MATHEMATICS OF DIGITAL IMAGES

Creation, Compression, Restoration, Recognition

Compression, restoration and recognition are three of the key components of digital imaging. The mathematics needed to understand and carry out all these components is here explained in a textbook that is at once rigorous and practical with many worked examples, exercises with solutions, pseudocode, and sample calculations on images. The introduction lists fast tracks to special topics such as Principal Component Analysis, and ways into and through the book, which abounds with illustrations. The first part describes plane geometry and pattern-generating symmetries, along with some text on 3D rotation and reflection matrices. Subsequent chapters cover vectors, matrices and probability. These are applied to simulation, Bayesian methods, Shannon's Information Theory, compression, filtering and tomography. The book will be suited for course use or for self-study. It will appeal to all those working in biomedical imaging and diagnosis, computer graphics, machine vision, remote sensing, image processing, and information theory and its applications.

DR S. G. HOGGAR is a research fellow and formerly a senior lecturer in mathematics at the University of Glasgow.

MATHEMATICS OF DIGITAL IMAGES

Creation, Compression, Restoration, Recognition

S. G. HOGGAR
University of Glasgow

CAMBRIDGE
UNIVERSITY PRESS

CAMBRIDGE UNIVERSITY PRESS
Cambridge, New York, Melbourne, Madrid, Cape Town, Singapore, São Paulo

Cambridge University Press
The Edinburgh Building, Cambridge CB2 2RU, UK

www.cambridge.org
Information on this title: www.cambridge.org/9780521780292

First published 2006

Printed in the United Kingdom at the University Press, Cambridge

A catalogue record for this book is available from the British Library

ISBN-13 978-0-521-78029-2 hardback
ISBN-10 0-521-78029-2 hardback

To my wife, Elisabeth

Contents

Preface *page* xi
Introduction xiii
A word on notation xxvii
List of symbols xxix
Part I The plane 1
1 Isometries 3
1.1 Introduction 3
1.2 Isometries and their sense 6
1.3 The classification of isometries 16
Exercises 1 21
2 How isometries combine 23
2.1 Reflections are the key 24
2.2 Some useful compositions 25
2.3 The image of a line of symmetry 31
2.4 The dihedral group 36
2.5 Appendix on groups 40
Exercises 2 41
3 The seven braid patterns 43
Constructing braid patterns 45
Exercises 3 46
4 Plane patterns and symmetries 48
4.1 Translations and nets 48
4.2 Cells 50
4.3 The five net types 56
Exercises 4 63
5 The 17 plane patterns 64
5.1 Preliminaries 64
5.2 The general parallelogram net 66
5.3 The rectangular net 67
5.4 The centred rectangular net 68

5.5 The square net 69
5.6 The hexagonal net 71
5.7 Examples of the 17 plane pattern types 73
5.8 Scheme for identifying pattern types 75
Exercises 5 77
6 More plane truth 79
6.1 Equivalent symmetry groups 79
6.2 Plane patterns classified 82
6.3 Tilings and Coxeter graphs 91
6.4 Creating plane patterns 99
Exercises 6 109
Part II Matrix structures 113
7 Vectors and matrices 115
7.1 Vectors and handedness 115
7.2 Matrices and determinants 126
7.3 Further products of vectors in 3-space 140
7.4 The matrix of a transformation 145
7.5 Permutations and the proof of Determinant Rules 155
Exercises 7 159
8 Matrix algebra 162
8.1 Introduction to eigenvalues 162
8.2 Rank, and some ramifications 172
8.3 Similarity to a diagonal matrix 180
8.4 The Singular Value Decomposition (SVD) 192
Exercises 8 203
Part III Here's to probability 207
9 Probability 209
9.1 Sample spaces 209
9.2 Bayes' Theorem 217
9.3 Random variables 227
9.4 A census of distributions 239
9.5 Mean inequalities 251
Exercises 9 256
10 Random vectors 258
10.1 Random vectors 258
10.2 Functions of a random vector 265
10.3 The ubiquity of normal/Gaussian variables 277
10.4 Correlation and its elimination 285
Exercises 10 302
11 Sampling and inference 303
11.1 Statistical inference 304
11.2 The Bayesian approach 324

11.3 Simulation 335
11.4 Markov Chain Monte Carlo 344
Exercises 11 389
Part IV Information, error and belief 393
12 Entropy and coding 395
12.1 The idea of entropy 395
12.2 Codes and binary trees 402
12.3 Huffman text compression 406
12.4 The redundancy of Huffman codes 412
12.5 Arithmetic codes 417
12.6 Prediction by Partial Matching 424
12.7 LZW compression 425
12.8 Entropy and Minimum Description Length (MDL) 429
Exercises 12 442
13 Information and error correction 444
13.1 Channel capacity 444
13.2 Error-correcting codes 469
13.3 Probabilistic decoding 493
13.4 Postscript: Bayesian nets in computer vision 516
Exercises 13 518
Part V Transforming the image 521
14 The Fourier Transform 523
14.1 The Discrete Fourier Transform 523
14.2 The Continuous Fourier Transform 540
14.3 DFT connections 546
Exercises 14 558
15 Transforming images 560
15.1 The Fourier Transform in two dimensions 561
15.2 Filters 578
15.3 Deconvolution and image restoration 595
15.4 Compression 617
15.5 Appendix 628
Exercises 15 635
16 Scaling 637
16.1 Nature, fractals and compression 637
16.2 Wavelets 658
16.3 The Discrete Wavelet Transform 665
16.4 Wavelet relatives 677
Exercises 16 683

Part VI See, edit, reconstruct 685
17 B-spline wavelets 687
17.1 Splines from boxes 687
17.2 The step to subdivision 703
17.3 The wavelet formulation 719
17.4 Appendix: band matrices for finding Q, A and B 732
17.5 Appendix: surface wavelets 743
Exercises 17 754
18 Further methods 757
18.1 Neural networks 757
18.2 Self-organising nets 783
18.3 Information Theory revisited 792
18.4 Tomography 818
Exercises 18 830
References 832
Index 845

Preface

This text is a successor to the 1992 *Mathematics for Computer Graphics*. It retains the original Part I on plane geometry and pattern-generating symmetries, along with much on 3D rotation and reflection matrices. On the other hand, the completely new pages exceed in number the total pages of the older book.

In more detail, topology becomes a reference and is replaced by probability, leading to simulation, priors and Bayesian methods, and the Shannon Information Theory. Also, notably, the Fourier Transform appears in various incarnations, along with Artificial Neural Networks. As the book's title implies, all this is applied to digital images, their processing, compresssion, restoration and recognition.

Wavelets are used too, in compression (as are fractals), and in conjuction with B-splines and subdivision to achieve multiresolution and curve editing at varying scales. We conclude with the Fourier approach to tomography, the medically important reconstruction of an image from lower-dimensional projections.

As before, a high priority is given to examples and illustrations, and there are exercises, which the reader can use if desired, at strategic points in the text; these sometimes form part of the exercises placed at the end of each chapter. Exercises marked with a tick are partly, or more likely fully, solved on the website. Especially after Chapter 6, solutions are the rule, except for implementation exercises. In the latter regard there are a considerable number of pseudocode versions throughout the text, for example ALGO 11.9 of Chapter 11, simulating the d-dimensional Gaussian distribution, or ALGO 16.1, wavelet compression with limited percentage error.

A further priority is to help the reader know, as the story unfolds, where to turn back for justification of present assumptions, and to point judiciously forward for coming applications. For example, the mentioned Gaussian of Chapter 11 needs the theory of positive definite matrices in Chapter 8. In the introduction we suggest some easy ways in, including journeys by picture alone, or by light reading.

Much of the material of this book began as a graduate course in the summer of 1988, for Ph.D. students in computer graphics at the Ohio State University. My thanks are due to Rick Parent for encouraging the idea of such a course. A further part of the book was developed from a course for final year mathematics students at the University of Glasgow.

I thank my department for three months' leave at the Cambridge Newton Institute, and Chris Bishop for organising the special period on Neural Nets, at which I learned so much and imbibed the Bayesian philosophy.

I am indebted to Paul Cockshott for kindly agreeing to be chief checker, and provoking many corrections and clarifications. My thanks too to Jean-Christoph Nebel, Elisabeth Guest and Joy Goodman, for valuable comments on various chapters. For inducting me into Computer Vision I remain grateful to Paul Siebert and the Computer Vision & Graphics Lab. of Glasgow University. Many people at Vision conferences have added to my knowledge and the determination to produce this book. For other valuable discussions at Glasgow I thank Adrian Bowman, Nick Bailey, Rob Irvine, Jim Kay, John Patterson and Mike Titterington.

Mathematica 4 was used for implementations and calculations, supplemented by the downloadable *Image* from the US National Institutes of Health. Additional images were kindly supplied by Lu, Healy & Weaver (Figures 16.35 and 16.36), by Martin Bertram (Figure 17.52), by David Salesin *et al.* (Figures 17.42 and 17.50), by Hughes Hoppe *et al.* (Figures 17.44 and 17.51), and by 'Meow' Porncharoensin (Figure 10.18). I thank the following relatives for allowing me to apply algorithms to their faces: Aukje, Elleke, Tom, Sebastiaan, Joanna and Tante Tini.

On the production side I thank Frances Nex for awesome text editing, and Carol Miller and Wendy Phillips for expertly seeing the book through to publication.

Finally, thanks are due to David Tranah, Science Editor at Cambridge University Press, for his unfailing patience, tact and encouragement till this book was finished.

Introduction

Beauty is in the eye of the beholder ...

Why the quote? Here beauty is a decoded message, a character recognised, a discovered medical condition, a sought-for face. It depends on the desire of the beholder. Given a computer image, beauty is to learn from it or convert it, perhaps to a more accurate original. But we consider creation too.

It is expected that, rather than work through the whole book, readers may wish to browse or to look up particular topics. To this end we give a fairly extended introduction, list of symbols and index. The book is in six interconnected parts (the connections are outlined at the end of the Introduction):

I	*The plane*	Chapters 1–6;
II	*Matrix structures*	Chapters 7–8;
III	*Here's to probability*	Chapters 9–11;
IV	*Information, error and belief*	Chapters 12–13;
V	*Transforming the image*	Chapters 14–16;
VI	*See, edit, reconstruct*	Chapters 17–18.

Easy ways in One aid to taking in information is first to go through following a substructure and let the rest take care of itself (a surprising amount of the rest gets tacked on). To facilitate this, each description of a part is followed by a quick trip through that part, which the reader may care to follow. If it is true that one picture is worth a thousand words then an easy but fruitful way into this book is to browse through selected pictures, and overleaf is a table of possibilities. One might take every second or third entry, for example.

Chapters 1–6 (Part I) The mathematics is geared towards producing patterns automatically by computer, allocating some design decisions to a user. We begin with *isometries* – those transformations of the plane which preserve distance and hence shape, but which may switch left handed objects into right handed ones (such isometries are called *indirect*). In this part of the book we work geometrically, without recourse to matrices. In Chapter 1 we show that isometries fall into two classes: the direct ones are rotations

Context	Figure (etc.)	Context	Figure (etc.)
Symmetry operations	2.16, 2.19	Daubechies wavelets	16.30
Net types	Example 4.20	Fingerprints	16.33
The global scheme	page 75	Wavelets & X-rays	16.35
Face under PCA & K–L	10.15	B-spline designs	17.15, 17.29
Adding noise	11.21	B-spline filter bank	17.32
MAP reconstruction	11.48	Wavelet-edited face	17.39
Martial cats and LZW	12.20	Progressive Venus face	17.52
The DFT	15.2, 15.4	Perceptron	18.15
Edge-detection	15.6, 15.23	Backpropagation	18.20
Removing blur	15.24, 15.27	Kohonen training	18.42
Progressive transmission	15.40	Vector quantisers	18.54
The DCT	15.42, 15.43	Tomography	18.66
Fractals	16.1, 16.17		

or translation, and the indirect ones reflections or glides. In Chapter 2 we derive the rules for combining isometries, and introduce groups, and the dihedral group in particular. In a short Chapter 3 we apply the theory so far to classifying all 1-dimensional or 'braid' patterns into seven types (Table 3.1).

From Chapter 4 especially we consider symmetries or 'symmetry operations' on a plane pattern. That is, those isometries which send a pattern onto itself, each part going to another with the same size and shape (see Figure 1.3 ff). A plane pattern is one having translation symmetries in two non-parallel directions. Thus examples are wallpaper patterns, floor tilings, carpets, patterned textiles, and the Escher interlocking patterns such as Figure 1.2. We prove the crystallographic restriction, that rotational symmetries of a plane pattern must be multiples of a 1/2, 1/3, 1/4 or 1/6 turn (1/5 is not allowed). We show that plane patterns are made up of parallelogram shaped cells, falling into five types (Figure 4.14).

In Chapter 5 we deduce the existence of 17 pattern types, each with its own set of interacting symmetry operations. In Section 5.8 we include a flow chart for deciding into which type any given pattern fits, plus a fund of test examples. In Chapter 6 we draw some threads together by proving that the 17 proposed categories really are distinct according to a rigorous definition of 'equivalent' patterns (Section 6.1), and that every pattern must fall into one of the categories provided it is 'discrete' (there is a *lower* limit on how far any of its symmetries can move the pattern).

By this stage we use increasingly the idea that, because the composition of two symmetries is a third, the set of all symmetries of a pattern form a group (the definition is recalled in Section 2.5). In Section 6.3 we consider various kinds of regularity upon which a pattern may be based, via techniques of Coxeter graphs and Wythoff's construction (they apply in higher dimensions to give polyhedra). Finally, in Section 6.4 we concentrate the theory towards building an algorithm to construct (e.g. by computer) a pattern of any type from a modest user input, based on a smallest replicating unit called a fundamental region.

Chapters 1–6: a quick trip Read the introduction to Chapter 1 then note Theorem 1.18 on what isometries of the plane turn out to be. Note from Theorem 2.1 how they can all be expressed in terms of reflections, and the application of this in Example 2.6 to composing rotations about distinct points. Look through Table 2.2 for anything that surprises you. Theorem 2.12 is vital information and this will become apparent later. Do the exercise before Figure 2.19. Omit Chapter 3 for now.

Read the first four pages of Chapter 4, then pause for the crystallographic restriction (Theorem 4.15). Proceed to Figure 4.14, genesis of the five net types, note Examples 4.20, and try Exercise 4.6 at the end of the chapter yourself. Get the main message of Chapter 5 by using the scheme of Section 5.8 to identify pattern types in Exercises 5 at the end of the chapter (examples with answers are given in Section 5.7). Finish in Chapter 6 by looking through Section 6.4 on 'Creating plane patterns' and recreate the one in Exercise 6.13 (end of the chapter) by finding one fundamental region.

Chapters 7–8 (Part II) After reviewing vectors and geometry in 3-space we introduce n-space and its vector subspaces, with the idea of independence and bases. Now come matrices, representing linear equations and transformations such as rotation. Matrix partition into *blocks* is a powerful tool for calculation in later chapters (8, 10, 15–17). Determinants test row/equation independence and enable n-dimensional integration for probability (Chapter 10).

In Chapter 8 we review complex numbers and eigenvalues/vectors, hence classify distance-preserving transformations (*isometries*) of 3-space, and show how to determine from the matrix of a rotation its axis and angle (Theorem 8.10), and to obtain a normal vector from a reflection matrix (Theorem 8.12). We note that the matrix M of an isometry in any dimension is *orthogonal*, that is $MM^{\mathrm{T}} = I$, or equivalently the rows (or columns) are mutually orthogonal unit vectors. We investigate the *rank* of a matrix – its number of independent rows, or of independent equations represented. Also, importantly, the technique of *elementary row operations*, whereby a matrix is reduced to a special form, or yields its inverse if one exists.

Next comes the theory of quadratic forms $\sum a_{ij}x_i x_j$ defined by a matrix $A = [a_{ij}]$, tying in with eigenvalues and undergirding the later multivariate normal/Gaussian distribution. Properties we derive for matrix norms lead to the *Singular Value Decomposition*: a general $m \times n$ matrix is reducible by orthogonal matrices to a general diagonal form, yielding approximation properties (Theorem 8.53). We include the Moore–Penrose *pseudoinverse* A^+ such that $AX = b$ has best solution $X = A^+b$ if A^{-1} does not exist.

Chapters 7–8: a quick trip Go to Definition 7.1 for the meaning of orthonormal vectors and see how they define an orthogonal matrix in Section 7.2.4. Follow the determinant evaluation in Examples 7.29 then 'Russian' block matrix multiplication in Examples 7.38. For vectors in coordinate geometry, see Example 7.51.

In Section 7.4.1 check that the matrices of rotation and reflection are orthogonal. Following this theme, see how to get the geometry from the matrix in 3D, Example 8.14.

Next see how the matrix row operations introduced in Theorem 8.17 are used for solving equations (Example 8.22) and for inverting a matrix (Example 8.27).

Now look at quadratic forms, their meaning in (8.14), the positive definite case in Table 8.1, and applying the minor test in Example 8.38. Finally, look up the pseudoinverse of Remarks 8.57 for least deviant solutions, and use it for Exercise 24 (end of chapter).

Chapters 9–11 (Part III) We review the basics of probability, defining an *event* E to be a subset of the sample space S of outcomes, and using axioms due to Kolmogorov for probability $P(E)$. After conditional probability, independence and Bayes' Theorem we introduce random variables $X\colon S \to R_X$, meaning that X allocates to each outcome s some value x in its range R_X (e.g. score x in archery depends on hit position s). An event B is now a subset of the range and X has a pdf (probability distribution function), say $f(x)$, so that the probability of B is given by the integral

$$P(B) = \int_B f(x)\,\mathrm{d}x,$$

or a sum if the range consists of discrete values rather than interval(s). From the idea of average, we define the *expected value* $\mu = E(X) = \int x f(x)\,\mathrm{d}x$ and *variance* $V(X) = E(X-\mu)^2$. We derive properties and applications of distributions entitled binomial, Poisson and others, especially the ubiquitous normal/Gaussian (see Tables 9.9 and 9.10 of Section 9.4.4).

In Chapter 10 we move to random vectors $\boldsymbol{X} = (X_1, \ldots, X_n)$, having in mind message symbols of Part IV, and pixel values. A joint pdf $f(x_1, \ldots, x_n)$ gives probability as an n-dimensional integral, for example

$$P(X < Y) = \int_B f(x, y)\,\mathrm{d}x\,\mathrm{d}y, \quad \text{where } B = \{(x, y)\colon x < y\}.$$

We investigate the pdf of a function of a random vector. In particular $X + Y$, whose pdf is the *convolution product* f^*g of the pdfs f of X and g of Y, given by

$$(f^*g)(z) = \int_{\mathbf{R}} f(t)g(z-t)\,\mathrm{d}t.$$

This gives for example the pdf of a sum of squares of Gaussians via convolution properties of the gamma distribution. Now we use *moments* $E(X_i^r)$ to generate new pdfs from old, to relate known ones, and to prove the *Central Limit Theorem* that $X_1 + \cdots + X_n$ (whatever the pdfs of individual X_i) approaches a Gaussian as n increases, a pointer to the important ubiquity of this distribution.

We proceed to the *correlation* $\mathrm{Cov}(X, Y)$ between random variables X, Y, then the covariance matrix $\mathrm{Cov}(\boldsymbol{X}) = [\mathrm{Cov}(X_i, X_j)]$ of a random *vector* $\boldsymbol{X} = (X_i)$, which yields a pdf for $\boldsymbol{X}$ if $\boldsymbol{X}$ is multivariate normal, i.e. if the X_i are normal but not necessarily independent (Theorem 10.61). Chapter 10 concludes with *Principal Component Analysis*, or PCA, in which we reduce the dimension of a data set, by transforming

to new uncorrelated coordinates ordered by decreasing variance, and dropping as many of the last few variables as have total variance negligible. We exemplify by compressing face image data.

Given a sample, i.e. a sequence of measurements $X_1, \ldots, X_n$ of a random variable X, we seek a statistic $f(X_1, \ldots, X_n)$ to test the hypothesis that X has a certain distribution or, assuming it has, to estimate any parameters (Section 11.1). Next comes a short introduction to the Bayesian approach to squeezing useful information from data by means of an initially vague prior belief, firmed up with successive observations. An important special case is *classification*: is it a tumour, a tank, a certain character, . . . ?

For testing purposes we need *simulation*, producing a sequence of variates whose frequencies mimic a given distribution (Section 11.3). We see how essentially any distribution may be achieved starting from the usual computer-generated uniform distribution on an interval [0, 1]. Example: as suggested by the Central Limit Theorem, the sum of uniform variables $U_1, \ldots, U_{12}$ on [0, 1] is normal to a good approximation.

We introduce Monte Carlo methods, in which a sequence of variates from a suitably chosen distribution yields an approximate n-dimensional integral (typically probability). The method is improved by generating the variates as a *Markov chain* $X_1, X_2, \ldots$, where X_i depends on the preceding variable but on none earlier. This is called Markov Chain Monte Carlo, or MCMC. It involves finding joint pdfs from a list of conditional ones, for which a powerful tool is a *Bayesian graph,* or *net*.

We proceed to Markov Random Fields, a generalisation of a Markov chain useful for conditioning colour values at a pixel only on values at nearest neighbours. *Simulated annealing* fits here, in which we change a parameter ('heat') following a schedule designed to avoid local minima of an 'energy function' we must minimise. Based on this, we perform Bayesian Image Restoration (Example 11.105).

Chapters 9–11: a quick trip Note the idea of *sample space* by reading Chapter 9 up to Example 9.2(i), then *random variable* in Definition 9.32 and Example 9.35. Take in the binomial case in Section 9.4.1 up to Example 9.63(ii). Now look up the *cdf* at (9.29) and Figure 9.11.

Review *expected value* at Definition 9.50 and the prudent gambler, then *variance* at Section 9.3.6 up to (9.39) and the gambler's return. Now it's time for normal/Gaussian random variables. Read Section 9.4.3 up to Figure 9.20, then follow half each of Examples 9.75 and 9.76. Glance at Example 9.77.

Check out the idea of a joint pdf $f(x, y)$ in Figure 10.1, Equation (10.4) and Example 10.2. Then read up the pdf of $X + Y$ as a convolution product in Section 10.2.2 up to Example 10.18. For the widespread appearance of the normal distribution see the introduction to Section 10.3.3, then the Central Limit Theorem 10.45, exemplified in Figure 10.7. See how the covariance matrix, (10.44), (10.47), gives the n-dimensional normal distribution in Theorem 10.61.

Read the introduction to Chapter 11, then Example 11.6, for a quick view of the hypothesis testing idea. Now the Bayesian approach, Section 11.2.1. Note the meaning of 'prior' and how it's made more accurate by increasing data, in Figure 11.11.

The Central Limit Theorem gives a quick way to simulate the Gaussian/normal: read from Figure 11.21 to 11.22. Then, note how the Choleski matrix decomposition from Chapter 8 enables an easy simulation of the n-dimensional Gaussian.

On to Markov chains, the beginning of Section 11.4 up to Definition 11.52, and their generalisation to Markov random fields, modelling an image, Examples 11.79 and preceding text. Take in Bayesian Image Restoration, Section 11.4.6 above Table 11.13, then straight on to Figure 11.48 at the end.

Chapters 12–13 (Part IV) We present Shannon's solution to the problem of measuring information. In more detail, how can we usefully quantify the information in a message understood as a sequence of symbols X (random variable) from an alphabet $\mathcal{A} = \{s_1, \ldots, s_n\}$, having a pdf $\{p_1, \ldots, p_n\}$. Shannon argued that the mean information per symbol of a message should be defined as the *entropy*

$$H(X) = H(p_1, \ldots, p_n) = \sum -p_i \log p_i$$

for some fixed basis of logarithms, usually taken as 2 so that entropy is measured in bits per symbol. An early vindication is that, if each s_i is encoded as a binary word c_i, the mean bits per symbol in any message cannot be less than H (Theorem 12.8). Is there an encoding scheme that realises H? Using a graphical method Huffman produced the most economical coding that was *prefix-free* (no codeword a continuation of another). This comes close to H, but perhaps the nearest to a perfect solution is an *arithmetic code*, in which the bits per symbol tend to H as message length increases (Theorem 12.35). The idea here extends the method of converting a string of symbols from $\{0, 1, \ldots, 9\}$ to a number between 0 and 1.

In the widely used LZW scheme by Lempel, Ziv and Welch, subsequences of the text are replaced by pointers to them in a dictionary. An ingenious method recreates the dictionary from scratch as decoding proceeds. LZW is used in GIF image encoding, where each pixel value is representable as a byte, hence a symbol.

A non-entropy approach to information was pioneered by Kolmogorov: the information in a structure should be measured as its Minimum Description Length, or MDL, this being more intrinsic than a probabilistic approach. We discuss examples in which the MDL principle is used to build prior knowledge into the description language and to determine the best model for a situation.

Returning to Shannon entropy, we consider protection of information during its transmission, by encoding symbols in a redundant way. Suppose k message symbols average n codeword symbols X, which are received as codeword symbols Y. The *rate* of transmission is then $R = k/n$. We prove Shannon's famous *Channel Coding Theorem*, which says that the transition probabilities $\{p(y|x)\}$ of the channel determine a quantity called the *channel capacity* C, and that, for any rate $R < C$ and probability $\varepsilon > 0$, there is a code with rate R and

$$P(\text{symbol error } Y \neq X) < \varepsilon.$$

The codes exist, but how hard are they to describe, and are they usable? Until recent years the search was for codes with plenty of structure, so that convenient algorithms could be produced for encoding and decoding. The codewords usually had alphabet $\{0, 1\}$, fixed length, and formed a vector space at the least. Good examples are the Reed–Solomon codes of Section 13.2.4 used for the first CD players, which in consequence could be surprisingly much abused before sound quality was affected.

A new breakthrough in closeness to the Shannon capacity came with the turbocodes of Berrou *et al.* (Section 13.3.4), probabilistic unlike earlier codes, but with effective encoding and decoding. They depend on belief propagation in Bayesian nets (Section 13.3.1), where Belief$(x) = p(x|\boldsymbol{e})$ quantifies our belief about internal node variables x in the light of evidence $\boldsymbol{e}$, the end node variables. Propagation refers to the algorithmic updating of Belief(x) on receipt of new information. We finish with a review of belief propagation in computer vision.

Chapters 12–13: a quick trip Look up Shannon's *entropy* at (12.7) giving least bits per symbol, Theorem 12.8. Below this, read 'codetrees', then Huffman's optimal codes in Construction 12.12 and Example 12.13. Proceed to LZW compression in Section 12.7 up to Example 12.38, then Table 12.7 and Figure 12.20.

For Kolmogorov's alternative to entropy and why, read Section 12.8.1 up to (12.34) and their ultimate convergence, Theorem 12.54. For applications see Section 12.8.3 up to 'some MDL features' and Figure 12.26 to 'Further examples'.

Get the idea of a *channel* from Section 13.1 up to *mutual entropy*, (13.3), then Figure 13.2 up to 'Exercise'. Look up *capacity* at (13.23) (don't worry about $C(\beta)$ for now). Next, channel coding in Section 13.1.6 to Example 13.33, the *Hamming code*, and we are ready for the Channel Coding Theorem at Corollary 13.36.

Read the discussion that starts Section 13.2.5. Get some idea of convolution codes at Section 13.3.2 to Figure 13.33, and turbocodes at Figures 13.39 and 13.40. For the belief network basis of their probabilistic handling, look back at Section 13.3.1 to Figure 13.24, then the Markov chain case in Figure 13.25 and above. More generally Figure 13.26. Finally, read the postscript on belief networks in Computer Vision.

Chapters 14–16 (Part V) With suitable transforms we can carry out a huge variety of useful processes on a computer image, for example edge-detection, noise removal, compression, reconstruction, and supplying features for a Bayesian classifier.

Our story begins with the Fourier Transform, which converts a function $f(t)$ to a new function $F(s)$, and its relative the N-point Discrete Fourier Transform or DFT, in which f and F are N-vectors:

$$F(s) = \int_{-\infty}^{\infty} f(t)\mathrm{e}^{-2\pi \mathrm{i}st}\,\mathrm{d}t, \quad \text{and} \quad F_k = \sum_{n=0}^{N-1} \mathrm{e}^{-2\pi \mathrm{i}kn/N} f_n.$$

We provide the background for calculus on complex numbers. Significantly, the relations between numbers of the form $\mathrm{e}^{-2\pi \mathrm{i}k/N}$ result in various forms of *Fast* Fourier Transform, in which the number of arithmetic operations for the DFT is reduced from order N^2 to

order $N \log_2 N$, an important saving in practice. We often need a *convolution* f^*g (see Part III), and the Fourier Transform sends

$$f^*g \to F \circ G \text{ (Convolution Theorem)},$$

the easily computed *elementwise* product, whose value at x is $F(x)G(x)$; similarly for the DFT. We discuss the DFT as approximation to the continuous version, and the significance of frequencies arising from the implied sines and cosines. In general a 1D transform yields an n-dimensional one by transforming with respect to one variable/dimension at a time. If the transform is, like the DFT, given by a matrix M, sending vector $f \to Mf$, then the 2D version acts on a matrix array g by

$$g \to MgM^{\mathrm{T}} \ (= G),$$

which means we transform each column of g then each row of the result, or vice versa, the order being unimportant by associativity of matrix products. Notice $g = M^{-1}G(M^{\mathrm{T}})^{-1}$ inverts the transform. The DFT has matrix $M = [w^{kn}]$, where $w = \mathrm{e}^{-2\pi \mathrm{i}/N}$, from which there follow important connections with rotation (Figure 15.4) and with statistical properties of an image. The Convolution Theorem extends naturally to higher dimensions.

We investigate *highpass filters* on images, convolution operations which have the effect of reducing the size of Fourier coefficients F_{jk} for *low* frequencies j, k, and so preserving details such as edges but not shading (lowpass filters do the opposite). We compare edge-detection by the Sobel, Laplacian, and Marr–Hildreth filters. We introduce the technique of *deconvolution* to remove the effect of image noise such as blur, whether by motion, lens inadequacy or atmosphere, given the reasonable assumption that this effect may be expressed as convolution of the original image g by a small array h. Thus we consider

$$\text{blurred image} = g^*h \to G \circ H.$$

We give ways to find H, hence G by division, then g by inversion of the transform (see Section 15.3). For the case when noise other than blur is present too, we use probability considerations to derive the *Wiener filter*. Finally in Chapter 15 we investigate compression by the Burt–Adelson pyramid approach, and by the Discrete Cosine Transform, or DCT. We see why the DCT is often a good approximation to the statistically based K–L Transform.

In Chapter 16 we first indicate the many applications of fractal dimension as a parameter, from the classical coastline measurement problem through astronomy to medicine, music, science and engineering. Then we see how the 'fractal nature of Nature' lends itself to fractal compression.

Generally the term *wavelets* applies to a collection of functions $\Psi_i^j(x)$ obtained from a *mother wavelet* $\Psi(x)$ by repeated translation, and scaling in the ratio $1/2$. Thus

$$\Psi_i^j(x) = \Psi(2x^j x - i), \quad 0 \le i < 2^j.$$

We start with *Haar* wavelets, modelled on the split box $\Psi(x)$ equal to 1 on $[0, 1/2)$, to -1 on $[1/2, 1]$ and zero elsewhere. With respect to the inner product $\langle f, g\rangle = \int f(x)g(x)\,\mathrm{d}x$

for functions on [0, 1] the wavelets are mutually orthogonal. For fixed resolution J, the *wavelet transform* is

$$f \to \text{its components with respect to } \phi_0(x) \text{ and } \Psi_i{}^j(x),\ 0 \le j \le J,\ 0 \le i < 2^j,$$

where $\phi_0(x)$ is the box function with value 1 on [0, 1]. Converted to 2D form in the usual way, this gives multiresolution and compression for computer images. We pass from resolution level j to $j + 1$ by adding the appropriate extra components. For performing the same without necessarily having orthogonality, we show how to construct the *filter bank*, comprising matrices which convert between components at different resolutions. At this stage, though, we introduce the orthogonal wavelets of Daubechies of which Haar is a special case. These are applied to multiresolution of a face, then we note the use for fingerprint compression.

Lastly in Part V, we see how the Gabor Transform and the edge-detectors of Canny and of Marr and Hildreth may be expressed as wavelets, and outline the results of Lu, Healy and Weaver in applying a wavelet transform to enhance constrast more effectively than other methods, for X-ray and NMR images.

Chapters 14–16: a quick trip Look at Equations (14.1) to (14.4) for the DFT, or Discrete Fourier Transform. Include Notation 14.3 for complex number foundations, then Figure 14.3 for the important frequency viewpoint, and Figure 14.6 for the related filtering schema.

For an introduction to convolution see Example 14.11, then follow the polynomial proof of Theorem 14.12. Read Remarks 14.14 about the Fast Transform (more details in Section 14.1.4). Read up the continuous Fourier Transform in Section 14.2.1 up to Figure 14.13, noting Theorem 14.22. For the continuous–discrete connection, see points 1, 2, 3 at the end of Chapter 14, referring back when more is required.

In Chapter 15, note the easy conversion of the DFT and its continuous counterpart to two dimensions, in (15.6) and (15.10). Observe the effect of having periodicity in the image to be transformed, Figure 15.3, and of rotation, Figure 15.4.

Notice the case of 2D convolution in Example 15.14 and the convolution Theorems 15.16 and 15.17. Look through the high-versus lowpass material in Sections 15.2.2 and 15.2.3, noting Figures 15.15, 15.18, and 15.20. Compare edge-detection filters with each other in Figure 15.23. Read up recovery from motion blur in Section 15.3.1, omitting proofs.

For the pyramid compression of Burt and Adelson read Section 15.4 up to Figure 15.37 and look at Figures 15.39 and 15.40. For the DCT (Discrete Cosine Transform) read Section 15.4.2 up to Theorem 15.49 (statement only). Note the standard conversion to 2D in Table 15.8, then see Figures 15.42 and 15.43. Now read the short Section 15.4.3 on JPEG. Note for future reference that the n-dimensional Fourier Transform is covered, with proofs, in Section 15.5.2.

For fractal dimension read Sections 16.1.1 and 16.1.2, noting at a minimum the key formula (16.9) and graph below. For fractal compression take in Section 16.1.4 up to

(16.19), then Example 16.6. A quick introduction to wavelets is given at the start of Section 16.2, then Figure 16.23. Moving to two dimensions, see Figure 16.25 and its introduction, and for image compression, Figure 16.27.

A pointer to filter banks for the discrete Wavelet Transform is given by Figure 16.28 with its introduction, and (16.41). Now check out compression by Daubechies wavelets, Example 16.24. Take a look at wavelets for fingerprints, Section 16.3.4. Considering wavelet relatives, look at Canny edge-detection in Section 16.4.3, then scan quickly through Section 16.4.4, slowing down at the medical application in Example 16.28.

Chapters 17–18 (Part VI) B-splines are famous for their curve design properties, which we explore, along with the connection to convolution, Fourier Transform, and wavelets.

The ith basis function $N_{i,m}(t)$ for a B-spline of order m, degree m-1, may be obtained as a translated convolution product $b^*b^*\cdots^*b$ of m unit boxes $b(t)$. Consequently, the function changes to a different polynomial at unit intervals of t, though smoothly, then becomes zero. Convolution supplies a polynomial-free definition, its simple Fourier Transform verifying the usually used Cox–de Boor defining relations. Unlike a Bézier spline, which for a large control polygon $P_0 \dots P_n$ requires many spliced component curves, the B-spline is simply

$$B_m(t) = \sum_{i=0}^{n} N_{i,m}(t) P_i.$$

We elucidate useful features of $B_m(t)$, then design a car profile, standardising on cubic splines, $m = 4$. Next we obtain B-splines by recursive subdivision starting from the control polygon. That is, by repetitions of

$$\text{subdivision} = \text{split} + \text{average},$$

where *split* inserts midpoints of each edge and *average* replaces each point by a linear combination of neighbours. We derive the coefficients as binomials, six subdivisions usually sufficing for accuracy. We recover basis functions, now denoted by $\phi_1^j(x)$, starting from hat functions. In the previous wavelet notation we may write

$$\Phi^{j-1} = \Phi^j P^j \text{ (basis)}, \quad \text{where } f^j = P^j f^{j-1},$$

the latter expressing level $j-1$ vectors in terms of level j via a matrix P^j. Now we aim for a filter bank so as to edit cubic-spline-based images. It is (almost) true that for our splines (a) for fixed j the basis functions are translates, (b) those at level $j+1$ are scaled from level j. As before, we take $V^j = \text{span } \Phi^j$ and choose wavelet space $W^{j-1} \subseteq V^j$ to consist of the functions in V^j orthogonal to all those in V^{j-1}. It follows that any f in V^j equals $g + h$ for unique g in V^{j-1} and h in W^{j-1}, this fact being expressed by

$$V^{j-1} \oplus W^{j-1} = V^j.$$

A basis of W^{j-1} (the wavelets) consists of linear combinations from V^j, say the vector of functions $\Psi^{j-1} = \Phi^j Q^j$ for some matrix Q^j. Orthogonality leaves many possible Q^j, and we may choose it to be antipodal (half-turn symmetry), so that one half determines

the rest. This yields matrices P, Q, A, B for a filter bank, with which we perform editing at different scales based on (for example) a library of B-spline curves for components of a human face.

In the first appendix we see how to determine simple formulae for filter bank matrices, using connections with polynomials. The second appendix introduces surfaces wavelets as a natural generalisation from curves, and we indicate how a filter bank may be obtained once more.

(Chapter 18) An artificial neural network, or just net, may be thought of firstly in pattern recognition terms, converting an input vector of pixel values to a character they represent. More generally, a permissible input vector is mapped to the correct output by a process in some way analogous to the neural operation of the brain (Figure 18.1). We work our way up from Rosenblatt's Perceptron, with its rigorously proven limitations, to multilayer nets which in principle can mimic any input–output function. The idea is that a net will generalise from suitable input–output examples by setting free parameters called *weights*. We derive the Backpropagation Algorithm for this, from simple gradient principles. Examples are included from medical diagnosis and from remote sensing.

Now we consider nets that are mainly *self-organising*, in that they construct their own categories of classification. We include the topologically based Kohonen method (and his Learning Vector Quantisation). Related nets give an alternative view of Principal Component Analysis. At this point Shannon's extension of entropy to the continuous case opens up the criterion of Linsker that neural network weights should be chosen to maximise mutual information between input and output. We include a 3D image processing example due to Becker and Hinton. Then the further Shannon theory of rate distortion is applied to vector quantisation and the LBG quantiser.

Now enters the Hough Transform and its widening possibilities for finding arbitray shapes in an image. We end with the related idea of *tomography*, rebuilding an image from projections. This proves a fascinating application of the Fourier Transform in two and even in three dimensions, for which the way was prepared in Chapter 15.

Chapters 17–18: a quick trip Go straight to the convolution definition, (17.7), and result in Figure 17.7, of the ϕ_k whose translates, (17.15) and Figure 17.10, are the basis functions for B-splines. (Note the Fourier calculation below Table 17.1.) See the B-spline Definition 17.13, Theorem 17.14, Figure 17.12, and car body Example 17.18. Observe B-splines generated by recursive subdivision at Examples 17.33 and 17.34.

We arrive at filter banks and curve editing by Figure 17.32 of Section 17.3.3. Sample results at Figure 17.37 and Example 17.46. For an idea of surface wavelets, see Figures 17.51 and 17.52 of the second appendix.

Moving to artificial neural networks, read *Perceptron* in Section 18.1.2 up to Figure 18.5, note the training ALGO 18.1, then go to Figure 18.15 and Remarks following. Proceed to the multilayer net schema, Figure 18.17, read 'Discovering Backpropagation' as far as desired, then on to Example 18.11. For more, see the remote sensing Example 18.16.

Now for self-organising nets. Read the introduction to Section 18.2, then PCA by Oja's method at (18.28) with discussion following, then the k-means method at Equation (18.30) and Remarks 18.20. Consider Kohonen's topologically based nets via Example 18.21 (note the use of 'neighbourhoods') and remarks following.

Revisit information theory with *differential entropy* in Table 18.3, and the Gaussian case in Theorem 18.29. Now observe the application of mutual entropy to nets, in Example 18.34 down to Equation (18.47). Pick up rate distortion from (18.60) and the 'compression interpretation' below, then look at Theorem 18.48 (without proof) and Example 18.49. With notation from (18.67) and (18.68), note Theorem 18.50. Read Section 18.3.6 to find steps A, B then see the LBG quantization in Example 18.59 and the discussion following.

The last topic is *tomography*. Read through Section 18.4.2 then note the key projection property, (18.79), and the paragraph below it. Observe Figure 18.63, representing the interpolation step, then see the final result in Examples 18.65 and 66. Finally, note 'higher dimensions'.

Which chapters depend on which

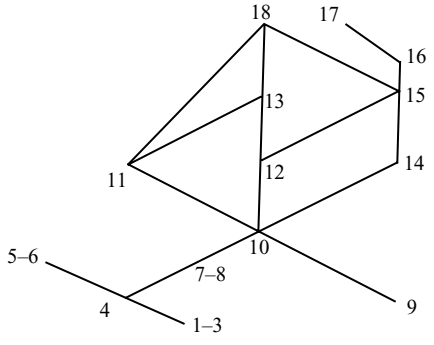
1–6	Each chapter depends on the previous ones.
7	Depends generally on Chapter 1.
8	Depends strongly on Chapter 7.
9	Little reliance on previous chapters. Uses some calculus.
10	Depends strongly on Chapters 8 and 9.
11	Builds on Chapter 10.
12	Basic probability from Chapter 9; last section uses random vectors from Chapter 10.
13	Section 13.1 develops entropy from Section 12.1, whilst Section 13.2 uses vector space bases from Section 7.1.5. Belief networks in Section 13.3 recapitulates Section 11.4.4 first.
14	Uses matrices from Section 7.2, complex vectors and matrices from Section 8.1.1, convolution from Section 10.2.2; evaluating the FFT uses big O notation of Section 10.3.3.
15	Builds on Chapter 14. The Rotation Theorem of Section 15.1.3 uses the Jacobian from Section 10.2.1, rotation from Section 7.4.1. Filter symmetry in Section 15.2.3 uses Example 8.21(iii). The Wiener filter, Section 15.3.4, needs functions of a random vector, Section 10.2, and covariance, Section 10.4.2. Compression, Section 15.4, uses entropy from Chapter 12.
16	Fractals, Section 16.1, uses the regression line from Section 11.1.4. Sections 16.2 and 16.3 use vector spaces from Section 7.1.5, with inner product as in (7.8), and the block matrices of Section 7.2.5. Also the general construction of 2D transforms in Section 15.1.1, and the DCT in Section 15.4.2. Section 16.4 makes wide use of the Fourier Transform from Chapters 14 and 15.

17 Depending strongly on Section 16.2 and Section 16.3, this chapter also requires knowledge of the 1D Fourier Transform of Chapter 14, whilst Section 17.3.2 uses dependent vectors from Section 7.1.5 and symmetry from Example 8.21(iii).

18 The first three sections need probabililty, usually not beyond Chapter 10 except for Bayes at Section 11.2. Section 18.3 builds on the mutual entropy of Chapter 13, whilst Section 18.4.1 uses the Sobel edge-detectors of Section 15.2.4, the rest (Hough and tomography) requiring the Fourier Transform(s) and Projection Theorem of Section 15.1.

Table of crude chapter dependencies.

A chapter depends on those it can 'reach' by going down the graph.



Some paths to special places

Numbers refer to chapters. Fourier Transform means the continuous one and DFT the discrete. ONB is orthonormal basis, PCA is Principal Component Analysis, Gaussian equals normal.

n-dim Gaussian or PCA (a choice)

ONB → eigenvalues/vectors → similarity → covariance → PCA or n-dim Gaussian
7 8 8 10 10

Simulating Gaussians

Independent vectors → moments → Central Limit Theorem → Cholesky factors → simulation
10 10 10 8 11

Huffman codes

Entropy → noiseless encoding → codetrees → Huffman
12 12 12 12

Channel Coding Theorem

Random variables → joint pdf → entropy → mutual entropy → capacity → Shannon Theorem
9 10 12 13 13 13

JPEG

ONBs & orthogonal matrices → complex numbers → Discrete Cosine Transform → JPEG
7 8 15 15

Wiener filter

Complex nos. → Fourier Transform → Convolution Thm → pdfs & Fourier → Wiener filter
8 14,15 15 15 15

Haar Wavelet Transform (images)

Inner product & ONBs → 1D Haar → 2D Haar→ Haar Image Transform
7 16 16 16

B-splines & Fourier

Fourier Transform (1D) → Convolution Thm. → ϕ_k as convolution → Fourier Transform
14 14 17 17

Perceptron

Dot product → perceptron → edge-detector → learning algorithm
7 18 18 18

Vector quantisation

Entropy → mutual entropy → rate distortion → LBG versus k-means
12 13 18 18

Tomography

Complex numbers → DFT → Fourier Transform → Rotation & Projection Thms → Tomography
8 15 15 15 18

A word on notation

(1) (*Vectors*) We write vectors typically as $x = (x_1, \ldots, x_n)$, with the option $x = (x_i)$ if n is known from the context. Bold $\boldsymbol{x}$ emphasises the vector nature of such x.

(2) (*Rows versus columns*) For vector–matrix multiplication we may take vector $\boldsymbol{x}$ as a row, indicated by writing $\boldsymbol{x}A$, or as a column, indicated by $A\boldsymbol{x}$. The row notation is used through Chapters 1–6 in harmony with the image of a point P under a transformation g being denoted by P^g, so that successive operations appear on the right, thus:

$$\boldsymbol{x}ABC\ldots \quad \text{and} \quad P^{fgh\ldots}.$$

Any matrix equation with vectors as rows may be converted to its equivalent in terms of columns, by transposition: e.g. $\boldsymbol{x}A = \boldsymbol{b}$ becomes $A^{\mathrm{T}}\boldsymbol{x}^{\mathrm{T}} = \boldsymbol{b}^{\mathrm{T}}$. Finally, to keep matrices on one line we may write Rows$[R_1, \ldots, R_m]$, or just Rows$[R_i]$, for the matrix with rows R_i, and similarly for columns, Cols$[C_1, \ldots, C_n]$.

(3) (*Block matrices*) Every so often it is expedient to perform multiplication with matrices which have been divided (partitioned) into submatrices called blocks. This is described in Section 7.2.5, where special attention should be paid to 'Russian multiplication'.

(4) (*Distributions*) Provided there is no ambiguity we may use the letter p generically for probability distributions, for example $p(x)$, $p(y)$ and $p(x, y)$ may denote the respective pdfs of random variables X, Y and (X, Y). In a similar spirit, the symbol list following concentrates on those symbols which are used more widely than their first context of definition.

Here too we should mention that the normal and Gaussian distributions are the same thing, the word Gaussian being perhaps preferred by those with a background in engineering.

Symbols

$\|\lambda\|$	Absolute value of real number λ, modulus if complex	*page* 8, 166, 167
$\|AB\|$, $\|\boldsymbol{a}\|$	Length of line segment AB, length of vector $\mathbf{a}$	7
$\underline{AB}$	Line segment directed from point A to B	7
$A(a_1, a_2)$	Point A with Cartesian coordinates (a_1, a_2)	7
$\boldsymbol{a} = (a_1, a_2)$	General vector a or *position vector* of point A	7
$\boldsymbol{a} \cdot \boldsymbol{b}$	Dot/scalar product $\sum a_i b_i$	127
$\boldsymbol{a} \times \boldsymbol{b}$	Vector product of vectors $\boldsymbol{a}, \boldsymbol{b}$	141
$g: X \to Y$	Function (mapping) from X to Y	
P^g and $g(P)$	Image of P under mapping g	11
T_{AB}	The translation that sends point A to point B	17
$T_{\boldsymbol{a}}$	The translation given by $\boldsymbol{x} \to \boldsymbol{x} + \boldsymbol{a}$	11
$R_A(\phi)$	Rotation in the plane about point A, through signed angle ϕ	11, 12
$R_A(m/n)$	Rotation as above, through the fraction m/n of a turn	12
R_m, R_{AB}	(In the plane) reflection in mirror line m, in line through A, B	12
I	The identity mapping, identity element of a group	39, 40
g^{-1}	The inverse of a function or group element g	32
h^g	The product $g^{-1}hg$, for transformations or group elements g, h	29
D_{2n}	The dihedral group of order $2n$	38
$\mathbf{R}$, $\mathbf{Q}$, $\mathbf{Z}$, $\mathbf{N}$	The real numbers, rationals, integers, and naturals 1, 2, 3, ...	23, 24
$[a, b]$, (a, b)	Closed interval $a \le x \le b$, open interval $a < x < b$	230

$\mathbf{R}^n$	Euclidean n-space	120, 121
$\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k}$	Unit vectors defining coordinate axes in 3-space	116
δ_{ik}	The Kronecker delta, equal to 1 if $i = k$, otherwise 0	119
$\delta(x)$	The Dirac delta function	544
$I = I_n$	The identity $n \times n$ matrix	127
$A_{m\times n}$	Matrix A with m rows and n columns	127
a_{ik} or $(A)_{ik}$	The entry in row i, column k, of the matrix A	126, 127
diag $\{d_1, \ldots, d_n\}$	The square matrix whose diagonal elements are d_i, the rest 0	128
A^{T}, A^{-1}	The transpose of matrix A, its inverse if square	128, 129
$\lvert A\rvert$ or det A	The determinant of a square matrix A	131
Tr A	The trace (sum of the diagonal elements a_{ii}) of a matrix A	164, 165
E_{ik}	Matrix whose i, k entry is 1, and the rest 0	140
$\langle A, B\rangle$	Inner product of matrices (as long vectors)	203
$\lVert A\rVert_{\mathrm{F}}$, $\lVert A\rVert_{\mathrm{R}}$	Frobenius and ratio norms of matrix A (subscript F may be omitted)	193
$R(A)$, $N(A)$	Row space, null space, of matrix A	172, 177
$\mathbf{C}$	The complex numbers $z = x + y\mathrm{i}$, where $\mathrm{i}^2 = -1$	162, 163
Re z, Im z, $\bar{z}$, $\lvert z\rvert$	Real part, imaginary part, conjugate, modulus of z	163
$\mathrm{e}^{\mathrm{i}\theta}$	The complex number $\cos\theta + \mathrm{i}\sin\theta$	163
$\{x_n\}_{n\geq 1}$	Sequence $x_1, x_2, x_3, \ldots$	276
$\mathbf{Z}_2$	The field of size 2	469
$\mathcal{A} = \mathcal{A}_X$	Alphabet for a channel random variable X	445
$\mathcal{A}$	Attractor of an Iterated Function system	653
$\log x$, $\log_2 x$, $\ln x$	Logarithm to given base, to base 2, to base e (natural)	398
S	Sample space in probability theory	210
A^{c}, $A \cup B$, $A \cap B$	Complement, union, intersection, of sets or events	210, 211
$P(A)$	Probability that event A occurs	212
${}_n\mathrm{C}_r$ or $\binom{n}{r}$	Number of ways to choose r things from n. Equals $n!/(r!(n-r)!)$	215
$\lvert A\rvert$	Size of set or event A	
$P(A \mid B)$	Conditional probability of A given B	219
$P(X = x)$	Probability that random variable X takes value x	227
$E(X)$ and $V(X)$	Expected value and variance of random variable X	235, 237

$X \sim N(\mu, \sigma^2)$	X is normal (Gaussian) with $E(X) = \mu$, $V(X) = \sigma^2$	245
$\Gamma(x)$	Gamma function	238
$\Gamma_{\alpha,u}(x)$	Gamma distribution with parameters α, u	249
$\overline{X}$	Sample mean	305
$\hat{\theta}$	Estimate of distribution parameter θ	309
f^*g	Convolution product: functions ('continuous'), arrays ('discrete')	271, 531
$h = f \circ g$	Elementwise product $h(x) = f(x)g(x)$	533
$U[a, b]$	Uniform distribution on interval $[a, b]$	230
$\mathrm{Cov}(X, Y)$	Covariance/correlation between random variables X, Y	285
$\mathrm{Cov}(X)$	Covariance matrix of random vector $X = (X_i)$	287
$H(X) =$ $\mathrm{H}(p_1, \ldots, p_n)$	Entropy of random variable X with pdf $\{p_1, \ldots, p_n\}$	397
$H(x)$	Same as $H(x, 1-x)$, $0 \le x \le 1$	399
$d(p\|q)$	Kullback–Liebler distance between probability distributions $p(x)$, $q(x)$	432
$H(X, Y)$, $H(X\|Y)$	Joint entropy, conditional entropy	446
$I(X; Y)$	Mutual entropy	446
$f \to F$	Fourier Transform. F is also written $\hat{f}$ or $\mathcal{F}[f]$	523, 524, 542
R_{fg}	Cross-correlation of functions f, g	544
$\mathrm{sinc}(x)$	$(\sin \pi x)/\pi x$ (sometimes $(\sin x)/x$, as in Section 14.3)	542
$\nabla^2 f$	Laplacian of f	573
LoG, DoG	Laplacian of Gaussian, Difference of Gaussians	594
$\psi(x)$, $\psi_i^j(x)$	Mother wavelet, derived wavelets	659
W^j, $\Psi^j(x)$	Wavelet space, basis	665
V^j, $\Phi^j(x)$	Scaling space, basis	665
$V \bigoplus W$	Sum of vector spaces	662
P, Q, A, B	Filter bank matrices	666, 667
$P_0 \ldots P_n$	Control polygon	668
$b(t)$, $\phi_k(x)$	Box function, spline function	693
$N_{i,m}(x)$	B-spline basis function of order m, degree m-1,	698
$(\ldots, r_{-1}, r_0, r_1, \ldots)$	Averaging mask for subdivision	711
$e_i^j(x)$	Hat function	711
$\langle f, h \rangle$	Inner product of functions f and h	660, 748
G	Gram matrix of inner products: $g_{ik} = \langle f_i, h_k \rangle$ (allows $h = f$)	721

$f_*(x)$	Reciprocal polynomial of $f(x)$ (coefficents in reverse order)	483
$\tanh(x)$	Hyperbolic tangent $(e^x - e^{-x})/(e^x + e^{-x})$	769
$\Delta \boldsymbol{w}$	Update to weight vector $\boldsymbol{w}$	771
$h(X)$	Differential entropy of continuous random variable X	794
R, $R(D)$	Code rate, Shannon's Rate Distortion function	464, 805