

*Digital Signal Processing: A Computer Science Perspective*

Jonathan Y. Stein

Copyright © 2000 John Wiley & Sons, Inc.

Print ISBN 0-471-29546-9 Online ISBN 0-471-20059-X

# **Digital Signal Processing**

# **Digital Signal Processing**

## **A Computer Science Perspective**

**Jonathan (Y) Stein**



**A Wiley-Interscience Publication**  
**JOHN WILEY & SONS, INC.**

**New York • Chichester • Weinheim • Brisbane • Singapore • Toronto**

Copyright © 2000 by John Wiley & Sons, Inc. All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic or mechanical, including uploading, downloading, printing, decompiling, recording or otherwise, except as permitted under Sections 107 or 108 of the 1976 United States Copyright Act, without the prior written permission of the Publisher. Requests to the Publisher for permission should be addressed to the Permissions Department, John Wiley & Sons, Inc., 605 Third Avenue, New York, NY 10158-0012, (212) 850-6011, fax (212) 850-6008, E-Mail: PERMREQ @ WILEY.COM.

This publication is designed to provide accurate and authoritative information in regard to the subject matter covered. It is sold with the understanding that the publisher is not engaged in rendering professional services. If professional advice or other expert assistance is required, the services of a competent professional person should be sought.

**ISBN 0-471-20059-X.**

This title is also available in print as ISBN 0-471-29546-9

For more information about Wiley products, visit our web site at [www.Wiley.com](http://www.Wiley.com).

***Library of Congress Cataloging in Publication Data***

Stein, Jonathan Y.

Digital signal processing: a computer science perspective/Jonathan Y. Stein  
p. cm.

“A Wiley-Interscience publication.”

Includes bibliographical references and index.

ISBN 0-471-29546-9 (cloth: alk. paper)

1. Signal processing—Digital techniques. I. Title.

TK5102.9. S745 2000  
621.382'2—dc21

00-035905

Printed in the United States of America.

10 9 8 7 6 5 4 3 2 1

*To Ethel, Hanna and Noga*

# Contents

<b>Preface</b>	<b>xv</b>	
<b>1</b>	<b>Introductions</b>	<b>1</b>
1.1	Prehistory of DSP . . . . .	2
1.2	Some Applications of Signal Processing . . . . .	4
1.3	Analog Signal Processing . . . . .	7
1.4	Digital Signal Processing . . . . .	10
<b>Part I</b>	<b>Signal Analysis</b>	
<b>2</b>	<b>Signals</b>	<b>15</b>
2.1	<i>Signal</i> Defined . . . . .	15
2.2	The Simplest Signals . . . . .	20
2.3	Characteristics of Signals . . . . .	30
2.4	Signal Arithmetic . . . . .	33
2.5	The Vector Space of All Possible Signals . . . . .	40
2.6	<i>Time</i> and <i>Frequency</i> Domains . . . . .	44
2.7	Analog and Digital Domains . . . . .	47
2.8	Sampling . . . . .	49
2.9	Digitization . . . . .	57
2.10	Antialiasing and Reconstruction Filters . . . . .	62
2.11	Practical Analog to Digital Conversion . . . . .	64
<b>3</b>	<b>The Spectrum of Periodic Signals</b>	<b>71</b>
3.1	Newton's Discovery . . . . .	72
3.2	Frequency Components . . . . .	74
3.3	Fourier's Discovery . . . . .	77
3.4	Representation by Fourier Series . . . . .	80
3.5	Gibbs Phenomenon . . . . .	86
3.6	Complex FS and Negative Frequencies . . . . .	90

3.7	Properties of Fourier Series . . . . .	94
3.8	The Fourier Series of Rectangular Wave . . . . .	96
<b>4</b>	<b>The Frequency Domain</b>	<b>103</b>
4.1	From Fourier Series to Fourier Transform . . . . .	103
4.2	Fourier Transform Examples . . . . .	110
4.3	FT Properties . . . . .	113
4.4	The Uncertainty Theorem . . . . .	117
4.5	Power Spectrum . . . . .	122
4.6	Short Time Fourier Transform (STFT) . . . . .	126
4.7	The Discrete Fourier Transform (DFT) . . . . .	132
4.8	DFT Properties . . . . .	135
4.9	Further Insights into the DFT . . . . .	141
4.10	The $z$ Transform . . . . .	143
4.11	More on the $z$ Transform . . . . .	151
4.12	The Other Meaning of Frequency . . . . .	155
<b>5</b>	<b>Noise</b>	<b>161</b>
5.1	Unpredictable Signals . . . . .	162
5.2	A Naive View of Noise . . . . .	164
5.3	Noise Reduction by Averaging . . . . .	171
5.4	Pseudorandom Signals . . . . .	174
5.5	Chaotic Signals . . . . .	180
5.6	Stochastic Signals . . . . .	192
5.7	Spectrum of Random Signals . . . . .	198
5.8	Stochastic Approximation Methods . . . . .	202
5.9	Probabilistic Algorithms . . . . .	203
<b>Part II</b>	<b>Signal Processing Systems</b>	
<b>6</b>	<b>Systems</b>	<b>207</b>
6.1	<i>System</i> Defined . . . . .	208
6.2	The Simplest Systems . . . . .	209
6.3	The Simplest Systems with Memory . . . . .	213
6.4	Characteristics of Systems . . . . .	221
6.5	Filters . . . . .	226
6.6	Moving Averages in the Time Domain . . . . .	228
6.7	Moving Averages in the Frequency Domain . . . . .	231
6.8	Why Convolve? . . . . .	237

6.9	Purely Recursive Systems . . . . .	241
6.10	Difference Equations . . . . .	245
6.11	The Sinusoid's Equation . . . . .	249
6.12	System Identification—The Easy Case . . . . .	252
6.13	System Identification—The Hard Case . . . . .	259
6.14	System Identification in the $z$ Domain . . . . .	265
<b>7</b>	<b>Filters</b>	<b>271</b>
7.1	Filter Specification . . . . .	272
7.2	Phase and Group Delay . . . . .	275
7.3	Special Filters . . . . .	279
7.4	Feedback . . . . .	289
7.5	The ARMA Transfer Function . . . . .	293
7.6	Pole-Zero Plots . . . . .	298
7.7	Classical Filter Design . . . . .	303
7.8	Digital Filter Design . . . . .	309
7.9	Spatial Filtering . . . . .	315
<b>8</b>	<b>Nonfilters</b>	<b>321</b>
8.1	Nonlinearities . . . . .	322
8.2	Clippers and Slicers . . . . .	324
8.3	Median Filters . . . . .	326
8.4	Multilayer Nonlinear Systems . . . . .	329
8.5	Mixers . . . . .	332
8.6	Phase-Locked Loops . . . . .	338
8.7	Time Warping . . . . .	343
<b>9</b>	<b>Correlation</b>	<b>349</b>
9.1	Signal Comparison and Detection . . . . .	350
9.2	Crosscorrelation and Autocorrelation . . . . .	354
9.3	The Wiener-Khinchine Theorem . . . . .	357
9.4	The Frequency Domain Signal Detector . . . . .	359
9.5	Correlation and Convolution . . . . .	361
9.6	Application to Radar . . . . .	362
9.7	The Wiener Filter . . . . .	365
9.8	Correlation and Prediction . . . . .	369
9.9	Linear Predictive Coding . . . . .	371
9.10	The Levinson-Durbin Recursion . . . . .	376
9.11	Line Spectral Pairs . . . . .	383
9.12	Higher-Order Signal Processing . . . . .	386

<b>10</b>	<b>Adaptation</b>	<b>393</b>
10.1	Adaptive Noise Cancellation . . . . .	394
10.2	Adaptive Echo Cancellation . . . . .	400
10.3	Adaptive Equalization . . . . .	404
10.4	Weight Space . . . . .	408
10.5	The LMS Algorithm . . . . .	413
10.6	Other Adaptive Algorithms . . . . .	420
<b>11</b>	<b>Biological Signal Processing</b>	<b>427</b>
11.1	Weber's Discovery . . . . .	428
11.2	The Birth of Psychophysics . . . . .	430
11.3	Speech Production . . . . .	435
11.4	Speech Perception . . . . .	439
11.5	Brains and Neurons . . . . .	442
11.6	The Essential Neural Network . . . . .	446
11.7	The Simplest Model Neuron . . . . .	448
11.8	Man vs. Machine . . . . .	452
<b>Part III</b>	<b>Architectures and Algorithms</b>	
<b>12</b>	<b>Graphical Techniques</b>	<b>461</b>
12.1	Graph Theory . . . . .	462
12.2	DSP Flow Graphs . . . . .	467
12.3	DSP Graph Manipulation . . . . .	476
12.4	RAX Externals . . . . .	481
12.5	RAX Internals . . . . .	487
<b>13</b>	<b>Spectral Analysis</b>	<b>495</b>
13.1	Zero Crossings . . . . .	496
13.2	Bank of Filters . . . . .	498
13.3	The Periodogram . . . . .	502
13.4	Windows . . . . .	506
13.5	Finding a Sinusoid in Noise . . . . .	512
13.6	Finding Sinusoids in Noise . . . . .	515
13.7	IIR Methods . . . . .	520
13.8	Walsh Functions . . . . .	523
13.9	Wavelets . . . . .	526

<b>14</b>	<b>The Fast Fourier Transform</b>	<b>531</b>
14.1	Complexity of the DFT . . . . .	532
14.2	Two Preliminary Examples . . . . .	536
14.3	Derivation of the DIT FFT . . . . .	539
14.4	Other Common FFT Algorithms . . . . .	546
14.5	The Matrix Interpretation of the FFT . . . . .	552
14.6	Practical Matters . . . . .	554
14.7	Special Cases . . . . .	558
14.8	Goertzel's Algorithm . . . . .	561
14.9	FIFO Fourier Transform . . . . .	565
<b>15</b>	<b>Digital Filter Implementation</b>	<b>569</b>
15.1	Computation of Convolutions . . . . .	570
15.2	FIR Filtering in the Frequency Domain . . . . .	573
15.3	FIR Structures . . . . .	579
15.4	Polyphase Filters . . . . .	584
15.5	Fixed Point Computation . . . . .	590
15.6	IIR Structures . . . . .	595
15.7	FIR vs. IIR . . . . .	602
<b>16</b>	<b>Function Evaluation Algorithms</b>	<b>605</b>
16.1	Sine and Cosine Generation . . . . .	606
16.2	Arctangent . . . . .	609
16.3	Logarithm . . . . .	610
16.4	Square Root and Pythagorean Addition . . . . .	611
16.5	CORDIC Algorithms . . . . .	613
<b>17</b>	<b>Digital Signal Processors</b>	<b>619</b>
17.1	Multiply-and-Accumulate (MAC) . . . . .	620
17.2	Memory Architecture . . . . .	623
17.3	Pipelines . . . . .	627
17.4	Interrupts, Ports . . . . .	631
17.5	Fixed and Floating Point . . . . .	633
17.6	A Real-Time Filter . . . . .	635
17.7	DSP Programming Projects . . . . .	639
17.8	DSP Development Teams . . . . .	641

## Part IV Applications

<b>18</b>	<b>Communications Signal Processing</b>	<b>647</b>
18.1	History of Communications . . . . .	648
18.2	Analog Modulation Types . . . . .	652
18.3	AM . . . . .	655
18.4	FM and PM . . . . .	659
18.5	Data Communications . . . . .	664
18.6	Information Theory . . . . .	666
18.7	Communications Theory . . . . .	670
18.8	Channel Capacity . . . . .	674
18.9	Error Correcting Codes . . . . .	680
18.10	Block Codes . . . . .	683
18.11	Convolutional Codes . . . . .	690
18.12	PAM and FSK . . . . .	698
18.13	PSK . . . . .	704
18.14	Modem Spectra . . . . .	708
18.15	Timing Recovery . . . . .	710
18.16	Equalization . . . . .	714
18.17	QAM . . . . .	716
18.18	QAM Slicers . . . . .	720
18.19	Trellis Coding . . . . .	723
18.20	Telephone-Grade Modems . . . . .	729
18.21	Beyond the Shannon Limit . . . . .	733
<b>19</b>	<b>Speech Signal Processing</b>	<b>739</b>
19.1	LPC Speech Synthesis . . . . .	740
19.2	LPC Speech Analysis . . . . .	742
19.3	Cepstrum . . . . .	744
19.4	Other Features . . . . .	747
19.5	Pitch Tracking and Voicing Determination . . . . .	750
19.6	Speech Compression . . . . .	753
19.7	PCM . . . . .	757
19.8	DPCM, DM, and ADPCM . . . . .	760
19.9	Vector Quantization . . . . .	765
19.10	SBC . . . . .	768
19.11	LPC Speech Compression . . . . .	770
19.12	CELP Coders . . . . .	771
19.13	Telephone-Grade Speech Coding . . . . .	775

<b>A</b>	<b>Whirlwind Exposition of Mathematics</b>	<b>781</b>
A.1	Numbers . . . . .	781
A.2	Integers . . . . .	782
A.3	Real Numbers . . . . .	784
A.4	Complex Numbers . . . . .	785
A.5	Abstract Algebra . . . . .	788
A.6	Functions and Polynomials . . . . .	791
A.7	Elementary Functions . . . . .	793
A.8	Trigonometric (and Similar) Functions . . . . .	795
A.9	Analysis . . . . .	800
A.10	Differential Equations . . . . .	803
A.11	The Dirac Delta . . . . .	808
A.12	Approximation by Polynomials . . . . .	809
A.13	Probability Theory . . . . .	815
A.14	Linear Algebra . . . . .	819
A.15	Matrices . . . . .	821
A.16	Solution of Linear Algebraic Equations . . . . .	826
	<b>Bibliography</b>	<b>829</b>
	<b>Index</b>	<b>849</b>

# Preface

I know what you are asking yourself—‘there are a lot of books available about DSP, is this book the one for *me*?’ Well that depends on who *you* are. If

- you are interested in doing research and development in one of the many state-of-the-art applications of DSP, such as speech compression, speech recognition, or modem design,
- your main proficiency is in computer science, abstract mathematics, or science rather than electronics or electrical engineering,
- your mathematical background is relatively strong (flip back now to the appendix—you should be comfortable with about half of what you see there),

then you are definitely in the *target group* of this book. If in addition

- you don’t mind a challenge and maybe even enjoy tackling brain-teasers,
- you’re looking for one comprehensive text in all aspects of DSP (even if you don’t intend reading all of it now) and don’t want to have to study several different books with inconsistent notations, in order to become competent in the subject,
- you enjoy and learn more from texts with a light style (such as have become common for computer science texts) rather than formal, dry tomes that introduce principles and thereafter endlessly derive corollaries thereof,

then this is probably the book you have been waiting for.

This book is the direct result of a chain of events, the first link of which took place in mid-1995. I had been working at a high-tech company in Tel

Aviv that was a subsidiary of a New York company. In Tel Aviv it was relatively easy to locate and hire people knowledgeable in all aspects of DSP, including speech processing, digital communications, biomedical applications, and digital signal processor programming. Then, in 1995, I relocated to a different subsidiary of the same company, located on Long Island, New York. One of my first priorities was to locate and hire competent DSP software personnel, for work on speech and modem signal processing.

A year-long search turned up next to no-one. Assignment agencies were uncertain as to what *DSP* was, advertisements in major New York area newspapers brought irrelevant responses (digital design engineers, database programmers), and, for some inexplicable reason, attempts to persuade more appropriate people from Silicon Valley to leave the California climate, for one of the worst winters New York has ever seen, failed.

It struck me as rather odd that there was no indigenous DSP population to speak of, in an area noted for its multitude of universities and diversity of high-tech industries. I soon found out that DSP was not taught at undergraduate level at the local universities, and that even graduate-level courses were not universally available. Courses that *were* offered were Electrical Engineering courses, with Computer Science students never learning about the subject at all. Since I was searching for people with algorithm development and coding experience, preferably strong enough in software engineering to be able to work on large, complex software systems, CS graduates seemed to be more appropriate than EEs. The ideal candidate would be knowledgeable in DSP and would in the *target group* mentioned above.

Soon after my move to New York I had started teaching graduate level courses, in Artificial Intelligence and Neural Networks, at the Computer and Informations Sciences department of Polytechnic University. I inquired of the department head as to why a DSP course was not offered to Computer Science undergraduates (it *was* being offered as an elective to Electrical Engineering graduate students). He replied that the main reason was lack of a suitable teacher, a deficiency that could be easily remedied by my volunteering.

I thus found myself 'volunteered' to teach a new Computer Science undergraduate elective course in DSP. My first task was to decide on course goals and to flesh out a syllabus. It was clear to me that there would be little overlap between the CS undergraduate course and the EE graduate-level course. I tried to visualize the ideal candidate for the positions I needed to fill at my company, and set the course objectives in order to train the perfect candidate. The objectives were thus:

- to give the student a basic understanding of the theory and practice of DSP, at a level sufficient for reading journal articles and conference papers,
- to cover the fundamental algorithms and structures used in DSP computation, in order to enable the student to correctly design and efficiently code DSP applications in a high-level language,
- to explain the principles of digital signal processors and the differences between them and conventional CPUs, laying the framework for the later in-depth study of assembly languages of specific processors,
- to review the background and special algorithms used in several important areas of state-of-the-art DSP research and development, including speech compression/recognition, and digital communications,
- to enable the student who completes the course to easily fit in and contribute to a high-tech R&D team.

Objectives defined, the next task was to choose a textbook for the course. I perused web sites, visited libraries, spoke with publisher representatives at conferences, and ordered new books. I discovered that the extant DSP texts fall into three, almost mutually exclusive, categories.

About 75% of the available texts target the EE student. These books assume familiarity with advanced calculus (including complex variables and ordinary differential equations), linear system theory, and perhaps even stochastic processes. The major part of such a text deals with semirigorous proofs of theorems, and the flavor and terminology of these texts would certainly completely alienate most of my target group. The CS student, for example, has a good basic understanding of derivatives and integrals, knows a little linear algebra and probably a bit of probability, but has little need for long, involved proofs, is singularly uninterested in poles in the complex plane, and is apt to view too many integral signs as just so many snakes, and flee in terror from them.

In addition, these *type-one* texts ignore those very aspects of the subject that most interest our target students, namely algorithm design, computational efficiency and special computational architectures, and advanced applications. The MAC instruction and Harvard architecture, arguably the defining features of digital signal processors, are generally not even mentioned in passing. Generally only the FFT, and perhaps the Levinson-Durbin recursion, are presented as algorithms, and even here the terminology is often alien to the CS student's ear, with no attention paid to their relation with other problems well known to the computer scientist. The exercises

generally involve extending proofs or dealing with simplistic signals that can be handled analytically; computer assignments are rare.

Finally, due perhaps to the depth of their coverage, the *type-one* texts tend to cover only the most basic theory, and no applications. In other words, these books finish before getting to the really interesting topics. Some cover the rudiments of speech processing, e.g. LPC and cepstral coefficients, but all consider speech compression and modem design beyond their scope. More advanced or specific texts are thus absolutely necessary before real-world applications can be tackled. These texts thus do not achieve our goal of preparing the student for participation in a real R&D team.

The next category, counting for about 20% of the texts, *do* target people who are more at home with the computer. *Type-two* texts tend to be ‘recipe books’, often accompanied by a diskette or CD. The newer trend is to replace the book with interactive instruction and experimentation software. These books usually contain between fifty and one hundred black box routines that can be called from a high-level language (e.g. C or MATLAB). The bulk of the text consists of instructions for calling these routines, with discussion of the underlying theory kept to a minimum.

While very useful for the computer professional who on odd occasions has need for some DSP procedures, these books do not instill a deep unified comprehension of the subject. Admittedly these books often explain algorithms in greater depth than *type-one* texts, but our target readers would benefit even more from a combination of *type-one* depth with *type-two* emphasis.

Of course there is nothing wrong with obtaining a well tested program or routine that fulfills the purpose at hand. Indeed it would not be prudent for the implementor to reinvent wheels in places where tire shops abound. However, due to their generality, library routines are often inefficient and may even be impractical for specific purposes. I wanted to enable my students to meet specific DSP needs by evaluating existing programs and library routines, or by writing original, tailored DSP code as required. The reader should also be able to port libraries to a new platform, understanding both the algorithm and the platform idiosyncrasies.

Finally, there are *type-three* texts, often written by DSP processor manufacturers. They emphasize the architecture, programming language, and programming tools of the manufacturer’s line of digital signal processors, and while they may explain some theory, they mostly assume prior knowledge or claim that such knowledge is not really required for the comprehension of the subject matter. The programming techniques developed, usually in lengthy detail, may be applicable to some extent to other manufacturers’

processors, but considerable adaptation would normally be required. *Type-three* texts tend to stress FIR and IIR filter structures, the radix 2 FFT algorithms, the LMS and perhaps Viterbi algorithms, and often describe various practical applications of these in great depth.

Due to their lack of mathematical sophistication, these books do not attempt to seriously treat DSP theory. Such critical topics as the sampling theorem, filtering, and adaptive systems are only trivially covered; true explanation of noise, filtering, and Fourier transforms are replaced by historical accounts, and algorithms are displayed in pseudocode *fait accompli* rather than derived. On the other hand, the manufacturers apparently feel that the typical reader will be lacking in CS background, and thus overly stress such obvious features as loops and numeric representation.

I thus reached the conclusion that none of the available DSP texts was truly suitable for the course, and was compelled to create my own course materials. These became the corner-stone of the present book. Often I found myself rethinking my own understanding of the subject matter, and frequently connections with other computer science subjects would only become clear during lecture preparation, or even during the lecture itself. I also found that the elimination of the conventional mathematical apparatus and rigorous proofs not only did not deplete the subject matter of meaning, but actually enriched it.

The topics included in this text may, at first, surprise the reader who is used to more conventional DSP texts. Subjects such as the matched filters, adaptive algorithms, the CORDIC algorithm, the Viterbi algorithm, speech compression, and modern modem theory are normally considered too complex and specialized for presentation at this level. I have found that these *advanced* topics are no more difficult for the newcomer to grasp than filter design or limit cycles, and perhaps more interesting and relevant. However, in order to keep the book size moderate, some of the more classical subjects had to be curtailed. These subjects are adequately covered in traditional texts, which may be consulted to supplement the present one.

Even so, the present book contains more material than can be actually taught in a single-semester course. A first course in DSP could cover most of the material in the early chapters, with the instructor then selecting algorithms and applications according to personal preference. The remaining subjects may be relegated to a more advanced course, or be assigned as self-study topics. My initial course went through the basic theory at break-neck speed, in order to rapidly get to speech compression and recognition. A second attempt emphasized modems and DSP for data communications.

Every section ends with a number of exercises that are designed to be entertaining and enriching. Some of these should not be difficult for the reader who understands the section, being designed to reinforce basic understanding of the material. Many are somewhat challenging, complementing the text, extending the theory, or presenting actual applications of the subject studied. Some are only loosely defined; for these one can give a quick answer, or develop them into a term project. Others introduce new material that will ease the understanding of the following sections, as well as widening the reader's DSP horizons.

I purposely avoid taking sides on the divisive issue of programming language and environment for algorithm design and test on general-purpose computers. Realizing that C, MATLAB, SPW, Mathematica and the like will all have their staunch supporters, and all have their strengths and weaknesses, I leave it to the student or instructor to select that language with which they are the most comfortable. Every seasoned programmer is most effective in his or her native language, and although some languages are obviously better DSP 'environments' than others, the difference can be minimized by the use of appropriate libraries.

Although the book was written to serve as a course textbook, it may be used by non-students as well. DSP practitioners are like master craftsmen; when they are called upon to construct some object they must exploit their box of tools. Novices have only a few such tools, and even these may not be sufficiently sharp. With time more tools are acquired, but almost all craftsmen tend to continue using those tools with which they have the most experience. The purpose of this book is to fill the toolbox with tools, and to help the DSP professional become more proficient in their proper use. Even people working in the field several years will probably find here new tools and new ways of using tools already acquired.

I would like to thank my students, who had to suffer through courses with no textbook and with continually changing syllabus, for their comments; my colleagues, particularly Yair Karelic, Mauro Caputi, and Tony Grgas, for their conscientious proofreading and insights; my wife Ethel for her encouragement (even allowing me untold late-night sessions banging away at the keyboard, although she had long ago banished all computers from the house); and our two girls, Hanna and Noga, who (now that this book is complete) will have their father back.

Jonathan (Y) Stein  
Jerusalem, Israel  
31 December 1999

# Index

page numbers of references to:

- definitions are underlined
- exercises are in *italics*

A-law, 213, *434*, 732, 757–759

A/D, 19, 64–70

ABS, 755, 773

AC, 21

accumulator, 591, 621

adaptive codebooks, 773–774

ADPCM, 762–765

ADSL, 124

adventure, 388

AGC, 4

Aiken, H, 625

algebraic codebooks, 772

aliasing, 54, 62–63, 99, *126*, 134,  
311, 337, 500, 585, 587

all-pass filter, 214, 272, *601*

all-pole filter, 295

all-zero filter, 219, 295

alternating current, *see* AC

alternation theorem, 812–814

AM, 158, 651, 653

AMDF, 751–752, 770

AMI, 700

amplification, 8, *12*, 34, 209–210,  
289–290

amplifier, 209–212

amplitude modulation, *see* AM

analog communications, 652–664

analog signal processing, 7–13

analog to digital conversion, *see* A/D

analog to digital converter, *see* A/D

analysis by synthesis, *see* ABS

antialiasing filter, 62–63

aperture time, 66, 67

arctangent, 609

ARMA, 245, 293–298, 475, 521

associative memory, 447

Asymmetric Digital Subscriber Line,  
*see* ADSL

attractor, 184–190

autocorrelation, 125, 263, 354, 751

Automatic Gain Control, *see* AGC

axon, 443

band-pass filter, 272

band-pass sampling theorem, 55,  
140, *769*

band-stop filter, 272

bandwidth, 17, 46

bandwidth expansion, 772, 773

bank of filters, 498–502

Bark scale, 434, 748

Barker code, 364

Barkhausen, HG, 434

baud rate, 703, 710

Baudot code, 650–653

Baudot, E, 650, 703

beamforming, 6, 315–319, 408, 425

beats, *33*, *76*, 157, 277–278

Bell, AG, 433, 650, 779

- BER, 726
- Berkeley, G, 431
- Bessel functions, 806
- Bessel, FW, 806
- bilinear mapping, 311
- biomedical signals, 5, *399, 420*
- bispectrum, 389
- bit, 783
- bit error rate, *see* BER
- bit reversal, 542–546
- block code, 682–690
- Box-Muller algorithm, 178
- boxcar signal, 63, 66
- breadth-first search, 695
- buffer delay, 276
- Burg, J, 522
- butterfly, 181, 540–542, 548–550, *626*
- Butterworth filter, 304–305, 307–308
  
- Cajal, SR, 443
- canonical form section, 595, 598–599
- carrier, *652*, 700
- carrier frequency, 156, 333, 653, 705, 710
- cascade structure, 329, 580–582
- causality, 224, 230, 570
- CELP, 771–775
- center clipper, 324
- cepstrum, 744–747, 752
- channel capacity theorem, 676
- chaos, 162–163, 180–191, 592
- Chebyshev filter, 306–308
- Chebyshev polynomials, 797, 804–806, 813–814
- checkbit, 683
- checkbyte, 683, 688
  
- Cholesky decomposition, 375, 424, 824
- circular buffer, 137–138, 491, *535*, 575, 636, *638*
- circular convolution, *138*, 139, 573–579
- circular functions, *see* trigonometric functions
- circular reference, *see* infinite loop
- clipping, 210–211, 324, *343*
- closed loop gain, 290
- clustering algorithms, 766
- code excited linear prediction, *see* CELP
- codewords, *681*
- coherent demodulators, 705
- comb graph, *28*
- communicating sequential processes, 493
- communications theory, 670–674
- complex cepstrum, 745
- constellation, *706*, 716–718, 720–723
- constraint length, 692
- context switch, 488, 631
- continuous phase FSK, *704*
- convolution, 95, 115, *219*, 227, 237–240
- convolutional code, 682, 690–698
- Cooley, J, 568
- CORDIC algorithm, 613–617
- correlation, 36, 349–364, 395
- correlator, *see* matched filter
- cosets, 685
- cost function, 408
- counting converter, 67
- CPFSK, *see* continuous phase FSK
- CRC code, *689*
- critical bands, 434
- crosscorrelation, 263, 352, *354*

- cumulants, 387–391
- cycle, 187–190, 592
- cyclic code, 685
- cyclic convolution, *see* circular convolution
- cyclic prefix, 578, 737
- cyclic suffix, 578
  
- D/A, 19, 69
- data communications, 4–5, 664–737
- data over voice, 775
- data overrun, 533
- dB, 31, 794
- DC, 20, 26, 698–700
- DC blocker, 8, 12, 301
- DDE, 407
- de-emphasis, 273, 661
- decibel, *see* dB
- decimation in time, *see* DIT, *see* DIF
- decision directed equalization, *see* DDE
- decision feedback equalizer, *see* DFE
- deconvolution, 267
- degree (of graph), 463, 465
- delay, 570, 571
  - algorithmic, 578
  - buffer, 578
  - system, 575
- delay line, 37, 579
- $\delta(t)$ , 23, 29
- $\delta_{n,m}$ , 22
- delta modulation, 761
- delta-PCM, 68–69
- demodulation, 159, 652
- demultiplex, 673
- dendrite, 443
- density, 802–803
- denumerably infinite, 783
- depth-first search, 695
  
- detection, 350–351, 359–361
- deterministic, 30, 162, 180
- DFE, 715
- DFT, 132–143
- DIF FFT, 547–548
- difference equations, 245–248, 515, 594, 606
- differentiation filter, 282–285, 662
- digital frequency, 55
- digital oscillator, 249–250, 606–608
- digital to analog conversion, *see* D/A
- digitizing, *see* A/D
- digraph, 465
- Dirac's delta function, 23, 29, 808–809
- Dirac, PAM, 23
- direct current, *see* DC
- direct form section, 595, 598
- direct form structure, 579
- directed graph, *see* digraph
- directed search, 694
- direction of arrival, *see* DOA
- Dirichlet's convergence conditions, 79, 87
- discrete Fourier transform, *see* DFT
- discrete multitone, *see* DMT
- DIT FFT, 539–546
- DMT, 735–737
- DOA, 316
- Doppler shift, 99, 365
- dot graphs, 28
- dot product, 789
- double buffering, 533
- double sideband AM, 659
- double-square distribution, 130
- DPCM, 760–763
- DSP processors, 619–638
- DTMF, 4, 6, 123, 490, 561–562, 664, 756
- DTW, 343

- dual differentiator, 662  
 Dudley, H, 779  
 dynamic programming, 343, 693, 696  
 dynamic range, 16, 57, 61, 763  
 dynamic time warping, *see* DTW  
 ECC, *see* error correcting codes  
 echo cancellation, 393, 400–404, 425  
 echo suppressor, 400–401  
 Edison, TA, 28  
 effective bits, 67  
 eigensignal, 38, 228, 254  
 elliptical filter, 307–308  
 elliptical functions, 307, 799, 806–807  
 empirical autocorrelation, 354  
 energy, 17, 18, 36, 41  
 ENIAC, 625  
 envelope detector, 656  
 equalizer, 222–223, 269, 393, 404–407, 425  
 ergodic, 196–197  
 error accumulation, 606  
 error correcting codes, 672, 680–698, 737  
 Euler, L, 77, 464  
 Eulerian cycle, 465  
 expectation, 815  
 exponential function, 794  
 exponential signal, 26–28  
  
 false alarm, 350, 366  
 fast Fourier transform, *see* FFT  
 fax, 4, 124, 329, 400, 401, 450, 490, 561, 647, 729, 730  
 FDM, 334  
 Fechner, GT, 431–433  
 feedback, 289–293, 474  
 FEQ, 269, 736  
  
 FEXT, 735  
 FFT, 135, 531–567  
 Fibonacci sequence, 145–147  
 field, 788–789  
 FIFO FFT, 565–567  
 filter, 214, 224, 226–228, 271–315  
 filter design, 235, 303–315  
 finite difference, 37–38, 235, 244, 246, 248, 282, 301, 497  
 FIR, 256  
 FIR filter, 219, 228–237  
 first-order section, 296–297  
 fixed point, 183–186, 590–595, 619, 628, 633–635, 784  
 flash converter, 65  
 FLL, 339–340  
 floating point, 619, 784  
 flow graph, 466–476  
 FM, 158, 651, 654, 659–664  
 FM chirp, 363–364  
 formant, 125, 437  
 four-quadrant arctangent, 91, 159, 609, 786  
 Fourier Integral Theorem, 106, 113  
 Fourier series, 77–101  
 Fourier transform, 103–117  
 Fourier, JBJ, 77–80  
 frequency domain, 106  
 frequency domain multiplexing, *see* FDM  
 frequency equalizer, *see* FEQ  
 frequency modulation, *see* FM  
 frequency response, 217, 228, 254–255  
 frequency shift keying, *see* FSK  
 FS, *see* Fourier series  
 FSK, 7, 653, 702–703  
 FT, *see* Fourier transform  
 FT pair, 106, 258, 358  
 fundamental theorem

- of algebra, [792](#)
- of arithmetic, [783](#)
- of calculus, [802](#)
- Gallager, RG, 678
- Gauss, 795
- Gaussian, 794–795, 816
- Geigel algorithm, 403, [404](#)
- generating function, 144–147
- generator matrix, 684
- Gibbs phenomenon, 86–90
- Gibbs, JW, 87
- Goertzel algorithm, 532, 561–565
- Golay code, [688](#)
- golden ratio, 147, 435
- Golgi, C, 443
- gradient descent, 398, 411–412
- graph, [463](#)
- graph theory, 462–467
- Gray code, [704](#)
- Green's function, 255
- group, 788
- group delay, [277](#)
- Hadamard matrix, 554
- Hamilton, Sir WR, 465, 782
- Hamiltonian cycle, 465
- Hamming codes, 683–686
- Hamming distance, 681
- Hamming, RW, 509, 681, 683
- hard limiter, 8, 211, 223, 330, 341, 634
- harmonic generation, 322
- harmonics, [74](#)
- Hartley transform, [526](#)
- Harvard architecture, 625
- Heaviside's step function, 21
- Hebb's principle, 451
- Hebb, DO, 451
- Hertz, H, 651
- Hessian matrix, 412
- heterodyning, *see* mixer
- hierarchical flow graph, 480
- high-pass filter, [272](#)
- higher-order signal processing, 194, 386–391
- Hilbert transform, 158–159, 287–288, 657
- HMM, [348](#)
- Hoare, CAR, 493
- Horner's rule, 562, 606, 791
- HPNA, 124, [308](#)
- hyperbolic functions, 798–799
- I-Q plot, [706](#), 716, 752
- ideal filter, 272–273
- IIR, [257](#)
- IIR filter, 220
- image processing, 18
- implementation, [471](#)
- impulse, *see* unit impulse
- impulse response, 224, [227](#), 255–258
- incoherent demodulators, 705
- infinite accumulator, 244
- infinite loop, *see* circular reference
- information theory, 666–670, 737
- inner product, 789
- instantaneous representation, 155–160, 495
- integration filter, 285–287
- intermodulation product, 323
- interrupt, 631
- intersymbol interference, *see* ISI
- inverse filter, 399
- inverse system, 221–223, 246
- inverse zT, 154
- inversion, 12, 34
- ISI, 406–407

- JND, 428–430  
 jokes, 161, 203, 240, 299, 348, 362,  
     432, 441, 478, 482, 506, 619,  
     640, 644  
 just noticeable difference, *see* JND
- Kalman filter, 368  
 Königsberg, 465  
 Kronecker delta, 22, 801  
 Kronecker, L, 781  
 kurtosis, 817
- Laplace transform, 148  
 lattice structure, 582–583  
 law of large numbers, 178, 179, 815  
 LBG algorithm, 767–768  
 Legendre polynomials, 804, 812  
 Levenshtein distance, 344–347  
 Levinson-Durbin recursion, 261, 376–  
     382, 521, 827  
 LFSR, 176–177, 179, 364, 700  
 limit cycle, 592  
 line spectral pairs, *see* LSP  
 linear algebra, 819–821  
 linear feedback shift register, *see*  
     LFSR  
 linear phase, 95, 276  
 linear system, 223  
 linear-phase, 215, 308, 584  
 LMS algorithm, 413–425, 451  
 logarithm, 610–611, 793–794  
 logarithmic companding, 213  
 logistic sigmoid, 330  
 logistics signal, 37, 39  
 long-term predictor, 743  
 Lorenz, E, 181  
 loss function, *see* cost function  
 low-pass filter, 272  
 LPC, 371–382, 740–744  
 LPC cepstral coefficients, 746–747
- LSB, 539, 545–546, 783  
 LSP, 383–386, 772  
 LTDFFT, 134
- MA, *see* moving average  
 MAC, 454, 473, 570, 579, 591, 620–  
     623  
 Markov model, 347  
 Markov signals, 194–195  
 masking, 441  
 matched filter, 5, 349, 361–365, 390,  
     701, 702  
 matrix, 821–828  
 Maxwell's equations, 650–651  
 Maxwell, JC, 650  
 mean opinion score, 755  
 mean square error, *see* MSE  
 median filter, 326–329  
 mel scale, 434  
 MELP, 744, 752, 777  
 memory location, 470  
 metric, 819  
 metric space, 790  
 Michelson, AA, 86  
 minimax approximation, 812–814  
 minimum mean square error, *see*  
     MMSE  
 minimum phase, 302  
 mixed-radix FFT, 551  
 mixer, 114, 137, 332–339, 560  
 MLP, 330, 451  
 MMSE, 409–410  
 modem, 4, 7, 124, 207, 208, 222,  
     269, 325, 334–335, 371, 400–  
     402, 404–406, 425, 488, 561,  
     619, 647, 698–737, 753, 775  
 modulation, 49, 157, 652  
 modulo, 783  
 Moler-Morrison algorithm, 612  
 moment functions, 194

- moments, 816–817
- Monte-Carlo integration, 203
- Morse code, 649–653, 710
- Morse, S, 649
- MOS, *see* mean opinion score
- moving average, 219
- MSB, 539, 545–546, 783
- MSE, 370, 373–375, 376, 378, 381, 382, 409–410, 414, 417, 421, 766
- $\mu$ -law, 213, 434, 732, 757–759
- multilayer perceptron, *see* MLP
- multiplex, 673, 711
- multiply-and-accumulate, *see* MAC
- music, 4, 26, 32, 39, 128, 204, 209, 215, 221, 222, 376, 433, 610, 673, 680, 742, 753
  
- Napoleon, 78, 649
- negative feedback, 290
- negative frequency, 46
- netlist, 485
- neural network, 329–332, 446–452, 750
- Newton, I, 72–73
- Newton-Raphson algorithm, 611
- NEXT, 735
- NLP, 325, 402–403
- node, 463
- noise, 31, 161–202, 700–702
- noise bandwidth, 508
- noise cancellation, 5, 393–399, 407, 408, 425
- nondenumerably infinite, 784
- nondeterministic algorithm, 203–204
- nonlinear processor, *see* NLP
- nonlinear systems, 210–212, 223, 226, 322–324, 335, 390, 403
- NOP, 629, 630, 638, 639
  
- norm, 819
- normal equations, 84, 280, 811, 817, 820
- notch filter, 250, 272, 301, 302, 339
- NRT, 701, 743
- NRZ, 667, 698–700, 708
- number, 781
  - complex, 782, 785–787
  - integer, 782–784
  - rational, 782
  - real, 782, 784–785
  - whole, 781
- Nyquist, H, 53
  
- OFDM, 736
- OOK, 651, 652, 700–702
- orthogonal, 36, 789, 820
- orthogonal frequency division multiplexing, *see* OFDM
- oscillator, 290
- OSI model, 665
- outliers, 326
- overflow, 555–558
- overlap add, 575
- overlap save, 575
  
- Paley-Wiener theorem, 275
- PAM, 49, 654, 698, 703
- PAR, 61, 737
- parallel form, 599–601
- parallel port, 632
- parity check matrix, 686
- Parseval's relation, 95, 115, 138, 358, 556
- partition, 388
- patch panel, 481
- pattern recognition, 208, 497
- PCM, 48, 49
- PCM modem, 7, 733–734
- perceptron, 330, 451

- perceptual weighting, 773, 775  
 periodogram, 502–506  
 phase modulation, *see* PM  
 phase shift keying, *see* PSK  
 phase-locked loop, *see* PLL  
 phased array, 317  
 phoneme, [436](#)  
 pipeline, 627–630  
 Pisarenko harmonic decomposition, 512–519  
 pitch, 125, 436, 750–753  
 PLL, 338–343, 712  
 PM, 654, 659–664  
 pole-zero plot, [295](#), 298–302  
 polynomial, 231, 279–285, 791–793  
 polynomial approximation, 809–814  
 polyphase filter, 584–590  
 polyspectrum, 389  
 post-filter, 761, 774  
 POTS, [124](#)  
 power, [18](#)  
 power cepstrum, 746  
 power law distortion, 211  
 power spectral density, *see* PSD  
 power spectrum, 91, [94](#), 95, [96](#), 116, 122–126  
 PPM, 49, 654  
 pre-emphasis, 273, 661  
 prediction, [49](#), 180–181, 252–253, 369–376, 760–765  
 PRF, PRI, 98  
 prime factor FFT algorithm, 551  
 probabilistic algorithm, 203–204  
 probability, 815–819  
 processing delay, 274  
 processing gain, 360  
 Prony's method, 518–519, 602  
 Prony, Baron de, 518  
 PSD, [123](#), 357–359, 495, 708–710  
 pseudorandom signals, 162, 174–179  
 PSK, 364, 654, 702, 704–708  
 PSTN, 775  
 pulse amplitude modulation, *see* PAM  
 pulse coded modulation, *see* PCM  
 pulse position modulation, *see* PPM  
 pulse width modulation, *see* PWM  
 PWM, 49, 654  
 pyramid algorithm, 528, 769  
 Pythagorean addition, 611–612  
  
 QAM, 716–723  
 QMF, 528, 769  
 quadrature component, 27  
 quadrature form, 91  
 Quadrature Mirror Filter, *see* QMF  
 quefrency, 744, 745  
  
 radar, 5–6, 98–100, 129, 161, 171–174, 207, 208, 271, 316, [319](#), 350–352, [353](#), 362–365, 367  
 radio frequency interference, *see* RFI  
 radix-4 FFT, 548–550  
 radix- $R$  FFT, 542, 550  
 raised cosine, 233  
 random number generators, 174–179  
 RASTA, 748  
 RAX, 481–493  
 real-time, 533, 570  
 rectifier, 9, 86, 211  
 recursive least squares, *see* RLS  
 Reed-Solomon codes, 688–689  
 Remez exchange algorithm, 314, 813–814  
 resampling, [370](#)  
 return map, 185  
 RFI, 324  
 RLS, 421–425

- RMS, 17, 19  
 ROC, 150–155  
 root mean squared, *see* RMS  
 roots of unity, 532, 787  
 rotation matrix, 613–615, 823–824  
  
 sample and hold, 66  
 sampling, *see* A/D  
 sampling theorem, 48, 53–56, 139  
 Šarkovskii's theorem, 190  
 saturation arithmetic, 592, 634  
 sawtooth, 24, 29, 67, 75  
 SBC, 768–769  
 Schuster, Sir A, 502, 530  
 Schwartz inequality, 122  
 scrambler, 700, 712  
 seismology, 6  
 sequency, 524  
 serial port, 632  
 set partitioning, 724  
 Shakespeare, W, 17, 674  
 Shannon C, 683  
 Shannon, C, 53, 671–680  
 shell mapper, 731  
 shift register, 176, 632, 682, 690  
 shifted unit impulse, *see* SUI  
 short-term predictor, 743  
 side information, 764  
 sidebands, 658, 662–663  
 sigma-delta digitizer, 68–69  
 signal, 16  
     chaotic, 162  
     characteristics, 30–33  
     complex, 18, 26, 30  
     constant, 20  
     deterministic, 30  
     exponential, 26  
     finite bandwidth, 33  
     finite time duration, 33, 571  
     finiteness, 16  
     impulse, 21  
     incompletely known, 162  
     logistics, 37  
     periodic, 32  
     pseudorandom, 162  
     sawtooth, 29  
     simplest, 20–28  
     sinusoid, 25  
     square wave, 23  
     stochastic, 30, 162  
     triangle, 29  
     unit step, 21  
 signal processing system, 208  
 signal separation, 407  
 signal to noise ratio, *see* SNR  
 simple difference, 470  
 Sinc, 88–89  
 sinc function, 45, 53, 126  
 single sideband AM, 659  
 sinusoid, 25–26, 29, 32, 33  
 sinusoidal representation, 749, 778  
 skew, 817  
 slicer, 325, 707, 715, 718, 720–723  
 slint graph, 28  
 slope converter, 67  
 smoothing, 231, 279–282  
 SNR, 31, 48, 57, 60, 61, 162, 173,  
     360, 365–367  
 soma, 443  
 sonogram, 128, 505  
 sort, 536, 538  
 speaker identification, 4, 71, 128,  
     739, 744, 747, 755, 770  
 spectral analysis, 5  
 spectrum, 45–46, 71–73, 74  
 speech compression, 4, 369, 371,  
     373, 633, 739, 753–778  
 speech recognition, 4, 343, 350, 441,  
     739, 744, 747–749, 768  
 speech signals, 4, 435–442, 739–778

- speech synthesis, 739  
 split-radix FFT, 550  
 square root, 611–613  
 square wave, 23, 76  
 stable system, 223, 244  
 state-space description, 214, 247,  
     481, 692–695  
 static buffer, 570  
 stationary signal, 193  
 statistically independent, 818  
 statistics, 161  
 step signal, *see* unit step  
 STFT, 126–132  
 stochastic signal, 30, 161–163, 192–  
     198  
 streamability, 224  
 structures, 579–584, 595–601  
 subband coding, *see* SBC  
 sufficient statistics, 164  
 SUI, 22, 41–43  
 sum formulas, 796  
 super-resolution, 512, 514  
 SVD, 812, 824  
 symbol, 405, 703  
 symmetries, 478–481  
 synapse, 444  
 synchronization, 339  
 syndrome, 686  
 system identification, 252–270  
 Szego polynomials, 812  
  
 Taylor expansion, 45–47, 145, 155,  
     279, 308, 323, 391, 606, 609,  
     811  
 TCM, 723–728  
 tee connector, 468  
 Tesla, N, 28  
 time advance operator, 36–37  
 time delay operator, 36–38  
 time domain, 106  
  
 time of arrival, *see* TOA  
 time reversal, 34–35, 95, 116, 137,  
     154, 361  
 time shift, 94–95, 114, 137, 153–  
     154, 255, 297, 352–353  
 time warping, 343–348, 739  
 time-frequency distribution, 129–  
     131, 529  
 time-invariant system, 223–224  
 timing recovery, 710–714  
 TLA, 642  
 TOA, 171, 319, 364  
 Toeplitz matrix, 240, 260, 261, 263,  
     375, 516, 519, 822, 827  
 Toeplitz, O, 270  
 toll quality, 756  
 Tomlinson equalization, 407, 715  
 Toom-Cook algorithm, 538  
 topology, 463  
 transceiver, 647  
 transfer function, 267, 293–298  
 transform, 105  
 transient, 511  
 transposed structure, 579–580, 595,  
     599  
 transposition theorem, 476, 599  
 trellis, 467, 693  
 triangle wave, 24  
 trigonometric functions, 795–798  
 Tukey, JW, 200, 568  
 twiddle factor, 541  
 two's complement, 633, 783  
  
 uncertainty theorem, 73, 89, 98,  
     118, 120, 121–122, 129, 156,  
     313, 503, 507, 514, 527, 560,  
     703, 704  
 unit delay, 470  
 unit impulse, 21–23  
 unit step, 21

- unitary matrix, *143*, 823
- V.34, 47, 718, 730–732
- VAD, 4, 743
- variance, 161, 816
- VCO, 339
- vector quantization, *see* VQ
- vector space, 40–44, 786, 789–790, 819–821
- Viterbi algorithm, 344, 693–698
- vocoder, 755
- voice activity detection, *see* VAD
- voice over data, 775
- voicing, 752–753
- voltage controlled oscillator, *see* VCO
- von Hann, J, 508
- von Kempelen, W, 779
- von Neumann, J, 177, 625
- Voronoy region, 721, 768
- VOX, *see* VAD
- VQ, 765–771
  
- Walsh functions, 523–526
- Watson-Watt, R, 171
- wave, 315
- waveform coders, 754
- waveform interpolation, 777–778
- wavelength, 315
- wavelets, 526–529
- Weber, E, 428–430
- white noise, 169, 189, 195
- whitening filter, 362, 367, 372
- Widrow-Hoff equation, 413, 415
- Wiener filter, 365–369
- Wiener, N, 270, 392, 530
- Wiener-Hopf equations, 263, 410–413
- Wiener-Khintchine theorem, 125, 201, 357–359, 386, 389, 521, 708
  
- Wigner-Ville distribution, 130
- window, 117, *126*, 129, 274, 505–512, 526
- Winograd FFT, 551
- Wold's decomposition theorem, 196
  
- Yule, GU, 270, 530
- Yule-Walker equations, 263–264, 374, 375, 378–380, 521
  
- z transform, 143–155, 265–270, 520
- zero crossings, 324–325, 496–497, 710–712, 743, 752
- zero-overhead, 621, *623*, 631, 638
- zero-padding, 550, 559, 574
- zoom FFT, 559–560