The Book Review Column[1]
by Frederic Green

Department of Mathematics and Computer Science
Clark University
Worcester, MA 02465
email: fgreen@clarku.edu

Three books are reviewed in this issue, and here they are in order of decreasing abstraction and increasing direct applicability:

1. **Set Theory: A First Course**, by Daniel W. Cunningham. An introduction to axiomatic set theory. Reviewed by Frederic Green.

2. **CryptoSchool**, by Joachim von zur Gathen. A detailed text on many aspects of cryptography, interleaved with a significant amount of its history. Reviewed by Steve Homer.

3. **R for Data Science: Import, Tidy, Transform, Visualize, and Model Data**, by Hadley Wickham and Garrett Grolemund. A practical introduction to data science via R. Review by Allan M. Miller.

BOOKS THAT NEED REVIEWERS FOR THE SIGACT NEWS COLUMN

Algorithms

1. *Distributed Systems: An algorithmic approach (second edition)*, by Ghosh
2. *Tractability: Practical approach to Hard Problems*, Edited by Bordeaux, Hamadi, Kohli
3. *Recent progress in the Boolean Domain*, Edited by Bernd Steinbach
4. *A Guide to Graph Colouring Algorithms and Applications*, by R.M.R. Lewis
5. *Twenty Lectures on Algorithmic Game Theory,* by Tim Roughgarden
6. *Compact Data Structures,* by Gonzalo Navarro
7. *Algorithms and Models for Network Data and Link Analysis,* by François Fouss, Marco Saerens, and Masashi Shimbo

Programming Languages

1. *Selected Papers on Computer Languages* by Donald Knuth
2. *Practical Foundations for Programming Languages,* by Robert Harper

Miscellaneous Computer Science

1. *Algebraic Geometry Modeling in Information Theory,* Edited by Edgar Moro
2. *CoCo: The colorful history of Tandy's Underdog Computer* by Boisy Pitre and Bill Loguidice
3. *Introduction to Reversible Computing*, by Kalyan S. Perumalla
4. *A Short Course in Computational Geometry and Topology*, by Herbert Edelsbrunner
5. *Network Science*, by Albert-László Barabási
6. *Actual Causality*, by Joseph Y. Halpern
7. *Partially Observed Markov Decision Processes,* by Vikram Krishnamurthy
8. *The Power of Networks*, by Christopher G. Brinton and Mung Chiang
9. *Game Theory, Alive,* by A. Karlin and Y. Peres
10. *Power Up: Unlocking the Hidden Mathematics in Video Games,* by Matthew Lane

Computability, Complexity, Logic

1. *The Foundations of Computability Theory,* by Borut Robič
2. *Models of Computation*, by Roberto Bruni and Ugo Montanari
3. *Proof Analysis: A Contribution to Hilbert's Last Problem* by Negri and Von Plato.

Cryptography and Security

1. *Cryptography in Constant Parallel Time,* by Benny Appelbaum
2. *Secure Multiparty Computation and Secret Sharing*, Ronald Cramer, Ivan Bjerre Damgård, and Jesper Buus Nielsen
3. *A Cryptography Primer: Secrets and Promises*, by Philip N. Klein

Combinatorics and Graph Theory

1. *Finite Geometry and Combinatorial Applications*, by Simeon Ball
2. *Introduction to Random Graphs*, by Alan Frieze and Michał Karoński
3. *Erdös–Ko–Rado Theorems: Algebraic Approaches*, by Christopher Godsil and Karen Meagher
4. *Combinatorics, Words and Symbolic Dynamics,* Edited by Valérie Berthé and Michel Rigo
5. *Words and Graphs*, by Sergey Kitaev and Vadim Lozin

Miscellaneous Mathematics and History

1. *Professor Stewart's Casebook of Mathematical Mysteries* by Ian Stewart

Review of[2]
Set Theory: A First Course
by Daniel W. Cunningham
Cambridge, 2016
250 pages, Hardcover, $45.00

Review by

Frederic Green fgreen@clarku.edu
Department of Mathematics and Computer Science
Clark University, Worcester, MA

When I first "learned" a thing or two about "set theory" in junior high school, it all seemed obvious, trivial, and of no apparent use whatsoever. And at that level, I think, I was right. One can barely scratch the surface with *naïve* naïve set theory, and I now regard the attempt to introduce the concept (devoid of any application) into the curriculum at such an early grade as misguided, at best. Set theory in and of itself can only be interesting to one in possession of enough mathematical background and sophistication to appreciate its more subtle aspects, e.g., the nature of infinite sets; in short, what might be expected of a math or CS major in their second year (or later) of their undergraduate eduction.

On the other hand, mathematics is built so essentially on the bedrock of set theory, and we use it in almost everything we do without thinking deeply about it, that it's easy to take for granted. While naïve set theory is probably all that most math or CS students (and even researchers) will need through their careers, there is much to be said in favor of introducing set theory in some depth as a more routine component of undergraduate curricula. This is not because working mathematicians routinely draw on techniques such as transfinite induction or ordinal arithmetic; but rather, because the subject provides an ideal medium for thinking abstractly, and cultivating skills in the development of rigorous mathematical proofs. As Cunningham points out in his preface, a background in set theory gives the mathematics student a distinct advantage in subsequent courses in analysis or algebra. It is also an intrinsically beautiful theory, which starts with virtually nothing and ascends to some of the deepest ideas and most universal themes in mathematics.

Cunningham's book develops axiomatic set theory truly *ab initio*, assuming little on the part of the reader beyond mathematical maturity. To get the best sense of how this is done, here is a run-down of the contents of the book, chapter by chapter:

1. *Introduction*: We begin with naïve set theory. There is also an introduction to propositional and predicate calculus, using them to establish a formal language for set theory (in which everything, including any element of a set, is a set). The chapter concludes with the statement of the Zermelo-Fraenkel axioms. All is very well-motivated historically and logically. Throughout, great care is taken to give examples. The ZF axioms are each briefly discussed in turn, and it is explained how they will be used, e.g., the regularity axiom is not used explicitly until chapter 8.

2. *Basic Set-Building Axioms and Operations*. This considers the first 6 axioms, further explaining each and then using them to build up useful facts and constructions. For example, the Subset Axiom leads to a handy theorem for proving when a class is a set (my impression is that this Theorem (2.1.3) is applied more than any other single theorem in the book). The Union Axiom can be used to construct

both the union and the intersection of the elements of a given set. These tools and the axioms are further applied to obtain basic results such as the De Morgan and Distributive laws.

3. *Relations and Functions* begins with Kuratowski's construction of ordered pairs entirely within the axioms of set theory. It moves on to the definition of relation and various associated notions such as the inverse, image, and restriction; and properties such as single-rootedness; and equivalence relations and their relationship to partitions. This is followed by the basic definitions of functions, including properties such as one-to-one functions and bijections. We then consider indexed functions and the axiom of choice, including a brief discussion of the distinction between ZF and ZFC. The chapter concludes with routine fundamentals on order relations (i.e., partial and total orders), congruence, and preorder relations.

4. *The Natural Numbers*. Up to this point the book covers subjects that are pretty standard fare in a discrete math course, but here we are entering somewhat deeper waters. We begin with the construction of the natural number (as sets, of course) and define the set of natural numbers ω and inductive sets, out of which the principle of mathematical induction emerges naturally. This is used to prove the Recursion Theorem, which leads to the Peano axioms and the theory of arithmetic and order on ω. The power of the techniques is illustrated by numerous applications of the same basic tools, in this case inductive arguments.

5. *On the Size of Sets* begins with careful definition of the size of a finite set in terms of a one-to-one function from the set to a natural number (each natural number being represented as a set here and throughout the book). Various properties of these functions and finite sets are explored, including the Pigeonhole Principle. Along the way, one obtains as a corollary that ω is infinite. The main agenda of the chapter is Cantor's theory generalizing the notion of the size of a set to infinite sets. Thus we study countable sets, and diverse ways to build them (e.g., from the countable union of countable sets, as finite sequences of countable sets, etc.). The proof of Cantor's theorem stating the uncountability of all functions from ω to $\{0, 1\}$ is very clearly stated and explained.

6. *Transfinite Recursion.* While the material gets significantly more difficult at this point, it also gets a lot more interesting. This is an exceptionally cogent presentation of transfinite recursion. The existence of functions defined by transfinite recursion is proved for ordinary functions and then generalized to "class functions" (a class that is functional in the sense that the defining relation is single-valued). The subtleties in going from the set form to the class form are very clearly explained. This is the first instance in which the Replacement Axiom is applied. The Transfinite Recursion Theorem is used to prove that every set is a subset of its transitive closure.

7. *The Axiom of Choice (Revisited).* The main theme of this chapter is Zorn's Lemma, its relationship to AC, and some of its consequences. The text proves that AC implies Zorn's Lemma; the opposite implication (from which equivalence follows) is relegated to an exercise. Applications include the comparability theorem (the cardinality of any two sets are comparable) and the order extension theorem (any partial order on a set can be extended to a total order). Filters, ultrafilters, and ideals are introduced and their interrelationships are explored in light of Zorn's Lemma. E.g., filters and ideals can be easily constructed one from the other, and the ideals corresponding to ultrafilters are prime. The chapter closes with a proof of the Well-Ordering Theorem from Zorn's Lemma.

8. *Ordinals.* The ordinal numbers are defined (effortlessly, given the background provided in earlier chapters) in terms of well-ordered transitive sets. Properties of ordinals are derived, and the ordinal

type belonging to any well-ordered structure is characterized. The ideas of supremum/infimum, and successor and limit ordinals are introduced, and the Burali-Forti theorem (that the class of ordinals is not a set) is proved. Transfinite induction is formulated as ordinal induction, resulting in the ordinal recursion theorem. The latter is applied to normal class functions, resulting in Veblen's Fixed Point Theorem. Transfinite recursion is then used to define arithmetic operations on the ordinals (by analogy with recursion on ω, used to define arithmetic on ω back in Chapter 4). Finally, the cumulative hierarchy is defined, its basic properties are proved, and it is identified (in part thanks to the applications of the regularity axiom promised in Chapter 1) as the universe of all sets.

9. *Cardinals.* Again, the cardinals are effortlessly defined given the previous chapters. What follows is, perhaps, the most technically challenging chapter. After some preliminaries, we proceed quickly to Hartogs' Theorem (the class of cardinals is proper), and, via ordinal class recursion, to the $\aleph$ numbers. The ideas of cofinality and regular and singular cardinals are introduced (in a sense *re*-introduced in the case of cofinality, as the notion of "cofinal" first appears in Chapter 7), and applied to the $\aleph$ numbers and the cardinality of (cartesian) products of ordinals. The next section constructs the arithmetic operations for cardinals. Exponentiation facilitates precise statements of the cardinality of the reals, the continuum hypothesis, and the generalized continuum hypothesis. König's Theorem (i.e., the theorem of that name which states that any infinite cardinal κ is contained in the cofinality of 2^κ) is proved, appealing to AC. The final section of the book deals with unbounded, closed and club sets, and proves some results on club filters and stationary sets.

As far as I can see, Cunningham neglects no opportunity to make the subject as accessible as possible. The mathematical development is rigorous, as it should be, but not excessively so. Although he starts from zero, that is not to say the book is easy, but any difficulty that arises is in the nature of the subject, and is no fault of the author's. Throughout the book, he offers many appropriate examples (or non-examples), and provides numerous and diverse exercises, which often prove results that are later used in the body of the text, drawing the reader into the subject. The reader participation increases in later chapters, in which a very significant fraction of the proofs is done in exercises. This is eminently suitable for a course at the undergraduate level, although once one hits Chapter 6 and beyond, decidedly at the junior or senior level. It would also be great for self-study.

In short, this is an excellent book! You will know a good deal about basic set theory after working through it.

Review of[3]
CryptoSchool
by Joachim von zur Gathen
Springer, 2015
888 pages, Hardcover, $44.99

Review by
Steve Homer (homer@bu.edu**)**
Department of Computer Science
Boston University, Boston, MA

1 Overview

The history of secret codes and their decoding (coding and decoding secure messages) contains repetitions of cryptographers creating unbreakable cryptosystems and cryptanalysts eventually breaking them. This endless cycle remains today but advances in the theory of computation and in mathematical cryptography have resulted in provable security bounds and an understanding of the formal capabilities they provide. Modern cryptography is a notable success story for computer science, and for theoretical computer science in particular. Under reasonable computational assumptions, theoretical advances have enabled proofs of strong and useful security claims and given increased confidence to security claims within its purview.. (Its advances have enabled limits of some systems under reasonable formal assumptions to justify confidence in many formal security claims.)

This is not to say that that today's cryptography solves all of the problems of modern digital security, or that secure crypto is applicable for all security risks. Failure is always a possibility as there are many modern cryptosystems whose security depends on the hardness of open theoretical or algorithmic problems, by complex interactions within a computer system even when made of individually secure parts, or on unknown patterns of algorithmic complexity within a security system. However most present day failures of security occur outside of the boundaries that provable security provides. Often these can be the result of side-channel attacks, or human oversight or error. Furthermore formal security methods have been successfully extended to new cryptological uses and applications, and presents the possibility of future uses (and failures) far beyond the original objectives of this field.

CryptoSchool, by Joachim von zur Gathen is a large volume focusing mainly on modern cryptography and on currently active new and important aspects of the field. It emphasizes new concepts, formal techniques, and tools in cryptography, and includes active research topics, detailed proofs and formal security methods. The book comprises 25 chapters, totaling 888 pages, and is made up of two intertwined parts. It contains much current cryptography than a single cryptography class could cover, and it intersperses them with brief snapshots of the cryptographic history that preceded it.

After a short introduction, the core cryptography chapters begin with four chapters focusing on important current cryptosystems. Following these central chapters comes a series of nine chapters on current cryptographic topics, and others on extended topics of current related interest. All of these are intermixed with ten episodic history chapters. Interspersed with these cryptographic chapters are 10 chapters helping to put the current achievements in context.

CryptoSchool is a very ambitious, carefully written and organized, and artfully designed book which is more than a standard cryptography text. It is quite different from other cryptography texts in its style

and scope, and a valuable resource to have on one's bookshelf. It is intended for mathematically capable students or professionals and written at a high, technically demanding level. A big payoff from this approach is an understanding of how security claims can be proved and what that entails in terms of formal security and correctness of related algorithms and crypto-systems. The text is a difficult read for undergraduates and requires some experience with understanding and analyzing algorithms, some knowledge of elementary discrete mathematics, and some proficiency with fundamental proof techniques. In addition, a background including discrete probability theory and basic algebraic methods would be very helpful.

2 Summary of Contents: Chapters 1-15

Following a brief introduction, CryptoSchool begins with Chapter 2 describing a central symmetric cryptosystem, the Advanced Encryption Standard (AES). It explains the algorithm by breaking it up into a series of mostly byte-sized algebraic operations. It then briefly introduces asymmetric encryption and the RSA algorithm and Diffie-Hellman key exchange, followed by a quick explanation of block and stream ciphers and secret sharing. The book then proceeds with three chapters on important asymmetric cryptosystems; (i). RSA again, in more detail with more background and analysis (Chapter 3), (ii). encryption based on group theory and the discrete logarithms problem (Chapter 4), and (iii). newer algebraic cryptosystems based on elliptic curves (Chapter 5). Chapter 6 discusses the method of differential cryptanalysts for symmetric encryption, specifically the analysis of a small version of AES, and a brief consideration of some other related cryptanalytic tools. Chapters 7 and 8 discuss two key cryptographic techniques and their applications; hash functions including several different examples and implementations, and digital signatures, specifically ElGamal, Schnorr and GHR signature schemes. Chapter 9 is a central chapter concerning the axiomatic security of cryptosystems, explores the the different bases upon which security is based, and introduces key ideas of security reductions for encryption and for signatures.

After these fundamental chapters comes a series of chapters on more advanced, currently active cryptographic topics. These include random and pseudorandom number generators, distinguishers and predictors, lattice based cryptography, zero knowledge proof systems and a chapter on quantum computation leading to a (very) brief discussion of quantum methods in cryptography. These chapters are quite formal and heavily mathematical. Many details are given, often with complete proofs and careful examples. The methods discussed are carefully motivated and comprehensiveness (is this a word?) is striven for. The topics for these more advanced chapters are well chosen to emphasize more recent theoretical constructs in the field. Other active topics could have been included, among them; database and data structure security, formal theoretical privacy, and ideas of program and function obfuscation. The choices of topics made here are ultimately reasonable and consistent with the theme of concentrating on the more conceptual and theoretical.

All of the cryptography chapters end with copious notes on the history and background of the results in the text, and often provide pointers to related results elsewhere. Each chapter also contains numerous varied and interesting homework problems. The problems are very diverse and often challenging, ranging from the working out of various examples, sometimes via programming, to challenging theoretical proofs.

Chapter 15, at the end of the text, is a carefully written and organized extended chapter on the algebraic background material needed for the book, and is meant to be read as needed. This long chapter includes sections on basic algebras and sections on linear algebra and on group theory, and on algorithms and complexity. There is also a small section on finite probability, but here more information would be welcome and really is needed as elementary and occasionally more advanced probability is used throughout. Those without a good probabilistic background will struggle to carry out proofs that depend on formal definitions many of which are fundamentally probabilistic.

3 Summary of Contents: Chapters A-J

The 10 varied chapters on cryptologic history barely scratch the surface of this large and diverse topic. Each of the history chapters (Chapters A-J) gives a brief view of one aspect of the cryptographic history which preceded the modern era, and each presents a different take on this history. The chapters start with classical cryptography and end with cryptography of WWII, the ENIGMA ciphers and their cryptanalysis, and the beginning of electro-mechanical devices taking over the computational work. In between (in Chapters C through I) there is a selection of topics, some based on examples of cryptography in important historical times, others on specific cryptographers or cryptosystems, and still others on particular types of codes such as transposition codes, codebooks, and steganography.

Each of the history chapters tells a story of its own. They are interesting and varied in their style, and quite densely written. This is especially true when there is photocopied material given with English translation and illustrating one-off cryptographic examples. Several of the less ancient historical examples, e.g Kasiski's cryptanalysis of Vigenère ciphers in Chapter C, are very well-described as are the interesting statistical methods of Friedman and others. Chapter J on World War II codes gives a strong though brief overview of some of the rotor machines and their encoding coding and decoding methods, and of the mathematicians involved in this work. Some of these chapters seem uneven with some extremely short sections interspersed with longer historical cryptographic exploits or very detailed examples, sometimes quite hard to follow in their examples from the more distant past. Here the full cryptographic content was hard to discern and understand as it is not explained but instead given by example. I particularly found this true in the chapter D on nomenclators and codebooks and in Chapter G where the cryptographic usage of 5 historical figures is described in historical terms and illustrations. At times it is hard to fully discern the details of the cryptographic methods or their historical importance as these historic episodes or coding examples appear between the chapters of modern methods.

4 Highlights

Here we mention a few of the many very positive features we find in this text. The mathematical presentation of results is notable, including the consistent clarity of results and the considerable time and care taken for proofs of correctness of the cryptosystems, and of the cryptanalytic results. This includes detailed discussion and proofs of security, and of the many tools and techniques that are necessary for these results. In modern cryptography precise definitions are key, and often are the crucial feature for enabling proofs of the desired correctness conclusions and security properties. These definitions are often, by necessity, extremely complex and take considerable getting used to. Getting their details and precise specification to fit perfectly with the cryptographic consequences make all the difference in the provable correctness and security of the methods. This book consistently makes strong attempts at explaining this formalism and often succeeds, though a good deal of mathematical background and experience will be required to fully understand the more difficult details.

This book is beautifully constructed, both physically and in its organization with well-designed illustrations and careful typesetting. Considerable care was taken by the author and the publisher (Springer) to construct, and decorate this very substantial and impressive volume. Each chapter begins with close to a full page of quotes, many with English translations. These quotes are very varied in their content, history and in their technical depth. The historical chapters each begin with an intricate ornament and a decorated initial, and each ends with another, smaller ornament, all fully annotated at the end of the book. There are numerous images and figures embedded throughout the text, many in color and finely reproduced. Overall

they are interesting, diverting and fun, though at times (particularly when in translation) somewhat obscure in their relevancy to the chapter to come. Clearly the author has taken a great deal of care in gathering these details, and they strongly add to the overall texture of the book.

Overall this is a unique and idiosyncratic text. It is unconventional in that the historical chapters that are intermixed with the cryptography provide a break from the technical material, and give a broader context to the results. These chapters are certainly not meant to be read linearly but do serve to give further perspective, as well as some connection with the history of codes to the radically new methods, tools and perspectives of current practice. I'll be very happy to have CryptoSchool on my bookshelf. It will be most useful as a resource book and as a sourcebook for this material in the future. It will as well be a valuable exemplar of a book with careful organization and an interesting juxtaposition of topics, and as a resource for many discussions on cryptology, both technical and historical.

Review of[4]
R for Data Science:
Import, Tidy, Transform, Visualize, and Model Data
by Hadley Wickham and Garrett Grolemund
Published by O'Reilly Media, 2016
552 pages, Softcover, $33.27 (ISBN 978-1491910399)

Review by
Allan M. Miller[5] (allan.m.miller@berkeley.edu)

1 Introduction

R for Data Science (R4DS) is the much anticipated book by the most prolific, if not leading figure in the worldwide R data science community, Hadley Wickham, and his RStudio colleague Garrett Grolemund. It is an introduction to data science using R, providing simultaneously both an introduction to the discipline of data science, and the R programming language.

The stated goal of the book is "to help you learn the most important tools in R that will allow you to do data science. After reading this book, youll have the tools to tackle a wide variety of data science challenges, using the best parts of R."

There are several other introductory data science books using R that are currently available. What makes this book different and so significant? R4DS incorporates a new approach to learning R and data science, based on (1) a functional programming approach to learning and writing R, which greatly simplifies learning and coding in R, (2) an early and strong emphasis on the use of data visualization in the data science process, and (3) use and extensive coverage of the "tidyverse," a set of R packages, many authored by Hadley Wickham, that provide a standard data format and functions for solving many of the most common data science problems. Using this approach, the book provides highly effective, substantial coverage of data science and R.

According to the authors, the book is specifically designed to provide effective, high quality ready-made tools for performing the basic data science tasks:

Import: bringing data into R, whether it be in text file, database, Internet API format, and loaded into an R data frame (actually, a tibble, a new and improved version of the base R data frame - see below).

Tidy: putting data in a standard, consistent format (a data frame where each column is a variable, and each row is an observation) for easy use with tidyverse functions.

Transform: filtering datasets to include observations of interest, removing unneeded columns, adding necessary new computed columns, and calculating summary statistics.

The tidying and transforming tasks together constitute what the authors refer to as data wrangling.

[4] ©2017, Allan M. Miller

[5] Allan Miller, Ph.D., is a San Francisco Bay Area based practicing freelance Data Scientist in the areas of economic, marketing, health care, and environmental analytics. He teaches highly popular courses in R and data science for the U.C. Berkeley Extension program, where he serves on their Data Science Advisory Committee. Dr. Miller is also the co-organizer of the Berkeley R users group.

The next two tasks are iterative, one often goes back and forth between them: visualization and modeling.

Visualization: essentially using graphics - plots of data to gain knowledge, raise new questions about the data, or discover that the data is inadequate to solve the problem at hand.

Models: described as complementary tools to visualization, used to answer precisely formulated questions.

Communication: communicating your results to others in the form of documents and visualizations, both statically and dynamically, in the form of documents and/or dynamic interactive data visualization and reporting applications deployed on the web.

2 Summary

The book consists of five parts reflecting the general steps in the data science process:

1. Explore
2. Wrangle
3. Program
4. Model
5. Communicate

Part 1, Explore consists of six chapters, focusing on data visualization, data transformation, exploratory data analysis and workflow.

The book begins with Chapter 1, Data Visualization with ggplot2, Hadley Wickham's powerful and highly popular R plotting and data visualization package. Most R books begin with coverage of the basic syntax of the (base) R language. By starting with data visualization using an R package, R4DS makes a radical break from this approach:

"Starting with data ingest and tidying is suboptimal because 80% of the time it's routine and boring, and the other 20% of the time it's weird and frustrating. That's a bad place to start learning a new subject! Instead, we'll start with visualization and transformation of data that's already been imported and tidied. That way, when you ingest and tidy your own data, your motivation will stay high because you know the pain is worth it."

By jumping right into using a high-level data visualization package, `ggplot2`, the authors avoid having to cover the details of base R language syntax before being able to discuss interesting and substantial data science problems.

Chapter 3, Data Transformation with `dplyr` introduces the powerful `dplyr` package for data manipulation, which is the core of the tidyverse ecosystem. The chapter covers `dplyr`'s basic functions for data manipulation: filter to filter rows (observations), arrange to sort observations by one or more columns, select to select or retain columns, and mutate, used to add new variables. Data grouping is performed by using the `group_by` function, which causes the verb transformations discussed above to be applied to groups. Each `dplyr` function represents a single data transformation verb or action: e.g., select rows, sort (arrange) a data frame, add (mutate) variables. In this way, a sequence of data transformations can be described in terms of english-like verbs comprising the operations. Approximately two thirds of the way through Chapter 3, the authors switch from using standalone `dplyr` verb functions to a more natural functional piping syntax,

where a group of verb functions can be used in a piped sequence of operations. This is a very significant part of the tidyverse approach, more is covered on this as part of the discussion of Chapter 14 Pipes with `magrittr` below.

Chapter 5, Exploratory Data Analysis (EDA), uses `dplyr` and `ggplot2` to perform EDA. The approach taken is to generate questions about the data, seek answers to these questions using `ggplot2` visualizations, `dplyr` transformations, and modeling, and then using the results in an iterative process to refine the questions and to generate new questions. Covered specifically are: visualizing variation within variables, variation between variables (both for discrete and continuous variables), and discovering missing values. The chapter includes a section Patterns and Models, which introduces modeling through analyzing patterns discovered in the EDA process described above.

Part 2, Wrangle covers, as explained above, importing and tidying data.

Chapter 7, Tibbles, introduces the tibble data structure defined in tidyverse package `tibble`, which is derived from the base R data frame. A tibble is a data frame with slightly different default behaviors, such as improved console display, extended creation syntax, and default character (versus factor) representation of strings.

Chapter 8, Data Import with `readr`, covers the `readr` package for reading data into R data frames or tibbles. `readr` provides multiple specialized functions for reading data into R. Base R input functions such as `read.csv` are replaced with more flexible specialized `readr` counterparts such as `read_csv`, `read_tsv`, etc. `readr` functions are highly optimized, running in some cases many times faster than their base R counterparts. Most of what is needed for data input can be found in the `readr` package.

Chapter 9, Tidy Data with `tidyr`, introduces the `tidyr` package, also part of the tidyverse, used to format data into a standard "tidy" format where each observation is a single row, and each variable is a single column. The goal is to place R in a standard, uniform (tidy) format that is expected by convention to be operated on by other tidyverse packages. `tidyr` provides functions to perform common transformations, such as converting data frames from wide to long format (gather function), or long to wide (spread function). The chapter also discusses other common transformative functions such as `separate`, used to split columns, and `unite`, which is used to combine them. Dealing with the common problem of missing values is also discussed, along with a substantial example at the end of the chapter demonstrating most of the concepts presented in the chapter.

Chapter 10, Relational Data with `dplyr`, jumps back into `dplyr`, but introduces functions for joining multiple tables. Much of the functionality of SQL joins is included in `dplyr`, such as mutating and filtering joins, as well as operations encapsulating pretty much all types of SQL joins. Since another tidyverse package, `dbplyr`, has the ability to connect to databases and other types of data sources, `dplyr` provides most of the functionality of SQL directly in R, using the intuitive verb-function piping syntax. Chapter 10 provides in-depth, systematic coverage of these concepts.

Chapter 11, Strings with `stringr`, discusses the tidyverse `stringr` package, which encapsulates a powerful set of functions for string manipulation. Virtually all of the string manipulation functionality present in base R is included in a single package, often with with greater flexibility and power, including subsetting, locales, pattern matching using regular expressions, and replacement.

Chapter 12, Factors with `forcats`, discusses a set of functions for useful manipulation of base R factors, such as modifying factor levels and order, that are used to represent categorical variables.

Chapter 13, Dates and Times with `lubridate`, presents the `lubridate` package containing everything you might need to do with date and time objects, including alternative date and time creation methods, decomposition, working with time zones and time spans, date and time arithmetic. This package predates

the full tidyverse, and includes as complete and intuitive an implementation of date and time operations as exists in any programming language.

Part 3, Program, puts forward programming as a vehicle of communication for instructing computers and communicating meaning to human collaborators. The authors state:

"Writing clear code is important so that others (like future-you) can understand why you tackled an analysis in the way you did. That means getting better at programming also involves getting better at communicating." The means for doing this using the tidyverse is presented in Chapters 14-17. This is the part of the book that also covers the parts of base R needed to program in R. The fact that they are introduced three quarters of the way through the book reflects how different this approach to R is (base R is covered in-depth at the start of most R books).

Chapter 14 covers the pipe operator, `%>%`, defined in the `magrittr` package and used throughout `dplyr` and other tidyverse packages. The piping syntax starts with a data frame (or tibble) on the left hand side of a pipe, and "pipes" it to the right to a (usually `dplyr`) verb function, such as filter, select, or mutate, which returns a tibble. This, in turn, can serve as the left hand side input to another pipe operator, sent to yet another `dplyr` function. A "piped chain" of data plus verbs encapsulates the data transformation process in a compact, intuitive, english-like way. This approach to programming is perhaps the most revolutionary part of the tidyverse approach to doing data science. The chapter covers how pipes work, and when, or not, to use them.

Chapter 15, Functions, provides in-depth coverage of R functions. In addition to the basic syntax for creating and using functions, the chapter covers best coding practices for how and when to use them. Other best coding practices, such as how and when to use comments, are also covered as are other important parts of R such as conditions (branching), function arguments syntax, and the R environment.

Chapter 16, Vectors covers the basic building block of R, the vector data structure. The approach in doing this is, as the authors state:

"So far this book has focused on tibbles and packages that work with them. But as you start to write your own functions, and dig deeper into R, you need to learn about vectors, the objects that underlie tibbles. If you've learned R in a more traditional way, you're probably already familiar with vectors, as most R resources start with vectors and work their way up to tibbles. I think it's better to start with tibbles because they're immediately useful, and then work your way down to the underlying components."

The chapter covers the syntax for creating vectors of the basic R data types (logical, numeric, character), accessing individual, and groups (slices) of elements, as well as R's recycling rules. The tidyverse `purrr` package is introduced, using its type testing functions (e.g., `is_numeric`) as an alternative to the corresponding base R functions.

The chapter also covers R lists. R lists can store any type of object, and unlike vectors, list elements can be heterogeneous (multiple types) and non-atomic, such as vectors. In fact, lists in R are implemented recursively, every element of a list is itself a list. R lists, like hashes or dictionaries in other languages, are thus ideal for representing non-symmetric and hierarchical data.

Chapter 17, Interaction with `purrr`, goes deeper into the `purrr` tyidverse package, used to write code that implements two iteration paradigms: imperative programming (e.g., for and while loops), and functional programming. `purrr` provides an alternative to the apply class of base R functions, one for each type of output (e.g., `map_dbl` or `map_chr`). Like the apply class of functions, most `purrr` functions take a vector as input, applies a function to each piece, and returns a vector result. `purr` also provides some shortcuts for simplifying the creation of anonymous functions, and applying functions to groups of objects (i.e., shortcuts). Implementing functional programming in this way avoids the necessity of knowing the

syntax and operation of base R's multiple apply class of functions.

`purrr` also supplies a function, `safely`, for dealing with situations where `purrr` function calls cause failures. When a function name is passed as an argument to `safely`, it returns a modified version that does not trigger exceptions, but instead always return a two-element list with a result and error object. This approach is like the base R `try` function, but more consistent in that it always returns the same result.

`purrr` includes additional, more advanced capabilities, including the ability to map function over multiple arguments, invoking different functions over data, applying functions with side effects (`walk`), predicate functions, and the ability to reduce complex lists to a simple list (`reduce` and `accumulate`).

In sum, `purrr` extends base R's functional paradigm by providing groups of functions that solve the basic types of iteration problems.

Part 4, Model, introduces new tools to facilitate building and understanding multiple predictive models. The approach is to use models "as a tool for exploration," and not for hypothesis confirmation, which, the authors confirm, is unusual. In doing so, the authors use models to partition data into patterns and residuals, exploring patterns in the residuals in order to gain insight into the data, and make corresponding changes to the model.

Chapter 18, Model Basics with `modelr` introduces the `modelr` package, designed to do exactly this.

Modeling involves first, choosing a family of models, e.g. linear or quadratic, to "express a precise, but generic, pattern that you want to capture." Then a fitted model is generated and the residuals are analyzed to detect patterns, and the model is adjusted accordingly.

The authors take the reader through several examples using a small computer generated dataset. The parameters of multiple linear models are randomly generated and fitted to the dataset. The residuals are plotted, using `modelr` convenience functions, and the results plotted and evaluated. First, the best fitting (with the lowest RMS residuals) parameters are selected, then a grid evenly spaced parameters are generated and searched, and finally, numerical methods are used to estimate the best fitting set of parameters. In each case the model results and residuals are plotted, and the results analyzed. The method is extended to include categorical variables, and various interactions between continuous and categorical variables.

The use of this method is continued through Chapter 19, Model Building, using real datasets and more complex models. As patterns in the residuals are identified visually, new parameters are added to the model to account for these.

The result is a highly visual, interactive, and intuitive approach to the modeling process, which is then leveraged in Chapter 20, Many Models, using the `purrr` and `broom` packages to increase the scale of the modeling process, generate more models and analyze more results without commensurate increases in the amount of code and effort necessary to produce those results.

Part 5, Communicate focuses on presenting results to clients by means of `RMarkdown` documents and more advanced treatment of `ggplot2` graphics, introduced in Chapter 1.

Chapter 21, R Markdown introduces `RMarkdown`, a version of standard `Markdown` adapted for R. `RMarkdown` documents are composed of simplified HTML-like (`RMarkdown`) tags, and can be processed to generate formatted HTML, PDF, Microsoft Word, or LaTeX, including embedded graphics, using the `knitR` package and RStudio IDE.

Whereas chapter 1 focuses on the basics of building plots for internal use using `ggplot2`, Chapter 22, Graphics for Communication with `ggplot2`, focuses on developing plots for external consumption. It shows how to enhance and customize `ggplot` plots using labels, annotations, customize scales, and using `ggplot2` themes, to customize background colors, grid lines, margins and other non-data elements of plots.

It provides an excellent followup to Chapter 1, effectively covering the the most commonly used methods for customizing and enhancing plots that are available in `ggplot2`.

Chapter 23 RMarkdown Formats, covers the output formats that can be generated from `RMarkdown` documents, including notebooks, where code and output are combined within a document, presentations, dashboards, interactive documents including leaflet maps, and web-based interactive data visualizations using the `shiny` package. The variety of these methods discussed in the chapter is impressive.

Chapter 24 R Markdown Worksflow discusses best practices for using the above tools for communicating results. An analogy is drawn between `RMarkdown` notebooks and classic laboratory notebooks, where results are documented, journaled, and archived even if they are not included in the final results. This short chapter provides valuable advice such as naming rules, project organization and coding practices for how to manage projects in a way that makes `RMarkdown` notebooks readable, verifiable and reproducible.

3 Opinion

R4DS represents a significant breakthrough in the methods used for both learning and using R for data science. The functional approach to programming in R that R4DS uses, based on the `dplyr` package, pipes, and the tidyverse, greatly simplifies R programming, while at the same time greatly enhances its power. It opens the possibility of learning R for data science in a way that circumvents having to learn a large amount of R language syntax up front. In addition to this, the tidyverse provides a complete, well-crafted set of packages (libraries) for solving the most common types of data science problems based on a standard, uniform (tidy) data format. Part 4 uses a novel and promising approach to modeling, made possible by the tidyverse. It warrants a book dedicated specifically to the subject.

I have used R4DS in a classroom setting and can testify that it can serve effectively as both an introductory text for beginners who wish to learn R for data science, as well as an advanced introduction to the functional tidyverse approach for those already familar with R. This book has a place on every serious data scientist's bookshelf.