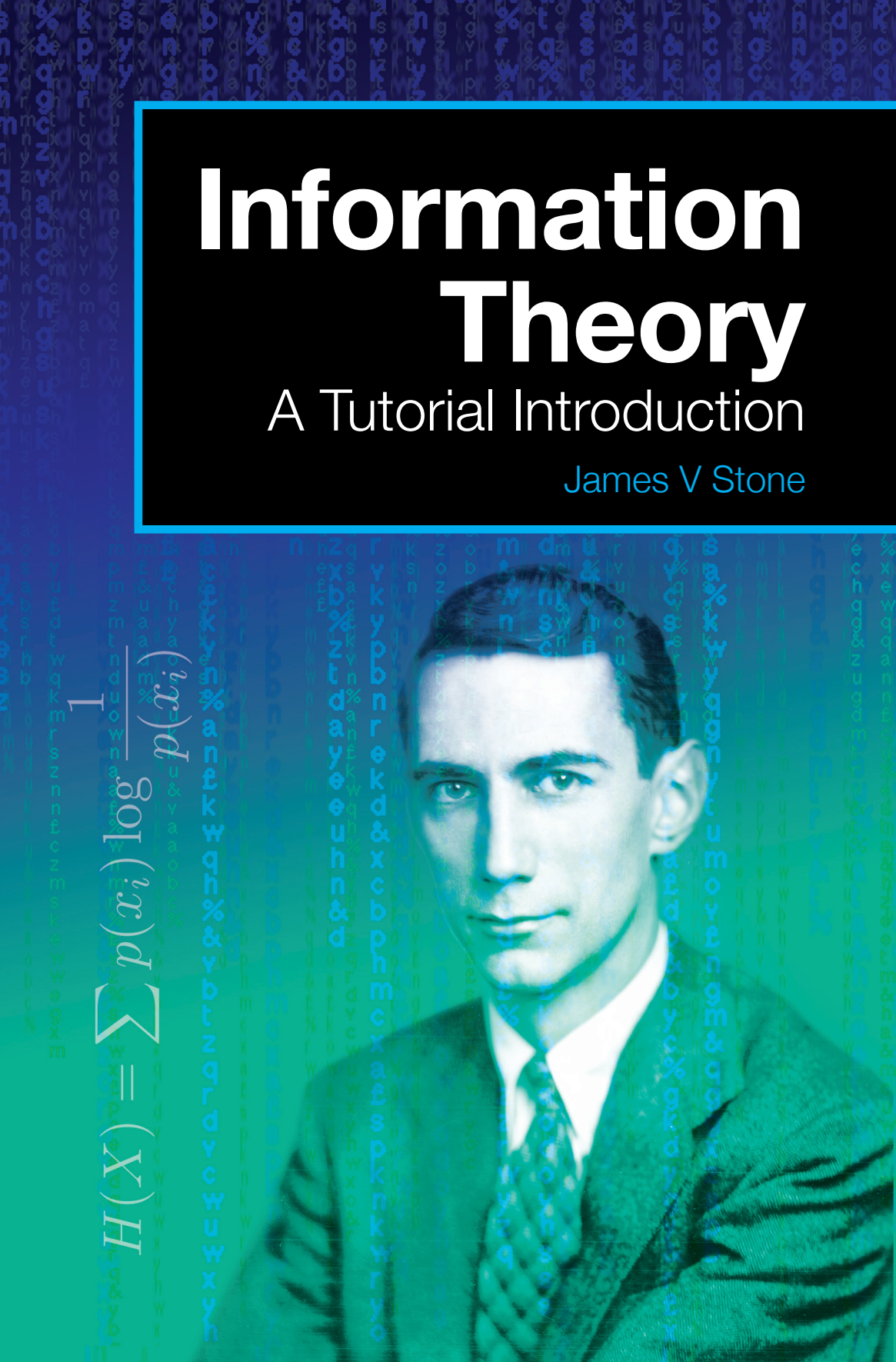
Information Theory

A Tutorial Introduction

James V Stone



Reviews of Information Theory

"Information lies at the heart of biology, societies depend on it, and our ability to process information ever more efficiently is transforming our lives. By introducing the theory that enabled our information revolution, this book describes what information is, how it can be communicated efficiently, and why it underpins our understanding of biology, brains, and physical reality. Its tutorial approach develops a deep intuitive understanding using the minimum number of elementary equations. Thus, this superb introduction not only enables scientists of all persuasions to appreciate the relevance of information theory, it also equips them to start using it. The same goes for students. I have used a handout to teach elementary information theory to biologists and neuroscientists for many years. I will throw away my handout and use this book."

Simon Laughlin, Professor of Neurobiology, Fellow of the Royal Society,
Department of Zoology, University of Cambridge, England.

"This is a really great book it describes a simple and beautiful idea in a way that is accessible for novices and experts alike. This "simple idea" is that information is a formal quantity that underlies nearly everything we do. In this book, Stone leads us through Shannons fundamental insights; starting with the basics of probability and ending with a range of applications including thermodynamics, telecommunications, computational neuroscience and evolution. There are some lovely anecdotes: I particularly liked the account of how Samuel Morse (inventor of the Morse code) pre-empted modern notions of efficient coding by counting how many copies of each letter were held in stock in a printer's workshop. The treatment of natural selection as "a means by which information about the environment is incorporated into DNA" is both compelling and entertaining. The substance of this book is a clear exposition of information theory, written in an intuitive fashion (true to Stone's observation that "rigour follows insight"). Indeed, I wish that this text had been available when I was learning about information theory. Stone has managed to distil all of the key ideas in information theory into a coherent story. Every idea and equation that underpins recent advances in technology and the life sciences can be found in this informative little book."

Professor Karl Friston, Fellow of the Royal Society. Scientific Director of the Wellcome Trust Centre for Neuroimaging,
Institute of Neurology, University College London.

Reviews of Bayes' Rule: A Tutorial Introduction

"An excellent book ... highly recommended."
CHOICE: Academic Reviews Online, February 2014.

"Short, interesting, and very easy to read, Bayes' Rule serves as an excellent primer for students and professionals ... "
Top Ten Math Books On Bayesian Analysis, July 2014.

"An excellent first step for readers with little background in the topic."
Computing Reviews, June 2014.

"The author deserves a praise for bringing out some of the main principles of Bayesian inference using just visuals and plain English. Certainly a nice intro book that can be read by any newbie to Bayes."
https://rkbookreviews.wordpress.com/, May 2015.

From the Back Cover

"Bayes' Rule explains in a very easy to follow manner the basics of Bayesian analysis."
Dr Inigo Arregui, Ramon y Cajal Researcher, Institute of Astrophysics, Spain.

"A crackingly clear tutorial for beginners. Exactly the sort of book required for those taking their first steps in Bayesian analysis."
Dr Paul A. Warren, School of Psychological Sciences, University of Manchester.

"This book is short and eminently readable. It introduces the Bayesian approach to addressing statistical issues without using any advanced mathematics, which should make it accessible to students from a wide range of backgrounds, including biological and social sciences."
Dr Devinder Sivia, Lecturer in Mathematics, St John's College, Oxford University, and author of Data Analysis: A Bayesian Tutorial.

"For those with a limited mathematical background, Stone's book provides an ideal introduction to the main concepts of Bayesian analysis."
Dr Peter M Lee, Department of Mathematics, University of York. Author of Bayesian Statistics: An Introduction.

"Bayesian analysis involves concepts which can be hard for the uninitiated to grasp. Stone's patient pedagogy and gentle examples convey these concepts with uncommon lucidity."
Dr Charles Fox, Department of Computer Science, University of Sheffield.

Information Theory

A Tutorial Introduction

James V Stone

Title: Information Theory: A Tutorial Introduction
Author: James V Stone

First Edition, 2015.
Typeset in LaTeX 2_ε.
Cover design: Stefan Brazzo.
Second printing.

ISBN 978–0–9563728–5–7

The front cover depicts Claude Shannon (1916-2001).

For Nikki

Suppose that we were asked to arrange the following in two categories – *distance, mass, electric force, entropy, beauty, melody.* I think there are the strongest grounds for placing entropy alongside beauty and melody ...

Eddington A, The Nature of the Physical World, 1928.

Contents

Preface

This book is intended to provide a coherent and succinct account of information theory. In order to develop an intuitive understanding of key ideas, new topics are first presented in an informal tutorial style before being described more formally. In particular, the equations which underpin the mathematical foundations of information theory are introduced on a need-to-know basis, and the meaning of these equations is made clear by explanatory text and diagrams.

In mathematics, rigour follows insight, and not *vice versa*. Kepler, Newton, Fourier and Einstein developed their theories from deep intuitive insights about the structure of the physical world, which requires, but is fundamentally different from, the raw logic of pure mathematics. Accordingly, this book provides insights into *how* information theory works, and *why* it works in that way. This is entirely consistent with Shannon's own approach. In a famously brief book, Shannon prefaced his account of information theory for continuous variables with these words:

> We will not attempt in the continuous case to obtain our results with the greatest generality, or with the extreme rigor of pure mathematics, since this would involve a great deal of abstract measure theory and would obscure the main thread of the analysis. ... The occasional liberties taken with limiting processes in the present analysis can be justified in all cases of practical interest.
> Shannon C and Weaver W, 1949[50].

In a similar vein, Jaynes protested that:

> Nowadays, if you introduce a variable x without repeating the incantation that it is some set or 'space' X, you are accused of dealing with an undefined problem ...
> Jaynes ET and Bretthorst GL, 2003[26].

Even though this is no excuse for sloppy mathematics, it is a clear recommendation that we should not mistake a particular species of pedantry for mathematical rigour. The spirit of this liberating and somewhat cavalier approach is purposely adopted in this book, which is intended to provide insights, rather than incantations, regarding how information theory is relevant to problems of practical interest.

MatLab and Python Computer Code

It often aids understanding to be able to examine well-documented computer code which provides an example of a particular calculation or method. To support this, MatLab and Python code implementing key information-theoretic methods can be found online. The code also reproduces some of the figures in this book.
MatLab code can be downloaded from here:
http://jim-stone.staff.shef.ac.uk/BookInfoTheory/InfoTheoryMatlab.html
Python code can be downloaded from here:
http://jim-stone.staff.shef.ac.uk/BookInfoTheory/InfoTheoryPython.html

PowerPoint Slides of Figures

Most of the figures used in this book are available for teaching purposes as a pdf file and as PowerPoint slides. These can be downloaded from
http://jim-stone.staff.shef.ac.uk/BookInfoTheory/InfoTheoryFigures.html

Corrections

Please email corrections to j.v.stone@sheffield.ac.uk.
A list of corrections can be found at
http://jim-stone.staff.shef.ac.uk/BookInfoTheory/Corrections.html

Acknowledgments

Thanks to John de Pledge, John Porrill, Royston Sellman, and Steve Snow for interesting discussions on the interpretation of information theory, and to John de Pledge for writing the Python code.

For reading draft versions of this book, I am very grateful to Óscar Barquero-Pérez, Taylor Bond, David Buckley, Jeremy Dickman, Stephen Eglen, Charles Fox, Nikki Hunkin, Danielle Matthews, Guy Mikawa, Xiang Mou, John de Pledge, John Porrill, Royston Sellman, Steve Snow, Tom Stafford, Paul Warren and Stuart Wilson. Shashank Vatedka deserves a special mention for checking the mathematics in a final draft of this book. Thanks to Caroline Orr for meticulous copy-editing and proofreading.

Online code for estimating the entropy of English was adapted from code by Neal Patwari (MatLab) and Clément Pit-Claudel (Python).

For permission to use the photograph of Claude Shannon, thanks to the Massachusetts Institute of Technology.

Jim Stone.

Chapter 1

What Is Information?

> Most of the fundamental ideas of science are essentially simple, and may, as a rule, be expressed in a language comprehensible to everyone.
> Einstein A and Infeld L, 1938.

1.1. Introduction

The universe is conventionally described in terms of physical quantities such as mass and velocity, but a quantity at least as important as these is *information*. Whether we consider computers[30], evolution[2;19], physics[15], artificial intelligence[9], quantum computation[46], or the brain[17;43], we are driven inexorably to the conclusion that their behaviours are largely determined by the way they process information.

Figure 1.1. Claude Shannon (1916-2001).

In 1948, Claude Shannon published a paper called *A Mathematical Theory of Communication*[48]. This paper heralded a transformation in our understanding of information. Before Shannon's paper, information had been viewed as a kind of poorly defined miasmic fluid. But after Shannon's paper, it became apparent that information is a well-defined and, above all, *measurable* quantity.

Shannon's paper describes a subtle theory which tells us something fundamental about the way the universe works. However, unlike other great theories such as the Darwin–Wallace theory of evolution, information theory is not simple, and it is full of caveats. But we can disregard many of these caveats provided we keep a firm eye on the physical interpretation of information theory's defining equations. This will be our guiding principle in exploring the theory of information.

1.2. Information, Eyes and Evolution

Shannon's theory of information provides a mathematical definition of information, and describes precisely how much information can be communicated between different elements of a system. This may not sound like much, but Shannon's theory underpins our understanding of how signals and noise are related, and why there are definite limits to the rate at which information can be communicated within *any* system, whether man-made or biological. It represents one of the few examples of a single theory creating an entirely new field of research. In this regard, Shannon's theory ranks alongside those of Darwin–Wallace, Newton, and Einstein.

When a question is typed into a computer search engine, the results provide useful information but it is buried in a sea of mostly useless data. In this internet age, it is easy for us to appreciate the difference between information and data, and we have learned to treat the information as a useful 'signal' and the rest as distracting 'noise'. This experience is now so commonplace that technical phrases like 'signal to noise ratio' are becoming part of everyday language. Even though most people are unaware of the precise meaning of this phrase, they have an intuitive grasp of the idea that 'data' means a combination of (useful) signals and (useless) noise.

The ability to separate signal from noise, to extract information from data, is crucial for modern telecommunications. For example, it allows a television picture to be compressed to its bare information bones and transmitted to a satellite, then to a TV, before being decompressed to reveal the original picture on the TV screen.

This type of scenario is also ubiquitous in the natural world. The ability of eyes and ears to extract useful signals from noisy sensory data, and to package those signals efficiently, is the key to survival[51]. Indeed, the *efficient coding hypothesis*[5;8;43;55] suggests that the evolution of sense organs, and of the brains that process data from those organs, is primarily driven by the need to minimise the energy expended for each bit of information acquired from the environment. More generally, a particular branch of brain science, *computational neuroscience*, relies on information theory to provide a benchmark against which the performance of neurons can be objectively measured.

On a grander biological scale, the ability to separate signal from noise is fundamental to the Darwin–Wallace theory of evolution by natural selection[12]. Evolution works by selecting the individuals best suited to a particular environment so that, over many generations, information about the environment gradually accumulates within the gene pool. Thus, natural selection is essentially a means by which information about the environment is incorporated into DNA (deoxyribonucleic acid). And it seems likely that the rate at which information is incorporated into DNA is accelerated by an age-old biological mystery, sex. These and other applications of information theory are described in Chapter 9.

1.3. Finding a Route, Bit by Bit

Information is usually measured in *bits*, and one bit of information allows you to choose between two equally probable alternatives. The word bit is derived from *bi*nary dig*it* (i.e. a zero or a one). However, as we shall see, bits and binary digits are fundamentally different types of entities.

Imagine you are standing at the fork in the road at point A in Figure 1.2, and that you want to get to the point marked D. Note that this figure represents a bird's-eye view, which you do not have; all you have

is a fork in front of you, and a decision to make. If you have no prior information about which road to choose then the fork at A represents two equally probable alternatives. If I tell you to go left then you have received one bit of information. If we represent my instruction with a *binary digit* (0=left and 1=right) then this binary digit provides you with one bit of information, which tells you which road to choose.

Now imagine that you stroll on down the road and you come to another fork, at point B in Figure 1.2. Again, because you have no idea which road to choose, a binary digit (1=right) provides one bit of information, allowing you to choose the correct road, which leads to the point marked C.

Note that C is one of four possible interim destinations that you could have reached after making two decisions. The two binary digits that allow you to make the correct decisions provided two bits of information, allowing you to choose from four (equally probable) possible alternatives; 4 happens to equal $2 \times 2 = 2^2$.

A third binary digit (1=right) provides you with one more bit of information, which allows you to again choose the correct road, leading to the point marked D.

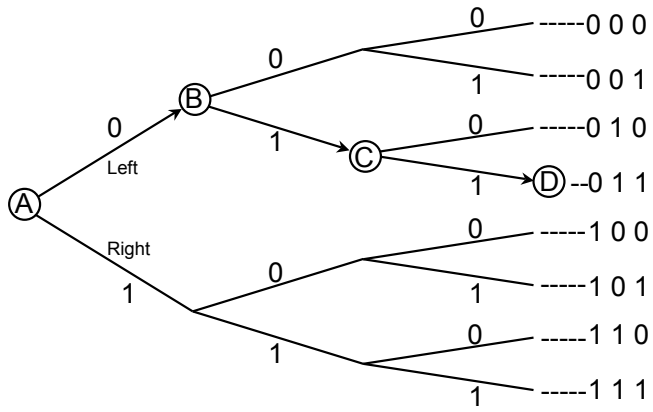


Figure 1.2. How many roads must a man walk down? For a traveller who does not know the way, each fork in the road requires one bit of information to make a correct decision. The 0s and 1s on the right-hand side summarise the instructions needed to arrive at each destination; a left turn is indicated by a 0 and a right turn by a 1.

There are now eight roads you could have chosen from when you started at A, so three binary digits (which provide you with three bits of information) allow you to choose from eight equally probable alternatives; 8 happens to equal $2 \times 2 \times 2 = 2^3 = 8$.

The decision taken at A excluded half of the eight possible destinations shown in Figure 1.2 that you could have arrived at. Similarly, the decision taken at each successive fork in the road halved the number of remaining possible destinations.

A Journey of Eight Alternatives

Let's summarise your journey in terms of the number of equally probable alternatives:

> If you have 1 bit of information then you can choose between 2 equally probable alternatives (i.e. $2^1 = 2$).
>
> If you have 2 bits of information then you can choose between 4 equally probable alternatives (i.e. $2^2 = 4$).
>
> Finally, if you have 3 bits of information then you can choose between 8 equally probable alternatives (i.e. $2^3 = 8$).

We can restate this in more general terms if we use n to represent the number of forks, and m to represent the number of final destinations. If you have come to n forks, then you have effectively chosen from

$$m \quad = \quad 2^n \text{ final destinations.} \tag{1.1}$$

Because the decision at each fork requires one bit of information, n forks require n bits of information, which allow you to choose from 2^n equally probable alternatives.

There is a saying that "a journey of a thousand miles begins with a single step". In fact, a journey of a thousand miles begins with a single decision: the direction in which to take the first step.

Key point. One bit is the amount of information required to choose between two *equally probable* alternatives.

Binary Numbers

We could label each of the eight possible destinations with a decimal number between 0 and 7, or with the equivalent *binary number*, as in Figure 1.2. These decimal numbers and their equivalent binary representations are shown in Table 1.1. Counting in binary is analogous to counting in decimal. Just as each decimal digit in a decimal number specifies how many 1s, 10s, 100s (etc) there are, each binary digit in a binary number specifies how many 1s, 2s, 4s (etc) there are. For example, the value of the decimal number 101 equals the number of 100s (i.e. 10^2), plus the number of 10s (i.e. 10^1), plus the number of 1s (i.e. 10^0):

$$(1 \times 100) + (0 \times 10) + (1 \times 1) \quad = \quad 101. \tag{1.2}$$

Similarly, the value of the binary number 101 equals the number of 4s (i.e. 2^2), plus the number of 2s (i.e. 2^1), plus the number of 1s (i.e. 2^0):

$$(1 \times 4) + (0 \times 2) + (1 \times 1) \quad = \quad 5. \tag{1.3}$$

The binary representation of numbers has many advantages. For instance, the binary number that labels each destination (e.g. 011) explicitly represents the set of left/right instructions required to reach that destination. This representation can be applied to any problem that consists of making a number of two-way (i.e. binary) decisions.

Logarithms

The complexity of any journey can be represented either as the number of possible final destinations or as the number of forks in the road which must be traversed in order to reach a given destination. We know that as the number of forks increases, so the number of possible destinations also increases. As we have already seen, if there are three forks then there are $8 = 2^3$ possible destinations.

Decimal	0	1	2	3	4	5	6	7
Binary	000	001	010	011	100	101	110	111

Table 1.1. Decimal numbers and their equivalent binary representations.

Viewed from another perspective, if there are $m = 8$ possible destinations then how many forks n does this imply? In other words, given eight destinations, what power of 2 is required in order to get 8? In this case, we know the answer is $n = 3$, which is called the *logarithm* of 8. Thus, $3 = \log_2 8$ is the number of forks implied by eight destinations.

More generally, the logarithm of m is the power to which 2 must be raised in order to obtain m; that is, $m = 2^n$. Equivalently, given a number m which we wish to express as a logarithm,

$$n = \log_2 m. \tag{1.4}$$

The subscript $_2$ indicates that we are using logs to the base 2 (all logarithms in this book use base 2 unless stated otherwise). See Appendix C for a tutorial on logarithms.

A Journey of $\log_2(8)$ Decisions

Now that we know about logarithms, we can summarise your journey from a different perspective, in terms of bits:

> If you have to choose between 2 equally probable alternatives (i.e. 2^1) then you need $1 (= \log_2 2^1 = \log_2 2)$ bit of information.

> If you have to choose between 4 equally probable alternatives (i.e. 2^2) then you need $2 (= \log_2 2^2 = \log_2 4)$ bits of information.

> If you have to choose between 8 equally probable alternatives (i.e. 2^3) then you need $3 (= \log_2 2^3 = \log_2 8)$ bits of information.

More generally, if you have to choose between m equally probable alternatives, then you need $n = \log_2 m$ bits of information.

Key point. If you have n bits of information, then you can choose from $m = 2^n$ equally probable alternatives. Equivalently, if you have to choose between m equally probable alternatives, then you need $n = \log_2 m$ bits of information.

1.4. A Million Answers to Twenty Questions

Navigating a series of forks in the road is, in some respects, similar to the game of '20 questions'. In this game, your opponent chooses a word (usually a noun), and you (the astute questioner) are allowed to ask 20 questions in order to discover the identity of this word. Crucially, each question must have a yes/no (i.e. binary) answer, and therefore provides you with a maximum of one bit of information.

By analogy with the navigation example, where each decision at a road fork halved the number of remaining destinations, each question should *halve* the number of remaining possible words. In doing so, each answer provides you with exactly one bit of information. A question to which you already know the answer is a poor choice of question. For example, if your question is, "Is the word in the dictionary?", then the answer is almost certainly, "Yes!", an answer which is predictable, and which therefore provides you with no information.

Conversely, a well-chosen question is one to which you have no idea whether the answer will be yes or no; in this case, the answer provides exactly one bit of information. The cut-down version of '20 questions' in Figure 1.3 shows this more clearly.

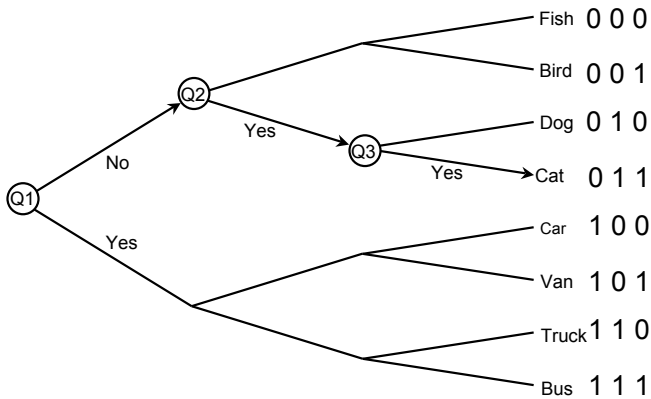


Figure 1.3. The game of '20 questions', here abbreviated to '3 questions'. Given an opponent who has one of eight words in mind, each yes/no question halves the number of remaining possible words. Each binary number on the right summarises the sequence of answers required to arrive at one word (no=0 and yes=1).

In this game, your opponent has a vocabulary of exactly eight words, and you know which words they are. Your first question (Q1) could be, "Is it inanimate?", and the answer should halve the number of possible words to four, leading you to your second question (Q2). If your second question (Q2) is, "Is it a mammal?", then the answer should again halve the number of possible words, leading to your third question (Q3). By the time you arrive at Q3, there are just two possible words left, and after you have asked the third question (e.g. "Is it 'cat'?"), your opponent's yes/no response leads you to the correct answer. In summary, you have asked three questions, and excluded all but one out of eight possible words.

More realistically, let's assume your opponent has the same vocabulary as you do (most of us have similar vocabularies, so this assumption is not entirely unreasonable). Specifically, let's assume this vocabulary contains exactly 1,048,576 words. Armed with this knowledge, each question can, in principle, be chosen to halve the number of remaining possible words. So, in an ideal world, your first question should halve the number of possible words to 524,288. Your next question should halve this to 262,144 words, and so on. By the time you get to the 19th question there should be just two words left, and after the 20th question, there should be only one word remaining.

The reason this works out so neatly is because 20 questions allow you to choose from exactly $1,048,576 = 2^{20}$ equally probable words (i.e. about one million). Thus, the 20 bits of information you have acquired with your questioning provide you with the ability to narrow down the range of possible words from about 1 million to just one. In other words, 20 questions allow you to find the correct word out of about a million possible words.

Adding one more question would not only create a new game, '21 questions', it would also double the number of possible words (to about 2 million) that you could narrow down to one. By extension, each additional question allows you to acquire up to one more bit of information, and can therefore double the initial number of words. In principle, a game of '40 questions' allows you to acquire 40 bits of information, allowing you to find one out of $2^{40} \approx 10^{12}$ words.

In terms of the navigation example, 40 bits would allow you to navigate 40 forks in the road, and would therefore permit you to choose one out of about a trillion possible routes. So the next time you arrive at your destination after a journey that involved 40 decisions, remember that you have avoided arriving at a trillion-minus-one incorrect destinations.

1.5. Information, Bits and Binary Digits

Despite the fact that the word *bit* is derived from *binary digit*, there is a subtle, but vital, difference between them. A binary digit is the value of a binary variable, where this value can be either a 0 or a 1, but a binary digit is not information *per se*. In contrast, a bit is a definite *amount of information*. Bits and binary digits are different types of entity, and to confuse one with the other is known as a *category error*.

To illustrate this point, consider the following two extreme examples. At one extreme, if you already know that you should take the left-hand road from point A in Figure 1.2 and I show you the binary digit 0 (=left), then you have been given a binary digit but you have gained no information. At the other extreme, if you have no idea about which road to choose and I show you a 0, then you have been given a binary digit and you have also gained one bit of information. Between these extremes, if someone tells you there is a 71% probability that the left-hand road represents the correct decision and I subsequently confirm this by showing you a 0, then this 0 provides you with less than one bit of information (because you already had some information about which road to choose). In fact, when you receive my 0, you gain precisely half a bit of information (see Section 5.8). Thus, even though I cannot give you a half a binary digit, I can use a binary digit to give you half a bit of information.

The distinction between binary digits and bits is often ignored, with Pierce's book[40] being a notable exception. Even some of the best textbooks use the terms 'bit' and 'binary digit' interchangeably. This does not cause problems for experienced readers as they can interpret the term 'bit' as meaning a binary digit or a bit's worth of information according to context. But for novices the failure to respect this distinction is a source of genuine confusion.

Sadly, in modern usage, the terms bit and binary digit have become synonymous, and MacKay (2003)[34] proposed that the unit of information should be called the *Shannon*.

Key point. A bit is the *amount of information* required to choose between two equally probable alternatives (e.g. left/right), whereas a binary digit is the *value of a binary variable*, which can adopt one of two possible values (i.e. 0/1).

1.6. Example 1: Telegraphy

Suppose you have just discovered that if you hold a compass next to a wire, then the compass needle changes position when you pass a current through the wire. If the wire is long enough to connect two towns like London and Manchester, then a current initiated in London can deflect a compass needle held near to the wire in Manchester.

You would like to use this new technology to send messages in the form of individual letters. Sadly, the year is 1820, so you will have to wait over 100 years for Shannon's paper to be published. Undeterred, you forge ahead. Let's say you want to send only upper-case letters, to keep matters simple. So you set up 26 electric lines, one per letter from A to Z, with the first line being A, the second line being B, and so on. Each line is set up next to a compass which is kept some distance from all the other lines, to prevent each line from deflecting more than one compass.

In London, each line is labelled with a letter, and the corresponding line is labelled with the same letter in Manchester. For example, if you want to send the letter D, you press a switch on the fourth line in London, which sends an electric current to Manchester along the wire which is next to the compass labelled with the letter D. Of course, lines fail from time to time, and it is about 200 miles from London to Manchester, so finding the location of the break in a line is difficult and expensive. Naturally, if there were fewer lines then there would be fewer failures.

With this in mind, Cooke and Wheatstone devised a complicated two-needle system, which could send only 23 different letters. Despite

the complexity of their system, it famously led to the arrest of a murderer. On the first of January 1845, John Tawell poisoned his mistress, Sarah Hart, in a place called Salt Hill in the county of Berkshire, before escaping on a train to Paddington station in London. In order to ensure Tawell's arrest when he reached his destination, the following telegraph was sent to London:

> A MURDER HAS GUST BEEN COMMITTED AT SALT HILL AND THE SUSPECTED MURDERER WAS SEEN TO TAKE A FIRST CLASS TICKET TO LONDON BY THE TRAIN WHICH LEFT SLOUGH AT 742 PM HE IS IN THE GARB OF A KWAKER ...

The unusual spellings of the words JUST and QUAKER were a result of the telegrapher doing his best in the absence of the letters J, Q and Z in the array of 23 letters before him. As a result of this telegram, Tawell was arrested and subsequently hanged for murder. The role of Cooke and Wheatstone's telegraph in Tawell's arrest was widely reported in the press, and established the practicality of telegraphy.

In the 1830s, Samuel Morse and Alfred Vail developed the first version of (what came to be known as) the *Morse code*. Because this specified each letter as dots and dashes, it could be used to send messages over a single line.

An important property of Morse code is that it uses short *codewords* for the most common letters, and longer codewords for less common letters, as shown in Table 1.2. Morse adopted a simple strategy to find out which letters were most common. Reasoning that newspaper

A ●-	J ●- - -	S ●●●
B - ●●●	K - ● -	T -
C - ●- ●	L ●- ●●	U ●●-
D - ●●	M - -	V ●●-
E ●	N - ●	W ●- -
F ●●- ●	O - - -	X - ●●-
G - - ●	P ●- - ●	Y - ●- -
H ●●●●	Q - - ●-	Z - - ●●
I ●●	R ●- ●	

Table 1.2. Morse code. Common letters (e.g. E) have the shortest codewords, whereas rare letters (e.g. Z) have the longest codewords.

printers would have only as many copies of each letter as were required, he went to a printer's workshop and counted the copies of each letter. As a result, the most common letter E is specified as a single dot, whereas the rare J is specified as a dot followed by three dashes.

The ingenious strategy adopted by Morse is important because it enables efficient use of the communication channel (a single wire). We will return to this theme many times, and it raises a fundamental question: how can we tell if a communication channel is being used as efficiently as possible?

1.7. Example 2: Binary Images

The internal structure of most images is highly predictable. For example, most of the individual *pic*ture *el*ements or *pixels* in the image of stars in Figure 1.4 are black, with an occasional white pixel, a star. Because almost all pixels are black, it follows that most pairs of adjacent pixels are also black, which makes the image's internal structure predictable. If this picture were taken by the orbiting Hubble telescope then its predictable structure would allow it to be efficiently transmitted to Earth.

Suppose you were in charge of writing the computer code which conveys the information in Figure 1.4 from the Hubble telescope to Earth. You could naively send the value of each pixel; let's call this method A. Because there are only two values in this particular image (black and white), you could choose to indicate the colour black with the binary digit 0, and the colour white with a 1. You would therefore need to send as many 0s and 1s as there are pixels in the image. For example, if the image was 100×100 pixels then you would need to send ten thousand 0s or 1s for the image to be reconstructed on Earth. Because almost all the pixels are black, you would send sequences of hundreds of 0s interrupted by the occasional 1. It is not hard to see that this is a wasteful use of the expensive satellite communication channel. How could it be made more efficient?

Another method consists of sending only the locations of the white pixels (method B). This would yield a code like $[(19, 13), (22, 30), \dots]$, where each pair of numbers represents the row and column of a white pixel.

Figure 1.4. The night sky. Each pixel contains one of just two values.

Yet another method consists of concatenating all of the rows of the image, and then sending the number of black pixels that occur before the next white pixel (method C). If the number of black pixels that precede the first white pixel is 13 and there are 9 pixels before the next white pixel, then the first row of the image begins with $0000000000000100000000001\ldots$, and the code for communicating this would be $[13, 9, \ldots]$, which is clearly more compact than the 24 binary digits which begin the first row of the image.

Notice that method A consists of sending the image itself, whereas methods B and C do not send the image, but they do send all of the *information* required to reconstruct the image on Earth. Crucially, the end results of all three methods are identical, and it is only the efficiency of the methods that differs.

In fact, whether A, B, or C is the most efficient method depends on the structure of the image. This can be seen if we take an extreme example consisting of just one white pixel in the centre of the image. For this image, method A is fairly useless, because it would require 10,000 binary values to be sent. Method B would consist of two numbers, $(50, 50)$, and method C would consist of a single number, $5,050$. If we ignore the brackets and commas then we end up with four decimal digits for both methods B and C. So these methods seem to be equivalent, at least for the example considered here.

For other images, with other structures, different *encoding methods* will be more or less efficient. For example, Figure 1.5 contains just two grey-levels, but these occur in large regions of pure black or pure

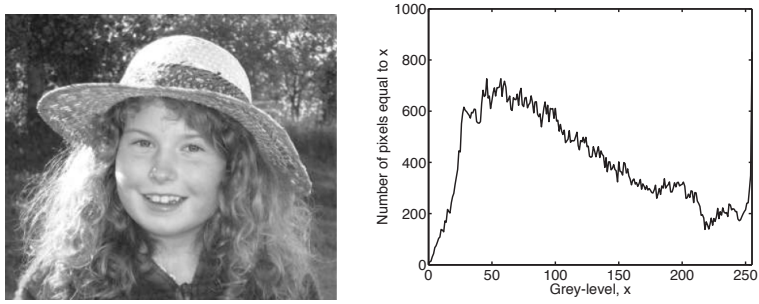
Figure 1.5. In a binary image, each pixel has 1 out of 2 possible grey-levels.

white. In this case, it seems silly to use method B to send the location of every white pixel, because so many of them occur in long runs of white pixels. This observation makes method C seem to be an obvious choice – but with a slight change. Because there are roughly equal numbers of black and white pixels which occur in regions of pure black or pure white, we could just send the number of pixels which precede the next change from black to white or from white to black. This is known as *run-length encoding*.

To illustrate this, if the distance from the first black pixel in the middle row to the first white pixel (the girl's hair) is 87 pixels, and the distance from there to the next black pixel is 31 pixels, and the distance to the next white pixel is 18 pixels, then this part of the image would be encoded as $[87, 31, 18, \dots]$. Provided we know the method used to encode an image, it is a relatively simple matter to reconstruct the original image from the encoded image.

1.8. Example 3: Grey-Level Images

Suppose we wanted to transmit an image of 100×100 pixels, in which each pixel has more than two possible grey-level values. A reasonable number of grey-levels turns out to be 256, as shown in Figure 1.6a. As before, there are large regions that look as if they contain only one grey-level. In fact, each such region contains grey-levels which are similar, but not identical, as shown in Figure 1.7. The similarity between nearby pixel values means that adjacent pixel values are not *independent* of



(a) (b)

Figure 1.6. Grey-level image. (a) An image in which each pixel has one out of 256 possible grey-levels, between 0 and 255, each of which can be represented by a binary number with 8 binary digits (e.g. 255=11111111). (b) Histogram of grey-levels in the picture.

each other, and that the image has a degree of *redundancy*. How can this observation be used to encode the image?

One method consists of encoding the image in terms of the differences between the grey-levels of adjacent pixels. For brevity, we will call this *difference coding*. (More complex methods exist, but most are similar in spirit to this simple method.) In principle, pixel differences could be measured in any direction within the image, but, for simplicity, we concatenate consecutive rows to form a single row of 10,000 pixels, and then take the difference between adjacent grey-levels. We can see the result of difference coding by 'un-concatenating' the rows to reconstitute an image, as shown in Figure 1.8a, which looks like a badly printed version of Figure 1.6a. As we shall see, both images contain the same amount of information.

If adjacent pixel grey-levels in a given row are similar, then the difference between the grey-levels is close to zero. In fact, a histogram of difference values shown in Figure 1.8b shows that the most common difference values are indeed close to zero, and only rarely greater than ± 63. Thus, using difference coding, we could represent almost every one of the 9,999 difference values in Figure 1.8a as a number between -63 and $+63$.

In those rare cases where the grey-level difference is larger than ± 63, we could list these separately as each pixel's location (row and column

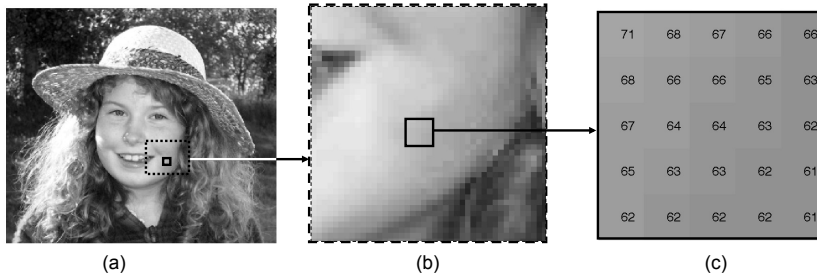


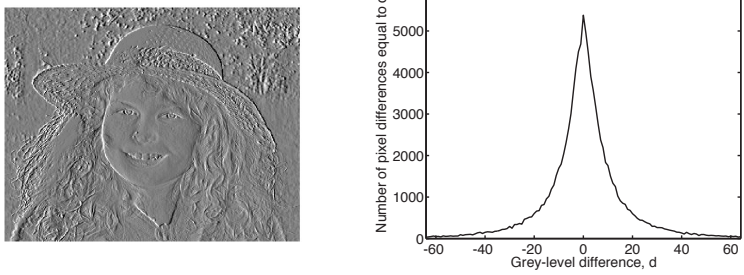
Figure 1.7. Adjacent pixels tend to have similar grey-levels, so the image has a large amount of redundancy, which can be used for efficient encoding. (a) Grey-level image. (b) Magnified square from a. (c) Magnified square from b, with individual pixel grey-levels indicated.

as 2×7 binary digits), and its grey-level (8 binary digits). Most coding procedures have special 'housekeeping' fragments of computer code to deal with things like this, but these account for a negligible percentage of the total storage space required. For simplicity, we will assume that this percentage is zero.

At first, it is not obvious how difference coding represents any saving over simply sending the value of each pixel's grey-level. However, because these differences are between -63 and $+63$, they span a range of 127 different values, i.e. $[-63, -62, \ldots, 0, \ldots, 62, 63]$. Any number in this range can be represented using seven binary digits, because $7 = \log 128$ (leaving one spare value).

In contrast, if we were to send each pixel's grey-level in Figure 1.6a individually, then we would need to send 10,000 grey-levels. Because each grey-level could be any value between 0 and 255, we would have to send eight binary digits ($8 = \log 256$) for each pixel.

Once we have encoded an image into 9,999 pixel grey-level differences $(d_1, d_2, \ldots, d_{9999})$, how do we use them to reconstruct the original image? If the difference d_1 between the first pixel grey-level x_1 and the second pixel grey-level x_2 is, say, $d_1 = (x_2 - x_1) = 10$ grey-levels and the grey-level of x_1 is 5, then we obtain the original grey-level of x_2 by adding 10 to x_1; that is, $x_2 = x_1 + d_1$ so $x_2 = 5 + 10 = 15$. We then continue this process for the third pixel ($x_3 = x_2 + d_2$), and so on. Thus, provided we know the grey-level of the first pixel in the original image (which can be encoded as eight binary digits), we can use the



(a) (b)

Figure 1.8. Difference coding. (a) Each pixel grey-level is the difference between adjacent horizontal grey-level values in Figure 1.6a (grey = zero difference). (b) Histogram of grey-level differences between adjacent pixel grey-levels in Figure 1.6a. Only differences between ±63 are plotted.

pixel grey-level differences to recover the grey-level of every pixel in the original image. The fact that we can reconstruct the original image (Figure 1.6a) from the grey-level differences (Figure 1.8a) proves that they both contain exactly the same amount of *information*.

Let's work out the total saving from using this difference coding method. The naive method of sending all pixel grey-levels, which vary between 0 and 255, would need eight binary digits per pixel, requiring a total of 80,000 binary digits. Using difference coding we would need seven binary digits per difference value, making a total of 70,000 binary digits. Therefore, using difference coding provides a saving of 10,000 binary digits, or 12.5%.

In practice, a form of difference coding is used to reduce the amount of data required to transmit voices over the telephone, where it is known as *differential pulse code modulation*. Using the differences between consecutive values, a voice signal which would otherwise require eight binary digits per value can be transmitted with just five binary digits.

As we shall see in subsequent chapters, a histogram of data values (e.g. image grey-levels) can be used to find an upper bound for the average amount of information each data value could convey. Accordingly, the histogram (Figure 1.6b) of the grey-levels in Figure 1.6a defines an upper bound of 7.84 bits/pixel. In contrast, the histogram (Figure 1.8b) of the grey-level differences in Figure 1.8a defines an upper bound of just 5.92 bits/pixel.

Given that the images in Figures 1.6a and 1.8a contain the same amount of information, and that Figure 1.8a contains no more than 5.92 bits/pixel, it follows that Figure 1.6a cannot contain more than 5.92 bits/pixel either. This matters because Shannon's work guarantees that if each pixel's grey-level contains an average of 5.92 bits of information, then we should be able to represent Figure 1.6a using no more than 5.92 binary digits per pixel. But this still represents an upper bound. In fact, the smallest number of binary digits required to represent each pixel is equal to the amount of information (measured in bits) implicit in each pixel. So what we really want to know is: how much information does each pixel contain?

This is a hard question, but we can get an idea of the answer by comparing the amount of computer memory required to represent the image in two different contexts (for simplicity, we assume that each pixel has eight binary digits). First, in order to display the image on a computer screen, the value of each pixel occupies eight binary digits, so the bigger the picture, the more memory it requires to be displayed. Second, a compressed version of the image can be stored on the computer's hard drive using an average of less than eight binary digits per pixel (e.g. by using the difference coding method above). Consequently, storing the (compressed) version of an image on the hard drive requires less memory than displaying that image on the screen. In practice, image files are usually stored in compressed form with the method used to compress the image indicated by the file name extension (e.g. '.jpeg').

The image in Figure 1.6a is actually 344 by 299 pixels, where each pixel grey-level is between 0 and 255, which can be represented as eight binary digits (because $2^8 = 256$), or one *byte*. This amounts to a total of 102,856 pixels, each of which is represented on a computer screen as one byte. However, when the file containing this image is inspected, it is found to contain only 45,180 bytes; the image in Figure 1.6a can be compressed by a factor of $2.28 (= 102856/45180)$ without any loss of information. This means that the information implicit in each pixel, which requires eight binary digits for it to be displayed on a screen,

can be represented with about four binary digits on a computer's hard drive.

Thus, even though each pixel can adopt any one of 256 possible grey-levels, and is displayed using eight binary digits of computer memory, the grey-level of each pixel can be stored in about four binary digits. This is important, because it implies that each set of eight binary digits used to display each pixel in Figure 1.6a contains an average of only four bits of information, and therefore each binary digit contains only *half a bit* of information. At first sight, this seems like an odd result. But we already know from Section 1.5 that a binary digit can represent half a bit, and we shall see later (especially in Chapter 5) that a fraction of a bit is a well-defined quantity which has a reasonably intuitive interpretation.

1.9. Summary

From navigating a series of forks in the road, and playing the game of '20 questions', we have seen how making binary choices requires information in the form of simple yes/no answers. These choices can also be used to choose from a set of letters, and can therefore be used to send typed messages along telegraph wires.

We found that increasing the number of choices from two (forks in the road) to 26 (letters) to 256 (pixel grey-levels) allowed us to transmit whole images down a single wire as a sequence of binary digits. In each case, the redundancy of the data in a message allowed it to be compressed before being transmitted. This redundancy emphasises a key point: a binary digit does not necessarily provide one bit of information. More importantly, a binary digit is *not* the same type of entity as a bit of information.

So, what is information? It is what remains after every iota of natural redundancy has been squeezed out of a message, and after every aimless syllable of noise has been removed. It is the unfettered essence that passes from computer to computer, from satellite to Earth, from eye to brain, and (over many generations of natural selection) from the natural world to the collective gene pool of every species.

Chapter 2

Entropy of Discrete Variables

> Information is the resolution of uncertainty.
> Shannon C, 1948.

2.1. Introduction

Now that we have an idea of the key concepts of information theory, we can begin to explore its inner workings on a more formal basis. But first, we need to establish a few ground rules regarding *probability*, *discrete variables* and *random variables*. Only then can we make sense of *entropy*, which lies at the core of information theory.

2.2. Ground Rules and Terminology

Probability

We will assume a fairly informal notion of probability based on the number of times particular events occur. For example, if a bag contains 40 white balls and 60 black balls then we will assume that the probability of reaching into the bag and choosing a black ball is the same as the proportion, or *relative frequency*, of black balls in the bag (i.e. $60/100 = 0.6$). From this, it follows that the probability of an event (e.g. choosing a black ball) can adopt any value between zero and one, with zero meaning it definitely will not occur, and one meaning it definitely will occur. Finally, given a set of mutually exclusive events (such as choosing a ball, which has to be either black or white), the probabilities of those events must add up to one (e.g. $0.4 + 0.6 = 1$). See Appendix F for an overview of the rules of probability.

Bibliography

[1] Applebaum, D. (2008). *Probability and Information: An Integrated Approach*. 2nd edition. Cambridge University Press.

[2] Avery, J. (2012). *Information Theory and Evolution*. World Scientific Publishing, New Jersey.

[3] Baeyer, H. (2005). *Information: The New Language of Science*. Harvard University Press, Cambridge, MA.

[4] Baldwin, J. (1896). A new factor in evolution. *The American Naturalist*, 30(354):441–451.

[5] Barlow, H. (1961). Possible principles underlying the transformation of sensory messages. In W.A. Rosenblith, editor, *Sensory Communication*, pp. 217–234. MIT Press, Cambridge, MA.

[6] Bennett, C. (1987). Demons, engines and the second law. *Scientific American*, 257(5):108–116.

[7] Bérut, A., Arakelyan, A., Petrosyan, A., Ciliberto, S., Dillenschneider, R., and Lutz, E. (2012). Experimental verification of Landauer's principle linking information and thermodynamics. *Nature*, 483(7388):187–189.

[8] Bialek, W. (2012). *Biophysics: Searching for Principles*. Princeton University Press, New Jersey.

[9] Bishop, C. (2006). *Pattern Recognition and Machine Learning*. Springer.

[10] Brown, P.F., Pietra, V.J.D., Mercer, R.L., Pietra, S.A.D. and Lai, J.C. (1992). An estimate of an upper bound for the entropy of English. *Computational Linguistics*, 1(18):31–40.

[11] Cover, T. and Thomas, J. (1991). *Elements of Information Theory*. John Wiley and Sons, New York.

[12] Darwin, C. (1859). *On the origin of species by means of natural selection, or the preservation of favoured races in the struggle for life*. 1st edition. John Murray, London.

[13] DeGroot, M. (1986). *Probability and Statistics*. 2nd edition. Addison-Wesley, New York.

[14] Deutsch, D. and Marletto, C. (2014). Reconstructing physics: The universe is information. *New Scientist*, 2970:30.

[15] Feynman, R., Leighton, R., and Sands, M. (1964). *Feynman Lectures on Physics*. Basic Books, New York.

[16] Frisby, JP and Stone, JV. (2010). *Seeing: The computational approach to biological vision*. MIT Press, Cambridge, MA.

[17] Friston, K. (2010). The free-energy principle: a unified brain theory? *Nature Review Neuroscience*, 11(2):127–138.

[18] Gabor, D. (1951). Lectures on communication theory. Technical report, Massachusetts Institute of Technology.

[19] Gatenby, R. and Frieden, B. (2013). The critical roles of information and nonequilibrium thermodynamics in evolution of living systems. *Bulletin of Mathematical Biology*, 75(4):589–601.

[20] Gibbs, JW. (1902). *Elementary Principles in Statistical Mechanics.* Charles Scribner's Sons, New York.

[21] Gleick, J. (2012). *The Information*. Vintage, London.

[22] Greene, B. (2004). *The Fabric of the Cosmos.* Knopf, New York.

[23] Grover, L. (1996). A fast quantum mechanical algorithm for database search. *Proceedings, 28th Annual ACM Symposium on the Theory of Computing*, pp. 212–219.

[24] Guizzo, E. (2003). The essential message: Claude Shannon and the making of information theory. MSc Thesis, Massachusetts Institute of Technology.
`http://dspace.mit.edu/bitstream/handle/1721.1/39429/54526133.pdf`

[25] Holland, J. (1992). *Adaptation in Natural and Artificial Systems*. MIT Press, Cambridge, MA.

[26] Jaynes, E. and Bretthorst, G. (2003). *Probability Theory: The Logic of Science*. Cambridge University Press, Cambridge, England.

[27] Jessop, A. (1995). *Informed Assessments: An Introduction to Information, Entropy and Statistics*. Ellis Horwood, London.

[28] Kolmogorov, A. (1933). *Foundations of the Theory of Probability*. English translation, 1956. Chelsea Publishing Company.

[29] Kostal, L., Lansky, P., and Rospars, J.-P. (2008). Efficient olfactory coding in the pheromone receptor neuron of a moth. *PLoS Computational Biology*, 4(4).

[30] Landauer, R. (1961). Irreversibility and heat generation in the computing process. *IBM J. Research and Development*, 5:183–191.

[31] Laughlin, S. (1981). A simple coding procedure enhances a neuron's information capacity. *Z Naturforsch C*, 36(9–10):910–912.

[32] Lemon, D. (2013). *A Student's Guide to Entropy*. Cambridge University Press, Cambridge, England.

[33] MacKay, A. (1967). Optimization of the genetic code. *Nature*, 216:159–160.

[34] MacKay, D. (2003). *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, Cambridge, England.

[35] Nemenman, I., Lewen, G., Bialek, W., and de Ruyter van Steveninck, R. (2008). Neural coding of natural stimuli: Information at sub-millisecond resolution. *PLoS Computational Biology*, 4(3).

[36] Nemenman, I., Shafee, F., and Bialek, W. (2002). Entropy and inference, revisited. In T.G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, pp. 471–478. MIT Press, Cambridge, MA.

[37] Nirenberg, S and Carcieri, SM and Jacobs, AL and Latham, PE. (2001). Retinal ganglion cells act largely as independent encoders. *Nature*, 411(6838):698–701.

[38] Olshausen, B. and Field, D. (1996). Natural image statistics and efficient coding. *Network: Computation in Neural Systems*, 7:333–339.

[39] Paninski, L. (2003). Estimation of entropy and mutual information. *Neural Computation*, 15(6):1191–1253.

[40] Pierce, J. (1961). *An Introduction to Information Theory: Symbols, Signals and Noise*. 2nd edition, Dover, 1980.

[41] Reza, F. (1961). *Information Theory*. McGraw-Hill, New York.

[42] Rieke, F., Bodnar, D., and Bialek, W. (1995). Naturalistic stimuli increase the rate and efficiency of information transmission by primary auditory afferents. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 262(1365):259–265.

[43] Rieke, F., Warland, D., van Steveninck, R., and Bialek, W. (1997). *Spikes: Exploring the Neural Code*. MIT Press, Cambridge, MA.

[44] Schneider, T. D. (2010). 70% efficiency of bistate molecular machines explained by information theory, high dimensional geometry and evolutionary convergence. *Nucleic acids research*, 38(18):5995–6006.

[45] Schürmann, T. and Grassberger, P. (1996). Entropy estimation of symbol sequences. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 6(3):414–427.

[46] Seife, C. (2007). *Decoding the Universe: How the New Science of Information Is Explaining Everything in the Cosmos, From Our Brains to Black Holes*. Penguin.

[47] Sengupta, B., Stemmler, M., and Friston, K. (2013). Information and efficiency in the nervous system: a synthesis. *PLoS Computational Biology*, 9(7).

[48] Shannon, C. (1948). A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423.

[49] Shannon, C. (1951). Prediction and entropy of printed English. *Bell System Technical Journal*, 30:47–51.

[50] Shannon, C. and Weaver, W. (1949). *The Mathematical Theory of Communication*. University of Illinois Press.

[51] Stone, J. (2012). *Vision and Brain: How we perceive the world*. MIT Press, Cambridge, MA.

[52] Stone, J. (2013). *Bayes' Rule: A Tutorial Introduction to Bayesian Analysis*. Sebtel Press, Sheffield, England.

[53] Wallis, K. (2006). A note on the calculation of entropy from histograms. Technical report, University of Warwick.

[54] Watkins, C. (2008). Selective breeding analysed as a communication channel: channel capacity as a fundamental limit on adaptive complexity. In *Symbolic and Numeric Algorithms for Scientific Computing: Proceedings of SYNASC'08*, pp. 514–518.

[55] Zhaoping, L. (2014). *Understanding Vision: Theory, Models, and Data*. Oxford University Press.

Index

About the Author.
Dr James Stone is a Reader in Computational Neuroscience at the University of Sheffield, England. Previous books are listed below.

Bayes' Rule: A Tutorial Introduction to Bayesian Analysis,
JV Stone, Sebtel Press, 2013.
Vision and Brain: How We Perceive the World,
JV Stone, MIT Press, 2012.
Seeing: The Computational Approach to Biological Vision,
JP Frisby and JV Stone, MIT Press, 2010.
Independent Component Analysis: A Tutorial Introduction,
JV Stone, MIT Press, 2004.