $U(d)$ in $\mathcal{C}$.

The adjunction between F and U is depicted in Figure 4. The key insight is that these morphism sets are isomorphic: every morphism from c to $U(d)$ in $\mathcal{C}$ (g in the figure) corresponds to exactly one morphism from $F(c)$ to d in $\mathcal{D}$ ($\hat{g}$ in the figure), and vice versa.

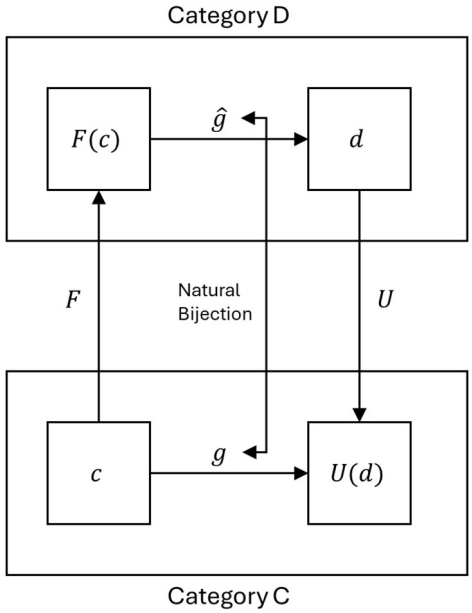


Figure 4. Adjunction between F and U

The natural bijection is equivalent to the existence of a **unit** natural transformation $\eta\colon 1_C \to U \circ F$ where for each object $c \in \mathcal{C}$, the morphism $\eta_c\colon c \to U\big(F(c)\big)$ satisfies the universal property that any morphism $g\colon c \to U(d)$ can be expressed uniquely as a composition $g = U(\hat{g}) \circ \eta_c$ for some morphism $\hat{g}\colon F(c) \to d$ in $\mathcal{D}$, where $U(\hat{g})$ is the image of some morphism $\hat{g}\colon F(c) \to d$ under the forgetful functor U. See Figure 5 for a depiction of the points made in this paragraph.

Note: $U(\hat{g})$ is not a composition but rather it is the result of applying the functor U to the morphism $\hat{g}$.

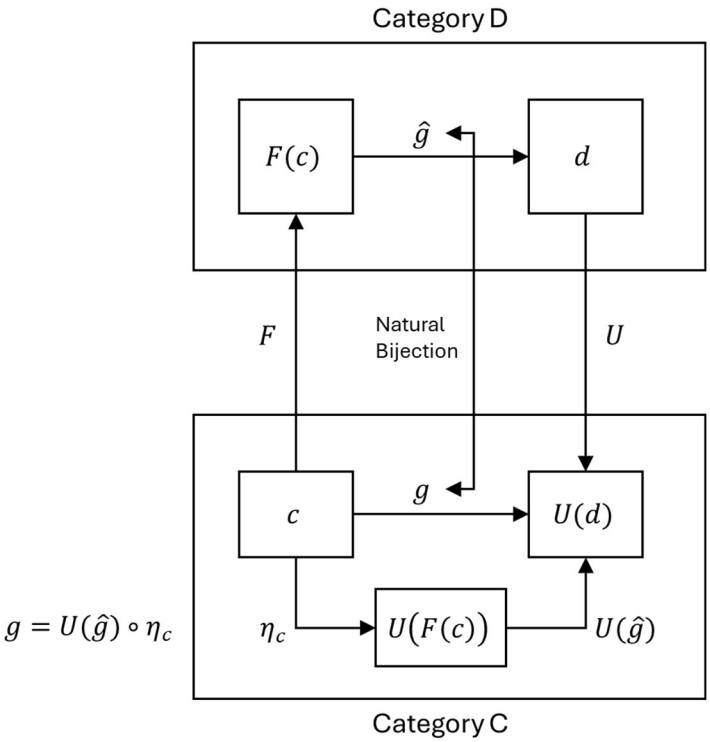


Figure 5. Adjunction between F and U (detailed version)

Concrete Example: Free Group ⊣ Forgetful Functor ($F \dashv U$)

- $\mathcal{C} = \text{Set},\ \mathcal{D} = \text{Grp}$
- $c = \{a, b\}$ (a set with two elements)
- $d = S^3$ (the symmetric group of degree 3)
- $F(c) = F(\{a, b\})$ (the free group on two generators, containing elements like $a, b, ab, a^{-1}b$, etc.)
- $U(d) = \{e, (12), (13), (23), (123), (132)\}$ (the underlying set of the group S^3, forgetting the group operation)

The bijection in action:

- A function $g: \{a, b\} \to U(S^3)$ in Set might be defined by:
  - $g(a) = (12)$
  - $g(b) = (23)$
    (This is just an arbitrary assignment of set elements.)

- The adjunction guarantees a unique group homomorphism $\hat{g}: F(\{a, b\}) \to S^3$ in Grp that *extends* this assignment. This homomorphism is defined for all elements of the free group by respecting the group structure. For example:
  - $\hat{g}(a) = (12)$
  - $\hat{g}(b) = (23)$
  - $\hat{g}(ab) = \hat{g}(a)\,\hat{g}(b) = (12)(23) = (123)$
  - $\hat{g}(a^{-1}) = \hat{g}(a)^{-1} = (12)^{-1} = (12)$

The diagram visualizes this powerful idea: specifying what happens to the generators (a map in $\mathcal{C}$) uniquely determines what happens to the entire structured object (a map in $\mathcal{D}$).

Canonical Example: Free Groups

Consider the Set and Grp categories:

- The forgetful functor $U: \mathrm{Grp} \to \mathrm{Set}$ sends a group to its underlying set (forgetting multiplication, inverses, etc.).
- Its left adjoint, the free functor $F: \mathrm{Set} \to \mathrm{Grp}$, constructs the free group generated by a set.
  - For a set S, $F(S)$ is the free group whose elements are all possible reduced words built from the letters in S and their formal inverses, with the group operation being concatenation of words.
- The unit of this adjunction provides, for each set S, a function $\eta_S: S \to U\big(F(S)\big)$. This function injects each element of S into the free group as a one-letter word (a generator).

This generalizes to free monoids, free modules, free algebras, etc., across algebraic categories.

Why "Free"?

The term emphasizes universality: $F(c)$ comes with a canonical map $\eta_c: c \to U\big(F(c)\big)$ such that any other map $c \to U(d)$ extends uniquely to a homomorphism $F(c) \to d$. No unnecessary relations are forced, making it the initial object under the forgetful functor.

Example 3: Power Set Functor

From Set to Set: To each set X, assign the power set $\mathcal{P}(X)$, i.e., the set of all subsets of X. To each morphism $f: X \to Y$, assign a morphism $F(f): \mathcal{P}(X) \to \mathcal{P}(Y)$, where $F(f)(S) = \{\, f(s) \mid s \in S \,\}$ for subset $S \subseteq X$.

Functors allow us to translate problems between categories, e.g., from algebra to geometry.

10.5 Natural Transformations: Maps Between Functors

Once we have functors as maps between categories (as defined in the previous subsection), the next level of abstraction is to consider maps between functors themselves. This is where natural transformations come in. Natural transformations provide a way to relate two functors that go from the same source category to the same target category, in a manner that respects the structure of both categories. They are essential for understanding how different functorial constructions can be transformed uniformly across all objects and morphisms.

To make this concrete, recall that functors preserve the shape of a category – objects go to objects, morphisms to morphisms, and compositions/identities are respected. A natural transformation is like a coherent bridge between two such functors: it assigns a morphism in the target category for each object in the source, but ensures these assignments are well behaved with respect to the morphisms in the source category.

10.5.1 Basic Definition

Given two functors $F: \mathcal{C} \to \mathcal{D}$ and $G: \mathcal{C} \to \mathcal{D}$, both mapping from category $\mathcal{C}$ to category $\mathcal{D}$, a **natural transformation** $\eta: F \Rightarrow G$ (read as "eta from F to G") satisfies the following properties:

- A natural transformation η assigns to each object A in $\mathcal{C}$ a morphism $\eta_A: F(A) \to G(A)$ in $\mathcal{D}$. This morphism η_A is called the component morphism at the object A (or simply the "component at A"). Think of it as the piece of the transformation η specific to A. It tells us how to convert the image of A under F to its image under G.
- These component morphisms must satisfy a compatibility condition called the naturality condition (explained below).

In other words, a natural transformation is a family of morphisms (one per object in $\mathcal{C}$) that transform the outputs of F into the outputs of G consistently, see Figure 6.

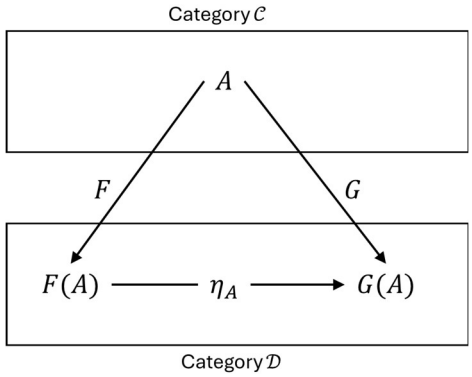


Figure 6. Natural Transformation

Natural transformations are the "morphisms" in the "category of functors."

10.5.2 The Naturality Condition: What Makes It "Natural"?

The key requirement is that the transformation must respect the morphisms in $\mathcal{C}$. Specifically, for every morphism $f: A \to B$ in $\mathcal{C}$, the following equation must hold:

$$G(f) \circ \eta_A = \eta_B \circ F(f)$$

This is the naturality condition. It ensures that whether you first apply the functor (F or G) to the morphism f and then transform, or transform first and then apply the functor, you get the same result. This coherence is what makes the transformation natural, i.e., it's uniform across the entire category, not ad hoc for specific objects.

To visualize this, we use a commutative diagram, a tool in category theory for showing that different paths of compositions lead to the same morphism. A diagram commutes if all paths from a starting object $F(A)$ to an ending object $G(B)$ yield equal morphisms, see Figure 7.

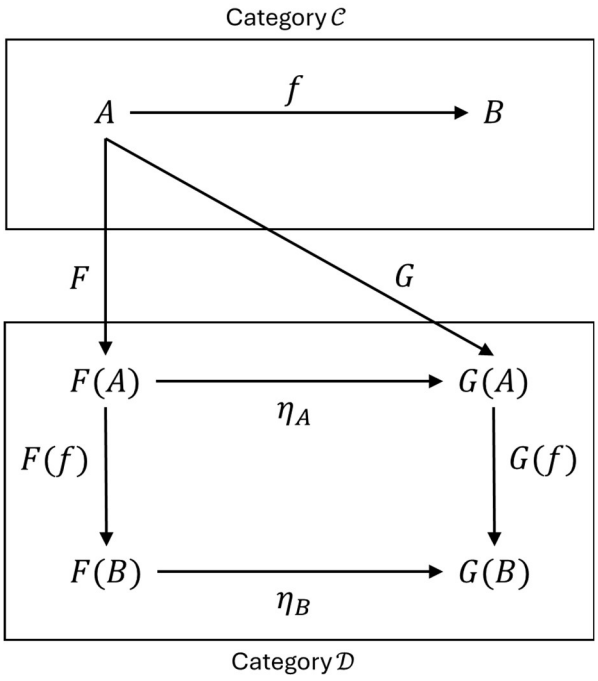


Figure 7. Commutative diagram for the naturality condition

Concerning the above figure:

- The "right then down" path: η_A followed by $G(f)$, i.e., $G(f) \circ \eta_A$.
- The "down then right" path: $F(f)$ followed by η_B , i.e., $\eta_B \circ F(f)$.
- **For the diagram to commute, these two paths must be equal**: $G(f) \circ \eta_A = \eta_B \circ F(f)$.
- The arrows from B to $F(B)$ and $G(B)$ were omitted as they would make the figure too busy.

If this holds for every morphism f and object A, the transformation is natural. Commutative diagrams are like equality checks in category theory, i.e., they make abstract relations visual and easier to verify.

10.5.3 Why Natural Transformations Matter

Natural transformations capture uniform changes between functorial views of a category. They arise in many areas:

- In algebra, they relate different representations (e.g., vector spaces versus their duals).
- In the context of this book, they appear in synthetic differential geometry, where they help define smooth maps involving infinitesimals, and in automatic differentiation, where they model transformations of computational graphs.

Without naturality, a collection of component morphisms might work for isolated objects but fail when morphisms connect them; naturality ensures global consistency.

10.5.4 Example 1: The Diagonal Transformation

Consider the category Set (sets and functions, from Section 10.2). Define two functors from Set to Set:

- The **identity functor** Id: Maps each set X to itself, i.e., $Id(S) = S$, and each function f to itself, i.e., $Id(f) = f$.
- The **double functor** D: Maps each set S to $S \times S$, i.e., the Cartesian product of S with itself, and each function $f: S \to T$ to $D(f): S \times S \to T \times T$ where $D(f)(x, y) = \big(f(x), f(y)\big)$.

A natural transformation $\eta: I \Rightarrow D$ assigns, to each set X, a component morphism $\eta_X: Id(S) \to D(S)$, i.e., $\eta_S: S \to S \times S$.

One such η is the **diagonal map**: $\eta_S(x) = (x, x)$ for each $x \in S$. This sends each element x to a pair (x, x).

Let's check for naturality. For any function $f: S \to T, x \in S$, we have

- "Right then down" path (relative to Figure 7): $(D(f) \circ \eta_S)(x) = D(f)(x, x) = (f(x), f(x))$.
- "Down then right" path: $\big(\eta_T \circ Id(f)\big)(x) = \eta_T\big(f(x)\big) = \big(f(x), f(x)\big)$.

Both sides equal $\big(f(x), f(x)\big)$, so the diagram commutes: $D(f) \circ \eta_S = \eta_T \circ Id(f)$. Thus, η is natural.

This example shows how natural transformations reuse simple ideas (like duplicating elements) to work across all sets and functions uniformly.

10.5.5 Example 2: Inclusion of Constants

In the category of groups Grp (from Section 10.2), consider two functors to Grp:

- The constant functor K maps every group to the trivial group $\{e\}$ (the group with only one element, e), and every group homomorphism $f: G \to H$ to the identity homomorphism $id_{\{e\}}: \{e\} \to \{e\}$ (which simply sends e to e).
- The identity functor I on Grp.

A natural transformation $\eta: K \Rightarrow I$ would assign, for each group $G, \eta_G: \{e\} \to G$. The only choice is $\eta_G(e) = e_G$ (identity element of G). Naturality holds because group homomorphisms preserve identities (see below for the details). This illustrates how natural transformations can embed trivial structures into more complex ones consistently.

Checking the naturality condition:

The condition we need to verify is: $I(f) \circ \eta_G = \eta_H \circ K(f)$ for groups G and H.

Let's evaluate both sides of this equation. Since the only element in the trivial group $\{e\}$ is e, we will follow what happens to this single element.

"Right then down" Path (relative to Figure 7): $I(f) \circ \eta_G$

1. Start with e in $\{e\}$.
2. Applying η_G, we have $\eta_G(e) = e_G$ (the identity in G).
3. Applying $I(f)$ (which is just f), we have $f(e_G) = e_H$ (because group homomorphisms must map the identity element to the identity element).

Result of Left Path: $e \to e_H$

"Down then right" Path: $\eta_H \circ K(f)$

1. Start with e in $\{e\}$.
2. Applying $K(f)$ (which is $id_{\{e\}}$, the identity map on $\{e\}$), we have $id_{\{e\}}(e) = e$.
3. Applying η_H, we have $\eta_H(e) = e_H$ (the identity in H).

Result of Right Path: $e \to e_H$

10.5.6 Common Pitfalls and Tips for Beginners

"Component at A" might sound vague, but it's just the specific morphism assigned to object A. Think of the transformation as a dictionary where each object A looks up its own conversion map η_A.

Commutative diagrams can seem intimidating, but they're just visual equations. If a diagram doesn't commute, the transformation isn't natural.

Not all families of morphisms are natural. Naturality is the extra condition that makes them categorical.

10.6 Cartesian Closed Categories: Prelude to Synthetic Differential Geometry

10.6.1 Introduction to Cartesian Closed Categories

Cartesian closed categories (CCCs) represent a class of categories that possess sufficient structure to support both product constructions (for combining objects) and exponential constructions (for modeling function spaces). These categories are closed in the sense that they allow for the internalization of hom-sets[10] as objects within the category itself, making them powerful for abstracting concepts like logic, computation, and smoothness. In the context of this book, CCCs serve as a foundational structure for synthetic differential geometry, where they enable the rigorous treatment of nilpotent infinitesimals in a categorical setting, bridging the algebraic approaches of dual numbers and the analytical frameworks of hyperreals.

To understand CCCs, we build on the basics of categories, morphisms, functors, and natural transformations discussed earlier in this section. A CCC is a category that is "Cartesian" (has finite products) and closed (has exponentials relative to those products). In what follows, we'll define these components step by step, with examples to illustrate their roles.

[10] In category theory, a hom-set (short for "homomorphism set" or more formally "hom-set of morphisms") is the collection of all morphisms (arrows) between two specific objects in a category. The term "hom" is derived from "homomorphism," reflecting its origin in algebra, where morphisms preserve structure, but in category theory, it's generalized to any structure-preserving map between objects.

10.6.2 Some Topological Background

In what follows, we use some terms from topology. To help the reader who is not familiar with topology, this subsection provides a high-level, intuitive description of several concepts from topology. For an introduction to topology, see Section 3 of “Mathematical Vignettes - Volume III” [54].

Hausdorff: “Points Can Be Shy Neighbors”

- The Core Idea: In a Hausdorff space, any two distinct points are “socially distant.” Each one can have its own private neighborhood that doesn't intrude on the other’s.
- The Analogy: Imagine a neighborhood with very large properties. For any two houses, you can draw a fence around each property that doesn't include the other house. The points never have to interact if they don't want to.
- Why it Matters: This is the fundamental “niceness” property of a space. It guarantees that sequences (or nets) can't converge to two different points at once, which is essential for doing calculus and analysis. It means points are “individuals” and not “stuck together.”

Weak Hausdorff: “Nice Shapes Don't Get Fuzzy”

- The Core Idea: A space where continuous images of other compact spaces don’t cause unexpected “clumping” or arbitrary limit points. While not every distinct pair of points is perfectly separated, any “compact blob” (the image of a compact space) is forced to behave nicely: it’s guaranteed to be a closed set.
- The Analogy: Imagine a well-run museum with flexible exhibits. Artists can install complex sculptures (compact shapes) in the galleries. The museum’s rule is: while visitors (individual points) might sometimes get close to each other, the entirety of any official sculpture must be neatly contained and can be roped off completely from the rest of the museum.
- Why it Matters (and the Trade-Off): The standard Hausdorff condition is very restrictive for constructing new spaces, especially via quotients (gluing points together), which often destroy the Hausdorff property. Weak Hausdorff is the crucial relaxation that makes the category of compactly generated weak Hausdorff spaces both well-behaved and cartesian closed. It retains just enough control—ensuring compact sets are closed—to avoid pathological behavior.

Compact Hausdorff: “The Perfect, Self-Contained Universe”

- The Core Idea: A space that is both Compact (“finite in essence”) and Hausdorff (“points are distinct”).
- The Analogy: Imagine a perfectly detailed, physical globe of the world.
    - Hausdorff (The Detail): On this high-quality globe, Paris and London are distinct, separate cities. You can always point to a tiny area around one that doesn't include the other.
    - Compact (The Globe Itself): The entire world is represented on a finite, self-contained object. There's no “edge” or “infinity”; if you travel far enough, you just come back to where you started. It's a complete and bounded representation.

- The “Magic” Property: These spaces are the “gold standard” of topological rigidity. The combination of these two properties creates a powerful mathematical lock. If you have a continuous, one-to-one map from a Compact Hausdorff space to another Hausdorff space, the map is automatically a homeomorphism (a perfect, reversible deformation). You don't need to check if the inverse is continuous—the compactness “traps” the map, forcing it to be perfectly well-behaved. It's like a seal of quality that guarantees perfect structure preservation.

10.6.3 Finite Products: Combining Objects

A category $\mathcal{C}$ has finite products if it supports the construction of product objects for any finite collection of objects, including the empty product, which is a terminal object, typically denoted as 1 or *. More precisely:

- For any two objects A and B in $\mathcal{C}$, there exists a product object $A \times B$ along with projection morphisms $\pi_1 : A \times B \to A$ and $\pi_2 : A \times B \to B$.
- This product satisfies a universal property: For any object C and morphisms $f : C \to A$, $g : C \to B$, there exists a unique morphism $\langle f, g \rangle : C \to A \times B$ (the “pairing” morphism) such that $\pi_1 \circ \langle f, g \rangle = f$ and $\pi_2 \circ \langle f, g \rangle = g$.

This universal property ensures that $A \times B$ is the most general object that projects to both A and B, see Figure 8.

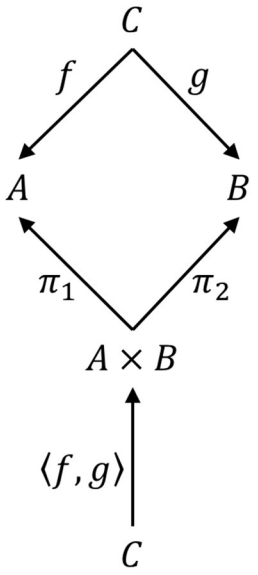


Figure 8. Diagram for finite product

The above diagram commutes, meaning that the morphism obtained by following the path $C \to A \times B \to A$ (i.e., $\pi_1 \circ \langle f, g \rangle$) is equal to the morphism f, and similarly for g.

Example in Set: The category of sets.

- Product $A \times B$ is the Cartesian product, i.e., a set of ordered pairs (a, b).
- Projections: $\pi_1(a, b) = a$, $\pi_2(a, b) = b$.

- Pairing: For morphisms $f: C \to A$ and $g: C \to B$, the unique pairing morphism is defined by $\langle f, g \rangle(c) = \big(f(c), g(c)\big)\ \forall c \in C$. This definition satisfies $\pi^1 \circ \langle f, g \rangle = f$ and $\pi^2 \circ \langle f, g \rangle = g$, since for any $c \in C$, we have $\pi^1\big(\langle f, g \rangle(c)\big) = \pi^1\big(f(c), g(c)\big) = f(c)$ and similarly $\pi^2\big(\langle f, g \rangle(c)\big) = g(c)$.
- Terminal: The terminal object is the singleton set $\{*\}$. For each set C, there exists a unique morphism (function) $! : C \to \{*\}$, which sends every element of C to the single element $*$. [The notation $\{*\}$ refers to the singleton set containing a single arbitrary element, conventionally denoted by the symbol $*$ (which could be any fixed element, like a point, a dummy variable, or even the empty set itself – it's just a placeholder).]

Finite products allow categories to model tupling or pairing operations, essential for multivariable functions in differentiation.

10.6.4 Exponentials: Modeling Function Spaces

A category $\mathcal{C}$ with finite products is Cartesian closed if it has exponential objects for every pair of objects — objects that internalize the notion of "function space" or "morphisms from A to B" as a genuine object inside $\mathcal{C}$ itself.

More precisely, a category with finite products is **Cartesian closed** if for all objects A and B:

- There exists an exponential object B^A in the category $\mathcal{C}$.
- There exists an **evaluation morphism** ev: $B^A \times A \to B$ in category $\mathcal{C}$, which applies a generalized function from B^A to an input from A to produce an output in B.
- **Universal Property (Currying):** For any object C and any morphism $h: C \times A \to B$ in category $\mathcal{C}$, there exists a **unique morphism** $\lambda h: C \to B^A$ (called the **transpose** or **currying** of h) in category $\mathcal{C}$ such that the following diagram commutes.

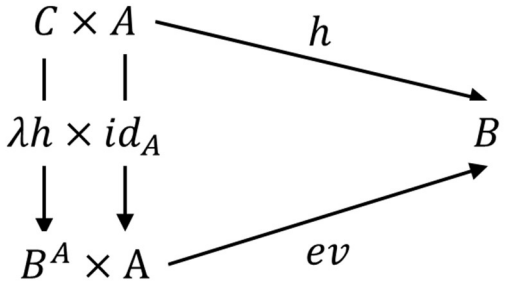


Figure 9. Commutative diagram concerning exponentials

This universal property is exactly currying: a map taking a pair (C, A) turns into a map that first takes C and returns "a function" (represented internally) that can later be applied to A.

The crucial point is that B^A must be an honest object of $\mathcal{C}$ — not an external set of morphisms, not a proper class, and not a set equipped with extra structure foreign to $\mathcal{C}$. Currying and evaluation must happen using only morphisms that exist in $\mathcal{C}$.

This single requirement is remarkably strong, and most familiar concrete categories fail it. Here are the classic reasons exponentials are missing:

- **Size issues**: In categories that admit proper classes (e.g. the category of sets and classes in Gödel–Bernays set theory), if A is a proper class, the would-be function class B^A is "larger" and not itself an object of the category.
- **Structure incompatibility**: The category **Ab** of abelian groups has products but is not cartesian closed. The functor $(-) \times A$ fails to have a right adjoint for many objects A (e.g., $A = \mathbb{Z}_2$).
- **Topological issues**: In the usual category **Top** of all topological spaces and continuous maps, the set of continuous functions $\mathbb{R} \to \mathbb{R}$ can be topologized in many ways (compact-open, product, etc.), but none of them make the evaluation map represent $\hom\big((-) \times \mathbb{R}, \mathbb{R}\big)$ universally inside Top. Many other pairs fail similarly.

So, even though in **Set** (and similar "set-like" categories) exponentials always exist and feel completely natural, cartesian closedness is a non-trivial extra property. Most algebraic categories (groups, rings, modules, etc.) and most analytic categories (manifolds, Banach spaces, etc.) are cartesian but not cartesian closed.

Example in Set:

In the category of sets, exponentials have a concrete interpretation.

- **Exponential Object B^A:** This is the set of all functions from A to B, denoted $B^A = \{f : A \to B\}$.
- **Evaluation Morphism:** The morphism *ev*: $B^A \times A \to B$ is defined by $ev(f, a) = f(a)$ for any function $f \in B^A$ and element $a \in A$.
- **Currying**: For a function $h: C \times A \to B$, the curried version $\lambda h: C \to B^A$ is defined as follows: For each element $c \in C$, the value $(\lambda h)(c)$ is a new function from A to B. This new function is defined by its action on any input $a \in A$, i.e., $\big((\lambda h)(c)\big)(a) = h(c, a)$. In the more compact lambda notation, this is written as $(\lambda h)(c) = \big(a \mapsto h(c, a)\big)$. Here, the symbol a is a bound variable used to define the new function; it is not a specific element of A. For each c, this construction yields a specific function $f_c: A \to B$ given by $f_c(a) = h(c, a)$.
- **Connection to Evaluation:** This definition is precisely what is needed to satisfy the universal property. The key equation $ev \circ (\lambda h \times id_A) = h$ must hold. Let's verify it point by point for any $(c, a) \in C \times A$:
  - $(\lambda h \times id_A)(c, a) = \big((\lambda h)(c), a\big)$
  - Applying ev to this pair gives: $ev\big((\lambda h)(c), a\big)$
  - But, by the very definition of *ev*, this equals $\big((\lambda h)(c)\big)(a)$
  - And by our definition of λh, this is exactly $h(c, a)$.

  Therefore, $\big(ev \circ (\lambda h \times id_A)\big)(c, a) = h(c, a)$ for all (c, a), proving the equality $ev \circ (\lambda h \times id_A) = h$.

10.6.5 Full Definition of Cartesian Closed Categories

A category $\mathcal{C}$ is **Cartesian closed** if:

1. It has all finite products (including a terminal object, denoted 1 or $*$).
2. For every pair of objects A and B, there exists an **exponential object** B^A and an **evaluation morphism** ev: $B^A \times A \to B$. This data must satisfy the following universal property: for any object C and morphism $h: C \times A \to B$, there exists a **unique** morphism $\lambda h: C \to B^A$ such that the diagram in Figure 9 commutes. Equivalently, $\text{ev} \circ (\lambda h \times \text{id}_A) = h$.

The existence of exponentials means the category is closed under its Cartesian monoidal structure. Categorically, this means the product functor $(-) \times A$ has a right adjoint, the exponential functor $(-)^A$.[11] This adjunction is the essence of the universal property, creating a natural bijection $\hom(C \times A, B) \cong \hom(C, B^A)$.

Not all categories with finite products are Cartesian closed. For example, the category Top of topological spaces and continuous maps has finite products (the usual product topology) but is not cartesian closed. The reason is that for arbitrary topological spaces A and B, although the set of continuous functions from A to B, i.e., B^A, can be equipped with many different topologies (the compact-open topology being the most common candidate), there is no single topology on the function space that works for all pairs of spaces in such a way that the evaluation map *ev*: $B^A \times A \to B$ is continuous universally, i.e., that makes the exponential law hold in the category for all objects. In other words, the functor $(-) \times A$ does not have a right adjoint for every space A in Top.

However, certain "nice" subcategories of Top are indeed cartesian closed, most famously the full subcategory of compactly generated (or k $-$) spaces with the compactly generated product and compact-open topology, often denoted kTop or CGTop. The category of compactly generated weak Hausdorff spaces is also cartesian closed and is the standard convenient category of spaces used in algebraic topology.

Why the standard attempt fails in Top:

If one tries to define the exponential object using the compact-open topology on the set of continuous maps from A to B:

- When A is locally compact Hausdorff, the compact-open topology **does** make evaluation continuous, and Top restricted to such spaces is cartesian closed.
- But for general (non-locally-compact) spaces, evaluation is not continuous with respect to the compact-open topology on the function space and the product topology on the domain. Classic counterexample: let $A = \mathbb{Q}$ with the subspace topology from $\mathbb{R}$, and $B = \mathbb{R}$. Then evaluation fails to be continuous.

Hence Top itself lacks exponentials for all objects, so it is not cartesian closed.

[11] The parentheses with a dash, $(-)$, represent a blank or a placeholder for an object. It signifies that we are not talking about a single object, but a *rule* or *operation* that can be applied to *any* object in the category.

Key Properties of CCCs:

- **Foundation for Functional Programming:** CCCs provide a categorical model for the simply typed lambda calculus. Products correspond to pair types (or tuples/records), the terminal object to the unit type, and exponentials to function types. Evaluation models function application, and currying (λh) is the process of converting a function that takes a tuple into a higher-order function.
- **Internal Logic:** CCCs support an internal logic, often intuitionistic propositional logic. In this interpretation, conjunction ($\wedge$) corresponds to products ($\times$), truth ($\top$) to the terminal object (1), and implication ($\Rightarrow$) to exponentials ($\rightarrow$).

10.6.6 Examples of Cartesian Closed Categories

- **Set** (Sets and Functions): As detailed previously, the Set category with Cartesian products and function sets is the canonical example and models classical set theory.
- **FinSet** (finite sets and functions): This is a CCC. It has all finite products (including the terminal object, i.e., the singleton set), and exponentials exist internally, i.e., for any two finite sets A and B, the set B^A is again a finite set (of cardinality $|B|^{|A|}$, which is finite), and the usual set-theoretic evaluation map works perfectly.
- **Top** (Topological Spaces and Continuous Maps): This is **not** a CCC in its standard form. The problem is that the set-theoretic exponential may not have a topology making the evaluation map continuous. However, the subcategory of compactly generated (weak Hausdorff) spaces is Cartesian closed and is often used as a convenient category for algebraic topology.
- **Cat** (Small Categories and Functors): This is a CCC. The product of two categories is the standard Cartesian product. The exponential $\mathcal{D}^{\mathcal{C}}$ is the functor category, whose objects are functors from $\mathcal{C}$ to $\mathcal{D}$ and whose morphisms are natural transformations.

These examples illustrate the wide applicability of CCCs in logic, computation, and geometry.

10.6.7 Prelude to Synthetic Differential Geometry

Synthetic Differential Geometry (SDG) is usually developed inside certain toposes[12] that are cartesian closed and satisfy additional axioms, the most famous being the Kock–Lawvere axiom[13]. This axiom guarantees the existence of an object $\mathbb{D} = \{d \in \mathbb{R} : d^2 = 0\}$ of nilpotent infinitesimals, allowing derivatives to be defined algebraically via the formal identity $f(x + d) = f(x) + df'(x)$ valid for all functions f and all $d \in \mathbb{D}$. Well-adapted models (e.g., the Dubuc topos, Cahiers topos, or the smooth topos $\mathcal{B}^{\infty}$) are the standard settings for SDG; a mere cartesian closed category is too weak on its own.

[12] A topos (plural: toposes) is a category that serves as a generalized framework for doing mathematics, combining the flexibility of category theory with the power of classical set theory.

[13] The Kock-Lawvere axiom is a foundational principle in synthetic differential geometry (SDG), developed by Anders Kock and William Lawvere in the 1970s. SDG uses category theory to provide a rigorous framework for working with infinitesimals, avoiding the traditional $\varepsilon - \delta$ limits of classical analysis. The Kock-Lawvere axiom specifically introduces nilpotent infinitesimals into a categorical setting, resembling the dual numbers but embedded within a topos (a special kind of category with rich structure). This axiom enables a synthetic approach to calculus, where derivatives and smooth functions are defined directly using infinitesimals.

10.6.8 Exercises

Exercise 1: Verify Set is CCC by constructing exponentials for specific sets.

Exercise 2: Show that the category Top is not CCC. This exercise assumes knowledge of topology.

Answer to Exercise 1:

The category Set (sets and functions) is a classic example of a Cartesian closed category (CCC). To verify this, we explicitly construct the exponential object B^A for specific finite sets A and B, define the evaluation morphism *ev*: $B^A \times A \to B$, and demonstrate the universal property with a concrete morphism h: $C \times A \to B$.

Specific Sets to be Used in the Answer

Let:

- $A = \{1,2\}$ (a set with two elements),
- $B = \{x, y\}$ (a set with two elements),
- $C = \{p, q\}$ (another set with two elements, for the universal property example).

Constructing the Exponential B^A

The exponential object B^A is the set of all functions from A to B. Since A and B each have 2 elements, there are $2^2 = 4$ such functions. Label them explicitly:

- f_1: $1 \to x, 2 \to x$ (constant x),
- f_2: $1 \to x, 2 \to y$,
- f_3: $1 \to y, 2 \to x$,
- f_4: $1 \to y, 2 \to y$ (constant y).

Thus, $B^A = \{f_1, f_2, f_3, f_4\}$, which is finite (as expected in Set).

Evaluation Morphism *ev*: $B^A \times A \to B$

The product $B^A \times A$ consists of pairs (f, a) where $f \in B^A$ and $a \in A$. Define:

$$\text{ev}(f, a) = f(a).$$

This applies the function f to a, yielding an element of B. We have that

- $ev(f_1, 1) = x$
- $ev(f_2, 2) = y$
- $ev(f_3, 1) = y$
- $ev(f_4, 2) = y$, and so on.

This is a well-defined function in Set.

Universal Property

To verify closure, take a morphism $h: C \times A \to B$. The product $C \times A = \{(p,1),(p,2),(q,1),(q,2)\}$. Define a specific h (as a function on pairs):

- $h(p,1) = x, \;\; h(p,2) = y$
- $h(q,1) = y, \;\; h(q,2) = x$

The universal property requires a unique $\lambda h: C \to B^A$ such that $ev \circ (\lambda h \times id_A) = h$.

Construct λh (the currying):

- For input $p \in C$, $\lambda h(p)$ should be the function that sends $a \in A$ to $h(p,a)$, i.e., $1 \to x, 2 \to y$. This is f_2.
- For input $q \in C$, $\lambda h(q)$ sends $1 \to y, 2 \to x$. This is f_3.

Thus, $\lambda h: p \to f_2, q \to f_3$ (functions from C to B^A).

Now verify the composition:

- $(\lambda h \times id_A)(p,1) = (f_2,1)$, so $ev(f_2,1) = x = h(p,1)$,
- $(\lambda h \times id_A)(p,2) = (f_2,2)$, so $ev(f_2,2) = y = h(p,2)$,
- Similarly, for q: $ev(f_3,1) = y = h(q,1)$, $ev(f_3,2) = x = h(q,2)$.

It commutes! Uniqueness follows because any other $\lambda h'$ would need to satisfy the same mappings on pairs, forcing f_2 and f_3.

This construction generalizes to arbitrary sets in Set, confirming it's a CCC.

Answer to Exercise 2:

A classic example of a category that has all finite products but is not Cartesian closed is Top, the category where objects are topological spaces and morphisms are continuous functions.

Why It Has Finite Products

- The categorical product of two spaces X and Y is the set-theoretic product $X \times Y$ equipped with the product topology (whose basis consists of rectangles $U \times V$ with U open in X and V open in Y).
- The projections are continuous and satisfy the universal property: for any space Z and continuous maps $f: Z \to X, \;\; g: Z \to Y$ there is a unique continuous $\langle f, g \rangle: Z \to X \times Y$ making the obvious diagrams commute.
- Finite products exist in general, and the terminal object is the one-point space $\{*\}$.

Why It Is Not Cartesian Closed

As noted, the category **Top** of topological spaces and continuous maps has all finite products (product topology, projections continuous, universal property holds) and a terminal object (the one-point space). However, it is **not** cartesian closed because the functor $(-) \times A$ does **not** have a right adjoint for every space A.

Although the set of continuous maps from B to A, denoted $Ct(B, A)$, always exists set-theoretically, there is no topology one can put on it once and for all (independent of B) such that:

- evaluation *ev*: $Ct(B, A) \times B \to A$ is continuous, and
- the universal property of the exponential holds internally in Top for every test object C.

The usual candidate topology is the compact-open topology. It works perfectly when B is locally compact Hausdorff, but fails in general. A standard counterexample is as follows:

> Consider the exponential $\mathbb{R}^{\mathbb{Q}}$, where $\mathbb{Q}$ is the rationals with the subspace topology and $\mathbb{R}$ the reals. When the function space is given the compact-open topology, the evaluation map $ev: \mathbb{R}^{\mathbb{Q}} \times \mathbb{Q} \to \mathbb{R}$ is not continuous. For the details of why this is true, see the paper by Hallam [55].

Another common example uses an infinite product space that is not locally compact.

However, there are useful smaller categories inside Top that are easier to work with and are cartesian closed. A main example is the category of “compactly generated weak Hausdorff spaces” (often shortened to CGWH or kHaus). In this category, spaces of continuous functions use a special topology called compact-open, and products use a slightly adjusted version of the usual product topology (called “Kelleyfied” or “k-ified”). These categories are commonly used in algebraic topology, as first suggested by mathematician Norman Steenrod in his work on “convenient” categories of spaces.

11 Synthetic Differential Geometry: A Categorical Lens

There is no such thing as a new idea. We just take a lot of old ideas and put them into a sort of mental kaleidoscope. — Mark Twain

11.1 Introduction

Synthetic differential geometry (SDG) is an axiomatic framework for differential geometry that utilizes category theory to formalize infinitesimals in a way that allows for intuitive, synthetic reasoning about smoothness and derivatives. Developed primarily by Anders Kock and William Lawvere in the 1970s, SDG operates within special categories called toposes, where infinitesimal objects are built-in, enabling a coordinate-free approach to calculus and geometry. SDG contrasts with classical differential geometry, which relies on limits and topology, and complements the non-standard analysis of hyperreals and the algebraic methods of dual numbers.

In SDG, the smooth real line $\mathcal{R}$ is a commutative ring with nilpotent infinitesimals, enabling direct algebraic manipulation like in dual numbers, but now embedded in a categorical structure called a *topos*. Key to this is the Kock-Lawvere axiom, which ensures that functions on infinitesimal neighborhoods are linear, facilitating synthetic definitions of derivatives, tangent bundles, and higher-order structures. This section explores SDG's foundations, axioms, and applications, showing how it unifies the infinitesimal approaches from previous sections, and provides a prelude to automatic differentiation which we discuss in the following section. We'll use examples to illustrate concepts, assuming familiarity with category theory basics (as discussed in the previous section).

11.2 Subobjects and Colimits in Category Theory

Before delving into the categorical foundations of SDG, it is essential to introduce two key concepts from category theory: **subobjects** and **colimits**. These notions extend the basic structures discussed in Section 8 (e.g., objects, morphisms, and limits) and are crucial for understanding toposes and the infinitesimal objects in SDG. We'll define them here with examples, assuming familiarity with categories and monomorphisms.

11.2.1 Subobjects: Generalized Subsets

In category theory, a subobject is a generalization of the concept of a subset or subgroup, adapted to arbitrary categories. It represents a part of an object that is embedded via a monomorphism (a "one-to-one" morphism).

- Definition: For an object A in a category $\mathcal{C}$, a subobject of A is a pair (S, i), where S is an object and $i: S \to A$ is a monomorphism (mono). Two subobjects (S, i) and (S', i') are equivalent if there is an isomorphism $\phi: S \to S'$ such that $i' \circ \phi = i$. Therefore, a subobject is properly understood as an equivalence class of monomorphisms into A.
- Intuition: In Set, a subobject of A is just a subset $S \subseteq A$, with i being the inclusion map. The mono property ensures no collapse (injectivity). In Grp, subobjects are subgroups with inclusion homomorphisms.
- Why Monomorphisms?: Monos act as embeddings, preserving distinctness, making subobjects faithful parts of A.

Example: In Set, for $A = \{1,2,3\}$, the pair (S, i) where $S = \{1,2\}$ and i is the inclusion map, is a subobject. This is equivalent to the subobject (S', i') where $S' = \{a, b\}$ and i' is defined by $i'(a) = 1, i'(b) = 2$. The equivalence is given by the isomorphism $\phi: S \to S'$ with $\phi(1) = a, \phi(2) = b$, which satisfies $i' \circ \phi = i$.

Subobjects are classified in toposes via the subobject classifier Ω, which generalizes the set of truth values {true, false} and enables the rich internal logic used to define and reason about infinitesimals in SDG.

11.2.2 Limits: Combining Objects via Universal Properties

A **limit** is a categorical construction that combines a collection of objects and morphisms (a diagram) into a single object, serving as a universal solution that respects all the relationships in the diagram. Limits are the dual (or opposite) of colimits, which we'll define in the following subsection, and they generalize concepts like products and equalizers.

- Definition: A limit of a diagram (a collection of objects D_i and some morphisms between them) in a category $\mathcal{C}$ is an object L together with morphisms $\pi_i: L \to D_i$ (called projections) for each D_i, satisfying a universal property:
    - For any other object M (*not necessarily* the limit L or equipped with the limit's projections) with morphisms $f_i: M \to D_i$ compatible with the diagram (i.e., for every morphism $g: D_i \to D_j$ in the diagram, we have $f_j = g \circ f_i$), there exists a unique morphism $\mu: M \to L$ such that $\pi_i \circ \mu = f_i$ for all i.
- Intuition: The limit L is the most general object that can project to each D_i while preserving the diagram's structure. It's like finding a common ancestor or intersection that fits all given relationships. Limits are greatest lower bounds in the categorical sense. The limit is the best (universal) way to map into the whole diagram at once.

As an example of the definition of a limit, consider a category $\mathcal{C}$ with only three objects, i.e., D_1, D_2, D_3. In this case, we have three morphisms **from** our limit L **to** the objects in $\mathcal{C}$, i.e., $\pi_1, \pi_2. \pi_3$, see the top part of Figure 10. Let M be another object (i.e., not one of the D_i) with morphisms $f_i: M \to D_i, i = 1,2,3$ (as shown in the figure). In terms of morphisms among the objects in $\mathcal{C}$, assume there is only one, i.e., $g: D_2 \to D_3$. To meet the compatible condition on the definition of a limit, we need to have $f_3 = g \circ f_2$. Finally, there needs to exist a morphism $\mu: M \to L$ such that $\pi_i \circ \mu$ applied to M points to the same object as f_i, i.e., the object D_i (as illustrated in the figure).

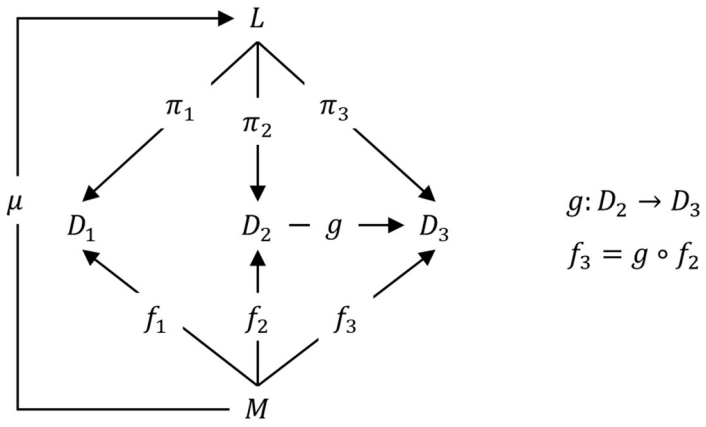


Figure 10. Limit diagram example

Common Types of Limits (Specific Instances): The following examples illustrate how the abstract definition of "limit" applies to typical constructions, forming the foundation for more advanced SDG tools like pullbacks.

- **Products**: For a diagram of two objects A and B with no morphisms between them, the limit is the product $A \times B$ with projections $\pi_1: A \times B \to A$, $\pi_2: A \times B \to B$. In the category Set, this is the Cartesian product of sets.
- **Equalizers**: For a diagram of parallel morphisms $f, g: A \to B$, the limit is the equalizer E with $e: E \to A$ such that $f \circ e = g \circ e$, selecting the "kernel" where f and g agree. In the category Set, $E = \{a \in A: f(a) = g(a)\}$.
- **Pullbacks**: For a diagram $f: A \to C$, $g: B \to C$, the limit is the pullback $P = A \times_C B$ with projections $p_1: P \to A$, $p_2: P \to B$, and $f \circ p_1 = g \circ p_2$, as shown in Figure 11. In the category Set, the limit P is the set of pairs (a, b) where $f(a) = g(b)$.
    - $A \times B$ (the product) is the collection of all possible pairs (a, b). $A \times_C B$ (the pullback) is the collection of all pairs (a, b) that agree when mapped into C. The condition $f(a) = g(b)$ is the agreement or matching condition. The object C acts as a reference or gluing point. The pullback is "A and B glued together along C."
    - Example in Set: Consider sets $A = \{1,2\}$, $B = \{2,3\}$, with $f: A \to \{2\}$ (constant to 2), $g: B \to \{2\}$ (constant to 2). The pullback P is the set of pairs (a, b) where $f(a) = g(b)$. Since both f and g are constant maps to $\{2\}$, this condition is always true. Thus, P is simply the full Cartesian product $\{(1,2), (1,3), (2,2), (2,3)\}$. This glues A and B over their common image in $\{2\}$.

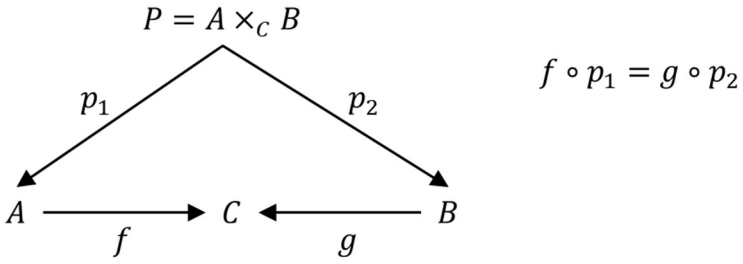


Figure 11. Pullback diagram

11.2.3 Colimits: Gluing Objects Together

Colimits are the dual (opposite) of limits. While limits pull back or combine objects via universal properties (e.g., products as greatest lower bounds), colimits push forward or glue objects together (e.g., quotients or unions).

- A colimit of a diagram in a category $\mathcal{C}$ is an object L together with a collection of morphisms $\gamma_i: D_i \to L$ (called the injections or coprojections) for each object in the diagram, such that:
    - Compatibility: For every morphism $g: D_i \to D_j$ in the diagram, we have $\gamma_j \circ g = \gamma_i$. (This ensures the diagram commutes when mapped into L).
    - Universal Property: For any object M in $\mathcal{C}$ with morphisms $f_i: D_i \to M$ that are similarly compatible (i.e., $f_j \circ g = f_i$ for every $g: D_i \to D_j$ in the diagram), there exists a unique morphism $\mu: L \to M$ such that $\mu \circ \gamma_i = f_i$ for all i.
- Intuition: Colimits merge objects along morphisms, identifying elements as specified. They are least upper bounds in the category's order.
- Dual to Limits: Limits use cones (arrows into the limit); colimits use cocones (arrows out of the colimit).

As an example of the definition of a colimit, consider a category $\mathcal{C}$ and a diagram in $\mathcal{C}$ with three objects, i.e., D_1, D_2, D_3. In this case, we have three morphisms **to** our limit L **from** the objects in the diagram, i.e., $\gamma_1, \gamma_2, \gamma_3$, see the bottom part of Figure 12. Let M be another object (also not in the diagram) with morphisms $f_k: D_k \to M, k = 1,2,3$ (as shown in the figure). In terms of morphisms among the objects in the diagram, assume there is only one, i.e., $g: D_1 \to D_2$. To meet the compatible condition on the definition of colimit, we need to have $f_1 = f_2 \circ g$. For the colimit L to be valid, the morphisms from the diagram to L (γ_k) must also satisfy the same condition, i.e., $\gamma_1 = \gamma_2 \circ g$. Finally, there needs to exist a morphism $\mu: L \to M$ such that $\mu \circ \gamma_k$ applied to D_k points to the same object as f_k, i.e., M (as illustrated in the figure).

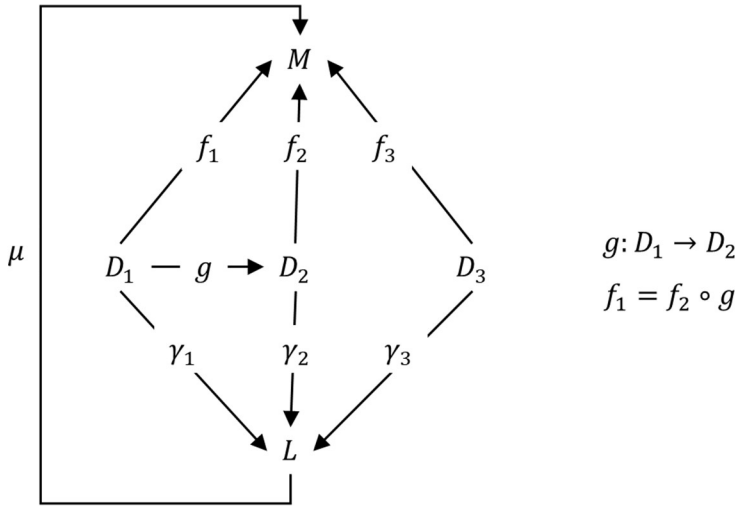


Figure 12. Colimit diagram example

Colimits are essential in SDG for constructing microlinear spaces[14], where infinitesimal diagrams are glued into colimits, modeling smooth structures like **manifolds**[15]. (Microlinear spaces are the SDG equivalent of manifolds.)

These concepts, subobjects for parts and colimits for gluings, provide the tools for the rich structure of toposes, setting the stage for SDG's infinitesimal calculus.

11.3 Classic Real Numbers versus The Smooth Real Line

A foundational clarification is necessary before we proceed. Synthetic Differential Geometry (SDG) constructs a universe for calculus that is intuitively smooth. The cornerstone of this universe is a "real line" that is richer than the classical one (which we have been denoting as $\mathbb{R}$). To avoid confusion, we will maintain a strict notational and conceptual distinction:

- $\mathbb{R}$ **(The Classical Real Numbers):** This denotes the set of real numbers as defined in classical logic and set theory (e.g., via Dedekind cuts or Cauchy sequences). In this world, the statement $d^2 = 0 \implies d = 0$ is a theorem, which logically forbids the existence of non-zero infinitesimals.
- $\mathcal{R}$ **(The Smooth Real Line):** This is the fundamental ring object of SDG. It is **not** a field in the classical sense. Its most important property, enforced by the Kock-Lawvere axiom (discussed below), is that it contains non-zero nilpotent infinitesimals. The subobject $D = \{d \in \mathcal{R} : d^2 = 0\} \subset \mathcal{R}$ is non-trivial.

Crucially, $\mathcal{R}$ is a single, fixed object that contains an entire hierarchy of infinitesimal spaces of different orders (such as $D_2 = \{d \in \mathcal{R} : d^3 = 0\}$, etc.). All functions, derivatives, and geometric constructions in this synthetic framework are built upon this smooth real line $\mathcal{R}$.

Throughout this text, the symbol $\mathcal{R}$ will exclusively refer to this smooth real line, the stage on which our synthetic calculus is performed, while $\mathbb{R}$ will refer exclusively to the classical real numbers.

11.4 Toposes: The Categorical Foundation

SDG is typically developed in a **topos**, a category that generalizes the category of sets (Set) while providing tools for logic, geometry, and infinitesimals. A topos is a Cartesian closed category with additional structure, including finite limits, colimits, and a subobject classifier.

Key Properties:

- Cartesian Closed: Has products $A \times B$ and exponentials B^A, allowing internal function spaces. This is an essential requirement for doing logic and geometry internally. Having exponentials B^A means the "space of functions from A to B" itself exists as an object inside the topos. This allows us to define and manipulate functions as first-class citizens, which is necessary for calculus.

[14] The concept of microlinearity essentially formalizes the idea that a space "looks like" a linear space at an infinitely small scale.

[15] A manifold is a topological space that locally resembles Euclidean space near each point. More formally, an n-dimensional manifold (or n-manifold) is a second-countable Hausdorff topological space M such that every point in M has a neighborhood that is homeomorphic to an open subset of $\mathbb{R}^n$ (the n-dimensional Euclidean space).

- Subobject Classifier Ω: An object that classifies subobjects (generalized subsets), with a morphism true: $1 \to \Omega$ (terminal to truth values). This is the engine of internal logic. It generalizes the idea of a set of truth values from the classical $\{\text{true}, \text{false}\}$ to something more nuanced. In the internal logic of a topos, we don't just reason about whether something is “true” or “false,” but rather “to what extent” it is true, with the “truth value” being an element of Ω.
- Finite Limits: Enables constructions like pullbacks (fibered products), which generalize intersections and allow spaces to be built from smaller pieces. (This property, along with a terminal object, also implies the existence of products $A \times B$).

Toposes support an internal intuitionistic logic, where “truth” is multi-valued (via Ω), making them suitable for infinitesimals that don't exist in classical Set but can exist in a topos. This is a critical point. The logic in a general topos is intuitionistic or constructive. This means the law of excluded middle (for any proposition P, either P is true or not-P is true) does not necessarily hold. You cannot prove things by contradiction in general.

- Why this matters for SDG: Intuitionistic logic is a weaker logic than classical logic. By working in this weaker system, we can consistently add new axioms (like the existence of infinitesimals) that would lead to contradictions (like $1 = 0$) in classical logic. The Kock-Lawvere axiom of SDG is a prime example of such a non-classical axiom that is consistent within the intuitionistic framework of a suitable topos.

Example: Set as a Topos

- Objects: Sets.
- Morphisms: Functions.
- Products: Cartesian products.
- Exponentials: Function sets B^A.
- Subobject Classifier: $\Omega = \{0,1\}$, where $0 = \text{false}, 1 = \text{true}$, classifying subsets via characteristic functions.

Set is the canonical example of a topos, but it’s a very “classical” one. Its subobject classifier is the two-element set $\{0, 1\}$, and its internal logic is standard classical Boolean logic. Consequently, Set is not the topos in which SDG is developed.

However, Set lacks nilpotent infinitesimals (non-zero d with $d^2 = 0$, as that would require zero divisors, contradicting field properties). This is the entire motivation for using a topos other than Set. In particular,

- In the classical real numbers $\mathbb{R}$ (or any field), if $d^2 = 0$ and $d \neq 0$, we could divide both sides by d to get $d = 0$, a contradiction.
- SDG gets around this by working in a topos where the real line $\mathcal{R}$ is not a field in the classical sense, but a ring that contains nilpotent elements. A fundamental axiom of SDG (the Kock-Lawvere axiom) states that for any function $f: D \to \mathcal{R}$ (where $D = \{d \in \mathcal{R}: d^2 = 0\}$), there exists a unique $b \in \mathcal{R}$ such that $f(d) = f(0) + b * d$ for all $d \in D$.
- This means the graph of f on D is literally a straight line. The slope b of this line is, by definition, the derivative $f'(0)$. This provides a simple, algebraic, and geometric foundation for calculus where the derivative is a genuine ratio of infinitesimals.

In summary,

1. The classical universe of Set is insufficient because it cannot host infinitesimals without contradiction.
2. A topos provides such a universe, with its own internal logic and geometry.
3. The internal logic of a general topos is intuitionistic, which is more flexible and can consistently accommodate axioms that are false in Set.
4. We explicitly choose a topos (e.g., the smooth topos built using the "Cahiers topos" or "Dubuc topos") where the Kock-Lawvere axiom is true (discussed in the following section).
5. In this specific topos, the object R (the smooth real line) contains non-zero nilpotent infinitesimals, allowing for a direct and intuitive definition of derivatives and differential forms.

11.5 The Kock-Lawvere Axiom

The Kock-Lawvere Axiom is the defining feature of SDG, providing a categorical axiom that guarantees the existence of nilpotent infinitesimals and ensures that smooth functions behave linearly in infinitesimal neighborhoods. This axiom allows for a synthetic (intuitive, non-analytic) approach to derivatives, where calculus is performed directly with infinitesimals rather than through limits (as defined in calculus). It draws inspiration from the algebraic nilpotency in dual numbers but embeds it within the topos structure, making it compatible with the category's internal logic.

To make this accessible, we'll first state and explain the basic axiom, with intuition and examples, before briefly touching on a generalization for higher-order infinitesimals.

11.5.1 The Basic Kock-Lawvere Axiom

The axiom postulates the existence of a specific infinitesimal object in the topos and enforces linearity on maps from that object.

- **Statement:** In a topos T with a real line object $\mathcal{R}$, there exists a non-trivial object $D = \{d \in \mathcal{R} : d^2 = 0\}$. Moreover, for every morphism $g : D \to \mathcal{R}$, there exist unique elements $a, b \in \mathcal{R}$ such that $g(d) = a + b \cdot d$ for all $d \in D$.
- **Intuition:** The object D represents first-order infinitesimals, i.e., tiny displacements where the square d^2 is indistinguishable from zero. The axiom states that any function $g : D \to \mathcal{R}$ from this infinitesimal object to the real line is exactly a straight line, i.e., a constant term a (the value at 0) plus a slope term $b \cdot d$. This linearity is not assumed but enforced by the topos's structure.
- **A Radical Departure:** This axiom is incompatible with classical logic. In any classical field (such as $\mathbb{R}$), $d^2 = 0$ implies $d = 0$. SDG works in a different logical universe (a topos) where such infinitesimal objects can exist without contradiction.
- **Immediate Consequence – The Derivative:** For any function $f : \mathcal{R} \to \mathcal{R}$, consider $g(d) = f(x + d)$. By the axiom, $f(x + d) = f(x) + f'(x) \cdot d$ for a unique $f'(x) \in \mathcal{R}$. The derivative is thus defined directly as the slope of the infinitesimal linear approximation, bypassing the limit process.

- **Why "Non-Trivial"?:** D must contain more than just 0. In the intended models, while defined internally as $\{d \in \mathcal{R}: d^2 = 0\}$, D is externally a rich object that provides a genuine "direction" for differentiation.

The Kock-Lawvere axiom is a condition we impose on a topos T and its real line object $\mathcal{R}$. We are not just observing that D exists; we are demanding that it exists and that the "axiom of microlinearity" (every map from D to $\mathcal{R}$ is linear) holds.

Example: Confirming Linearity

Consider the function $f(x) = x^2 + 2x$. To find its derivative at zero using the axiom, we examine its behavior on the infinitesimal object D by defining $g(d) = f(d) = d^2 + 2d$.

For any $d \in D$, the nilpotency condition $d^2 = 0$ simplifies this to $g(d) = 2d$.

The Kock-Lawvere axiom mandates that g must be uniquely expressed as $a + b \cdot d$. Comparing this with $g(d) = 2d$, we see that $a = 0$ and $b = 2$. This coefficient b is, by definition, the derivative $f'(0)$, confirming the expected result.

11.5.2 Generalized Axiom

The basic Kock-Lawvere axiom, which uses the object $D = \{d \in \mathcal{R}: d^2 = 0\}$, is just the first in a hierarchy of axioms that empower Synthetic Differential Geometry (SDG). The core idea indeed extends to higher-order infinitesimals, leading naturally to a synthetic theory of Taylor series and higher derivatives.

The Formal Framework: Weil Algebras

For completeness, the axiom generalizes systematically using Weil algebras (see Section 9). A Weil algebra is a finite-dimensional commutative R-algebra with a nilpotent ideal. The prototypical example is the ring of dual numbers $\mathcal{R}[\varepsilon] \,/\, \varepsilon^2$, which corresponds to the basic axiom. The generalization is $\mathcal{R}[\varepsilon] \,/\, \varepsilon^{n+1}$, where the ideal $I = (\varepsilon)$ satisfies $I^{n+1} = \{0\}$. This nilpotency ensures that all elements behave as polynomials of degree at most n. The ring $\mathcal{R}[\varepsilon] \,/\, \varepsilon^{n+1}$ is a Weil algebra, and the object D_n is its "spectrum" in the topos.

The corresponding generalized Kock-Lawvere axiom states that for any Weil algebra W, the space of functions $\mathcal{R}^W$ (the object of all morphisms from the infinitesimal space defined by W to $\mathcal{R}$) is isomorphic to $\mathcal{R}^m$ for some m. This guarantees that any function from this infinitesimal space to $\mathcal{R}$ is a unique polynomial, with the maximum degree determined by the nilpotency index of W.

A central and intuitive case is the following formal statement:

Generalized Kock-Lawvere Axiom: Let $D_n = \{d \in \mathcal{R}: d^{n+1} = 0\}$. For every morphism $g: D_n \to \mathcal{R}$, there exist unique elements $a_0, a_1, \dots, a_n \in R$such that:

$$g(d) = a_0 + a_1 d + a_2 d^2 + \cdots + a_n d^n \text{ for all } d \in D_n.$$

This means that the function space $\mathcal{R}^{D_n}$ (all maps from D_n to $\mathcal{R}$) is isomorphic to $\mathcal{R}^{n+1}$, where each function is uniquely determined by its "Taylor coefficients" $a_0, a_1, \dots, a_n$.

Example: Finding the 1st and 2nd Derivatives with D_2

Let's use this to find the first and second derivative of a function at a point. We'll work with $D_2 = \{d \in \mathcal{R}: d^3 = 0\}$.

Step 1: Define the Function and its Scaled Version

Consider the function $f(x) = x^3 + 2x^2 - 5x + 1$. We want to find its derivatives at $x = 0$.

Following the same logic as the previous example, we define a function g on the infinitesimal object D_2 by shifting f:

$$g(d) = f(0 + d) = f(d) = d^3 + 2d^2 - 5d + 1$$

Step 2: Apply the Nilpotency Condition

Since $d \in D_2$, we know $d^3 = 0$. This simplifies our expression for g:

$$g(d) = (0) + 2d^2 - 5d + 1 = 1 - 5d + 2d^2$$

Step 3: Apply the Generalized Axiom

The generalized Kock-Lawvere axiom for $n = 2$ tells us that $g(d)$ must be a unique polynomial of degree at most 2:

$$g(d) = a_0 + a_1 d + a_2 d^2 \text{ for all } d \in D_2.$$

Step 4: Identify the Coefficients (The Synthetic Taylor Coefficients)

We now have two expressions for $g(d)$:

1. From the axiom: $g(d) = a_0 + a_1 d + a_2 d^2$
2. From our calculation: $g(d) = 1 + (-5)d + (2)d^2$

The uniqueness clause of the axiom forces these coefficients to be identical:

$$a_0 = 1, a_1 = -5, a_2 = 2$$

Step 5: Interpret the Coefficients as Derivatives

This is the crucial step that creates the synthetic Taylor series. In standard calculus, the Taylor expansion of f around 0 is:

$$f(0 + d) = f(0) + f'(0)d + \frac{f''(0)}{2!}d^2 + \frac{f'''(0)}{3!}d^3 + \cdots$$

In our synthetic world, since $d^3 = 0$ for $d \in D_2$, the expansion *within this infinitesimal neighborhood* stops at the d^2 term. Comparing the synthetic form $g(d) = a_0 + a_1 d + a_2 d^2$ with the classical Taylor series, we are led to the following **definitions**:

- The constant term a_0 is the value of the function at the point: $a_0 = f(0)$.
- The linear coefficient a_1 is the first derivative: $a_1 = f'(0)$.
- The quadratic coefficient a_2 is *half* of the second derivative: $a_2 = \frac{f''(0)}{2!}$.

Therefore, from our calculated coefficients:

- $f(0) = a_0 = 1$
- $f'(0) = a_1 = -5$
- $\frac{f''(0)}{2} = a_2 = 2$ which implies $f''(0) = 4$

Step 6: Verification with Classical Calculus

Let's verify this using standard differentiation:

$$f(x) = x^3 + 2x^2 - 5x + 1$$

$$f'(x) = 3x^2 + 4x - 5 \Rightarrow f'(0) = -5$$

$$f''(x) = 6x + 4 \Rightarrow f''(0) = 4$$

The results match perfectly.

Conclusion and the Link to Synthetic Taylor Series

This example demonstrates the power of the generalization. The axiom for D_n doesn't just give us the first derivative; it gives us the *first n derivatives* simultaneously.

The general synthetic Taylor's theorem becomes a simple definition: For any function $f: \mathcal{R} \to \mathcal{R}$ and any point $x \in \mathcal{R}$, the behavior of f on the infinitesimal neighborhood $x + D_n$ is given by:

$$f(x + d) = f(x) + f'(x)d + \frac{f''(x)}{2!}d^2 + \cdots + \frac{f^{(n)}(x)}{n!}d^n \text{ for all } d \in D_n$$

The derivatives $f'(x), f''(x), \ldots, f^{(n)}(x)$ are *defined* as the unique coefficients that make this equation true. This is a purely algebraic, limit-free foundation for calculus.

Why This Generalization Matters

This generalization is not merely a mathematical curiosity. It is essential for advanced concepts in SDG, such as the theory of jets (which represent higher-order approximations of functions) and connections (which define parallel transport on manifolds). It also creates a powerful link to numerical analysis concepts like hyper-dual numbers, used for exact second-derivative calculation.

However, for understanding the core philosophical and practical ideas of SDG—such as defining derivatives intuitively and constructing the tangent bundle—the basic first-order Kock-Lawvere axiom is both necessary and sufficient. The generalization completes the picture, showing that the entire edifice of differential calculus can be built synthetically upon these elegant algebraic foundations.

11.5.3 Implications and Proof Sketch

The Kock-Lawvere axiom has the following implications:

- **Universal Differentiability:** In the universe defined by SDG, every morphism $f: \mathcal{R} \to \mathcal{R}$ is automatically differentiable (in fact, smooth) everywhere. Its derivative f' is defined synthetically as the map that assigns to each x the unique slope b from the expansion

$f(x+d) = f(x) + b \cdot d$. This formulation effectively banishes the pathological, nowhere-differentiable functions of classical analysis from the smooth world of SDG.

- **Synthetic Calculus:** The need for epsilon-delta limit proofs is eliminated. Derivatives are computed through direct algebraic manipulation of nilpotent elements, much like working with dual numbers.
- **Compatibility with Logic:** The axiom is consistent within intuitionistic topos, which have a richer internal logic than classical set theory. This avoids the classical contradiction that would arise from a non-zero d with $d^2 = 0$ in a field like $\mathbb{R}$, by situating $\mathbb{R}$ in the ring $\mathcal{R}$ that can have nilpotents.

A brief proof sketch for uniqueness: The axiom postulates that for any $g: D \to \mathcal{R}$, there exists a linear representation. The goal of the proof is to show this representation is unique.

- **By the Axiom:** There exist some $a, b \in \mathcal{R}$ such that for all $d \in D$, $g(d) = a + b \cdot d$.
- **Uniqueness of *a*:** To isolate a, evaluate the function at $d = 0$. This gives $g(0) = a + b \cdot 0 = a$. Therefore, a is uniquely determined and must be $g(0)$.
- **Uniqueness of *b*:** Since D is non-trivial, we can consider a non-zero element $\delta \in D$ (where non-zero is understood in the internal logic of the topos). Evaluate the function at this point: $g(\delta) = a + b \cdot \delta$. Since we now know $a = g(0)$, we can rearrange to solve for b:

  $$b \cdot \delta = g(\delta) - g(0)$$

  A key property enforced by the axioms of SDG is that this equation has a unique solution for b, which can be written synthetically as

  $$b = \frac{g(\delta) - g(0)}{\delta}$$

Since both a and b are determined by the function g itself (via evaluation at 0 and δ), the linear representation $g(d) = a + b \cdot d$ is unique.

11.6 Infinitesimals in SDG

In SDG, infinitesimals are not numbers but objects like D, defined by nilpotency within the internal logic of a topos. This contrasts sharply with the invertible infinitesimals of Non-Standard Analysis (Hyperreals).

- **Nilpotent Infinitesimals:** The basic object is $D = \{d \in \mathcal{R} : d^2 = 0\}$. Like the ε in dual numbers, these are so small that their square is exactly zero. This property is enforced categorically by the Kock-Lawvere axiom, which guarantees that all functions are linear on D.
- **Higher-Order Infinitesimals:** Objects like $D_k = \{d \in \mathcal{R} : d^{k+1} = 0\}$ allow for synthetic Taylor series, where functions are exactly polynomials on these spaces.
- **Microlinear Spaces: Microlinear Spaces:** These are objects in the topos that behave like **manifolds**—smooth, finite-dimensional spaces that locally resemble the smooth real line $\mathcal{R}$—from the perspective of infinitesimals. Formally, a microlinear space is defined by how it interacts with infinitesimal shapes: it sends infinitesimal diagrams (built from objects like D) to colimit diagrams in the topos.

This technical property enables the synthetic definition of complex geometric concepts. For example, the **tangent bundle** (which, classically, is the collection of all tangent spaces – each being the space of possible velocities at a point) has an elegant synthetic definition. The tangent bundle of a microlinear space M is simply the space of all infinitesimal displacements, M^D. This allows for the definition of other structures, such as **connections**, which provide rules for parallel transport of vectors across the manifold.

Table 10. Infinitesimals Comparison

Aspect	**SDG Infinitesimals**	**Dual Numbers**	**Hyperreals**
Structure	Nilpotent (e.g., $d^2 = 0$)	Nilpotent ($\varepsilon^2 = 0$)	Invertible ($\frac{1}{\delta}$ infinite)
Framework	Topos	Ring	Field extension of $\mathbb{R}$
Method of Differentiation	Synthetic (Axiomatic)	Algebraic truncation	Standard part
Derivatives	$f'(x)$ is the unique b in $f(x+d) = f(x) + b \cdot d$	$f'(x) =$ coefficient of ε in $f(x+\varepsilon)$	$f'(x) = st\left(\frac{f(x+\delta) - f(x)}{\delta}\right)$

Key Contrast: The nilpotent structure of SDG infinitesimals makes derivatives a direct, algebraic consequence of the foundational axioms. In contrast, the invertible infinitesimals of the hyperreals require the application of an external standard part function (st) to collapse the infinitesimal difference quotient back to a real number.

11.7 Applications to Differentiation and Geometry

To illustrate how SDG applies these concepts, we'll start with basic differentiation and build to geometric structures. The goal is to show how SDG's synthetic approach simplifies proofs and computations, paving the way for categorical models of automatic differentiation, where derivatives are computed as functors or morphisms in a category.

11.7.1 Synthetic Derivatives

In SDG, derivatives are defined directly using the infinitesimal object D from the Kock-Lawvere axiom. For a map $f: \mathcal{R} \to \mathcal{R}$ (where $\mathcal{R}$ is the real line object), the derivative at $x \in \mathcal{R}$ is the unique map $f'(x): \mathcal{R} \to \mathcal{R}$ such that for all $d \in D$, the following holds:

$$f(x+d) = f(x) + f'(x) \cdot d$$

By the Kock-Lawvere axiom, any map from D to $\mathcal{R}$ is of the form $d \to a \cdot d$ for a unique $a \in \mathcal{R}$; here, that unique real number a is the derivative $f'(x)$.

Intuition: There is no need for the calculus definition of derivative, i.e., $\lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$ since the infinitesimal d makes the linear approximation exact on D, with higher-order terms vanishing due to the axiom $d^2 = 0$ for $d \in D$. It's important to note that $f'(x)$ itself is an element of $\mathcal{R}$, and the associated tangent map is the function $d \to f'(x) \cdot d$. It's important to note that for a fixed x, the value $f'(x)$ is an element of $\mathcal{R}$, and the associated tangent vector at x is the function $d \to f'(x) \cdot d$.

Example 1: The Chain Rule

For $f = g \circ h$, we have $f'(x) = g'\big(h(x)\big) \cdot h'(x)$. The synthetic proof is remarkably simple:

$$f(x + d) = g\big(h(x + d)\big)$$

$$= g(h(x) + h'(x) \cdot d) \quad \text{(by applying the derivative rule to } h\text{)}$$

$$= g\big(h(x)\big) + g'\big(h(x)\big) \cdot (h'(x) \cdot d) \quad \text{(by applying the derivative rule to } g\text{, and that } h'(x) \cdot d \text{ is itself a perfectly good infinitesimal in D)}$$

$$= f(x) + \big[g'\big(h(x)\big) \cdot h'(x)\big]d$$

By the uniqueness property of the derivative, the term in the brackets must be $f'(x)$. This proof is not only shorter than the classical $\epsilon - \delta$ version but also avoids the explicit manipulation of limits and generalizes elegantly to multivariable cases.

11.7.2 Geometric Structures: Tangent Bundles and Vector Fields

SDG models geometry using infinitesimal extensions. A key structure is the tangent bundle of a space M (a microlinear object representing a manifold), denoted TM. In SDG, TM is defined as the exponential object M^D, representing maps from D to M.

- **Intuition:** A tangent vector at $m \in M$ is an "infinitesimal curve" or a "displacement" rooted at m, i.e., a map $v: D \to M$ with $v(0) = m$. The bundle TM collects all such vectors over all points. The projection $\pi: TM \to M$ is given by $\pi(v) = v(0)$, recovering the base point.

Example 2: Tangent Vectors on the Real Line

For $M = \mathcal{R}$, we have $TM = \mathcal{R}^D$. A tangent vector at $x \in \mathcal{R}$ is a map $v: D \to \mathcal{R}$ with $v(0) = x$. By the Kock-Lawvere axiom, this must be of the form $v(d) = x + v_1 \cdot d$, where $v_1 \in \mathcal{R}$ is the velocity component. The projection is $\pi(v) = v(0) = x$.

A vector field is a section of the tangent bundle, i.e., a map $X: M \to TM$ such that $\pi \circ X = \mathrm{id}_M$. This means for every point m, $X(m)$ is a tangent vector at m. For instance, the constant field $X(x)(d) = x + 1 \cdot d$ has velocity 1 everywhere.

- **Connection to Automatic Differentiation**: This structure is the theoretical foundation for forward-mode automatic differentiation. A dual number (x, x') can be seen as representing a tangent vector $v(d) = x + x' \cdot d$. Evaluating a function f on this dual number automatically computes the function's value and its derivative applied to the tangent vector, effectively lifting the function to the tangent bundle.

11.7.3 Connections and Curvature

An affine connection on M is a structure that defines parallel transport of tangent vectors along curves. Synthetically, it can be represented as a map $\nabla: TM \times_M TM \to T(TM)$ that takes two tangent vectors with the same base point (a direction and a vector to be transported) and returns the derivative of the parallel transport. Curvature arises from the non-commutativity of parallel transport around an infinitesimal parallelogram.

- **Intuition:** A connection provides a rule for comparing vectors in different tangent spaces (i.e., for taking derivatives of vector fields). Curvature quantifies the path-dependence of this comparison. In the flat Euclidean space, the connection is trivial, and parallel transport is path independent.

Example 3: Flat Connection on $\mathcal{R}$

On the real line $\mathcal{R}$, the natural connection is flat. For vector fields X and Y, the covariant derivative $\nabla_X Y$ is simply the ordinary derivative of the component of Y in the direction of X. The curvature is zero.

The geometric machinery in SDG provides a profound foundation for optimization. In this view, gradients are tangent vectors (or covectors in the cotangent bundle), and Hessians can be understood as objects related to the curvature of the space, informing the behavior of second-order optimization algorithms.

11.8 Connections to Hyperreals and Duals

SDG offers a categorical synthesis of the infinitesimal methods from earlier sections. It provides a general framework in which the analytical infinitesimals of hyperreals and the algebraic nilpotents of dual numbers can be seen as special, realizable cases within specific toposes.

Connection to Dual Numbers: The fundamental infinitesimal object D in SDG behaves precisely like the set of nilpotent elements in the dual ring $\mathcal{R}[\varepsilon]/\varepsilon^2$. The Kock-Lawvere axiom enforces that every map $f: D \to \mathcal{R}$ is linear, i.e., of the form $f(d) = a \cdot d$, which is the categorical formulation of dual number differentiation. In fact, one of the most important models of SDG is the topos of convenient vector spaces, where the smooth real line object is represented by the ring of dual numbers. This connection highlights how SDG generalizes the algebraic engine of dual numbers by embedding it into a rich geometric context.

- **Example:** The first-order Taylor expansion $f(x + d) = f(x) + f'(x) \cdot d$ in SDG is operationally identical to evaluating a function on dual numbers. However, SDG's power lies in its ability to generalize this principle to define sophisticated geometric structures such tangent bundles $TM = M^D$, which is the foundation for multivariable and higher-order automatic differentiation.

Connection to Hyperreals: There is a conceptual kinship but a fundamental technical difference. Hyperreals $\mathbb{R}^*$ employ invertible infinitesimals δ (with infinitely small, non-zero magnitude), while SDG's D is explicitly nilpotent ($d^2 = 0$). However, both are approaches to making infinitesimals rigorous. SDG does not typically use the hyperreals themselves, but non-standard analysis (NSA) can be formulated within certain topoi constructed via ultrapower or similar techniques. In such a non-standard topos, the process of taking a standard part (shadow) can be seen as a kind of limit, allowing intuitive infinitesimal evaluations to replace epsilon-delta limit arguments.

- **Example:** In a topos modeling NSA, a statement such as $\lim_{h \to 0} g(h) = L$ is equivalent to $g(\delta)^* \approx L$ for all non-zero infinitesimals δ, where $g(\delta)^*$ is the hyperreal number that is infinitely close to the standard real number L. This mirrors the synthetic simplicity of SDG proofs but is achieved through a different, set-theoretically complex mechanism (the ultrafilter construction).

Overall, SDG serves as a unifying framework. It views infinitesimals as first-class logical objects in a topos. Dual numbers provide the canonical *algebraic and computational* model for the simplest (first-order) infinitesimals, while hyperreals and non-standard analysis provide a powerful *analytical intuition* based on a more intuitive concept of infinitely small but non-zero numbers. This synthesis motivates modern categorical automatic differentiation, where the derivative is fundamentally a functor on a tangent category, elegantly capturing the chain rule and its generalizations.

11.9 Examples

To ground these ideas, we'll walk through several examples, focusing on how SDG simplifies familiar calculus and geometry, with ties to automatic differentiation.

11.9.1 Example 1: The Fundamental Theorem of Calculus (FTC)

In SDG, the Fundamental Theorem of Calculus is not a deep theorem to be proven with limits, but a direct consequence of the logical structure. We start with the concept of an **antiderivative**.

Synthetic Formulation: Let $f: \mathcal{R} \to \mathcal{R}$ be a function. An **antiderivative** of function f is a function $F: \mathcal{R} \to \mathcal{R}$ whose derivative is f, i.e., a function that satisfies the following condition for all $x \in \mathcal{R}$ and $d \in D$:

$$F(x + d) = F(x) + f(x) \cdot d$$

This is not an approximate statement; it is the exact, defining property of the derivative $F' = f$ enforced by the Kock-Lawvere axiom.

Proof Sketch (of the FTC): The FTC states that antiderivatives define integration. In SDG, this is immediate from the definition above. Suppose F is an antiderivative of f.

1. The equation $F(x + d) = F(x) + f(x) \cdot d$ tells us that the change in F over an infinitesimal interval $[x, x + d]$ is exactly $f(x) \cdot d$.
2. This means that $f(x)$ is the instantaneous rate of change (the derivative) of F, and conversely, the total change in F is accumulated from these infinitesimal contributions of f.
3. Therefore, the function F already encapsulates the process of integrating f. If we fix a starting point a, the function $G(x) = F(x) - F(a)$ can be understood as the total accumulation of f from a to x.

This synthetic formulation elegantly unifies differentiation and integration: to integrate f, one simply finds a function whose derivative is f.

Connection to AD: This is the philosophical underpinning of forward-mode automatic differentiation. In AD, when you compute a function $F(x)$ and its derivative $f(x)$ simultaneously, you are effectively tracking both the total accumulation $F(x)$ and its infinitesimal rate of change $f(x)$. The process of computing $F(x)$ by following a sequence of operations is analogous to building up the finite integral from its infinitesimal parts.

11.9.2 Example 2: Vector Fields and Flow on the Circle

SDG provides a powerful framework for differential geometry by expressing classical concepts through infinitesimal constructions. Consider the circle S^1, defined as the microlinear space $\{(x, y) \in R^2 : x^2 + y^2 = 1\}$.

Vector Field: A vector field $X: S^1 \to TS^1$ assigns a tangent vector to each point. The tangent bundle TS^1 is defined synthetically as $(S^1)^D$, the space of all infinitesimal curves on the circle.

Tangent Vector Intuition: At a point $p = (\cos\theta, \sin\theta)$, a tangent vector must represent an infinitesimal displacement that stays on the circle. Classically, the tangent space is spanned by the vector $s = (-\sin\theta, \cos\theta)$. We can verify this is tangent because its dot product with the radius vector p is zero, i.e.,

$$p \circ s = (\cos\theta)(-\sin\theta) + (\sin\theta)(\cos\theta) = 0$$

Synthetic Vector Field Definition: A constant-speed vector field that rotates points around the circle is given synthetically by defining the new position after an infinitesimal time d:

$$X(p)(d) = (\cos(\theta + vd), \sin(\theta + vd))$$

Here, v is the angular velocity. This definition is geometrically intuitive, i.e., it simply rotates the point p by an infinitesimal angle vd.

Synthetic Simplification (The Key Step): Let's show that this definition conforms to the synthetic notion of a tangent vector. We use the trigonometric identities for cosine and sine of a sum, and critically apply the property that $d^2 = 0$ for $d \in D$, which eliminates all higher-order terms.

$$X(p)(d) = (\cos(\theta + vd), \sin(\theta + vd))$$

$$= (\cos\theta\cos(vd) - \sin\theta\sin(vd), \sin\theta\cos(vd) + \cos\theta\sin(vd))$$

Next, we use the synthetic Taylor series (which is exact on D) for $\cos(vd)$ and $\sin(vd)$:

- $\cos(vd) = 1 + \frac{(vd)^2}{2!} + \ldots = 1$ (since $d^2 = 0$ eliminates all higher terms)
- $\sin(vd) = (vd) - \frac{(vd)^3}{3!} + \ldots = vd$

Substituting these exact values gives:

$$X(p)(d) = \big(\cos\theta \cdot 1 - (\sin\theta)(vd), \sin\theta \cdot 1 + (\cos\theta)(vd)\big)$$

$$= (\cos\theta, \sin\theta) + (-(\sin\theta)vd, (\cos\theta)vd)$$

$$= p + vd(-\sin\theta, \cos\theta)$$

$$= p + (vd)s$$

This final form, $X(p)(d) = p + (vd)s$, is the standard synthetic form of a tangent vector at p. It clearly shows the base point p and the direction s, scaled by the infinitesimal d and the velocity v.

Flow: The flow of this vector field—the path a point follows over time—is computed infinitesimally.

- **Infinitesimal Update Rule:** If $\theta(t)$ is the angular position at time t, then for an infinitesimal time step d, the new position is given by the vector field itself. From the definition of X, the new angle is $\theta(t+d) = \theta(t) + vd$.
- **Finite Flow via Integration:** This infinitesimal update rule, $\theta(t+d) = \theta(t) + vd$, is a simple differential equation. "Patching together" these infinitesimal steps (a process formalized by integration in the synthetic sense) yields the finite flow: $\theta(t) = \theta^0 + vt$, where θ^0 is the initial angle at time $t = 0$. This describes uniform circular motion.

Connection to AD and Optimization: This synthetic framework provides the mathematical semantics for optimization on manifolds. Algorithms like Riemannian gradient descent work as follows:

1. At a point p on a constraint manifold (such as S^1), the gradient of a function is a tangent vector (a map $D \to M$).
2. An optimization step is computed by moving along this tangent vector, which is exactly applying its infinitesimal flow: $p \to p(d) = X(p)(d)$.
3. For the circle, this is a rotation by an infinitesimal angle. In practice, a finite step size is used, but the underlying geometric principle is this infinitesimal displacement. SDG thus elegantly captures the core mechanic of moving along a tangent space to remain on a manifold.

11.9.3 Example 3: The Symmetry of Second Derivatives (Clairaut's Theorem)

In classical calculus, Clairaut's theorem states that for a sufficiently smooth function $f: \mathbb{R}^2 \to \mathbb{R}$, the mixed partial derivatives are equal: $\frac{\partial^2 f}{\partial x \partial y} = \frac{\partial^2 f}{\partial y \partial x}$. In SDG, this is not a theorem to be proven with limits, but a trivial algebraic identity that follows directly from the definitions.

Synthetic Formulation: We define the second-order behavior of a function using two independent nilsquare infinitesimals, $d_1, d_2 \in D$. The object $D_1 \times D_2 = \{(d_1, d_2) : d_1^2 = 0, d_2^2 = 0\}$ allows us to probe how the function changes when perturbed in both directions.

The core of the proof lies in how we define the second partial derivatives. We define the coefficients A and B in the second-order expansion as the mixed partials:

1. **Definition (Mixed Partial $\frac{\partial^2 f}{\partial x \partial y}$):** We define the number $\frac{\partial^2 f}{\partial x \partial y}$ to be the unique coefficient such that:

$$f(x + d_1, y + d_2) = f(x, y) + \left(\frac{\partial f}{\partial x}\right) d_1 + \left(\frac{\partial f}{\partial y}\right) d_2 + \left(\frac{\partial^2 f}{\partial x \partial y}\right) d_1 d_2$$

2. **Definition (Mixed Partial $\frac{\partial^2 f}{\partial y \partial x}$):** Equivalently, we can define the number $\frac{\partial^2 f}{\partial y \partial x}$ to be the unique coefficient such that:

$$f(x + d_1, y + d_2) = f(x, y) + \left(\frac{\partial f}{\partial y}\right) d_2 + \left(\frac{\partial f}{\partial x}\right) d_1 + \left(\frac{\partial^2 f}{\partial y \partial x}\right) d_2 d_1$$

Proof of Equality: These are two expressions for the same quantity, $f(x + d_1, y + d_2)$. Since addition is commutative and $d_1 d_2 = d_2 d_1$, the two expansions must be identical. Comparing them:

$$f(x,y) + \left(\frac{\partial f}{\partial x}\right) d_1 + \left(\frac{\partial f}{\partial y}\right) d_2 + \left(\frac{\partial^2 f}{\partial x \partial y}\right) d_1 d_2 = f(x,y) + \left(\frac{\partial f}{\partial y}\right) d_2 + \left(\frac{\partial f}{\partial x}\right) d_1 + \left(\frac{\partial^2 f}{\partial y \partial x}\right) d_2 d_1$$

The terms $f(x,y)$, $\left(\frac{\partial f}{\partial x}\right) d_1$, and $\left(\frac{\partial f}{\partial y}\right) d_2$ cancel from both sides, leaving:

$$\left(\frac{\partial^2 f}{\partial x \partial y}\right) d_1 d_2 = \left(\frac{\partial^2 f}{\partial y \partial x}\right) d_1 d_2$$

By the uniqueness of the coefficients guaranteed by the Kock-Lawvere axiom on $D \times D$, we can equate the coefficients of $d_1 d_2$ to get:

$$\frac{\partial^2 f}{\partial x \partial y} = \frac{\partial^2 f}{\partial y \partial x}$$

Thus, the above equality is built into the very definition of what a second derivative is in the synthetic context.

Connection to AD: This symmetry is crucial for the efficiency of reverse-mode automatic differentiation (backpropagation). It ensures that the Hessian matrix is symmetric, which simplifies computations and is a key assumption in second-order optimization algorithms. SDG reveals this foundational property not as a hard-won theorem, but as an inherent feature of its infinitesimal algebra.

12 Automatic Differentiation: Principles and Modes

By reviewing the old, one learns the new. — Confucius, Analects

12.1 Introduction

Automatic differentiation (AD), also known as “auto diff” or “algorithmic differentiation,” is a computational technique for evaluating derivatives of functions defined by computer programs with machine precision accuracy and efficiency. Unlike numerical differentiation (which approximates via finite differences and suffers from truncation errors) or symbolic differentiation (which manipulates expressions and can be computationally expensive for complex programs), AD leverages the chain rule to compute exact derivatives by propagating sensitivities through the program’s structure. In the context of this book, AD draws directly from the infinitesimal frameworks explored earlier: dual numbers provide the algebraic foundation for forward-mode AD, while category theory and synthetic differential geometry offer abstract models for forward-mode and reverse-mode (or backward-mode) AD.

AD is indispensable in fields such as machine learning (e.g., backpropagation in neural networks), optimization, and scientific simulation, where gradients guide parameter updates. This section introduces AD's principles, the two primary modes (forward and reverse), and their connections to the book's themes, culminating in an application to Bayesian inference.

12.2 Principles of Automatic Differentiation

Automatic Differentiation breaks down complex functions into elementary operations (primitives) with pre-programmed derivatives and then applies the chain rule systematically throughout the computation. AD is not based on the symbolic manipulation of expressions (like computer algebra systems) nor on numerical approximations (like finite difference methods). Instead, it is a set of techniques that leverage the fact that any complex function, no matter how intricate, is ultimately composed of a sequence of elementary arithmetic operations and functions (e.g., addition, multiplication, exponentiation, trigonometric functions). AD works by breaking down the function into these constituents and systematically applying the chain rule of calculus.

The two core concepts that underpin all AD implementations are the computational graph and the elementary operation.

12.2.1 Computational Graphs

A computational graph is a directed acyclic graph (DAG) that represents the flow of computation for a given function. It provides a visual and structural blueprint that AD algorithms traverse. In this graph:

- Nodes represent variables. These can be:
    - Input variables (the independent variables of the function).
    - Intermediate variables (the results of elementary operations inside the function).
    - Output variables (the final dependent variables of the function).
- Edges represent the functional dependencies between variables. An edge from node u to node v indicates that the value of v depends on the value of u.

An alternative, but equally valid representation, is to have nodes representing the elementary operations themselves, with edges carrying the data.

Example 1: Consider the function $f(x, y) = (\sin x) + xy$. Its computational graph can be represented as shown in Figure 13 (which should be read from left to right). The nodes x and y are inputs. The node v_3 is an intermediate from the $\sin x$ operation, and v_4 is an intermediate from the $*$ operation. The node f is the output from the $+$ operation.

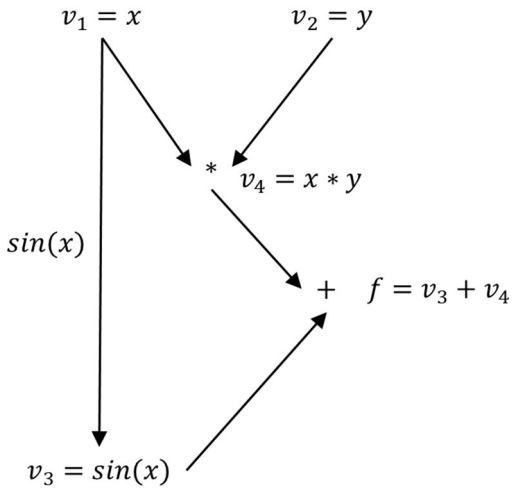


Figure 13. Computational graph for Example 1

12.2.2 Decomposition into Elementary Operations

The power of AD comes from decomposing a function into a list of elementary operations for which the derivatives are known (i.e., pre-programmed). For each elementary operation, we can compute both its value and its local derivative with minimal cost.

The process of AD involves a forward pass to compute the values of all nodes and then a propagation of derivative information through the graph. This propagation is governed by the chain rule.

In its simplest form, if a variable z depends on a variable y which in turn depends on a variable x, the chain rule states:

$$\frac{dz}{dx} = \frac{dz}{dy} \cdot \frac{dy}{dx}$$

In the context of a computational graph with multiple paths, this generalizes. If a variable u_i has multiple immediate successors, the total derivative $\frac{\partial f}{\partial u_i}$ is the sum of the derivative contributions flowing through each of these successors. This is calculated efficiently by summing over the immediate successors of u_i:

$$\frac{\partial f}{\partial u_i} = \sum_{j \in \text{successors}(u_i)} \frac{\partial f}{\partial v_j} \cdot \frac{\partial v_j}{\partial u_i}$$

This reverse accumulation formula, which sums contributions from a variable's successors, will be the basis for reverse-mode AD (Section 12.4). In contrast, forward-mode AD propagates derivatives from a variable to its successors.

Example 2: Let's trace a single derivative through the graph from Figure 13. Suppose we want $\frac{\partial f}{\partial x}$. There are two paths from x to f: one through v_3 and one through v_4. Applying the chain rule:

$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial v_3} \cdot \frac{\partial v_3}{\partial x} + \frac{\partial f}{\partial v_4} \cdot \frac{\partial v_4}{\partial x}$$

We know the local derivatives of the elementary operations:

- $\frac{\partial f}{\partial v_3} = 1$ (derivative of $v_3 + v_4$ w.r.t. v_3)
- $\frac{\partial v_3}{\partial x} = \cos(x)$ (derivative of $\sin(x)$)
- $\frac{\partial f}{\partial v_4} = 1$ (derivative of $v_3 + v_4$ w.r.t. v_4)
- $\frac{\partial v_4}{\partial x} = y$ (derivative of xy w.r.t. x)

Therefore,

$$\frac{\partial f}{\partial x} = (1 \cdot \cos(x)) + (1 \cdot y) = \cos(x) + y$$

This matches the result from symbolic differentiation.

By breaking the function into a graph of elementary operations, AD algorithms can compute exact derivatives by combining these known local derivatives via the chain rule. The specific order and direction in which this combination occurs define the two main modes of AD: forward and reverse.

12.3 Forward-Mode Automatic Differentiation

Forward-mode Automatic Differentiation (AD) is the most intuitive of the two main AD modes. It computes derivatives by propagating them forward through the computational graph, from inputs to outputs, simultaneously with the function evaluation. This mode is a direct computational realization of the algebra of dual numbers discussed in Section 6.

12.3.1 Theory: Tangents and the Connection to Dual Numbers

In forward-mode AD, every variable in the computation is augmented to not only hold its primal value (the value of the function itself) but also to hold its tangent value (the value of its derivative with respect to a chosen input variable). This pair $(v, \dot{v})$ is the computational analog of a dual number $v + \dot{v}\varepsilon$, where ε is the nilpotent element satisfying $\varepsilon^2 = 0$.

The process is seeded by choosing an input variable with respect to which we wish to differentiate. If we want the derivative with respect to x_i, we set its tangent $\dot{x}_i = 1$, and the tangents of all other independent inputs to 0. This single “seed vector” defines a **directional derivative**. As the computation proceeds, for every elementary operation, we simultaneously compute both the primal value and the tangent value of the result.

The rules for propagating tangents are derived directly from the local derivative of each elementary operation and the chain rule. We can view the entire computation as a function of a single,

underlying independent variable t. The tangents $\dot{x}, \dot{y}$, and $\dot{z}$ then represent the derivatives $\frac{dx}{dt}, \frac{dy}{dt}$, and $\frac{dz}{dt}$ respectively.

Given an operation $z = f(x, y)$:

- The primal computation is performed as normal: $z = f(x, y)$.
- The tangent computation applies the chain rule: $\dot{z} = \left(\frac{\partial f}{\partial x}\right)\dot{x} + \left(\frac{\partial f}{\partial y}\right)\dot{y}$.

For the specific elementary operation of multiplication, where $f(x, y) = xy$, the above becomes:

- Primal: $z = xy$
- Tangent: $\dot{z} = y\dot{x} + x\dot{y}$

This exact calculation is what dual number multiplication performs automatically:

$$(x + \dot{x}\varepsilon)(y + \dot{y}\varepsilon) = xy + (x\dot{y} + \dot{x}y)\varepsilon$$

Each elementary operation in the computational graph has a corresponding dual number operation that simultaneously computes both the primal value and the correct tangent value according to the chain rule.

12.3.2 The Forward-Mode AD Algorithm

The algorithm proceeds by traversing the computational graph in a topological order, evaluating each node. For each node v_k that is the result of an operation $v_k = g\left(v_{i_1}, \dots, v_{i_r}\right)$ where $v_{i_1}, \dots, v_{i_r}$ are the immediate predecessors of v_k in the computational graph.

1. **Primal Update:** $v_k = g\left(v_{i_1}, \dots, v_{i_r}\right)$
2. **Tangent Update:** $\dot{v}_k = \frac{\partial g}{\partial v_{i_1}} \cdot \dot{v}_{i_1} + \cdots + \frac{\partial g}{\partial v_{i_r}} \cdot \dot{v}_{i_r}$

All tangent values $\dot{v}_{i_1}, \dots, \dot{v}_{i_r}$ represent derivatives with respect to the same underlying independent variable, determined by the initial seed values.

The final output will be a pair (f, f') representing the function value and its derivative in the chosen direction.

12.3.3 Step-by-Step Example

Let us compute the value and derivative of the function $f(x) = \sin(x) + x^2$ at the point $x = 2$. We seed the tangent for the input: $(x, \dot{x}) = (2,1)$. The computational graph (Figure 14) and trace are shown below.

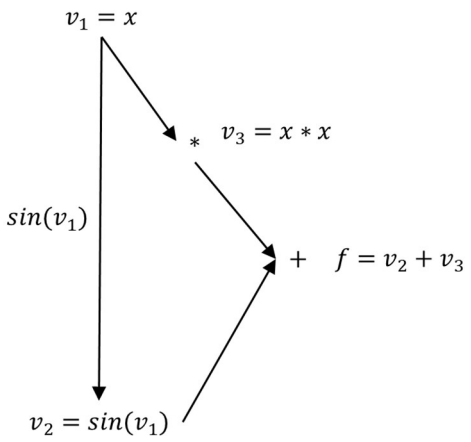


Figure 14. Computation graph for f(x) = sin(x) + x^2

Primal Trace (Value Computation):

1. $v_1 = x = 2$
2. $v_2 = \sin(v_1) = \sin(2) \approx 0.90930$
3. $v_3 = v_1 \cdot v_1 = 2 \cdot 2 = 4$
4. $f = v_2 + v_3 \approx 0.90930 + 4 = 4.90930$

Tangent Trace (Derivative Computation):

1. $\dot{v}_1 = \dot{x} = 1$ *(Seed)*
2. $\dot{v}_2 = \cos(v_1) \cdot \dot{v}_1 = \cos(2) \cdot 1 \approx -0.41615 \cdot 1 = -0.41615$ *(Chain rule)*
3. $\dot{v}_3 = v_1 \cdot \dot{v}_1 + \dot{v}_1 \cdot v_1 = 2 \cdot 1 + 1 \cdot 2 = 4$ *(Product rule)*
4. $f' = \dot{v}_2 + \dot{v}_3 \approx -0.41615 + 4 = 3.58385$

Result: The function value is $f(2) \approx 4.90930$and its derivative is $f'(2) \approx 3.58385$.

The following table presents this dual evaluation in a compact, easy-to-follow format, mirroring the execution of a program that uses operator overloading for dual numbers.

Table 11. Forward-mode AD trace

#	Elementary Operation	Primal Value (v)	Tangent Value ($\dot{v}$)	Derivative Rule Applied
-	Input Seed	$x = 2$	$\dot{x} = 1$	-
1	$v_1 = x$	2	1	-
2	$v_2 = \sin(v_1)$	$\sin(2) \approx 0.90930$	$\cos(v_1) \cdot \dot{v}_1 \approx -0.41615$	$\frac{d}{dx}\sin(x) = \cos(x)$ and Chain Rule
3	$v_3 = v_1 \cdot v_1$	4	$v_1 \cdot \dot{v}_1 + \dot{v}_1 \cdot v_1 = 4$	Product Rule
4	$f = v_2 + v_3$	4.90930	$\dot{v}_2 + \dot{v}_3 \approx 3.58385$	$\frac{d}{dx}(a + b) = a' + b'$

12.3.4 Complexity and Use Cases

The computational cost of a single forward-mode AD evaluation (computing the function value and derivatives in one chosen direction) is proportional to the cost of the original function evaluation, typically by a small constant factor (2-5 times). For a function $f: \mathbb{R}^n \to \mathbb{R}^m$, computing the full Jacobian matrix requires n separate forward-mode evaluations, one for each independent input direction, resulting in an overall complexity of $O(n)$ times the cost of a single function evaluation.

This makes forward-mode AD highly efficient when the number of inputs n is small. It is the method of choice for applications like:

- **Sensitivity analysis**, where one analyzes how a single output is influenced by a small number of inputs.
- **Tangent-linear models** in numerical weather prediction.
- The internal computation within a single step of reverse-mode AD when applied to a function with a single output (as the "forward sweep" is effectively a forward-mode AD pass).

In the next subsection, we will explore reverse-mode AD, which provides a complementary efficiency profile, excelling when the number of outputs m is small.

12.4 Reverse-Mode Automatic Differentiation (Backpropagation)

While forward-mode AD is intuitive, reverse-mode Automatic Differentiation is often dramatically more efficient for the most common computational scenarios, particularly in machine learning and optimization. Instead of propagating derivatives from inputs to outputs, reverse-mode AD performs a forward pass to compute the function value and then a backward pass to propagate derivatives from the output back to the inputs. This algorithm is famously known as backpropagation in the context of training neural networks.

12.4.1 Theory: Adjoints and the Reverse Pass

The core idea of reverse-mode AD is to compute the gradient of a single output with respect to all inputs simultaneously. It does this by introducing an **adjoint** for each variable. The adjoint of a variable v, denoted $\bar{v}$, represents the partial derivative of the final output f with respect to that variable, i.e., $\bar{v} = \frac{\partial f}{\partial v}$.

The process consists of two sweeps through the computational graph:

1. **Forward Pass:** The graph is evaluated from inputs to output, exactly as in a standard function evaluation. The key difference is that all intermediate variable values must be stored in memory, as they will be needed during the backward pass.
2. **Backward Pass:** The graph is traversed in reverse, from the output back to the inputs. Starting from the output ($\bar{f} = 1$), we use the stored intermediate values and the chain rule to compute the adjoints of each preceding variable.

The chain rule in reverse mode is applied at each node. If a variable v is used to compute several other variables $v_1, v_2, \ldots, v_k$ (its successors in the graph), then its adjoint is accumulated from all of them:

$$\bar{v} = \sum_{j \in \text{successors}(v)} \bar{v}_j \cdot \frac{\partial v_j}{\partial v} = \sum_{j \in \text{successors}(v)} \frac{\partial f}{\partial v_j} \cdot \frac{\partial v_j}{\partial v}$$

This rule is the heart of the backpropagation algorithm. It ensures that the derivative information flows backwards from the output, efficiently aggregating the contributions of each variable to the final result.

12.4.2 Why the Backward Pass Works

The Graph as a Computational Recipe

Think of the computation as a directed acyclic graph. Each node is an elementary operation (e.g., +, *, sin, exp). The key is that we know the *local derivative* for each of these operations.

For example, if at a node, we compute $c = a + b$, then we know $\frac{\partial c}{\partial a} = 1$ and $\frac{\partial c}{\partial b} = 1$.

If at a node, we compute $c = a \cdot b$, then we know $\frac{\partial c}{\partial a} = b$ and $\frac{\partial c}{\partial b} = a$.

The Chain Rule in Multivariable Calculus

The standard chain rule for a path $f \to v_j \to v$ is:

$$\frac{\partial f}{\partial v} = \frac{\partial f}{\partial v_j} \cdot \frac{\partial v_j}{\partial v}$$

However, a variable v can be used in multiple places in the graph. For instance, a variable a might be used to compute c_1 and c_2, which both eventually influence f.

The total derivative of f with respect to a must account for **all paths** from a to f. The multivariable chain rule tells us to sum over all these paths:

$$\frac{\partial f}{\partial a} = \sum_{\text{path } p} \left[\frac{\partial f}{\partial a}\right]_p = \frac{\partial f}{\partial c_1} \cdot \frac{\partial c_1}{\partial a} + \frac{\partial f}{\partial c_2} \cdot \frac{\partial c_2}{\partial a}$$

Translating to the Adjoint Formulation

This is exactly what the adjoint accumulation equation captures (referring to the summation in the previous subsection). Let's rephrase it:

If a variable v is used to compute several other variables $v_1, v_2, \dots, v_k$ (its successors in the graph), then its adjoint is accumulated from all of them:

$$\bar{v} = \frac{\partial f}{\partial v} = \sum_{j \in \text{successors}(v)} \bar{v}_j \cdot \frac{\partial v_j}{\partial v} = \sum_{j \in \text{successors}(v)} \frac{\partial f}{\partial v_j} \cdot \frac{\partial v_j}{\partial v}$$

Why is this true?

- $\bar{v}_j = \frac{\partial f}{\partial v_j}$ is the "global" derivative, representing how much f changes when v_j changes.
- $\frac{\partial v_j}{\partial v}$ is the "local" derivative, representing how much v_j changes when v changes.
- The product $\bar{v}_j \cdot \frac{\partial v_j}{\partial v}$ is the contribution of the path $f \to v_j \to v$ to the total derivative $\frac{\partial f}{\partial v}$.
- The summation $\sum_j \bar{v}_j \cdot \frac{\partial v_j}{\partial v}$ ensures we aggregate the contributions from *all* paths leading from v to f via its immediate successors.

Therefore, this equation is a direct implementation of the multivariable chain rule applied to the computational graph.

The Algorithm in Action: Why It's Efficient

The algorithm works backward not just because of the chain rule, but for computational efficiency.

1. **Initialization:** Start at the output. $\bar{f} = 1$.
2. **Recursive Application:** For each node whose adjoint $\bar{v}_j$ has been computed, it "donates" its share of the derivative to each of its parent nodes v. This is the "local" application of the chain rule.
3. **Accumulation:** When a node v receives a "donation" from a successor v_j, it adds it to its own adjoint $\bar{v}$. This is the summation $\sum_j \bar{v}_j \cdot \frac{\partial v_j}{\partial v}$.

This process ensures that each edge in the graph is traversed exactly once in the backward direction. The cost of computing the gradient of a scalar function f with respect to all inputs is only a small constant factor (typically $3 - 5$ times) more expensive than computing the function f itself. This is far more efficient than finite differences, which requires $O(n)$ evaluations for n inputs.

12.4.3 The Reverse-Mode AD Algorithm

1. **Forward Pass:**
    - Traverse the graph in topological order.
    - For each node v_i, compute and store its primal value v_i.
2. **Backward Pass:**
    - Initialize the output adjoint: $\bar{f} = 1$.
    - Traverse the graph in reverse topological order.
    - For each node v_i, if it was computed as $v_i = g$ (parents), then for each parent u of v_i, update its adjoint:

$$\overline{u} \mathrel{+}\leftarrow \overline{v}_i \cdot \frac{\partial v_i}{\partial u}$$

The use of $+\leftarrow$ (add-and-assign) is crucial, as a single parent variable u may be used in multiple child nodes, and its total adjoint is the sum of its contributions to all of them.

12.4.4 Step-by-Step Example

Let us compute the gradient of the function $f(x, y) = \sin(x) + xy$ at the point $(x, y) = \left(\frac{\pi}{2}, 3\right)$. We break it down into elementary operations following the computational graph in Figure 13.

Table 12. Forward Pass (Compute and Store Primal Values)

#	Variable	Elementary Operation	Value Computed & Stored
1	v_1	$v_1 = x$	$\frac{\pi}{2} \approx 1.5708$
2	v_2	$v_2 = y$	3
3	v_3	$v_3 = \sin(v_1)$	$\sin(1.5708) = 1$
4	v_4	$v_4 = v_1 \cdot v_2$	$1.5708 \times 3 \approx 4.7124$
5	f	$f = v_3 + v_4$	$1 + 4.7124 = 5.7124$

For the backward pass, we initialize $\bar{f} = 1$ and then process each node in reverse order.

Table 13. Backward Pass (Compute Adjoints $\bar{v} = \frac{\partial f}{\partial v}$)

#	Variable	Adjoint Calculation & Explanation	Adjoint Value $\bar{v}$
5	$\bar{f}$	**Initialize output.** $\bar{f} = 1$	1
4	$\bar{v}_3, \bar{v}_4$	**Process** $f = v_3 + v_4$ $\bar{v}_3 \leftarrow \bar{f} \cdot \frac{\partial f}{\partial v_3} = \bar{f} \cdot 1 = 1 \cdot 1 = 1$ $\bar{v}_4 \leftarrow \bar{f} \cdot \frac{\partial f}{\partial v_4} = \bar{f} \cdot 1 = 1 \cdot 1 = 1$	$\bar{v}_3 = 1$ $\bar{v}_4 = 1$
3	$\bar{v}_1$	**Process** $v_3 = \sin(v_1)$ $\bar{v}_1 \leftarrow \bar{v}_3 \cdot \frac{\partial v_3}{\partial v_1} = \frac{\partial f}{\partial v_3} \cdot \frac{\partial v_3}{\partial v_1} = 1 \cdot \cos(v_1) = \cos(1.5708) \approx 0$	$\bar{v}_1 = 0$
2	$\bar{v}_1, \bar{v}_2$	**Process** $v_4 = v_1 \cdot v_2$ $\bar{v}_1 \leftarrow \bar{v}_4 \cdot \frac{\partial v_4}{\partial v_1} = \frac{\partial f}{\partial v_4} \cdot \frac{\partial v_4}{\partial v_1} = 1 \cdot v_2 = 1 \cdot 3 = 3$ $\bar{v}_2 \leftarrow \bar{v}_4 \cdot \frac{\partial v_4}{\partial v_2} = \frac{\partial f}{\partial v_4} \cdot \frac{\partial v_4}{\partial v_2} = 1 \cdot v_1 \approx 1.5708$	Accumulate to existing $\bar{v}_1$, i.e., $\bar{v}_1 = 0 + 3 = 3$ $\bar{v}_2 = 1.5708$
1	$\bar{x}, \bar{y}$	**Process Inputs** $v_1 = x, v_2 = y$ The inputs are the roots of the graph. Their adjoints are the final gradients. $\bar{x} = \bar{v}_1 = 3$ $\bar{y} = \bar{v}_2 \approx 1.5708$	$\bar{x} = 3$ $\bar{y} = 1.5708$

Result: The function value is $f\left(\frac{\pi}{2,3}\right) \approx 5.7124$and its gradient is:

$$\nabla f = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right) = (\bar{x}, \bar{y}) = (3, 1.5708)$$

This matches the analytical gradient $\nabla f = (\cos(x) + y, x)$ evaluated at $\left(\frac{\pi}{2,3}\right)$.

12.4.5 Complexity and Use Cases

The computational cost of the forward and backward passes together is only a small constant factor (typically $2 - 5$ times) more expensive than the original function evaluation because each elementary operation in the forward pass has a corresponding, similarly cheap, adjoint update operation in the backward pass. This is true *regardless of the number of inputs*.

For a function $f: \mathbb{R}^n \to \mathbb{R}^m$, computing the full Jacobian matrix requires m separate evaluations of the reverse-mode algorithm, one for each output.

This makes reverse-mode AD exceptionally efficient when the number of outputs m is small, especially when $m = 1$. This is the defining characteristic of many critical applications:

- **Machine Learning:** Training neural networks via backpropagation. The loss function $L(\theta)$ is a scalar ($m = 1$) that depends on millions of parameters θ (n is very large). Reverse-mode AD computes the entire gradient ∇L in a time comparable to a few function evaluations.
- **Numerical Optimization:** Most nonlinear optimization algorithms (e.g., gradient descent, conjugate gradient) require the gradient of a scalar objective function with respect to a high-dimensional input vector.
- **Sensitivity Analysis:** Analyzing how a single, critical output depends on a vast number of input parameters.

The primary trade-off for this efficiency is **memory**. Reverse-mode AD must store the values of all intermediate variables from the forward pass for use in the backward pass. For very large computations, this can lead to significant memory pressure, leading to research into techniques like **checkpointing** (recomputing some intermediate values instead of storing them) to trade computation for memory.

12.5 Practical Guidance: Choosing Between Forward and Reverse Mode

The choice between forward-mode and reverse-mode AD is not a matter of correctness, both compute exact derivatives, but of computational efficiency. The decision hinges on the relative dimensions of the function's input and output spaces. This section provides practical matrix-based examples to illustrate this critical distinction.

12.5.1 Case Study 1: The Jacobian-Vector Product (Forward-Mode Domain)

Consider a function $f: \mathbb{R}^n \to \mathbb{R}^m$ and a fixed vector $v \in \mathbb{R}^n$. Our goal is not to compute the full $m \times n$ Jacobian matrix J, but rather the Jacobian-vector product Jv.

Example: Sensitivity of a Multi-output System to a Single Input Direction

Let $f(x) = Ax$, where A is a 100×500 matrix ($m = 100, n = 500$), and v is a specific direction in the input space (e.g., a known perturbation vector). We want $Jv = Av$.

- **Forward-Mode Approach:**
  We seed the tangent of the input as $\dot{x} = v$. The forward pass then computes:
  - Primal: $y = Ax$
  - Tangent: $\dot{y} = A\dot{x} = Av$

    This requires a single pass of forward-mode AD. The cost is similar to one function evaluation.
- **Reverse-Mode Approach:**
  Computing $J\boldsymbol{v}$ with reverse-mode is highly inefficient, since it would require initializing the output adjoint, propagating backwards, and would effectively involve computing the entire Jacobian $J = A$ internally, only to then multiply it by v. The computational cost would be proportional to the number of outputs, $m = 100$, making it about 100 times more expensive than the forward-mode approach for this specific task.

Analytically, since the Jacobian of $f(x) = Ax$ is simply $J = A$, the Jacobian-vector product is indeed $Jv = Av$. Forward-mode AD computes this directly without explicitly constructing the full Jacobian matrix.

Conclusion: Forward-mode AD is optimal for tasks involving Jacobian-vector products, where the number of input directions (vectors v) is small. This is common in tangent-linear models and certain sensitivity analyses.

12.5.2 Case Study 2: The Vector-Jacobian Product (Reverse-Mode Domain)

Now consider the dual problem. For the same function $f: \mathbb{R}^n \to \mathbb{R}^m$ and a vector $u \in \mathbb{R}^m$, our goal is to compute the vector-Jacobian product $u^T J$.

Example: The Gradient of a Scalar Function (The Most Important Case)

Let $L(\theta): \mathbb{R}^{1,000,000} \to \mathbb{R}$ be a scalar loss function in a machine learning model with one million input parameters ($n = 10^6, m = 1$). We want the gradient $\nabla L(\theta) = \left(\frac{\partial L}{\partial \theta_1}, \dots, \frac{\partial L}{\partial \theta_{10^6}}\right)^T$, which is exactly the vector-Jacobian product $u^T J_L(\theta)$ where $u^T = 1$ and $J_L(\theta) = \left(\frac{\partial L}{\partial \theta_1}, \dots, \frac{\partial L}{\partial \theta_{10^6}}\right)$.

- **Reverse-Mode Approach (Backpropagation):**
  We seed the output adjoint as $\bar{L} = 1$. The single backward pass propagates this scalar back through the entire computation, efficiently accumulating the gradient with respect to every one of the million parameters. The cost is only a small constant factor (e.g., $2 - 5$ times) more expensive than a single evaluation of $L(\theta)$.
- **Forward-Mode Approach:**
  To compute the full gradient ∇L using forward-mode, we would need to perform $n = 1{,}000{,}000$ separate forward passes. In each pass i, we would set $\dot{\theta}_i = 1$ and all other tangents to zero, to compute the single partial derivative $\frac{\partial L}{\partial \theta_i}$. This would be over a million times more expensive than the reverse-mode approach.

Analytically, for a scalar function $L: \mathbb{R}^n \to \mathbb{R}$, the Jacobian J_L is a $1 \times n$ row vector equal to the transpose of the gradient ∇L. The vector-Jacobian product $u^T J_L$ with $u^T = 1$ thus yields the row vector J_L, which is equivalent to the gradient ∇L as a column vector. Reverse-mode AD computes this entire gradient efficiently in a single backward pass.

The analytical perspectives, in this case study and the previous one, reveal why the efficiency trade-offs are so stark: forward-mode naturally computes **Jacobian-vector products** Jv while reverse-mode naturally computes **vector-Jacobian products** $u^T J$; operations that are dual to each other, just as the two modes are.

Conclusion: Reverse-mode AD is optimal for tasks involving vector-Jacobian products, most notably when computing the gradient of a scalar-valued function with respect to a large number of inputs. This is the foundational operation in machine learning and large-scale optimization.

12.5.3 Summary and Decision Matrix

The following table summarizes the guiding principle for choosing an AD mode. Let $f: \mathbb{R}^n \to \mathbb{R}^m$.

Task	Mathematical Operation	Number of Passes Required	Recommended Mode
Compute gradient of a scalar function	$\nabla f, m = 1$	1 reverse pass	Reverse-Mode
Compute full Jacobian ($n \ll m$)	$J \in \mathbb{R}^{m \times n}$	n forward passes	Forward-Mode
Compute full Jacobian ($m \ll n$)	$J \in \mathbb{R}^{m \times n}$	m reverse passes	Reverse-Mode
Compute Jacobian-vector product	$J\mathrm{v}$	1 forward pass	Forward-Mode
Compute vector-Jacobian product	$\mathrm{u}^T J$	1 reverse pass	Reverse-Mode

Practical Advice: Most modern AD frameworks (like JAX and PyTorch) provide functions for both JVP (Jacobian-Vector Product) and VJP (Vector-Jacobian Product). The user can then compose these primitives efficiently based on their specific problem structure, often letting the library make the optimal choice automatically.

12.6 Implementations and Efficiency

The theoretical elegance of Automatic Differentiation is matched by its practical implementation. AD is not a single algorithm but a technique that can be realized through different software strategies, each with distinct trade-offs. Furthermore, the choice between forward and reverse mode introduces a fundamental tension between computational speed and memory usage.

12.6.1 Implementation Techniques

There are two primary methods for implementing AD: Operator Overloading and Source Transformation.

***** Operator Overloading *****

This is the most common approach for implementing AD in dynamic, high-level languages. The core idea is to define a new custom data type (e.g., DualNumber for forward-mode, Tensor with a computational graph for reverse-mode) and to **overload** the elementary operations (like +, *, sin, exp) to work on this new type.

How it works:

- The user writes a function using standard code and operators.
- The inputs are "wrapped" as instances of the custom AD type.
- When operations are performed on these wrapped variables, the overloaded functions are called. These functions simultaneously compute the primal value and record the derivative

information (e.g., by building a computational graph in memory for reverse-mode or by propagating tangents for forward-mode).

Example (Forward-Mode in Python):

```
# ============================================================================
# (1) FUNCTION UNDER CONSIDERATION:
# We are computing the function f(x) = x * x (i.e., f(x) = x²)
# and its derivative f'(x) = 2x simultaneously using forward-mode AD.
# ============================================================================

# ============================================================================
# (2) REDEFINED/OVERLOADED OPERATIONS:
# We define a custom DualNumber class that OVERLOADS basic arithmetic operations
# to work with both primal values and tangent values simultaneously.
# For simplicity, it is assumed that all operands are DualNumbers.
# ============================================================================

class DualNumber:
    def __init__(self, value, tangent):
        # Store both primal value (function value) and tangent (derivative)
        self.value = value    # Primal: f(x)
        self.tangent = tangent  # Tangent: f'(x)

    # ========================================================================
    # OVERLOADED MULTIPLICATION OPERATOR:
    # Redefines what * means when used with DualNumber objects
    # ========================================================================
    def __mul__(self, other):
        # Extract primal values
        x = self.value
        y = other.value

        # Extract tangent values
        x_dot = self.tangent
        y_dot = other.tangent

        # ====================================================================
        # SIMULTANEOUS COMPUTATION:
        # - Primal: z = x * y (normal function evaluation)
        # - Tangent: ż = x·ẏ + ẋ·y (product rule for derivatives)
```

```
        # ========================================================================
        primal_result = x * y
        tangent_result = x * y_dot + x_dot * y

        return DualNumber(primal_result, tangent_result)

    # Note: Similar overloads would be defined for __add__, __sub__, etc.
    # and for elementary functions like sin(), exp(), log()

# ============================================================================
# USER CODE DEMONSTRATION:
# The user writes code that looks completely normal, but gets derivatives for free
# thanks to the overloaded operations.
# ============================================================================

# Evaluate f(x) = x * x at x = 3, with derivative seeding
# The DualNumber(3.0, 1.0) means: evaluate at x=3, and compute derivative w.r.t. x
x = DualNumber(3.0, 1.0)  # primal = 3.0, tangent = 1.0 (seed for ∂x/∂x = 1)

# This line looks like normal multiplication, but uses our overloaded __mul__
result = x * x  # Actually calls DualNumber.__mul__(x, x)

# The result contains both f(3) and f'(3)
print(f"f(3) = {result.value}, f'(3) = {result.tangent}")
# Output: f(3) = 9, f'(3) = 6
```

Advantages:

- Ease of Use: Requires minimal changes to user code. Often, just changing the input type is sufficient.
- Flexibility: Well-suited for dynamic control flow (if-statements, loops) as the execution path is determined at runtime.

Disadvantages:

- Performance Overhead: The constant creation of new objects and function calls can introduce runtime overhead compared to statically compiled code.
- Memory Usage (for Reverse-Mode): The graph built during the forward pass can consume significant memory.

***** Source Transformation *****

This is a more complex but often higher-performance approach. A source transformation tool (a compiler or preprocessor) analyzes the source code of the original function and generates new, derived source code that explicitly computes the desired derivatives.

How it works:

- The user provides the source code for a function f.
- The AD tool parses this code, constructs an internal representation (like an abstract syntax tree), and then applies the chain rule to generate new code for a function $\mathrm{f}_{\mathrm{grad}}$ that computes both the value and the gradient.
- This generated code is then compiled and executed, often with performance very close to that of hand-written derivative code.

Advantages:

- High Performance: Eliminates the overhead of operator overloading and can be heavily optimized by the compiler.
- Explicit Control: The generated code is often readable and can be integrated into larger projects.

Disadvantages:

- Complexity: The implementation of the AD tool itself is complex.
- Limited Flexibility: Can struggle with highly dynamic or reflective language features where the execution path is not known until runtime.

Common Libraries:

- Operator Overloading: PyTorch and JAX (Python) use this for reverse-mode. The stan-math library (C++) uses it for forward- and reverse-mode.
- Source Transformation: The tool Tapenade (for Fortran and C) is a classic example. The ML compiler XLA (Accelerated Linear Algebra), used by JAX and TensorFlow, performs source-like transformations on a computational graph.

12.6.2 Memory and Computational Trade-offs

The choice of AD mode has profound implications for resource usage, forming a classic time-memory trade-off.

Forward-Mode AD Trade-offs:

- Memory: Efficient. It requires storing only the current primal and tangent values, similar to the original function evaluation. The memory footprint is $O(1)$ with respect to the number of operations.
- Computation: The cost of computing the full $n \times m$ Jacobian scales with the number of inputs n. It is efficient for $n \ll m$.

Reverse-Mode AD Trade-offs:

- Memory: High. This is its primary disadvantage. The entire tape (the record of all intermediate operations, their inputs, and their values from the forward pass) must be stored in memory for the backward pass. The memory footprint is $O(P)$, where P is the number of operations in the forward pass. For very deep computations (e.g., large neural networks), this can be a severe bottleneck.
- Computation: The cost of computing the full Jacobian scales with the number of outputs m. It is exceptionally efficient for $m \ll n$, especially when $m = 1$.

Mitigating the Memory Cost of Reverse-Mode: Checkpointing

Checkpointing (or rematerialization) is a crucial technique to trade computation for memory in reverse-mode AD. Instead of storing the entire tape, the forward pass is divided into segments or "checkpoints."

- The system only stores the values at the boundaries of these segments.
- During the backward pass, when the adjoints for a segment are needed, the forward pass for that segment is recomputed from the last checkpoint.
- This significantly reduces memory usage at the cost of roughly doubling the computational effort for the recomputed segments.

Checkpointing is essential for training very large models that would otherwise exceed the available memory of hardware accelerators like GPUs.

In summary, the implementation of AD involves a choice between the flexibility of operator overloading and the performance of source transformation. Simultaneously, the user must navigate the fundamental trade-off between the computational efficiency of reverse-mode and its high memory demand, often employing strategies like checkpointing to make large-scale problems tractable.

12.7 A Categorical Perspective on Automatic Differentiation

The preceding sections detailed the mechanics of Automatic Differentiation as a set of practical algorithms. However, a deeper, more unifying understanding emerges when we view AD through the lens of category theory. This perspective reveals that forward-mode and reverse-mode AD are not merely clever computational tricks but are natural manifestations of functorial and algebraic structures. This framework, often called Differentiable Programming**,** provides a formal foundation that generalizes beyond calculus on Euclidean spaces.

12.7.1 Forward-Mode AD as a Functor

The essence of forward-mode AD is the augmentation of every number with its derivative, transforming a function into its derivative-augmented version. Category theory captures this precisely using the concept of a **functor**.

Consider a category where:

- Objects are Euclidean spaces $(\mathbb{R}^n)$.
- Morphisms are smooth functions $(f: \mathbb{R}^n \to \mathbb{R}^m)$.

We can define a tangent functor T that acts as follows:

- On Objects: It sends a space $\mathbb{R}^n$ to its tangent bundle, $T(\mathbb{R}^n) = \mathbb{R}^n \times \mathbb{R}^n$. An element of $T(\mathbb{R}^n)$ is a pair $(x, \dot{x})$, representing a point and a tangent vector (the primal and the tangent from Section 11.3).
- On Morphisms: It sends a smooth function $f: \mathbb{R}^n \to \mathbb{R}^m$ to its pushforward (or total derivative), $Tf: T(\mathbb{R}^n) \to T(\mathbb{R}^m)$. This is defined by:

  $$Tf(x, \dot{x}) = \big(f(x), J_f(x)\ \dot{x}\big)$$

  where $J_f(x)$is the Jacobian of f at x. This is exactly the operation performed by forward-mode AD: given a point and an input direction, it returns the output value and the output direction.

This construction T is a functor because it preserves identity morphisms and composition. The preservation of composition, $T(g \circ f) = Tg \circ Tf$, is the functorial chain rule, guaranteeing that AD will correctly compute the derivative of a composed function by composing the derivatives of its parts.

This functor is a computational realization of the algebra of dual numbers (Section 6). The tangent bundle $T(\mathbb{R}^n)$ is isomorphic to $\frac{\mathbb{R}^n[\varepsilon]}{\varepsilon^2}$, and the action of Tf is equivalent to evaluating f on dual numbers.

Concrete Example: Consider the function $f(x, y) = \sin(x)\, e^y$. The tangent functor T acts as follows:

- **On the object** $\mathbb{R}^2$: $T(\mathbb{R}^2) = \mathbb{R}^2 \times \mathbb{R}^2$, representing points (x, y) with tangents $(\dot{x}, \dot{y})$
- **On the morphism** f: $Tf\big((x, y), (\dot{x}, \dot{y})\big) = \big(f(x, y),\ J_f(x, y) \cdot [\dot{x}\ \ \dot{y}]^T\big)$

Where the Jacobian is:

$$J_f(x, y) = \begin{bmatrix} \frac{\partial f}{\partial x} & \frac{\partial f}{\partial y} \end{bmatrix} = [\cos(x)\, e^y \quad \sin(x)\, e^y\ ]$$

So, for input $\big((x, y), (\dot{x}, \dot{y})\big) = \big((0, 0), (1, 2)\big)$:

$$Tf\big((0, 0), (1, 2)\big) = (\sin(0)\, e^0, [\cos(0)\, e^0 \quad \sin(0)\, e^0\ ] \cdot [\ 1\ \ 2]^T)$$

$$= (0, [1 \quad 0][1\ \ 2]^T) = (0,1)$$

This matches forward-mode AD: we get the function value $f(0,0) = 0$ and directional derivative 1 in direction $(1, 2)$.

12.7.2 Reverse-Mode AD and the Reverse Derivative Category

Reverse-mode AD is more subtle to capture categorically because it involves a "backward pass." The key insight is to model it not as a functor from functions to functions, but as a higher-order operation that provides a generalized transpose.

This is formalized using the concept of a Reverse Derivative Category (RDC). An RDC is a category equipped with a structure that for every morphism $f: A \to B$, there is an associated reverse derivative morphism $\mathcal{R}[f]: A \times B \to A$.

- Given an input $x \in A$ and a cotangent (or adjoint) $\bar{y} \in B$, the reverse derivative $\mathcal{R}[f]$ produces the corresponding input cotangent $\bar{x} \in A$.
- In the category of smooth functions between Euclidean spaces, this is defined as:

$$\mathcal{R}[f](x, \bar{y}) = (x, J_f(x)^T \cdot \bar{y})$$

This is precisely the core operation of the backward pass in reverse-mode AD (Section 12.4): at a point x, and given the adjoint of the output $\bar{y}$, it computes the contribution to the adjoint of the input $\bar{x}$.

The axioms of a reverse derivative category ensure that this operation behaves correctly with respect to composition (the chain rule for reverse-mode), products (handling multiple inputs), and other structural rules. This abstract formulation shows that reverse-mode AD is not an ad-hoc algorithm but a natural, composable primitive in a suitably defined categorical structure.

Concrete Example: For the same function $f(x, y) = \sin(x)e^y$, the reverse derivative morphism is:

$$R[f]\left((x, y), \hat{f}\right) = \left((x, y), J_f(x, y)^T \cdot \hat{f}\right)$$

where

$$J_f(x, y) = \begin{bmatrix} \cos(x)\, e^y \\ \sin(x)\, e^y \end{bmatrix}$$

So, for input $\left((x, y), \hat{f}\right) = \big((0, 0), 1\big)$:

$$\begin{aligned} R[f]\big((0, 0), 1\big) &= \big((0, 0), (\cos(0)\, e^0, \sin(0)\, e^0) \cdot 1\big) \\ &= \big((0, 0), (1,0) \cdot 1\big) \\ &= \big((0, 0), (1, 0)\big) \end{aligned}$$

This is exactly reverse-mode AD (backpropagation): starting with output adjoint $\hat{f} = 1$, we compute the gradient $\nabla f = (1, 0)$ at point $(0, 0)$.

Composition Example: Consider $h = g \circ f$ where $f(x, y) = \sin(x)\, e^y$ and $g(z) = z^2$. The reverse derivative composition works as:

$$R[h]\left((x, y), \hat{h}\right) = R[f]\left((x, y), R[g]\big(f(x, y), \hat{h}\big)\right)$$

We have that $R[g]\big(f(x, y), \hat{h}\big) = R[g]\big(z, \hat{h}\big) = \big(z, 2z \cdot \hat{h}\big)$ where $z = f(x, y)$.

At $\big((0, 0), 1\big)$:

- First, $R[g](f(0,0), 1) = R[g](0, 1) = (0, 2 \cdot 0 \cdot 1) = (0, 0)$
- Then, $R[f]\big((0, 0), 0\big) = \big((0, 0), (0, 0)\big)$

This gives a gradient of value zero, which is correct since $h(0,0) = \big(f(0,0)\big)^2 = 0^2 = 0$ has a zero valued derivative.

These concrete examples show exactly how the abstract categorical definitions correspond to computational AD operations.

12.7.3 Unification and Generalization

This categorical perspective provides a powerful unifying framework:

1. **Synthetic Differential Geometry (SDG):** The connection to SDG (Section 11) becomes clear. The tangent functor T and the notion of infinitesimals in SDG are deeply related. In SDG, the space of infinitesimal tangents at a point is a fundamental object. Forward-mode AD can be seen as the computational interpretation of applying a function to an element of this infinitesimal space, $x + \dot{x}d$, where d is a nilpotent infinitesimal satisfying $d^2 = 0$.
2. **Generalization Beyond $\mathbb{R}^n$:** The categorical definitions are not tied to Euclidean spaces. The same structures can be defined for categories of functions on manifolds, in discrete calculus, or even in logical and probabilistic models. This allows the principles of AD to be applied to a vast range of problems, from optimizing over quantum circuits to Bayesian inference in probabilistic programming languages, all under the umbrella of "differentiable programming."
3. **Correctness by Construction:** When an AD system is built on these categorical foundations, the correctness of its derivatives—specifically, the adherence to the chain rule—is guaranteed by the functorial laws or the RDC axioms. The implementation becomes a matter of providing concrete instances for elementary operations.

In conclusion, category theory elevates Automatic Differentiation from a numerical technique to a fundamental mathematical principle. It reveals that differentiation is a structurally inherent property of many computational worlds, and AD is the faithful, algorithmic realization of that structure. This bridges the intuitive, infinitesimal-based calculus of Leibniz with the rigorous, compositional framework of modern computer science and mathematics.

12.8 Bayesian Inference with Automatic Differentiation

Bayesian inference provides a coherent framework for updating beliefs in the presence of uncertainty. However, for complex models with high-dimensional parameters, the required computations are often analytically intractable and numerically challenging. The integration of Automatic Differentiation has been a revolutionary advance, making previously infeasible Bayesian analyses routine. This synergy is the engine behind modern Probabilistic Programming Languages (PPLs) like Stan, PyMC, and TensorFlow Probability.

12.8.1 The Core Computational Problem in Bayesian Inference

The goal of Bayesian inference is to compute the posterior distribution of model parameters θ given observed data y. Bayes' theorem states that

$$P(\theta \mid y) = \frac{P(y \mid \theta) \cdot P(\theta)}{P(y)} \propto P(y \mid \theta) \cdot P(\theta)$$

where

- $P(\theta)$ is the prior distribution,
- $P(\mathrm{y} \mid \theta)$ is the likelihood function,
- $P(\theta \mid \mathrm{y})$ is the posterior distribution
- $P(y) = \int P(y \mid \theta) P(\theta) d\theta$ is the marginal likelihood (evidence)

- The proportionality symbol ∝ means “equal up to a multiplicative constant”

The central computational challenge is to characterize the posterior distribution. For all but the simplest models, we cannot compute the posterior analytically. Instead, we rely on numerical methods, primarily Markov Chain Monte Carlo (MCMC), to draw samples from the posterior.

12.8.2 The Role of Gradients: Hamiltonian Monte Carlo

While simple MCMC algorithms like Random-Walk Metropolis exist, they are notoriously inefficient in high-dimensional parameter spaces, as they explore the posterior distribution via random, undirected steps. Hamiltonian Monte Carlo (HMC) and its adaptive variant, the No-U-Turn Sampler (NUTS), solve this problem by using gradient information to propose distant, high-acceptance moves that efficiently explore the posterior.

HMC borrows an analogy from physics. It introduces an auxiliary “momentum” variable p and conceptualizes the negative log-posterior as a potential energy surface:

$$U(\theta) = -\log[P(y \mid \theta)P(\theta)]$$

The system's total energy (the Hamiltonian) is $H(\theta, p) = U(\theta) + K(p)$, where $K(p)$ is the kinetic energy. HMC generates proposals by simulating the Hamiltonian dynamics of this system:

$$\frac{d\theta}{dt} = \frac{\partial H}{\partial p} = M^{-1}p, \qquad \frac{dp}{dt} = -\frac{\partial H}{\partial \theta} = -\nabla U(\theta)$$

where M is a mass matrix.

The critical term here is $\nabla U(\theta) = -\nabla \log[P(y \mid \theta)P(\theta)]$. This is the gradient of the log unnormalized posterior density. An accurate and efficient calculation of this gradient is non-negotiable for HMC to be practical.

12.8.3 How Automatic Differentiation Enables Modern Bayesian Inference

This is where Automatic Differentiation becomes indispensable. The log unnormalized posterior, $\log P(y \mid \theta) + \log P(\theta)$, is a scalar-valued function of a potentially very high-dimensional parameter vector θ (n can be in the thousands or millions).

- **The Perfect Fit for Reverse-Mode AD:** As established in Section 12.6, reverse-mode AD is optimally efficient for computing the gradient of a scalar function with respect to many inputs. A single backward pass of reverse-mode AD through the computation of the log-posterior density yields the exact gradient $\nabla U(\theta)$ for all parameters simultaneously, at a cost only a small constant multiple of the cost of evaluating the log-posterior itself.
- **Enabling Complex Models:** Before the widespread use of AD, practitioners were limited to models with conjugate priors or simple enough likelihoods for which gradients could be derived by hand. AD removes this restriction. A user can now define a model by simply writing code for the log-likelihood and log-prior, no matter how complex—involving loops, conditionals, and complex transformations. The PPL automatically computes the necessary gradients via AD, making HMC and NUTS applicable.
- **Accuracy and Stability:** Unlike numerical finite-difference methods, which are prone to truncation and round-off errors, AD provides gradients that are exact up to machine precision. This accuracy is crucial for the stable and correct simulation of Hamiltonian dynamics, where errors in the gradient can lead to poor sampling or divergent transitions.

Workflow in a Modern Probabilistic Programming Language (e.g., Stan):

1. The user specifies a Bayesian model by defining the log-prior and log-likelihood functions in a high-level modeling language.
2. The PPL compiler, using operator overloading or source transformation, generates C++ code that computes the log-posterior density and, crucially, its gradient via reverse-mode AD.
3. The HMC/NUTS sampler uses this generated code. For every "leapfrog" step in its Hamiltonian dynamics simulation, it calls the AD-generated function to obtain the exact gradient, enabling efficient exploration of the posterior.
4. The result is a set of samples from the posterior distribution $P(\theta \mid y)$ that can be used for estimation, prediction, and uncertainty quantification.

In summary, Automatic Differentiation is not merely an optimization for Bayesian computation; it is a foundational enabling technology. By providing efficient and exact gradients for arbitrary model specifications, it has unlocked the practical use of powerful gradient-based MCMC samplers like HMC, thereby revolutionizing the scope and scale of models that can be tackled within a Bayesian framework.

12.9 Exercises

The following exercises are designed to solidify your understanding of Automatic Differentiation principles, modes, and applications. They range from manual calculations that mirror the AD process to analytical questions about complexity and connections to other mathematical concepts.

12.9.1 Fundamental Practice

Exercise 1: Forward-Mode AD Trace
Consider the function $f(x) = \log(1 + e^{-x})$, a common softplus-like function used in machine learning.

a) Draw the computational graph for $f(x)$.

b) Manually perform a forward-mode AD trace to compute $f(0)$ and $f'(0)$. Use the dual number approach, showing both the primal and tangent value for each intermediate variable. A suggested decomposition is $v_1 = -x, v_2 = e^{v_1}, v_3 = 1 + v_2, f = \log(v_3)$.

Exercise 2. Reverse-Mode AD (Backpropagation) Trace

Consider the function $f(w_1, w_2) = \frac{w_1 w_2}{\max(w_1, w_2)}$, evaluated at the point $(w_1, w_2) = (3,2)$. This function resembles a simplified, unnormalized attention score.

a) Draw the computational graph. (*Hint:* The max operation is piecewise linear; its derivative is 1 for the chosen argument and 0 for the other, assuming $\omega_1 \neq \omega_2$).

b) Perform a forward pass, storing all intermediate values.

c) Perform a reverse pass to compute the gradient $\nabla f(3,2)$. Show the adjoint $\bar{v}_i$ for every variable at each step.

Exercise 3. Complexity Analysis
For each scenario below, state whether you would use forward-mode or reverse-mode AD to compute the requested derivative most efficiently, and justify your answer.

a) A function $g: \mathbb{R}^5 \to \mathbb{R}^{100}$; you need the full Jacobian matrix.

b) A scalar loss function $L: \mathbb{R}^{1,000,000} \to \mathbb{R}$; you need the gradient ∇L.

c) A function $h: \mathbb{R}^{100} \to \mathbb{R}^{100}$; you need the product of its Jacobian, J_h, with a known vector $\mathbf{v} \in \mathbb{R}^{100}$.

12.9.2 Implementation and Design

Exercise 4. Simple Operator Overloading (Forward-Mode)
Using a programming language of your choice, implement a simple DualNumber class to perform forward-mode AD. Your class should have:

- Fields for the value (primal) and derivative (tangent).
- Overloaded methods for __add__, __mul__, and __sub__ (if you're using Python).
- A method for a nonlinear function, such as sin or exp.

Use your class to verify your manual calculation from Exercise 1.

Exercise 5. Conceptual Implementation (Reverse-Mode)
Describe the data structure you would use to build a computational graph for reverse-mode AD in a library like PyTorch. What information must be stored in each node during the forward pass to enable the subsequent backward pass? List at least three key pieces of information per node.

12.9.3 Theoretical and Conceptual

Exercise 6. The Categorical Chain Rule
In Section 12.7.1, we discussed the tangent functor T, which implements forward-mode AD. The functoriality of T means $T(g \circ f) = Tg \circ Tf$.

a) Write down the explicit mathematical expression for $T(g \circ f)(x, \dot{x})$.

b) Now, write down the expression for $(Tg \circ Tf)(x, \dot{x})$.

c) Show that these two expressions are equal and explain how this demonstrates the chain rule of calculus.

Exercise 7. AD and the Kock-Lawvere Axiom
Synthetic Differential Geometry postulates the Kock-Lawvere axiom, which states that the space $D = \{ d \in R \mid d^2 = 0 \}$ of nilsquare infinitesimals is sufficient to determine derivatives. Specifically, for any function $f: R \to R$, there exists a *unique* $b \in R$ such that for all $d \in D$, $f(x + d) = f(x) + b \cdot d$. We then define $f'(x) = b$.

Explain how a single evaluation of a function using dual numbers (i.e., computing $f(x + \varepsilon)$) is a direct computational realization of the Kock-Lawvere axiom.

12.9.4 Challenge Problem

Exercise 8. From Forward to Reverse
A colleague claims that reverse-mode AD is “just forward-mode AD run backwards on the computational graph.”

a) Explain the intuitive appeal of this statement.

b) Identify at least two fundamental reasons why this statement is an oversimplification. Consider the processes of seeding, the operations performed at each node, and the overall computational complexity.

12.9.5 Hints and Solutions

Exercise 1.

- **Hint:** The derivative of $\log(u)$ is $\frac{u'}{u}$. At $x = 0$, $v_1 = 0, v_2 = 1, v_3 = 2$.
- **Final Answer:** $f(0) = \log(2)$, $f'(0) = -\frac{1}{2}$.

Exercise 2.

Computational graph

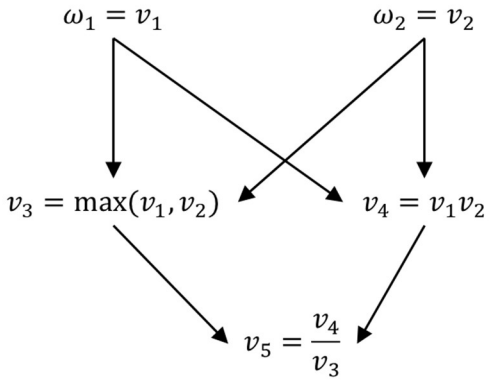
We are given:

$$f(w_1, w_2) = \frac{\omega_1 \omega_2}{\max(\omega_1, \omega_2)}, \qquad (\omega_1, \omega_2) = (3,2)$$

Let's introduce intermediate variables:

$$v_1 = \omega_1 = 3$$
$$v_2 = \omega_2 = 2$$
$$v_3 = \max(v_1, v_2)$$
$$v_4 = v_1 v_2$$
$$v_5 = \frac{v_4}{v_3} = f$$

The computational graph is shown below:



Forward pass (at $w_1 = 3, w_2 = 2$)

$$v_1 = \omega_1 = 3$$
$$v_2 = \omega_2 = 2$$
$$v_3 = \max(v_1, v_2) = \max(3,2) = 3$$
$$v_4 = v_1 v_2 = 3 \times 2 = 6$$
$$v_5 = \frac{v_4}{v_3} = \frac{6}{3} = 2$$

So, $f(3,2) = 2$.

Reverse-mode AD (Backpropagation)

We initialize $\bar{v}_5 = \frac{\partial v_5}{\partial v_5} = 1$.

Let's write each node's local gradient:

For $v_5 = \frac{v_4}{v_3}$:

$$\frac{\partial v_5}{\partial v_4} = \frac{1}{v_3} = \frac{1}{3}$$

$$\frac{\partial v_5}{\partial v_3} = -\frac{v_4}{v_3^2} = -\frac{6}{9} = -\frac{2}{3}$$

For $v_4 = v_1 v_2$:

$$\frac{\partial v_4}{\partial v_1} = v_2 = 2$$

$$\frac{\partial v_4}{\partial v_2} = v_1 = 3$$

For $v_3 = \max(v_1, v_2)$:

Since $v_3 = 3$ comes from v_1 (because $3 > 2$), we have:

$$\frac{\partial v_3}{\partial v_1} = 1, \qquad \frac{\partial v_3}{\partial v_2} = 0$$

at this evaluation point.

Reverse pass:

Step 1: $\bar{v}_5 = 1$

Step 2:

To $\bar{v}_4$:

$$\bar{v}_4 = \bar{v}_5 \cdot \frac{\partial v_5}{\partial v_4} = 1 \times \frac{1}{3} = \frac{1}{3}$$

To $\bar{v}_3$:

$$\bar{v}_3 = \bar{v}_5 \cdot \frac{\partial v_5}{\partial v_3} = 1 \times \left(-\frac{2}{3}\right) = -\frac{2}{3}$$

Step 3: From v_4 to v_1, v_2:

$$\bar{v}_1 += \bar{v}_4 \cdot \frac{\partial v_4}{\partial v_1} = \frac{1}{3} \times 2 = \frac{2}{3}$$

$$\bar{v}_2 += \bar{v}_4 \cdot \frac{\partial v_4}{\partial v_2} = \frac{1}{3} \times 3 = 1$$

Step 4: From v_3 to v_1, v_2:

$$\bar{v}_1 += \bar{v}_3 \cdot \frac{\partial v_3}{\partial v_1} = \left(-\frac{2}{3}\right) \times 1 = -\frac{2}{3}$$

$$\bar{v}_2 += \bar{v}_3 \cdot \frac{\partial v_3}{\partial v_2} = \left(-\frac{2}{3}\right) \times 0 = 0$$

Sum contributions to $\bar{v}_1$**:**

From v_4: $+\frac{2}{3}$

From v_3: $-\frac{2}{3}$

$$\bar{v}_1 = \frac{2}{3} - \frac{2}{3} = 0.$$

Sum contributions to $\bar{v}_2$**:**

From v_4: $+1$

From v_3: $+0$

$$\bar{v}_2 = 1 + 0 = 1.$$

Since $v_1 = w_1$ and $v_2 = w_2$:

$$\frac{\partial f}{\partial w_1} = \bar{v}_1 = 0$$

$$\frac{\partial f}{\partial w_2} = \bar{v}_2 = 1$$

Final gradient:

$$\boxed{(0,1)}$$

We can check manually: at (3,2), $\max \quad = 3$, so $f = \frac{3\times 2}{3} = 2$.

If ω_1 increases slightly, we have that $\max(\omega_1, \omega_2) = \omega_1$. So, $f = \omega_2$ which equals the constant 2, and the derivative w.r.t. ω_1 is 0.

If ω_2 increases slightly, we have that $\max(\omega_1, \omega_2) = \omega_1$. So, $f = \omega_2$, the derivative w.r.t ω_2 is 1.

This matches the backpropagation result.

Exercise 3.

Part a: $g: \mathbb{R}^5 \to \mathbb{R}^{100}$; need full Jacobian matrix of dimension 100×5

- Output dimension: $m = 100$
- Input dimension: $n = 5$

Forward-mode AD computes one column of the Jacobian per forward pass (using one seed vector for input perturbations).

Reverse-mode AD computes one row of the Jacobian per backward pass (using one seed vector for output perturbations).

Here, $n \ll m$, so forward-mode is more efficient:

$$\text{Cost} \approx O(n) \cdot \text{cost}(g) \quad \text{(forward-mode)}$$

versus

$$O(m) \cdot \text{cost}(g) \quad \text{(reverse-mode)}$$

Since $n = 5, m = 100$, forward-mode wins.

Choice: Forward-mode.

Part b: Scalar $L: \mathbb{R}^{1,000,000} \to \mathbb{R}$; need gradient ∇L.

- $m = 1, n = 1{,}000{,}000$

Reverse-mode computes gradient in time $O(1) \cdot \text{cost}(L)$ (in terms of number of backward passes), while forward-mode would require n forward passes.

Thus reverse-mode is vastly more efficient.

***Choice*:** Reverse-mode.

Part c: $h: \mathbb{R}^{100} \to \mathbb{R}^{100}$; need $J_h v$ (Jacobian–vector product).

Jacobian–vector products can be done with forward-mode AD in a single forward pass at a cost of $\sim 2 \cdot \text{cost}(h)$ (dual number forward propagation), independent of n or m. Reverse-mode could compute it via a backward pass with a vector-Jacobian product after computing function values, but that's less direct and may require storing intermediates. For a *single* JVP, forward-mode is natural and efficient.

***Choice*:** Forward-mode.

Exercise 4.

The following is a Python program with some examples embedded at the end of the code.

```
import math

class DualNumber:
    def __init__(self, value, derivative=1.0):
        """
        Initialize a dual number with a value (primal) and derivative (tangent).
        By default, derivative is 1.0 (useful for independent variables).
        """
        self.value = value  # primal
        self.derivative = derivative  # tangent
```

```
    def __repr__(self):
        return f"DualNumber(value={self.value}, derivative={self.derivative})"

    def __str__(self):
        return f"({self.value} + {self.derivative}ε)"

    # Basic arithmetic operations
    def __add__(self, other):
        """Addition: f + g"""
        if isinstance(other, DualNumber):
            return DualNumber(
                self.value + other.value,
                self.derivative + other.derivative
            )
        else:  # constant
            return DualNumber(
                self.value + other,
                self.derivative
            )

    def __radd__(self, other):
        """Addition when constant is on the left: c + f"""
        return self.__add__(other)

    def __sub__(self, other):
        """Subtraction: f - g"""
        if isinstance(other, DualNumber):
            return DualNumber(
                self.value - other.value,
                self.derivative - other.derivative
            )
        else:  # constant
            return DualNumber(
```

```
            self.value - other,
            self.derivative
        )

    def __rsub__(self, other):
        """Subtraction when constant is on the left: c - f"""
        if isinstance(other, DualNumber):
            return other.__sub__(self)
        else:  # constant
            return DualNumber(
                other - self.value,
                -self.derivative
            )

    def __mul__(self, other):
        """Multiplication: f * g"""
        if isinstance(other, DualNumber):
            # Product rule: (fg)' = f'g + fg'
            return DualNumber(
                self.value * other.value,
                self.derivative * other.value + self.value * other.derivative
            )
        else:  # constant
            return DualNumber(
                self.value * other,
                self.derivative * other
            )

    def __rmul__(self, other):
        """Multiplication when constant is on the left: c * f"""
        return self.__mul__(other)
```

```
    def __truediv__(self, other):
        """Division: f / g"""
        if isinstance(other, DualNumber):
            # Quotient rule: (f/g)' = (f'g - fg')/g²
            return DualNumber(
                self.value / other.value,
                (self.derivative * other.value - self.value * other.derivative) / (other.value ** 2)
            )
        else:  # constant
            return DualNumber(
                self.value / other,
                self.derivative / other
            )

    def __rtruediv__(self, other):
        """Division when constant is on the left: c / f"""
        if isinstance(other, DualNumber):
            return other.__truediv__(self)
        else:  # constant
            return DualNumber(
                other / self.value,
                -other * self.derivative / (self.value ** 2)
            )

    def __pow__(self, other):
        """Calculates self ** other (f(x)^c)"""
        if not isinstance(other, DualNumber):
            value = self.value ** other
            # Derivative of f(x)^c is c * f(x)^(c-1) * f'(x)
            derivative = other * (self.value ** (other - 1)) * self.derivative
            return DualNumber(value, derivative)
        # If 'other' is also a DualNumber (i.e., we have f(x)^g(x))
        # This requires the complex derivative rule (d/dx f^g), which we intentionally
```

```
        # exclude from this simplified implementation.
        else:
            raise NotImplementedError(
                "Power Rule for variable exponents (f(x)**g(x)) is not implemented "
                "in this basic DualNumber class. Please use constant exponents."
            )

    # Note: __rpow__ (e.g., 5 ** DualNumber) must also be handled for a full implementation,
    # but that follows the simpler rule for c^f(x): c^f(x) * ln(c) * f'(x)

    def __pow__(self, power):
        """Power: f ** n"""
        if isinstance(power, DualNumber):
            # For f^g, we need more complex rule
            # For simplicity, assuming constant power here
            return DualNumber(
                self.value ** power.value,
                power.value * (self.value ** (power.value - 1)) * self.derivative
            )
        else:  # constant power
            return DualNumber(
                self.value ** power,
                power * (self.value ** (power - 1)) * self.derivative
            )

    # Nonlinear functions (as static methods)
    @staticmethod
    def sin(x):
        """sin(x)"""
        if isinstance(x, DualNumber):
            return DualNumber(
                math.sin(x.value),
                math.cos(x.value) * x.derivative
```

```
            )
        else:
            return math.sin(x)

    @staticmethod
    def cos(x):
        """cos(x)"""
        if isinstance(x, DualNumber):
            return DualNumber(
                math.cos(x.value),
                -math.sin(x.value) * x.derivative
            )
        else:
            return math.cos(x)

    @staticmethod
    def exp(x):
        """exp(x)"""
        if isinstance(x, DualNumber):
            exp_val = math.exp(x.value)
            return DualNumber(
                exp_val,
                exp_val * x.derivative
            )
        else:
            return math.exp(x)

    @staticmethod
    def log(x):
        """log(x) - natural logarithm"""
        if isinstance(x, DualNumber):
            return DualNumber(
                math.log(x.value),
```

```
            (1 / x.value) * x.derivative
        )
    else:
        return math.log(x)

@staticmethod
def sqrt(x):
    """sqrt(x)"""
    if isinstance(x, DualNumber):
        sqrt_val = math.sqrt(x.value)
        return DualNumber(
            sqrt_val,
            (0.5 / sqrt_val) * x.derivative
        )
    else:
        return math.sqrt(x)

# Example usage and testing
if __name__ == "__main__":
    print("=== Testing DualNumber Class ===\n")

    # Create dual numbers for variables
    # Derivative set to 1.0 for independent variables
    x = DualNumber(2.0, 1.0)  # x = 2, dx/dx = 1
    y = DualNumber(3.0, 0.0)  # y = 3, dy/dx = 0 (constant w.r.t x)

    print(f"x = {x}")
    print(f"y = {y}\n")

    # Test addition
    z1 = x + y
    print(f"x + y = {z1}")  # Should be: (5 + 1ε)
```

```
    # Test subtraction
    z2 = x - y
    print(f"x - y = {z2}")  # Should be: (-1 + 1ε)

    # Test multiplication
    z3 = x * y
    print(f"x * y = {z3}")  # Should be: (6 + 3ε)

    # Test nonlinear function: sin
    z4 = DualNumber.sin(x)
    print(f"sin(x) = {z4}")
    print(f"  Derivative of sin(2) = {z4.derivative}")
    print(f"  cos(2) = {math.cos(2)}\n")

    # Test nonlinear function: exp
    z5 = DualNumber.exp(x)
    print(f"exp(x) = {z5}")
    print(f"  Derivative of exp(2) = {z5.derivative}")
    print(f"  exp(2) = {math.exp(2)}\n")

    # Test composite function: sin(x) * exp(x)
    z6 = DualNumber.sin(x) * DualNumber.exp(x)
    print(f"sin(x) * exp(x) = {z6}")
    print(f"  Derivative: {z6.derivative}")
    print(f"  Expected: sin(2)*exp(2) + cos(2)*exp(2) = {math.sin(2)*math.exp(2) +
math.cos(2)*math.exp(2)}\n")

    # Test with a more complex example
    def example_function(x_val):
        """Example function: f(x) = sin(x) * exp(x) + x^2"""
        x = DualNumber(x_val, 1.0)
        return DualNumber.sin(x) * DualNumber.exp(x) + x**2
```

```
result = example_function(2.0)
print(f"f(x) = sin(x)*exp(x) + x^2")
print(f"f(2) = {result.value}")
print(f"f'(2) = {result.derivative}")
```

Exercise 5.

For reverse-mode automatic differentiation (backpropagation) in libraries like PyTorch, each node in the computational graph stores the following critical information:

- Node Value (Primal)
  - What: The actual output value/tensor computed during forward pass
  - Why needed: Required for gradient computation during backward pass (many gradient formulas depend on the forward values)
  - Example: For $z = x * y$, we need to store x and y values to compute $\frac{\partial z}{\partial x} = y$ and $\frac{\partial z}{\partial y} = x$
- Gradient Accumulator
  - What: Accumulated gradient with respect to the output $\left(\frac{\partial L}{\partial node}\right)$
  - Why needed: Stores the gradient flowing back from downstream nodes; accumulates gradients when a node feeds into multiple operations
  - Initially: Set to "None" or zero; accumulates during backward pass
  - Note: For leaf nodes (inputs/parameters), this is what gets used for parameter updates
- Operation Reference and Gradient Function
  - What: A "backward function" that knows how to compute gradients of inputs given gradient of output
  - Why needed: Encapsulates the chain rule for that specific operation
  - Contains:
    - Function pointer to compute $\frac{\partial node}{\partial inputs}$
    - References to input nodes
    - Any saved tensors needed for gradient computation

Additional Important Information

- References to Input Nodes (Parents)
  - What: Pointers/identifiers to the nodes that produced a given node's inputs
  - Why needed: To traverse the graph backward during gradient computation
  - Example: For $z = x + y$, store references to nodes x and y

- Topological Information
    - What: Graph structure metadata (fan-out count, visited flags, etc.)
    - Why needed: For efficient graph traversal and memory management
    - Includes:
        - Number of times this node is used by downstream nodes
        - Graph version/timestamp for caching
        - Requires “grad flag” to enable/disable gradient tracking

Example Node Structure in PyTorch/TensorFlow

```
class Node:
  def __init__(self):
    self.data = None        # Tensor value (primal)
    self.grad = None        # Accumulated gradient (∂L/∂node)
    self.grad_fn = None     # Gradient function for this operation
    self._ctx = None        # Context/operation details
    self.requires_grad = False # Whether to track gradients
    self.is_leaf = False    # Whether this is a leaf (input/parameter)
    # Additional implementation details …
```

Forward Pass Operations

During forward computation, each operation:

- Computes the output value
- Creates a new node with the result
- Attaches a gradient function (“grad_fn”) that knows:
    - Which operation was performed (add, mul, matmul, etc.)
    - References to input nodes
    - Any intermediate values needed for gradient computation
- Sets up the backward edges in the computational graph

Example: Simple Multiplication (Python)

```
# Forward pass
x = Tensor(2.0, requires_grad=True)  # Leaf node
y = Tensor(3.0, requires_grad=True)  # Leaf node
z = x * y  # Creates new node
```

```
# Node 'z' would store:
# - data: 6.0
# - grad: None (initially)
# - grad_fn: MultiplicationBackward (inputs=[x, y])
# - is_leaf: False
```

Why These Three Are Most Critical

1. Node Value: Without the forward values, we cannot compute most gradients
2. Gradient Accumulator: Without storage for $\frac{\partial L}{\partial node}$, gradients cannot propagate
3. Gradient Function: Without knowledge of how to compute $\frac{\partial node}{\partial inputs}$, the chain rule cannot be applied

This structure enables efficient reverse-mode AD where:

- Forward pass: Builds graph and stores values needed for gradients
- Backward pass: Traverses graph backward, applying chain rule via stored "grad_fn" functions
- Memory: Only stores necessary information (trades off memory for speed)

Exercise 6.

Understanding T for forward-mode AD

In forward-mode automatic differentiation, the tangent functor T transforms a function

$$f: \mathbb{R}^n \to \mathbb{R}^m$$

to a function on tangent bundles:

$$Tf: \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^m \times \mathbb{R}^m$$

given by:

$$Tf(x, \dot{x}) = (f(x), Df(x) \cdot \dot{x})$$

where $Df(x)$ is the Jacobian matrix of f at x.

Problem setup

We have $f: \mathbb{R}^n \to \mathbb{R}^p$ and $g: \mathbb{R}^p \to \mathbb{R}^m$, with $g \circ f: \mathbb{R}^n \to \mathbb{R}^m$.

We want to compute:

(a) $T(g \circ f)(x, \dot{x})$

By definition of T:

$$T(g \circ f)(x, \dot{x}) = \big((g \circ f)(x), D(g \circ f)(x) \cdot \dot{x}\big).$$

Using the chain rule, $D(g \circ f)(x) = Dg\big(f(x)\big)\, Df(x)$ and so,

$$D(g \circ f)(x) \cdot \dot{x} = Dg\big(f(x)\big)\,(Df(x)\,\dot{x}).$$

Thus,

$$T(g \circ f)(x, \dot{x}) = \Big(g\big(f(x)\big), Dg\big(f(x)\big)(Df(x)\,\dot{x})\Big).$$

(b) $Tg \circ Tf$ applied to $(x, \dot{x})$

By definition, we have

$$Tf(x, \dot{x}) = (f(x), Df(x)\,\dot{x}).$$

Let $y = f(x)$ and $\dot{y} = Df(x)\,\dot{x}$.

Now apply Tg to $(y, \dot{y})$:

$$Tg(y, \dot{y}) = (g(y), Dg(y)\,\dot{y})$$

Substitute $y = f(x)$ and $\dot{y} = Df(x)\dot{x}$, we have

$$Tg\big(Tf(x, \dot{x})\big) = \Big(g\big(f(x)\big), Dg\big(f(x)\big)\,(Df(x)\,\dot{x})\Big)$$

Equality

From (a):

$$T(g \circ f)(x, \dot{x}) = \Big(g\big(f(x)\big), Dg\big(f(x)\big)(Df(x)\dot{x})\Big)$$

From (b):

$$(Tg \circ Tf)(x, \dot{x}) = Tg\big(Tf(x, \dot{x})\big) = \Big(g\big(f(x)\big), Dg\big(f(x)\big)(Df(x)\dot{x})\Big)$$

These are identical and therefore, we have

$$T(g \circ f) = Tg \circ Tf$$

How this demonstrates the chain rule

The equality $T(g \circ f) = Tg \circ Tf$ is a functorial property.

Breaking it down:

- On the first component (primal part), it just says $(g \circ f)(x) = g\big(f(x)\big)$.
- On the second component (tangent part), the equality says:

From $T(g \circ f)$: tangent = $D(g \circ f)(x)\dot{x}$.

From $Tg \circ Tf$: tangent = $Dg\big(f(x)\big)\,[Df(x)\dot{x}]$.

Equating these gives:

$$D(g \circ f)(x)\dot{x} = Dg\big(f(x)\big)\,[Df(x)\dot{x}].$$

Since $\dot{x}$ is arbitrary, this implies the matrix (or linear map) equality:

$$D(g \circ f)(x) = Dg\big(f(x)\big)\,Df(x),$$

which is precisely the **chain rule** for total derivatives.

Thus, the functoriality of T is an abstract, higher-level formulation of the chain rule: it says that *the derivative of a composition is the composition of the derivatives* (in the sense of tangent maps). Forward-mode AD is essentially implementing this functor mechanically.

Exercise 7.

In the dual numbers, ε is a nilsquare infinitesimal. Evaluating $f(x + \varepsilon)$ yields $f(x) + f'(x)\varepsilon$. The coefficient $f'(x)$ is the unique number b promised by the axiom, computed directly.

Exercise 8.

a) Both involve traversing the graph and applying the chain rule.

b) 1) **Seeding:** Reverse-mode seeds the *output* adjoint ($\bar{f} = 1$), while forward-mode seeds *input* tangents. 2) **Node Operations:** Reverse-mode requires storing intermediate values from the forward pass and performs different (adjoint accumulation) operations. 3) **Complexity:** The computational complexity for a full gradient is $O(1)$ reverse passes vs. $O(n)$ forward passes, which is not a simple "reversal" but a fundamental efficiency gain.

Acronyms and Symbols

Outside of a Dog, a Book is Man's Best Friend. Inside of a Dog, It's Too Dark to Read.

Quote popularized by Groucho Marx

$\equiv$ - indicates a definition (rather than equality)

$\forall$ - for every

$\exists$ - there exists

$\ni$ - such that

A_n – Alternating Group

$\mathbb{C}$ - Complex Numbers

$\mathbb{D}$ – Dual Numbers

$\mathbb{H}$ - Quaternions

$\mathbb{N}$ - Natural Numbers, i.e., 1,2,3,…

$\mathcal{P}(A)$ – Power set of the set A, i.e., the set of all subsets of A

$\mathbb{Q}$ - Rational Numbers

$\mathbb{R}$ - Real Numbers

$\mathbb{R}^n$ – n dimensional Euclidean space

$\mathbb{R}[X]$ – set of polynomials with coefficients from the real numbers

S_n – Symmetric Group

$\mathbb{Z}$ - Integers

$\mathbb{Z}_n$ – Integers mod n

AD – Automatic Differentiation (Auto Diff)

CCC – Cartesian Closed Category

DAG – Directed Acyclic Graph

FTC – Fundamental Theorem of Calculus

$GL_n(F)$ – General Linear group of invertible $n \times n$ matrices over the field F

HMC – Hamiltonian Monte Carlo

JVP – Jacobian-Vector Product

MCMC – Markov Chain Monte Carlo

ML – Machine Learning

NSA – Non-Standard Analysis

NUTS – No-U-Turn Sampler

PPL - Probabilistic Programming Language

RDC – Reverse Derivative Category

SDG – Synthetic Differential Geometry

$SL_n(F)$ – Special Linear group of $n \times n$ matrices with determinant 1 over the field F

UFD – Unique Factorization Domain

VJP – Vector-Jacobian Product

w.r.t. – with respect to

ZFC – **Z**ermelo–**F**raenkel set theory with the axiom of **C**hoice included

References

[1] Robinson, Abraham (1996), *Non-standard analysis**, Princeton University Press, ISBN 978-0-691-04490-3. The classic introduction to nonstandard analysis. https://archive.org/details/nonstandardanaly0000robi/page/n5/mode/2up

[2] Thomas, J.R., Introduction to Hyperreals, Otterbein College, http://faculty.otterbein.edu/TJames/Into%20to%20HyperReals.ppt, accessed on 3 October 2025.

[3] Clifford, William Kingdon (1873), *Preliminary Sketch of Bi-quaternions*, Proceedings of the London Mathematical Society. 4: 381–395. https://londmathsoc.onlinelibrary.wiley.com/doi/abs/10.1112/plms/s1-4.1.381

[4] Eilenberg, Samuel, Mac Lane, Saunders (1945), *General theory of natural equivalences*, Transactions of the American Mathematical Society. https://www.ams.org/journals/tran/1945-058-00/S0002-9947-1945-0013131-6/S0002-9947-1945-0013131-6.pdf

[5] Weisstein, Eric W. "Field." From MathWorld--A Wolfram Resource. https://mathworld.wolfram.com/Field.html, accessed on 18 September 2025.

[6] Judson, T.W., *Abstract Algebra: Theory and Applications*, Publisher: University of Puget Sound, available from the Open Textbook Library at https://open.umn.edu/opentextbooks/textbooks/217.

[7] Neumann, P.M., *A breakthrough in Algebra: Classification of the Finite Simple Groups*, YouTube video, https://youtu.be/s88bfJzyA78, accessed on 3 October 2025.

[8] *Modular arithmetic*, Wikipedia, https://en.wikipedia.org/wiki/Modular_arithmetic, accessed on 3 October 2025.

[9] *Multiplicative group of integers modulo n*, Wikipedia, https://en.wikipedia.org/wiki/Multiplicative_group_of_integers_modulo_n, accessed on 3 October 2025.

[10] *Symmetric group*, Wikipedia, https://en.wikipedia.org/wiki/Symmetric_group, accessed on 3 October 2025.

[11] *Bijection*, Wikipedia, https://en.wikipedia.org/wiki/Bijection, accessed on 3 October 2025.

[12] *Alternating group*, Wikipedia, https://en.wikipedia.org/wiki/Alternating_group, accessed on 3 October 2025.

[13] *Dihedral group*, Wikipedia, https://en.wikipedia.org/wiki/Dihedral_group, accessed on 3 October 2025.

[14] *General linear group*, Wikipedia, https://en.wikipedia.org/wiki/General_linear_group, accessed on 3 October 2025.

[15] *Determinant*, Wikipedia, https://en.wikipedia.org/wiki/Determinant, 3 October 2025.

[16] *Special linear group*, Wikipedia, https://en.wikipedia.org/wiki/Special_linear_group, 3 October 2025.

[17] Rotman, J., *A First Course in Abstract Algebra (Third Edition)**, Prentice Hall, 2005.

[18] *Lagrange's theorem (group theory)*, Wikipedia, https://en.wikipedia.org/wiki/Lagrange%27s_theorem_(group_theory), accessed on 3 October 2025.

[19] *Determinant: Multiplicativity and matrix groups*, Wikipedia, https://en.wikipedia.org/wiki/Determinant#Multiplicativity_and_matrix_groups, accessed on 3 October 2025.

[20] Baumslag, Benjamin (2006), "A simple way of proving the Jordan-Hölder-Schreier theorem", American Mathematical Monthly, 113 (10): 933–935, doi:10.2307/27642092. https://www.jstor.org/stable/27642092

[21] O. Hölder, Die einfachen Gruppen in ersten und zweiten Hundert der Ordnungszahlen, Math. Annalen 40 (1892), 55–88.

[22] W. Burnside, On a class of groups of finite order, Trans. Cambridge Phil. Soc. 18 (1899), 269–276.

[23] Gorenstein, D., Finite Simple Groups; *An Introduction to Their Classification*, Plenum, New York, 1982.

[24] Gorenstein, D., *The Classification of the Finite Simple Groups*, Volume I, Plenum, New York, 1983.

[25] Aschbacher, Michael (2004). "The Status of the Classification of the Finite Simple Groups". Notices of the American Mathematical Society. Vol. 51, no. 7. pp. 736–740. https://www.ams.org/notices/200407/fea-aschbacher.pdf

[26] *Classification of finite simple groups*, Wikipedia, https://en.wikipedia.org/wiki/Classification_of_finite_simple_groups, accessed on 3 October 2025.

[27] *Abelian Group is Simple iff Prime, Proof Wiki*, https://proofwiki.org/wiki/Abelian_Group_is_Simple_iff_Prime, accessed on 3 October 2025.

[28] *Fundamental Theorem of Finite Abelian Groups*, Proof Wiki, https://proofwiki.org/wiki/Fundamental_Theorem_of_Finite_Abelian_Groups, accessed on 3 October 2025.

[29] *List of finite simple groups*, Wikipedia, https://en.wikipedia.org/wiki/List_of_finite_simple_groups, accessed on 3 October 2025.

[30] Al Doerr, Ken Levasseur, *Applied Discrete Structures*, LibreTexts™, Section 16.3 “Polynomial Rings”, https://math.libretexts.org/Bookshelves/Combinatorics_and_Discrete_Mathematics/Applied_Discrete_Structures_(Doerr_and_Levasseur)/16%3A_An_Introduction_to_Rings_and_Fields/16.03%3A_Polynomial_Rings, accessed on 3 October 2025.

[31] *Polynomial ring*, Wikipedia, https://en.wikipedia.org/wiki/Polynomial_ring, accessed on 16 April 2023.

[32] *Ring homomorphism*, Wikipedia, https://en.wikipedia.org/wiki/Ring_homomorphism, accessed on 3 October 2025.

[33] *Intersection Distributes over Symmetric Difference*, Proof Wiki, https://proofwiki.org/wiki/Intersection_Distributes_over_Symmetric_Difference, 3 October 2025.

[34] *Set Intersection Not Cancellable*, Proof Wiki, https://proofwiki.org/wiki/Set_Intersection_Not_Cancellable, accessed on 3 October 2025.

[35] *Pointwise product*, Wikipedia, https://en.wikipedia.org/wiki/Pointwise_product, accessed on 3 October 2025.

[36] *Ideal (ring theory)*, Wikipedia, https://en.wikipedia.org/wiki/Ideal_(ring_theory), accessed on 3 October 2025.

[37] *Polynomial long division*, Wikipedia, https://en.wikipedia.org/wiki/Polynomial_long_division, accessed on 15 June 2023.

[38] *Field (mathematics)*, Wikipedia, https://en.wikipedia.org/wiki/Field_(mathematics), accessed on 3 November 2025.

[39] *Principle ideal*, Wikipedia, https://en.wikipedia.org/wiki/Principal_ideal, accessed on 3 October 2025.

[40] *Finite field*, Wikipedia, https://en.wikipedia.org/wiki/Finite_field, accessed on 3 October 2025.

[41] Schilling, A., Nachtergaele, B., and Lankham, I., *Linear Algebra*, LibreTexts™, https://math.libretexts.org/Bookshelves/Linear_Algebra/Book%3A_Linear_Algebra_(Schilling_Nachtergaele_and_Lankham), accessed on 14 October 2025.

[42] *Integral Domain*, Wikipedia, https://en.wikipedia.org/wiki/Integral_domain, accessed on 3 October 2025.

[43] Weisstein, Eric W. "Zorn's Lemma." From MathWorld--A Wolfram Resource. https://mathworld.wolfram.com/ZornsLemma.html, accessed on 21 September 2025.

[44] Goldblatt, R., Lectures on the Hyperreals, Springer-Verlag, 1998.

[45] Weisstein, Eric W. "Equivalence Relation." From MathWorld--A Wolfram Resource. https://mathworld.wolfram.com/EquivalenceRelation.html, accessed on 21 September 2025.

[46] Weisstein, Eric W. "Ring." From MathWorld--A Wolfram Resource. https://mathworld.wolfram.com/Ring.html, accessed on 21 September 2025.

[47] *Diagonal Embedding*, Planet Math, https://planetmath.org/DiagonalEmbedding, accessed on 21 September 2025.

[48] Weisstein, Eric W. "Totally Ordered Set." From MathWorld--A Wolfram Resource. https://mathworld.wolfram.com/TotallyOrderedSet.html, accessed on 21 September 2025.

[49] Szudzik, Matthew and Weisstein, Eric W. "Continuum Hypothesis." From MathWorld--A Wolfram Resource. https://mathworld.wolfram.com/ContinuumHypothesis.html, accessed on 23 September 2025.

[50] Renze, John and Weisstein, Eric W. "Commutative Algebra." From MathWorld--A Wolfram Resource. https://mathworld.wolfram.com/CommutativeAlgebra.html, accessed on 23 September 2025.

[51] Knuth, D., *Surreal Numbers**, Addison-Wesley Professional, 1974. https://archive.org/details/surrealnumbersho0000knut

[52] Weisstein, Eric W. "Artinian Ring." From MathWorld--A Wolfram Resource. https://mathworld.wolfram.com/ArtinianRing.html, accessed on 23 September 2025.

[53] Fratini, S., *The Shape of Space: A Guided Tour of Vectors, Matrices, Tensors and Markov Chains*, self-published on Amazon. Electronic version: https://github.com/sfratini33/art-of-managing-things-external/blob/master/free_books/ShapeOfSpace.pdf.

[54] Fratini, S., *Mathematical Vignettes - Volume III: Non-Euclidean Geometry, Topology and Complex Analysis (2nd Edition)*, self-published on Amazon, Electronic version: https://github.com/sfratini33/art-of-managing-things-external/blob/master/free_books/MathVig-III-2nd-edition.pdf.

[55] Michael Hallam, *Internal Hom-Objects in the Category of Topological Spaces (2015/2016 Vacation Research Scholarships report*, Australian Mathematical Sciences Institute). https://vrs.amsi.org.au/wp-content/uploads/sites/84/2016/03/Michael-Hallam-Report.pdf

* Indicates the book or article is available for borrowing from the Internet Archive at https://archive.org/. In some cases, only an earlier edition of a book will be available.

Index of Terms

Adjointness ... 111
Algebra ... 61
Alternating group ... 29
Basis of a vector space ... 59
Binary operation ... 26
Cartesian closed category ... 119
Category ... 107
Closure with respect to an operation ... 26
Cocones ... 132
Cofinite ... 71
Colimits ... 129
Commutative (or abelian) group ... 27
Commutative ring ... 41
Composition operation ... 107
Composition series ... 39
Cones ... 132
Coordinate vector ... 60
Coset of a subgroup ... 35
Currying ... 122
Cyclic group ... 35
Cyclic subgroup ... 35
Dedekind cut ... 64
Diagonal embedding ... 75
Dihedral group ... 30
Direct product of groups ... 40
Division algorithm ... 49
Division ring ... 51
Dyadic numbers ... 66
Exponentiation for groups ... 33
Field ... 50, 89
Field homomorphism ... 52
Field isomorphism ... 52
Filter ... 71
Fréchet filter ... 71
Functor ... 110
Galois field ... 52
General linear group ... 31
Generalized associativity ... 33
Generator of a group ... 27
Group (abstract algebra) ... 26
Homomorphic groups ... 38
Hyperreals ... 74
Ideal (ring theory) ... 46
Inner product ... 60
Integral domain ... 42
Isomorphic groups ... 39
Isomorphic rings ... 43
Limit ... 130
Linear combination of vectors ... 58
Linear span of a set of vectors ... 59
Linear subspace of a vector space ... 58
Linearly dependent vectors ... 58
Linearly independent vectors ... 58
Manifold ... 133
Morphism ... 107
Multiplicative cancellation law ... 42
Multiplicative group of integers modulo n ... 27
Natural transformation ... 116
Non-principal (or free) ultrafilter ... 72
Normal subgroup ... 37
Order of a group ... 27
Order of an element of a group ... 33
Orthogonal vectors ... 60
Principal ideal ... 46
Principal ultrafilter ... 71
Proper ideal ... 46
Quaternion ... 51
Quaternions ... 68
Quotient (or factor) ring ... 48
Quotient group ... 38
Ring ... 41, 89
Simple group ... 39
Special linear group ... 31
Subgroup ... 33
Subobjects ... 129
Surreal numbers ... 94
Symmetric group ... 28
Topos ... 133
Transposition (2-cycle permutation) ... 29
Ultrafilter ... 71
Ultrapower ... 74
Unit element of a ring ... 42
Unit ideal ... 46
Vector space ... 56
Weil algebra ... 104
Zero divisor ... 42
Zero ideal ... 46