

Xuhua Xia

# Bioinformatics and the Cell

Modern Computational Approaches  
in Genomics, Proteomics and  
Transcriptomics

*Second Edition*



# Bioinformatics and the Cell

Xuhua Xia

# Bioinformatics and the Cell

Modern Computational Approaches in  
Genomics, Proteomics and Transcriptomics

Second Edition



Springer

Xuhua Xia  
University of Ottawa CAREG and Biology Department  
Ottawa, ON, Canada

ISBN 978-3-319-90682-9                    ISBN 978-3-319-90684-3 (eBook)  
<https://doi.org/10.1007/978-3-319-90684-3>

Library of Congress Control Number: 2018941096

© Springer Science+Business Media LLC 2007, 2018

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors, and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, express or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Printed on acid-free paper

This Springer imprint is published by the registered company Springer International Publishing AG part of Springer Nature.

The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

# Preface to the Second Edition

A Christian author once remarked that a book is very much like a religion. A religion will slowly die if there is no revival movement, and a book will slowly die if there is no revision and new edition. I also heard a Christian joke about the revival movement. One pastor said that their revival movement gained them four new families, another declared to have gained six, and a third confessed to have just got rid of ten biggest troublemakers.

The second edition of this book did get rid of some chapters and gain some new ones. Those deleted are obsolete, and those new ones, I hope, are not troublemakers but represent new blood that will extend the life of this book and spread its message much beyond what has been achieved in the first edition. The new edition that you are holding in your hands is far better than the first edition, and colleagues who had read the book agreed with me.

The first edition of the book aimed to achieve two objectives. The first is to make bioinformatics as simple as possible, but not simpler. Rendering equations to numbers with a lot of numerical illustrations sets the book apart from many other books on bioinformatics. The second objective is to bridge bioinformatics to molecular biology and evolution by examples and applications. These two objectives remain the same in the second edition, but the coverage has expanded and improved by mending incomplete and sometimes incorrect presentations of some algorithms and their applications in the first edition.

My effort to revive this book gained a number of readers of various chapters in both editions. They were all troublemakers, avidly pointing out typos and errors scattered throughout the book. They include Christian Delamarche, Amir Hajianpour, Amanda Perozzo, Ran Prabhakaran, Lisha Tang, Antony Vincent, Xiguang Wang, and Yongxin Ye. To them I am very grateful.

Some materials in the book have been used in my preaching at Peking University in three consecutive summers from 2013 to 2015, and I thank the participants for discussion and Dr. Jian Lu and Dr. Jingchu Luo for organizing the summer courses. I have not heard of any new converts so far, and I begin to develop an uneasy feeling

that my patron, in the form of Discovery Grants from Canadian Natural Science and Engineering Research Council (NSERC), may not be sufficiently pleased with its money for my mission in research and teaching. I do feel obliged to take this opportunity to thank NSERC for its sponsorship.

My students have shaped what I write, especially those taking my Bioinformatics and Molecular Evolution courses who have offered me ample feedback on how to explain bioinformatic algorithms from different perspectives. Incidentally, my Molecular Evolution course was taught in a church sanctuary due to the lack of lecture halls at the University of Ottawa. This arrangement surprised me at first, and I did not recognize it as a good omen. Students appeared to be extraordinarily attentive in a spiritual environment bestowed by the cross and the bell tower. The unique combination of church and evolution must have helped them to cross the disciplinary boundaries. For this reason I wish to thank La paroisse francophone du Sacré-Coeur for offering its sanctuary for teaching evolution, and I sincerely hope that more lectures on evolution will take place in religious sanctuaries. As a friendly gesture of reciprocity, I have intentionally started this Preface with “A Christian author. . . .”

All authors, Christian or not, are supposed to thank their publisher for taking the risk of publishing their books, but I do have good reasons to thank Noreen Henson of Springer for her patience and encouragement during the preparation of this second edition. I signed a contract promising to deliver the finished product in June 2016 and have forgotten how many times Noreen has to reschedule the date of publication. Just as it is unwise for a prophet to predict the exact time for the return of the messiah, it is unwise for an author to predict the delivery of the finished manuscript.

Someone told me that it is no longer cool to acknowledge one’s family members in a book of science. However, for a person who lectures twice a week in a church, omitting his family in his writings is unthinkable. So here comes my wife, Zheng Xie, whose care of me I appreciate immensely. I should also mention that the joint revival effort between Zheng and me has gained us five children (Kim, Jeff, Jeremy, Jadon, and Mary-Jane), and they are not troublemakers.

Ottawa, ON, Canada

Xuhua Xia

# Contents

<b>1</b>	<b>String Mathematics, BLAST, and FASTA . . . . .</b>	<b>1</b>
1	Introduction . . . . .	1
2	Mathematics of String Matching . . . . .	4
2.1	Basic Concepts . . . . .	4
2.2	BLAST Parameters $\lambda$ , $K$ , and $H$ . . . . .	10
3	String-Matching Algorithms in FASTA and BLAST . . . . .	15
3.1	FASTA Algorithm . . . . .	16
3.2	BLAST Algorithm . . . . .	20
4	BLAST and FASTA in Genomics, Transcriptomics, and Proteomics . . . . .	23
4.1	Genomics . . . . .	23
4.2	Transcriptomics . . . . .	25
4.3	Proteomics . . . . .	26
5	BLAST and FASTA in Biomedical Research and Drug Discovery . . . . .	26
5.1	Human Genetic Disorders . . . . .	26
5.2	BLAST and FASTA Facilitate the Development of Drugs Against Human Pathogens . . . . .	27
	Appendix: Being Colorful Is Not Enough—How Are Stop Codons Decoded? . . . . .	29
<b>2</b>	<b>Sequence Alignment . . . . .</b>	<b>33</b>
1	Introduction . . . . .	33
2	Dynamic Programming for Pairwise Alignment . . . . .	35
2.1	Pairwise Alignment with Constant Gap Penalty . . . . .	36
2.2	Match-Mismatch Matrix . . . . .	40
2.3	Pairwise Alignment with Gap Penalty Specified by the Affine Function . . . . .	55

<b>3</b>	<b>Multiple Sequence Alignment (MSA) . . . . .</b>	<b>62</b>
3.1	Dynamic Programming for Profile Alignment . . . . .	63
3.2	Replacing a Sequence Profile by a Reconstructed Ancestral Sequence . . . . .	66
3.3	Multiple Alignment with a Guide Tree . . . . .	68
3.4	The Inconsistency Problem with Pairwise Alignments . . . . .	70
<b>4</b>	<b>Sequence Alignment with Secondary Structure . . . . .</b>	<b>71</b>
<b>5</b>	<b>Align Nucleotide Sequences Against Amino Acid Sequences . . . . .</b>	<b>72</b>
	Postscript . . . . .	75
<b>3</b>	<b>Position weight matrix and Perceptron . . . . .</b>	<b>77</b>
1	Introduction . . . . .	77
2	Position Weight Matrix (PWM) . . . . .	79
2.1	Basic Concepts of PWM . . . . .	79
2.2	Specification of the Background Frequencies . . . . .	82
2.3	Specification of Pseudocounts . . . . .	83
2.4	Statistical Significance Tests for PWM . . . . .	84
2.5	Using PWM to Refine Multiple Sequence Alignment . . . . .	90
3	Perceptron . . . . .	93
3.1	Perceptron Algorithm . . . . .	93
3.2	Perceptron and XOR Problem . . . . .	97
	Postscript . . . . .	98
<b>4</b>	<b>Gibbs sampler . . . . .</b>	<b>99</b>
1	Introduction . . . . .	99
1.1	Gibbs Sampler and Its Biological Applications . . . . .	99
1.2	What Is De Novo Motif Discovery? . . . . .	100
1.3	What Are the Estimated Parameters When Applying Gibbs Sampler in Motif Discovery? . . . . .	101
2	Computational Details of Gibbs Sampler . . . . .	102
2.1	Initialization . . . . .	103
2.2	Predictive Update . . . . .	104
2.3	Motif Sampler . . . . .	110
	Postscript . . . . .	111
<b>5</b>	<b>Transcriptomics and RNA-Seq Data Analysis . . . . .</b>	<b>113</b>
1	Introduction . . . . .	113
2	Reduce File Size Without Losing Sequence Information . . . . .	115
3	Assigning Sequence Reads to Paralogous Genes . . . . .	118
3.1	Allocating Reads to a Two-Member Paralogous Gene Family . . . . .	118
3.2	Allocating Sequence Reads in Gene Family with More Than Two Members . . . . .	119

4	Comparison Between ARSDA and Cufflinks in Characterizing Gene Expression . . . . .	121
5	Use Transcriptomic Data to Refine Genome Annotation . . . . .	124
6	Other Applications Using RNA-Seq Data . . . . .	126
	Postscript . . . . .	127
6	<b>Self-Organizing Map and Other Clustering Methods in Transcriptomics . . . . .</b>	129
1	Introduction . . . . .	129
2	Similarity and Distance Indices . . . . .	131
3	Clustering Algorithms . . . . .	133
3.1	UPGMA: A Hierarchical Clustering Algorithm . . . . .	133
3.2	Self-Organizing Map (SOM): A Nonhierarchical Cluster Algorithm . . . . .	137
	Postscript . . . . .	144
7	<b>Hidden Markov Models and Protein Secondary Structure Prediction . . . . .</b>	145
1	Introduction . . . . .	145
2	Markov Models . . . . .	146
3	Hidden Markov Models . . . . .	151
3.1	The Essential Elements in a Hidden Markov Model . . . . .	151
3.2	Training HMM for Predicting Protein Secondary Structure . . . . .	153
3.3	The Viterbi Algorithm . . . . .	157
3.4	Forward Algorithm . . . . .	164
3.5	HMM and Gene Prediction . . . . .	169
	Postscript . . . . .	171
8	<b>Bioinformatics and Translation Initiation . . . . .</b>	173
1	Introduction . . . . .	173
2	Translation Initiation in Bacteria . . . . .	175
2.1	Linear Structure of a Bacterial Gene and Its mRNA . . . . .	175
2.2	A Model of SD/aSD Interaction and Other Factors Contributing to Translation Initiation . . . . .	176
2.3	$D_{\text{toStart}}$ Is Constrained in a Narrow Range . . . . .	179
2.4	Genes with High $D_{\text{toStart}}$ Tend to Have Better Codon Adaptation . . . . .	180
2.5	Secondary Structure and Translation Initiation Efficiency . . . . .	181
2.6	Translation Initiation and Phage Host Specificity . . . . .	183
3	Translation Initiation in Eukaryotes . . . . .	184
3.1	Models of Eukaryotic Translation Initiation . . . . .	184
3.2	Effect of Poly(A) in 5'UTR on Translation Initiation in <i>Saccharomyces cerevisiae</i> . . . . .	186

3.3	Internal Ribosome Entry Site Activity and Stability of Secondary Structure . . . . .	190
3.4	Two Alternative Hypotheses on Kozak Consensus . . . . .	193
	Postscript . . . . .	194
<b>9</b>	<b>Bioinformatics and Translation Elongation . . . . .</b>	<b>197</b>
1	Introduction . . . . .	197
1.1	Basic Notations, Definitions, and Abbreviations . . . . .	197
1.2	Elongation Efficiency Depends on Amino Acid and Codon Usage . . . . .	204
1.3	Empirical Illustration of Codon-Anticodon Adaptation . . . . .	204
1.4	Effect of Biased Mutation on Codon Usage and Some Misconceptions . . . . .	208
1.5	Two Hypotheses on Translation Elongation Efficiency . . . . .	210
1.6	Wobble Hypothesis and Its Extensions . . . . .	211
2	Commonly Used Codon Usage Indices . . . . .	214
2.1	RSCU (Relative Synonymous Codon Usage) . . . . .	216
2.2	CAI (Codon Adaptation Index) . . . . .	221
2.3	$I_{TE}$ (Index of Translation Elongation) . . . . .	227
3	Translation Elongation Efficiency and Accuracy . . . . .	232
4	Amino Acid Usage and Translation Elongation Efficiency . . . . .	234
4.1	Factors Related to Selection for Translation Elongation Efficiency . . . . .	234
4.2	Number of Synonymous Codons . . . . .	235
4.3	Genomic Mutation Bias . . . . .	236
	Postscript . . . . .	236
<b>10</b>	<b>Bioinformatics and Translation Termination in Bacteria . . . . .</b>	<b>239</b>
1	Introduction . . . . .	239
1.1	The Release Factors . . . . .	239
1.2	Stop Codons as Part of the Stop Signal . . . . .	244
2	Three Key Factors Affecting Stop Codon Usage . . . . .	245
2.1	Genomic Mutation Bias and Codon Usage . . . . .	246
2.2	Differential Abundance of Release Factors and Stop Codon Usage . . . . .	247
2.3	Near-Cognate tRNA and Stop Codon Usage . . . . .	249
2.4	Why UAG Is Rarely Used as Stop Codons? . . . . .	251
3	Quantification of Translation Termination Efficiency . . . . .	253
	Postscript . . . . .	254
<b>11</b>	<b>Genomic Features: Content Sensors, Nucleotide Skew Plot, Strand Asymmetry, and DNA Methylation . . . . .</b>	<b>255</b>
1	Introduction . . . . .	255
2	Genomic Strand Asymmetry and GC Skew . . . . .	256

<b>3</b>	<b>Content Sensors Related to DNA Methylation and Spontaneous Deamination . . . . .</b>	258
3.1	Indices of DNA Methylation . . . . .	258
3.2	Are These Indices Useful in Discriminating Between Coding and Noncoding Sequences? . . . . .	260
<b>4</b>	<b>Do <i>Mycoplasma</i> Genomes Challenge the Association Between DNA Methylation and CpG Deficiency? . . . . .</b>	261
<b>5</b>	<b>Epigenetics and New Perspectives on Human Diseases . . . . .</b>	263
5.1	DNA Methylation and Conventional View of Methylated Gene Silencing . . . . .	263
5.2	Epigenetics and New Perspective on Methylation-Mediated Gene Interaction . . . . .	264
5.3	Bioinformatics and Epigenetics . . . . .	265
	Postscript . . . . .	267
<b>12</b>	<b>Nucleotide Substitution Models and Evolutionary Distances . . . . .</b>	269
1	Introduction . . . . .	269
2	Three Methods to Obtain Transition Probabilities . . . . .	272
2.1	Probability Reasoning to Obtain Transition Probabilities and Evolutionary Distances . . . . .	272
2.2	Obtaining Transition Probabilities by Solving Differential Equations . . . . .	291
2.3	Obtain Transition Probabilities from the Rate Matrix by Using Matrix Exponential . . . . .	294
3	How Far Can We Trace Back the Evolutionary History? . . . . .	309
4	Selecting the Best-Fitting Substitution Model . . . . .	311
4.1	Likelihood Ratio Test for Alternative Substitution Models . . . . .	312
4.2	Information-Theoretic Indices for Substitution Model Selection . . . . .	314
	Postscript . . . . .	314
<b>13</b>	<b>Protein Substitution Model and Evolutionary Distance . . . . .</b>	315
1	Introduction . . . . .	315
2	Deriving $P$ and $Q$ Matrices for Computing Evolutionary Distances . . . . .	317
<b>14</b>	<b>Maximum Parsimony Method in Phylogenetics . . . . .</b>	327
1	Introduction . . . . .	327
2	The Sankoff Algorithm . . . . .	329
3	The Uphill Search and Branch-and-Bound Search Algorithms . . . . .	334
4	The Long-Branch Attraction Problem . . . . .	335
5	Bootstrapping and Delete-Half Jackknifing . . . . .	337

6	Statistical Tests of Alternative MP Topologies . . . . .	338
6.1	What Sites Are Relevant to the Test? . . . . .	338
6.2	What OTUs to Include? . . . . .	339
6.3	Include Sites as a Random Effect in a Mixed Model . . . . .	340
	Postscript . . . . .	341
15	<b>Distance-Based Phylogenetic Methods</b> . . . . .	343
1	Introduction . . . . .	343
2	Simultaneously Estimated Distances and Paralinear/LogDet Distances . . . . .	344
2.1	Simultaneously Estimated Distances . . . . .	344
2.2	Paralinear and LogDet Distances . . . . .	347
2.3	Other Distances Not from Aligned Sequences . . . . .	349
3	Distance-Based Phylogenetic Algorithms . . . . .	350
3.1	Least-Squares (LS) Methods and Minimum Evolution (ME) Criterion . . . . .	350
3.2	Neighbor-Joining (NJ) Method . . . . .	352
4	Dating Speciation and Gene Duplication Events . . . . .	356
4.1	The Least-Squares Method for Conventional Internal Node-Calibrated Dating . . . . .	356
4.2	Tip-Dating for Rapidly Evolving Viruses . . . . .	367
4.3	Obtaining Confidence Intervals by Using Bootstrapping or Jackknifing . . . . .	369
4.4	Application of the Dating Methods . . . . .	370
	Postscript . . . . .	379
16	<b>Maximum Likelihood in Molecular Phylogenetics</b> . . . . .	381
1	Introduction . . . . .	381
2	The Rationale of Maximum Likelihood Approach . . . . .	382
3	Likelihood for a Phylogenetic Tree . . . . .	385
3.1	The Brute-Force Approach . . . . .	385
3.2	The Pruning Algorithm . . . . .	387
4	Calculating Likelihood by Imposing a Molecular Clock . . . . .	389
5	Handling of Missing Data with the Pruning Algorithm (and the Potential Bias) . . . . .	392
	Postscript . . . . .	395
17	<b>Protein Isoelectric Point and <i>Helicobacter pylori</i></b> . . . . .	397
1	What Is Protein Isoelectric Point and Why It Is Important to Know Its Computation . . . . .	397
2	Computation of Protein Isoelectric Point . . . . .	399
3	Genomic <i>pI</i> Profiling . . . . .	403

<b>Contents</b>		xiii
<b>4</b>	<b>A Case Study with the Gastric Pathogen <i>Helicobacter pylori</i> . . . . .</b>	<b>404</b>
4.1	Two Hypothesized Mechanisms of Acid Resistance in <i>H. pylori</i> . . . . .	405
4.2	Where Do Positively Charged Proteins in <i>H. pylori</i> Come From? . . . . .	407
4.3	Is the Basic Proteome in <i>H. pylori</i> Really an Adaptation to the Acidic Environment? . . . . .	409
4.4	Discriminate Between AAH and PAH . . . . .	410
	Postscript . . . . .	411
<b>18</b>	<b>Bioinformatics and In Silico 2D Gel Electrophoresis . . . . .</b>	<b>413</b>
1	Scientific Rationale Behind the 2D-SDS-PAGE . . . . .	413
2	In Silico Gel with Expected Separation Pattern . . . . .	414
3	Differences Between the In Silico and the Actual 2D Gel . . . . .	418
<b>19</b>	<b>Fundamentals of Proteomics . . . . .</b>	<b>421</b>
1	Introduction . . . . .	421
2	Protein Mass Spectrometry . . . . .	422
3	Charge Deconvolution . . . . .	424
4	Peptide Mass Fingerprinting . . . . .	429
4.1	Peptide Digestion . . . . .	429
4.2	MS Determination of Peptide Mass . . . . .	432
4.3	Protein Database and In Silico Digestion . . . . .	432
4.4	Protein Identification . . . . .	434
	Postscript . . . . .	435
<b>Appendix</b>		<b>437</b>
	The Delta Method for Deriving Parameter Variance . . . . .	437
	Variance of Allele Frequency for a Recessive Allele . . . . .	438
	Variance of JC69 Distance . . . . .	439
	The Variance of K80 Distance . . . . .	439
	Illustration of Expectation-Maximization (EM) Algorithm . . . . .	440
<b>References</b>		<b>443</b>
<b>Index</b>		<b>485</b>

# Chapter 1

## String Mathematics, BLAST, and FASTA



### 1 Introduction

Why should we start a book on bioinformatics with BLAST (Altschul et al. 1990) and FASTA (Lipman and Pearson 1985; Pearson 1990; Pearson and Lipman 1988)? There surely were bioinformatic analyses before BLAST and FASTA. For example, both global sequence alignment (Needleman and Wunsch 1970) and local sequence alignment (Smith and Waterman 1981) by dynamic programming were developed before BLAST and FASTA. Sequence alignment and comparison contributed significantly to the discovery of Shine-Dalgarno sequences (Shine and Dalgarno 1974a, b, 1975; Steitz and Jakes 1975; Taniguchi and Weissmann 1978) and Kozak consensus (Kozak 1981, 1991, 1999) that facilitate the localization of translation start codon in prokaryotes and eukaryotes, respectively. The perceptron algorithm was used to characterize prokaryotic translation initiation signals in 1982 (Stormo et al. 1982a). The sequence similarity between an oncogene, *v-sis*, and the platelet-derived growth factor, PDGF, was discovered in 1983 (Doolittle et al. 1983; Waterfield et al. 1983). A codon bias index was developed in 1982 (Bennetzen and Hall 1982). Using codon information to derive biological insights occurred even earlier, including the deduction of the first genetic code in the 1960s. In fact, the successful inference of the first stop codon UAG from codon mutation data represents such a beautiful piece of work that I can't resist retelling the story at the end of this chapter.

There are, however, good reasons for starting a book on bioinformatics and the cell with BLAST and FASTA. The reason would speak itself if we briefly review the development of genomics, transcriptomics, and proteomics and position ourselves in proper historical context. BLAST and FASTA permeate every aspect of bioinformatics arising from genomics, transcriptomics, and proteomics.

Genomics is often considered to have started in 1986 when two significant events happened. First, the US Department of Energy (DOE) announced the Human Genome Initiative aiming to produce a reference human genome sequence. Second,

Leroy Hood developed the first automatic DNA sequencer, which paved the way for the official beginning of the Human Genome Project in 1990. The year 1995 witnessed the completion of the first three bacterial genome sequencing projects, with the *Haemophilus influenzae* Rd. KW20 genome in July (Fleischmann et al. 1995), the *Synechocystis* sp. PCC 6803 genome in August (Kaneko et al. 1995, 1996), and the *Mycoplasma genitalium* G-37 genome in October (Fraser et al. 1995). The first working draft of the entire human genome was completed in 2000, soon to be followed by the simultaneous publication of the human genome in *Nature* and *Science* in February, 2001 (Lander et al. 2001; Venter et al. 2001). A large number of sequences would be of little practical use if we do not have efficient searching algorithms embedded in BLAST and FASTA.

Transcriptomics started mainly with the development of gene arrays such as macroarrays (Chuang et al. 1993; Tao et al. 1999) and microarrays (Schena 1996, 2003) and serial analysis of gene expression (Madden et al. 1997; Saha et al. 2002; Velculescu et al. 1995, 1997; Zhang et al. 1997). Modern transcriptomic characterization is now done almost exclusively with RNA-Seq technology popularized by Mortazavi et al. (2008). RNA-Seq studies generate huge number of nucleotide sequences. Among the 6472 transcriptomic studies on humans available at NCBI/DDBJ/EBI by Apr. 14, 2016, 196 studies each contribute more than 1 terabyte (TB) of nucleotide bases, with the top one contributing 86.4 TB. Without fast-searching algorithms in BLAST, FASTA, and the like that map these huge number of sequences to genomes, there is no way to transform these data into gene-specific expression values.

The terms “proteome” and “proteomics” were coined by Marc Wilkins and colleagues in 1994 (Ezzell 2002), and large-scale proteomic research started with sodium dodecyl sulfate polyacrylamide gel electrophoresis, SDS-PAGE (Laemmli 1970). Subsequent perfection of isoelectric focusing leads to the most frequently used protein separation method, the 2D-SDS-PAGE. Large-scale peptide analysis methods have been developed by John Yates and colleagues (Washburn et al. 2001; Yates 2004a, b). Mass spectrometry used in combination with affinity purification and/or chemical cross-linking has made significant contributions to protein interaction networks (Figeyns 2003a, b; Vasilescu and Figeyns 2006). Protein arrays have recently been developed to directly assess protein-protein interactions (Figeyns 2002; Sloane et al. 2002; Wilson and Nock 2002). One rapidly developing aspect of proteomics is the ChIP-on-chip and ChIP-Seq experiments which aim to identify protein-DNA interactions (Robertson et al. 2007). A large number of DNA sequences containing the putative protein-binding sites need BLAST and FASTA to map them to genomes.

Starting a bioinformatics book with BLAST and FASTA is in fact dictated by an important pedagogical principle, i.e., any theme of presentation should begin with an easily recognizable and ideally universal entry point. This pedagogical principle points unambiguously to BLAST and FASTA. Many colleagues of mine share the opinion that BLAST and FASTA in bioinformatics are equivalent to PCR in molecular biology wet laboratories and deserve more recognition. Furthermore, the mathematics and computation underlying these two widely used computational

tools represent the common denominator of mathematics and algorithms in many other bioinformatics tools. While a large number of researchers use BLAST and FASTA daily, not many of them actually appreciate the statistical and computational aspects of these two sets of programs. A simple but systematic presentation of the mathematical, statistical, and computational aspects of these bioinformatics tools will not only help researchers to better interpret their BLAST and FASTA output and implement these methods in their own programs but also foster better appreciation of bioinformatics as an interdisciplinary science and of its major players.

BLAST and FASTA belong to the category of sequence search and annotation tools. Once genomic scientists obtain a long contig or a genomic sequence by contig assembly, the next step is obviously to find out what the long stretch of A, C, G, and T means. This is the subject of sequence annotation of which the pivotal component is gene finding. There are two major categories of computational tools for gene finding. The first is based on known genes in molecular databases (gene dictionaries) and uses homology search tools such as FASTA (Pearson and Lipman 1988) and BLAST (Altschul et al. 1990, 1997). This approach has become increasing more effective with better and more complete gene dictionaries. The second, better known as gene prediction, is based on known gene structures and represented by GENSCAN (Burge and Karlin 1997). Existing software for gene finding often combine both approaches, e.g., GenMark (Hayes and Borodovsky 1998), GLIMMER (Salzberg et al. 1998), Orpheus (Frishman et al. 1998), Projector (Meyer and Durbin 2004), YACOP (Tech and Merkl 2003), and Prodigal (Hyatt et al. 2010).

Sequence matching caught the fancy of biologists when an oncogene (i.e., a gene in a virus that causes a cancer-like transformation of the infected cell), *v-sis*, was found to be similar to PDGF, the platelet-derived growth factor (Doolittle et al. 1983; Waterfield et al. 1983). This finding not only resulted in PDGF being used as a cancer drug target (Bergsten et al. 2001; Ehnman et al. 2013; Pietras et al. 2003) but also led to two new lines of thinking. First, the viral transforming factor may work simply by changing transient expression of a growth factor to constitutive expression, suggesting growth factors as targets for anticancer drug development. Second, any factors modulating gene expression patterns can potentially contribute to cancer. This new conceptual framework of cancer biology contributed to the progress of mechanism-based anticancer drug development in the following years (Gibbs 2000; Moffat et al. 2014; Shoemaker 2006).

Conventional methods for similarity searchers are based on local sequence alignment using dynamic programming (Smith and Waterman 1981). For a given scoring scheme, such methods will guarantee the finding of the optimal alignment. However, such methods are computation-intensive. FASTA and BLAST use heuristic methods for similarity search. They may miss homologous sequences but are very fast. With terabytes of molecular sequences in the database to search through, the speed becomes more important than sensitivity of homology detection.

A similarity search will generate a similarity score, which needs to be evaluated for its statistical significance. BLAST became more popular than FASTA partially because the early versions of FASTA did not evaluate the statistical significance of the resulting sequence matches. Latter versions of FASTA have incorporated such

evaluations. Currently, among the three synchronized international database hosts, NCBI and DDBJ use BLAST as the default search engine, whereas EBI offers both FASTA and BLAST for sequence searching. Japan’s GenomeNet also offers both BLAST and FASTA as search engines.

We will first offer a simple but detailed presentation of the mathematics involving string matching, which allows us to design a filter to eliminate a large number of insignificant matches. The string search algorithms used in FASTA and BLAST are then presented at a level to allow programmers to implement the algorithms in their own software.

## 2 Mathematics of String Matching

### 2.1 Basic Concepts

Given  $P_A$ ,  $P_C$ ,  $P_G$ , and  $P_T$  in a target (database) sequence, the probability of a query sequence (Q), say, ATTGCC, having a perfect match against a nonhomologous target sequence (D) is

$$p = P_A P_C^2 P_G P_T^2 \quad (1.1)$$

Let  $L_D$  be the target sequence length and  $L_Q$  be the query sequence length; the number of possible “matching operations,” i.e., number of times one can shift Q against D in search for a perfect match of  $L_Q$  letters, is

$$n = (L_D - L_Q + 1) \quad (1.2)$$

For example, with Q = ATG and D = CGATTGCCCG,  $L_Q = 3$ ,  $L_D = 10$ ,  $n = 8$ .

The probability distribution of the number of matches follows (approximately) a binomial distribution with  $p$  defined in Eq.(1.1),  $n$  defined in Eq.(1.2), and  $q = 1 - p$ :

$$(p + q)^n = p^n + \frac{n!}{(n-1)!1!} p^{n-1} q + \cdots + \frac{n!}{x!(n-x)!} p^x q^{n-x} + \cdots + q^n$$

$$p(x) = \frac{n!}{x!(n-x)!} p^x q^{n-x} \quad (1.3)$$

If  $P_A = P_C = P_G = P_T = 0.25$ , then  $p = 0.25^3 = 0.015625$ , and  $q = 1 - p = 0.984375$ . With  $L_Q = 3$ ,  $L_D = 10$ , we have  $n = 8$ . So the probabilities of having 0, 1, 2, ...,  $r$  exact matches of three letters are  $p(0) = 0.881626$ ,  $p(1) = 0.111953$ ,  $p(2) = 0.00622$ , and so on. The probability of having at least one match is simply  $1 - p(0) = 0.118373565$ .

### 2.1.1 Approximation of Binomial Distribution by Poisson Distribution

Binomial distribution is troublesome in computation when  $n$  is large because computers will generate an overflow error to get the factorial with a large  $n$ . When  $np < 1$  and  $n$  is very large, the binomial distribution can be approximated by the Poisson distribution with mean and variance equal to  $np$ . The mathematical detail of converting the binomial probability distribution to the Poisson distribution is shown below:

$$\begin{aligned} p(x) &= \frac{n!}{(n-x)!x!} p^x q^{n-x} = \frac{n!}{(n-x)!x!} p^x q^{-x} q^n \\ &= \frac{n(n-1)(n-2)\dots(n-x+1)}{x!} \left(\frac{p}{q}\right)^x (1-p)^n \\ &\approx \frac{n^x}{x!} p^x e^{-np} = \frac{n^x p^x}{x!} e^{-np} = \frac{(np)^x}{x!} e^{-np} = e^{-\lambda} \frac{\lambda^x}{x!} \end{aligned} \quad (1.4)$$

The approximation assumes a large  $n$  and a small  $p$  near 0, so that  $(n-x+1) \approx n$ ,  $p/q \approx p/1 = p$ , and  $(1-p)^n \approx e^{-np}$ . The Poisson distribution has only one parameter typically symbolized by  $\lambda$ , and  $P(x)$  is very easy to compute. To use the Poisson distribution to approximate the binomial example above with  $n = 8$  and  $p = 0.015625$ , we have  $\lambda = np = 0.125$ , and  $p(0) = 0.882496903$ ,  $p(1) = 0.110312113$ ,  $p(2) = 0.006894507$ , and so on. The probability of having at least one match is simply  $1 - p(0) = 0.117503097$ . Although our  $n$  is not very large and  $p$  not very small, these values are still quite similar to those from the binomial distribution. In nearly all practical searches against GenBank sequences,  $n$  is indeed very large and  $p$  very small.

The simple mathematics concerning string matching has used to check the chance of a DNA sequence ( $D$ ) getting cut by a restriction enzyme, e.g., NlaIII with the recognition site (GTAC). If  $D$  is slightly GC biased with  $P_C = P_G = 0.3$  and  $P_A = P_T = 0.2$ , what is the probability that  $D$  will not contain GTAC (and consequently will not get cut)? In this case  $Q = \text{GTAC}$ . Assuming  $L_D = 3000$ , we have  $n = (3000-4 + 1)$ ,  $p = 0.3^2 \times 0.2^2$ , and the Poisson parameter  $\lambda = 10.7892$ . The probability that a transcript does not have the restriction site is 0.000020621. However, if  $L_D = 300$ , the probability that a transcript does not have the restriction site is 0.343283.

### 2.1.2 E-Value

What we have covered so far involve the matching of an entire query sequence ( $Q$ ) against a data base sequence, but we often need to address the following question. Given a query sequence  $Q$  of length  $L_Q$  and a target string  $D$  of length  $L_D$ , what is the probability of finding a match of  $L$  consecutive characters, where  $L \leq L_Q$  and  $L \leq L_D$ ? Assuming  $P_A = P_C = P_G = P_T = 0.25$ , the probability of finding an exact match of length  $L$  consecutive letters is  $p = 0.25^L = 2^{-2L}$ . The

match of  $L$  consecutive letters between Q and D can happen at  $(L_Q - L + 1)$  positions on Q and at  $(L_D - L + 1)$  positions on D. We define

$$\begin{aligned} m &= L_Q - L + 1 \\ w &= L_D - L + 1 \end{aligned} \quad (1.5)$$

In BLAST literature and BLAST output,  $m$  and  $w$  are termed effective lengths. You might recall that, in the first edition of this book, I used  $m$  and  $n$  instead of  $m$  and  $w$ . Now inspired by liberal extremists such as Canadian Prime Minister Justin Trudeau who insists that we should use “peoplekind” instead of “mankind,” I made a politically correct move by replacing  $n$  by  $w$  to balance the use of  $m$ . In fact, given the observation that straightening a bent wire often requires overbending it in the opposite direction, one should idealogically replace both  $m$  and  $n$  by  $w$ . This might appear absurd in a short term but is said to have tremendous value over the long term.

There are  $mw$  possible matching operations, each with a probability of  $0.25^L$  of getting a match of  $L$  consecutive letters. The expected number of matches with length  $L$  (the  $E$ -value in the ungapped BLAST) is therefore

$$E = mw0.25^L \quad (1.6)$$

which, with  $mw$  kindling romantic imaginations, is far superior than the previous lifeless version with  $mn$ .

Most BLAST web interfaces allow you to input a cutoff  $E$ -value. Suppose we are to BLAST sequence Q of length 10 against genomic sequence D of length 10,000,000. Does it make sense to set the  $E$ -value to 0.01? It does not, because we expect nearly 10 perfect matches of Q against D by random chance in this case. In other words, it is impossible for any returned match to have an  $E$ -value of 0.01. The smallest E-value among the returned matches (i.e., those perfect matches) will be nearly 10. A user-friendly implementation of the BLAST or FASTA algorithm would ignore the input  $E$ -value and return all perfect matches or at least display a tactful message saying that the user has committed a forgivable error, but a strict implementation of the algorithm will simply return you with no match (a computational tool is user-friendly when the programmer works under the assumption that users may be foolish sometimes).

In BLAST and FASTA literature, one may also encounter an equation in the following form:

$$E = mwe^{-\lambda R} \quad (1.7)$$

where  $R$  is the raw score equal to  $L$  times the match score in ungapped string matching. If the match score is 1, then  $R = L$ . Eq. (1.7) and Eq. (1.6) are equivalent with  $\lambda = -\ln(0.25) = 1.386294361$  and  $R = L$ , where the value of 0.25 arises from the assumption of equal nucleotide frequencies.

Noting that  $0.25 = 2^{-2}$ , we can also rewrite Eq. (1.6) as

$$E = mw2^{-2L} = mw2^{-S} \quad (1.8)$$

where  $S = 2L$  for ungapped matches. Equation (1.8) has become particularly useful because it was found through computer simulations (Altschul 1996; Altschul et al. 1997; Pearson 1998; Waterman and Vingron 1994) that, when  $S$  is computed with a particular scoring scheme (illustrated in Fig. 1.1), Eq. (1.8) can be applied to situations involving two strings not only with consecutive matched letters but also with mismatches and gaps (i.e., gapped BLAST).

Figure 1.1 shows the output from BLASTing a nucleotide query sequence against a local BLAST database containing all annotated *Mycoplasma genitalium* coding sequences. From the output (Fig. 1.1), we see 35 matches, 3 mismatches, 1 gap open, and 2 gap extensions. The scoring scheme has the match score ( $s_{ii}$ ) equal to 1, mismatch score ( $s_{ij}$ ) equal to  $-3$ , gap open penalty ( $G_o$ ) equal to 5, and gap extension penalty ( $G_e$ ) equal to 2 (Fig. 1.1). When the raw score ( $R$ ) is computed according to this scoring scheme, and the bit-score ( $S$ ) is computed with  $R$  and two scaling factors  $\lambda$  and  $K$ , Eq. (1.8) can be used to obtain the  $E$ -value:

```
BLASTN 2.2.4 [Aug-26-2002]
...
Query= Seq1 38
Database: MgCDS
        480 sequences; 526,317 total letters
Score      E
Sequences producing significant alignments: (bits) Value
MG001 1095 bases
Score = 34.2 bits (17), Expect = 7e-004
Identities = 35/40 (87%), Gaps = 2/40 (5%)

Query: 1 atgaataacg--attatttccaacgacaaaacaaaaccac 38
       ||||||||| ||||||||| ||||| ||||| |
Sbjct: 1 atgaataacgttattatttccaataacaaaataaaaccac 40

Lambda      K      H
1.37      0.711    1.31
Matrix: blastn matrix:1 -3
Gap Penalties: Existence: 5, Extension: 2
...
effective length of query: 26
effective length of database: 520,557
```

**Fig. 1.1** Output from a nucleotide BLAST query

$$\begin{aligned}
R &= 35 \times 1 + 3 \times (-3) - 1 \times 5 - 2 \times 2 = 17 \\
S &= \frac{\lambda R - \ln(K)}{\ln(2)} = \frac{1.37 \times 17 - \ln(0.711)}{\ln(2)} \approx 34.1 \\
E &= mw2^{-S} = 26 \times 520557 \times 2^{-S} = 0.000735
\end{aligned} \tag{1.9}$$

where  $\lambda$  and  $K$ , shown in every BLAST output, are scale and location parameters (Altschul et al. 1997, and literature cited therein) estimated from computer simulation. Note that the E-value is calculated in the same way as in Eq. (1.8).

There are discrepancies in literature concerning the counting of gap extensions. Given the gap we see in Fig. 1.1, it would make more sense to count one gap open and one gap extension, so that the total penalty would be  $-7$  ( $= -5 - 2$ ). However, in BLAST, a gap of length  $K$  is penalized by  $-(G_o + KG_e)$  where  $K$  is the total length of gaps. In the alignment in Fig. 1.1, the gap penalty is therefore  $-(5 + 2*2) = -9$ .

One may note that  $L$  in Eq. (1.6) is the same as  $R$  given  $s_{ii} = 1$ , because  $R = L \times s_{ii} = L$  in case of exact string match with no mismatches or gaps involved. Because  $S = 2L$  in Eq. (1.8) and because  $L = R$  with ungapped match and  $s_{ii} = 1$ , we have  $S = 2R$ . This relationship between  $S$  and  $R$  is similar to their relationship specified in the second equation in Eq. (1.9). Note that  $R$  will increase with the length of the query and target sequences. With large  $R$ ,  $S$  is approximately equal to  $2R$ , i.e.,

$$S = \frac{\lambda R - \ln(K)}{\ln(2)} \approx 2R \text{ (with large } R\text{)} \tag{1.10}$$

The formula of  $E = mw2^{-S}$ , when  $S$  is represented as in Eq.(1.10), is also often expressed as

$$E = Kmwe^{-\lambda R} \tag{1.11}$$

The two are equivalent because  $2^{-S} = Ke^{-\lambda R}$ . You can see this by simply taking the natural logarithm of both sides:

$$\begin{aligned}
-S \ln 2 &= \ln K - \lambda R \\
S &= \frac{\lambda R - \ln K}{\ln 2}
\end{aligned} \tag{1.12}$$

The  $E$ -value can be used as the  $\lambda$  parameter in the Poisson distribution in Eq. (1.4) to get the probability of having  $0, 1, \dots, x$  matches that are as good as or better than the reported match. For our example,  $p(0) = 0.999265217$ ,  $p(1) = 0.000734513$ , and so on. According to the Poisson distribution in Eq. (1.4), the probability of having at least one match (i.e.,  $x \geq 1$ ) that is as good or better than the reported match, or in other words the probability of having a raw score ( $R$ ) at least as large as the observed raw score ( $R_{\text{obs}}$ ), is

$$pr(x \geq 1) = pr(R \geq R_{\text{obs}}) = 1 - p(0) = 1 - e^{-E} = 1 - e^{-mw e^{-\lambda R}} \tag{1.13}$$

where we have substituted  $E$  with the expression in Eq. (1.7). Note that BLAST scales the  $E$ -value with a parameter  $K$ , which tends to bias the BLAST output of the  $E$ -value on the conservative side.

The last term in Eq. (1.13) has a rather special term associated with it in probability theory. In short, the probability distribution

$$p(R) = e^{-mwe^{-\lambda R}} \quad (1.14)$$

with the ugly and cumbersome exponential of an exponential is a special form of the extreme value distribution or EVD, also referred to as the Gumbel distribution in honor of the pioneer of the statistics of extremes (Gumbel 1958). The EVD is used in BLAST (Altschul et al. 1990, 1997) and FASTA (Pearson 1998) to attach statistical significance to a match score between two sequences.

The  $E$ -value is the expected number of random matches with match scores that are equally good or better than the reported one. It is not a probability, and it can take a value of 10 or a million. However, when  $E$  is very small, then it can be approximately interpreted as the probability of finding exactly one match. This is shown below based on the Poisson distribution:

$$p(1) = e^{-E}E \approx E \text{ because } \lim_{E \rightarrow 0} e^{-E} = 1 \quad (1.15)$$

An inverse problem is to obtain the length of an exact match given a critical  $E$ -value. For example, the  $E$ -value is an input parameter during a BLAST search. One can take such an input  $E$ -value to find the critical length of exact string matches, designated as  $L_{\text{crit}}$ . This  $L_{\text{crit}}$  allows us to quickly eliminate all exact string matches shorter than  $L_{\text{crit}}$ . So how to obtain  $L_{\text{crit}}$ ?

A naïve approach is to try to solve the following equation for  $L_{\text{crit}}$ :

$$E_{\text{crit}} = mw2^{-2L_{\text{crit}}} = (L_Q - L_{\text{crit}} + 1)(L_D - L_{\text{crit}} + 1)2^{-2L_{\text{crit}}} \quad (1.16)$$

It turns out that Eq. (1.16) does not have an analytical solution for  $L_{\text{crit}}$  because  $L_{\text{crit}}$  is an implicit function of  $E_{\text{crit}}$ . To obtain  $L_{\text{crit}}$  given  $E_{\text{crit}}$ , one would have to use numerical solution by iteration. Efficient iteration methods are available (Press et al. 1992), but one may also use the approximate method below:

$$L_{\text{crit}} = L_{\text{guess}} + \frac{\ln\left(\frac{E_{\text{guess}}}{E_{\text{crit}}}\right)}{2 \ln(2)} \quad (1.17)$$

where

$$E_{\text{guess}} = L_Q L_D 2^{-2L_{\text{guess}}} \quad (1.18)$$

$$L_{\text{guess}} = -\frac{\ln\left(\frac{E_{\text{crit}}}{L_Q L_D}\right)}{2 \ln(2)} \quad (1.19)$$

**Table 1.1** Selected results computing  $L_{\text{crit}}$  by applying Eq. (1.17) with  $E_{\text{crit}} = 0.01$ . The values in last column are the  $E$ -value based on  $L_{\text{crit}}$  and are very close to the preset critical value of 0.01

$L_Q$	$L_D$	$L_{\text{guess}}$	$E_{\text{guess}}$	$L_{\text{crit}}$	$E$
1000	10,000	14.94868	0.009847	14.93754	0.0100
1000	1000	13.28771	0.009756	13.26988	0.0100
10,000	100	13.28771	0.008760	13.19225	0.0100
1000	15	10.25827	0.003792	9.55885	0.0112
100	15	8.59730	0.004560	8.03089	0.0108
15	15	7.22882	0.003419	6.45470	0.0118
15	1000	10.25827	0.003792	9.55885	0.0112
15	10,000	11.91923	0.002718	10.97942	0.0123
15	1,000,000	14.00000	0.007450	13.78770	0.0111

$L_{\text{guess}}$  in Eq. (1.19) is derived from Eq. (1.16) by assuming  $L_Q \gg L_{\text{crit}}$  and  $L_D \gg L_{\text{crit}}$ , so that  $(L_Q - L_{\text{crit}} + 1) \approx L_Q$  and  $(L_D - L_{\text{crit}} + 1) \approx L_D$ . For this reason  $L_{\text{guess}}$  should be constrained to be no larger than  $(L_D - 1)$  or  $(L_Q - 1)$ .

Now if we have  $L_Q = 100$ ,  $L_D = 10,000$ , and  $E_{\text{crit}} = 0.01$ , then  $L_{\text{guess}} = 13.28771$  and

$$\begin{aligned} E &= mw2^{-2L_{\text{guess}}} = 0.876 \\ L_{\text{crit}} &= L_{\text{guess}} + \ln(E/E_{\text{crit}})/\ln(4) = 13.192251 \end{aligned} \quad (1.20)$$

Thus, when BLASTing a query sequence of 100 bases against a database sequence of 10,000 bases with a critical  $E$ -value of 0.01, we may ignore those with an exact string match shorter than 13 bases. Table 1.1 shows that Eq. (1.17) is good over a wide range of  $L_Q$  and  $L_D$  values. I should emphasize here again that one should set sensible  $E_{\text{crit}}$  for calculating  $L_{\text{crit}}$ . For example, if  $L_Q = 10$  and  $L_D = 10,000,000$  bases, one would be silly to compute  $L_{\text{crit}}$  by setting  $E_{\text{crit}} = 0.01$  or smaller. As I have mentioned before, it is impossible to have any sequence match with  $E_{\text{crit}} = 0.01$  or smaller in this case. Consequently, any  $L_{\text{crit}}$  calculated from such an  $E_{\text{crit}}$  would be of no use.

## 2.2 BLAST Parameters $\lambda$ , $K$ , and $H$

In the calculation of the  $E$ -value for the BLAST output in Fig. 1.1, we have used parameters  $\lambda$  (lambda),  $K$  (Karlin parameter or Karlin-Altschul parameter), and  $H$  which were given in Fig. 1.1. How are these parameters obtained? Are they constants or search-specific? Because we cannot claim to have a good understanding of the  $E$ -value from BLAST and FASTA programs without knowing how to obtain these parameters, this section explains these parameters.

The parameters are not constants. The scale parameter  $\lambda$  depends on two sets of variables. The first is the match and mismatch scores, designated  $s_{ij}$  for nucleotide or

amino acid  $i$  and  $j$  ( $i$  and  $j$  take the value of 1–4 for nucleotides A, C, G, and T, or 1 to 20 for the 20 amino acids). For nucleotide BLAST output in Fig. 1.1,  $s_{ii} = 1$  (for matches) and  $s_{ij} = -3$  (for mismatches), but you can set your own  $s_{ii}$  and  $s_{ij}$  values in most BLAST servers which would change  $\lambda$  (as well as  $K$  and  $H$ ). The second set is the nucleotide or amino acid frequencies, designated by  $p_i$  (where the subscript  $i$  takes the value of 1–4 for nucleotide sequences and 1–20 for amino acid sequences). Changing either  $s_{ii}$  and  $s_{ij}$  values or  $p_i$  values would change  $\lambda$  (as well as  $K$  and  $H$ ). This would become clear given that  $\lambda$  is estimated from the following equations, the first for nucleotide BLAST and the second for protein BLAST:

$$\begin{aligned} \sum_{i=1}^4 \sum_{j=1}^4 p_i p_j e^{s_{ij}\lambda} &= p_A^2 e^{s_{AA}\lambda} + p_A p_C e^{s_{AC}\lambda} + \cdots p_T^2 e^{s_{TT}\lambda} = 1 \\ \sum_{i=1}^{20} \sum_{j=1}^{20} p_i p_j e^{s_{ij}\lambda} &= p_{aa_1}^2 e^{s_{aa_1,aa_1}\lambda} + p_{aa_1} p_{aa_2} e^{s_{aa_1,aa_2}\lambda} \\ &\quad + \cdots p_{aa_{20}}^2 e^{s_{aa_{20},aa_{20}}\lambda} = 1 \end{aligned} \tag{1.21}$$

In the simplest case with equal nucleotide frequencies, i.e., when  $p_i = 0.25$ , and for a given  $s_{ii} = 1$  and  $s_{ij} = -3$  as in default nucleotide BLAST, we have

$$\begin{aligned} \sum_{i=1}^4 \sum_{j=1}^4 p_i p_j e^{s_{ij}\lambda} &= 4 \times 0.25^2 e^\lambda + 12 \times 0.25^2 e^{-3\lambda} \\ &= 0.25e^\lambda + 0.75e^{-3\lambda} = 1 \end{aligned} \tag{1.22}$$

Solving the equation in the real domain (i.e., ignoring solutions with complex numbers) for a non-zero solution of  $\lambda$ , we have  $\lambda = 1.374063122$ , which is the same as in the BLAST output in Fig. 1.1 (ignoring rounding errors). Note that 0.25 in  $0.25e^\lambda$  in Eq. (1.22) is the probability of having a match (when  $i = j$ ) and 0.75 is the probability of having a mismatch ( $i \neq j$ , and assuming equal nucleotide frequencies). For an approximate solution, one may substitute different  $\lambda$  values until the left side is close to 1. This is illustrated in Fig. 1.2 for one to experiment with different nucleotide frequencies.

The illustration above shows that different  $\lambda$  values imply different nucleotide frequencies of the database sequences. A  $\lambda = 1.37$  is associated with nucleotide sequences of roughly equal frequencies, whereas a  $\lambda = 0.658295$  is associated with strongly biased frequencies. For this reason, BLAST parameters  $\lambda$ ,  $K$  and  $H$  are computed for each BLAST database created. If you BLAST your query sequence against a BLAST database, the output will show  $\lambda$ ,  $K$  and  $H$  values specific to that database.

If you have very different nucleotide frequencies, e.g., if  $p_A = p_T = 0.1$  and  $p_C = p_G = 0.4$ , then the probabilities that the two nucleotides from such a nucleotide pool being identical  $p_{\text{Match}}$  and being different  $p_{\text{Mismatch}}$  will be

(a)		A	G	C	T	(a')		A	G	C	T
		0.25	0.25	0.25	0.25			0.49	0.01	0.01	0.49
A	0.25	<b>0.0625</b>	0.0625	0.0625	0.0625	A	0.49	<b>0.2401</b>	0.0049	0.0049	0.2401
G	0.25	0.0625	<b>0.0625</b>	0.0625	0.0625	G	0.01	0.0049	<b>0.0001</b>	0.0001	0.0049
C	0.25	0.0625	0.0625	<b>0.0625</b>	0.0625	C	0.01	0.0049	0.0001	<b>0.0001</b>	0.0049
T	0.25	0.0625	0.0625	0.0625	<b>0.0625</b>	T	0.49	0.2401	0.0049	0.0049	<b>0.2401</b>

Match-mismatch matrix						Match-Mismatch matrix					
(b)	A	1	-3	-3	-3	(b')	A	1	-3	-3	-3
	G	-3	1	-3	-3		G	-3	1	-3	-3
	C	-3	-3	1	-3		C	-3	-3	1	-3
	T	-3	-3	-3	1		T	-3	-3	-3	1

Lambda	1.374063					Lambda	0.658295				
(c)		0.246961	0.001013	0.001013	0.001013	(c')	0.463752	0.00068	0.00068	0.03332	
		0.001013	0.246961	0.001013	0.001013		0.00068	0.000193	1.39E-05	0.00068	
		0.001013	0.001013	0.246961	0.001013		0.00068	1.39E-05	0.000193	0.00068	
		0.001013	0.001013	0.001013	0.246961		0.03332	0.00068	0.00068	0.463752	
					1						1

**Fig. 1.2** Different  $\lambda$  values imply different nucleotide frequencies. (a) The four equal nucleotide frequencies and their associated 16  $p_i p_j$  terms in Eq. (1.21), (b) a match-mismatch matrix, (c) the 16  $p_i p_j e^{s_{ij}\lambda}$  terms in Eq. (1.21) which should add up to 1 when  $\lambda = 1.374063$ . (a'), (b'), and (c') corresponds to (a), (b), and (c) except that the nucleotide frequencies are strongly AT-biased. The 16 terms in (c') will add up to 1 when  $\lambda = 0.658295$

$$\begin{aligned} p_{\text{Match}} &= 2 \times 0.1^2 + 2 \times 0.4^2 = 0.34 \\ P_{\text{Mismatch}} &= 1 - p_{\text{Match}} = 0.66 \end{aligned} \quad (1.23)$$

So we now find  $\lambda = 1.0501$  by applying Eq. (1.21):

$$p_{\text{Match}} e^\lambda + p_{\text{Mismatch}} e^{-3\lambda} = 0.34e^\lambda + 0.66e^{-3\lambda} = 1 \quad (1.24)$$

Computing  $\lambda$  for amino acid sequences is similar, except that  $s_{ii}$  and  $s_{ij}$  are typically replaced by a score matrix such as PAM30 (Table 1.2). If we assume equal amino acid frequencies, i.e.,  $p_i = 1/20 = 0.05$ , then Eq. (1.22) is reduced to the following form which will yield  $\lambda = 0.3073$ . You may verify this by replacing  $\lambda$  with 0.3073 to check if the left-hand side of the equation is very close to 1. Note that a different  $s_{ij}$  matrix would lead to a different  $\lambda$ . The chapter on sequence alignment provides a detailed explanation of different amino acid score matrices:

$$\begin{aligned} \sum_{i=1}^{20} \sum_{j=1}^{20} p_i p_j e^{s_{ij}\lambda} &= p_A^2 e^{s_{AA}\lambda} + p_A p_R e^{s_{AR}\lambda} + \dots + p_V^2 e^{s_{VV}\lambda} \\ &= 0.05^2 (e^{6\lambda} + e^{-7\lambda} + \dots + e^{7\lambda}) = 1 \end{aligned} \quad (1.25)$$

Amino acid frequencies are typically not equal. How much will  $\lambda$  be affected by different amino acid sequences? The yeast (*Saccharomyces cerevisiae*) has an AT-biased genome, and its protein sequences have amino acid frequencies quite different from being equal (Table 1.3). By using the PAM30 matrix and amino acid frequencies in Table 1.3, we have  $\lambda = 0.3548$ .

The Karlin-Altschul parameter  $K$  also depends on  $s_{ij}$  values, as well as on the probabilities associated with the  $s_{ij}$  values (which depend on nucleotide or amino acid frequencies). For example, if we set the match score  $s_{ii} = 1$  and the mismatch

Table 1.2 PAM30 matrix

**Table 1.3** Amino acid count (Number) and frequencies (Freq) from all 5865 proteins from the yeast, *Saccharomyces cerevisiae*

AA	Number	Freq
Ala	159,844	0.054993
Arg	129,008	0.044384
Asn	178,871	0.061539
Asp	170,370	0.058614
Cys	36,571	0.012582
Gln	115,137	0.039612
Glu	190,478	0.065532
Gly	144,605	0.04975
His	62,925	0.021649
Ile	190,314	0.065476
Leu	276,067	0.094978
Lys	213,573	0.073478
Met	60,205	0.020713
Phe	128,282	0.044134
Pro	127,299	0.043796
Ser	261,382	0.089926
Thr	171,964	0.059163
Trp	30,156	0.010375
Tyr	97,950	0.033699
Val	161,628	0.055607

score  $s_{ij} = -3$ , then these values span a range of 5 integer values constituting an  $s_{ij}$  vector  $[-3, -2, -1, 0, 1]$ . If we assume equal nucleotide frequencies, then the probability for two randomly drawn nucleotides to match is 0.25 and that for two nucleotides to mismatch is 0.75. So the probability vector is  $[0.75, 0, 0, 0, 0.25]$  corresponding to the  $s_{ij}$  vector  $[-3, -2, -1, 0, 1]$ . In other words, we have a probability of 0.75 to get a mismatch with an  $s_{ij} = -3$  and a probability of 0.25 to get a match with an  $s_{ii} = 1$ . Given specification that  $s_{ii} = 1$  and  $s_{ij} = -3$ , there is no probability for us to get  $s_{ij}$  scores other than  $-3$  and  $1$ , so the probabilities corresponding to  $-2$ ,  $-1$ , and  $0$  are all zero in the probability vector.

The  $s_{ij}$  vector and the probability vector (which depends on the frequency parameters) completely determine the parameters  $\lambda$ ,  $K$ , and  $H$ . Given the two vectors in the previous paragraph, we will have  $\lambda = 1.3741$ ,  $H = 1.3072$ , and the Karlin-Altschul parameter  $K = 0.7106$ . One may have a more complicated  $s_{ij}$  vector and the associated probability vector. For example, one may break the mismatch into transitions and transversions, with  $s_{ij} = -1$  for transitions and  $s_{ij} = -3$  for transversions. Now the  $s_{ij}$  vector would still be  $[-3, -2, -1, 0, 1]$ , but the associated probability vector would be  $[0.5, 0, 0.25, 0, 0.25]$  assuming equal nucleotide frequencies. Note that, assuming equal nucleotide frequencies, the probability of having a transversional mismatch is 0.5 (penalized by  $-3$ ), that of having a

transitional mismatch is 0.25 (penalized by  $-1$ ), and that of having a match is 0.25 (awarded by 1). This new probability vector will lead to  $\lambda = 1.3054$ ,  $H = 1.0765$ , and  $K = 0.6011$ .

While the theory behind the computation of these parameters is beyond this book, the actual computation is not too complicated. You can obtain these parameters by using DAMBE (Xia 2013, 2017d). Click “Alignment!Karlin-Altschul parameters” and choose either nucleotide or amino acid sequences by clicking the “Nuc” or “AA” option button. The default is nucleotide. You need to paste into the input text box an  $s_{ij}$  matrix and four nucleotide frequencies for A, G, C, and T, respectively. The sample input is provided, with a score of 1 for a match, a score of  $-1$  for a transition difference, and a score of  $-3$  for a transversional difference. You can either modify the sample input or paste into the box your own  $s_{ij}$  matrix and nucleotide frequencies. Click the “Get  $L$ ,  $K$ ,  $H$ ” button to calculate  $\lambda$ ,  $K$ , and  $H$  parameters. For amino acid sequences, the sample input has a PAM30 matrix as the  $s_{ij}$  matrix. Click the “Get  $L$ ,  $K$ ,  $H$ ” button, and  $\lambda$ ,  $K$ , and  $H$  parameters appropriate for the  $s_{ij}$  matrix and the specific amino acid frequencies will be calculated.

### 3 String-Matching Algorithms in FASTA and BLAST

Heuristic local similarity search algorithms, which both FASTA and BLAST algorithms belong to, generally include the following three steps: (1) finding an exactly or inexactly matched string segment, (2) evaluating the statistical significance of the match, and (3) if the match is statistically significant, extending the matched string segment in both directions by dynamic programming. The second step has already been covered in the previous section, and the third step will be covered in the chapter on sequence alignment. This section will cover the FASTA and BLAST algorithms used in the first step. It is the difference in this first step that is responsible for the higher sensitivity of the FASTA algorithm than the BLAST algorithm.

The database sequences are always preprocessed and indexed to speed up the sequence matching between a query and database sequences. For example, suppose a database protein sequence (D) of 300 amino acid residues. We preprocess D to find that it has no Trp, that it contains a single Arg at site 222, and that its three Lys residues occur at sites 3, 22, and 123. If we now have a query oligopeptide (Q) that contains no Trp, then we know immediately that there is no exact match between Q and D without any need to search Q along the length of D. Similarly, if Q does not have Trp but has an Arg at site 2, then we can immediately match the Arg at site 2 of Q against the Arg at site 222 of D and check if the neighboring sites match, again without any need to search Q against the length of D. For this reason, a BLAST (or FASTA) database will always contain a (compressed) sequence file together with index files.

### 3.1 FASTA Algorithm

The FASTA suite of programs (Pearson 1994; Pearson and Lipman 1988) in fact implements a number of different search algorithms, and I will illustrate only the basic ones here. The purpose is to give computer programmers sufficient details for them to implement the algorithm for homology searching in their own programs.

Suppose we wish to find the similarity between the following query sequence (Q) and the target sequence (D), with sites numbered from 0:

```
0123456789012345678
Q: ACCGCGACCCTGACGAATA
D: ACCGCGATGACGAATA
```

The FASTA algorithm consists of three steps in achieving the heuristic local alignment. First, it indexes D for a given word length. For simplicity, we start with a word length of 1 (in which case “word” and “letter” become synonymous, i.e., a letter is a word of length 1), with the resulting index table shown in Table 1.4. We note that word A occurs at positions 0, 6, 9, 12, 13, and 15 in D, and these numbers occupy the first row in Table 1.4. Word C occurs at positions 1, 2, 4, and 10, and the numbers occupy the second row of Table 1.4 and so on. The numbers in Table 1.4 will be referred to as  $I_D$  (Index of D) numbers or  $I_D$  values.  $I_D$  values are needed in the second step. For a fixed word length, the computation of  $I_D$  values is done before the user has submitted any query and consequently would belong to what is called database preprocessing. In algorithmic terms, Table 1.4 is represented by an array of four elements (designated A, C, G, and T) each with a linked list of length 6, 4, 4, and 2, respectively.

In the second step of the FASTA algorithm, we designate the site number of Q as  $S_Q$  and compute  $S_Q - I_D$  (Table 1.4). For example, nucleotide A occurs at site 0 in Q, and the differences (i.e.,  $S_Q - I_D$ ) between this 0 and the 6  $I_D$  values for A in Table 1.4 (0, 6, 9, 12, 13, and 15) are consequently (0–0), (0–6), (0–9), (0–12), (0–13), and (0–15), respectively (first row of  $S_Q - I_D$  values in Table 1.5). Similarly, nucleotide C occurs at site 1 in Q. Because the  $I_D$  values for C are 1, 2, 4, and 10, respectively (Table 1.4), we have (1–1) = 0, (1–2) = −1, (1–4) = −3, and (1–10) = −9. These values occupy the second row of the  $S_Q - I_D$  values in Table 1.5.

**Table 1.4** First step in the FASTA algorithm: generating an index table of the target sequence D with a word length of 1

Base	$I_D$					
A	0	6	9	12	13	15
C	1	2	4	10		
G	3	5	8	11		
T	7	14				

$I_D$  values are the sites of the corresponding base (A, C, G, and T) found in D, e.g., base A is found at site 0, 6, 9, 12, 13, and 15, respectively

**Table 1.5** Second step in the FASTA algorithm: computing the difference between the site number of the query sequence Q ( $S_Q$ ) and  $I_D$ 

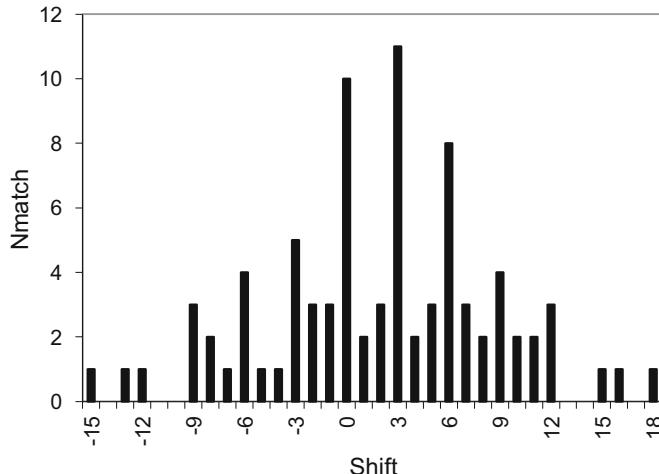
Q	$S_Q$	$S_Q - I_D$					
A	0	0	-6	-9	-12	-13	-15
C	1	0	-1	-3	-9		
C	2	1	0	-2	-8		
G	3	0	-2	-5	-8		
C	4	3	2	0	-6		
G	5	2	0	-3	-6		
A	6	6	0	-3	-6	-7	-9
C	7	6	5	3	-3		
C	8	7	6	4	-2		
C	9	8	7	5	-1		
T	10	3	-4				
G	11	8	6	3	0		
A	12	12	6	3	0	-1	-3
C	13	12	11	9	3		
G	14	11	9	6	3		
A	15	15	9	6	3	2	0
A	16	16	10	7	4	3	1
T	17	10	3				
A	18	18	12	9	6	5	3

Note that each  $(S_Q - I_D)$  value has meanings. For example, the first  $(S_Q - I_D)$  value in the first row (corresponding to the first nucleotide A in Q) is 0. This means that this first nucleotide A in Q, without shifting Q with respect to D, will find a matched A in D. Similarly, the second  $(S_Q - I_D)$  value in the first row is -6 which means that the first A in Q will find a matched A in D if we shift Q 6 positions to the right with respect to D. In short, each negative  $(S_Q - I_D)$  value corresponding to a nucleotide in Q means that the nucleotide will find a match in D if Q is shifted right by  $|(S_Q - I_D)|$  positions with respect to D, and each positive  $(S_Q - I_D)$  value corresponding to a nucleotide in Q means that the nucleotide will find a match in D if Q is shifted  $(S_Q - I_D)$  positions to the left with respect to D.

The third (and the last) step is simply to compile a frequency distribution of  $(S_Q - I_D)$  values. We note that there are 10  $(S_Q - I_D)$  values equal to 0. This means 10 matched letters between Q and D without shifting either sequence left or right. We also find 11  $(S_Q - I_D)$  values equal to 3 in Table 1.5, which means that by shifting D 3 positions to the right against Q (or shifting Q 3 positions to the left against D), we will get a match of 11 letters:

Q: ACCGCGACCCCTGACGAATA  
D: ---ACCGCGATGACGAATA

For matching two long sequences, it is more informative to generate a histogram of the  $(S_Q - I_D)$  values (Fig. 1.3). The histogram not only helps us visually see which



**Fig. 1.3** Frequency distribution of  $S_Q - I_D$  values in Table 1.5

value has the highest frequency but is also very useful for obtaining nonintersecting matched segments. For example, knowing that one can get a match of 10 letters without shifting D left or right relative to Q and that one can get a match of 11 letters by shifting D 3 spaces to the right against Q helps us identify two nonintersecting matched segments as follows, with the first having 7 consecutive matches and the second having 9 consecutive matches:

Q: ACCGCGACCCCTGACGAATA  
D: ACCGCGA---TGACGAATA

One might wish to have a formal definition of “nonintersecting matched segments.” They are matched segments that do not overlap and are often defined with reference to intersecting matched segments. Take the following two sequences, for example:

Q: ACCGCGACCCCTGACGAATA  
D: ACCGCGACGACCCCTGACGAATA

There are two intersecting matched segments below, i.e., matched segments that overlap:

Q: ACCGCGAC.....  
D: ACCGCGAC.....

Q: .....GACCCCTGACGAATA  
D: .....GACCCCTGACGAATA

While FASTA algorithms are for local sequence alignment, the trick it offers for finding nonintersecting segments is very important in aligning two very long sequences with sequence length in the range of millions. For such sequences, direct application of dynamic programming is often impractical because of limited memory in computers to store the score and backtrack matrices. For example, two sequences with their lengths in the order of 1,000,000, and with the scoring matrix containing only integer values each taking up 4 bytes, a single score matrix would consume  $4 \times 10^{12}$  bytes. Few computers today have such a large amount of memory. However, one can use FASTA to first find nonintersecting segments and use them as anchor points and then align the sequences between anchor points by applying the dynamic programming algorithms which will be explained in the chapter on sequence alignment.

It is easy to see that, with increasing length of Q and D, ( $S_Q - I_D$ ) values will be too many, their frequency distribution will become less informative, and the computation will be more tedious, if we continue to use a word length of 1. In practical FASTA applications, the word length is almost always larger than 1 and would increase with the sequence length.

We now illustrate the application of the algorithm with a word length of 2. There are 16 dinucleotides and their occurrences in sequence D are listed in the first four columns of Table 1.6. These four columns are equivalent to Table 1.4 containing  $I_D$  values. The sequence Q is then represented as overlapping dinucleotides. For example, a sequence “AACG” would be represented as three overlapping dinucleotides as “AA”, “AC,” and “CG.” Such a representation of sequence Q, together with  $S_Q$  and  $S_Q - I_D$  values, is compiled in the last six columns in Table 1.6.

Note that the index table in Table 1.6, represented by the first four columns, is made of an array of 16 elements each with a linked list with its length varying from 0 to 3. Also note that the index table would be mostly empty had we used a word length of 3 or 4 for comparing Q and D. The word length should increase with the sequence length of Q and D.

We have many fewer ( $S_Q - I_D$ ) values with a word length of 2 (Table 1.6) than with a word length of 1 (Table 1.5). This implies a substantial saving of computational time. We note eight ( $S_Q - I_D$ ) values equal to 3 (Table 1.6), implying eight dinucleotide matches between Q and D. These eight overlapping dinucleotides are TG, GA, AC, CG, GA, AA, AT, and TA:

```
Q: ACCGCGACCCTGACGAATA
D: ---ACCGCGATGACGAATA
```

In practical computation involving nucleotide sequences, a word length of 4 is frequently used partly because tetranucleotides AAAA, AAAC, ..., TTTT correspond to byte values 0, 1, ..., 255. In other words, each tetranucleotide can be stored in only 1 byte, with A, C, G, and T each represented by two bits, i.e., 00, 01, 10 and 11, respectively. This means that AAAA is coded by the binary number 00000000, AAAC by 00000001, ..., and TTTT by 11111111. BLAST databases also use this encoding to save storage space for nucleotide sequences.

**Table 1.6** Illustration of the FASTA algorithm with word length of 2. (1) An index table of 16 dinucleotides (DiNuc) for D, with the numbers indicating the position of the dinucleotides in D, e.g., dinucleotide AA occurs at position 12 of D). (2) Q in overlapping dinucleotide representation and its corresponding site index ( $S_Q$ ). (3) Computation of the  $(S_Q - I_D)$  values. For example, the first AC in Q occurs at site 0, so its  $S_Q = 0$ . AC occurs in D at positions 0 and 9, respectively, which are its  $I_D$  values. So the two  $(S_Q - I_D)$  values for the first dinucleotide AC in Q are (0–0) and (0–9), respectively

(1)				(2)		(3)		
DiNuc	$I_D$			Q	$S_Q$	$S_Q - I_D$		
AA	12			AC	0	0	-9	
AC	0	9		CC	1	0		
AG				CG	2	0	-2	-8
AT	6	13		GC	3	0		
CA				CG	4	2	0	-6
CC	1			GA	5	0	-3	-6
CG	2	4	10	AC	6	6	-3	
CT				CC	7	6		
GA	5	8	11	CC	8	7		
GC	3			CT	9			
GG				TG	10	3		
GT				GA	11	6	3	0
TA	14			AC	12	12	3	
TC				CG	13	11	9	3
TG	7			GA	14	9	6	3
TT				AA	15	3		
				AT	16	10	3	
				TA	17	3		

Note that we do not need to write separate FASTA algorithms for string match involving different word length. If we need to do FASTA involving two nucleotide sequences with a word length of 2, we simply recode the nucleotide sequences into two new sequences with 16 different codes and use the same algorithm for a word length of 1.

### 3.2 BLAST Algorithm

While FASTA is often found in many European data centers, BLAST is the main, and often the only, search engine in sequence database servers hosted in North America. So it is relevant for a bioinformatics student to gain some familiarity with the algorithm.

The BLAST algorithm has three steps. For BLAST servers, the first is the preprocessing of the database sequences and does not consume query time (because it is done before the user submits the query). Each database sequence of length  $L_D$  is chopped into overlapping words of a fixed length  $L$ , with word  $i$  starting at position

**Table 1.7** Index table for the partial COI sequence from *Masturus lanceolatus*. Tetranucleotides not present in the sequence will have an empty linked list

Word	Index	Linked list				
AAAA	0					
AAAC	1	501	526			
AAAG	2	24				
AAAT	3	142	224	389		
AACA	4	279	422	485		
AACC	5	14	18	115	502	527
AACG	6	77	356			
AACT	7	474				
AAGA	8	25				
AAGC	9	86	343			
AAGG	10					
...						
TTTG	254	51	174	219	292	537
TTT	255	6	7	184	291	429

$i$  where  $i = 1, 2, \dots, (L_Q - L + 1)$ . The frequently occurring words are eliminated to reduce the chance of random matching, and low-complexity words (e.g., AAAA) are eliminated to reduce the bias in calculating probabilities and expected values. For example, if sequence D = “AAAAA” and sequence Q = “AAAA,” then matching the two sequences will lead to two exact matches of length 4. Our expectation, according to Eq. (1.6) and assuming equal nucleotide frequencies, is much smaller than two:

$$\begin{aligned} m &= L_Q - 4 + 1 = 4 - 4 + 1 = 1 \\ w &= L_D - 4 + 1 = 5 - 4 + 1 = 2 \\ \mu &= mw2^{-2L} = 1 \times 2 \times 2^{-2 \times 4} = 0.0078125 \end{aligned} \quad (1.26)$$

The remaining words are organized into index tables. For a powerful BLAST server, these index tables for database sequences can be stored in memory. If the word length is 4, then an index table will contain an array of 256 elements each with a linked list. The index table for the following partial COI sequence from *Masturus lanceolatus* is shown in Table 1.7.

```
CGCTGATTTCTCAACCAACCATAAAGA
TATCGGCACCCCTTATTAGTATTTGGTCATG
AGCCGAATAGTGGAACGGCCTTAAGCCT
GCTCATTGAGCGGAGCTAACGACCTGGGG
CTCTCCTGGAGACGACCAAATTACAATGTC
ATCGTCACAGCACATGCATTGTAATAATT
TTCTTTATAGTAATACCAATTATGATCGGGGC
TTTGGAAATTGACTCATCCCTCTTATGAT
TGGGGCCCTGATATGGCCTTCCCCGGATGAA
CAATATGAGCTTTGACTATTACCCCCCT
CTTCCCTCCTCCTGCTTCTTCAGGCCTCGAA
```

```

GCAGGTGCCGGAACGGGGTGACTGTC
TACCCCTCTTTAGCCGAAATTAGCCCCACGCAGG
CGCCTCTGTTGACTTAACAATCTTTTC
CCTTCATCTGGCCGGCATCTCCTCAATTCTAGGGC
CATTAACCTTATCACAAACAATCATTA
ATATGAAACCACCTGCAATTCTCAATACCAAACCCC
CTTGTGTTGTGAGCAGTCCTCATC
ACGGCAGTACTCTTCTCTCGCTCCAGTCCTGCAGCT

```

In the second step, the query sequence (Q) is chopped into words of the same length, and frequently occurring words (e.g., if A is very frequent, then a word made of all A's is also expected to be frequent and have a high probability of being encountered by random chance) as well as words of low complexity (e.g., known sequence repeats) are discarded. The remaining words are then searched against the index table of the database sequences (Ds). For example, if the word length is set to 4, and the first tetranucleotide in Q is AAGG, then its index is  $0 \times 64 + 0 \times 16 + 2 \times 4 + 2 = 10$  (note that A, C, G, T are coded here as 0, 1, 2, 3), and we can immediately confirm its absence in D because there is no entry in the linked list indexed by 10 (Table 1.7). Similarly, if the search word is TTTG, then its index is  $3 \times 64 + 3 \times 16 + 3 \times 4 + 2 = 254$ , and we immediately know that it occurs at positions 51, 174, 219, 292, and 537 (these numbers, generated by computer, are 0-based, i.e., the first nucleotide is at position 0). Each match is a seed to be extended in both directions.

Third, the length of the consecutive exact matches is checked against a cutoff score. For example, one can use Eq. (1.17). Alternatively, the bit-score (S) can be used as a critical cutoff value:

$$S_{\text{critical}} = \log_2(mw) - \log_2(E) = \log_2(mw/E) \quad (1.27)$$

where  $m$  and  $w$  are effective query and database length, respectively, and the E-value is taken from user input (see the previous section for more details). If an observed  $S$  value ( $S_{\text{obs}}$ ), calculated according to Eq. (1.9), is greater than  $S_{\text{critical}}$ , then the match between query and the database sequences will be extended by the computation-intensive dynamic programming and eventually reported, otherwise it will be ignored.

FASTA is generally considered to be more sensitive than BLAST in homology detection. This may be attributed to the fact that BLAST starts with exact string matching, while FASTA starts with inexact string matching. The BLAST algorithm has problems in finding sequence similarities between extremely GC-biased and extremely AT-biased sequences, e.g.,

```

S: CCA CGA GGT AAA ATT . . .
T: CCG CGG GGC AAG ATC . . .

```

The two sequences are identical at the amino acid level and differ only at the third codon positions. One is AT-rich with its third codon positions all being A or T, whereas the other is GC-rich with its third codon positions all being C or G. BLAST

will fail to report the sequence similarity because it does not have an exact match to start with (unless it uses the translated amino acid sequences). In contrast, FASTA will readily identify the sequence similarity. You should apply the FASTA algorithm to these two sequences as an exercise.

Although FASTA is generally more sensitive than BLAST, I should add here that both are heuristic algorithms. They can find a solution that is quite good but not necessarily optimal. To guarantee the finding of the best match(es), one needs to use the sequence alignment method with dynamic programming (the subject of the next chapter).

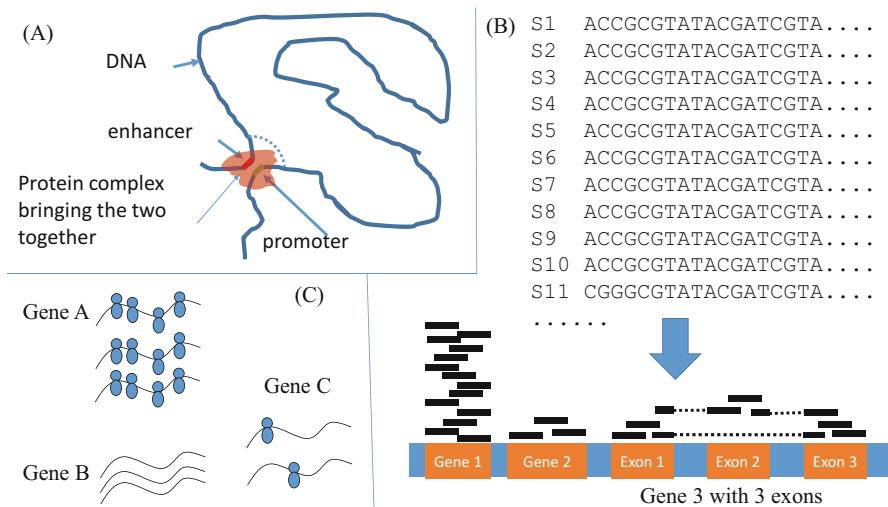
Heuristic algorithms are used often in solving problems that are computation-intensive and are of great practical value. A well-known example to illustrate the point is the problem of searching for the largest corncob in a large cornfield versus the problem of searching for a “very large” corncob, say within the top 1%. The first involves the measuring and comparison of all corncobs in the cornfield, which may be extremely laborious, although you are guaranteed to find the largest corncob. In contrast, the second problem can be easily solved by taking a random sample of corns, fitting a statistical distribution to the corns, finding the size of the corn that lies above 99% of the distribution, and using this criterion to search for the corn that is larger than the fixed criterion (or simpler, if the random sample is sufficiently large, e.g., 100 or more, picking the largest corn in the sample). The formulation and solution of the second problem belong to the heuristic approach.

The heuristic approach is used not only in biology or science but also in sociology, psychology, and economics. For example, Herbert A. Simon, Nobel laureate in economics in 1978, contributed significantly to the revelation of practical human decision-making as a process of searching for satisfactory solutions by heuristic methods rather than optimal solutions by optimality models. Choosing a religion to guide our behavior is in most cases based on the result of a heuristic approach. In our lifetime, we will never be able to do an exhaustive comparison of different religions or even different churches of the same religion, and people typically adopt one that seems to work pretty well for them and for their families. The happy ending in the movie “Pretty Woman” is a good (but rare) example of a successful application of an ultrafast heuristic approach. Indeed, one may never get married if one is determined to find the best spouse, so a heuristic approach is essential. However, I am not responsible for your hasty heuristic approach in real life.

## 4 BLAST and FASTA in Genomics, Transcriptomics, and Proteomics

### 4.1 *Genomics*

A recent exciting development in genomics is to characterize genome architecture by the Hi-C method (Lieberman-Aiden et al. 2009) to understand long-range



**Fig. 1.4** Bioinformatic applications where BLAST, FASTA, and the like serve as an essential component. (a) Characterizing long-range DNA interactions. (b) Relative gene expression (with Gene 1 more highly expressed than Gene 1) and alternative splicing, with dashed line indicating a sequence fragment mapped to two exons. (c) Ribosome profiling with mRNA from Gene A being actively translated, Gene B not translated, and Gene C translated with low activity

interactions. Enhancers can interact with promoters that are up to one million bases apart along the linear DNA. The long-range interaction, mediated by DNA-binding proteins that bring together enhancers and promoters (Fig. 1.4a), had spawned the formulation of the enhancer hub model of gene regulation (Palstra et al. 2003; Tolhuis et al. 2002). That is, the hub contains one or more enhancers and a gene with its promoter looping close to the hub will be expressed; deletion of such a hub will silence the expression of all genes that depend on their physical proximity to the hub to be expressed. The hypothesized interaction can explain why some  $\beta$ -thalassemia patients with the correct version of the  $\beta$ -globin gene does not produce the expected protein because of mutations that occurred far away from it (Kioussis et al. 1983; Taramelli et al. 1986).

The Hi-C method (Lieberman-Aiden et al. 2009) cross-links the physically juxtaposed DNA segments, cuts off the DNA strands close to the cross-linked sites, ligates the two loose ends (indicated by the dashed line in Fig. 1.4a), and generates a large number of sequence fragments with the putative enhancer and promoter at the two ends. Mapping these fragments by computer algorithms such as BLAST and FASTA (or the like) to the genomic sequences leads to an accurate representation of physical contact points in the entire genome. Such results would shed light on how many enhancers and promoters are physically in contact and how their spatial relationship change over time and among different tissues. Such knowledge would help and guide us to reposition the enhancer and the  $\beta$ -globin promoter so that the gene will be properly expressed (Deng et al. 2012, 2014b; Hou et al. 2008).

Another application of BLAST and FASTA is in genome annotation by using RNA-Seq data. Transcriptomic data analysis had revealed that most of the human genome is transcribed (Birney et al. 2007), which suggests that many genes may remain unannotated. This is highlighted by the finding that even the extensively studied phage lambda may have unannotated protein-coding genes (Liu et al. 2013). BLASTing the sequence reads from the transcriptome against other primate genomes and checking for sequence conservation will help identify new genes.

## 4.2 *Transcriptomics*

Transcriptomic data obtained from RNA-Seq can be used to identify differential gene expression and alternative splicing isoforms. Abnormal and constitutive expression, especially of certain growth factors (Bergsten et al. 2001; Ehnman et al. 2013; Pietras et al. 2003), can cause cancer (Gibbs 2000; Moffat et al. 2014; Shoemaker 2006). RNA-Seq data allow accurate characterization of gene expression and facilitate sensitive comparison between normal and abnormal cellular transcriptomes. Figure 1.4b shows the application of BLAST or FASTA or the like to map the sequence reads from a transcriptome to genes on the genome so that relative gene expression can be quantified. For example, Gene 2 is expressed much more than Gene 1 because far more sequence reads were mapped to Gene 2 than to Gene 1 (Fig. 1.4).

Alteration of spatial and temporal distributions of different splicing isoforms often leads to diseases (Poulos et al. 2011) such as Alzheimer's disease (AD) associated with abnormal splicing of the amyloid precursor protein (APP). RNA-Seq data can be readily used to detect alternative splicing isoforms and evaluate relative abundance of different isoforms. Figure 1.4b illustrates a case of alternative splicing in Gene 2 with exon 2 skipped in some mRNA, detected by BLAST or FASTA mapping sequence reads to exons.

Transcriptomic data are instrumental in discovering new genes. The observation that most of the human genome is transcribed (Birney et al. 2007) compel us to think whether many of these transcripts are in fact functional but missed by existing annotation tools. Even the extensively studied phage lambda may have unannotated protein-coding genes (Liu et al. 2013), most likely more so in humans and mouse where ribosomes are frequently found on transcripts not annotated as protein-coding sequences, with the consequent production of polypeptides (Yoon et al. 2014). One way to confirm functional importance of a transcript is through comparative genomics (Blanchette et al. 2006; Blanchette and Tompa 2002). From an evolutionary point of view, a functionally important sequence is one that is expected to be conserved among related species. One can use BLAST and FASTA suites of programs to check for sequence conservation.

### 4.3 Proteomics

BLAST and FASTA are for analyzing ribosome profiling data for characterizing translation activities of different mRNAs (Fig. 1.4c). Ribosome profiling data, traditionally from microarray (Arava et al. 2003; MacKay et al. 2004), is now almost exclusively from deep sequencing of ribosome-protected fragments (RPF, ~30 nucleotides) of mRNA (Ingolia et al. 2009, 2011). A bound ribosome protects a sequence fragment of about 30 nt. Deep sequencing these 30mers and mapping them to genes by BLAST and FASTA give us information about which gene has its mRNA actively translated and which has not (Fig. 1.4).

Translation regulation represents an important cellular mechanism capable of responding to extracellular environment. In the yeast *Saccharomyces cerevisiae*, a dozen or so genes are transcribed but not normally translated; they are translated when the surface nutrients have been depleted and their products enable yeast cells to burrow down into the culture medium to extract nutrients for growth (Gilbert et al. 2007). Ribosome profiling data can reveal the translation status of these translation regulated messages and consequently help us understand how organisms use translation regulation in response to environmental changes.

Ribosome profiling data, coupled with transcriptomic data, are expected to generate good predictions of protein production rate. Transcriptomic and ribosome profiling data provide information on mRNA abundance and translation efficiency, respectively. If genes A and B have mRNA abundance values  $N_A$  and  $N_B$ , respectively, from transcriptomic data, and their translation rates are  $R_A$  and  $R_B$ , respectively, from ribosome profiling data, then their relative protein production rate is  $N_A * R_A$  and  $N_B * R_B$ , respectively. Differences between such predicted protein abundance and observed protein abundance can be used to measure protein degradation rate. Such prediction should be facilitated by obtaining transcriptomic and proteomic data in the same experiment (Smircich et al. 2015), ideally from a single cell (Heath et al. 2016; Saadatpour et al. 2015; Wu and Tzanakakis 2013).

## 5 BLAST and FASTA in Biomedical Research and Drug Discovery

### 5.1 Human Genetic Disorders

BLAST and FASTA are the crucial tools for detecting disease-causing mutations in patients. Once we obtain a set of genomes from patients with the same inherited disease (Group1), and another set from genomes from normal individuals of the same population, ideally gender- and age-matched (Group2), the first task is to check whether Group1 individuals share the same DNA modifications relative to Group2 individuals, and BLAST and FASTA are powerful in screening the two sets of genomes for such differences. Such genome-wide scanning is often essential to

relate phenotype to genotype. For example, the lactase persistence (LP) phenotype (i.e., the trait of producing lactase all throughout adulthood) has long been suspected to be due to changes in gene regulation. We all have the right copy of the lactase gene (*LCT*) because it functions well before weaning. However, the gene has been turned off in some people but not in others. It turns out that most mutations that give rise to LP phenotype occurred about 14,000 nt upstream of the *LCT* gene, in intron 13 of the *MCM6* gene (reviewed in Segurel and Bon 2017). Genetic diseases associated with overproduction or underproduction of proteins may also be associated with mutations that are far from the gene coding for the protein. Some β-thalassemia patients have the correct version of the β-globin gene but do not produce the expected protein, and this may well be due to mutations that occurred far away from the β-globin gene (Kioussis et al. 1983; Taramelli et al. 1986).

Human genomes have many duplicated genes that offer functional redundancy in some cases. For example, the function of paralogous genes *USP4* and *USP15* in mice is partially redundant (Vlasschaert et al. 2015). If a deleterious mutation is identified to be a loss-of-function mutation, then BLAST and FASTA can help identify a paralogous gene that can compensate for the mutation. For example, human adrenoleukodystrophy (ALD) is caused by partial deletion of the 10-exon gene *ABCD1* resulting in the accumulation of very long chain fatty acids (Krasemann et al. 1996), which suggests not only diet limitation of very long chain fatty acids (VLCFA) in disease management but also activation of alternative metabolic pathways for VLCFA through regulating another gene involved in fatty acid metabolism (*ABCD2*) and suppression of the activity of elongase involved in generating VLCFA (Morita et al. 2011). Another example of activating alternative biological pathways or genes with partial functional redundancy involves sickle cell anemia (Pauling et al. 1949) caused by a single amino acid replacement in human β-globin gene (Ingram 1956, 1957). Fetal hemoglobin gene (*HbF*) is a promising drug target because HbF reduces hemoglobin polymerization and clumping. A drug that could revive the silenced *HbF* would alleviate the symptoms of sickle cell anemia and thalassemia in adults (Kutlar 2007; Steinberg and Rodgers 2001). BLAST and FASTA can efficiently identify the paralogous gene that are the most similar to the mutated gene.

## 5.2 *BLAST and FASTA Facilitate the Development of Drugs Against Human Pathogens*

The first step in identifying drug targets in pathogens is to identify essential genes for the survival and reproduction of pathogens in humans. BLAST and FASTA have often been used to identify weak links in a pathogen's biochemical pathways. For example, genes essential for nucleotide synthesis are typically highly conserved. BLASTing such genes from several protest genomes against the genome of protest pathogen *Trypanosoma brucei* will show that *T. brucei* has lost functional genes for

de novo synthesis of ATP, GTP, and TTP but maintains functional genes for limited capacity for de novo synthesis of CTP, presumably because CTP generally has much lower centration than the other three nucleotides in the cell and cannot be reliably obtained through salvage. This points to CTP synthesis pathway as a drug target. Indeed, inhibiting CTP synthesis arrests the growth and replication of the pathogen (Hofer et al. 2001). Essential genes are often highly conserved and can be revealed by genomic comparisons between pathogens and their phylogenetic relatives.

Essential cellular processes are often functionally redundant, and understanding such functional redundancy is crucial in developing effective drugs against pathogens. Because functional redundancy is often served by paralogous genes (through gene duplication and subsequent neofunctionalization or subfunctionalization), BLAST and FASTA are efficient tools to identify such paralogous genes. In *Mycobacterium tuberculosis*, arabinofuranosyltransferases Mt-EmbA and Mt-EmbB contribute to the synthesis of cell wall mycolyl-arabinogalactan-peptidoglycan complex and are targeted by the drug ethambutol. Bioinformatic analyses revealed another arabinofuranosyltransferase, Mt-AftA, which is not inhibited by ethambutol and consequently would serve as a drug target (Alderwick et al. 2006). A combination of drugs against all three arabinofuranosyltransferases will not only be more effective against the pathogen but also reduce the chance of the pathogen developing drug resistance. Activating alternative biological pathways to satisfy the need of growth and survival has been known in bacterial species since the discovery of the *lac* operon and the glucose/lactose genetic switch (Jacob and Monod 1961), and a drug cannot be effective against a pathogen or a cancer cell unless we know how cells do things with alternative pathways that can be activated in response to the drug.

The second step in developing drugs against pathogens is to check if such essential genes have homologues in the host. If they do, then inhibiting such essential genes in the pathogen may have adverse effect on the function of the host homologue, and we consequently need to perform sequence comparisons between the pathogen and host homologues to identify unique part in the pathogen homologue to assist in the design of pathogen-specific drugs. Bioinformatic analysis revealed a glutamate transport system that is present in the pathogen *Clostridium perfringens* but absent in mammals and birds (Bhatia et al. 2014). Drugs developed against such a transport system will protect not only humans but also domesticated mammals and fowls. In the human parasite *Giardia intestinalis*, the phosphoinositide-3 kinase (PI3K) signaling pathways are essential and could serve as a drug target. However, the PI3K pathway is also essential in many eukaryotes so it is important to identify what is unique in the PI3K homologues (Gipi3k1 and Gipi3k2) in *G. intestinalis* relative to mammals. Sequence comparisons revealed a unique insertion only in the parasite that can serve as a pathogen-specific drug target (Cox et al. 2006). The same approach is used in targeting *Pseudomonas aeruginosa* (Fernandez-Pinar et al. 2015). Similarly, in developing anti-HIV-1 drugs, one can target genes involved in reverse transcription and protease digestion of its translated polyprotein because these processes not only are essential for viral survival and transmission but also have no close homologues in humans so their inhibition should have minimal side effect on humans.

The third step in drug development is minimize the chance of pathogen developing drug resistance. Bacterial resistance to penicillin became known soon after its discovery in 1928 and its regular medical applications in 1940 (Abraham and Chain 1940; Abraham et al. 1941). Such resistance can also develop quickly in eukaryotic pathogens, e.g., in malaria parasite *Plasmodium falciparum*, against the most effective anti-malaria drug artemisinin, soon after the large-scale application of artemisinin in Asian countries (Noedl et al. 2008, 2009, 2010). Drug resistance often renders a costly developed drug useless, contributing to the high cost of drug development (David et al. 2009; Drews and Ryser 1997). The rapid development of drug resistance in HIV-1 (Smyth et al. 2012, 2014) highlights the importance of understanding drug resistance.

To reduce the probability of pathogen developing drug resistance, it is important for the drug to target at specific pathogen and not its phylogenetic relatives that are not pathogenic. For this reason, pathogenicity islands on the genome that are unique in pathogenic bacteria but not in their nonpathogenic relatives have increasingly become the preferred source of drug targets (Gal-Mor and Finlay 2006; Hacker et al. 1997; Hacker and Kaper 2000; Trudel et al. 2016). Mutual BLASTing between virulent and avirulent strains can help us quickly identify such pathogenic islands.

Genomics has also contributed to understanding drug actions. The venom protein PcfK1 of spider *Psalmopoeus cambridgei* was able to inhibit the growth of *Plasmodium falciparum*, but the mechanism was unknown. One may BLAST PcfK1 against protein databases and will find sequence homology between PcfK1 and the protein substrate of *P. falciparum* enzyme PfSUB1, leading to the hypothesis that PcfK1 is an antagonist of PfSUB1. Subsequent docking prediction and in vitro experiments confirm this hypothesis, pointing to PfSUB1 as a drug target (Bastianelli et al. 2011).

## Appendix: Being Colorful Is Not Enough—How Are Stop Codons Decoded?

In the old and not-so-good days of 1965, three stop codons were known to exist. At that time, almost all molecular biologists study bacteriophages ( $T4$  and  $\lambda$ ) and their hosts (typically *E. coli*). Sometimes a phage would have a mutation from a sense codon to a stop codon, leading to a truncated protein. Such a phage can only grow in certain *E. coli* strains containing a suppressor mutation (which is typically a tRNA with a mutated anticodon that can base-pair with a stop codon). Such strains were then called suppressor strains. The first set of nonsense mutations were discovered and isolated by Richard Epstein and Charles Steinberg. The resulting stop codon from these nonsense mutations was named after their friend Harris Bernstein whose last name means “amber” in German. The associated suppressor was termed amber suppressor. Any phage with a nonsense mutation that can grow only in this particular set of suppressor strains was said to harbor an amber mutation. The observation that

an amber stop codon could be suppressed (i.e., recognized as a sense codon) in suppressor strains but not in other strains suggests some ambiguity in the meaning of the stop codon, i.e., the stop codon is interpreted differently between the suppressor strains and non-suppressor strains.

Phages with another set of nonsense mutations can grow only in a different set of *E. coli* strains, and the stop codon resulting from these nonsense mutations was named ochre. The associated suppressor is termed ochre suppressor. The third stop codon was named opal. In short, there are three disjoint sets of suppressor strains corresponding to three stop codons, but biologists did not know which stop codon is associated with amber, ochre, or opal.

Martin Weigert and Alan Garen (1965) studied one particular amino acid site in the alkaline phosphatase gene in *E. coli*. This site is occupied by tryptophan coded by UGG. An amber mutation (i.e., a nonsense mutation that can be suppressed by an amber suppressor) occurred at this site. In several revertants (in which the amber mutation had reversed to a sense codon), the original amino acid site was found to be occupied by seven different amino acids, Glu, Lys, Leu, Gln, Ser, Trp, and Tyr (Fig. 1.5). Can we now match the amber codon to one of the stop codons?

Because mutation is rare, we may assume that the revertants differ from the amber mutant by a single nucleotide. These candidate codons are shown in Fig. 1.6 for each of the three stop codons. UGA is the least likely candidate because it cannot generate a codon for Glu, Lys, Gln, and Tyr by a single nucleotide change. UAA can also be excluded because (1) it cannot generate a Trp UGG codon by a single nucleotide change, and (2) it would need to have a double mutation from the original UGG codon. In contrast, UAG can mutate to one of the codons for all seven amino acids, and it can result from the original UGG through a single nucleotide replacement. So the amber codon is UAG. (I have made things easier because Weigert and Garen were not so sure about the sense codons which were only partially decoded in 1965).

This example illustrates several decision-making principles that are useful to us. The first is the KISS principle (“Keep it simple, stupid”). That is, to use simplifying assumptions to make decision-making easier. For example, if we assume that double mutation (i.e., mutations at two sites of the codon) is so rare as to be

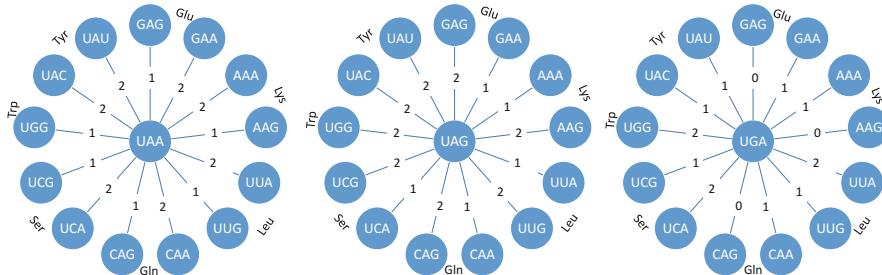
**Fig. 1.5** Data for matching the amber mutation to one of the three possible stop codons

```

... NNN UGG NNN ...
                ↓ mutation
... NNN END NNN ...
                ↓
Revertants:
... NNN Glu NNN ...
... NNN Lys NNN ...
... NNN Leu NNN ...
... NNN Gln NNN ...
... NNN Ser NNN ...
... NNN Trp NNN ...
... NNN Tyr NNN ...

```

The diagram illustrates the mutation process. It starts with a sequence of three nucleotides (NNN) followed by a tryptophan codon (UGG) and another three nucleotides (NNN), ending with a period. A blue arrow labeled "mutation" points down to the next line, where the same sequence is shown but the tryptophan codon (UGG) has been replaced by a stop codon (END). Another blue arrow points down to the "Revertants:" section, which lists seven different amino acids (Glu, Lys, Leu, Gln, Ser, Trp, and Tyr) each preceded by the three nucleotides (NNN) and followed by three more nucleotides (NNN), separated by periods.



**Fig. 1.6** Bioinformatic analysis to match the amber codon to UAG. The numbers indicate the number of matched nucleotides because a sense codon and a stop codon. Amino acids are shown next to its codons

negligible, then we can immediately reject UAA and UGA because both require double mutations to generate codons for the seven amino acids. In terms of model selection, we state that only the UAG model fits the data given the model condition (the assumption of no double mutation).

The second is the parsimony principle. We have three alternative hypotheses corresponding to the three stop codons. The UAG hypothesis needs a minimum of seven single point mutations to generate the seven revertants each with a different amino acid. It also needs a point mutation to connect the original UGG codon. The UGA hypothesis needs at least two point mutations to change to a Glu, Lys, Gln, or Try codon, and one mutation to each of the other three amino acids. This comes to a minimum number of 11 point mutations, plus a point mutation to connect the original UGG codon. The UAA hypothesis needs to have two point mutations to a UGG revertant as well as two point mutations from the original UGG codon, leading to a minimum of ten point mutations. The UAG hypothesis is therefore the most parsimonious and is chosen over the other two alternatives.

The third is the likelihood principle, but we will not go there without first learning substitution models which is covered in a later chapter.

# Chapter 2

## Sequence Alignment

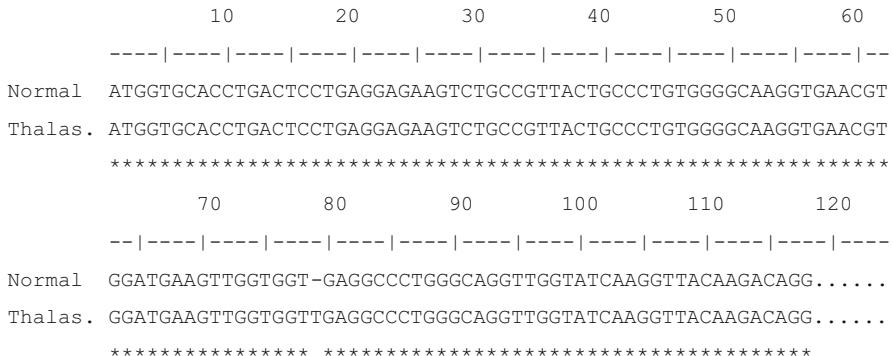


### 1 Introduction

Sequence alignment is not only the essential first step in molecular phylogenetics, quantification of substitution patterns, and dating of speciation and gene duplication events but also a powerful tool for identifying mutations leading to genetic diseases. For example, aligning the  $\beta$ -hemoglobin gene sequence from one type of  $\beta$ -thalassemia against the normal  $\beta$ -hemoglobin gene immediately reveals an insertion of T at site 79 (Fig. 2.1) which creates an inframe stop codon and results in a truncated  $\beta$ -hemoglobin protein. When the  $\beta$ -hemoglobin locus is heterozygous with one mutant and one normal gene, the carrier is said to have  $\beta$ -thalassemia minor and can be easily detected by SDS gel electrophoresis because the mutant  $\beta$ -hemoglobin, being much shorter than the normal, would migrate much faster on the gel under an electric field.

Sequence alignment aims to maximize the number of matches and minimizes mismatches and gaps. Operationally, the best alignment is one with the highest alignment score which can be calculated given an alignment and a scoring scheme. A scoring scheme has two components. The first is a match/mismatch matrix which specifies how many points to gain from a match or to deduct from a mismatch. The second is gap penalty which specifies how gaps are penalized by deducting points.

There are two popularly used gap penalty schemes in sequence alignment, the constant gap penalty and the affine function gap penalty. The former is mainly for classroom illustration of the dynamic programming algorithm for sequence alignment. It does not distinguish between gap open and gap extension, and the total cost of an alignment is  $N \times G$ , where  $N$  is the total gap length and  $G$  is penalty for one gap against a single nucleotide or amino acid. The affine function gap penalty distinguishes between gap open and gap extension, with greater penalty for gap open than for gap extension. The total gap penalty for an alignment is  $N_{GO} \times GO + N_{GE} \times GE$ , where  $N_{GO}$  and  $N_{GE}$  are the number of gap open and gap extension, respectively, and  $GO$  and  $GE$  are penalties for one gap open and one gap extension, respectively.



**Fig. 2.1** Alignment between the normal and mutant  $\beta$ -hemoglobin gene sequences

The default GO and GE in BLAST (Altschul et al. 1990) are  $-5$  and  $-2$ , respectively. Alignment algorithm using the affine function gap penalty is implemented in essentially all practical computational tools where local or global sequence alignment is needed.

Two slightly different formulations of the affine function gap penalty exist, in both literature and programming source codes. For an indel with a gap length  $N$ , one formulation specifies the gap penalty as equal to  $GO + N \times GE$  (e.g., Gusfield 1997; Pevzner 2000). The BLAST suite of programs uses this formulation as well. The other formulation specifies the gap cost to be equal to  $GO + (N-1) GE$  (e.g., Higgins and Attwood 2005). The latter is theoretically more preferable than the former because it is reduced to constant gap penalty when  $GO = GE$ . It is the second formulation that I will use here in detailing the alignment with the affine function gap penalty.

This chapter covers (1) pairwise global and local alignment by dynamic programming with different scoring schemes, from the simplest scoring scheme with two-valued match/mismatch scores and constant gap penalties to more useful scoring scheme with match-mismatch matrices and affine function gap penalties, (2) detailed derivation of PAM and BLOSUM matrices, (3) profile alignment between one sequence and a set of aligned sequences which is essential for practical implementation of multiple sequence alignment, and (4) multiple alignment that is reduced to pairwise alignment and profile alignment by using a guide tree. Most textbooks on bioinformatics omit the dynamic programming algorithm using affine function gap penalty (Gotoh 1982), and no textbook I know of includes any detailed explanation of profile alignment.

Dynamic programming algorithms constitute a general class of algorithms not only used in sequence alignment but also in many other applications. For example, the Viterbi algorithm and the forward algorithm used in hidden Markov models (HMMs) are also dynamic programming algorithms, so are Fitch and Sankoff algorithms for maximum parsimony (MP) and the pruning algorithm for maximum likelihood (ML) reconstruction of phylogenetic trees. We will cover HMM, MP, and

ML in great detail and illustrate with numerical examples latter. Learning the dynamic programming algorithms used in sequence alignment paves the way for more advanced applications in latter chapters.

Dynamic programming algorithms are too slow to be useful in multiple sequence alignment, and nearly all practical multiple sequence alignment would use a tree-guided progressive alignment to reduce the multiple alignment problem to a pairwise alignment problem. This creates a chicken-egg problem. To obtain a good alignment, one needs a good guide tree, but to get a good guide tree, one needs to have a good alignment. One way to get out of this problem is to build a guide tree based on pairwise alignment only using the PhyPA (phylogenetics with pairwise alignment) method (Xia 2016).

There is another chicken-egg problem associated with sequence alignment, and that is the match-mismatch (MM) matrix, e.g., BLOSUM (Henikoff and Henikoff 1992), derived from an empirical substitution matrix to score an alignment. If we are to have a good alignment for highly diverged sequences, then we need a good MM matrix such as BLOSUM45. However, we cannot have a good BLOSUM45 if we do not have a set of diverged but well-aligned sequences. The BLOSUM approach alleviates this problem by excluding alignment segments with indels. In contrast, PAM matrices (Dayhoff et al. 1978) do not have this chicken-egg problem because they are all derived from a transition probability matrix based on very similar sequences whose alignment is not ambiguous. This chapter also includes detailed derivation of PAM and BLOSUM matrices.

Match-mismatch matrices such as PAM and BLOSUM matrices have been given other names, such as scoring matrices and substitution matrices. To avoid confusion, I will refer to these matrices as match-mismatch matrices or simply as MM matrices in this chapter. I reserve the use of “scoring matrix” for the matrix (or matrices) used for keeping alignment scores during the execution of the dynamic programming algorithm.

## 2 Dynamic Programming for Pairwise Alignment

Given two strings S ( $=s_1s_2\dots s_n$ ) and T ( $=t_1t_2\dots t_m$ ), a pairwise alignment of S and T is defined as an ordered set of pairings of  $(s_i, t_j)$  and of gaps  $(s_i, -)$  and  $(-, t_j)$ , with the constraint that the alignment is reduced to the two original strings when all gaps in the alignment are deleted. A prefix of S, specified here as  $S_i$ , is a substring of S equal to  $s_1s_2\dots s_i$ , where  $i \leq n$ .

An optimal alignment is operationally defined as the pairwise alignment with the highest alignment score for a given scoring scheme. For this reason, an optimal alignment is meaningless without the specification of the scoring scheme.

Alignment by dynamic programming guarantees that the resulting alignment is the optimal alignment or one of the equally optimal alignments. We will first illustrate the Needleman-Wunsch algorithm for global pairwise alignment (Needleman and Wunsch 1970) and then the Smith-Waterman algorithm for local

pairwise alignment (Smith and Waterman 1981). Local sequence alignment is for searching local similarities between sequences, e.g., homeobox genes which are not similar globally but all share a very similar homeodomain motif.

Here we will first learn a simple dynamic programming algorithm for pairwise alignment using a simple scoring scheme with constant gap penalty. This is then extended in two ways, first by introducing a similarity matrix to replace match and mismatch scores and second by introducing the affine function to better approximate the origin of the insertion and deletion during sequence evolution. My experience is that an average student can understand pairwise alignment with constant gap penalty, but only a very good student can understand the two extensions.

## 2.1 Pairwise Alignment with Constant Gap Penalty

### 2.1.1 Global Alignment

Suppose we want to align two sequences S and T with  $S = \text{ACGT}$  and  $T = \text{ACGGCT}$ . A simple scoring scheme is used with a constant gap penalty ( $G$ ) of  $-2$ , a match score ( $s_{ii}$ ) of  $2$ , and a mismatch score ( $s_{ij}$ ) of  $-1$ . Global alignment with the dynamic programming approach is illustrated numerically in Fig. 2.2. One sequence of the two sequences occupies the top row and will be referred to hereafter as the row sequence (sequence S in our example). The other sequence occupies the first column and will be referred to hereafter as the column sequence (sequence T in our example). Based on these two sequences, two matrices are computed. The first is the scoring matrix (SM) to obtain the alignment score, with the dimensions  $(n + 1, m + 1)$ . A value in row  $i$  and column  $j$  in the scoring matrix ( $\text{SM}_{i,j}$ ) is the alignment score between prefixes  $S_j$  and  $T_i$ . The second is the backtrack matrix needed to obtain the actual alignment,

**Fig. 2.2** Aligning sequences S (top row) and T (left column) by dynamic programming, with the score and the backtrack matrices superimposed. The scoring scheme has a match score of 2, mismatch score of  $-1$ , and constant gap penalty of  $-2$

		A	C	G	T
	0	-2	-4	-6	-8
A	-2	↖ 2	← 0	← -2	← -4
C	-4	↑ 0	↖ 4	← 2	← 0
G	-6	↑ -2	↑ 2	↖ 6	← 4
G	-8	↑ -4	↑ 0	↖ 4	↖ 5
C	-10	↑ -6	↖ -2	↑ 2	↑ 3
T	-12	↑ -8	↑ -4	↑ 0	↖ 4

with the dimensions  $(n, m)$ . In Fig. 2.2, the two matrices are superimposed, with the scoring matrix being the numbers and the backtrack matrix being made of arrows. The backtrack matrix is sometimes called the traceback matrix.

The first row ( $SM_{0,j}$ ) and the first column ( $SM_{i,0}$ ) of the scoring matrix are filled with  $i \times G$  (where  $i = 0, 1, \dots, n$ ) and  $j \times G$  (where  $j = 0, 1, \dots, m$ ), respectively. They represent consecutive insertion of gaps. For example,  $SM_{0,4} = -8$  (Fig. 2.2) implies the alignment of S against four consecutive gaps. Similarly,  $SM_{6,0} = -12$  (Fig. 2.2) implies the alignment of T with six consecutive gaps.

All other  $SM_{i,j}$  values are computed as

$$SM_{i,j} = \max(DIAGONAL, LEFT, UP), \text{ where}$$

$$DIAG = SM_{i-1,j-1} + IF(T_i = S_j, s_{ii}, s_{ij}), \text{ where } DIAG \text{ is short for diagonal.}$$

$$LEFT = SM_{i,j-1} + G$$

$$UP = SM_{i-1,j} + G$$

The IF function above takes the value of  $s_{ii}$  if  $T_i = S_j$  or  $s_{ij}$  if  $T_i \neq S_j$ . For  $SM_{1,1}$ ,

$$DIAG = SM_{0,0} + IF(T_1 = S_1, s_{ii}, s_{ij}) = 0 + 2 = 2$$

$$LEFT = SM_{1,0} + G = -2 + (-2) = -4$$

$$UP = SM_{0,1} + G = -2 + (-2) = -4$$

The maximum of these three values is  $DIAG$ , i.e., 2, so  $SM_{1,1} = 2$ , and we put an up-left arrow in the cell (Fig. 2.2). If  $LEFT$  (or  $UP$ ) happened to be the maximum of the three, we would have put a left-pointing (or up-pointing) arrow in the corresponding cell in the backtrack matrix. Keep in mind that  $SM_{i,j}$  is the optimal alignment score between prefixes  $T_i$  and  $S_j$ . For example,  $SM_{1,1} = 2$  is the optimal alignment score for aligning T1 and S1.

The computation is from left to right and from top to bottom.  $SM_{1,2}$  is maximum of the following three values:

$$DIAG = -2 - 1 = -3$$

$$LEFT = 2 - 2 = 0$$

$$UP = -4 - 2 = -6$$

These three alignment scores represent the following three alternative alignments with the alignment scores shown below:

DIAG	LEFT	UP
AC	AC	AC-
-A	A-	--A
-3	0	-6

The maximum of the three values is  $LEFT (= 0)$ , and the corresponding cell in the backtrack matrix is filled with a left-pointing arrow. We continue the computation to the bottom-right cell to obtain  $SM_{6,4} = 4$ . This is the optimal alignment score for

(a)	(b)
654321	654321
ACG--T	AC-G-T
ACGGCT	ACGGCT

**Fig. 2.3** The protocol of obtaining the sequence alignment by following the backtrack matrix. The numbers in the first row show the order of obtaining the alignment site by site from the last to the first (i.e., backtracking)

The aligned sequences are obtained directly from the backtrack matrix without any reference to the scoring matrix. We start from the bottom-right cell and follow the direction of the arrow in the cell. The up-left arrow in the bottom-right cell means that we should stack the two corresponding nucleotides (T and T) in the row and column sequences (Fig. 2.3a). Note that you would be stacking the two corresponding nucleotides regardless of whether they are the same or different as long as an up-left arrow is in the cell. A left-pointing or up-pointing arrow in the cell means a gap in the column sequence or row sequence, respectively.

The up-left arrow in the bottom-right cell leads us to the cell containing an up-pointing arrow, meaning a gap in the row sequence S, i.e., we stack a gap character “–” over the corresponding nucleotide (C) in the column sequence T (Fig. 2.3a). This up-pointing arrow brings us to the special cell with two arrows, one pointing up-left and the other up (Fig. 2.2). This leads to alternative construction of the sequence alignment. If we choose the up-pointing arrow, we will stack a gap character over the corresponding nucleotide (G) in the column sequence and proceed to the cell with a value of 6 and an up-left arrow. This ultimately leads to the sequence alignment in Fig. 2.3a. Alternatively, we may choose to follow the up-left arrow and stack the two nucleotides (G in both sequences) as shown in Fig. 2.3b. This ultimately leads to the alternative sequence alignment in Fig. 2.3b.

Both alignments in Fig. 2.3 have four matches, two gaps, and zero mismatch. So the alignment score is  $4 \times s_{ii} + 2 \times G + 0 \times s_{ij} = 4$ , which we already know after completing the scoring matrix whose bottom-right cell contains the alignment score.

When sequences are long, there might be many equally optimal alignments, and few computer programs would try to find and output all of them. Instead, only one path is followed, leading to the output of only one of the potentially many equally optimal alignments.

Dynamic programming guarantees that the resulting alignment is optimal given the scoring scheme. In other words, there is no alignment that can have an alignment score greater than 4 given the two sequences and the scoring scheme of  $s_{ii} = 2$ ,  $s_{ij} = -1$ , and  $G = -2$ . However, an optimal alignment may change when the scoring

**Fig. 2.4** Illustration of the dependence of optimal alignment on scoring scheme. Score 1 and Score 2 in the bottom table refer to alignment scores obtained with scoring scheme 1 and scoring scheme 2, respectively

	Alignment 1: ACCCAGGGCTTA ACCCGGGCTTAG					
	Alignment 2: ACCCAGGGCTTA- ACCC-GGGCTTAG					
	Scoring scheme 1: M = 2, MM = 0, G = -5					
	Scoring scheme 2: M = 2, MM = -1, G = -1					
Alignment	Match	Mismatch	Gap	Score1	Score2	
1	7	5	0	14	9	
2	11	0	2	12	20	

scheme is changed. This is illustrated in Fig. 2.4, where the use of scoring scheme 1 would result in Alignment 1 (with alignment score = 14) better than Alignment 2 (with alignment score = 12), but the use of scoring scheme 2 would lead to the opposite, with Alignment 2 (alignment score = 20) much better than Alignment 1 (alignment score = 9).

Because there is no objective way of choosing the right scoring scheme, it is therefore important to keep in mind that sequence alignment is a method of data exploration instead of an analytical method that will lead to a single best solution. For this reason, nearly all computer programs for sequence alignment allow the user to try various scoring schemes and post-alignment manual editing. The rule of thumb is that higher penalties are given to rarer changes. For example, if indel (insertion/deletion) events are rarer than nucleotide substitutions leading to mismatches, then gaps should be penalized more than mismatches. Similarly, transitions occur more frequently and should be penalized less than transversions.

### 2.1.2 Local Alignment

The Smith-Waterman algorithm for local sequence alignment (Smith and Waterman 1981) is a variation of the Needleman-Wunsch algorithm for global alignment presented above but with three major differences. First, the first row and the first column of the scoring matrix are filled with zero instead of  $i \times G$ . Second, whenever the cell value becomes negative (i.e., the maximum of the three values is smaller than 0), the cell value is set to 0. Thus, when two sequences have a short but perfect local match and little similarity elsewhere, the alignment score of the short but perfect match is not affected by the low similarity elsewhere. Third, because a local alignment can end anywhere in the matrix, we will not trace back from the cell in the bottom-right corner of the scoring matrix. Instead, we find the maximal score in the matrix and trace back from that point until we reach a cell with a value of 0, which indicates the start of the local alignment.

In Chap. 1 we have already covered two widely used heuristic methods for local alignment, i.e., BLAST and FASTA. In a later chapter, we will learn Gibbs sampler which is another method for searching local similarities and local alignment.

## 2.2 Match-Mismatch Matrix

Each scoring scheme has two components, the match and mismatch scores and the gap penalty. The simple scoring scheme that we have used so far, specified by only three values, constant gap penalty and match and mismatch scores, needs to be extended in two ways to approximate biological reality. The constant gap penalty will be replaced by affine function gap penalty which distinguish gap open ( $G_O$ ) from gap extension ( $G_E$ ), and the two-valued match and mismatch scores will be replaced by a match-mismatch (MM) matrix (or a series of matrices for sequences of difference divergence). This section details the derivation of MM matrices for nucleotide and amino acids used in practical sequence alignment. The emphasis is on understanding the rationale and mathematical details underlying the PAM and BLOSUM matrices. The next section deals with alignment algorithms using affine function gap penalty. These two sections demand significant effort in your part. One incentive to learn the derivation of MM matrices is that you will gain an excellent perspective into molecular phylogenetics, especially in understanding substitution models and evolutionary distances, although the benefit may not materialize until you get to the part of the book dealing with molecular phylogenetics.

There are three reasons for replacing the two-valued match and mismatch scores by a matrix. For nucleotide sequences, transitions (substitution of a purine by another purine or a pyrimidine by another pyrimidine, i.e.,  $A \leftrightarrow G$  or  $C \leftrightarrow T$ ) generally occur more frequently than transversions (substitution of a purine by a pyrimidine or vice versa, i.e.,  $\{A,G\} \leftrightarrow \{C,T\}$ ). This suggests that we should not treat transitional differences and transversional differences with the same mismatch score. Instead, transitions should be penalized less than transversions. Second, nucleotide frequencies often differ. If A is more frequent than C, then a C/C match is a stronger indication of homology than an A/A match and therefore should be given a higher match score than the latter.

The same line of reasoning also applies to amino acid sequences. Some amino acids (e.g., Ala, Gly) are typically more frequent than others (e.g., Trp, Cys), so the match score for Ala:Ala should be smaller than that for Trp:Trp. Also, substitutions between some amino acids (e.g., between Lys and Arg or between Glu and Asp) replace each other quite frequently while others (e.g., between Trp and Gly) rarely, so a mismatch score between Lys and Arg should be smaller than that between Trp and Gly. Thus, the ability to understand currently used MM matrices and, in particular, to generate good MM matrices for your own particular data is an essential skill for any bioinformatician.

The second reason for extending the two-valued match and mismatch scores by a matrix is that the likelihood of amino acids replacing each other changes with time. With a very short time, Gly is very unlikely to be replaced by Arg or Trp, but the probability increases with increasing length of time, so different match/mismatch scores should be used for sequences of different divergence. Readers who have done practical amino acid sequence alignment would have already encountered Dayhoff (Dayhoff et al. 1978) or BLOSUM (Henikoff and Henikoff 1992) matrices that aim to accommodate differences in amino acid frequencies and in substitution rates.

Each type is represented by a series of matrices (e.g., PAM1, PAM100, PAM120, . . . , PAM250 or BLOSUM90, BLOSUM80, . . . , BLOSUM45). How to know which matrix is good for aligning your sequences? Should you use PAM1 or PAM100 or PAM250 for aligning your amino acid sequences? Generally, a reasonable MM matrix should give you a positive alignment score. If your amino acid sequences differ by only 1% of the sites, then PAM1 will give you a positive alignment score. However, if your amino acid sequences are highly diverged, then even using PAM100 may generate a negative alignment score even if the two sequences still have significant signal of homology, suggesting that PAM250 or any MM matrix for more divergent sequences should be used (otherwise you would arrive at a wrong conclusion that the two sequences have no homology).

The third reason for extending the two-valued match and mismatch scores is that there are often ambiguous bases in input sequences, e.g., R for A or G and Y for C or T. An A/R pair is neither a strict match nor a strict mismatch but has a probability of 0.5 being a match and a probability of 0.5 being a transition. The simple scoring scheme we have used cannot handle these problems.

### 2.2.1 Match-Mismatch Matrix for Nucleotide Sequences

One example of a similarity matrix is the “transition bias matrix” (Table 2.1). The meaning of the top  $4 \times 4$  matrix (bolded values in Table 2.1) is easy to understand. The first four diagonal values of 30 are equivalent to the match score. A mismatch score involving a transversion or a transition is  $-30$  or  $0$ , respectively, because transitions in general occur much more frequently than transversions and consequently penalized less (Table 2.1). The rest of the matrix (Table 2.1) involves ambiguous codes specified in Table 2.2, according to the Nomenclature Committee of the International Union of Biochemistry (1985).

The coding scheme is often referred to as the IUB code or IUB notation. For example, R represents either A or G, so an A/R pair has a probability of 0.5 being an A/A match and a probability of 0.5 being an A/G transition. The corresponding score ( $= 15$  in Table 2.1) is consequently the average score of (1) a perfect match with a match score of 30 and (2) a transition with a score of 0. In contrast, Y stands for either C or T/U, and an A/Y pair is always a transversion, with a score of  $-30$  (Table 2.1).

What is the justification of a match score of 30, a transition score of 0, and a transversion score of  $-30$ ? In the default BLAST, a match score is 1 and a mismatch score is  $-3$ . What is the justification of such scores and their relative magnitude? There is a mathematical approach to derive MM matrices based on Markov chain models, which we will cover in the chapter on substitution models. Here we will first follow the empirical approach used for deriving MM matrices for amino acid sequences developed by Dayhoff and her colleagues (1978), although the approach is equally applicable to nucleotide sequences. The BLOSUM approach (Henikoff and Henikoff 1992) will follow. Keep in mind that PAM and BLOSUM matrices are typically associated with amino acid substitutions and protein sequence alignment.

**Table 2.1** A similarity matrix accommodating the transition bias frequently observed in nucleotide substitutions

**Table 2.2** IUB codes of nucleotides

Code	Meaning	Complement
A	A	T
C	C	G
G	G	C
T/U	T	A
M	A or C	K
R	A or G	Y
W	A or T	W
S	C or G	S
Y	C or T	R
K	G or T	M
V	A or C or G	B
H	A or C or T	D
D	A or G or T	H
B	C or G or T	V
X/N	G or A or T or C	X
-	Gap (not G or A or T or C)	-

**Table 2.3** Empirical substitution matrix (often referred to as matrix A) from closely related nucleotide sequences with divergence smaller than 1%

	A	G	C	T	$N_i$	$\pi$	$m$
A	7335	20	10.5	12.5	7378	0.3686	0.0058
G	20	4815	10	12.5	4857.5	0.2427	0.0087
C	10.5	10	3778	39.5	3838	0.1917	0.0156
T	12.5	12.5	39.5	3878	3942.5	0.1970	0.0164

Nucleotide frequencies  $\pi$  and mutability ( $m$ , which measures the propensity of a nucleotide being replaced by others) are also shown

However, the principle can be applied to nucleotide sequences (and indeed can be more easily explained with nucleotide sequences).

Table 2.3 represents an empirical substitution matrix between non-redundant sequence pairs that differ at less than 1 site out of 100 sites. The reason for using sequence pairs of low divergence is to approximate the assumption of few multiple substitutions. A column nucleotide represents the state of one sequence and a row nucleotide the state of the other sequence. Because the sequences are from extant species and we do not know the ancestral state, the two corresponding off-diagonal elements are averaged so that the matrix is symmetrical. The large values in the diagonals means that most sequence sites between the two sequences are identical. Among sites where the two sequences differ, most are C↔T transitions, followed by A↔G transitions. Transversions are relatively less frequent than transitions. The pattern is suggestive of the TN93 model (Tamura and Nei 1993) that we will explain in the chapter on substitution models.

It is helpful to first define the notations. The values in the  $4 \times 4$  empirical substitution matrix is often referred to as  $A(i,j)$  and the number of nucleotide  $i$  ( $i = A, G, C$  or  $T$ ) as  $N_i$ ,  $N = \Sigma N_i$ . Given these, we have

$$\pi_i = \frac{N_i}{N}$$

$$m_i = \frac{\sum_{i=1, j=1, i \neq j}^4 A(i, j)}{N_i} \quad (2.1)$$

where  $m_i$  is the proportion of nucleotide  $i$  involved in observed nucleotide replacement and is often referred to as the index of mutability.

### 2.2.1.1 PAM Matrix for Nucleotide Sequences

We will follow the convention and define one PAM time unit as the time needed for one nucleotide substitution to occur in 100 nucleotide sites. The transition probability matrix ( $M_1$ , where the subscript 1 indicates one PAM unit of time) can be derived from matrix  $A$ . We only need to obtain the off-diagonal elements because the diagonal elements of a transition probability matrix are constrained by each row summing up to 1, i.e., a nucleotide can either remain the same or be replaced by another nucleotide. The off-diagonal elements of  $M$  are

$$M_1(i, j) = \frac{\lambda A(i, j)}{N_i} \quad (2.2)$$

where  $\lambda$  is obtained from the following equation, assuming that multiple substitution is negligible:

$$\lambda = \frac{0.01}{\sum_{i=1}^4 \pi_i m_i} = 0.0953 \quad (2.3)$$

Note that the denominator in Eq. (2.3) is the summation of all substitutions averaged over the four nucleotides (in other words, weighted by nucleotide frequencies) and that  $\lambda$  is to scale this substitution to a sequence divergence of 1%. Because  $M$  is a transition probability matrix with each row values summing up to 1, the diagonal elements of  $M$  is simply 1 minus the three off-diagonal elements on the same row. The result  $M_1$  matrix is shown in Table 2.4.  $M_1$  specifies the probability of a nucleotide staying the same or being replaced by another after one PAM unit of time.

Each transition probability matrix can be used to obtain an associated PAM matrix. The PAM matrix associated from  $M_1$  is computed as

**Table 2.4** Transition probability matrix ( $M_1$ ) with a time interval of 1 nucleotide substitution per 100 nucleotide sites, together with PAM1 matrix

	Matrix $M_1$				PAM1 matrix			
	A	G	C	T	A	G	C	T
A	0.9944	0.0026	0.0014	0.0016	4.3102	-19.7283	-21.5036	-20.8631
G	0.0039	0.9917	0.0020	0.0025	-19.7283	6.1133	-19.9003	-19.0478
C	0.0026	0.0025	0.9851	0.0098	-21.5036	-19.9003	7.1075	-13.0279
T	0.0030	0.0030	0.0095	0.9844	-20.8631	-19.0478	-13.0279	6.9878

$$\text{PAM1}(i, j) = 10 \lg \frac{M_1(i, j)}{\pi_j} \quad (2.4)$$

where  $\lg$  stands for base-10 logarithm. The resulting PAM1 matrix is shown in Table 2.4, although PAM matrices used in sequence alignment are typically rounded to integers to speed up computation.

One may note that the PAM1 matrix in Table 2.4 is symmetrical which is characteristic of a time-reversible substitution model in which  $\pi_i M_{ij} = \pi_j M_{ji}$ . This time-reversibility is a necessary consequence of our not knowing the root between the two aligned sequences and consequently averaged the off-diagonal elements in matrix  $A$  (Table 2.3).

A few points are worth highlighting in PAM1 (Table 2.4). First, among the four match scores on the diagonal, an A/A match has the smallest score and a C/C match the largest. This is because A is the most frequent and C is the rarest nucleotide. Thus, an A/A match is more likely to arise by random chance than a C/C match and consequently is a weaker indication of homology than a C/C match. Second, a C/T mismatch is not penalized as much as an A/G match because C↔T transitions occur much more frequently than A↔G transitions. In general, the rarer the substitution, the heavier the penalty. Third, the average mismatch score is -19, and the average match score is 6. This may be taken as the justification in BLAST default with mismatch penalty three times greater as the match score (i.e., match score = 1 and mismatch score of -3 in BLAST default).

PAM1 is useful as a MM matrix only for sequences of low divergence. For example, if 2 sequences have diverged to 25%, i.e., 25 sites differ for every 100 sites, then the alignment score ( $S_a$ ) for the 2 sequences of length L, given the average match score of 6 and mismatch score of -19, will be approximately

$$S_a = L[0.25 \times (-19) + 0.75 \times 6] = -0.25L \quad (2.5)$$

A negative  $S_a$  suggests either that the two sequences have no homology or that the MM matrix is faulty. With the PAM approach, we would take  $M_1$  in Table 2.4 to generate  $M_{100}$ ,  $M_{150}$ , etc. and obtain PAM100, PAM150, etc., for increasingly divergent sequences.  $M_n$  is obtained by multiplying  $M_1$  n times, i.e.,  $M_n = M_1^n$ , and a PAM matrix for n PAM units of time is computed as

**Table 2.5** Transition probability matrix ( $M_{100}$ ) and the associated PAM100 matrix

	Matrix $M_{100}$				PAM100 matrix			
	A	G	C	T	A	G	C	T
A	0.6306	0.1592	0.1012	0.1090	2.3319	-1.8319	-2.7755	-2.5680
G	0.2418	0.4931	0.1269	0.1382	-1.8319	3.0793	-1.7910	-1.5401
C	0.1945	0.1607	0.3636	0.2812	-2.7755	-1.7910	2.7792	1.5458
T	0.2041	0.1702	0.2737	0.3520	-2.5680	-1.5401	1.5458	2.5213

$$\text{PAM}_n(i, j) = 10 \lg \frac{M_n(i, j)}{\pi_j} \quad (2.6)$$

$M_{100}$  and PAM100 obtained in this way are shown in Table 2.5. The average match score is 2.677925 and the average mismatch score is -1.49345. The alignment score ( $S_a$ ) calculated using the same approach as in Eq. (2.5) would be  $0.25^*(-1.49345) + 0.75*2.677925 = 1.635081$ . In fact, even two sequences with 60% divergence will still have positive alignment scores with PAM100.

Contrasting  $M_1$  and  $M_{100}$ , we note that a nucleotide A will have a probability of 0.9944 of remaining as A after 1 PAM unit of time but only 0.6306 after 100 PAM units of time. This is why we need different PAM matrices for aligning sequences of different divergence. We also notice that the difference between the four match scores and the 12 mismatch scores is larger in PAM1 than in PAM100. The difference will be even small with PAM250. The values in a PAM matrix will eventually all approach zero when homology is completely lost due to substitution saturation. Keep in mind that one PAM unit of time (resulting in 1% sequence divergence) may mean millions of years, so 100 or 250 PAM units of time could be billions of years.

Because  $M_n$  is obtained by extrapolation of  $M_1$  through  $M_n = M_1^n$ , any error inherent in  $M_1$  will be compounded in  $M_n$ . For this reason, PAM250, derived from  $M_{250}$  and used for highly diverged sequences, may not perform as well as BLOSUM45 which is not extrapolated but derived directly from comparison of sequences being no more than 45% identical.

### 2.2.1.2 BLOSUM Matrix for Nucleotide Sequences

The BLOSUM approach (Henikoff and Henikoff 1992) is based on empirical sets of sequences with different degrees of divergence. For example, those sequences being no more than 45% identical are used to generate BLOSUM45 matrix, the remaining sequence pairs being no more than 52% identical are used to generate BLOSUM52 matrix, and so on for BLOSUM62 all the way to BLOSUM90 for aligning sequences of little divergence. Keep in mind that a PAM matrix indicated with a larger number is for more divergent sequences, whereas a BLOSUM matrix indicated with a larger number is for less divergence sequences. While ideally one should use different PAM or BLOSUM matrices for sequences of different divergence, a multiple

alignment often involves a number of sequences with different degrees of divergence. If one has to use a single BLOSUM matrix to align them all, then BLOSUM62 is a good compromise and is consequently used as default in sequence alignment programs. Aligning closely related sequences are easy and almost any PAM or BLOSUM matrix can work well.

The same empirical substitution data in Table 2.3, i.e.,  $A(i,j)$ , can also be used to derive a BLOSUM matrix. We could call it BLOSUM95 matrix because the sequence pairs used to generate the data in Table 2.3 all have sequence identity approaching 95%. We first compute the expected value of  $A(i,j)$ , denoted by  $E(i,j)$ :

$$\begin{aligned} E(i, j) &= \frac{N_i N_j}{N} \\ E(A, A) &= \frac{N_A N_A}{N} = \frac{7335^2}{20016} = 2719.569 \\ E(A, G) &= \frac{N_A N_G}{N} = \frac{7335 \times 4857.5}{20016} = 1790.499 \end{aligned} \quad (2.7)$$

We next compute the log-odds (Table 2.6), with two numerical illustrations below:

$$\begin{aligned} \ln\left(\frac{p_{ij}}{p_i p_j}\right) &= \ln\left(\frac{A(i, j)}{E(i, j)}\right) \\ \ln\left(\frac{p_{AA}}{p_A p_A}\right) &= \ln\left(\frac{A(A, A)}{E(A, A)}\right) = \ln\left(\frac{7335}{2719.569}\right) = 0.9922 \\ \ln\left(\frac{p_{AG}}{p_A p_G}\right) &= \ln\left(\frac{A(A, G)}{E(A, G)}\right) = \ln\left(\frac{20}{1790.499}\right) = -4.4945 \end{aligned} \quad (2.8)$$

Finally, the resulting log-odds (Table 2.6) are scaled and rounded to integers to generate the final BLOSUM matrix. The scaling factor ( $\lambda$ ) is to allow the log-odds to be well represented by integers. If we round the original log-odds, then the rounding error will be large. If we multiply the values by 100 before rounding, then the rounding error would be small, but the values would take too much space. A good compromise is to use some value in between. BLOSUM62 divides log-odds by  $\lambda = \ln(2)/2$  (half-bit units) and BLOSUM50 by  $\lambda = \ln(2)/3$  (third-bit units). The rule

**Table 2.6** Deriving BLOSUM matrix, with log-odds and scaled BLOSUM matrix by  $\lambda = \ln(2)/3$  and rounded to integers

	Log-odds				BLOSUM			
	A	G	C	T	A	G	C	T
A	0.9922	-4.4945	-4.9033	-4.7558	4	-19	-21	-21
G	-4.4945	1.4072	-4.5341	-4.3378	-19	6	-20	-19
C	-4.9033	-4.5341	1.6358	-2.9517	-21	-20	7	-13
T	-4.7558	-4.3378	-2.9517	1.6082	-21	-19	-13	7

of thumb is to use a scaling factor so that the difference between the average match score and the average mismatch score is about 30. The BLOSUM matrix in Table 2.6 is produced by dividing the log-odds by  $\lambda = \ln(2)/3$  and then rounded to integers.

The values in the BLOSUM matrix in Table 2.6 and those in the PAM1 matrix in Table 2.4 are essentially perfectly correlated ( $r = 0.999998$ ). However, because all PAM matrices are extrapolated from  $M_1$  while each BLOSUM matrix results from separate compilation of empirical substitution matrix, BLOSUM matrices are likely more robust than PAM matrices for highly diverged sequences.

Note that, if we know the frequencies of the nucleotides and the BLOSUM matrix, we could infer  $\lambda$ . This we have already learned in the first chapter on string mathematics with the following equation:

$$\sum_{i=1}^4 \sum_{j=1}^4 \pi_i \pi_j e^{s_{ij}\lambda} = 1 \quad (2.9)$$

where  $s_{ij}$  refers to entries in the BLOSUM. Given the  $\pi$  values in Table 2.3 and BLOSUM matrix in Table 2.6, we can obtain  $\lambda = 0.2368$  which is not exactly  $\ln(2)/3 (=0.2310)$ . This discrepancy between the estimated  $\lambda$  and the true  $\lambda$  is because the values in BLOSUM matrix in Table 2.6 have been rounded; otherwise the two would be identical.

### 2.2.2 Match-Mismatch Matrix for Proteins

Amino acids differ from each other in volume, charge, polarity, and many other properties (Table 2.7 and Fig. 2.5), and amino acid residues in a protein confer to the protein's different properties. For example, proteins with long half-life ( $> 1$  day) typically have glycine, valine, or methionine at their N-terminus, whereas those with short half-life (a few minutes) typically have positively charged residues (arginine, lysine) at their N-terminus. A small and nonpolar amino acid residue such as glycine and alanine at the penultimate site (second amino acid site in the nascent peptide immediately following the initiator methionine) allows the initiator methionine to be efficiently cleaved, whereas a large and/or charged residue will not (Moerschell et al. 1990). Although the word “penultimate” is explained as meaning “second to the last” in English dictionaries, it has become standard in protein literature to mean the amino acid right after the initiator methionine. Amino acid replacements involving very different amino acids are generally selected against (Xia and Li 1998). For this reason, a scoring scheme with only match and mismatch is rarely used for protein sequence alignment.

Hydropathy index (Table 2.7) as a good measure of amino acid polarity and molecular weight (Table 2.7) as a measure of amino acid size have been used for deriving amino acid dissimilarity indices of which two kinds are used frequently. Grantham's distance (Grantham 1974) is based on polarity, size, and chemical composition of amino acid residues, whereas Miyata's distance (Miyata et al. 1979)

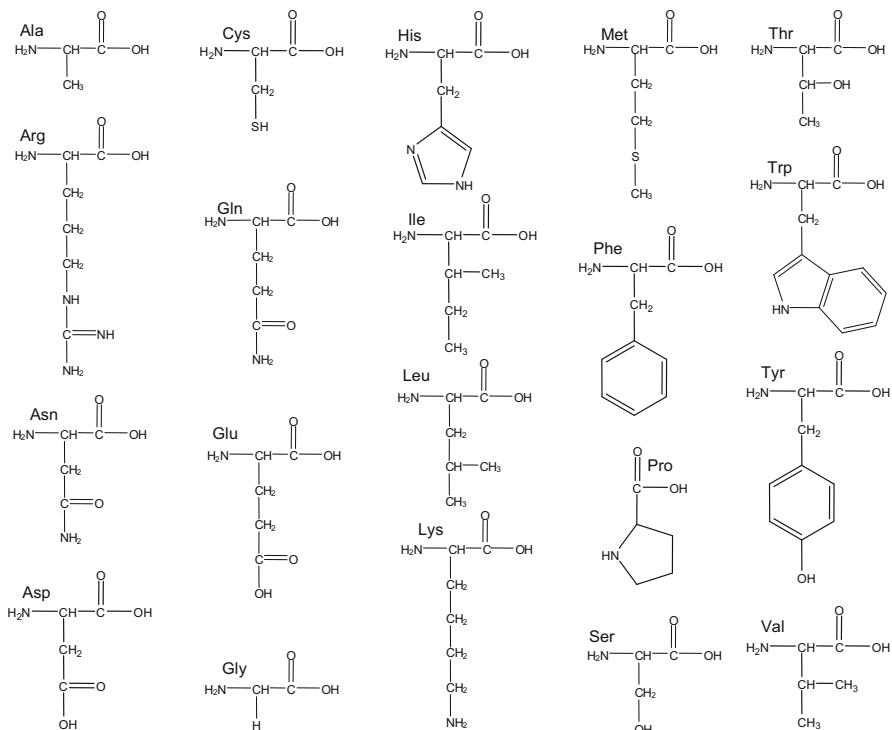
**Table 2.7** IUB letter codes of amino acids, proposed by the Nomenclature Committee of the International Union of Biochemistry (1985). These codes are now universally adopted by the scientific community. Also listed are molecular weight (MW), hydropathy index (HI, Kyte and Doolittle 1982), and usage in *E. coli* K12 (*E. coli*) and *Saccharomyces cerevisiae* (yeast)

One-letter	Three-letter	Meaning	Codon <sup>a</sup>	HI	MW	<i>E. coli</i>	Yeast
A	Ala	Alanine	GCT,GCC,GCA, GCG	1.8	89.094	125,332	160,810
R	Arg	Arginine	CGT,CGC,CGA, CGG,AGA,AGG	-4.5	174.203	72,502	130,068
N	Asn	Asparagine	AAT,AAC	-3.5	132.119	51,075	179,836
D	Asp	Aspartic	GAT,GAC	-3.5	133.104	67,349	171,072
C	Cys	Cysteine	TGT,TGC	2.5	121.154	15,188	37,093
Q	Gln	Glutamine	CAA,CAG CAG	-3.5	146.146	58,360	115,741
E	Glu	Glutamic	GAA, GAG GAG	-3.5	147.131	75,786	191,267
G	Gly	Glycine	GGT,GGC,GGA, GGG	-0.4	75.067	96,701	145,433
H	His	Histidine	CAT AT,CAC	-3.2	155.156	29,751	63,505
I	Ile	Isoleucine	ATT,ATC,ATA	4.5	131.175	78,845	191,677
L	Leu	Leucine	TTG,TTA,CTT, CTC,CTA,CTG	3.8	131.175	140,571	277,988
K	Lys	Lysine	AAA,AAG	-3.9	146.189	57,620	214,842
M	Met	Methionine	ATG	1.9	149.208	37,093	60,672
F	Phe	Phenylalanine	TTT,TTC	2.8	165.192	51,131	129,516
P	Pro	Proline	CCT, CCCCCC, CCA,CCG	-1.6	115.132	58,293	128,177
S	Ser	Serine	TCT,TCC,TCA, TCG,AGT,AGC	-0.8	105.093	75,661	263,096
T	Thr	Threonine	ACTCT,ACC,ACA, ACG	-0.7	119.119	70,494	173,084
W	Trp	Tryptophan	TGG	-0.9	204.228	20,060	30,387
Y	Tyr	Tyrosine	TATAT,TAC	-1.3	181.191	37,134	98,746
V	Val	Valine	GTT,GTC,GTG, GTG	4.2	117.148	93,061	162,642
*	End	Terminator	TAA, TAGTAG, TGA				

<sup>a</sup>Assuming the standard genetic code

is based on polarity and size only. These distances are typically evaluated against empirical amino acid substitution matrix because a good index of amino acid dissimilarity should be negatively correlated with the frequency of amino acid substitution. Paradoxically, Miyata's distance generally performs better than Grantham's distance in predicting the frequency of amino acid substitutions, presumably because the chemical composition of the side chain is a messy variable that probably contributes more noise than information.

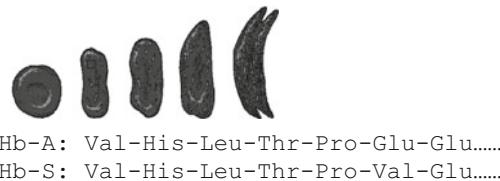
Another amino acid dissimilarity, based not on physiochemical properties of amino acid but on structural consideration, has also been proposed (Xia and Xie 2002).



**Fig. 2.5** Structural formula of 20 amino acids

Different protein structural components, such as  $\alpha$ -helix,  $\beta$ -sheet, etc., feature their unique amino acid compositions because some amino acids are incompatible with certain structures. For example, amino acids Ala and Glu are good  $\alpha$ -helix formers, whereas Pro is not as it introduces a sharp bend in the polypeptide chain. Similarly, Ile and Val are good  $\beta$ -sheet formers, whereas Glu and Pro are not. Thus, structurally compatible amino acids tend to be neighbors, so a distance derived in this way (Xia and Xie 2002) may be termed structural compatibility distance. It is also highly significantly and negatively correlated with the frequency of amino acid substitutions.

A frequently used example to illustrate the effect of an amino acid being replaced by a different amino acid is the sickle cell anemia. Sickle cell anemia is caused by a single amino acid replacement in the  $\beta$ -chain of the human hemoglobin at the sixth position, with a glutamate residue replaced by a valine residue (Fig. 2.6). Glutamate is negatively charged and hydrophilic and tends to stay on the surface of the protein in the aqueous environment in the red blood cell. In contrast, valine is a nonpolar and hydrophobic residue and tends to shrink into the middle of the protein. The deformed protein molecules then form bundles and distort the red blood cell that carries them, resulting in the characteristic shape of a sickle (Fig. 2.6). It is generally true that



**Fig. 2.6** Sickle cell anemia is caused by a single amino acid replace of a glutamate residue at the sixth position (Hb-A allele) by a valine residue (Hb-S allele). The mutant-deformed hemoglobin molecules distort the red blood cell which progresses from the normal disk-like shape to the sickle-like shape

amino acids of different polarity rarely replace each other (Xia and Li 1998), whereas amino acids with similar polarity can replace each other quite often (Xia and Kumar 2006).

Frequently used MM matrices for aligning protein sequences are of two types, the PAM matrix (Dayhoff et al. 1978) and the BLOSUM matrix (Henikoff and Henikoff 1992). While these MM matrices can be used for nucleotide sequences as we have shown in the previous section, they are originally developed for protein sequences. The derivation of PAM and BLOSUM matrices for amino acid sequences is exactly the same as what we have used for nucleotide sequences in the previous section, except that now the matrix is  $20 \times 20$  instead of  $4 \times 4$ . The same steps are involved in generating the matrices. For PAM, we (1) compile an empirical substitution matrix from closely related sequences, (2) obtain  $\lambda$ , (3) compute  $M_1$ , and (4) generate  $M_n$  as  $M_1^n$  and the associated PAM matrices. For BLOSUM, we compile sets of sequences with different degrees of divergence and use them to generate respective BLOSUM matrices.

Suppose we have an empirical substitution matrix A (Table 2.8) obtained from comparing HIV-1 gag sequences with about 85% similarity. The numbers are just small enough to fit the page and are for illustration purposes only because a good MM matrix would need significantly more data. From this empirical substitution matrix, we can obtain a series of PAM matrices and a BLOSUM85 matrix (so named because the sequence similarity is about 85%).

Amino acid frequencies ( $\pi$ ) and mutability ( $m$ ), shown in Table 2.9, vary much among the 20 amino acids. In general, two factors strongly affect the frequency of amino acid replacement. The first is amino acid similarity in physicochemical properties (Grantham 1974; Miyata et al. 1979; Xia and Li 1998). Almost all high-replacement amino acid pairs are polar ones (Xia and Kumar 2006). A polar amino acid tends to stay on the protein surface, whereas a nonpolar one tends to shrink into the center. For this reason, polar residues are rarely replaced by nonpolar ones, and one of the deleterious consequences of a polar amino acid by a nonpolar one is illustrated in Fig. 2.6. The second factor is similarity in their codons (Xia 1998b). Arg and Lys have the highest  $A(i,j)$  value (Table 2.8) among mismatched amino acids, although the two have only close to average frequencies (Table 2.9). Such a high-replacement rate can be attributed to both amino acids (1) being positively

**Table 2.8** Empirical substitution matrix A from aligned HIV-1 gag protein sequences with about 85% identity

	Ala	Arg	Asn	Asp	Cys	Gln	Glu	Gly	His	Ile	Leu	Lys	Met	Phe	Pro	Ser	Thr	Trp	Tyr	Val
Ala	792	1.5	5.5	4	0.5	1.5	5.5	9	0.5	4	0.5	6	1	1.5	6	10.5	<b>24</b>	0.5	0.5	12
Arg	1.5	670	6.5	1	1	8.5	6	12.5	2	1	1.5	<b>55.5</b>	2	0.5	2	5.5	7.5	1	0.5	2
Asn	5.5	6.5	<b>645</b>	<b>16.5</b>	0.5	2	6.5	6.5	5.5	5.5	1.5	14	2	1	1.5	<b>34.5</b>	<b>20</b>	0.5	1.5	2.5
Asp	4	1	<b>16.5</b>	426	0.5	0.5	<b>30.5</b>	8.5	2	1	1	5	0.5	0.5	1	7	4.5	0.5	1	1.5
Cys	0.5	1	0.5	0.5	346	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	1	0.5	1	2	0.5
Gln	1.5	8.5	2	0.5	0.5	715	6.5	1	6.5	0.5	5	<b>17.5</b>	1	0.5	9.5	2.5	2.5	0.5	1	0.5
Glu	5.5	6	6.5	<b>30.5</b>	0.5	6.5	724	16	1	1	1.5	<b>18</b>	0.5	0.5	2	3	6.5	0.5	1.5	3.5
Gly	9	12.5	6.5	8.5	0.5	1	16	865	1	1	0.5	3.5	3	0.5	1	13	3.5	3.5	1.5	1
His	0.5	2	5.5	2	0.5	6.5	1	1	255	0.5	2	1.5	0.5	1	1.5	1.5	1.5	0.5	7	0.5
Ile	4	1	5.5	1	0.5	0.5	1	1	0.5	741	<b>31.5</b>	5.5	13	3.5	1	5	<b>17.5</b>	0.5	1.5	<b>47</b>
Leu	0.5	1.5	1	0.5	5	1.5	0.5	2	<b>31.5</b>	1031	1	10	8.5	6	4	3.5	3	1	1	11
Lys	6	<b>55.5</b>	14	5	0.5	<b>17.5</b>	<b>18</b>	3.5	1.5	5.5	1	695	2.5	0.5	4.5	7	0.5	0.5	1.5	
Met	1	2	2	0.5	0.5	1	0.5	3	0.5	13	10	2.5	344	0.5	0.5	1.5	4	0.5	0.5	1.5
Phe	1.5	0.5	1	0.5	1.5	0.5	0.5	1	3.5	8.5	0.5	0.5	372	0.5	2.5	1	1	9	1	
Pro	6	2	1.5	1	0.5	9.5	2	1	1.5	1	6	0.5	0.5	0.5	636	7.5	5.5	0.5	0.5	0.5
Ser	10.5	5.5	<b>34.5</b>	7	1	2.5	3	13	1.5	5	4	4.5	1.5	2.5	7.5	627	<b>22</b>	1	1	1
Thr	<b>24</b>	7.5	<b>20</b>	4.5	0.5	2.5	6.5	3.5	1.5	<b>17.5</b>	3.5	7	4	1	5.5	<b>22</b>	677	0.5	1	6
Trp	0.5	1	0.5	0.5	1	0.5	0.5	3.5	0.5	0.5	3	0.5	0.5	1	0.5	1	367	1.5	0.5	
Tyr	0.5	0.5	1.5	1	2	1	1.5	1.5	7	1.5	1	0.5	0.5	9	0.5	1	1.5	299	1	
Val	12	2	2.5	1.5	0.5	0.5	3.5	1	0.5	<b>47</b>	11	1.5	1.5	1	0.5	1	6	0.5	1	645

Amino acid pairs replacing each other frequently are in bold

**Table 2.9** Amino acid frequencies ( $\pi$ ) and mutabilities ( $m$ ) computed from Table 2.8

AA	$\pi$	$m$	AA	$\pi$	$m$
Ala	0.0654	0.1066	Leu	0.0830	0.0831
Arg	0.0582	0.1497	Lys	0.0621	0.1731
Asn	0.0575	0.1720	Met	0.0288	0.1168
Asp	0.0379	0.1696	Phe	0.0301	0.0871
Cys	0.0265	0.0376	Pro	0.0505	0.0702
Gln	0.0578	0.0868	Ser	0.0558	0.1701
Glu	0.0616	0.1329	Thr	0.0602	0.1698
Gly	0.0703	0.0914	Trp	0.0284	0.0468
His	0.0216	0.1267	Tyr	0.0246	0.1021
Ile	0.0651	0.1599	Val	0.0546	0.1284

charged and (2) differing at only one codon site (AAR for Lys and AGR for Arg). Similarly, Asp and Glu also replace each other frequently, both being negatively charged and differing by a single codon site (GAY for Asp and GAR for Glu). Another example is Val and Ile, both being nonpolar, similar in size, and differing at only one codon site (AUY and AUA for Ile and GUN for Val). The genetic code has evolved in such a way that nonsynonymous mutations often lead to similar amino acids with little phenotypic effect on protein function. We should also note that the two factors affect amino acid replacement with different time scales. When divergence time is short, the similarity in two families of nonsynonymous codons strongly affects the frequency of amino acid replacement. However, with a long divergence time, multiple nucleotide substitutions would have happened, and the similarity in physiochemical properties becomes a key predictor of the frequencies of amino acid replacement. For this reason, we expect a mismatch between Arg and Lys to have negative mismatch score with a short time interval but to have a positive mismatch score with a long time interval.

To obtain the series of PAM matrices from the A matrix in Table 2.8, we can obtain  $\lambda = 0.08096$  from Table 2.9 by using Eq. (2.3) and then compute  $M1$ . We can then obtain  $M_{100}$ ,  $M_{200}$ , ...,  $M_{250}$  and their associated PAM100, PAM200, ..., PAM250 (which is shown in Table 2.10). As expected, a mismatch between Lys and Arg has a positive score, so does that between Asp and Glu. In fact, a site with Arg in one sequence and Lys in another suggests as much sequence homology as a site with a Lys on both sequences (PAM250 score = 8 in both cases, Table 2.10). However, a site with an Arg in both sequences suggests a stronger homology (PAM250 score = 10). This is because Arg is rarer than Lys in this set of sequences (Table 2.9). If  $n$  is small, e.g.,  $n = 1$ , then a PAM1 score for a Lys:Arg mismatch would be negative, and that for a Lys:Lys match would be positive. In other words, with a short divergence time, a Lys:Lys match would suggest homology, but a Lys:Arg would not, and this makes sense.

While we can derive a whole series of PAM matrices from the empirical matrix A in Table 2.8, we can obtain only one BLOSUM matrix from the data.

Table 2.10 PAM250 derived from the  $A$  matrix in Table 2.8

	Ala	Arg	Asn	Asp	Cys	Gln	Glu	Gly	His	Ile	Leu	Lys	Met	Phe	Pro	Ser	Thr	Trp	Tyr	Val
Ala	12	-3	1	0	-11	-6	-1	0	-5	-1	-6	-2	-4	-6	0	2	4	-10	-7	2
Arg	-3	10	1	0	-8	3	2	-1	-6	-9	8	-3	-9	-4	0	-1	-7	-7	-7	-6
Asn	1	1	7	4	-9	-3	2	1	1	-3	-6	1	-3	-5	-3	5	3	-9	-3	-3
Asp	0	0	4	10	-9	-3	8	4	0	-5	-9	1	-5	-7	-5	2	1	-8	-4	-5
Cys	-11	-8	-9	-9	36	-10	-10	-11	-3	-10	-10	-10	-8	-4	-10	-8	-10	1	7	-11
Gln	-6	3	-3	-3	-10	17	0	-5	5	-8	-5	5	-6	-8	4	-3	-4	-10	-3	-9
Glu	-1	2	2	8	-10	0	11	4	-2	-6	-9	3	-6	-9	-5	0	-1	-9	-5	-5
Gly	0	2	1	4	-11	-5	4	14	-4	-7	-11	0	-4	-9	-7	2	-1	0	-5	-7
His	-5	-1	1	0	-3	5	-2	-4	20	-5	-4	-1	-5	3	-1	-1	-2	-4	12	-6
Ile	-1	-6	-3	-5	-10	-8	-6	-7	-5	10	7	-5	6	0	-6	-2	1	-8	-4	9
Leu	-6	-9	-6	-9	-10	-5	-9	-11	-4	7	14	-8	6	5	-2	-5	-3	-2	-3	5
Lys	-2	8	1	1	-10	5	3	0	-1	-5	-8	8	-3	-9	-5	0	0	-9	-6	-5
Met	-4	-3	-3	-5	-8	-6	-6	-4	-5	6	6	-3	18	-2	-6	-3	0	-6	-5	3
Phe	-6	-9	-5	-7	4	-8	-9	3	0	5	-9	-2	24	-7	-4	-5	0	15	-2	
Pro	0	-4	-3	-5	-10	4	-5	-7	-1	-6	-2	-5	-6	-7	21	0	-1	-10	-7	-7
Ser	2	0	5	2	-8	-3	0	2	-1	-2	-5	0	-3	-4	0	7	3	-7	-4	-3
Thr	4	-1	3	1	-10	-4	-1	-1	-2	1	-3	0	0	-5	-1	3	6	-9	-5	1
Trp	-10	-7	-9	-8	1	-10	-9	0	-4	-8	-2	-9	-6	0	-10	-7	-9	33	3	-9
Tyr	-7	-7	-3	-4	7	-3	-5	-5	12	-4	-3	-6	-5	15	-7	-4	-5	3	24	-5
Val	2	-6	-3	-5	-11	-9	-5	-7	-6	9	5	-5	3	-2	-7	-3	1	-9	-5	13

Scaled by dividing the values from Eq. (2.6) by  $\lambda = \ln(2)/2$

This BLOSUM matrix may be called BLOSUM85 because the sequence pairs that generated the matrix  $A$  have sequence similarity close to 85%. This BLOSUM85 matrix will only be good for very closely related protein sequences. If we want to align highly diverged HIV-1 gag proteins, we have to obtain a different set of aligned sequences with a much lower sequence similarity. The BLOSUM85 matrix derived from Table 2.8 is shown in Table 2.11.

We note that the score is 7 for a Lys:Lys pair but 0 for a Lys:Arg pair. In other words, for low-divergence sequences, a Lys:Arg mismatch does not suggest homology, but a Lys:Lys pair does. If we use a set of highly diverged sequences, then the resulting BLOSUM matrix is expected to have a positive score for a Lys:Arg pair.

## 2.3 *Pairwise Alignment with Gap Penalty Specified by the Affine Function*

The need for replacing constant gap penalty by affine function gap penalty may be illustrated by the two alignments in Fig. 2.3. The alignment in Fig. 2.3a is generally considered by biologists as more likely than that in Fig. 2.3b because indels are rare evolutionary events, with a single indel of length 2 more likely than two independent indels of length 1. Given the simple scoring scheme with constant penalty (match score = 2 and constant gap penalty of -2), the two alignments are equally good. However, if  $G_O$  is distinguished from  $G_E$ , e.g., assign  $G_O = -5$  and  $G_E = -2$  (which is the BLAST default), then the alignment score will be 1 ( $=4*2-5-2$ ) for the alignment in Fig. 2.3a but -2 ( $=4*2-5-5$ ) for the alignment in Fig. 2.3b. With this new gap penalty scheme, the alignment in Fig. 2.3a is better than that in Fig. 2.3b. The rule of thumb is that  $G_O > G_E$  and that the difference between  $G_O$  and  $G_E$  should increase with the frequency of long indels. The absolute values of  $G_O$  and  $G_E$  should be set in such a way that the alignment score between homologous sequences is positive. Setting  $G_O = 3*$  (average mismatch score) appears to be a good compromise.

Pairwise alignment with the affine function gap penalty is complicated, and I will take a verbose approach instead of a condensed one to make sure that all details are covered to smooth your understanding. The cost of failing to understand is much greater than that of a few extra minutes of reading.

### 2.3.1 Scoring Matrices

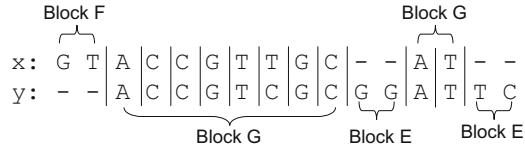
In contrast to dynamic programming with constant gap penalty where only one alignment scoring matrix is needed, alignment with affine function gap penalty needs to have three to keep alignment scores (Gotoh 1982). The algorithm is based on the insight that there are three kinds of alignment blocks (Fig. 2.7). Block  $G$  represents alignments with  $x(i)$  aligned with  $y(j)$ , Block E represents

**Table 2.11** BLOSUM85 matrix derived from data in Table 2.8, scaled  $\lambda = \ln(2)/2$ 

	Ala	Arg	Asn	Asp	Cys	Gln	Glu	Gly	His	Ile	Leu	Lys	Met	Phe	Pro	Ser	Thr	Trp	Tyr	Val
Ala	8	-10	-6	-6	-11	-10	-7	-6	-10	-8	-14	-6	-9	-8	-6	-4	-2	-11	-11	-4
Arg	-10	8	-6	-10	-9	-5	-6	-4	-6	-11	-11	0	-7	-11	-9	-6	-5	-9	-11	-9
Asn	-6	-6	8	-2	-11	-9	-6	-6	-3	-6	-11	-4	-7	-9	-9	-1	-2	-11	-7	-8
Asp	-6	-10	-2	9	-10	-12	0	-4	-5	-10	-11	-5	-10	-10	-9	-4	-6	-10	-7	-8
Cys	-11	-9	-11	-10	10	-11	-11	-8	-11	-12	-11	-9	-6	-10	-9	-11	-7	-4	-11	-11
Gln	-10	-5	-9	-12	-11	8	-6	-12	-3	-13	-7	-3	-9	-11	-4	-8	-8	-11	-9	-13
Glu	-7	-6	-6	0	-11	-6	8	-4	-8	-12	-11	-3	-11	-11	-9	-8	-6	-11	-8	-7
Gly	-6	-4	-6	-4	-11	-12	-4	7	-9	-12	-15	-8	-6	-12	-11	-4	-8	-6	-8	-11
His	-10	-6	-3	-5	-8	-3	-8	-9	11	-10	-7	-7	-8	-6	-7	-7	-7	-8	0	-10
Ile	-8	-11	-6	-10	-11	-13	-12	-12	-10	7	-2	-7	-2	-6	-11	-7	-3	-11	-8	0
Leu	-14	-11	-11	-12	-7	-11	-15	-7	-2	7	-12	-3	-4	-6	-8	-9	-7	-10	-5	-5
Lys	-6	0	-4	-5	-11	-3	-3	-8	-7	-7	-12	7	-7	-11	-13	-7	-6	-11	-11	-10
Met	-9	-7	-7	-10	-9	-9	-11	-6	-8	-2	-3	-7	10	-9	-11	-8	-5	-9	-9	-8
Phe	-8	-11	-9	-10	-6	-11	-11	-12	-6	-6	-4	-11	-9	10	-11	-6	-9	-7	0	-9
Pro	-6	-9	-9	-9	-10	-4	-9	-11	-7	-11	-6	-13	-11	-11	8	-5	-6	-11	-10	-12
Ser	-4	-6	-1	-4	-9	-8	-8	-4	-7	-7	-8	-7	-8	-6	-5	8	-2	-9	-8	-11
Thr	-2	-5	-2	-6	-11	-8	-6	-8	-7	-3	-9	-6	-5	-9	-6	-2	8	-11	-9	-6
Trp	-11	-9	-11	-10	-7	-11	-11	-6	-8	-11	-7	-11	-9	-7	-11	-9	-11	10	-5	-11
Tyr	-11	-11	-7	-7	-4	-9	-8	-8	0	-8	-10	-11	-9	0	-10	-8	-9	-5	10	-8
Val	-4	-9	-8	-8	-11	-13	-7	-11	-10	0	-5	-10	-8	-9	-12	-11	-6	-11	-8	8

The matrix is for illustration only because of the limited data in Table 2.8

**Fig. 2.7** Three kinds of alignment blocks in dynamic programming with affine function gap penalty



**Table 2.12** Match-mismatch matrix, designated by S in the text, used for computing alignment scores

	A	G	C	T
A	2	-1	-1	-1
G	-1	2	-1	-1
C	-1	-1	2	-1
T	-1	-1	-1	2

The alignment algorithm can use any reasonable S

those with  $y(j)$  facing a gap character (a deletion), and Block  $F$  represents those with  $x(i)$  facing a gap character (Fig. 2.7). We need three matrices, designated by  $G$ ,  $E$ , and  $F$ , respectively, to keep track of these three blocks. For clarity, I will also use an extra matrix, designated by  $V$ , to keep track of the maximum of  $G(i,j)$ ,  $E(i,j)$ , and  $F(i,j)$ . These designations follow Gusfield (1997).

To align two sequences  $x$  and  $y$  with lengths  $m$  and  $n$ , respectively, we designate  $x$  as the column sequence and  $y$  as the row sequence and set up the matrices with the following values (if leading gaps at the beginning of the alignment are penalized). If leading gaps are not penalized, then set  $V(i,0) = V(0,j) = 0$ . The leading gaps are penalized in the numerical illustration below.

$$\begin{aligned} V(0,0) &= 0 \\ V(i,0) &= -GO - (i-1)GE, \quad \text{for } 1 \leq i \leq m \\ V(0,j) &= -GO - (j-1)GE, \quad \text{for } 1 \leq j \leq m \end{aligned} \tag{2.10}$$

$$\begin{aligned} E(0,j) &= 0, \quad \text{for } 0 \leq j \leq n \\ E(i,0) &= -GO - (i-1)GE, \quad \text{for } 1 \leq i \leq m \\ F(i,0) &= 0, \quad \text{for } 0 \leq i \leq m \\ F(0,j) &= -GO - (j-1)GE, \quad \text{for } 1 \leq j \leq n \\ G(i,0) &= 0, \quad \text{for } 0 \leq i \leq m \\ G(0,j) &= 0, \quad \text{for } 0 \leq j \leq n \end{aligned} \tag{2.11}$$

With a match-mismatch matrix designated by S (Table 2.12), GO (gap open penalty) = 3, and GE (gap extension penalty) = 2, the general recurrences are

$$\begin{aligned}
 G(i, j) &= V(i - 1, j - 1) + S(x_i, y_j) \\
 E(i, j) &= \max[E(i, j - 1) - GE, V(i, j - 1) - GO] \\
 F(i, j) &= \max[F(i - 1, j) - GE, V(i - 1, j) - GO] \\
 V(i, j) &= \max[G(i, j), E(i, j), F(i, j)]
 \end{aligned} \tag{2.12}$$

In the spirit of recycling, we will reuse the two sequences used previously for illustrating the dynamic programming for constant gap penalty, i.e.,  $x = \text{"ACGGCT"}$  and  $y = \text{"ACGT"}$ . The initialization values according to Eq. (2.10) and Eq. (2.11) are in the first row and first column of  $G$ ,  $E$ ,  $F$ , and  $V$  matrices in Table 2.13. Sequences  $x$  and  $y$  are also referred to as the column sequence (i.e., taking up the first column in the matrices) and row sequence (i.e., taking up the first row), respectively, as set up in the matrices shown in Table 2.13.

We fill up the values of  $G$ ,  $E$ ,  $F$ , and  $V$  matrices according to Eq. (2.12) from left to right and from the top, just as we have done previously with the constant gap penalty. The first values in the matrices are

$$\begin{aligned}
 G(1, 1) &= V(0, 0) + S(A, A) = 0 + 2 = 2 \\
 E(1, 1) &= \max[E(1, 0) - GE, V(1, 0) - GO] = \max[-3 - 2, -3 - 3] = -5 \\
 F(1, 1) &= \max[F(0, 1) - GE, V(0, 1) - GO] = \max[-3 - 2, -3 - 3] = -5 \\
 V(1, 1) &= \max[G(1, 1), E(1, 1), F(1, 1)] = 2
 \end{aligned} \tag{2.13}$$

What is nice with the dynamic programming is that once we know how to compute the first value, we know how to compute all the values because they are all computed exactly the same way. The complete scoring matrices  $G$ ,  $E$ , and  $F$  for aligning the two sequences are shown in Table 2.13, with the final optimal alignment score being 3 in  $V(m,n)$  which is the maximum of  $G(m,n)$ ,  $E(m,n)$ , and  $F(m,n)$  in Table 2.13. Note that, because  $G$  matrix is for aligning sections containing no indels, its entries are always the alignment score  $V(i-1, j-1)$  plus the match-mismatch score for the next nucleotide pair  $x(i)$  and  $y(j)$ . Also note that each entry in the  $V$  matrix is the alignment score up to the corresponding nucleotide. For example, the bolded number 4 in the  $V$  matrix (Table 2.13) is the alignment score for aligning sequences ACG and AC, the bolded number 2 in the  $V$  matrix is the alignment score for aligning ACGT and ACGC, and the final number of 3 at the bottom right is the alignment score of aligning ACGT and ACGGCT.

In progressive multiple alignment with a guide tree, the first step is to build a tree based on the alignment score from each pairwise alignment. For example, with four sequences, we have six pairwise alignments and six alignment scores (i.e.,  $s_{12}$ ,  $s_{13}$ ,  $\dots$ ,  $s_{34}$ , between sequences 1 and 2, 1 and 3,  $\dots$ , 3 and 4). In our example with only two sequences,  $s_{x,y} = 3$ . These alignment scores are similarity indices, i.e., the larger they are, the more similar the two sequences in the pair. The simplest way to convert them to a distance index is by  $D_{ij} = \text{Max}(s_{ij}) - s_{ij}$ , so that  $D$  is 0 between the two most similar sequences and is largest between the two sequences with the smallest  $s_{ij}$ .

**Table 2.13** Alignment scoring matrices  $G(i,j)$ ,  $E(i,j)$ ,  $F(i,j)$ , and  $V(i,j)$  with GO = 3, GE = 2, and the match-mismatch scores as shown in Table 2.12

$G$ matrix				$E$ matrix				$F$ matrix				$V$ matrix				
	A	C	G	T	A	C	G	T	A	C	G	T	A	C	G	T
0	0	0	0	0	0	0	0	0	-3	-5	-7	-9	0	-3	-5	-7
A	0	-2	-4	-6	-8	-3	-5	-1	-3	-5	0	-5	-7	-9	-11	-3
C	0	-4	4	-2	-4	-5	-7	-4	1	-1	0	-1	-4	-6	-8	-5
G	0	-6	-2	6	0	-7	-9	-6	-2	3	0	-3	1	-2	-4	-7
G	0	-8	-4	3	5	-9	-11	-8	-4	0	0	-5	-1	3	0	-9
C	0	-10	-3	-2	2	-11	-13	-10	-6	-2	0	-7	-3	1	2	-11
T	0	-12	-8	-4	3	-13	-15	-12	-8	-4	0	-9	-5	-1	0	-13

These  $D_{ij}$  values can then be input into a distance-based phylogenetic method to generate the first guide tree. Thus, for this tree-building step, only pairwise alignment score is needed, and no effort is spent in obtaining backtrack matrices and the actual sequence alignment.

Most multiple alignment programs will use this guide tree to generate the first multiple alignment, build a new tree from the alignment, and use the new tree (which typically is better than the first guide tree from alignment scores) as a new guide tree to guide another round of multiple alignment. Such iteration can go a few more rounds until an index of multiple alignment quality does not improve. Both MAFFT (Katoh et al. 2005; Katoh and Toh 2010) and MUSCLE (Edgar 2004) take this iterative approach.

### 2.3.2 Backtrack Matrices

We need backtrack matrices to obtain the actual sequence alignment. The use of affine function gap penalty demands three backtrack matrices, designated  $B_0$ ,  $B_1$ , and  $B_2$ , in contrast to just one with constant gap penalty. Backtrack matrices are computed concurrently with the scoring matrices. It is only for clarity that I illustrate their computation separately below. I am not sure if this is a good approach.

The three backtrack matrices are filled from left to right and from top to bottom, according to Eqs. (2.14), (2.15), and (2.16). I have used values of 0, 1, and 2 for different states, but readers and teachers using the book are welcome to use arrows or whatever symbols to improve the clarity.

$$\begin{aligned} B_0(i, j) &= 0 \text{ when } \max[G(i, j), E(i, j), F(i, j)] = G(i, j) \\ B_0(i, j) &= 1 \text{ when } \max[G(i, j), E(i, j), F(i, j)] = E(i, j) \\ B_0(i, j) &= 2 \text{ when } \max[G(i, j), E(i, j), F(i, j)] = F(i, j) \end{aligned} \quad (2.14)$$

$$\begin{aligned} B_1(i, j) &= 1 \text{ when } E(i, j - 1) - GE \leq V(i, j - 1) - GO \\ B_1(i, j) &= 2 \text{ when } E(i, j - 1) - GE > V(i, j - 1) - GO \end{aligned} \quad (2.15)$$

$$\begin{aligned} B_2(i, j) &= 1 \text{ when } F(i - 1, j) - GE \leq V(i - 1, j) - GO \\ B_2(i, j) &= 2 \text{ when } F(i - 1, j) - GE > V(i - 1, j) - GO \end{aligned} \quad (2.16)$$

Note that I used the  $\leq$  sign for  $B_1$  and  $B_2$  matrices; otherwise we will have the equivalent of two-arrowed monster cells seen in Fig. 2.2. This implies that the backtrack matrices will recover only one of potentially several equally optimal alignments. The numerical illustration for the first two cells in each of the tree matrices is in Eq. (2.17). Note that the backtrack matrices here are indexed from 1, whereas the three scoring matrices are indexed from 0. The  $x$  and  $y$  sequences are also indexed from 1, e.g.,  $x(1) = "A"$  and  $y(1) = "A."$

**Table 2.14** Three backtrack matrices ( $B_0$ ,  $B_1$ , and  $B_2$ ) for obtaining sequence alignment, with the match-mismatch matrix  $S$  specified in Table 2.12, GO (gap open penalty) = 3, and GE (gap extension penalty) = 2

	$B_0$				$B_1$				$B_2$			
	A	C	G	T	A	C	G	T	A	C	G	T
A	0	1	1	1	2	1	2	2	2	2	2	2
C	2	0	1	1	2	1	1	2	1	1	1	1
G	2	2	0	1	2	1	1	1	2	1	1	1
G	2	2	0	0	2	1	1	1	2	2	1	1
C	2	0	2	0	2	1	1	1	2	2	2	1
T	2	2	2	0	2	1	1	1	2	2	2	2

$$B_0(1, 1) = 0 \text{ because } \max[G(1, 1), E(1, 1), F(1, 1)] = G(1, 1)$$

$$B_1(1, 1) = 2 \text{ because } E(1, 0) - \text{GE} > V(1, 0) - \text{GO} \quad (2.17)$$

$$B_2(1, 1) = 2 \text{ because } F(0, 1) - \text{GE} > V(0, 1) - \text{GO}$$

$$B_0(1, 2) = 1 \text{ because } \max[G(1, 2), E(1, 2), F(1, 2)] = E(1, 2)$$

$$B_1(1, 2) = 1 \text{ because } E(1, 1) - \text{GE} < V(1, 1) - \text{GO} \quad (2.18)$$

$$B_2(1, 2) = 2 \text{ because } F(0, 2) - \text{GE} > V(0, 2) - \text{GO}$$

The complete backtrack matrices, shown in Table 2.14, are needed to obtain sequence alignment. However, they do not need extra memory space because scoring matrices are not needed for obtaining sequence alignment and the memory space for scoring matrices can be used by backtrack matrices.

### 2.3.3 Obtain Sequence Alignment from Backtrack Matrices

Many students get confused with applying the backtrack matrices to obtain the sequence alignment. The confusion mainly arises from the necessity of jumping from one backtrack matrix to another in recovering the alignment. The only case when you do not need to jump from one backtrack matrix to another is when the optimal alignment has no indel.

We start by checking the bottom-right value of matrix  $B_0$ , i.e.,  $B_0(m, n)$ . If it is 0, then we align  $x(m)$  against  $y(n)$  and move up-left to  $B_0(m-1, n-1)$ . If we continue to encounter 0 in  $B_0(m-1, n-1)$ ,  $B_0(m-2, n-2)$ , etc., we will align  $x(m-1)$  against  $y(n-1)$ ,  $x(m-2)$  against  $y(n-2)$ , . . . until  $B_0(i, j) <> 0$ .

If  $B_0(i, j) = 1$ , then we move to the corresponding cell in matrix  $B_1$ , i.e.,  $B_1(i, j)$  which takes a value of either 1 or 2. If  $B_1(i, j) = 2$ , then we will (1) align a gap against  $y(j)$  and (2) move left to the next cell, i.e.,  $B_1(i, j-1)$ . As long as the next cell is 2 in  $B_1$  matrix, we will continue to align a gap against nucleotide in  $y$  and move left until we encounter a 1 in  $B_1$  matrix. Encountering a value of 1 in  $B_1$  matrix means that we will (1) align a gap against  $y(j)$ , (2) move left to the next cell, and do the additional

```

Seq x: ACGGCT
Seq y: ACG--T
Step   654321

```

**Fig. 2.8** Steps taken to align the two sequences using the backtrack matrices

(3) jump back to the corresponding cell in  $B_0$  matrix and decide what to do by checking the cell value in  $B_0$ .

If  $B_0(i,j) = 2$ , then we move to the corresponding cell in matrix  $B_2$ , i.e.,  $B_2(i,j)$ . If  $B_2(i,j) = 2$ , then we will (1) align a gap character against  $x(i)$  and (2) move up the matrix to the next cell, i.e.,  $B_2(i-1,j)$ . We continue to align the gap character against nucleotide in  $x$  until we encounter a value of 1 in  $B_2$ . A value of 1 in  $B_2$  means that we will (1) align a gap against  $x(i)$ , (2) move upward to the next cell, and do the additional (3) jump back to the corresponding cell in  $B_0$ .

Take, for example, the backtrack matrices  $B_0$ ,  $B_1$ , and  $B_2$  shown in Table 2.14. We note that  $B_0(m,n) = B_0(6,4) = 0$ , so we align  $x(6)$ , i.e., the last T in  $x$ , against  $y(4)$ , i.e., the last T in  $y$  (Step 1 in Fig. 2.8). We always move to the up-left cell after encountering a 0 in  $B_0$ , so this moves us from  $B_0(6,4)$  to  $B_0(5,3)$ . Whenever we encounter a 2 in  $B_0$ , we move to the corresponding cell in matrix  $B_2$  which is  $B_2(5,3)$ .  $B_2(5,3) = 2$ . A number 2 in  $B_2$  always means that (1) we align a gap against the nucleotide in the column sequence, i.e., against  $x(5) = "C"$  (Step 2 in Fig. 2.8), and (2) move upward to the next cell, i.e.,  $B_2(4,3)$  which is 1. A change of value from 2 to 1 in  $B_2$  always means that (1) we align a gap against the nucleotide in the column sequence, i.e., against  $x(4) = G$  (Step 3 in Fig. 2.8), (2) move upward to the next cell, i.e.,  $B_2(3,3)$ , and (3) jump back to the corresponding cell in  $B_0$ , i.e.,  $B_0(3,3)$ . Because  $B_0(3,3) = 0$ , we align the two corresponding nucleotides (G/G, Step 4 in Fig. 2.8) and move to  $B_0(2,2)$ . Because both  $B_0(2,2)$  and  $B_0(1,1)$  are zero, we align the corresponding nucleotides (Steps 5 and 6) to complete the alignment.

One simple way to check if our backtracking is correct is to compute the alignment score in Fig. 2.8. There are four matches each getting two points, one gap open with penalty 3 and one gap extension with penalty 2. The alignment score is therefore  $2*4 - 3 - 2 = 3$  which is the same as the value in  $V(6,4)$  in Table 2.13.

### 3 Multiple Sequence Alignment (MSA)

It is impractical to perform multiple sequence alignment by extending the dynamic programming approach from two dimensions to multiple dimensions. An alternative is to use an approximate evaluation criterion termed the sum-of-pairs (SP) criterion (Althaus et al. 2002; Gupta et al. 1995; Lipman et al. 1989; Reinert et al. 2000; Stoye et al. 1997). Each multiple alignment of  $N$  sequences implies  $N(N-1)/2$  pairwise alignments. The SP criterion is simply the summation of all pairwise alignment scores without penalizing shared gaps. Evaluating MSAs with the SP criterion is

easy. All we need is a scoring scheme, i.e., gap open and gap extension penalties plus a match/mismatch matrix. However, applying the SP criterion to choose the best MSA is much harder. A conceptually sound but even slower approach is the simultaneous construction of MSA and the phylogeny (Sankoff 1975; Sankoff et al. 1976, 1973).

An alternative is to use a guide tree to reduce the multiple alignment problem to the pairwise alignment problem (Feng and Doolittle 1987; Hogeweg and Hesper 1984). All practical multiple alignment programs, e.g., Clustal series of programs (Higgins 1994; Thompson et al. 1994), MAFFT (Katoh et al. 2005; Katoh and Toh 2010), and MUSCLE (Edgar 2004), take this approach. In short, we start from the tips of the tree and progress toward the root by aligning sequences represented by the two sister nodes. If the sister nodes are two tips on the tree, then it is the conventional pairwise alignment. If the sister nodes involve a tip and one internal node, then the daughter sequences descending down the internal node is represented either by a sequence profile or a reconstructed ancestral sequence. Aligning a reconstructed ancestral sequence and a real sequence from the tip needs only the conventional pairwise alignment we have covered extensively in previous sections. However, with the daughter sequences descending from the internal node that are represented by a sequence profile, then the alignment is somewhat different. If the sister nodes are both internal nodes and sequence profile representation is used for multiple sequences, then we need the pairwise alignment between two profiles. The pairwise alignment of sister nodes progresses to the root of the guide tree to complete the multiple alignment. This progressive multiple alignment represents a practical compromise between accuracy and speed.

There are advantages and disadvantages of representing ancestral sequences by either a profile or an ancestral sequence. We will make a brief comparison later in this chapter.

### 3.1 Dynamic Programming for Profile Alignment

Profile alignment aligns either one sequence (designated T) against a set of already aligned sequences in the form of a profile (designated S) or aligns two profiles  $S_1$  and  $S_2$ . It is an essential technique for multiple sequence alignment. There are various approaches to profile alignment. The simplest is to get a consensus sequence from S (designated  $C_S$ ) and align T and  $C_S$  by using the pairwise alignment method we learned in previous sections. Whenever we insert a gap in  $C_S$ , we insert a corresponding gap in all sequences in S. However, we will learn a mathematically more acceptable approach in this section.

Suppose we want to align sequence T = “ACG” against S derived from the following three aligned sequences:

**Fig. 2.9** Application of dynamic programming to profile alignment. T (“ACG”) is the column sequence, and the “row sequence” is a sequence profile of length 5. Christian Delamarche noted an error in the bottom row with the value  $4/3$  in the previous edition

	A	2/3	0	0	1/3	0
	C	0	1	1/3	0	0
	G	1/3	0	0	2/3	0
	T	0	0	0	0	1
	-	0	0	2/3	0	0
	0	-3	-6	-9	-12	-15
A	-3	5/3	-4/3	-5/3	-14/3	-23/3
C	-6	-4/3	11/3	10/3	1/3	-8/3
G	-9	-13/3	2/3	4/3	5	2

AC-GT

AC-GT

GCCAT

The first step in profile alignment is to represent  $S$  with a site-specific frequency profile. The set of three aligned sequences have five symbols (A, C, G, T, and the gap symbol “-”) and can be represented by the profile shown in the first five rows in Fig. 2.9. The first column is a list of the five symbols, followed by five data columns corresponding to five aligned sites. Each data column corresponds to frequencies of symbols in an aligned site. The first data column represents the frequencies of symbols in the first aligned site, with the frequencies of A and G being  $2/3$  and  $1/3$ , respectively. The second data column represents the frequencies of the second aligned site with the frequency C being 1 and the frequencies of other symbols being 0 and so on. Thus,  $S$  can always be represented by a site-specific profile in the form of an  $N \times L$  matrix where  $N$  is the number of symbols and  $L$  is the sequence length. It is important to note that any phylogenetic information among sequences in  $S$  is lost in converting the set of aligned sequences to a profile.

The second step is to perform a special version of the dynamic programming to generate the scoring matrix and backtrack matrix. I will illustrate the profile alignment principle by using the constant gap penalty. With the length of T being 3 and the length of S being 5, there are only 15 cells to fill in. Because one needs to compute three values (DIAG, UP, and LEFT) for each cell, the total number of values to compute is 45. Yet even for such a small problem, manual computation is quite difficult and error-prone.

The score matrix and the backtrack matrix (Fig. 2.9) were obtained with a special scoring scheme. There are two kinds of matches, a match between two identical nucleotides and a match between two gap symbols. The match scores for the former and latter are designated  $M_{\text{Nuc}}$  and  $M_{\text{Gap}}$ , respectively, with corresponding values set to 2 and 1, respectively, in the example. There are also two kinds of mismatches, one involving a transitional difference and the other a transversional difference. They are

designated as  $MM_s$  and  $MM_v$ , respectively, with corresponding values set to 1 and  $-1$ , respectively, in this example. We set the constant gap penalty with  $G = -3$ .

We now illustrate how the scoring matrix and backtrack matrix (Fig. 2.9) are computed. For the first cell, the UP and LEFT values are simple:

$$\begin{aligned} \text{UP} &= -3 + G = -3 - 3 = -6 \\ \text{LEFT} &= -3 + G = -3 - 3 = -6 \end{aligned}$$

For the DIAG value, we should keep in mind that the nucleotide A in the column sequence has a probability of  $2/3$  of an A/A match and a probability of  $1/3$  of an A/G transition. This leads to

$$\text{DIAG} = 0 + 2/3 \times M_{\text{Nuc}} + 1/3 \times MM_s = 5/3$$

Because DIAG is the maximum of the three, it is used to fill the first cell, together with the associated up-left arrow (Fig. 2.9). The second cell (to the right of the first cell) is simple because the profile at the second site contains C only. So the computation is the same as in regular pairwise alignment:

$$\begin{aligned} \text{DIAG} &= -3 - 1 = -4 \\ \text{UP} &= -6 - 3 = -9 \\ \text{LEFT} &= 5/3 - 3 = -4/3 \end{aligned}$$

Because LEFT is the largest of the three, the cell is filled with  $-4/3$  together with a left-pointing arrow.

The cell likely to cause some confusion is the third, i.e., the one with a value of  $-5/3$  and a left-pointing arrow. Note that an up-left arrow in that cell implies that A will pair with C with a probability of  $1/3$ , penalized by  $MM_v = -1$ , and pair with “-” with a probability of  $2/3$ , penalized by  $G = -3$ . Therefore,

$$\text{DIAG} = -6 - 2/3 \times 3 - 1/3 \times 1 = -25/3$$

The calculation of UP is simply  $\text{UP} = -9 - 3 = -12$ . A left-pointing arrow, however, implies a gap in the column sequence, so we have a gap with a probability of  $2/3$  of facing a gap in the row profile, with  $M_{\text{gap}} = 1$ , and a probability of  $1/3$  of facing a C, with  $G = -3$ . Therefore,

$$\text{LEFT} = -4/3 + 2/3 \times 1 - 1/3 \times 3 = -5/3.$$

Because LEFT is the largest of the three, the cell is filled with  $-5/3$  and a left-pointing arrow. The rest of the cells are relatively straightforward. The alignment can again be obtained by tracing the backtrack matrix (Fig. 2.9):

```
AC-GT
AC-GT
GCCAT
AC-G-
```

The profile alignment outlined above represents an extension of the pairwise alignment, with the row sequence replaced by a profile. One can also replace the column sequence by a profile to align two profiles instead of two sequences. This approach is used in Clustal and many other programs for multiple sequence alignment.

One might also argue that the profile alignment has a serious problem as follows. T may be phylogenetically more closely related to some sequences than others in S. However, the profile alignment approach does not take this into consideration. This critique is justified. Unfortunately, alternative approaches by combining both phylogenetic reconstruction and multiple sequence alignment (Hein 1990, 1994; Sankoff et al. 1973) are generally too computationally intensive to be practical. However, recent advances in Gibbs sampler has suggested alternative ways for pairwise sequence alignment (Zhu et al. 1998) and multiple sequence alignment conditional on a phylogenetic tree (Holmes and Bruno 2001; Jensen and Hein 2005).

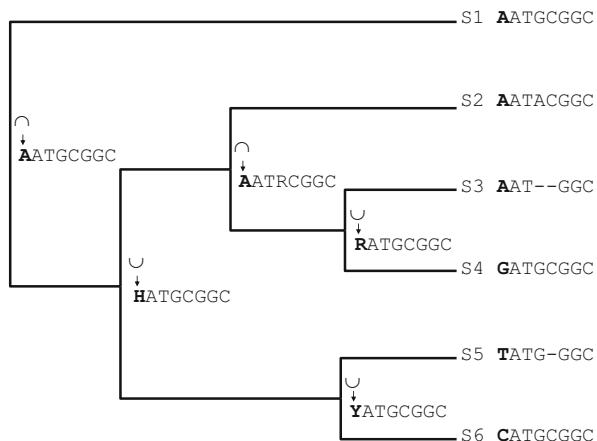
### ***3.2 Replacing a Sequence Profile by a Reconstructed Ancestral Sequence***

There is a major problem with representing ancestral nodes by sequence profiles. With the increasing number of sequences, the identity of each site is gradually eroded. For example, the frequencies at each site become progressively more and more similar to the global frequencies when more and more divergent sequences are included. For this reason, representing ancestral node with reconstructed sequences may have two advantages. First, it may alleviate the problem of information erosion. Second, aligning two sequences with ambiguous codes is simpler than aligning two profiles. I will illustrate this with ancestral sequences reconstructed from a rapid but very rough algorithm, the Fitch algorithm (Fitch 1971). There are more advanced algorithms available.

The Fitch algorithm (Fitch 1971) reconstructs the ancestral sequences based on the parsimony principle, illustrated for the first sequence site in Fig. 2.10. For each site, the ancestral state is the intersection of the daughter sequence states if the intersection is not empty and is the union of the daughter states if the intersection is empty. For example, if the daughter sequence states are Y (which stands for C or T) and C, respectively, then the ancestral state is C (the intersection of {C, T} and {C}). If the daughter sequence states are R (which stands for A or G) and T, respectively, then the intersection is empty, and the ancestral state is then the union of {A,G} and {T}. The letter W is used to represent either A, T, or G (Table 2.2) according to the International Union of Pure and Applied Chemistry (IUPAC).

Programmers use bitwise operations to achieve high speed in obtaining the union and intersections. First, we represent nucleotides and the ambiguous codes ACMGR...N by 1, 2, 3, 4, 5, ..., 15 as shown in Table 2.15. We can now obtain the union and intersection by using two bitwise operations, the bitwise OR

**Fig. 2.10** Reconstructing ancestral sequence states along the tree with the first site illustrated. “R” stands for purine, “Y” for pyrimidine, and “H” for either A, C, or T

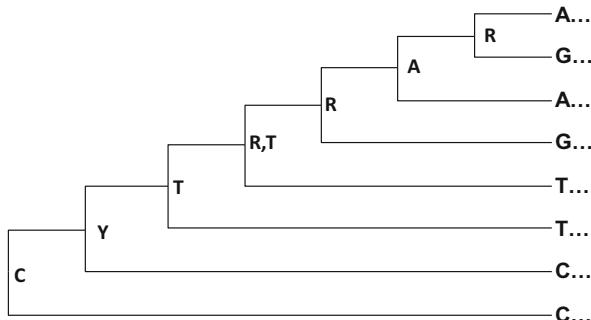


**Table 2.15** Nucleotides (Nuc), their ASCII code, assigned decimal (binary) values, and their meanings

Nuc	ASCII	Value	Meaning
A	65	1 (00000001)	A
C	67	2 (00000010)	C
M	77	3 (00000011)	A or C
G	71	4 (00000100)	G
R	82	5 (00000101)	A or G
S	83	6 (00000110)	C or G
V	86	7 (00000111)	A or C or G
T	84	8 (00001000)	T
W	87	9 (00001001)	A or T
Y	89	10 (00001010)	C or T
H	72	11 (00001011)	A or C or T
K	75	12 (00001100)	G or T
D	68	13 (00001101)	A or G or T
B	66	14 (00001110)	C or G or T
N	78	15 (00001111)	G or A or T or C

(represented by “l” in C and “OR” in Visual Basic) and bitwise AND (represented by “&” in C and “AND” in Visual Basic). Thus, the union of A (represented in binary as 00000001, Table 2.15) and G (represented in binary as 00000100, Table 2.15) is  $00000001 \mid 00000100 = 00000101$ , which equals a decimal value of 5 for R (for either A or G according to the IUPAC coding). Similarly, the union of C and T is  $00000010 \mid 00001000 = 00001010$ , which equals a decimal value of 10 for Y (for either C or T according to IUPAC coding). The union of R and Y is N (any of the four nucleotides), obtained by  $00000101 \mid 00001010 = 00001111$  which equals the decimal value of 15.

**Fig. 2.11** An 8-species topology with one nucleotide site shown. A profile for the root node would have equal nucleotide frequencies for the site (i.e., a fully ambiguous site), whereas a reconstructed sequence for the root node will have nucleotide C



Similarly, the intersection can be obtained by using the bitwise AND, which combines corresponding bits of two binary numbers in such a way that the result is 1 if both operand bits are 1 and is 0 otherwise. Thus, the intersection of A (00000001) and R (00000101) is 00000001 (= A & R) which means A (Table 2.15), and the intersection of Y (00001010, representing C or T) and H (00001011, representing A, C, or T) is 00001010, i.e., Y (= Y & H).

With these bitwise operations, the ancestral state in the parental node is the intersection of the two daughter states if the intersection is not empty and is the union of the two daughter states if the intersection is empty. Bitwise operations are extremely fast for any programming languages.

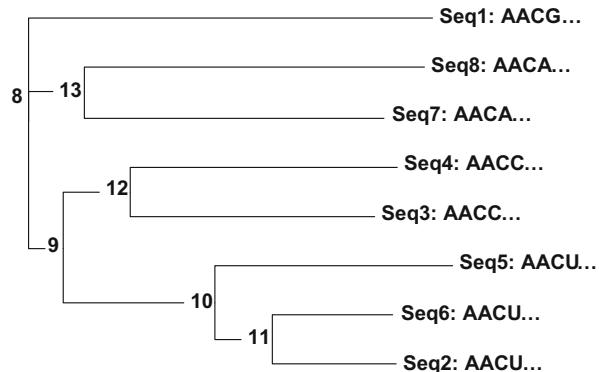
One advantage of a reconstructed sequence over a profile is illustrated in Fig. 2.11 with a topology and a nucleotide site shown. A sequence profile for the root node will have the site fully ambiguous, whereas the site is C when reconstructed by the Fitch algorithm.

In summary, for multiple sequence alignment with a guide tree, we perform pairwise alignment between sister taxa from the leaves, reconstruct ancestral sequences for their parental nodes by using the bitwise operations, and traverse down the tree to the root and align the sequences for the sister nodes until all sequences have been aligned. When a gap is inserted in the sequence for the ancestral node, then sequences for all the descendent leaf nodes will receive the gap character as well.

### 3.3 Multiple Alignment with a Guide Tree

Multiple alignment with a guide tree consists of three steps. The first is to perform all pairwise alignments by dynamic programming. With  $N$  sequences, there are  $N(N-1)/2$  pairwise alignments, leading to a triangular matrix of alignment scores. The second step is to construct a guide tree by using the alignment score matrix as a sequence similarity matrix in conjunction with a clustering algorithm. Alternatively, one can convert the similarity matrix into a distance matrix and then use either

**Fig. 2.12** An example of a guide tree for multiple sequence alignment of eight sequences, called leaves or terminal nodes. The internal nodes are numbered from 8 to 13 (with terminal nodes, or leaves, numbered from 0 to 7)



UPGMA or neighbor-joining method (Saitou and Nei 1987) to build a guide tree such as the one shown in Fig. 2.12. The third step is to traverse the node to align sequences by pairwise alignment and profile alignment. Some multiple alignment programs such as MAFFT (Katoh et al. 2005; Katoh and Toh 2010) and MUSCLE (Edgar 2004) take an iterative approach. That is, after obtaining the first multiple alignment, they will build a tree from this first multiple alignment and then use this tree (which is typically better than the original guide tree from pairwise alignment scores) to guide the multiple alignment again. This can be repeated a number of times until an index measuring the quality of multiple alignment does not show further improvement.

The multiple alignment starts from the most similar sequences. So we move to internal node 11 and align Seq2 and Seq6 (Fig. 2.12). We then move to internal node 10 and align Seq5 against either an ancestral sequence for node 11 or a sequence profile representing aligned Seq2 and Seq6 using the method outlined in Fig. 2.9. A profile or an ancestral sequence for node 10 is then created to represent the three aligned sequences (Seq2, Seq6, and Seq5). Moving to internal node 9, we found one of its two child nodes (internal node 12) having two child nodes with unaligned sequences (Seq3 and Seq4) which are then first aligned by using the dynamic programming method. A profile or an ancestral sequence for node 12 is then created to represent the aligned Seq3 and Seq4. This profile is then aligned against the profile presenting the aligned Seq2, Seq6, and Seq5. The process continues until all sequences are aligned.

It is easy to see why we should start with the most similar sequences because any alignment error will be propagated in subsequent alignment. Obviously, a wrong guide tree will bias the subsequent alignment which in turn will bias subsequent phylogenetic reconstruction based on the alignment. Unfortunately, a guide tree built from alignment scores is typically a very poor tree. For this reason, it is better to input a well-established tree, whenever available, as a guide tree for multiple sequence alignment. All frequently used multiple alignment programs allow users to input their own guide tree. My program DAMBE (Xia 2013) can build high-quality trees based on pairwise alignment with simultaneously estimated

evolutionary distances (“simultaneous estimation” means the distances are estimated using all pairs of sequences instead of just one pair) by using the PhyPA method (Xia 2016). One can use the PhyPA function to build a tree and then input to a multiple sequence alignment program as a guide tree.

### 3.4 The Inconsistency Problem with Pairwise Alignments

Pairwise alignment can be inconsistent (Fig. 2.13. P. Foster, pers. comm.). Given the three amino acid sequences (S1 to S3) in Fig. 2.13a, we will get three pairwise alignments shown in Fig. 2.13b with the scoring scheme defined by BLOSUM62, gap open equal to 20 and gap extension equal to 2. The residue W in S1 at position 3 (designated by  $W_{13}$ , where the first subscript 1 indicates the first sequence and the second subscript 3 indicates the third site in S1) is inferred to be homologous to  $W_{22}$  based on the pairwise alignment between S1 and S2 and homologous to  $W_{33}$  based on the pairwise alignment between S1 and S3 (Fig. 2.13b). These two homologous site pairs ( $W_{13}/W_{22}$ ,  $W_{13}/W_{33}$ ) imply a homologous site pair  $W_{22}/W_{33}$  which however is not true in the pairwise alignment between S2 and S3 (Fig. 2.13b) where site  $W_{22}$  pairs with  $K_{32}$  instead of  $W_{33}$ . This inconsistency in site homology identification in pairwise alignment disappears in a multiple alignment (Fig. 2.13c) obtained with the same scoring scheme. When pairwise alignment is derived from the multiple alignment (Fig. 2.13d), then two inferred homologous pairs ( $W_{13}/W_{22}$ ,  $W_{13}/W_{33}$ ) imply a homologous site pair  $W_{22}/W_{33}$  which is observed in Fig. 2.13d.

(a)	(c)
S1 DKWWGAAP	S1 DKWWGA--AP
S2 DWWGARRAP	S2 D-WWGARRAP
S3 DKWARRAP	S3 DKW--ARRAP

(b)	(d)
S1 DKWWGA--AP	S1 DKWWGA--AP
S2 D-WWGARRAP	S2 D-WWGARRAP
S3 DKWARRAP	S3 DKW--ARRAP

S1 DKWWGAAP	S1 DKWWGA--AP
S3 DKWARRAP	S3 DKW--ARRAP
S2 DWWGARRAP	S2 D-WWGARRAP
S3 DKW-ARRAP	S3 DKW--ARRAP

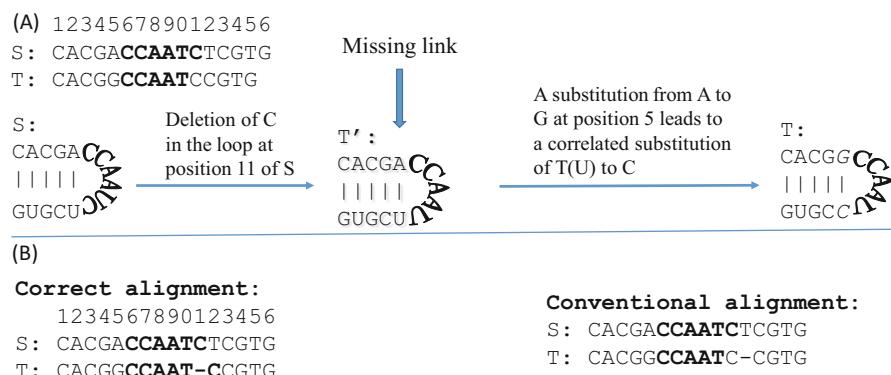
**Fig. 2.13** Identification of homologous sites in pairwise and multiple alignment, illustrating the problem of inconsistency in site homology identification associated with pairwise alignment (P. Foster, pers. comm.). (a) Three amino acid sequences S1 to S3; (b) three PSAs from the three sequences in (a), with scoring scheme of BLOSUM62, gap open equal to 20 and gap extension equal to 2; (c) MSA of three sequences in (a) with the same scoring scheme; (d) three PSAs derived from the MSA in (c)

For any particular pair of sequences S1 and S2, other sequences may contribute both phylogenetic information and noise to the identification of homologous sites between S1 and S2. With low sequence divergence, phylogenetic information contributed by other sequences overwhelms noise and reduces the problem of inconsistency in homologous site identification often seen in pairwise alignment. Thus, when a reliable multiple alignment can be obtained, one should use this multiple alignment for phylogenetic inference instead of pairwise alignment (Xia 2016). However, when sequences are so diverged that additional sequences will contribute mostly noise instead of phylogenetic information, then building trees with only pairwise alignment might be more preferable (Xia 2016).

## 4 Sequence Alignment with Secondary Structure

Figure 2.14 shows two sequences, S and T, being a fragment of a fictitious rRNA gene from two related species. The fragment forms a stem-loop structure. For simplicity, suppose that T was derived from S through the intermediate “T” in two steps. First, the C at position 11 was deleted. Second, a substitution from A to G at position 5 leads to a correlated substitution from T(U) to C to maintain the stem of length 5. The resulting sequence is T (Fig. 2.14a).

If we constrain the alignment with the secondary structure information, i.e., the first five and the last five nucleotides of S, being respectively homologous to the first five and the last five nucleotides of T, then the resulting alignment (designated as the correct alignment in Fig. 2.14b) correctly identifies the gap at position 11. However, any currently used alignment program based on linear sequence information, with any sensible scoring scheme, would recover the “conventional alignment” (Fig. 2.14b) that identified the gap at position 12. The two Cs at position 11 in the “conventional alignment” are nicely aligned but are not homologous given the



**Fig. 2.14** Illustration that the correct alignment may differ from the optimal alignment. Note that T becomes U in the secondary structure

evolutionary steps given above in generating T. It is easy to see that the conventional alignment will have a greater alignment score than the correct alignment and consequently is more “optimal.” Thus, optimal alignment (the alignment with the largest alignment score) may not necessarily be the correct alignment.

Aligning rRNA genes with the constraint of secondary structure has now been frequently used in practical research in molecular evolution and phylogenetics (Hickson et al. 2000; Kjer 1995; Notredame et al. 1997; Xia 2000; Xia et al. 2003a). However, one cannot always assume that rRNA secondary structure is stable over evolutionary time in different lineages. It is now well established that variation in rRNA secondary structure is strongly affected by the optimal growth temperature in prokaryotes (Galtier and Lobry 1997; Hurst and Merchant 2001; Nakashima et al. 2003; Wang and Hickey 2002; Wang et al. 2006). Two programs can automate sequence alignment guided by secondary structure: PROMALS3D (Pei et al. 2008) for protein sequences and PHASE (Gowri-Shankar and Rattray 2007) for rRNA and tRNA sequences.

## 5 Align Nucleotide Sequences Against Amino Acid Sequences

During the evolution of protein-coding genes, an entire codon or multiple codons may be deleted or inserted, but it is much rarer to see an insertion or deletion (often abbreviated as indel) of one or two nucleotides because such indel events lead to frameshifting mutations that almost always disrupt the original protein function and are strongly selected against. However, alignment of protein-coding nucleotide sequences often produce indels of one or two bases as alignment artifacts. The correctly aligned sequences should have complete codons, not one or two nucleotides, inserted or deleted.

One way to avoid the above alignment problem is to align the protein-coding nucleotide sequences against amino acid sequences, which was implemented in software DAMBE since 1999 (Xia 2001; Xia and Xie 2001b). The approach obviously requires amino acid sequences which can be obtained in two ways. First, if you have protein-coding nucleotide sequences of good quality, then you can translate the sequences into amino acid sequences, which can be done automatically in DAMBE which implements all known genetic codes for translating protein-coding sequences from diverse organisms. Second, if you are working on nucleotide sequences deposited in GenBank, then typically you will find the corresponding translated amino acid sequences.

The alignment of protein-coding nucleotide sequences is typically done in three steps. First, the nucleotide sequences are translated into amino acid sequences. These amino acid sequences are then aligned, and the nucleotide sequences are then aligned against the aligned amino acid sequences.

Here is a simple illustration. Suppose we are to align the following two protein-coding sequences designated S1 and S2, respectively:

S1 ATG CCG GGA CAA  
S2 ATG CCC GGG ATT CAA

Step 1: Translate the sequences into amino acid sequences (one-letter notation) to get:

S1 MPGQ  
S2 MPGIQ

Step 2: Align the amino acid sequences:

S1 MPG-Q  
S2 MPGIQ

Step 3. Align the protein-coding nucleotide sequences against aligned amino acid sequences. This is done by essentially mapping the codon sequences to the aligned amino acid sequences. Keep in mind that a gap in the aligned amino acid sequences corresponds to a triplet gap in nucleotide sequences:

S1 ATG CCG GGA --- CAA  
S2 ATG CCC GGG ATT CAA  
\*\*\*\* \* \* \* \* \*

This alignment, designated as Alignment 1, has ten matches, two mismatches, and one gap of length 3. Recall that the main objective of sequence alignment is to identify homologous sites and it is important to note that different alignments may lead to different interpretations of sequence homology. With the alignment above, site 6 of the two sequences (G in S1 and C in S2) is interpreted as a homologous site, so is site 9 (A in S1 and G in S2). These interpretations are not established facts. They are only inferences of what might have happened.

Depending on the scoring scheme, a nucleotide-based sequence alignment, i.e., without using aligned amino acid sequences as a mapping reference, may well generate the following alignment designated Alignment 2, with 12 matches, 0 mismatch, and 2 gaps of lengths 1 and 2, respectively:

S1 ATG CC- GGG A-- CAA  
S2 ATG CCC GGG ATT CAA  
\*\*\*\* \* \* \* \* \*

Note three different interpretations of the homologous sites between Alignment 1 and Alignment 2. First, the nucleotide G at site 6 of S1 is now interpreted to be homologous to the nucleotide G at site 7 of S2. Second, the nucleotide A at site 9 of S1 is now interpreted as homologous to the nucleotide A at site 10 of S2. Which of the two alignments makes more sense? If Alignment 1 is correct but we used a nucleotide-based alignment method and end up with Alignment 2, then the estimation of the genetic distance between the two sequences will be biased. The genetic distance measures the evolutionary dissimilarity between two sequences, often

estimated by ignoring the indel sites. It is often used as an index of sequence divergence time in molecular phylogenetics, when calibrated by fossils with known divergence time. In this particular case, if we perform site-wise deletion of indels, then S1 and S2 would appear more similar to each other in Alignment 2 than in Alignment 1. Biased estimation of the genetic distance often results in failure in molecular phylogenetic reconstruction.

Given that a protein-coding gene is unlikely to remain functional after two consecutive indel mutations as in Alignment 2, we may argue that Alignment 1 based on the alignment of amino acid sequences is better than Alignment 2. In addition to eliminating indels of length 1 or 2 as alignment artifacts, aligning protein-coding amino acid sequences against aligned amino acid sequences also has the benefit associated with the fact that amino acid sequences can be aligned more reliably. Aligning amino acid sequences are also faster than aligning the corresponding codon sequences because the amino acid sequences are only 1/3 as long as the corresponding codon sequences. The size of the match-mismatch matrix has little effect of the speed of alignment because computers take a memory address to retrieve a value in a matrix and this is the same for a  $4 \times 4$  as for a  $20 \times 20$  matrix.

However, there are also cases where a nucleotide-based alignment may be better. Now consider the following two protein-coding sequences:

```
3 6 9 12 15
S1 ATG CCC GTA TAA
S2 ATG CCC GTG TTA TAA
```

Of the following three alignments, designated as Alignment 1, Alignment 2, and Alignment 3, all involving one indel of length 3, which one makes more sense to you?

Alignment 1 :

```
S1 ATG CCC GTA --- TAA
S2 ATG CCC GTG TTA TAA
*** *** * * ***
```

Alignment 2 :

```
S1 ATG CCC GT- --A TAA
S2 ATG CCC GTG TTA TAA
*** *** * * ***
```

Alignment 3

```
S1 ATG CCC G- --- TA TAA
S2 ATG CCC GTG TTA TAA
*** *** * * ***
```

Alignment 1 is the outcome of aligning nucleotide sequences against aligned amino acid sequences, and the other two alignments are from nucleotide-based alignments. The three alignments represent three alternative hypotheses, all involving a gap of length 3, but differ in the position of the gap. Alignment 1 has only

11 matches, whereas the other two alignments each have 12 matches. In this case, the two hypotheses represented by Alignments 2 and 3 are better than Alignment 1 based on the parsimony criterion. Alignment 1 implies a triplet indel and an A↔G substitution, whereas Alignments 2 and 3 each require only a triplet deletion. However, from Alignment 1, one can easily obtain Alignments 2 or 3 by automated post-alignment adjustment (Xia 2016).

## Postscript

A student of mine once told me that she always found it miraculous to see human sequences from diverse ethnic groups aligning so well against each other. The various forms of nature's creation are so intricately and closely related to each other, and people of different nations are so similar to each other genetically that she found it a mystery that many so-called civilized humans were still so ready and willing to kill and torture each other. "Aren't we killing ourselves by killing people with their genomes essentially identical to ourselves?", she asked.

I think we are, and we should stop. I can't resist the temptation of concluding this chapter with a quote from Albert Einstein (Einstein et al. 1931, p. 6) when he was discussing man's relationship to others: "This subject brings me to that vilest offspring of the herd mind – the odious militia. The man who enjoys marching in line and file to the strains of music falls below my contempt; he received his great brain by mistake – the spinal cord would have been amply sufficient. This heroism at command, this senseless violence, this accursed bombast of patriotism – how intensely I despise them! War is low and despicable, and I had rather be smitten to shreds than participate in such doings."

Einstein signed his last letter, 1 week before his death, giving permission to have his name on a manifesto urging all nations to give up nuclear weapons. How would he despise those politicians who bend on arms races!

Yet in spite of Einstein's antiwar stance, his successes have often been depicted with military analogies, such as how the established castles of physics came tumbling down at the trumpet of his theory of relativity. In a similar vein, Louis Pasteur, who had spent all his life saving others, was accredited with military and strategic genius that "he had something of Napoleon in his way of always taking the initiative, of suddenly changing the terrain, of showing up where he was least expected, of suddenly concentrating his forces in a narrow sector to make the breakthrough, ..... Without a doubt, Pasteur's saga was as stirring as Napoleon's!" (Jacob 1988, p. 248).

Why does a life of saving have to be glorified with a life of killing?

# Chapter 3

## Position weight matrix and Perceptron



### 1 Introduction

A genome would be dead if it does not have many signals (or genetic switches) recognized and acted upon by RNAs and proteins. Most genetic switches are in the form of sequence motifs that interact with proteins (Ptashne 1986). This chapter covers position weight matrix (PWM) and perceptron. Both are implemented in DAMBE (Xia 2013). PWM is for characterizing the pattern of a motif present in a set of sequences of equal length (e.g., a set of 5' splice sites, each with 5 nucleotides from the exon site and 12 nucleotides from the intron site). The objective is to find which site contains strong nucleotide or amino acid bias above background frequencies. In contrast, perceptron is used for classifying two groups of sequences of equal length, one being the positive group and the other the negative group. The objective is to find which sites contribute to the discrimination of the two groups of sequences.

PWM (Claverie and Audic 1996; Hertz et al. 1990; Hertz and Stormo 1999; Staden 1984; Stormo et al. 1986), also known as position-specific weight matrix (PSWM) or position-specific scoring matrix (PSSM), is one of the key bioinformatic tools used extensively in characterizing and predicting motifs in nucleotide and amino acid sequences. The popularity of PWM has been further increased since its implementation as a component in PSI-BLAST for detecting remote homology (Bhagwat and Aravind 2007), which is frequently used to generate PWM for motif characterization and prediction (Hwang et al. 2007; Kim and Park 2004; Rashid et al. 2007; Sim et al. 2005).

PWM has been applied extensively to studies of cis-regulatory elements in the genome such as translation initiation sites (Li and Leong 2005), transcription initiation sites (Grech et al. 2007), transcription factor binding sites (Aerts et al. 2007; Fickett 1996; Frank et al. 2005; Hertzberg et al. 2007; Jin et al. 2004b; Kamalakaran et al. 2005; Lemay and Hwang 2006; Ostrin et al. 2006; Yuan et al. 2006), intron splicing sites in yeast (Ma and Xia 2011) and in mammals (Vlasschaert et al. 2016), whole-genome identification of transcription units (Kobayashi et al. 2007), and

whole-genome screening of transcription regulatory elements (Monteiro et al. 2008; Young et al. 2008). The PWM scores (PWMSs) for individual motifs have been found to be useful as a measure of the motif strength (Dewey et al. 2006; Zheng et al. 2005). For example, yeast intron-containing genes with strong splice sites (high PWMS values) are more likely to recruit spliceosome proteins than those with weak ones (Ma and Xia 2011).

PWM has been used not only as an independent tool for summarizing and predicting sequence motifs, but also as a key component in more advanced bioinformatic algorithms such as the variable-order Bayesian network (Ben-Gal et al. 2005), Gibbs sampler (Lawrence et al. 1993; Mannella et al. 1996; Xia 2012b) and related algorithms based on the Monte Carlo method (Liang et al. 2008), MEME (Bailey et al. 2006), and support vector machines (Hua and Sun 2001; Kumar and Shelokar 2008; Kuznetsov et al. 2006; Vert 2002; Zien et al. 2000). While PWM has been used mainly to characterize and predict motifs in nucleotide sequences, recent studies have demonstrated its potential in characterizing and predicting functional protein motifs (Brumme et al. 2004; Chakrabarti and Lanczycki 2007; Delorenzi and Speed 2002), signal peptides (Hiller et al. 2004), and protein-protein binding sites (Obenauer et al. 2003). In particular, the method was successful in predicting tyrosine sulfation sites (Lin et al. 2003; Liu et al. 2008; Nicholas et al. 1999; Yu et al. 2002).

A PWM-based sequence analysis involves three types of output: the site-specific frequency distribution, the PWM itself, and PWMS for each input sequence (and optionally PWMS resulting from scanning new sequences with the trained PWM). I have also included statistical significance tests appropriate for each of the three types of PWM output, because such tests are often not found in literature on PWM. Understanding PWM is essential for materials in the next chapter on Gibbs sampler.

One of previously unexplored utilities is to use PWM to refine a multiple sequence alignment. I include an illustration for this application and hope that it serves as a starting point for future development. PWM, as well as its various derivatives and Gibbs sampler that uses PWM as a central component, is implemented in DAMBE (Xia 2013, 2017d) which is a comprehensive workbench for a variety of molecular sequence analysis.

The perceptron is a binary classifier, being one of the simplest artificial neural networks invented in 1957 at the Cornell Aeronautical Laboratory by Frank Rosenblatt (Rosenblatt 1958). I will abbreviate “artificial neural networks” as just “neural networks” from now on. Perceptron is also one of the earliest neural networks that became widely known and contributed to the increased research effort on neural networks in the early 1960s. The fancy name must have contributed to its early popularity. Had it been labeled just as “a novel classifier” or “a linear classifier,” it would probably never see its light of day. Note that these latter labels are in fact far more meaningful than the word “perceptron.” It seems that human beings, through some miraculous mechanisms of evolution, have developed a particular fascination with words suffixed with “-tron.” Whoever can understand and exploit this human fascination seems to be one step closer to success. Evolution favors animals with traits to exploit preexisting fascination of other animals,

especially females of the same species, to increase their chance of survival and reproduction, leading to the proposal of the sensory exploitation hypothesis (Arnqvist 2006; Ryan et al. 1990; Sakaluk 2000).

## 2 Position Weight Matrix (PWM)

### 2.1 Basic Concepts of PWM

The simplest input for a PWM-based method consists of an aligned set of sequences and the specification of the background (prior) frequencies. The main output of PWM, other than the PWM itself, consists of the site-specific information content and the motif information content (Hertz and Stormo 1999) as well as PWMS for individual motifs, together with the associated statistical tests.

We first illustrate the PWM method by applying it to the 246 5' splice sites (5'SS) of yeast (*Saccharomyces cerevisiae*) introns, each represented by 5 nucleotide sites on the exon side and 12 nucleotide sites on the intron side (Table 3.1). The yeast genome has only major-class introns all starting with GU, but there are many GU dinucleotides in both exons and introns. How would spliceosome recognize the particular GU at the 5'SS as the start of the intron? One may hypothesize that there

**Table 3.1** Site-specific frequencies and position weight matrix (PWM) for 246 5' splice sites (each represented by 5 sites on the exon side and 12 sites on the intron side)

Site	A	C	G	U	$\chi^2$	P	A	C	G	U
1	83	30	49	84	10.10	0.0177	0.0525	-0.6332	-0.0260	0.3143
2	103	44	46	53	10.04	0.0182	0.3613	-0.0878	-0.1162	-0.3434
3	121	36	38	51	30.01	0.0000	0.5920	-0.3739	-0.3886	-0.3981
4	122	38	33	53	32.16	0.0000	0.6038	-0.2969	-0.5893	-0.3434
5	81	40	81	44	28.33	0.0000	0.0177	-0.2238	0.6933	-0.6081
6	0	1	245	0	948.34	0.0000	-6.6464	-5.0056	<b>2.2841</b>	-6.6469
7	0	9	0	237	582.23	0.0000	-6.6464	-2.3190	-6.6480	<b>1.8032</b>
8	239	1	2	4	462.46	0.0000	<b>1.5693</b>	-5.0056	-4.3320	-3.8633
9	16	24	1	205	387.81	0.0000	-2.2655	-0.9496	-5.0680	<b>1.5946</b>
10	2	0	243	1	928.96	0.0000	-4.8476	-6.6483	<b>2.2723</b>	-5.3416
11	9	7	2	228	521.06	0.0000	-3.0427	-2.6612	-4.3320	<b>1.7475</b>
12	87	15	34	110	53.66	0.0000	0.1198	-1.6111	-0.5468	0.7006
13	84	49	30	83	11.71	0.0085	0.0696	0.0659	-0.7246	0.2971
14	111	39	33	63	19.09	0.0003	0.4684	-0.2599	-0.5893	-0.0969
15	106	38	31	71	17.24	0.0006	0.4024	-0.2969	-0.6781	0.0738
16	92	30	40	84	13.69	0.0034	0.1997	-0.6332	-0.3155	0.3143
17	80	38	36	92	14.32	0.0025	-0.0001	-0.2969	-0.4655	0.4445

The  $\chi^2$  test is performed for each site against the expected background frequencies with A = 0.3279, C = 0.1915, G = 0.2043, and U = 0.2763. Sites that have been experimentally verified to be important are in bold

are more signals in the nucleotides flanking the particular GU at 5'SS. This hypothesis can be easily assessed by PWM.

The four columns on the left side of Table 3.1 headed by A, C, G, and U are the site-specific counts of nucleotides A, C, G, and U. When all site-specific counts are greater than zero, each element in the PWM, designated by  $\text{PWM}_{ij}$  (where  $i = 1, 2, 3$ , and 4 correspond to A, C, G, and U, respectively, and  $j$  is site index), is computed as

$$\text{PWM}_{ij} = \log_2 \left( \frac{p_{ij}}{p_i} \right) \quad (3.1)$$

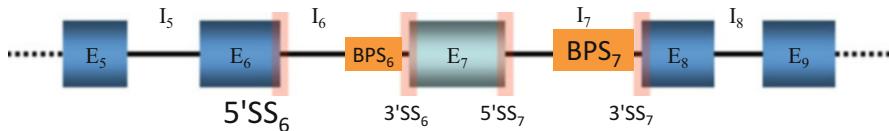
where  $p_i$  is the background frequency of nucleotide  $i$  and  $p_{ij}$  is the site-specific nucleotide frequency for nucleotide  $i$  at site  $j$  (e.g.,  $p_{A1} = 83/246$  in Table 3.1). For studying splice sites, one could use the nucleotide frequencies of intron sequences as  $p_i$ . Plotting these site-specific  $\text{PWM}_{ij}$  values graphically over sites yields the sequence logo (Gorodkin et al. 1997; Schneider and Stephens 1990). Note that  $\text{PWM}_{ij}$  is expected to be zero when there is no nucleotide bias against the background nucleotide frequencies, i.e., when  $p_{ij} = p_i$ , so it is easy to see which nucleotide at which site is overused or underused. Second, if the background nucleotide frequencies are all equal to 0.25, then the maximum  $\text{PWM}_{ij}$  value is 2, i.e., when  $p_{ij} = 1$ . However,  $\text{PWM}_{ij}$  is not defined when  $p_{ij} = 0$ , which necessitates the use of pseudocounts to avoid the potential problem of taking a logarithm of zero. Also, there is no lower bound for  $\text{PWM}_{ij}$  since it can be very small when  $p_{ij}$  approaches 0. The PWM score (PWMS) for a particular motif is computed as

$$\text{PWMS} = \sum_{i=1}^L \text{PWM}_{i,S_i} \quad (3.2)$$

where  $i$  is the site index,  $L$  is the length of the motif which equals 17 for our example shown in Table 3.1, and  $S_i$  is the nucleotide in the motif sequence. For example, PWMS for sequence UAAAAGUAUGUUUUUUU is

$$\text{PWMS} = 0.3143 + 0.3613 + 0.5920 + \cdots + 0.3143 + 0.4445 \quad (3.3)$$

We can immediately see that the splice signal at yeast 5'SS is GUAUGU (Table 3.1, last four columns). These six sites are all at the intron side, and the rest of the sites do not have very strong signals. We can use this PWM to compute PWMS for each 5'SS and expect PWMS to serve as a proxy of splicing signal strength. One may predict that mRNAs from yeast intron-containing genes (ICGs) should recruit spliceosome proteins more efficiently if they have high PWMS values. Indeed PWMS is significantly and positively correlated with the experimentally measured efficiency of spliceosome protein recruitment (Ma and Xia 2011). Furthermore, some ICGs have strongly negative values. Recall that a lack of the signal (with  $p_{ij} = p_i$ ) will lead to  $\text{PWMS} = 0$  but not strongly negative. A strongly negative PWMS value for a splice site means that the involved ICG has evolved to avoid being spliced by the yeast spliceosome. Indeed, some of such ICGs were reported to be spliced by alternative splicing pathways (Ma and Xia 2011).



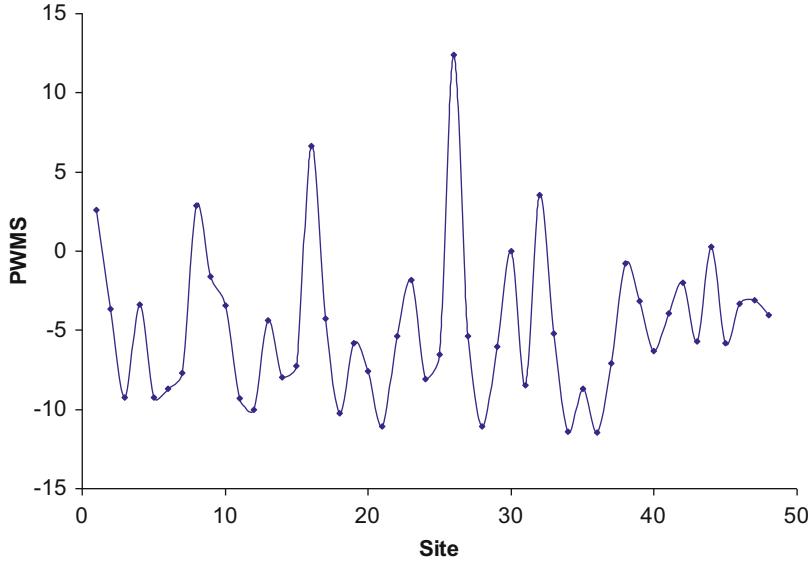
**Fig. 3.1** Partial *USP4* mRNA with schematic representation of exon-intron configuration and splice signals such as 5' and 3' splice sites (5'SS and 3'SS) and branchpoint site (BPS). Strong 5'SS<sub>6</sub> and BPS<sub>7</sub> coupled with weak BPS<sub>6</sub> and 5'SS<sub>7</sub> could lead to E7 exclusion

One may also predict that weak splice signals could contribute to alternative splicing. For example, the gene coding ubiquitin-specific protease 4 (USP4) in human and mouse has 22 exons, of which exon 7 (E<sub>7</sub>) is often skipped (Fig. 3.1). If BPS<sub>6</sub> and 5'SS<sub>7</sub> are weak, and BPS<sub>7</sub> and 5'SS<sub>6</sub> strong, then the spliceosome will treat E<sub>7</sub> and its flanking introns as one long intron so that BPS<sub>7</sub> will attack 5'SS<sub>6</sub> instead of 5'SS<sub>7</sub>, leading to skipping of E<sub>7</sub>. PWMS for 5'SS<sub>7</sub> in mouse and human *Usp4* is indeed much smaller (i.e., a much weaker splice signal) than that for 5'SS<sub>6</sub>. One may therefore generate two predictions (Vlasschaert et al. 2016). First, there should be no E<sub>7</sub>-skipping in vertebrate lineages where PWMS for 5'SS<sub>7</sub> is not smaller than that for 5'SS<sub>6</sub>. A search of *Usp4* gene in 14 vertebrate species revealed two species (zebrafish and chicken) whose PWMS for 5'SS<sub>7</sub> is not smaller than 5'SS<sub>6</sub> and indeed E<sub>7</sub> is not skipped in these two species (Vlasschaert et al. 2016). Second, there should be no E<sub>7</sub>-skipping in mouse and human if the 5'SS<sub>7</sub> is experimentally changed to the same signal as 5'SS<sub>6</sub>. This prediction is again strongly confirmed by experiments (Vlasschaert et al. 2016).

One may also use the 17-site PWM to scan yeast genome for real or “fake” splice signals (e.g., an mRNA that does not have intron but happen to have a GUAUGU segment that may fool some parts of the spliceosome). This is important to understand splicing dynamics. It is known that most spliceosome proteins in the yeast are recruited by mRNAs that contain no introns, which represents a significant waste of resources. One would predict that signals such as GUAUGU should be strongly selected against in mRNAs that do not have full-fledged real intron. Such a “fake” signal should be present either as a remnant of a real ancestral 5'SS or in genes that are not highly expressed and consequently not subject to strong selection.

Figure 3.2 shows an example of using a PWM derived from translation initiation site (a 13mer with the start codon and 5 nucleotides on both sides) from mammalian genes to scan the 5'-end of the *NCF4* gene with a 13-nucleotide window. The 13mer with the real start codon in the middle has the highest PWMS, but there are also weak initiation signals nearby. One would predict that the real initiation signal should stand out much more strongly in highly expressed genes than lowly expressed genes.

Note that PWMS is the logarithm of a likelihood ratio, or log-odds. Given a 17mer, say,  $S = \text{ACGGTACCACTACGTAAAGTT}$ , we have two hypotheses. The first hypothesis is that the 17mer belongs to a motif, constrained to have specific nucleotides at specific sites ( $\theta_{\text{Yes}}$ ), and the second is that each site in the 17mer is sampled from a nucleotide pool with no site-specific constraints ( $\theta_{\text{No}}$ ). The likelihoods of observing sequence  $S$ , given the two different hypotheses, are specified, respectively, as



**Fig. 3.2** Illustration of scanning the 5'-end of the *NCF4* gene (30 bases upstream of the initiation codon ATG and 27 bases downstream of ATG. The highest peak, with PWMS = 12.3897, corresponds to the 13mer with the real start codon ATG in the middle. Output from DAMBE (Xia 2013)

$$\begin{aligned} L_{\text{Yes}} &= p(S|\theta_{\text{Yes}}) = p_{A1}p_{C2}p_{G3}p_{G4}\cdots p_{T17} \\ L_{\text{No}} &= p(S|\theta_{\text{No}}) = p_A^5 p_C^4 p_G^4 p_T^4 \end{aligned} \quad (3.4)$$

This leads to the following that is identical to Eq. (3.2).

$$\begin{aligned} \text{PWMS} &= \log_2 \left( \frac{L_{\text{Yes}}}{L_{\text{No}}} \right) = \log_2 \frac{p_{A1}}{p_A} + \log_2 \frac{p_{C2}}{p_C} + \cdots + \log_2 \frac{p_{T17}}{p_T} \\ &= \text{PWM}_{A1} + \text{PWM}_{C2} + \cdots + \text{PWM}_{T17} \end{aligned} \quad (3.5)$$

Note that  $L_{\text{Yes}}$  and  $L_{\text{No}}$  both involve the multiplication of many  $p$  values that are typically smaller than 1, and the product could be quite small, so small as to cause rounding errors in computation. Taking the logarithms not only changes the multiplication to addition but also alleviates the problem of rounding errors.

## 2.2 Specification of the Background Frequencies

The background frequencies ( $p_i$ ) have been specified in three different ways in previous publications. The first is simply to assume equal background frequencies (Schwartz et al. 2008; Sheth et al. 2006) in characterizing splice sites with PWM.

This is equivalent to the classic sequence logo method for graphic display of site patterns (Schneider and Stephens 1990) which does not take background frequencies into consideration. In sequences with biased nucleotide frequencies, equal  $p_i$  values will generate a false site pattern when there is in fact no pattern. For example, the AT-biased background genome in the yeast implies that  $\text{PWM}_{A,j}$  and  $\text{PWM}_{T,j}$  will be greater than  $\text{PWM}_{C,j}$  and  $\text{PWM}_{G,j}$  on average even when the sequences contain no site-specific information. Similarly, the classic sequence logo will display A and T more prominently than C and G even when the sequences of interest contains no site-specific information.

The second approach to specify  $p_i$  is to compute it from the input sequences. As such, in our example,  $p_i$  can be computed from the four columns headed by A, C, G, and U on the left side of Table 3.1. This approach also has a problem. Suppose a certain motif is a poly-U sequence and all input sequences are “UUUUUUUUU.” This will generate background nucleotide frequencies with  $p_U = 1$  and  $p_A = p_C = p_G = 0$ . Note that the site-specific frequencies, given the input sequences all being “UUUUUUUU,” are  $p_{U,j} = 1$  and  $p_{A,j} = p_{C,j} = p_{G,j} = 0$ . So the resulting PWM would then suggest that the motif is not informative, which is contrary to our intuition, i.e., a stretch of UUUUUUUU conserved across a set of aligned sequences is likely to be biologically informative.

The third approach is to specify  $p_i$  according to the specific problem one wishes to solve. For example, when characterizing splice sites of introns in a particular species, one may use the nucleotide frequencies of all transcripts (including all exons and introns) annotated in the genome as the background frequencies (Dewey et al. 2006). Similarly, a study of site patterns of branchpoint sequences in introns could have  $p_i$  values computed from all intron sequences. I suggest that only the third approach be used to avoid the impression that PWM could have an infinite number of null hypotheses (each associated with a different specification of  $p_i$ ).

The computer program DAMBE (Xia 2013) offers different choices for specifying  $p_i$  in computing PWM. Similarly, the new sequence logo method allows more appropriate specification of background (prior) frequencies (Gorodkin et al. 1997). The resulting PWM for the 246 5'SS of yeast introns, with background frequencies computed from all introns, is shown in Table 3.1.

### 2.3 Specification of Pseudocounts

When some  $p_{ij}$  values are zero, as is the case in our example, Eq. (3.1) is inapplicable because the logarithm of zero is undefined. Three approaches can be taken to avoid this problem (Brown et al. 1993; Claverie 1994). The first is to compute  $p_{ij}$  by

$$p_{ij} = \frac{f_{ij} + p_i}{N + 1} \quad (3.6)$$

where  $N$  is the number of sequences,  $f_{ij}$  is the site-specific count for nucleotide or amino acid  $i$  at site  $j$ , and  $p_{ij}$  approaches  $f_{ij}/N$  with increasing  $N$ .  $\text{PWM}_{ij}$  values can then be computed by Eq. (3.1). This approach is poor when  $N$  is small.

The second approach is to use explicit pseudocounts by defining

$$\begin{aligned} f_{i,\text{pseudo}} &= \alpha f_i \\ f_{\text{pseudo}} &= \sum_{i=1}^M f_{i,\text{pseudo}} \end{aligned} \quad (3.7)$$

where  $f_i$  is the frequency of nucleotide  $i$  in all sequences and  $p_{ij}$  is then

$$p_{ij} = \frac{f_{ij} + f_{i,\text{pseudo}}}{N + f_{\text{pseudo}}} \quad (3.8)$$

It is important to keep  $\alpha$  small (e.g., 0.0001) because the expected  $\text{PWM}_{ij}$  from random sequences is 0 in Eq. (3.1). A large  $\alpha$  will substantially increase  $\text{PWM}_{ij}$  above 0 with random sequences.

The two approaches above share one main disadvantage. Suppose we have ten aligned motifs of ten amino acids each. Position 3 is occupied by amino acids K (lysine) and R (arginine) and position 5 by amino acid E (glutamic acid). The two approaches above will specify pseudocounts for positions 3 and 5 in the same way, which is unreasonable for the following reason. If position 3 requires a positively charged amino acid, and position 5 a negatively charged amino acid, then amino acids K, R, and H (histidine) should be more likely found than other amino acids at position 3, and amino acid D (aspartic acid) should be more likely found than other amino acids at position 5. By using other aligned protein sequence data of roughly the same divergence, we can derive frequency distributions for positions that require a positively charged or negatively charged amino acid and use these frequency distributions to produce pseudocounts (Brown et al. 1993). In our case, the pseudocounts at positions 3 and 5 will be assigned quite differently because the frequency distribution for a position requiring a positively charged amino acid is typically quite different from that for a position requiring a negatively charged amino acid.

PWM and PWMS can potentially be used to measure codon usage bias. For example, given the frequency of nucleotide  $i$  as  $p_i$ , the background frequency of a codon, say AGC, can be specified as  $p_A \cdot p_G \cdot p_C$  and compared to the observed frequency of AGC. Such an approach would eliminate one major weakness of commonly used codon bias indices such as CAI (Sharp and Li 1987; Xia 2007c) and Nc (Sun et al. 2013; Wright 1990).

## 2.4 Statistical Significance Tests for PWM

The PMW, be it from alignment of known motifs or from running the Gibbs sampler, needs to be assessed for its statistical significance. One continuous problem with PWM is the lack of generally applicable and accurate significance tests, either for individual sites of the motif, on PWM, or on PWMS. There are two reasons why

accurate significance tests are desirable. First, after characterizing a motif with PWM, one naturally wants to know whether the characterized PWM is significant, which sites contribute to the significance and which sequence has a PWMS that is significantly greater than random expectation. Second, after finding a significant PWM, one typically would want to use the PWM to scan other sequences to identify new motifs, and one needs a good significance test to show the reliability of the identified motif. This would reduce the number of putative sequence motifs going through experimental verification which is typically tedious and expensive (Hiard et al. 2007; Jin et al. 2007).

In short, three separate significance tests are required: one for individual sites, one for PWM per se, and one for PWMS. These tests are detailed in the following sections.

#### 2.4.1 Statistical Significance Tests for Individual Sites

The statistical significance of individual sites can be done by  $\chi^2$ -tests with type I error rate controlled for by the false discovery rate (Benjamini and Hochberg 1995; Benjamini and Yekutieli 2001). Take the data in Table 3.1, for example. The background frequencies are A = 0.3279, C = 0.1915, G = 0.2043, and U = 0.2763, which allow us to obtain expected counts of A, C, G, and T. With 17  $\chi^2$ -tests (Table 3.1), we face the problem of multiple comparisons and need to control for the familywise error rate (Nichols and Hayasaka 2003) which is synonymous to experimentwise error rate.

Designate the error rate by  $\alpha_0$ , and then the exact critical  $\alpha$  for rejection in individual tests is

$$\alpha = 1 - \left[ (1 - \alpha_0)^{1/N} \right] \quad (3.9)$$

where  $N$  is the number of tests and is equal to 17 in our case. If we set  $\alpha_0 = 0.05$ , then  $\alpha = 0.003012705$ . The Bonferroni criterion is based on the approximation that

$$\alpha = 1 - \left[ (1 - \alpha_0)^{1/N} \right] \approx \alpha_0/N \quad (3.10)$$

which leads to  $\alpha = 0.002941176$ . The second order Bonferroni  $\alpha$ , when relevant assumptions are met (Nichols and Hayasaka 2003), is based on

$$N\alpha - (N - 1)\alpha^2 = \alpha_0 \quad (3.11)$$

which leads to  $\alpha = 0.0029493634$ . In practice, these different  $\alpha$  values make little difference. In our case, all three  $\alpha$  values lead to the conclusion that the frequency distribution at sites 1, 2, 13, and 16 do not deviate significantly from the background frequencies.

The statistical protocol for controlling for the familywise error rate has been considered too conservative, and the protocol for controlling for the false discovery rate (FDR) has consequently been proposed recently (Benjamini and Hochberg 1995; Benjamini and Yekutieli 2001). The classical FDR approach (Benjamini and Hochberg 1995), now commonly referred to as the Benjamini-Hochberg procedure or simply the BH procedure, sorts  $p$  values in descending order and computes  $p_{\text{critical.BH},i}$  for the  $i$ th  $p$  value (where the subscript BH stands for the BH procedure) as

$$p_{\text{critical.BH},i} = \frac{q \cdot i}{N} \quad (3.12)$$

where  $q$  is FDR (e.g., 0.05),  $i$  is the rank of the  $p$  value in the sorted array of  $p$  values, and  $N$  is the number of tests (i.e., the number of  $p$  values). If  $k$  is the largest  $i$  satisfying the condition of  $p_i \leq p_{\text{critical.BH},i}$ , then we reject hypotheses from  $H_1$  to  $H_k$ . In our case, all the sites are statistically significant based on  $p_{\text{critical.BH},i}$  (Table 3.2).

The FDR procedure above assumes that the test statistics are independent or positively dependent (in the extreme case of perfect positive dependence, all tests are the same and there is really just one test with no multiple comparison problem). A more conservative FDR procedure has been developed that relaxes the assumption (Benjamini and Yekutieli 2001). This method, now commonly referred to as the Benjamini-Yekutieli or simply the BY procedure, computes  $p_{\text{critical.BY},i}$  for the  $i$ th hypothesis as

$$p_{\text{critical.BY},i} = \frac{q \cdot i}{N \sum_{i=1}^N \frac{1}{i}} = \frac{p_{\text{critical.BH},i}}{\sum_{i=1}^N \frac{1}{i}} \quad (3.13)$$

With  $N = 17$  in our case,  $\sum_{i=1}^N \frac{1}{i} = 3.439552523$ . Based on  $p_{\text{critical.BY},i}$ , the  $\chi^2$ -tests pertaining to sites 1 and 2 are not statistically significant (Table 3.2). The BY procedure was found to be too conservative, and several alternatives have been proposed (Ge et al. 2008). For large  $N$ ,  $\sum_{i=1}^N \frac{1}{i}$  converges to  $\ln(N) + \gamma$  (Euler's constant equal approximately to 0.57721566). Thus, for  $N = 10,000$ ,  $\sum_{i=1}^N \frac{1}{i}$  is close to 10. So  $p_{\text{critical.BY}}$  is nearly ten times smaller than  $p_{\text{critical.BH}}$ . Both FDR procedures above have been used in significance tests concerning yeast splicing sites (Ma and Xia 2011).

#### 2.4.2 Evaluating Statistical Significance of PWM When Pseudocounts Are Used

Whether a PWM represents a motif with site-specific constraints can be tested by using the  $F$  statistic (Hertz and Stormo 1999) specified in Eq. (3.9). However, the distribution of  $F$  is altered by pseudocounts as specified in Eq. (3.6) and Eq. (3.8). For example, the expectation of  $F$  is no longer zero with pseudocounts when there is no site-specific pattern.

**Table 3.2** Evaluating statistical significance of individual sites by two types of false discovery rate

Site	<i>P</i>	<i>p</i> <sub>BH</sub> <sup>a</sup>	<i>p</i> <sub>BY</sub> <sup>b</sup>
6	*0.0000000000	0.002941	0.000855
10	*0.0000000000	0.005882	0.001710
7	*0.0000000000	0.008824	0.002565
11	*0.0000000000	0.011765	0.003420
8	*0.0000000000	0.014706	0.004276
9	*0.0000000000	0.017647	0.005131
12	*0.0000000000	0.020588	0.005986
4	*0.0000004842	0.023529	0.006841
3	*0.0000013734	0.026471	0.007696
5	*0.0000030965	0.029412	0.008551
14	*0.0002619304	0.032353	0.009406
15	*0.0006307900	0.035294	0.010261
17	*0.0025004071	0.038235	0.011116
16	*0.0033589734	0.041176	0.011971
13	*0.0084455695	0.044118	0.012827
1	*0.0177349476	0.047059	0.013682
2	*0.0182291629	0.050000	0.014537

<sup>a</sup>Critical *p* based on Benjamini and Hochberg (1995)<sup>b</sup>Critical *p* based on Benjamini and Yekutieli (2001)

\*Significant by the criterion in Benjamini and Hochberg (1995)

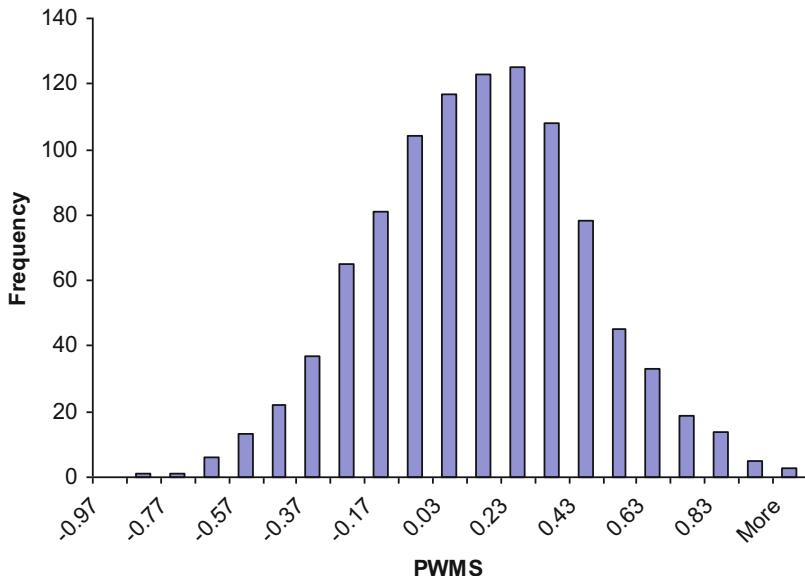
Significant by the criterion in Benjamini and Yekutieli (2001)

A more straightforward method for evaluating the significance of PWM is by resampling. With the tetranomial distribution, defined by  $(p_A + p_C + p_G + p_T)^N$ , where  $p_i$  is the nucleotide frequency of nucleotide *i*, we can obtain a new set of sequences (246 sequences of 17 nt each) and compute *F*. This is repeated for, say, 5000 times to obtain 5000 *F* values. The 95th or 99th percentile of the *F* values can be taken as critical *F* values at 0.05 and 0.01 significance levels, respectively. An observed *F* for the PWM is significant if it is greater than the critical *F*. Based on this criterion, the PWM from the 246 5'SS is highly significant. The same resampling technique can also be used to evaluate the significance of the site-specific patterns in the previous section or the significance of PWMS in the next section.

### 2.4.3 Statistical Significance of PWMS

One of the purposes of constructing a PWM is to facilitate the computation of PWMSs. For example, the PWMS for sequence UAAAGGUAUGUUUAAU, given the PWM in Table 3.1 (the four columns headed by A, C, G, and U on the right side), is simply

$$\text{PWMS} = \text{PWM}_{\text{U}1} + \text{PWM}_{\text{A}2} + \cdots + \text{PWM}_{\text{U}17} \quad (3.14)$$



**Fig. 3.3** PWMS from random sequences follows approximately the normal distribution, based on 1000 random sequences of length 17 drawn from the pool of nucleotides with frequencies of A, C, G, and T equal to 0.3279, 0.1915, 0.2043, and 0.2763, respectively. The distribution has mean equal to 0.068884 and standard deviation equal to 0.314714254

Thus, we can use the PWM to predict a new 5'SS by scanning a nucleotide sequence with a window of 17 nucleotide sites and computing the PWMS. The larger the PWMS, the more likely the 17mer is a 5'SS. However, we need to address the question of how large is large in such in silico predictions.

PWMS from random sequences follows approximately the normal distribution (Fig. 3.3), with mean 0 (or slightly greater than 0 when pseudocounts are used with a small  $\alpha$ ). The distribution in Fig. 3.3 has a mean equal to 0.068884 and a standard deviation equal to 0.314714254.

Suppose we are to use our  $4 \times 17$  PWM to scan a target sequence  $S$  of 1000 nt for a possible 5'SS. There are 984 ( $= 1000 - 17 + 1$ ) different 17mers along the sequence  $S$ , resulting in 984 PWMS values. If the maximum PWMS is 1, how statistically significant is it?

If the length of the target sequence  $S$  was only 17 nt instead of 1000 nt, then the answer would be easy. The upper 99% confidence limit for a normal distribution with mean equal to 0.0689 and standard deviation equal to 0.3147 is 0.8808 ( $= 0.06888 + 2.58 \times 0.3147$ ), which implies that a PWMS of 1 is significant at the 0.01 level. However, because our target sequence  $S$  is 1000 nt, with the maximum of PWMS equal to 1 out of a total of 984 PWMS values, we need to go a long way to evaluate the significance of this maximum PWMS value.

Suppose we perform many sampling experiments from the same normal distribution  $p(x)$  as in Fig. 3.3:

$$p(x) = \frac{e^{-(x-\mu)^2/(2\sigma^2)}}{\sigma\sqrt{2\pi}} \quad (3.15)$$

In each experiment, we sample  $N$  times to obtain  $x_1, x_2, \dots, x_N$ . The maximum  $x$  in each experiment is  $x_{\max}$ . This is equivalent to use PWM to scan a sequence to obtain PWMS<sub>1</sub>, PWMS<sub>2</sub>, ..., PWMS <sub>$N$</sub> , with the maximum PWMS designated as PWMS <sub>$\max$</sub> . What is the distribution of  $x_{\max}$ , designated as  $F(x_{\max})$ ? Note that  $x_{\max}$  is an extreme value of  $N$   $x$  values, so it is natural to call  $F(x_{\max})$  an extreme value distribution (EVD).

Extreme value distribution or EVD, also referred to as the Gumbel distribution in honor of the pioneer of the statistics of extremes (Gumbel 1958), is used in BLAST (Altschul et al. 1990, 1997) and new versions of FASTA (Pearson 1998) to attach statistical significance to a match score between two sequences. It is also used to perform significance tests involving PWM (Claverie 1994; Claverie and Audic 1996; Hertz and Stormo 1999). Here I will outline the mathematical framework of EVD pertaining to PWMS.

The probability of getting an  $x$  value smaller than  $x_{\max}$  is

$$G(x < x_{\max}) = \int_0^{x_{\max}} p(x) dx \quad (3.16)$$

Note that  $x_{\max}$  can be either  $x_1, x_2, \dots, x_N$ , with  $N$  possibilities. ( $N-1$ )  $x_i$  values are smaller than  $x_{\max}$  in each experiment. This leads us to

$$F(x_{\max}) = Np(x_{\max})G(x < x_{\max})^{N-1} \quad (3.17)$$

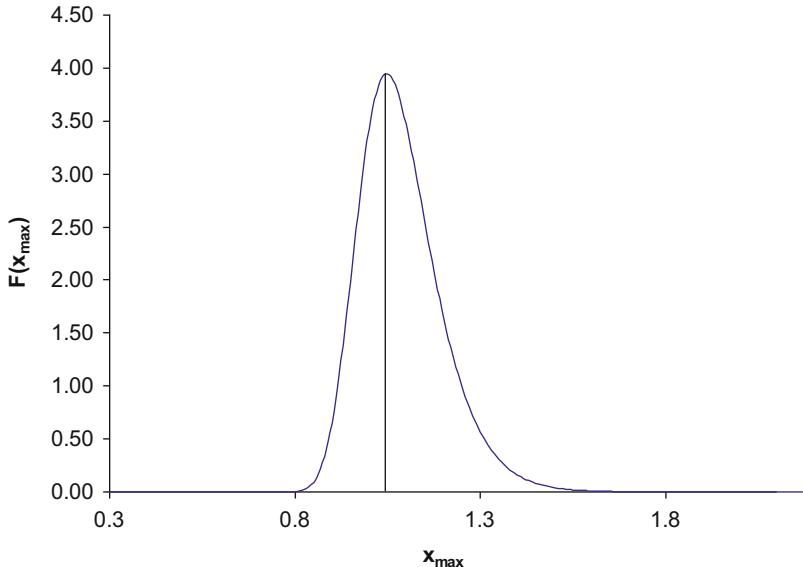
which is plotted for  $\mu = 0.068884$ ,  $\sigma = 0.314714254$ , and  $N = 984$  (Fig. 3.4). Compared to the distribution of  $p(x)$  in Fig. 3.3, the distribution of  $F(x_{\max})$  has been shifted substantially to the right and peaks at  $x_{\max} = 1.05$ .

Now we can answer the question of whether our observed  $x_{\max} = 1$  is statistically significant. The probability of observing an  $x_{\max}$  value equal to 1 or greater is

$$p(x_{\max} \geq x_{\text{obs}}) = \int_{x_{\text{obs}}}^{\infty} F(x_{\max}) dx_{\max} \quad (3.18)$$

which is approximately 0.7986, i.e., it is not statistically significant.

A much simpler, but likely less accurate, method based on  $p(x)$  only without deriving Eqs. (3.16), (3.17), and (3.18) is to use the Bonferroni criterion in Eq. (3.10). With  $\alpha_0 = 0.05$ ,  $\alpha = \alpha_0/986 = 0.00005081$  which requires a PWMS value equal to 1.292076814 to be marginally significant, given  $\mu = 0.068884$  and



**Fig. 3.4** Extreme value distribution as specified in Eq. (3.17), with  $\mu = 0.068884$ ,  $\sigma = 0.314714254$ , and  $N = 984$

$\sigma = 0.314714254$ . As our observed maximum PWMS is  $1 < 1.292076814$ , it is not significant at the 0.05 significance level.

In summary, a PWM-based sequence analysis involves three types of output: the site-specific deviation from the background frequencies, the position weight matrix itself, and the position weight matrix score for each input sequence. The significance of the first can be evaluated with  $\chi^2$ -tests using the false discovery rate as the criterion for rejection of the null hypothesis, the second by the resampling method, and the third by statistics based extreme value distribution. These tests have been implemented in the most recent versions of DAMBE (Xia 2001; Xia and Xie 2001b).

## 2.5 Using PWM to Refine Multiple Sequence Alignment

Suppose we have the alignment in Fig. 3.5, with 17 sequences of 57 aligned sites (19 codons). Even with eyeballing, we can see that the first codon on the 5' side of the indel in sequences 2–4 is GTN (coding for valine). Shifting them to the right of the indel would improve the alignment. However, moving a single nucleotide from 5' side of the gap (e.g., the bolded T and A in Fig. 3.5) to the 3' side of the gap tends to worsen the alignment. PWM can transform this eyeballing and manual adjustment to automation.

A PWM derived from the sequences (Table 3.3) allows us to quickly assess the PWMS contribution from the codon at the 5' side of indel. For sequence

	10	20	30	40	50	
FauNEOPT	GGAAAACGTAAAGAAATATTACCTTCAGATGTTCCACCTCCAGTAGAATTTCAC					
MgaARACH	GGTAAAAGAAATCATTAAAGAAATCGCTCCGT <b>T</b> -----					GAATTCTTCAT
ApaukNEOPT	GGCAAGAGGAACGATAAACCGGAGGGYCGCCTCCGGTA-----					GAACTGTTGAG
CpoNEOPT	GGCAAGAGGAATCAGCCCGGAGGGCGGAACGAACTGTG---					GAACTGTTGAG
PquNEOPT	GGCAAGCGGAATGGGACAACAAAGGTGAACGGTCGCGCAGCGGTGAGCTGTACGAG					
ClizYGEN	GGGAAACCGGAAGGGCACGCCCGAAAATTGGCAGGCACCCGTCGAATTCTTCAC					
HariARACH	GGAAAAAGAAAAACAACCACTCAGTTGAAAATGCTCCAGTT---					GAGTTCTTCAT
HspARACH	GGAAAACGAAAACAACAACTCAACTGAAAATGCACCAGTT---					GAATTCTTCAC
JapDIPLUR	GGCAAGAGGAAGGGCCAGCGCAGGGACTGGCAGGCCGGTGGACTCTTCCAC					
PamNEOPT	GGCAAGAGAAAGGAGACGTTACCACGTGACACACCACCTCCAGTGGATTCTAC					
ufsBRANCH	GGAAAACGTAAACCACTGTTAAAGACGCCGATTGGATCGCGCCGTCGGATTTTATCAG					
AdoNEOPT	GGGGAAAGGAAAGAAATCCAGACCCCAATTGCCCACCTCCAGTGGAGTTTATCAC					
AmaDIPLO	GGAAAGCGAAAATCAGAACATGTGAAGATGCAACCGATCAACCGAGTGAAGCTATACTAC					
Scu3SYMPH	GGGAGGAGGAAGCCCTGGTCCGAGAACCCGCCCTCCGTG---					GAGTTCTTCAG
EgigARACH	GGAAAACGAAAATTCTGTTCCACAGAAAGAAAATCCACGACCCTGCAATTCTCCAT					
Pma2ARACH	GGCAAGAGAAAGGCTTTAACACGCCGAATTCTGCTCTGTT---					GAATTTTTCCAT
StpARACH	GGCAAGAGAAAGGCTCTAAAGATCCATCTGGAAATGTTCTGTTGAGTTTTTCA					*
	** *    * **					* * * *

**Fig. 3.5** Subset of sequences from Regier et al. (2010), sites 160–294 (but with 57 sites remaining after deleting shared gaps), for illustrating alignment adjustment by PWM. Moving the bolded T or A to the 3' side of the indel does not improve the multiple alignment, but moving the entire codon GTT or GTA does

2 (MgaARACH), GTT at sites 34–36 corresponds to  $\text{PWM}_{ij}$  values of 0.3174, −0.5461, and 0.9543, respectively (Table 3.3), which add up to 0.7256. If we move GTT to the 3' side of the indel, i.e., to occupy sites 43–45, then the corresponding  $\text{PWM}_{ij}$  values are 1.1349, 1.5247, and −1.4278, respectively, which add up to 1.2318. Because  $1.2318 > 0.7256$ , moving GTT from 5' side to the 3' side of the indel improves the alignment. At this point, the PWM should be updated, and the updating will favor other GTN codons to move to sites 43–45. If the contribution of the GTT codon to PWMS is smaller at the new sites than the old sites, then we will not do any shifting but will inspect the next indel-containing sequence which is sequence 3 (ApaukNEOPT). Again, the contribution of GTA to PWMS at the original sites 37–39 is 0.0491. Moving the codon to sites 43–45 increases its PWMS contribution to 0.6627. So we will move GTA to sites 43–45 and update PWM. This procedure will eventually lead to moving all GTN codons on the 5' side of the indel (Fig. 3.5) to the 3' side of the indel. Now we again repeat the process to see if it is better to move the next codon (most CCN) from 5' side of the indel to the 3' side of the indel. The process can continue until there is no further improvement.

DAMBE (Xia 2013, 2017d) implements this PWM-based alignment-refining approach. To access this function, click “FileOpen standard sequence file” to read a file with multiple sequence alignment. Click “AlignmentRefine sequence alignment.” Sometimes one may have multiple files each with a set of aligned sequences. To refine sequence alignment in all these files, after clicking “AlignmentRefine sequence alignment,” select “Refine MSA in a set of files each with an MSA.” One

**Table 3.3** Partial PWM for sites 22–48

22	-1.1764	0.4124	0.7833	-0.0028	C	G	G	A	C	T	T	G	C	G	T	C	A	C	G
23	0.8366	0.6648	-0.9062	-4.0954	C	A	A	C	C	A	C	A	A	G	G	C	A	A	A
24	0.0261	0.1062	0.3174	-0.0028	T	A	G	G	C	A	G	A	C	C	T	C	A	G	T
25	-1.1764	0.1062	1.2831	-1.4278	T	A	G	G	G	A	G	C	G	G	G	C	C	C	C
26	-0.2753	0.8796	-0.3751	-0.0028	C	T	G	G	T	A	T	C	C	G	C	A	A	C	C
27	0.2752	0.1062	0.0123	-0.0028	A	C	Y	C	G	A	T	T	G	T	C	C	A	G	A
28	0.0261	-1.6756	1.2831	-1.4278	G	G	C	G	A	A	G	G	G	G	A	G	A	G	A
29	1.3525	-0.8173	-0.9062	-4.0954	A	C	G	G	A	A	A	A	A	A	A	A	A	A	C
30	-0.6566	0.6648	-4.0935	1.3584	T	T	C	C	C	T	A	A	C	C	T	T	C	A	T
31	0.0261	-0.2829	0.0123	0.6999	G	C	C	A	G	T	A	A	T	A	T	G	C	A	T
32	-0.2753	0.4124	0.3174	-0.0028	T	C	T	A	G	G	A	A	G	C	G	T	C	C	G
33	-0.6566	-1.6756	0.5691	1.1704	T	T	C	G	T	G	T	G	A	G	A	G	T	T	A
34	-1.1764	1.3803	0.3174	-4.0954	C	G	C	C	C	C	G	G	C	C	A	C	G	C	G
35	0.0261	1.2319	-1.7559	-0.5461	C	T	C	A	G	A	C	C	A	C	T	C	A	C	A
36	0.0261	-0.2829	-0.3751	0.9543	A	T	G	A	C	G	T	A	G	A	C	A	T	T	T
37	-1.9969	1.2319	0.5691	-4.0954	C	-	G	C	G	C	C	G	C	C	C	C	A	C	G
38	-1.9969	1.7513	-4.0935	-0.5461	C	-	T	C	C	C	C	C	C	C	A	C	C	C	T
39	0.0261	-1.6756	0.0123	0.9543	T	-	A	T	G	A	A	G	T	G	T	A	C	G	T
40	-1.9969	1.2319	0.3174	-4.0954	C	-	G	C	C	G	C	C	C	C	C	C	A	G	C
41	-4.0923	1.3803	-4.0935	0.6999	C	-	T	C	C	T	T	C	C	C	C	C	T	C	C
42	-0.2753	-0.8173	0.3174	0.3909	A	-	G	G	C	T	T	G	A	G	A	A	G	C	T
43	-4.0923	-4.0939	1.1349	-1.4278	G	-	-	G	G	-	G	G	T	G	G	-	G	-	G
44	-4.0923	-1.6756	-4.0935	1.5247	T	-	-	T	T	-	T	T	C	T	T	-	T	-	T
45	-1.9969	-0.2829	0.3174	-1.4278	A	-	-	C	C	-	G	G	G	G	-	C	-	T	-
46	-1.9969	-4.0939	1.9481	-4.0954	G	G	G	G	G	G	G	G	G	G	A	G	G	G	G
47	1.7317	-4.0939	-4.0935	-4.0954	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
48	0.8366	-1.6756	0.5691	-1.4278	A	A	A	G	A	G	A	C	A	T	G	G	A	A	G

Sequences are shown vertically, with columns 1, 2, ..., 17 corresponding to sequences in Fig. 3.5 in the same order

may also check the alignment quality before and after refinement by checking the sum-of-pair alignment scores, with high score indicating better alignment. This one can be done with DAMBE by clicking “Alignment|Evaluate a multiple alignment.” DAMBE will output all pairwise alignment scores and their sum. One may also use profile HMM (hidden Markov model) for refining sequence alignment. HMM is covered in a later chapter.

## 3 Perceptron

Perceptron has been used in bioinformatics research since the 1980s. The identification of translational initiation sites in *E. coli* (Stormo et al. 1982a) is perhaps the first publication of applying the perceptron algorithm in identifying gene features. The algorithm has also been used recently for finding the ATP-/GTP-binding motif (Hirst and Sternberg 1991).

The perceptron is conceptually similar to Fisher’s two-group linear discriminant function analysis (Fisher 1936) for continuous variables, often referred to as LDA (for linear discriminant function analysis). LDA is used when we have  $N$  variables,  $M$  cases, and a  $M \times N$  matrix, with the first  $m_1$  cases belonging to one group and the next  $m_2 (=M-m_1)$  cases belonging to the other group. The objective is to derive a linear function of the  $N$  variables to maximize the difference between the two groups (or, in the machine-learning literature, maximize the signal/noise ratio).

To use the perceptron to discriminate between two groups of sequences, we designate one group as the positive group and the other as the negative group. The objective is to obtain a weight matrix that can be used to obtain scores for the unknown sequences and assign these sequences different membership according to the scores.

### 3.1 Perceptron Algorithm

Here is how perceptron works. Suppose we have two groups of sequences designated as POS (for positive) and NEG (for negative) groups, respectively. The following two groups of sequences represent one of the simplest perceptron problems. We will use this fictitious data set to illustrate the perceptron algorithm:

POS1 ACGT  
POS2 GCGC

NEG1 AGCT  
NEG2 GGCC

In practical applications, the sequences will typically be longer than 4, and each group will typically contain a very large number of sequences. The two groups do not necessarily have to contain exactly the same number of sequences, although statistics involved would be simpler when they do. Here we will ignore the statistics part entirely and focus only on the algorithmic aspect of perceptron. Note that we will often refer to sequences in the POS group as POS sequences and those in the NEG group as NEG sequences.

Our task is to find a weighting matrix that can be used to assign a score to each sequence in such a way that sequences in the POS group will have high scores (typically larger than 0) and those in the NEG group will have low scores (typically smaller than 0). The weight matrix and sequence scores are the principal output of perceptron training. From now on, we will use symbols  $S$ ,  $PS$ , and  $W$  to designate an input sequence, the perceptron score for the input sequence, and the weight matrix, respectively. A cell in the matrix that does not contribute to the discrimination should have a value of 0, i.e., it does not contribute to classify the sequence to either the positive or negative group.

The first step is to initialize  $W$ , which is a  $4 \times L$  matrix for nucleotide sequences and a  $20 \times L$  matrix for amino acid sequences. If the two sets of sequences are linearly separable and if four nucleotides (or 20 amino acids) are all represented in every site, then initialize the cells with any value will end up with the same final result upon convergence. However, we may have discrepancies if some nucleotides or amino acids are not presented in one or more site. For example, if A is not represented at site 2 and if we initialize all cells with a value of 1, then the value of 1 in the cell corresponding to A at site 1 will never been visited or modified, contradicting our expectation that a cell with no information should have a value of 0.

The perceptron algorithm involves the iteration of two steps: (1) taking sequences from the NEG and POS groups sequentially (or randomly when a very large number of sequences are present in each group or when the perceptron cannot converge) and computing  $PS$  for each input sequence and (2) updating the values in  $W$  based on  $PS$ . For each sequence  $S$ ,  $PS$  is computed as

$$PS = \sum_{j=1}^L W_{S_j, j} \quad (3.19)$$

where  $j$  is site index and  $S_j$  is the nucleotide at site  $j$ .

Take the POS1 sequence in the fictitious example, its  $PS$  based on the initialized  $W$  in Table 3.4 is

$$PS_{POS1} = W_{A,1} + W_{C,2} + W_{G,3} + W_{T,4} = 0 \quad (3.20)$$

In fact,  $PS$  will be 0 for every sequence with the freshly initialized  $W$  with values of 0. What might be slightly confusing is the updating of  $W$ . It is done according to

**Table 3.4** The weighting matrix ( $W$ ) for the fictitious example with two sequences of length 4 in each group, initialized with values of 0

Base	1	2	3	4
A	0	0	0	0
C	0	0	0	0
G	0	0	0	0
T	0	0	0	0

The first row designates sites 1–4

the following rules (there could be slight variation of the rules in different applications):

$$\begin{aligned} W_{S,j} &= W_{S,j,j} + 1, \text{if } S \text{ is from POS group and } PS < 0 \\ W_{S,j,j} &= W_{S,j,j} - 1, \text{if } S \text{ is from NEG group and } PS \geq 0 \\ &\text{No change in } W_{S,j,j} \text{ otherwise.} \end{aligned} \quad (3.21)$$

where  $j = 1, 2, \dots, L$  and  $L$  is the sequence length (=4 in our fictitious example).

Let us start with the NEG sequences in the fictitious example. Note that you would waste computational time by starting with sequences in the POS group because the resulting  $PS$  will all be 0 and consequently, according to the rules for updating  $W$  in Eq. (3.21), no updating is made with  $PS = 0$ .

$PS$  for NEG1, i.e.,  $PS$  for  $S = AGCT$ , is 0. According to the rules for updating  $W$  in Eq. (3.21), we should update  $W$  by reducing the relevant  $W_{ij}$  values by 1. The updated  $W$  is shown in Table 3.5a, with  $W_{A,1}$ ,  $W_{G,2}$ ,  $W_{C,3}$ , and  $W_{T,4}$  in the original  $W$  (Table 3.4) reduced by 1, with updated values highlighted in bold.

The next input sequence is NEG2 (=GGCC) which has  $PS = -2$  based on the updated  $W$  in Table 3.5a. According to the rules of updating  $W$  in Eq. (3.21), no update is made, so the  $W$  matrix in Table 3.5b is the same as that in Table 3.5a.

We can proceed with POS1 and POS2 sequences. POS1 (=ACGT) has  $PS = -2$ . According to the rules of updating  $W$  in Eq. (3.21), we should add 1 to the cells corresponding to ACGT. The updated matrix  $W$  is shown in Table 3.5c. However, POS2 (=GCGC) has  $PS = 2$ . According to the rules in Eq. (3.21), there should be no updating, so  $W$  in Table 3.5d is the same as that in Table 3.5c. This ends the first round of iteration.

Now we restart with NEG1, NEG2, POS1, and POS2. We found that NEG1 and NEG2 both have negative  $PS$  and POS1 and POS2 both have positive  $PS$ . So we declare convergence, with the two groups clearly classified correctly.

One problem with the perceptron algorithm is that it may not converge, e.g., when it is applied to solving an XOR problem that will be detailed later. When convergence fails, i.e., when some sequences in the negative group repeatedly have positive values and some sequences in the positive group repeatedly have negative values, the offending sequences will be marked and then excluded from further iteration so that a final classification can be reached.

**Table 3.5** The first round of the training process in the perceptron algorithm

	Base	1	2	3	4
NEG1: AGCT, $PS = 0$ , update					
(a)	A	<b>-1</b>	0	0	0
	C	0	0	<b>-1</b>	0
	G	0	<b>-1</b>	0	0
	T	0	0	0	<b>-1</b>
NEG2: GGCC, $PS = -2$ , no update					
(b)	A	-1	0	0	0
	C	0	0	-1	0
	G	0	-1	0	0
	T	0	0	0	-1
POS1: ACGT, $PS = -2$ , update					
(c)	A	0	0	0	0
	C	0	1	-1	0
	G	0	-1	1	0
	T	0	0	0	0
POS2: GCGC, $PS = 2$ , no update					
(d)	A	0	0	0	0
	C	0	1	-1	0
	G	0	-1	1	0
	T	0	0	0	0

Updated values are highlighted in bold

Inspecting the last  $W$  (Table 3.5d), we see that sites 1 and 4 contribute nothing to the classification, i.e., the discrimination comes entirely from sites 2 and 3. Nucleotides A and T do not contribute to discrimination. If we have an input sequence such as TAAA, then the  $W$  matrix will give it a PS of 0, meaning that the trained perceptron is entirely unable to classify TAAA into either the positive or the negative group. You may think that these results are quite obvious by just looking at the four sequences. However, with many sequences and many sites, eyeballing is often inefficient.

It is important to keep in mind that clear separation of the POS and NEG groups may not necessarily mean anything biologically meaningful. For illustration, suppose you randomly generate 20 sequences of length 10 and randomly assign 10 sequences to the POS group and the rest 10 to the NEG group. If you now run perceptron, you will have a good chance of achieving a clear separation of the two groups of sequences. As the sequences and their affiliation to the groups are random, such a separation clearly does not mean anything except that some sites may differ by chance between two groups of randomly generated sequences when the number of sequences is small. It is for this reason that perceptron typically is run on two groups each with a large number of sequences.

The perceptron is ideally suited for two groups of objects (be they sequences or any other objects) that are linearly separable. There are two scenarios in which the two groups will not be separable. First, if there are sequences that are present in both the positive and the negative group, then obviously such sequences cannot have positive scores and negative scores at the same time and perceptron will not converge. Second, some sequences are not linearly separable, and one such case is the XOR problem). We will illustrate this in an intuitive way, with nucleotide sequences in the form of “ACGTXYACGT.” Suppose that the “XY” is “TT” in sequences of the POS group and can be any other non-“TT” dinucleotides in the NEG group. These two groups of sequences can be easily separated by the perceptron algorithm. Note that the XY in the sequences contains either zero, one, or two T’s, and the sequences in the positive group, with their XY containing 2 T’s, are linearly separable from sequences with their XY containing zero or one T’s.

Now suppose the XY in our sequences in the POS group contains only one T, i.e., either X or Y is a T but not both. The XY in the sequences in the negative group can either be TT or contain no T (e.g., XY = “CG”). This is one of the simplest XOR problems that a conventional perceptron cannot handle. The iteration will continue forever without convergence. Note that XY in POS sequences in this case contains only one T and they are no longer linearly separable from the NEG sequences containing either zero or two T’s. This XOR problem has plagued the first application of the perceptron algorithm to the study of the translation initiation sites in *Escherichia coli* (Stormo et al. 1982a), resulting in failure to converge on a solution to separate the sequences with the translation initiation site from those without.

### 3.2 Perceptron and XOR Problem

The perceptron has not been used often to solve problems in molecular biology and bioinformatics, and this is mainly caused by the overemphasis that perceptrons cannot deal with the XOR problem. In fact, many XOR problems can be reduced to non-XOR problems and be solved easily by perceptrons. For example, the XOR problem in the previous paragraph can be reduced to a non-XOR problem by using dinucleotide sequences, i.e., coding “ACGTXYACGT” to AC, CG, GT, TX, XY,..., GT and then solved by the perceptron method. In this case, we will have a W with dimensions  $16 \times (L - 1)$  because there are 16 possible dinucleotides.

With the availability of genomic sequences, it is easy to generate positive and negative groups for perceptron analysis. For example, if one is interested in how a ribosome is able to recognize a start codon among other AUG codons and whether the nucleotides flanking the start codon supply additional initiation signals, then the positive group could consist of sequences each with a true start codon flanked by five nucleotides on both sides, and the negative group could consist of sequences each with a non-starting AUG flanked by five nucleotides on both sides. Taking this approach for mammalian genes would readily yield the Kozak consensus RccAUGG (which can be generated by PWM as well, with just the positive group of sequences).

In one study using the perceptron algorithm to characterize the translation start site (TSS) in *E. coli* (Stormo et al. 1982b) involving 124 true TSSs and 78,000 other sites (OSs), the perceptron was unable to converge. However, all 124 true TSSs have perceptron score (PS) greater than 0, whereas only 64 out of 78,000 OSs have PS greater than 0. A well-implemented perceptron will have an option to exclude these 64 offending sequences so that a converged solution can be found.

To summarize, the input to a perceptron consists of two groups of sequences of the same length, and our objective is to find a scoring function to maximize the difference between the two groups. The scoring function is in the form of a weighting matrix, derived from training the perceptron with the two groups of sequences. The weighting matrix can be used to compute a score for a sequence according to Eq. (3.19). The output from training a perceptron is a weighting matrix and a score for each input sequence. A perceptron that has achieved convergence will have positive sequence scores in the POS group and negative sequence scores in the NEG group. An unknown sequence is classified into the POS group when its score is larger than 0 and into the NEG group when its score is smaller than 0. Perceptron algorithms are implemented in my program DAMBE (Xia 2013, 2017d).

## Postscript

Many bioinformatic algorithms are based on inductive reasoning. These include position weight matrix, perceptron, Gibbs sampler, hidden Markov model, self-organizing map, and other artificial neural network algorithms. They all need training data, and any insight derived from applying these algorithms typically does not go beyond the training data. For this reason, I am offering an example to illustrate the risk of inductive reasoning coupled with extrapolation.

Jane Doe gave 1 cent to John Dean and received 1 dollar back. She gave him 10 cents and got 10 dollars back. She gave him 1 dollar and got 100 dollars back. So she came up with a hypothesis that  $X$  dollars that she gave to John Dean will be rewarded by  $100*X$  dollars. She tested the hypothesis by giving John Dean \$10 and got \$1000, which is perfectly consistent with her hypothesis. Driven by the desire to get rich quick, she borrowed \$100,000 from her relatives and friends, gave it all to John Dean, and waited and waited for the fantastic reward. But John Dean vanished and was never heard of again.

This example has compelled me to state explicitly here that all chapters in this book are offered without any warranty. You read, learn, and practice at your own risk. . . . .

# Chapter 4

## Gibbs sampler



### 1 Introduction

#### 1.1 Gibbs Sampler and Its Biological Applications

A Gibbs sampler (Geman and Geman 1984), named after the mathematical physicist, J. W. Gibbs, is one of the Monte Carlo algorithms that rely on repeated random sampling to estimate desired parameters. Monte Carlo method was envisioned by the famous mathematician Stanislaw Ulam, following the successful assembly of the first electronic computer ENIAC in 1945 and further developed by physicists and mathematicians working on nuclear weapon projects in the Los Alamos National Laboratory in mid-1940s (Metropolis 1987). The term “Monte Carlo method” was coined by Nicholas Metropolis to designate this class of computational algorithms. While the general application of the method unsurprisingly followed the operation of ENIAC in 1945, the physicist Enrico Fermi is known to have independently developed and applied the method nearly 15 years earlier with mechanical calculators (Metropolis 1987).

Gibbs sampling simplifies computation in parameter estimation when analytical solution is very difficult or impossible to obtain. In biology, it has been used in the identification of functional motifs in proteins (Mannella et al. 1996; Neuwald et al. 1995; Qu et al. 1998), biological image processing (Samso et al. 2002), pairwise sequence alignment (Zhu et al. 1998), and multiple sequence alignment (Holmes and Bruno 2001; Jensen and Hein 2005). However, the most frequent biological application of Gibbs sampler remains in the identification of regulatory sequences of genes (Aerts et al. 2005; Coessens et al. 2003; Lawrence et al. 1993; Qin et al. 2003; Thijs et al. 2001, 2002a, b; Thompson et al. 2003, 2004).

Gibbs sampler is implemented in DAMBE (Xia 2013, 2017d) which comes with a sample FASTA file including yeast intron sequences. An intron to be spliced by spliceosome needs to have a branchpoint sites (BPSs) somewhere in the intron, in addition to the 5' and 3' splice sites. One can use Gibbs sampler to find where BPSs are located in the intron and what they look like.

## **1.2 What Is De Novo Motif Discovery?**

While a position weight matrix covered in the previous chapter is a technique for characterizing a set of identified motifs, a Gibbs sampler can be used for de novo motif discovery. For example, given a set of yeast intron sequences, what and where is the branchpoint site? All information we have is that each intron should have one branchpoint site, but what sequence signature does it have and where is it located along the intron sequence? This scenario (Fig. 4.1) is where the Gibbs sampler will shine. A functionally equivalent bioinformatics tool is MEME (Bailey et al. 2006).

A similar scenario involves the discovery of regulatory sequence motifs given a set of co-expressed genes (i.e., genes that increase or decrease their transcription level synchronously over time) by microarray (Schena 1996, 2003), SAGE (Saha et al. 2002; Velculescu et al. 1995), or deep-sequencing(Maher et al. 2009; Prensner et al. 2011; Wang et al. 2009) experiments. If the co-expressed genes are also co-regulated, then they may share a certain yet-unknown transcription factor binding site controlled by the same or similar transcription factor. Given that the binding site is

**Fig. 4.1** De novo motif discovery with Gibbs sampler. The intron sequences in the top panel represent the input information to the Gibbs sampler. The bottom panel represents part of the output showing the identified motif (i.e., TAATAAC, in red) shared among the sequences. Output from DAMBE (Xia 2013). The input intron sequence file (YeastAllIntron.fas) is in DAMBE installation directory in FASTA format.

often located upstream of the translation initiation codon, one may extract the upstream sequences from these co-expressed genes and let the Gibbs sampler to find the candidate regulatory motifs. A recent study has shown that shared motifs may also present in the 5' UTR of mRNA to modulate translation initiation (Xia et al. 2011).

### 1.3 *What Are the Estimated Parameters When Applying Gibbs Sampler in Motif Discovery?*

We want to find a shared motif buried in the input sequences (e.g., the 7mer highlighted in red in Fig. 4.1) that will generate a PWM with a strong pattern. The motifs are equivalent to parameters in a statistical model, e.g., the slope and intercept in linear regression. Just like the slope and intercept that can take different values, a motif can take the form of many different 7mers. So we need a criterion to decide which parameter or motif is better than others. In linear regression, we typically have the least-squares criterion; the slope and intercept are the best if the summation of squared deviations of data points to the regression line is minimized. In maximum likelihood estimation, the best parameter values are those that maximize the likelihood. In Gibbs sampling for motif discovery, the best motifs are those that will maximize the Kullback-Leibler information (Kullback 1959, 1987; Kullback and Leibler 1951), often designated by the letter F. We start by picking a random 7mer from each sequence and progress through an uphill search by replacing the 7mer by better ones from each sequence (“better ones” are those that generate larger F values). We continue the process until F value no longer increases. The process is equivalent to finding the highest building on campus by dropping your students randomly on campus, asking them to immediately climb up the building next to them, and reporting their height upon reaching the top. After recording the highest building reached, you again drop your students randomly on campus and ask them to do the same. If we always have the same building recorded as the highest, we will stop and report that building as the highest building on campus.

There are two slightly different applications of Gibbs sampler in motif prediction. The first assumes that each sequence contains exactly one motif (Lawrence et al. 1993) and the associated algorithm is called a site sampler. The second is more flexible and allows each sequence to have none or multiple motifs (Neuwald et al. 1995), and the algorithm is termed a motif sampler. We will illustrate the site sampler and then briefly discuss the motif sampler. Much of this chapter is based on a previous tutorial on Gibbs sampler (Rouchka 1997).

Gibbs sampler for de novo motif discovery is implemented in DAMBE (Xia 2013). The main output of the Gibbs sampler is typically of three parts. The first is the shared motif in an aligned format (bottom panel in Fig. 4.1). The second is a PWM summarizing the discovered motif, and the third contains the associated significance tests which will be reviewed in a later section. The derived PWM, just like any other PWM, can be used to scan sequences not in the input data to discover the presence of the motif present elsewhere.

## 2 Computational Details of Gibbs Sampler

We will use the erythroid nucleotide sequences (Rouchka 1997), listed in Fig. 4.2, to illustrate the Gibbs sampler algorithm. Our main objective is to infer the location and sequence of the unknown motif shared among the sequences so that we can align the motifs as shown in the bottom panel of Fig. 4.1. The aligned motifs will allow us to generate a PWM that characterizes the motif by site-specific nucleotide frequency distributions. The PWM can be used to scan for the presence of the identified motif in other sequences.

We need first to count all nucleotides, with their numbers designated as  $F_A$ ,  $F_C$ ,  $F_G$ , and  $F_T$ , respectively, in the sequences. The total number of nucleotides of all 29 sequences (Fig. 4.2) is 1209, with  $F_A$ ,  $F_C$ ,  $F_G$ , and  $F_T$  equal to 325, 316, 267, and 301, respectively. These values are needed for specifying pseudocounts (which we encountered in the previous section on PWM).

```

S1  TCAGAACCAAGTTATAAATTATCATTTCCTCTCCACTCCT
S2  CCCACGCAGCCGCCCTCCTCCCCGGTCACTGACTGGTCCTG
S3  TCGACCCTCTGAACCTATCAGGGACCACAGTCAGCCAGGCAAG
S4  AAAACACTTGAGGGAGCAGATAACTGGGCCAACCATGACTC
S5  GGGTGAATGGTACTGCTGATTACAACCTCTGGTGCTGC
S6  AGCCTAGAGTGTGACTCCTATCTGGGTCCCCAGCAGGA
S7  GCCTCAGGATCCAGCACACATTATCACAACCTTAGTGTCCA
S8  CATTATCACAAACTTAGTGTCCATCCATCACTGCTGACCCT
S9  TCGGAACAAGGCAAAGGCTATAAAAAAAATTAAGCAGC
S10 GCCCCTCCCCACACTATCTCAATGCAAATATCTGTCTGAAACGGTTCC
S11 CATGCCCTCAAGTGTGCAGATGGTCACAGCATTCAAGG
S12 GATTGGTCACAGCATTTCAAGGGAGAGACCTCATTGTAAG
S13 TCCCCAACTCCCAACTGACCTTATCTGTGGGGGAGGCTTTGA
S14 CCTTATCTGTGGGGGAGGCTTTGAAAAGTAATTAGGTTAGC
S15 ATTATTTCCCTATCAGAACAGAGAGACAAGCCATTCTCTTCCCTCCC
S16 AGGCTATAAAAAAAATTAAGCAGCAGTATCCTCTGGGGGCCCTTC
S17 CCAGCACACACACTTATCCAGTGGTAAATACACATCAT
S18 TCAAATAGGTACGGATAAGTAGATATTGAAGTAAGGAT
S19 ACTTGGGGTTCAGTTGATAAGAAAAGACTTCTGTGGA
S20 TGGCCGCAGGAAGGGTGGCCTGGAAGATAACAGCTAGTAGGCTAAGGCCA
S21 CAACCACAACCTCTGTATCCGGTAGTGGCAGATGGAAA
S22 CTGTATCCGGTAGTGGCAGATGGAAGAGAAACGGTAGAA
S23 GAAAAAAAATAATGAAGTCTGCCTATCTCCGGGCCAGAGCCCCCT
S24 TGCCTTGTCTGTTGAGATAATGAATCTATCCTCCAGTGACT
S25 GGCCAGGCTGATGGGCCTTATCTCTTACCCACCTGGCTGT
S26 CAACAGCAGGTCTACTATCGCCTCCCTCTAGTCTCTG
S27 CCAACCGTTAATGCTAGAGTTATCACTTCTGTTATCAAGTGGCTTCAGC
S28 GGGAGGGTGGGGCCCTATCTCTCTAGACTCTGTG
S29 CTTTGTCACTGGATCTGATAAGAACACCACCCCTGC

```

**Fig. 4.2** The erythroid sequences (Rouchka 1997) for illustrating the Gibbs sampler algorithm, with the 3'-end trimmed to the maximum length 50 bases to fit the page

Let  $N$  be the number of input sequences designated as  $S_1, S_2, \dots, S_i, \dots, S_N$ . Let  $L_i$  be the length of  $S_i$  and  $m$  be the length of the motif, which typically is of length 4–8. For our illustration, we will use  $m = 6$ . One typically would run the Gibbs sampler several times with different  $m$  values if one knows little about the length of the motif. The PWM is of dimension  $4 \times m$  for nucleotide sequences and  $20 \times m$  for amino acid sequences. Let  $A_i$  be the unknown starting position of the motif in  $S_i$ .

The main algorithm of Gibbs sampler is of two steps. The first is random initialization in which a random set of  $A_i$  values is assigned and site-specific nucleotide frequencies are calculated. The second step is predictive updating until a local solution of  $A_i$  values is obtained, together with site-specific nucleotide frequencies that can be made into a PWM. This is repeated multiple times, and previously stored locally optimal solutions are replaced by better ones. Convergence is typically declared when two or more local solutions are identical. These steps are numerically illustrated in the following sections.

## 2.1 Initialization

The initiation step randomly assigns a value to  $A_i$ , with the constraint that  $1 \leq A_i \leq L_i - m + 1$ . So our first set of  $N$  “motifs” is essentially a random set of sequences of length  $m$  and is not expected to have any pattern. For readers who are curious, the first set of 29 random  $A_i$  values happen to be 29, 31, 23, 28, 10, 2, 18, 32, 20, 15, 11, 25, 24, 30, 18, 15, 10, 23, 14, 15, 26, 36, 8, 6, 30, 19, 27, 26, and 14. The site-specific distribution of nucleotides from the 29 random motifs is shown in Table 4.1. There is hardly any site-specific pattern, as one would have expected.

The second column in Table 4.1 will be referred to as the  $C0$  vector with  $C0_A$ ,  $C0_C$ ,  $C0_G$ , and  $C0_T$  equal to 278, 279, 230, and 248, respectively. The  $4 \times 6$  matrix, occupying the last six columns in Table 4.1, will be referred to as the  $C$  matrix. The  $C$  matrix is tabulated from the 29 random motifs whereas the  $C0$  vector is tabulated from nucleotides outside of the motifs. Thus, the sum of the first, second, third, and fourth rows of Table 4.1 should be equal to  $F_A$ ,  $F_C$ ,  $F_G$ , and  $F_T$ , respectively. Also note that each of the six columns in the  $C$  matrix should add up to 29.

**Table 4.1** Site-specific distribution of nucleotides from the 29 random motifs of length 6

Nuc	C0	Site					
		1	2	3	4	5	6
A	278	8	7	9	6	10	7
C	279	3	8	5	10	6	5
G	230	7	5	6	5	3	11
T	248	11	9	9	8	10	6

The second column lists the distribution of nucleotides outside the 29 random motifs

## 2.2 Predictive Update

The predictive update consists of obtaining  $N$  ( $= 29$  in our example) random numbers ranging from 1 to  $N$  and uses these numbers as an index to choose the sequences sequentially to update the site-specific distribution of nucleotides (the  $C$  matrix) and the associated frequencies (the  $C0$  vector). For example, the  $N$  random numbers in my first run of the Gibbs sampler happen to be 11, 18, 26, 22, 2, 28, 12, 9, 7, 3, 17, 16, 1, 4, 21, 15, 14, 24, 19, 27, 29, 6, 10, 20, 13, 8, 23, 25, and 5, respectively. This means that  $S_{11}$  will be used first, and  $S_5$  last, for the first cycle of the predictive update. It is important to use a random series of numbers instead of choosing sequences according to the input order. The latter increases the likelihood of trapping a Gibbs sampler within a local optimum.

Our first randomly chosen sequence happens to be  $S_{11}$ , and its randomly chosen motif starts at site 11, i.e.,  $A_{11} = 11$ , with the motif being AGTGTG. This initial motif will now be taken out of the  $C$  matrix and put into the  $C0$  vector. This motif has one  $A$ , zero  $C$ , three  $G$ s, and two  $U$ s. By adding these values to the  $C0$  vector in Table 4.1, we obtain the  $C0$  vector in Table 4.2. We also need to take this motif out of the  $C$  matrix by subtracting the first  $A$  from the first value in the first column in the  $C$  matrix in Table 4.1 (i.e., new  $C_{A,1} = \text{old } C_{A,1} - 1$ ), the second  $G$  from the third value in the second column in the  $C$  matrix in Table 4.1 (i.e., new  $C_{G,2} = \text{old } C_{G,2} - 1$ ), and so on. This converts the  $C$  matrix in Table 4.1 to the  $C$  matrix in Table 4.2.

At this point the  $C$  matrix is made of the 28 randomly chosen motifs, one from each sequence (excluding  $S_{11}$ ). You will notice that each of the six columns in the  $C$  matrix has a sum of 28. The reason for taking the initial motif in  $S_{11}$  out of the  $C$  matrix and putting it back into the  $C0$  vector is that we are going to find a better motif in  $S_{11}$  and put it into the  $C$  matrix so that the  $C$  matrix will again be based on 29 motifs. How are we going to get a better motif? Recall that a position weight matrix (PWM) can be used to scan a sequence in a sliding window of length  $m$  to get position weight matrix scores (PWMSs) for each window. Recall that PWM depends on site-specific frequencies and background frequencies. We will make a PWM out of the  $C0$  vector (from which we derive background frequencies) and the  $C$  matrix (from which we derive site-specific frequencies). We can then use the resulting PWM to scan  $S_{11}$  and get a new motif that has the highest PWMS. Because the input sequences are typically much longer than the hidden motif, the frequencies from  $C0$

**Table 4.2** Site-specific distribution of nucleotides from the 28 random motifs of length 6, after removing the initial motif in  $S_{11}$

Nuc	C0	Site					
		1	2	3	4	5	6
A	279	7	7	9	6	10	7
C	279	3	8	5	10	6	5
G	233	7	4	6	4	3	10
T	250	11	9	8	8	9	6

The second column lists the distribution of nucleotides outside the 28 random motifs

will not be much affected by the motif even if the motif itself has biased nucleotide or amino acid usage. However, if the input sequences are short, then it is better to input background frequencies instead of using the  $C$  to obtain background frequencies.

One may wonder why such a practice would get us anywhere given the fact that the  $C$  matrix is initially made of random motifs. The resulting PWM would exhibit no pattern, and the resulting PWMSs will therefore be uninformative. The key concept here is that when one takes a random walk over a terrain with multiple peaks, one sooner or later will encounter a peak, and climbing the peak will at least bring us to a local maximum. After reaching the top of one peak and recording the height, we will land ourselves at another randomly chosen location and start climbing local peaks again. This process continues until we reach the highest peak or after a fixed number of computer iterations without finding any higher peak.

For pseudocounts, we may use  $\alpha = 0.0001$ . The resulting PWM is then used to scan  $S_{11}$  which is 40 bases long, with  $35 (= 40 - m + 1)$  possible motif starting points (i.e., possible  $A_i$  values along the sequence). The 35 PWMS values for these 35 possible motifs in  $S_{11}$  (Table 4.3) are normalized to have a sum of 1 ( $P_{\text{Norm}}$  in Table 4.3). We now proceed to update the initial  $A_{11}$  ( $= 11$ ) by a new  $A_{11}$  value based on result in Table 4.3. How should we choose the new  $A_{11}$  value?

There are two strategies to choose the new  $A_{11}$  value. The first is to randomly pick up an  $A_i$  value according to the magnitude of  $P_{\text{Norm}}$  (Table 4.3). You may visualize a dartboard with 35 slices with their respective areas being proportional to  $P_{\text{Norm}}$  values.

**Table 4.3** Possible locations of the 6-mer motif along  $S_{11}$ , together with the corresponding motifs and their position weight matrix scores expressed as odds ratios

Site	6-mer	Odds ratio	$P_{\text{Norm}}$	Site	6-mer	Odds ratio	$P_{\text{Norm}}$
1	CATGCC	0.153	0.004	19	GATTGG	1.128	0.028
2	ATGCC	0.850	0.021	20	ATTGGT	0.408	0.010
3	TGCCCT	0.664	0.016	21	TTGGTC	1.194	0.030
4	GCCCTC	0.944	0.023	22	TGGTCA	0.888	0.022
5	CCCTCA	0.254	0.006	23	GGTCAC	1.005	0.025
6	CCTCAA	0.843	0.021	24	GTCACA	0.596	0.015
7	CTCAAG	0.609	0.015	25	TCACAG	5.888	0.146
8	TCAAGT	0.717	0.018	26	CACAGC	0.064	0.002
9	CAAGTG	0.613	0.015	27	ACAGCA	0.569	0.014
10	AAGTGT	0.426	0.011	28	CAGCAT	0.569	0.014
11	AGTGTG	0.967	0.024	29	AGCATT	0.381	0.009
12	GTGTGC	0.546	0.014	30	GCATT	2.024	0.050
13	TGTGCA	0.594	0.015	31	CATTTC	0.474	0.012
14	GTGCAG	4.034	0.100	32	ATTTCA	1.317	0.033
15	TGCAGA	0.251	0.006	33	TTTCAA	4.293	0.107
16	GCAGAT	1.084	0.027	34	TTCAAG	2.475	0.061
17	CAGATT	0.343	0.009	35	TCAAGG	1.279	0.032
18	AGATTG	1.812	0.045				

The last column lists the odds ratios normalized to have a sum of 1

When you throw a dart at the dartboard, large slices will have a better chance of being hit than small slices. If the dart happens to land on the seventh slice, then the initial  $A_{11} = 11$  will be updated to  $A_{11} = 7$ , with the original motif AGTGTG replaced by the new motif CTCAAG.

The second strategy is simply to use the largest  $P_{\text{Norm}}$  value for updating initial  $A_{11}$  to the new  $A_{11}$  value. As the motif starting at site 25 has the largest  $P_{\text{Norm}}$ , we will set the new  $A_{11}$  equal to 25 and replace the initial motif (=AGTGTG) by the new motif (=TCACAG). With this approach we do not need  $P_{\text{norm}}$  as we can choose  $A_{11}$  based on the largest odds ratio in Table 4.3. This strategy is faster than the first, but did not seem to lose any sensitivity in motif discovery based on limited simulation studies. However, if one is concerned about the possibility of missing motifs, one should use the first strategy.

Regardless of how the new  $A_{11}$  is chosen, the updating is the same. Suppose we have taken the second strategy and set the new  $A_{11}$  equal to 25. The  $C$  matrix in Table 4.1 is then revised by replacing the original  $A_{11}$  motif (=AGTGTG) by the new motif (=TCACAG). This leads to an updated  $C_0$  vector and  $C$  matrix (Table 4.4).

We repeat this process for the rest of the sequences to update the rest of  $A_i$  values. After the last sequence has been updated, we have obtained a new set of  $A_i$  values, a new set of 29 motifs, together with the PWM based on the associated  $C_0$  vector and  $C$  matrix. At this point we compute a weighted alignment score (i.e., a weighted PWMS) as follows:

$$F = \sum_{i=1}^{N_{\text{Code}}} \sum_{j=1}^m C_{i,j} \text{PWM}_{ij} = \sum_{i=1}^{N_{\text{Code}}} \sum_{j=1}^m C_{ij} \log_2 \frac{p_{ij}}{p_i} \quad (4.1)$$

where  $m$  is the motif width, and  $N_{\text{Code}}$  is the number of different symbols in the sequences (4 for nucleotide and 20 for amino acid sequences).  $F$  measures the strength of the pattern. The larger the  $F$  value, the stronger the pattern. Table 4.5 illustrates the calculation of  $F$  for two contrasting site-specific nucleotide distributions each with five nucleotide sites, using equal background frequencies, i.e.,  $p_i = 0.25$ . Distribution 1 does not exhibit a strong site-specific pattern, and  $F$  is small (= 6.318). In contrast, Distribution 2 represents a very strong site-specific pattern, and  $F$  is large (= 215.545).

The  $F$  value, as defined in Eq. (4.1), has many different names. It has been called the Kullback-Leibler information or Kullback-Leibler divergence in information

**Table 4.4** Site-specific distribution of nucleotides from the 29 initial motifs of length 6, after replacing the initial  $A_{11}$  motif (=AGTGTG) by the new motif (=TCACAG)

Nuc	C0	Site					
		1	2	3	4	5	6
A	277	7	7	10	6	11	7
C	277	3	9	5	11	6	5
G	232	7	4	6	4	3	11
T	249	12	9	8	8	9	6

**Table 4.5** Calculation of  $F$  for two contrasting site-specific nucleotide distributions each with five nucleotide sites labeled 1 to 5

	Distribution 1					Distribution 2				
$C_{ij}$	1	2	3	4	5	1	2	3	4	5
A	12	9	7	13	10	35	1	1	18	1
C	10	10	6	10	8	1	1	1	2	19
G	8	11	14	7	12	2	1	2	18	1
T	10	10	13	10	10	2	37	36	2	19
$p_{ij}$	$pi1$	$pi2$	$pi3$	$pi4$	$pi5$	$pi1$	$pi2$	$pi3$	$pi4$	$pi5$
A	0.300	0.225	0.175	0.325	0.250	0.875	0.025	0.025	0.450	0.025
C	0.250	0.250	0.150	0.250	0.200	0.025	0.025	0.025	0.050	0.475
G	0.200	0.275	0.350	0.175	0.300	0.050	0.025	0.050	0.450	0.025
T	0.250	0.250	0.325	0.250	0.250	0.050	0.925	0.900	0.050	0.475
				$F =$	6.318				$F =$	215.545

Equal background frequencies are used, i.e.,  $p_i = 0.25$

theory (Kullback 1959, 1987; Kullback and Leibler 1951), or large-deviation rate function in statistical estimation (Bucklew 1990). In bioinformatics, especially in motif characterization and prediction involving a PWM, it is most often referred to as the information content (Hertz and Stormo 1999). The fact that the Kullback-Leibler information is a special case of the so-called  $f$ -divergence that measures the difference between two probability distributions  $P$  and  $Q$  leads naturally to the use of the letter  $F$  in Eq. (4.1).

The predictive updating is repeated again and again. Each time when we get a new set of  $A_i$  values, a new set of motifs, and the PWM based on the  $C0$  vector and the  $C$  matrix, we compute a new  $F$  value. If the new  $F$  value is greater than the previously stored  $F$  value, then the new  $F$  value, the new set  $A_i$  values, and the new set of motifs will replace the previously stored ones. This continues until we reach a local maximum of  $F$  or when the preset maximum number of local loops has been reached. The resulting  $F$  value, the set of  $A_i$  values, the new set of motifs, and the associated PWM are stored as the locally optimal output. In the hill-climbing analogy,  $F$  represents the height of a local peak.

The entire process is now repeated from the very beginning, i.e., we again perform the initialization by choosing another random set of  $A_i$  values, and go through the local iteration to obtain another locally optimal output. If the new locally optimal output is better than previously stored ones (i.e., the new  $F$  value is larger than the previously stored one), the new output will replace the previously stored output. This process is repeated multiple times until convergence is reached, i.e., when new  $F$  values are consistently the same as the previously stored one, or until a fixed number of computation iteration has been reached without finding an  $F$  value better than what has already been recorded. The final site-specific nucleotide distribution (Table 4.6) displays a much stronger pattern than the initial distribution (Table 4.1) from 29 randomly chosen motifs.

**Table 4.6** Final site-specific distribution of nucleotides from the 29 identified motifs. Output from DAMBE

Nuc	C0	Site					
		1	2	3	4	5	6
A	275	3	0	22	0	9	16
C	285	11	0	0	0	19	1
G	252	0	7	7	0	0	1
T	223	15	22	0	29	1	11

The final aligned motifs (Fig. 4.3) share in general a consensus of (C/T)TATC (A/T). Its reverse complement (A/T)GATA(A/G) is known to be the binding site of GATA-binding transcription factors (Aird et al. 1994; Fong and Emerson 1992; Moi et al. 1992; Nishimura et al. 2000; Orkin 1992; Zon et al. 1991). This discovery of the motif suggests that this set of sequences may indeed be co-regulated by the same type of GATA-binding transcription factors. Such findings are crucial in transcriptomic and proteomic studies aiming to understand gene regulation networks. Algorithms such as Gibbs sampler help us understand interactions among genes and gene products.

It might be relevant here to summarize essential biology about the GATA box and GATA-binding transcription factors. A living cell is a system with many genetic switches that can be turned on or off in response to intracellular and extracellular environment. It is these switches that distinguish a normal living cell from a cancer cell or a dead cell. The GATA motif (or GATA box) is one of such switches, and it is switched on or off by specific transcription factors (which are proteins that bind to the motif and turn on or off the transcription of the gene containing such motifs). One of the better known GATA-binding transcription factors is GATA-1 which binds to the GATA motif found in cis-elements of the vast majority of erythroid-expressed genes of all vertebrate species examined (Evans et al. 1990; Orkin 1990). The core promoter of the rat platelet factor 4 (PF4) gene contains such a GATA motif and the binding of such GATA motif by GATA-binding proteins such as GATA-1 suppresses the transcription of the PF4 gene (Aird et al. 1994). It is now known that GATA regulatory motifs and the GATA-binding transcription factors are present in a variety of organisms ranging from cellular slime mold to vertebrates, including plants, fungi, nematodes, insects, and echinoderms (Lowry and Atchley 2000), suggesting that the function of the genetic switch is far beyond erythropoiesis. In human, the GATA motif and the GATA-binding proteins are implicated in several diseases (Van Esch and Devriendt 2001). The sequence divergence of GATA motifs and their binding proteins should shed light on the coevolution of the components of genetic switches.

One may have noted that some sequences have a strong (C/T)TATC(A/T) motif, whereas others (e.g., the second, the fourth, and the fifth sequences) have only weak and highly doubtful signals. Computer programs implementing Gibbs sampler typically would output a quantitative measure of the strength of the signal, and PWMS is the most often used index for this purpose (Table 4.7). Recall that PWMS

```

1      TCAGAAACCAAGTTATAAATTATCATTTCCCTCTCCACTCCCT
2      CCCACGCAGCGCCCTCCTCCCCGTCACTGACTGGTCTG
3          TCGACCCCTGTGAACTATCAAGGGACCACAGTCAGCCAGGCAAG
4          AAAACACTTGAGGGAGCAGATAACTGGCCAACCACAGTCAG
5          GGGTGAATGGTACTGCTGATTACAACCTCTGGTCTGC
6          AGGCTAGATGATGACTCTTATCTGGGTCCCCAGCAGGA
7          GCCTCAGGATCCACACACATTATCACAAACCTAGTGTCCA
8          CATTATCACAAACCTAGTGTCCAATCCACTGCTGACCC
9          CGGAAACAAAGGCAAAGGTATAAAAAAAATTAAGCAGC
10         GCCCCTCCCCACACTATCTCAATGCAAATATCTGCTGAAACGGTCC
11         CATGCCCTCAAGTGTGAGATTGGTCACAGCATTCAAGG
12 GATTGGTCACAGCATTTCAAGGGAGAGACCTCATTGTAG
13         TCCCCAACTCCCAACTGACCTTATCTGTGGGGGAGGCTTTGA
14         CTTATCTGTGGGGGAGGCTTTGAAAAGTAATTAGTTTAGC
15         ATTATTTCCTATCAGAAGCAGAGAGACAAGGCCATTCTCTCC
16         AGGTATAAAAAAAATTAAGCAGCAGTATCCTTGGGGCCCTTC
17         CCAGCACACACACTTATCCAGTGGTAAATACACATCAT
18         TCAAATAGGTACGGATAAGTAGATATGAAGTAAGGAT
19         ACTGGGGTCCAGGTGATAAGAAAGACTTCTGTGGA
20         TGGCCGCAGGAAGGTGGGCTGGAAGATAACAGCTAGTAGGCTAAGGCCA
21         CAACACAAACCTCTGTATCCGGTAGTGGCAGATGGAAA
22         CTGTATCCGGTAGTGGCAGATGGAAAAGAGAACGGTTAGAA
23         GAAAAAAAATAATGAAGTGTGCTATCTCCGGGCCAGAGCCCC
24         TGCCTTGTCTGTGTTAGATAATGAATCTACCTCCAGTGACT
25         GGCCAGGCTGATGGGCCTTATCTCTTACCCACTGGCTGT
26         CAACAGCAGGTCTATATCGCCCTCTAGTCTCTG
27         CCAACCGTTAATGCTAGAGTATCACTTCTGTTATCAAGTGGCTTCAGC
28         GGGAGGGTGGGGCCCTATCTCTCTAGACTCTGTG
29         CTTGTCACTGGATCTGATAAGAAACACCACCCCTGC

```

**Fig. 4.3** Aligned motifs generated from Gibbs sampler. Output from DAMBE (Xia 2013)

is the log-odds, but one may use the odds ratio directly as a measure of relative motif strength. Also recall that an odds ratio is the ratio of two probabilities associated with two hypotheses. Define  $\theta_{\text{Yes}}$  as the hypothesis that the 6-mer is a motif with its site-specific constraints and  $\theta_{\text{No}}$  as the hypothesis that the 6-mer is not a motif and has its probabilities specified only by the four overall nucleotide frequencies. The odds ratio is the ratio of the probability that  $\theta_{\text{Yes}}$  is true over the probability that  $\theta_{\text{No}}$  is true. One generally should take a cutoff value of 20, i.e.,  $\theta_{\text{Yes}}$  is 20 times more likely than  $\theta_{\text{No}}$ .

One should note that a Gibbs sampler, being started from a random set of  $A_i$  values, may not necessarily converge to the same motif. This is both an advantage and a disadvantage of the algorithm. The advantage is that repeated running of the algorithm will allow us to identify other types of hidden motifs (i.e., other than the reverse complement of the GATA motif) in the sequences. The disadvantage is that users not familiar with the algorithm often get confused when the same input generates quite different results. For example, another set of putative motifs, in the form of RGVAGR (where R is A or G and V is “not T”), has been found to be shared among the sequences (Xia 2007b, p. 146).

It is possible that the input sequences may contain two or more different biologically significant motifs. If one motif is much stronger (more overrepresented among the input sequences) than other motifs, and if the search by the Gibbs sampler algorithm outlined before is exhaustive, then we will always end up with the strongest motif and miss all other biologically interesting motifs. However, one

**Table 4.7** Output of PWMS as a quantitative measure of the strength of the identified motifs. Output from DAMBE

SeqName	Motif	Start	Odds ratio	SeqName	Motif	Start	Odds ratio
Seq1	TTATCA	18	163.6602	Seq16	CTATAA	3	58.1420
Seq2	CGGTCA	22	14.5511	Seq17	TTATCC	13	34.3258
Seq3	CTATCA	14	101.8203	Seq18	AGATAT	20	8.1245
Seq4	AGATAA	17	9.1127	Seq19	TGATAA	16	32.0835
Seq5	TGATTA	16	12.9266	Seq20	AGATAA	24	9.1127
Seq6	CTATCT	18	90.7790	Seq21	CTGTAT	12	21.5783
Seq7	TTATCA	20	163.6602	Seq22	CTGTAT	0	21.5783
Seq8	TTATCA	2	163.6602	Seq23	CTATCT	23	90.7790
Seq9	CTATAA	17	58.1420	Seq24	TTGTCT	4	60.7395
Seq10	CTATCT	14	90.7790	Seq25	TTATCT	17	145.9129
Seq11	TGGTCA	21	23.3886	Seq26	CTATCG	15	21.2368
Seq12	TTGTAA	33	38.9024	Seq27	TTATCA	19	163.6602
Seq13	TTATCT	20	145.9129	Seq28	CTATCT	15	90.7790
Seq14	TTATCT	2	145.9129	Seq29	TTGTCA	2	68.1272
Seq15	TTATCA	10	163.6602				

Odds ratio mean and standard deviation are 76.3120 and 57.8163, respectively

**Fig. 4.4** An alternative motif that may be returned from running the Gibbs sampler on the same set of input sequences. The left column lists the sequence numbers

4	AAAACACTTGAGGG <b>AGCAGATA</b> ...
12	GATTGGTCACAGCACATTCAA <b>GGGAGAGA</b> ...
13	TCCCCAACCTCCAACTGACCTTATCTGTGG <b>GGGAGGCT</b> ...
14	CCTTATCTGTGG <b>GGGAGGCT</b> ...
15	ATTATTTTCCTTATCAGA <b>AGCAGAGA</b> ...
20	TGGCCGCAGGAAGGTGGGCCT <b>GGAAGATA</b> ...
21	CAACCACAACCTCTGTATCCGTAGT <b>GGCAGATG</b> ...
22	CTGTATCCGGTAGT <b>GGCAGATG</b> ...
26	CAAC <b>AGCAGGTC</b> ...
28	<b>GGGAGGGT</b> ...

could run a Gibbs sampler by specifically exclude the strongest motif already identified so that weaker motifs can then be identified. For example, another set of putative motifs is partially shown in Fig. 4.4.

## 2.3 Motif Sampler

The Gibbs sampler has two versions. The one that we have just illustrated is called site sampler. It assumes that each sequence contains exactly one motif (Lawrence et al. 1993). The other version is more flexible and allows each sequence to have none or multiple motifs (Neuwald et al. 1995), and the algorithm is termed motif sampler. The GATA-binding transcription factors comprise a protein family whose members contain either one or two highly conserved zinc finger DNA-binding

domains (Lowry and Atchley 2000), and it is consequently likely that a sequence may contain more than one GATA box. For example, the erythroid Kruppel-like factor (EKLF, which is a zinc finger transcription factor required for  $\beta$ -globin gene expression) has in its 5'-region two GATA motifs flanking an E box motif characterized by CANNTG (Anderson et al. 1998). This calls for an algorithm that can identify multiple motifs in a single sequence.

The site sampler can be extended to motif sampler by post-processing. The PWM generated from the site sampler can be used to re-scan the sequences for motifs and compute the associated PWMS or odds ratio for all 6-mers in each sequence. All what we need is to have a cutoff score to keep those motifs with a PWMS or odds ratio greater than the cutoff score.

## Postscript

Gibbs sampler for motif discovery illustrates the magic of a random process guided by a selection process. We start with a random set of motifs and apply a selection process that favors certain motifs against others, based on the criterion that the chosen motifs should contribute to a site-specific frequency distribution with a larger F defined in Eq. (4.1). The starting set of random motifs, shaped by the selection process, eventually converges to a final set of motifs with a strong nonrandom pattern. Sometimes we may have sequences with two or more different types of signal motifs. If we run Gibbs sampler with different starting sets of random motifs, we may converge to different sets of highly informative motifs. Thus, the same random process coupled with the same selection process may generate quite different outcomes.

My Christian friends often assert that Darwinian evolutionary theory is all wrong because random collision of molecules cannot generate highly structured patterns. Random collision of molecules indeed is limited by their potential to generate highly structured patterns. However, Darwinian evolution is not a random process. In fact, Darwin's most significant contribution to biology is the substantiation of a particular force that he named natural selection. The combination of a random process guided by this particular force can do miracles in generating biodiversity of all colors and shades. It is when Darwin visualized the miracles generated by this ubiquitous force that he proclaimed that "There is grandeur in this view of life."

This force has been with us, from time immemorial, and continues to shape all forms of life, including the life of those who deny its existence.

# Chapter 5

## Transcriptomics and RNA-Seq Data Analysis



### 1 Introduction

With the completion of a number of genomic sequences, the next step in understanding the operation of the genetic program encoded in genomic DNA is to know how the subroutines (genes) interact with each other. Take human development, for example. If we designate the time of zygote formation as  $t_0$ , what genes are activated at  $t_1, t_2, \dots, t_n$ ? Does an oocyte always carry the same maternal proteins and RNA species? What genes are activated at  $t_0$  in the zygote in response to the largely maternally determined cellular environment? How do the products of these activated genes activate other genes and lead to the developmental cascade? How could this orderly gene activation go wrong leading to developmental defects? How could genes in a cell fail to respond to proper signals to stop growing and replication leading to cancer? If we know that Gene A activates Gene B which in turn activates Gene C, then we would be at a good position to understand what input Gene B takes and what output it generates. If Genes A, B, and C are all activated by Gene D, then we know that Genes A, B, and C share the same input, although they may produce different output. Identifying Genes A, B, and C as co-regulated genes represents valuable information toward understanding gene regulation networks. The general protocol of identifying co-regulated genes is to first identify co-expressed genes and then try to find if these co-expressed genes share similar cis-regulating motifs (genetic switches) that can be flipped on or off by trans-regulatory proteins or RNAs.

Two experimental transcriptomic approaches have been used to understand gene regulation networks. The first is to characterize differential gene expression between the control and the treatment, or between cancerous liver cells and normal liver cells, or between wild type and mutants, etc. Transcriptomic data have been increasingly used to identify differentially regulated genes, alternatively spliced isoforms, and different transcription start and termination sites between patient and matched control (Arvaniti et al. 2016; Bell et al. 2016; Berger et al. 2010; Furukawa et al. 2016; Haustead et al. 2016; Mlera et al. 2016).

The other experimental transcriptomic approach is to quantify gene expression over time. This includes changes in gene expression of yeast cells over time, of stem cells over the differentiation process, of fruit flies during the starvation process, etc. Identification of co-expressed genes is partially covered in the next chapter on self-organizing map and other clustering algorithms, and the identification of shared regulatory motifs is covered in the chapter on Gibbs sampler.

This chapter will cover more fundamental issues on transcriptomic data analysis, focusing on RNA-Seq data and gene expression characterization. There are three main technologies in transcriptomics: the microarray (Schena 1996, 2003; Schena et al. 1995, 1998), SAGE (Saha et al. 2002; Velculescu et al. 1995, 1997, 1999, 2000), and the RNA-Seq (Deng et al. 2014a; Mortazavi et al. 2008; Trapnell 2015), with the latter quickly replacing the first two.

DNA microarray technology was once the method of choice for exploring gene interactions at the genomic scale (Diehn et al. 2000; Epstein and Butow 2000; Gaasterland and Bekiranov 2000). Centralized databases had been established with standardized data specification to make the rapidly increasing data available to the public (Brazma et al. 2001). Unfortunately, in spite of numerous publications with microarray experiments, very little biological insights have been gained. While some have tried to understand the reason for the failure (e.g., Livesey 2002), most have just chosen to ignore the microarray technology and embrace RNA-Seq. This is unfortunate because one key reason for the microarray failure is shared by SAGE and RNA-Seq, and that is the failure to properly characterize gene expression from paralogous genes. Microarray and SAGE worked well in prokaryotic species or unicellular eukaryotes where few duplicated genes exist. However, in multicellular eukaryotes whose genomes feature many paralogues, it becomes difficult to identify which transcript is from which parologue. This problem is alleviated but not eliminated with RNA-Seq. Many short sequence reads are still mapped equally well to multiple genes, and few bioinformatic tools are available to properly allocate such reads to paralogues, in spite of extensive effort spent in developing bioinformatic methods for RNA-Seq data analysis (Dobin et al. 2013; Langmead et al. 2009a, b, 2010; Langmead and Salzberg 2012; Roberts et al. 2013a; Roberts and Pachter 2013; Trapnell et al. 2009, 2012). We will present one unique algorithm for such allocation later in the chapter.

RNA-Seq (Morin et al. 2008a, b) is now used not only in characterizing differential gene expression but also in many other fields where it replaces the traditional microarray approach. Ribosomal profiling, traditionally done through microarray (Arava et al. 2003; MacKay et al. 2004), is now almost exclusively done with deep sequencing of ribosome-protected segments of messages (Ingolia 2010, 2014, 2016; Ingolia et al. 2009), although the results from the two approaches for ribosomal profiling are largely concordant (Xia et al. 2011). Similarly, EST-based (Rogers et al. 2012) and microarray-based (Pleiss et al. 2007) methods for detecting alternative splicing events and characterizing splicing efficiency are now replaced by RNA-Seq (Kawashima et al. 2014), especially by lariat sequencing (Awan et al. 2013; Stepankiw et al. 2015). Methods included in this chapter are relevant to all these diverse applications with RNA-Seq data.

In spite of the potential of RNA-Seq data in solving many practical biological problems, severe under-usage of RNA-Seq data has been reported (Team 2011). One major stumbling block in using the RNA-Seq data is the large file size. Among the 6472 transcriptomic studies on human available at NCBI/DDBJ/EBI by Apr. 14, 2016, 196 studies each contribute more than 1 terabyte (TB) of nucleotide bases, with the top one contributing 86.4 TB. Few laboratories would be keen on downloading and analyzing this 86.4 TB of nucleotides, not to mention comparing this study to RNA-Seq data from other human transcriptomic studies.

The explosive growth of RNA-Seq data in recent years has caused serious problems in data storage, transmission, and analysis (Kodama et al. 2012; Leinonen et al. 2011). Because of the high cost of maintaining such data, coupled with the fact that few researchers had been using such data, NCBI had planned the closure of the sequence read archive a few years ago (Team 2011) but continued the support only after DDBJ and EBI decided to continue their effort of archiving the data. The incident highlights the prohibitive task of storing, transmitting, and analyzing RNA-Seq data and motivated the joint effort of both industry and academics to search for data compression solutions (Janin et al. 2014; Numanagic et al. 2016; Zhu et al. 2015b).

A recent study, together with the associated software ARSDA (Xia 2017a), offers to solve two most fundamental problems in RNA-Seq data analysis. The first is to shrink file size without losing sequence information. The second is to improve accuracy in quantifying gene expression by employing a new method that properly allocates multi-match reads to paralogous genes. This second problem is particularly serious in characterization eukaryotic gene expression. Most bioinformatic software packages for gene expression either ignore multi-matched reads or simply allocated them equally among match genes. This would result in poor characterization of gene expression for most of the eukaryotic genes. Although numerous papers have been published, most researchers are unaware of this problem. I will use an RNA-Seq data set to contrast gene expression characterization between ARSDA and the commonly used Cufflinks (Trapnell et al. 2010).

## 2 Reduce File Size Without Losing Sequence Information

RNA-Seq data files do not need to be so huge. Take, for example, the characterized transcriptomic data for *Escherichia coli* K12 in the file SRR1536586.sra (where SRR1536586 is the SRA sequence file ID in NCBI/DDBJ/EBI). The file contains 6,503,557 sequence reads of 50 nt each, but 195,310 sequences are all identical (TGTTATCACGGGAGACACACGGCGGGTGCTAACGTCCGTGAAGA-GGG), all mapping exactly to sites 929–978 in *E. coli* 23S rRNA genes. The study (Pobre and Arraiano 2015) did use the MICROBExpress Bacterial mRNA Enrichment Kit to remove the 16S and 23S rRNA. The RNA-removal kit targets the conserved RNA regions at the 5' and 3' ends, but broken RNA fragments without the 5' and 3' ends will remain in the RNA sample.

There are much more extreme cases. For example, 1 of the 12 RNA-Seq files from a transcriptomic study of *Escherichia coli* (SRR922264.sra) contains a read with 1,606,515 identical copies among its 9,690,570 forward reads. There is no sequence information lost if all these 1,606,515 identical reads are stored by a single sequence with a sequence ID such as UniqueSeqX\_1606515 (i.e., SeqID\_CopyNumber format which I dub as FASTA+ format with file type.fasP). Such storage scheme not only results in dramatic saving in data storage and transmission but also leads to dramatic reduction in computation time in downstream data analysis. At present, all software packages for RNA-Seq data analysis will take the 1,606,515 identical reads separately and search them individually against the *E. coli* genome (or target gene sequences such as coding sequences). The SeqID\_CopyNumber storage scheme reduces the 1,606,515 searches to a single one.

The conversion process involves the construction of a large dictionary of unique reads and recording their counts. Most programming languages such as C#, Java, Visual Basic, and Python have a dictionary class for the purpose. The output from processing the SRR1536586.sra file (with part of the read-matching output in Table 5.1) highlights two points. First, many sequences in the file are identical. Second, although the transcriptomic data characterized in SRR1536586 have undergone rRNA depletion by using the RNA-removal kit (Pobre and Arraiano 2015), there are still numerous reads in the transcriptomic data that are from rRNA genes. This suggests that storing mRNA reads separately from rRNA reads can dramatically reduce file size because most researchers are not interested in the abundance of rRNAs.

While the conversion of FASTA/FASTQ files to FASTA+ files is time-consuming and requires 16 GB to 32 GB of RAM to store the dictionary object, it needs to be done only once for data storage, preferably at NCBI/EBI/DDBJ, and the resulting saving in storage space, internet traffic, and computation time in downstream data analysis is tremendous. The file size is 1.49 GB for the original FASTQ file derived from SRR1536586.sra but is only 66 MB for the new FASTA+ file, both being plain text files.

Conventional compression programs compress FASTQ/FASTA files into binary format, and one has to uncompress the file before using it for data analysis.

**Table 5.1** Part of read-matching output from ARSDA, with 195,310 identical reads matching a segment of large subunit (LSU) rRNA, 86,308 identical reads matching another segment of LSU rRNA, and so on. Results generated from ARSDA analysis of the SRR1536586.sra file from NCBI

Gene	Ncopy	Gene	Ncopy
LSU rRNA	195,310	SSU rRNA	30,417
LSU rRNA	86,308	LSU rRNA	29,508
LSU rRNA	58,400	5S rRNA	28,187
SSU rRNA	47,323	LSU rRNA	24,982
LSU rRNA	45,695	SSU rRNA	23,286
LSU rRNA	36,258	LSU rRNA	19,991
5S rRNA	33,674	SSU rRNA	19,268

In contrast, the FASTA+ and FASQ+ files are plain text files and can be used by any other programs direct and in fact dramatically save computation time in downstream data analysis. Take SRR1536586, for example, FASTQ+ would reduce computation time for read matching (which is the most time-consuming part of any transcriptomic data analysis) to a fraction of roughly 0.075 ( $\approx 119,596,093 / 1,604,183,348$ ). The fact that FASTQ+/FASTA+ files are plain text files also implies that the FASTAQ+/FASTA+ files can be further compressed by conventional sequence-compressing software. Again take SRR1536586, for example. LFQC (Nicolae et al. 2015) can compress the FASTQ file from 1,604,183,348 bytes to 101,191,680 bytes (0.0631 of the original). Using ARSDA and LFQC jointly reduces the file size to 16,506,880 bytes, which is on 0.0103 of the original (Xia 2017a).

One of the frequently used sequence-compression scheme is to use a reference genome so that each read can be represented by a starting and an ending number on the genome (Benoit et al. 2015; Kingsford and Patro 2015; Zhu et al. 2015a). This approach has two problems. First, many reads do not map exactly to the genomic sequence because of either somatic mutations or sequencing errors, so representing a read by the starting and ending numbers leads to loss of information. Second, RNA-editing and processing can be so extensive that it becomes impossible to map a transcriptomic read to the genome (Abraham et al. 1988; Alatortsev et al. 2008; Lamond 1988; Li et al. 2009; Simpson et al. 2016). Furthermore, there are still many scientifically interesting species that do not have a good genomic sequence available.

A huge chunk of SRA data stored in NCBI/DDBJ/EBI consists of ribosome profiling data (Ingolia 2010, 2014, 2016; Ingolia et al. 2009), which is obtained by sequencing the mRNA segment (~30 bases) protected by the ribosome after digesting all the unprotected mRNA segments. Mapping these ribosome-protected segments to the genome allows one to know specifically where the ribosomes are located along individual mRNAs. In general, such data are essential to understand translation initiation, elongation, and termination efficiencies. For example, a short poly(A) segment with about eight or nine consecutive A immediately upstream of the start codon in yeast (*Saccharomyces cerevisiae*) genes is significantly associated with ribosome density and occupancy (Xia et al. 2011), confirming the hypothesis that short poly(A) upstream of the start codon facilitates the recruitment of translation initiation factors but long poly(A) would bind to poly(A)-binding protein and interfere with cap-dependent translation. Sequence redundancy is high in such ribosomal profiling data, and the FASTA+ format can lead to dramatic saving in the disk space of data storage and time in data transmission.

The availability of RNA-Seq data has dramatically accelerated the test of biological hypotheses. For example, a recent study on alternative splicing (Vlasschaert et al. 2016) showed that skipping of exon 7 (E<sub>7</sub>) in human and mouse *USP4* is associated with weak signals of splice sites flanking E<sub>7</sub>. The researchers predicted that, in some species where the splice site signals are strong, E<sub>7</sub> skipping would disappear. This prediction is readily tested and confirmed with existing RNA-Seq data, i.e., E<sub>6</sub>–E<sub>8</sub> mRNA was found in species with weak splice signals flanking E<sub>7</sub>, and E<sub>6</sub>–E<sub>7</sub>–E<sub>8</sub> mRNA in species with strong splice signals flanking E<sub>7</sub> (Vlasschaert et al. 2016).

### 3 Assigning Sequence Reads to Paralogous Genes

One of the most fundamental objectives of RNA-Seq analysis is to generate an index of gene expression (FPKM or RPKM: matched fragment/reads per kilobases of transcript per million mapped reads) that can be directly compared among different genes and among different experiments with different total number of matched reads (Mortazavi et al. 2008). The main difficulty in quantifying gene expression arises with sequence reads matching multiple paralogous genes (Rogozin et al. 2014; Trapnell et al. 2013). This problem, which has plagued microarray data analysis, is now plaguing RNA-Seq analysis. Most publications of commonly used RNA-Seq analysis methods (Deng et al. 2014a; Dobin et al. 2013; Langmead et al. 2009b, 2010; Langmead and Salzberg 2012; Roberts et al. 2011, 2013b; Trapnell et al. 2009, 2012) often avoided mentioning read allocation to paralogous genes. The software tools associated with these publications share two simple options for handling matches to paralogous genes. The first is to use only uniquely matched reads, i.e., reads that match to multiple genes are simply ignored. The second is to assign such reads equally among matched genes. These options are obviously unsatisfactory. Here I describe a new approach which should substantially improve the accuracy of RNA-Seq data analysis such as gene expression characterization.

#### 3.1 Allocating Reads to a Two-Member Paralogous Gene Family

We need a few definitions to explain the allocation. Let  $L_1$  and  $L_2$  be the sequence length of the two paralogous genes. Let  $N_{U,1}$  and  $N_{U,2}$  be the number of reads that can be uniquely assigned to paralogous gene 1 or 2, respectively (i.e., the reads that matches one gene better than the other). Now for those reads that match the two genes equally well, the proportion of the reads contributed by paralogous gene 1 may be simply estimated as

$$P_1 = \frac{N_{U,1}}{N_{U,1} + N_{U,2}} \quad (5.1)$$

Now for any read that matches the two paralogous genes equally well, we will assign  $P_1$  to paralogous gene 1 and  $(1-P_1)$  to paralogous gene 2. In the extreme case when paralogous genes are all identical, then  $N_{U,1} = N_{U,2} = 0$ , and we will assign 1/2 of these equally matched read to genes 1 and 2. We should modify Eq. (5.1) to make it more generally applicable as follows:

$$P_1 = \frac{0.01 + N_{U,1}}{0.02 + N_{U,1} + N_{U,2}} \quad (5.2)$$

where 0.01 in the numerator and 0.02 in the denominator are pseudocounts. The treatment in Eq. (5.2) implies that when  $N_{U,1} = N_{U,2} = 0$  (e.g., when two paralogous genes are perfectly identical), then a read matching equally well to these paralogous genes will be equally divided among the two paralogues.

One problem with this treatment is its assumption of  $L_1 = L_2$ . If paralogous gene 1 is much longer than the other, then  $N_{U,1}$  is expected to be larger than  $N_{U,2}$ , everything else being equal. One may standardize  $N_{U,1}$  and  $N_{U,2}$  to number of unique matches per 1000 nt, designated by  $SN_{U,i} = 1000N_{U,i}/L_i$  (where  $i = 1$  or 2) and replace  $N_{U,i}$  in Eq. (5.2) by  $SN_{U,i}$  as follows (Mortazavi et al. 2008):

$$P_1 = \frac{0.01 + SN_{U,1}}{0.02 + SN_{U,1} + SN_{U,2}} = \frac{0.01 + \frac{1000N_{U,1}}{L_1}}{0.02 + 1000\left(\frac{N_{U,1}}{L_1} + \frac{N_{U,2}}{L_2}\right)} \quad (5.3)$$

### 3.2 Allocating Sequence Reads in Gene Family with More Than Two Members

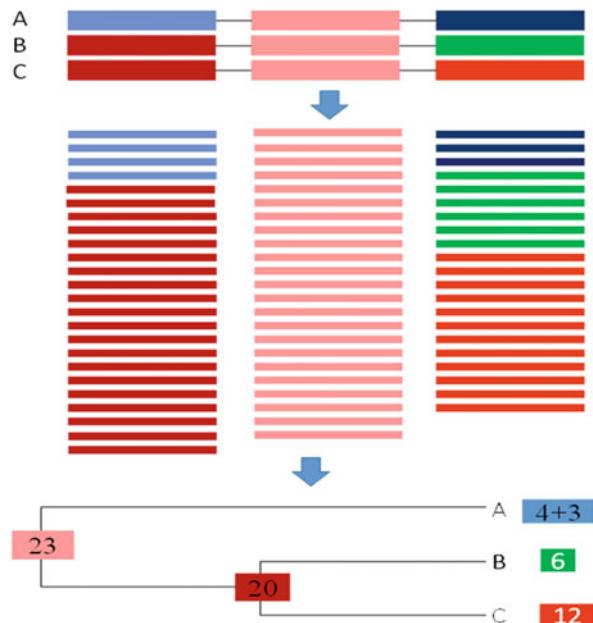
One might, mistakenly, think that it is quite simple to extend Eq. (5.3) for a gene family of two members to a gene family with  $F$  members by writing

$$P_i = \frac{0.01 + \frac{1000N_{U,i}}{L_i}}{0.01F + 1000 \sum_{i=1}^F \frac{N_{U,i}}{L_i}} \quad (5.4)$$

This does not work. For example, if we have three paralogous genes designated A, B, and C, respectively. Suppose that the gene duplication that gave rise to B and C occurred very recently so that B and C are identical, but A and the ancestor of B and C have diverged for a long time. In this case,  $N_{U,B} = N_{U,C} = 0$  because a read matching B will always match C equally well, but  $N_{U,A}$  may be greater than 0. This will result in unfair allocation of many transcripts from B and C to A according to Eq. (5.4). I outline the approach below for dealing with gene families with more than two members.

With three or more paralogous genes, one may benefit from a phylogenetic tree for proper allocation of sequence reads. I illustrate the simplest case with a gene family with three paralogous Genes A, B, and C idealized into three segments in Fig. 5.1. The three genes shared one identical middle segment with 23 matched reads (that necessarily match equally well to all three paralogues). Genes B and C share an identical first segment to which 20 reads matched. Gene A has its first segment different from that of B and C and got four matched reads. The three genes also have a diverged third segment where A matched 3 reads, B matched 6, and C matched 12. Our task is then to allocate the 23 reads shared by all three and 20 reads shared by B and C to the three paralogues.

**Fig. 5.1** Allocation of shared reads in a gene family with three paralogous Genes A, B, and C with three idealized segments with a conserved identical middle segment, strongly homologous first segment that is identical in B and C, and a diverged third segment. Reads and the gene segment they match to are of the same color



One could apply maximum likelihood or least-squares method for the estimation, but ARSDA uses a simple counting approach by applying the following:

$$\begin{aligned}
 P_A &= \frac{3 + 4}{3 + 4 + 20 + 6 + 12} = 0.15556 \\
 P_B &= (1 - P_A) \frac{6}{6 + 12} = 0.28148 \\
 P_C &= (1 - P_A) \frac{12}{6 + 12} = 0.56296
 \end{aligned} \tag{5.5}$$

Thus, we allocate the 23 reads (that matched three genes equally) to paralogous Genes A, B, and C according to  $P_A$ ,  $P_B$ , and  $P_C$ , respectively. For the 20 reads that matched B and C equally well, we allocate  $20*6/(6 + 12)$  to B and  $20*12/(6 + 12)$  to C. This gives the estimated number of matches to each gene as

$$\begin{aligned}
 N_A &= 3 + 4 + 23P_A = 10.57778 \\
 N_B &= 6 + 23P_B + 20\left(\frac{6}{6 + 12}\right) = 19.14074 \\
 N_C &= 12 + 23P_C + 20\left(\frac{12}{6 + 12}\right) = 38.28148
 \end{aligned} \tag{5.6}$$

These numbers are then normalized to give FPKM (Mortazavi et al. 2008). The current version of ARSDA assumes that gene families with more than two members have roughly the same sequence lengths. This is generally fine with prokaryotes but may become problematic with eukaryotes.

In practice, one can obtain the same results without actually undertaking the extremely slow process of building trees for paralogous genes. One first goes through reads shared by two paralogous genes (e.g., the 20 reads shared by Genes B and C in Fig. 5.1) and allocate the reads according to  $P_B = 6/(6 + 12) = 1/3$  and  $P_C = 12/(6 + 12) = 2/3$ . Now Genes B and C will have  $12.66667 (=6 + 20*P_B)$  and  $25.33333 (=12 + 20*P_C)$  assigned reads, i.e.,  $N_{U.B} = 12.66667$  and  $N_{U.C} = 25.33333$ . Once we have done with reads shared by two paralogous genes, we go through reads shared by three paralogous genes, e.g., the 23 reads shared by Genes A, B, and C in Fig. 5.1. With  $N_{U.A} = 7$ ,  $N_{U.B} = 12.66667$ ,  $N_{U.C} = 25.333333$ , and  $N = N_{U.A} + N_{U.B} + N_{U.C} = 45$ , so we have

$$\begin{aligned} P_A &= \frac{N_{U.A}}{N} = 0.15556 \\ P_B &= \frac{N_{U.B}}{N} = 0.28148 \\ P_C &= \frac{N_{U.C}}{N} = 0.56296 \end{aligned} \quad (5.7)$$

$$\begin{aligned} N_A &= 7 + 23P_A = 10.57778 \\ N_B &= 12.66667 + 23P_B = 19.14074 \\ N_C &= 25.33333 + 23P_C = 38.28148 \end{aligned} \quad (5.8)$$

which are the same as shown in Eq. (5.6). This progressive process continues until we have allocated reads shared by the largest number of paralogous genes.

The method above has been implemented in ARSDA (Xia 2017a) which is available at <http://dambe.bio.uottawa.ca/ARSDA/ARSDA.aspx>. A sample output of gene expression generated from ARSDA is shown in Table 5.2.

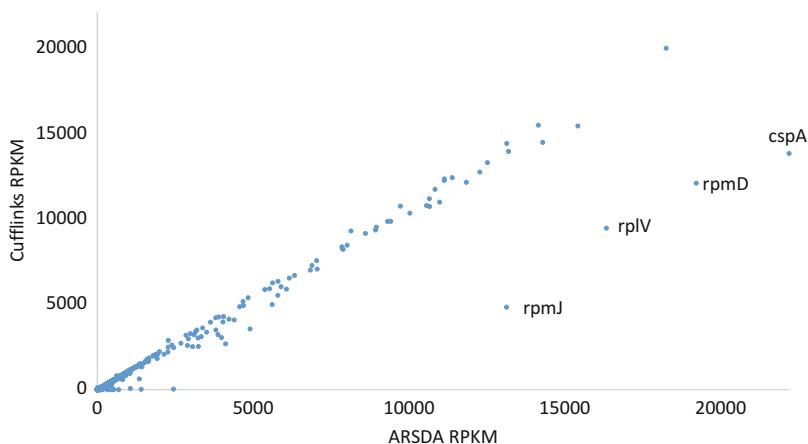
## 4 Comparison Between ARSDA and Cufflinks in Characterizing Gene Expression

The most frequently used software for gene expression is Cufflinks (Trapnell et al. 2012). Is ARSDA more accurate than Cufflinks in characterizing gene expression? I will address this question with a real data set, i.e., the transcriptomic data for an *E. coli* wild type (Pobre and Arraiano 2015), archived in NCBI's SRA database as SRR1536586.sra.

The Cufflinks-quantified gene expression for this file is in file GSM1465035\_WT.txt.gz from NCBI Geo DataSets GSM1465035. Gene expression from ARSDA and Cufflinks is mostly concordant (Fig. 5.2), but four points (labeled in Fig. 5.2) stand out as outliers (although many more discordant points will be revealed by a log-log plot). Such dramatic differences demand an explanation. Take *rpmJ*, for example. Either ARSDA severely overestimated or Cufflinks

**Table 5.2** Partial output of gene expression, with the gene locus\_tag (together with start and end sites) as Gene ID

Gene ID	SeqLen	Count	Count/Kb	FPKM
b0001l190_255	66	76	1151.515	389.894
b0002l337_2799	2463	2963	1203.004	407.328
b0003l2801_3733	933	1121	1201.501	406.819
b0004l3734_5020	1287	1782	1384.615	468.82
b0005l5234_5530	297	97	326.599	110.584
b0006lC5683_6459	777	113	145.431	49.242
b0007lC6529_7959	1431	143	99.93	33.836
b0008l8238_9191	954	1561	1636.268	554.028
b0009l9306_9893	588	289	491.497	166.417
b0010lC9928_10494	567	100	176.367	59.716
b0011lC10643_11356	714	13	18.207	6.165
b0013lC11382_11786	405	2	4.938	1.672
b0014l12163_14079	1917	6863	3580.073	1212.186
b0015l14168_15298	1131	1671	1477.454	500.255
...	...	...	...	...



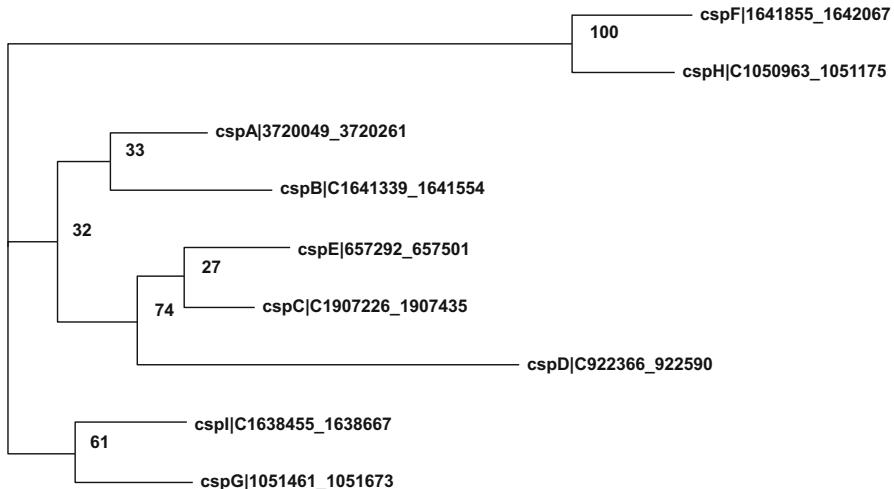
**Fig. 5.2** Contrast in gene expression (RPKM) between ARSDA and Cufflinks output for the same transcriptomic data in file SRR1536586.sra for *E. coli* wild type

severely underestimated the gene expression (Fig. 5.2). I originally expected the discrepancy to be due to different allocation of paralogous genes. The expectation is only partially true.

There are 6426 reads that can be mapped unambiguously to *rpmJ* (which is in fact a single-copy ribosomal protein gene). Although there are *rpmA*, *rpmB*, ..., *rpmJ* genes in *E. coli*, they are not paralogous. One particular read “AGTGCC-GAGCGGAAGCATAAACAGCGCCAAGGGCTGATTTCGCATATT” alone occurs 2684 times in SRR1536586.sra. It matches perfectly to the 36 nt at the 3'

end of *rpmJ* and 14 nt immediately downstream. However, Cufflinks output reported a count of only 2114 reads for *rpmJ* instead of 6426 (and consequently the much reduced RPKM in Fig. 5.2). I checked if *rpmJ* is in an operon with an immediate downstream gene so that a read overlapping *rpmJ* and the downstream gene might be divided between the two in Cufflinks. However, the downstream gene, which is *rpsM*, is 146 nt away. It is difficult to reconcile 6426 nonambiguous read matches to Cufflinks' 2114. Similarly, *rpmD* and *rplV* (Fig. 5.2) have 14,468 and 22,747 unambiguous read matches, respectively, but the corresponding counts in Cufflinks output are only 8108 and 11,801, respectively. Note that *rpmD* and *rplV* are also single-copy genes with no ambiguous read matches. *E. coli* genes *rpmA* to *rpmJ* are not paralogous, neither are *rplA* to *rplY*.

The last outlying gene (*cspA* in Fig. 5.2) does involve a paralogous gene family (Fig. 5.3). *cspA* has 19,776 unambiguous read matches, but Cufflinks output has only 10,957 which resulted in a much lower RPKM than that from ARSDA (Fig. 5.2). Most genes in Fig. 5.2 are sufficiently diverged except for *cspF* and *cspH*. There are 264 unambiguous read matches to *cspF* and 58 to *cspH*. There are also 55 reads that match well to both *cspF* and *cspH*, with 27 of them matching *cspF* better than *cspH* and 28 matching *cspH* better than *cspF*. So we may assign (264 + 27) reads to *cspF* and (58 + 28) reads to *cspH*, with relative proportions of 0.7719 and 0.2281 for *cspF* and *cspH*, respectively. Twelve reads match *cspF* and *cspH* equally well (the same e-value and the same BitScore), so we assign them



**Fig. 5.3** Phylogenetic relationship among paralogous genes *cspA* to *cspI* in *E. coli*, based on coding sequences, with bootstrap values next to internal nodes. Sequences were aligned by MAFFT (Katoh and Toh 2008) with accurate L\_INS-i option and a maximum of 16 iterations. Coding sequences were first translated in amino acid sequences which are aligned with BLOSUM62 matrix. Nucleotide sequences were then aligned against aligned amino acid sequences. Phylogenetic analysis was done with PhyML (Guindon et al. 2010). All these analyses were automated in DAMBE (Xia 2013, 2017c)

proportionally to the two genes, i.e.,  $12 * 0.7719$  to *cspF* and  $12 * 0.2281$  to *cspH*. The final counts for *cspF* and *cspH* are 300.2626 and 88.7374, respectively. However, Cufflinks output shows counts of 2 and 63 for *cspF* and *cspH*, respectively. The discrepancy is particularly striking given that gene expression from ARSDA and Cufflinks are mostly concordant (Fig. 5.2). The alternative ways of allocating paralogous genes (Xia 2017a) does not help reconcile the discrepancy. Analyzing the same data set with Rockhopper (Tjaden 2015) resulted in RPKM values similar to those from ARSDA. I hope that these numbers will prompt authors of Cufflinks to be more explicit about how they treat counts. One aspect of RNA-Seq data that we have not discussed is that one segment of an mRNA may be represented by many reads but another segment of the same mRNA is not represented at all. It is possible that Cufflinks may have implemented some corrections for this leading to discrepancies.

## 5 Use Transcriptomic Data to Refine Genome Annotation

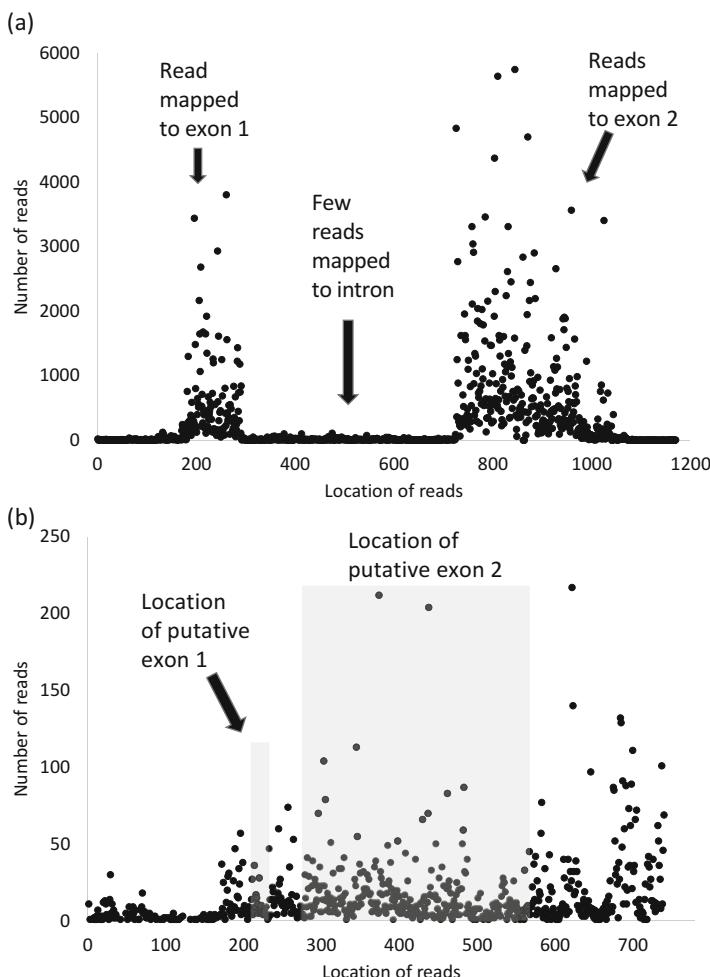
I will use two intron-containing genes (ICGs, Table 5.3) in the yeast to illustrate the application of transcriptomic data to refine genome annotation. *YKL006W/RPL14A* is a typical ICG in yeast chromosome 11 (NC\_001143), whereas *YJR112W-A* is a putative ICG in yeast chromosome 10 (NC\_001142, 637790..637809,637859..638168). According to GenBank annotation, both genes have two exons with a single intron in between.

We will address two questions concerning *YJR112W-A*. First, is it transcribed (which is the minimum empirical confirmation that it is a protein-coding gene)? Second, does it really have an intron correctly annotated in the GenBank file? There are two reasons for doubting the accuracy of the GenBank annotation file. First, the intron does not end with YAG (where Y stands for a pyrimidine) as do other introns. Second, it does not have a branchpoint site bearing any similarity to the consensus UACUAAC. We also have a question on both genes. If the intron annotation is correct, how efficiently are the introns in the two genes spliced? We can answer these questions by using yeast transcriptomic data (Rahi et al. 2016) consisting of 44 SRA files deposited in the SRA database hosted in NCBI (BioProject accession: PRJNA319604). The total number of reads in these 44 files is 393,355,070, each read being about 50 nt long.

**Table 5.3** Descriptive information of the two yeast genes according to GenBank annotation

Gene	YJR112W-A	YKL006W
Chromosome	10	11
Accession	NC_001142	NC_001143
L_E1	20	129
L_intron	49	398
L_E2	310	288

Figure 5.4 displays the pattern of mapping transcriptomic reads to the two ICGs. For *YKL006W/RPL14A* (Fig. 5.4a), many reads mapped to the two exons, but very few mapped to the intron region, suggesting very efficient splicing as well as rapid lariat degradation. In contrast, there is no indication of an intron in *YJR112W-A* (Fig. 5.4b), although a more direct validation is simply to map the reads against exon-exon junctions. Figure 5.4b also shows that *YJR112W-A* is clearly transcribed. However, although the transcription start is about 30 nt upstream of the start codon, there is no clear transcription termination (Fig. 5.4b). The gene is likely much longer than what is annotated in the GenBank file.

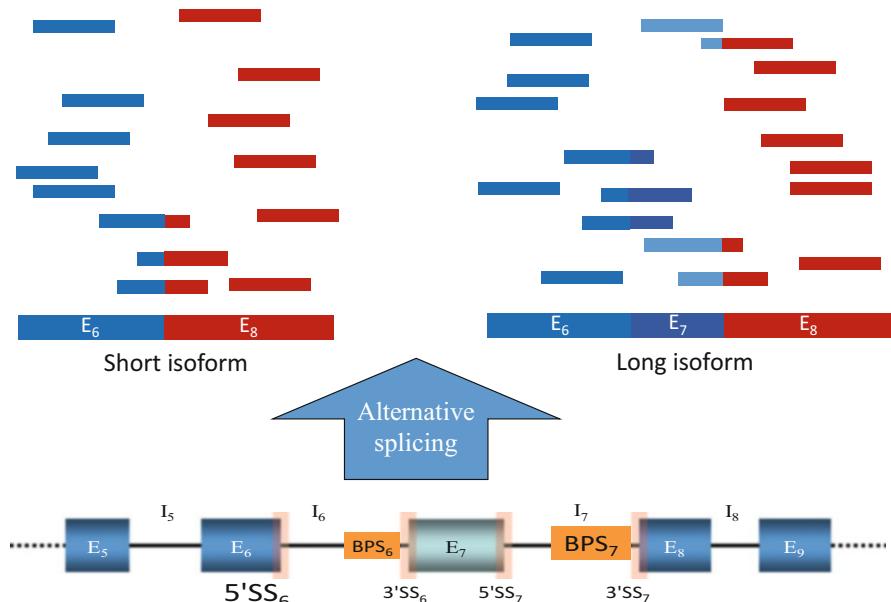


**Fig. 5.4** Pattern of transcriptomic reads mapped to two yeast intron-containing genes (ICGs): (a) well-characterized *YKL006W/RPL14A* and (b) putative *YJR112W-A*. “Location of reads” axis includes 200 nt upstream of the start codon (which starts at site 201) and 200 nt downstream of the stop codon. Each point represents a set of transcriptomic reads with the first site mapped to the same location of the gene sequence. Analyzed by ARSDA (Xia 2017a)

## 6 Other Applications Using RNA-Seq Data

RNA-Seq data has been used to test hypothesis on alternative splicing. The gene coding ubiquitin-specific protease 4 (USP4) in human and mouse has 22 exons, of which exon 7 ( $E_7$ ) is often skipped (Fig. 5.5, bottom). RNA-Seq data for human and mouse have both reads matching the junction between  $E_6$  and  $E_8$  (short isoform) and the junctions between  $E_6$  and  $E_7$  and between  $E_7$  and  $E_8$  (Fig. 5.5). Vlasschaert et al. (2016) hypothesized that, if  $BPS_6$  and  $5'SS_6$  are weak and  $BPS_7$  and  $5'SS_7$  strong, then the spliceosome will treat  $E_7$  and its flanking introns as one long intron so that  $BPS_7$  will attack  $5'SS_6$  instead of  $5'SS_7$ , leading to skipping of  $E_7$ . Splicing signal strengths for  $5'SS_7$  and  $BPS_6$ , as characterized by PWMS detailed in a previous chapter, are indeed weaker than those for  $BPS_7$  and  $5'SS_6$  (Vlasschaert et al. 2016), consistent with the hypothesis. One may further predict that there should be no  $E_7$ -skipping in vertebrate lineages where splice signal at  $5'SS_7$  is not weaker than that for  $5'SS_6$ . A search of *Usp4* gene in 14 vertebrate species revealed 2 species (zebra fish and chicken) whose PWMS for  $5'SS_7$  is not weaker than  $5'SS_6$ , and indeed  $E_7$  is not skipped in these 2 species (Vlasschaert et al. 2016), because all reads from their RNA-Seq data are consistent with those from the long isoform (Fig. 5.5).

In bacterial species, the localization of the start codon in many genes is accomplished by the base pairing between Shine-Dalgarno (SD) sequence on mRNA and



**Fig. 5.5** Partial *USP4* mRNA with schematic representation of exon-intron configuration and splice signals such 5' and 3' splice sites (5'SS and 3'SS) and branchpoint site (BPS). Strong 5'SS<sub>6</sub> and BPS<sub>7</sub> coupled with weak BPS<sub>6</sub> and 5'SS<sub>7</sub> are hypothesized to result in two alternative splicing isoforms, short isoform (with  $E_7$  skipped) and long form

the anti-SD (aSD) sequence on 3' tail of small subunit rRNA (3'TAIL). A change in 3' end of SSU rRNA (3'TAIL for short) in evolution will alter SD usage in protein-coding genes in descendent lineages, especially in highly expressed ones (Abolbaghaei et al. 2017). Studying coevolution between SD and aSD among different bacterial lineages requires accurate determination of 3'TAIL. Because of RNA processing and degradation, 3'TAIL is heterogeneous in the cell. RNA-Seq data provide an accurate and quantitative characterization of the rRNA pool with heterogeneous 3'TAIL (Wei et al. 2017).

Transcriptomic data has been used in large-scale gene discovery in the genome, soon after the invention of DNA microarray (Shoemaker et al. 2001) and SAGE (Saha et al. 2002). That many protein-coding genes may remain unannotated is highlighted by the finding that even the extensively studied phage lambda may have unannotated protein-coding genes (Liu et al. 2013). In human and mouse, ribosomes are frequently found on transcripts not annotated as coding sequences, with the consequent production of polypeptides (Yoon et al. 2014). The ENCODE pilot project shows that “the genome is pervasively transcribed, such that the majority of its bases can be found in primary transcripts” (Birney et al. 2007). Many of these transcripts could be functional genes, but only a very small fraction of them are annotated as known protein or RNA genes.

mRNA abundance, coupled with a measure of translation rate, can be used to predict the rate of protein production. The difference between protein production and protein abundance allows us to evaluate the protein degradation rate. Protein translation rate can now be assessed by several methods. The first is by using ribosomal profiling data, traditionally from microarray (Arava et al. 2003; MacKay et al. 2004) and now almost exclusively from deep sequencing of ribosome-protected fragments (RPF, ~30 nucleotides) of mRNA (Ingolia 2010, 2014, 2016; Ingolia et al. 2009). The two experimental approaches, however, exhibit high concordance with data from the yeast (Xia et al. 2011). If Genes A and B have mRNA abundance values  $N_A$  and  $N_B$ , respectively, from transcriptomic data, and their translation efficiency is  $R_A$  and  $R_B$ , respectively, from ribosome profiling data, then their relative protein production rate is  $N_A * R_A$  and  $N_B * R_B$ , respectively. Such prediction should be facilitated by obtaining transcriptomic and proteomic data in the same experiment (Smircich et al. 2015), ideally from a single cell (Heath et al. 2016; Saadatpour et al. 2015; Trapnell 2015; Wu and Tzanakakis 2013).

## Postscript

The most frequent use of RNA-Seq data is to identify differentially expressed genes between a patient and a normal person (between an animal disease model and a normal animal). However, the identified differences may often lead to wrong prescriptions instead of a good cure.

In the mid-1970s, the United Nations noted, correctly, that many differences in people between poor regions and wealthy regions could be attributed to malnutrition.

Consequently, it made a valiant effort to ship milk powder to poor regions in Africa and Asia to alleviate the perennial problem of malnutrition, without knowing the problem of lactose intolerance shared by many in these poor regions. The result was unsatisfactory and exemplifies the evil of one-dimensional thinking.

Alcohol consumption in a city in North America is almost perfectly correlated with number of churches in a city. Is it because people are happier and party more with more churches? Most people would immediately recognize that the correlation is due to the fact that both alcohol consumption and number of churches depend on population size of the city. The larger the city, the more the people and consequently the more churches and the more alcohol consumption.

# Chapter 6

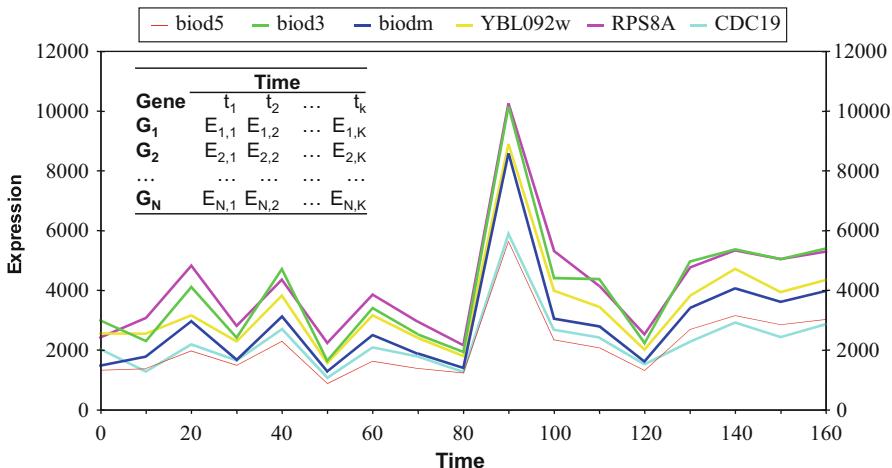
## Self-Organizing Map and Other Clustering Methods in Transcriptomics



### 1 Introduction

Transcriptomic data can be used to solve many problems such as refining genome annotation, predicting protein production when a measure of translation efficiency is available, identifying co-expressed genes to facilitate inference of co-translated genes, quantifying differential gene expression between individuals of different phenotypes, characterizing relative abundance and dynamics of alternatively spliced isoforms, and evaluating the relative usage of different transcription start and termination sites. However, in this chapter, we will focus on only one aspect of transcriptomic data analysis, and that is to use clustering algorithms to identify co-expressed genes and eventually identify co-regulated genes. Co-expressed genes can often lead to co-regulated genes. For example, a set of co-expressed genes may share a transcription binding site which is often located upstream of the regulated genes. We may extract such upstream sequences and run them through Gibbs sampler to find the nature and location of the shared regulatory motif.

The transcriptomic data for identifying co-expressed genes are typically from taking transcriptomic snapshots of cells over time. For example, one could synchronize yeast cell cycle, obtaining gene expression data at time  $t_0, t_1, t_2, \dots, t_k$ . Similarly, one can take such snapshots during the process when stem cells are triggered to differentiate into specialized cell types. The resulting data is an  $N \times K$  matrix where  $N$  is the number of genes and  $K$  is the number of time points (Fig. 6.1, inset). The objective is to find which genes have their transcript abundance increased or decreased synchronously (Fig. 6.1). Those genes that do increase or decrease synchronously in transcript abundance become candidates of co-regulated genes. One can then check if they share regulatory motifs that bind to the same transcription factor. Because regulatory motifs are often located upstream of the start codon, one can retrieve the upstream sequences and subject them to Gibbs sampler, covered in a previous chapter, to identify the shared motifs.



**Fig. 6.1** A subset of six co-expressed genes from yeast gene expression data (Cho et al. 1998). These genes have similar expression profile and form a tight cluster in a gene expression tree built from distances that measure differences in expression profiles. Generated from AMADA (Xia and Xie 2001a)

Co-expressed genes that have similar transcription profiles (Fig. 6.1) can be identified as forming tight clusters. Two classes of cluster algorithms are frequently used, (1) hierarchical and nonhierarchical. Representatives of the hierarchical clustering algorithms include conventional single-linkage, complete-linkage, and average-linkage algorithms, with the average linkage being used most often. The UPGMA (unweighted pair-group method with arithmetic mean) algorithm used during the early stage of molecular phylogenetics (Sneath 1962; Sokal and Michener 1958) is one of the average-linkage algorithms. Representatives of the nonhierarchical clustering algorithms include the K-means (Hartigan 1975) and self-organizing map or SOM (Kohonen 2001). SOM is used extensively in analyzing transcriptomic data (Chen et al. 2001; Covell et al. 2003; Kim et al. 2005; Lamendola et al. 2003; Ordway et al. 2005; Seo et al. 2005; Toronen et al. 1999; Trutschl et al. 2005; Wang et al. 2002; Xiao et al. 2003). I will numerically illustrate two algorithms, the UPGMA algorithm (Sneath 1962) as a representative of the hierarchical clustering and the SOM algorithm (Kohonen 2001) as a representative of the nonhierarchical clustering. These two algorithms are perhaps the most frequently used in analyzing transcriptomic data to identify co-expressed genes. Because almost all clustering algorithms require a distance or a similarity index, I will also have a subsection on distances and similarities to help the reader choose which distance or similarity index to use.

## 2 Similarity and Distance Indices

Almost all clustering algorithms require a distance or a similarity index. So we need to have a basic understanding of distances and similarities. Representative distances that have been used in gene expression studies include the scale-dependent Euclidean distance (Bickel 2003; Sawa and Ohno-Machado 2003) and scale-independent Mahalanobis distance (Chilingaryan et al. 2002) and  $(1 - r)$  where  $r$  is Pearson correlation coefficient (Bickel 2003; Eisen et al. 1998; Sawa and Ohno-Machado 2003). Other distances that could be used in clustering algorithms include Manhattan metric, percent remoteness, chord distance, and geodesic distance (Pielou 1984).

Euclidean distance is perhaps the most used distance. Given a vector  $x = [x_1, x_2, \dots, x_N]$  and another vector  $y = [y_1, y_2, \dots, y_N]$  in an  $N$ -dimensional space, Euclidean distance ( $d$ ) is defined as

$$d = \sqrt{\sum_{i=1}^N (x_i - y_i)^2} \quad (6.1)$$

Mahalanobis distance becomes identical to Euclidean distance with standardized data, i.e., when variable  $X$  is transformed to  $z$  by

$$z_i = \frac{X_i - \bar{X}}{s_X} \quad (6.2)$$

so that mean and standard deviation of the resulting variable  $z$  is 0 and 1, respectively. Euclidean distance based on standardized data and  $(1-r)$  are perhaps used most frequently in clustering gene expression data. Representative similarity indices include various correlation coefficients such as the parametric Pearson correlation and nonparametric Spearman correlation and others.

An ideal distance or similarity index for clustering analysis should be metric. Nonmetric distances are way more likely to produce negative branch lengths than metric distances in cluster analysis, when the least-squares criterion is used to fit the distance matrix to the tree. Negative branch lengths are difficult to interpret. In addition, metric distances enable the points to be visualized in an  $n$ -dimensional space, whereas nonmetric distances do not.

So what is a distance in the first place? Designating  $x$  and  $y$  as two points in space, a distance is defined to have the following properties:

$$D(x,x) = 0 \text{ (distance of a point to itself, which is typically 0)}$$

$$D(x,y) \geq d_0$$

$$D(x,y) = D(y,x)$$

What is a metric distance? Designating  $x$ ,  $y$ , and  $z$  as three points in space, a metric distance is defined to have the following additional properties:

$$D(x,y) = d_0 \text{ if and only if } x = y.$$

$$D(x,z) \leq D(x,y) + D(y,z), \text{ or triangular inequality in Euclid geometry.}$$

An example of a metric distance is the Euclidean distance. If  $d$  is a metric distance (e.g., Euclidean  $d$ ), then the following are metric similarities:

$$1/d$$

$Cd$ , where  $C$  is a constant

$$d_{\max} - d$$

If  $s$  is a metric similarity index, then the following are also metric distances:

$$1/s$$

$Cs$ , where  $C$  is a constant

$$s_{\max} - s$$

An example of a nonmetric distance, given in Pielou [Pielou 1984 #6058], is

$$D(A, B) = \frac{D_{\max}}{D_{\max} - d(A, B)} \quad (6.3)$$

where  $d(A, B)$  is the Euclidian distance between points A and B. Suppose we have three points (A, B, and C) with Euclidian distances  $d(A, B)$ ,  $d(A, C)$ , and  $d(B, C)$ . The condition for satisfying triangle inequality for  $D(A, B)$ ,  $D(A, C)$ , and  $D(B, C)$  is

$$\frac{D_{\max}}{D_{\max} - d(A, B)} \leq \frac{D_{\max}}{D_{\max} - d(A, C)} + \frac{D_{\max}}{D_{\max} - d(B, C)} \quad (6.4)$$

which implies that the triangle inequality will be violated unless  $d(A, B) = d(A, C) = d(B, C)$ . In other words,  $D(A, B)$  is not a metric distance.

Pearson  $r$  is a similarity index, and its maximum is 1. Is  $(1 - r)$ , which is frequently used in clustering microarray data, a metric distance? To satisfy the triangle inequality, we need to have

$$\begin{aligned} 1 - r_{A,B} &\leq 1 - r_{A,C} + 1 - r_{B,C} \\ r_{A,C} + r_{B,C} &\leq 1 + r_{A,B} \end{aligned} \quad (6.5)$$

Thus, the triangle inequality cannot be satisfied unless the three  $r$  values are identical. For example, the inequality is violated if  $r_{A,B}$  is the largest of the three  $r$  values. In this sense, using Euclidean distance computed from standardized variables for clustering is better than using  $(1 - r)$ . Euclidean distance is scale-dependent, but  $(1 - r)$  is scale-independent. The scale effect is illustrated in Fig. 6.2. We note that Gene 1 and Gene 3 increase and decrease in their expression synchronously, so do Gene 2 and Gene 4. In other words, Gene 1 and Gene 3 are co-expressed, so are Gene 2 and Gene 4. However, because Gene 1 and Gene 2 are highly expressed and Gene 3 and Gene 4 are relatively lowly expressed, the synchronous change in gene expression between Gene 1 and Gene 3 and between Gene 2 and Gene 4 will not be reflected by Euclidean distance on the original data, and the clustering analysis will not help us discover co-expressed genes.

You may note that, before standardization, Euclidean distance  $d_{12}$  between Gene 1 and Gene 2 is smaller than  $d_{13}$  or  $d_{14}$ , and  $d_{34}$  is the smallest of all pairwise

	T0	T10	T20	T30	T40	T50	T60	T70
Gene 1	600	200	300	600	300	500	300	300
Gene 2	300	500	400	700	400	200	400	400
Gene 3	60	20	30	60	30	50	30	30
Gene 4	30	50	40	70	40	20	40	40

After standardization								
	T0	T10	T20	T30	T40	T50	T60	T70
Gene 1	1.369	-1.208	-0.564	1.369	-0.564	0.725	-0.564	-0.564
Gene 2	-0.772	0.600	-0.086	1.972	-0.086	-1.458	-0.086	-0.086
Gene 3	1.369	-1.208	-0.564	1.369	-0.564	0.725	-0.564	-0.564
Gene 4	-0.772	0.600	-0.086	1.972	-0.086	-1.458	-0.086	-0.086

**Fig. 6.2** Original data (top) and standardized data (bottom) to illustrate the effect of scale. T0, T10, etc., indicate time points. Standardization refers to the data transformation specified in Eq. (6.2)

distances. Application of a clustering algorithm typically will cluster Gene 3 and Gene 4 together and then Gene 1 and Gene 2 together. In other words, the clustering analysis does not cluster co-expressed genes together.

After standardization (Fig. 6.2, bottom half) which removes the scale effect,  $d_{13} = d_{24} = 0$ , which implies that Gene 1 should be clustered with Gene 3 and Gene 2 should be clustered with Gene 4.

In contrast to Euclidean distance which differs between standardized and non-standardized data, the distance  $d' = (1 - r)$  is scale-independent, and  $d'_{13} = d'_{24} = 0$  for either original or standardized data. Applying any cluster algorithm will result in Gene 1 and Gene 3 clustered together and Gene 2 and Gene 4 clustered together, after standardization. If our purpose is to cluster co-expressed genes together, then using  $d'$  with either original or standardized data or using  $d$  with standardized data is better than using  $d$  with the original data.

Note that one does not have to standardize the data to remove the scale effect. For example, one can just transform the data so that all variables will have the same mean and variance, but the mean and variance do not need to be equal to 0 and 1, respectively.

### 3 Clustering Algorithms

#### 3.1 UPGMA: A Hierarchical Clustering Algorithm

We have learned two frequently used distance measures in the previous section, the Euclidean distance and  $(1 - r)$ . Given  $N$  genes, there are  $N(N-1)/2$  pairwise

distances (designated as  $d_{i,j}$  between gene  $i$  and gene  $j$ ). We can apply the UPGMA algorithm to perform hierarchical clustering. Readers may check the chapter of molecular phylogenetics for more tree-building methods.

A matrix of  $d_{i,j}$  values for five genes (designated as  $G_i$ ) is shown in Fig. 6.3. At this point we do not know the relationship among the genes, and our lack of knowledge is represented by what is called a star tree, represented as  $(G_1, G_2, G_3, G_4, G_5)$ . The UPGMA algorithm starts by finding the smallest  $d_{i,j}$  (designated as  $d_{\min}$ ) in the matrix and clustering the two associated genes. There are two smallest  $d_{i,j}$  in the matrix,  $d_{1,3} = d_{4,5} = 0.0076$ . So what should we do?

In this particular example, we can start by either clustering  $G_1$  and  $G_3$  or clustering  $G_4$  and  $G_5$ , and the final tree will be the same. However, identical  $d_{i,j}$  values sometimes may lead to alternative clustering outcome. In particular, if  $d_{i,j} = d_{i,k} = d_{\min}$  (or  $d_{i,j} = d_{j,k} = d_{\min}$ ), then there will always be alternative trees, because there is a clear conflict between clustering  $(G_i, G_j)$  and clustering  $(G_i, G_k)$ . In other words, clustering  $G_i$  and  $G_j$  rules out the possibility of clustering  $G_i$  and  $G_k$  and vice versa. The neighbor-joining algorithm (Saitou and Nei 1987) sometimes also experiences the problem of conflicting trees, although almost all phylogenetic programs implementing the UPGMA and neighbor-joining algorithms typically output only a single tree. While elegant algorithms are available to handle conflicts (Murtagh 1984), researchers typically feel more comfortable with just one tree.

Fortunately, our simple example does not require us to resolve such a conflict. So we will proceed to cluster  $G_1$  and  $G_3$ . This leads to a slightly more structured tree  $((G_1, G_3), G_2, G_4, G_5)$ , together with a reduced matrix shown in Fig. 6.4.

Note that some distances (e.g.,  $d_{2,4}$ ,  $d_{2,5}$ ,  $d_{4,5}$ ) in Fig. 6.4 are directly transferred from the matrix in Fig. 6.3. There are three new distances in the reduced matrix (Fig. 6.4), computed as

**Fig. 6.3** Distance matrix for illustrating UPGMA

	G1	G2	G3	G4
G2	0.0916			
G3	0.0076	0.0840		
G4	0.0611	0.0611	0.0534	
G5	0.0534	0.0687	0.0458	0.0076

**Fig. 6.4** Intermediate result of UPGMA after clustering  $G_1$  and  $G_3$

	( $G_1, G_3$ )	$G_2$	$G_4$
$G_2$	0.0878		
$G_4$	0.0573	0.0611	
$G_5$	0.0496	0.0687	0.0076

```

graph TD
    S(( )) --- G1[G1]
    S --- G3[G3]
    G1 --- G2[G2]
    G1 --- G4[G4]
    G1 --- G5[G5]
  
```

$$\begin{aligned}
 d_{2,(1,3)} &= \frac{d_{1,2} + d_{2,3}}{2} = \frac{0.0916 + 0.0840}{2} = 0.0878 \\
 d_{4,(1,3)} &= \frac{d_{1,4} + d_{3,4}}{2} = \frac{0.0611 + 0.0534}{2} = 0.0573 \\
 d_{5,(1,3)} &= \frac{d_{1,5} + d_{3,5}}{2} = 0.0496
 \end{aligned} \tag{6.6}$$

Now we again find  $d_{\min}$  in the reduced matrix in Fig. 6.4, which is  $d_{45} = 0.0076$ . So we cluster  $G_4$  and  $G_5$  and obtain a more structured tree, together with a more reduced new matrix (Fig. 6.5).

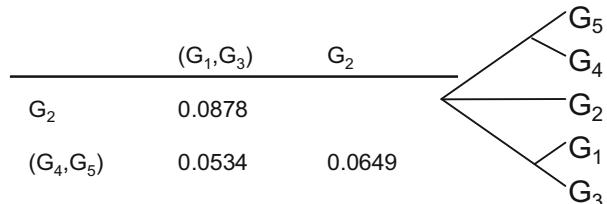
There are two new distances in Fig. 6.5, i.e.,

$$\begin{aligned}
 d_{(1,3),(4,5)} &= \frac{d_{(1,3),4} + d_{(1,3),5}}{2} = 0.0534 \\
 d_{2,(4,5)} &= \frac{d_{2,4} + d_{2,5}}{2} = 0.0649
 \end{aligned} \tag{6.7}$$

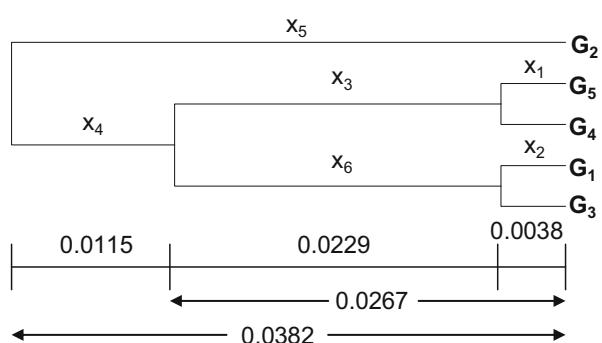
The smallest distance now is  $d_{(1,3),(4,5)} = 0.0534$ . This implies the clustering of  $(G_1, G_3)$  with  $(G_4, G_5)$ . Now we have a fully resolved tree (Fig. 6.6), together with the last distance computed as

$$d_{((1,3),(4,5)),2} = \frac{d_{(1,3),2} + d_{(4,5),2}}{2} = 0.0764 \tag{6.8}$$

**Fig. 6.5** Intermediate result of UPGMA

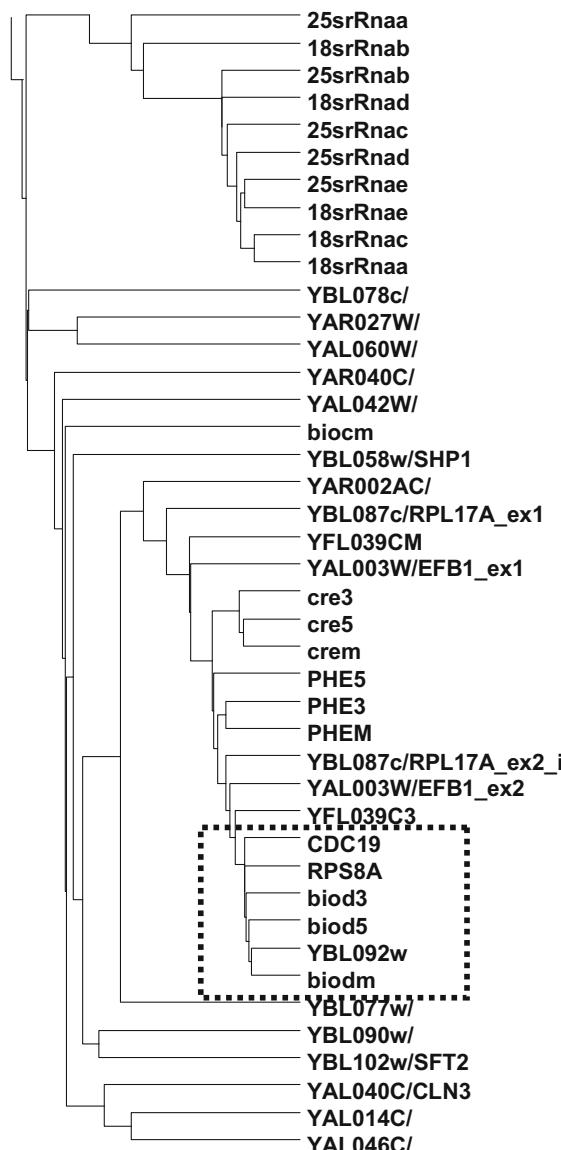


**Fig. 6.6** Final UPGMA tree



The branch lengths of the tree can be easily computed. Because  $d_{1,3} = d_{45} = 0.0076$ , we have  $x_1 = x_2 = d_{4,5}/2 = d_{1,3}/2 = 0.0038$ , i.e., the branch length from  $G_1$  or  $G_3$  to their closest shared node is 0.0038. Similarly,  $d_{(1,3),(4,5)} = 0.0534$ , so we have  $(x_1 + x_3) = (x_2 + x_6) = d_{(1,3),(4,5)}/2 = 0.0267$ . Because  $x_1 = 0.0038$ , so  $x_3 = 0.0267 - x_1 = 0.0229$ . Finally, because  $d_{2,((1,3),(4,5))} = 0.0764$ , so  $x_5 = 0.0764/2 = 0.0382$ . This also implies that  $(x_1 + x_3 + x_4) = 0.0382$ , so  $x_4 = 0.0382 - x_1 - x_3 = 0.0115$ . Also note that, in our example,  $x_3 = x_6$ . Fig. 6.7 shows a more realistic output from a cluster analysis.

**Fig. 6.7** Partial output from clustering analysis of gene expression data in the yeast, *Saccharomyces cerevisiae*. Only the first 200 genes from the original data (Cho et al. 1998) were used. The expression profiles of the six genes within the box of dashed lines are previously shown in Fig. 6.1. Tree generated by AMIADA (Xia and Xie 2001a)



Genes clustered together with short branch lengths connecting them should have similar expression profiles and are considered as co-expressed genes. I will illustrate the application of clustering analysis with a real example. The budding yeast (*Saccharomyces cerevisiae*) has a very useful property that the development of yeast cells in a culture can be synchronized. This allows one to monitor gene expression during the progression through the yeast cell cycle. In one of such studies (Cho et al. 1998), 6220 transcripts were monitored, and the data set is available to the public. Application of the UPGMA algorithm to the yeast gene expression data results in many clusters of co-expressed genes. Figure 6.7 displays a partial output of the clustering analysis using UPGMA and the standardized Euclidean distance for the first 200 genes.

My program AMIADA (Xia and Xie 2001a) allows the user to easily visualize gene expression profiles. Right-clicking a node on the tree and choosing “Plot expression profiles” will display the expression profile of genes clustered under a node. Figure 6.1 shows a plot of the expression profiles for a set of six genes clustered together in Fig. 6.7. The synchronized increase and decrease in their expression is obvious. Finding co-expressed genes is the first step toward identifying co-regulated genes, i.e., genes that share regulatory sequences.

## 3.2 Self-Organizing Map (SOM): A Nonhierarchical Cluster Algorithm

Being one of the unsupervised learning algorithms, SOM, like UPGMA, takes data that do not have prior group affiliation. It takes a training data set and goes through a training process to obtain a SOM of nodes (or artificial neurons) which can then be used for classification. Because SOM is also one of the artificial neural network (ANN) algorithms, it is necessarily associated with concepts such as nodes (neurons) and learning rate. All ANN algorithms have neurons and need learning (training).

### 3.2.1 The SOM Algorithm

We start with data in Table 6.1, where data are not standardized. If one is interested only in uncovering co-expressed genes, then one should standardize the data. The computation is the same regardless of our data being standardized or not.

Data in Table 6.1 is our training data. Most computation in SOM is in the training part. Once we have finished training, we will be able to use the finished SOM for classification of data points which are not in the training set.

We first need to decide the size of SOM, i.e., the number of nodes to have and whether the nodes should be arranged in one dimension, two dimensions, or higher dimensions. Most SOMs use a two-dimensional grid of nodes. Choosing the right number of nodes may be tricky. Too many nodes increase computation, and too few

**Table 6.1** Fictitious gene expression data for illustrating the SOM algorithm

Gene	T0	T10	T20	Sum
1	93	76	87	256
2	80	81	85	246
3	89	88	85	262
4	69	74	96	239
5	95	89	93	277
6	65	96	76	237
7	87	85	96	268
8	78	89	88	255
9	87	80	97	264
10	67	96	55	218
11	91	90	95	276
12	76	72	67	215
13	79	78	94	251
14	96	76	78	250
15	66	64	63	193

T0, T10, and T20 represent three time points. The values are between 0 and 100. Real data would have many more genes and more time points

**Table 6.2** A  $3 \times 3$  grid of nine nodes each with three randomly initialized values T0, T10, and T20

	1			2			3		
	T0	T10	T20	T0	T10	T20	T0	T10	T20
1	2.2	11.5	33.4	41.9	27.6	0.8	6	63.8	51.2
2	28.5	30.2	47	28.7	51.3	9	61.6	38.2	17.9
3	40.5	76.1	71.2	79.8	94.6	23.2	76.2	40.9	23.9

nodes may not provide sufficient fit to the data. For example, if we have only two nodes, then forcing all data points into these two nodes may result in very heterogeneous groupings. We will learn later that a finished SOM provides some ways for us to see poorly fit points. If such points do exist, then we should rerun SOM with a larger grid of nodes.

Let us just start with a  $3 \times 3$  grid of nine nodes (Table 6.2), with randomly initialized values between 0 and 100, which are the expected lower and upper bounds of gene expression data. Later on, we will learn a better way of initialization by principal component analysis. The random initialization symbolizes a beginning of no knowledge. The knowledge will be gained through the learning process.

We now randomly choose one gene, and suppose we happen to have chosen Gene 4 with T0, T10, and T20 equal to 69, 74, and 96, respectively (Table 6.1). The Euclidean distances (designated hereafter as  $d$ ) between this gene, and each of the nine nodes (Table 6.3) shows that Gene 4 is closest to node (3,1), with

**Table 6.3** Euclidean distance between Gene 4 and each of the nine nodes

	1	2	3
1	111.0	109.0	78.0
2	77.2	98.5	86.2
3	37.8	76.4	79.6

$$d = \sqrt{(69 - 40.5)^2 + (74 - 76.1)^2 + (96 - 71.2)^2} = 37.8378 \quad (6.9)$$

This node, with the smallest  $d$  to Gene 4, is then called a winning node. You may use a distance other than the Euclidean, but the procedure is the same, i.e., you find the winning node which has the smallest distance to Gene 4.

The winning node is the node that will get the first chance to learn from Gene 4, and its three values will be updated as a consequence of the learning. Recall that SOM is an algorithm in neural networks, and all neural networks learn. The updated values are given by the following equation referred to hereafter as a learning function:

$$w'_i = w_i(1 - \alpha) + p_i\alpha \quad (6.10)$$

where  $w_1$ ,  $w_2$ , and  $w_3$  refer to the winning node's values in T0, T10, and T20, respectively, and  $p_1$ ,  $p_2$ , and  $p_3$  refer to the chosen gene's values in T0, T10, and T20, respectively. In our case,  $p_1$ ,  $p_2$ , and  $p_3$  equal 69, 74, and 96, respectively, and  $w_1$ ,  $w_2$ , and  $w_3$  equal 40.5, 76.1, and 71.2, respectively. Of course one can devise many alternative learning functions, but the one I have used is the simplest, and it works fine. What is a bit tricky is the  $\alpha$  parameter in Eq. (6.10), which is called the learning rate.

What should be the value of  $\alpha$ ? An  $\alpha$  equal to 0 implies  $w'_i = w_i$ , which means that the winning node does not learn anything from the chosen gene and will never change. An  $\alpha$  equal to 1 implies  $w'_i = p_i$ , which means that the winning node cannot retain any prior knowledge and will always mirror the knowledge of the chosen gene that has the smallest distance to it. This simple reasoning leads us to conclude that the  $\alpha$  value should be greater than 0 but smaller than 1. In Canada, conservatives typically have an  $\alpha$  that is too small, and liberals typically have an  $\alpha$  that is too large.

If  $\alpha$  is close to 0, then the learning process is slow, and SOM takes a long time to converge. If  $\alpha$  is close to 1, the values of the winning node will change back and forth too fast, and the node values may fail to stabilize. As a compromise,  $\alpha$  will initially be large (close to 1) but will diminish with each iteration. This essentially implies that, in a society that is quite broken, it is better to have liberals in charge. In a society that is working pretty well, it is better to have conservatives in charge.

For illustration, let's start with  $\alpha = 0.5$ . This leads to

$$\begin{aligned} w_1' &= 40.5(1 - 0.5) + 69 \times 0.5 = 54.7 \\ w_2' &= 75.0 \\ w_3' &= 83.6 \end{aligned} \quad (6.11)$$

These three values will replace the three original values (i.e., 40.5, 76.1, and 71.2, respectively) of node(3,1). The update of the winning node is now complete. If the training data have many genes with gene expression values similar to those of Gene 4, then this node(3,1) will move quickly to their centroid.

The next step is to modify the neighbors of the winning node as, the neighboring nodes will also learn from the chosen gene, just as relatives and friends of a government official will often benefit from his/her position. Updating the values of neighbors is governed by the following learning function:

$$w_i' = w_i(1 - \alpha_n) + p_i \alpha_n \quad (6.12)$$

where  $\alpha_n$  is the learning rate for the neighbors. Again one can use one of many possible alternative learning functions, but we will just use Eq. (6.12) to keep things simple. In practice, the learning function of neighbors depends on how we define neighbors, and  $\alpha_n$  will be larger for immediate neighbors than for remote neighbors. For obvious reasons, they should also be smaller than  $\alpha$ . If we designate  $\alpha_{n1}$  as the learning rate for the immediate neighbors (i.e., nodes in physical contact with the winning nodes),  $\alpha_{n2}$  as the learning rate for the neighbors of the immediate neighbors, and so on, then one simple way of choosing  $\alpha_n$  values is to set  $\alpha_{n1} = \alpha/2$ ,  $\alpha_{n2} = \alpha_{n1}/2$ , and so on.

For our illustration, we will just define each node to have a maximum of four neighbors, i.e., the one to its left, the one to its right, the one above it, and the one below it. Thus defined, we need only one  $\alpha_n$  value which we will set to  $\alpha/2$ . Our winning node is in a corner and consequently has only two neighbors, with one above it and one to its right. The updated values of SOM according to Eq. (6.12) are shown in Table 6.4.

Now that we are done with Gene 4, we again repeat the process by randomly choosing a gene, computing the Euclidean distance to find the winning node, and carrying out the updating of the values. We perform this with decreasing  $\alpha$  and  $\alpha_n$  values with each cycle of iteration until  $\alpha$  equals a preset  $\alpha_{\min} > 0$  (we do not want to decrease  $\alpha$  to zero because SOM will stop learning when  $\alpha = 0$ ).

**Table 6.4** SOM after updating the winning node, i.e., node(3,1), and its two neighbors, node(2,1) and node(3,2)

	1			2			3		
	T0	T10	T20	T0	T10	T20	T0	T10	T20
1	2.2	11.5	33.4	41.9	27.6	0.8	6.0	63.8	51.2
2	38.7	41.1	59.3	28.7	51.3	9.0	61.6	38.2	17.9
3	54.7	75.0	83.6	77.1	89.5	41.4	76.2	40.9	23.9

**Table 6.5** Trained SOM

	1			2			3		
	T0	T10	T20	T0	T10	T20	T0	T10	T20
1	80.7	77.6	73.3	74	77.9	67.3	69.2	79.5	72.2
2	84.4	81.9	82.1	82.5	81.2	81.4	78.5	77.6	78.2
3	89.6	85.4	91	88.1	82.9	91.7	79.9	79.1	89.4

How should we decrease  $\alpha$  and  $\alpha_n$  with each cycle of updating? There is no optimal way of decreasing  $\alpha$ . In my SOM implementation in AMIADA (Xia and Xie 2001a), I used the following equation:

$$Q = \left(1 - \frac{1}{N_G}\right) \quad (6.13)$$

where  $N_G$  is the number of genes. In our case,  $N_G = 15$  and  $Q = 0.933$ , i.e.,  $\alpha$  will be multiplied by  $Q$  after each cycle of iteration.

Continuing the learning process will eventually lead to convergence, i.e., when the values in the nodes do not change any more or the change is smaller than a prefixed small value in two consecutive cycles of iteration. The result of the learning process (Table 6.5) is a trained SOM ready for classification.

Different nodes (Table 6.5) have different properties reflecting our data structure, e.g., we know that there are highly expressed genes and lowly expressed in the data set. The lower left nodes, i.e., node(3,1) and node(3,2), have relatively high expression values at all three time points, and node(1,2) and node(1,3) have relatively low expression at all three time points. Node(1,1) has its expression decreasing with time, and node(3,3) has its expression increased at time T20 relative to the two previous time points.

The trained SOM can now be used for classification. During the classification stage, the node values do not change. A new gene with its expression values at T0, T10, and T20 can be assigned to a node by computing the Euclidean distance between this node and each of the nine nodes. The node with the smallest Euclidian distance will have the new gene assigned to it. The node to which Gene  $i$  is assigned is called the host node of Gene  $i$ .

Before we use the trained SOM to do the classification of new genes, it is crucial to check how well the SOM fits the training data. This is typically done by first assigning the genes in the training data to the nine nodes and then computing the Euclidian distance (or its square) between each gene and its host node (Table 6.6).

We instantly notice a few genes that fit poorly into their respective host nodes (Table 6.6). For example, gene 10, with three gene expression values being 67, 96, and 55 at T0, T10, and T20, respectively, is classified to node(1,2) with its node values being 74, 77.9, and 67.3. Although both Gene 10 and its host node have the largest value at T10 and the smallest value at T20, the classification is deemed poor because of the large Euclidean distance ( $= 23$ , Table 6.6). The classification of Gene 7 to node(3,2) is similar, albeit with a smaller distance ( $= 17.43$ ). Such large

**Table 6.6** Classification of the 15 genes to the nine nodes

Gene	T0	T10	T20	Row	Col	<i>d</i>
1	93	76	87	3	2	9.69
2	80	81	85	2	2	4.41
3	89	88	85	3	1	6.54
4	69	74	96	3	3	13.77
5	95	89	93	3	1	6.80
6	65	96	76	1	3	17.43
7	87	85	96	3	2	4.90
8	78	89	88	3	3	10.17
9	87	80	97	3	2	6.12
10	67	96	55	1	2	23.00
11	91	90	95	3	1	6.30
12	76	72	67	1	2	6.19
13	79	78	94	3	3	4.84
14	96	76	78	2	1	13.63
15	66	64	63	1	2	16.54

Row and Col indicate the coordinates of the host nodes. The last column is the Euclidian distance between each gene and its host node

Euclidean distances suggest that the SOM does not provide good fit to the data, and we should rerun SOM with a larger (more accommodating) grid.

Application of SOM to the yeast gene expression data (Cho et al. 1998) generates many co-expressed genes. Most are similar to those recovered by the UPGMA methods, but there are also different ones. These co-expressed genes are candidates for further study to check if they are co-regulated, i.e., whether they share similar regulatory sequences that are activated by the same transcription factors. The Gibbs sampler covered in a previous chapter is one of the key data mining tools used to identify such regulatory sequences.

It is important to recognize the fact that, although each input data point in our example is a vector of three numbers, SOM is not limited to data points represented as a vector of numbers. It can be applied to any data for which we can (1) define a distance between a data point and a node and (2) update the value of the winning nodes and neighboring nodes in response to the input. For example, the input data points can be 20-base sequences flanking the 5'-splicing site in eukaryotic protein-coding genes, and the node can be represented by a sequence profile (illustrated in previous chapter on sequence alignment). The distance between the input sequence and the node sequence profile can just be a function of an alignment score or a position weight matrix score (Xia 2017d). The updating of the nodes can be done easily by revising the sequence profile by adding the input sequence. SOM for de novo motif discovery was illustrated in detail in Xia (2017d). Its advantage over Gibbs sampler in de novo motif discovery is that Gibbs sampler can report only one motif, and SOM can report multiple heterogeneous motifs.

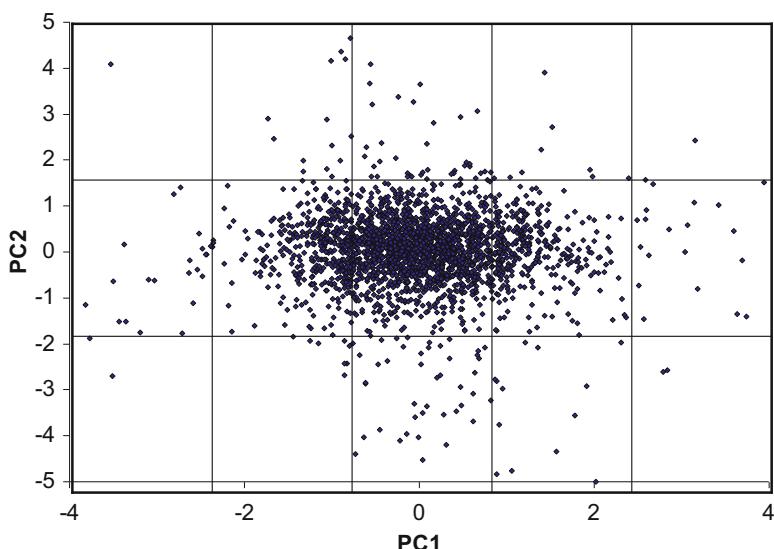
### 3.2.2 Variations of the Basic SOM Algorithm

I will briefly mention two variations of the basic SOM algorithm as presented in the previous section. The first is to replace the random initialization step by using the first two principal component scores from principal component analysis (PCA) of the matrix containing the gene expression data. PCA is a dimension reduction technique to project high dimensional data into a low dimensional space, and in this sense it serves as a good initial approximate to SOM.

To visualize the application of PCA to SOM initialization, plot the two principal component scores, and superimpose the grid of nodes onto the two-dimensional plot (Fig. 6.8). The values of each node, e.g.,  $w_1$ ,  $w_2$ , and  $w_3$  in our example, are then the averages of those points falling within that node. The node values are then updated by using the same protocol as we have already learned. This dramatically reduces the computation time needed for SOM training.

One may wonder why the grid in Fig. 6.8 has five columns but only three rows. The reason is that PC1 (the first principal component) typically accounts for far more variation in the data than PC2, and the dimension of the grid of nodes should be proportional to the variation accounted for by each principal component.

The other variation of the SOM algorithm involves the updating process. Instead of updating the winning node and its neighbors with every input data point, we simply find a host node for each data point and assign all data points to their respective host nodes without updating. Once all data points have been mapped onto the grid of nodes, we then compute the node values as the mean or median of those data points assigned to the node. This process is repeated until convergence is achieved. This variation of the SOM algorithm is often referred to as the batch mode of SOM training.



**Fig. 6.8** Using the first two principal component scores to initialize the grid of nodes

## Postscript

A researcher compiles demographic, political, economic, and educational data from many countries in the world and used clustering algorithms and self-organizing map to analyze them. Almost all affluent western countries were mapped to a few closely spaced nodes whereas all poor countries were scattered all over the place.

“All happy families are alike; each unhappy family is unhappy in its own way.” The researcher concluded his presentation with a quote from Leo Tolstoy, highlighting the sharing of democracy among the affluent western countries.

I was impressed, but then the ensuing discussion became disturbing, at least to me, when someone expressed the perhaps noble wish that “It would be so nice if all those poor countries embrace democracy and live like us.”

Spreading democracy and changing regimes have been used as a pretext for wars in recent years, often resulting in millions of homeless refugees.

Have we really developed a social system that can be grafted onto another country and spawn prosperity and happiness?

We as scientists often do our research in different ways, although we all believe in the general principle of scientific method. I surely would pay attention to how successful scientists conduct their research and imitate what they do if it benefits my own research, but I would be appalled if someone walks into my laboratory and demands that I have to do research in his or her way.

Human history has witnessed many wars that erupted because some people thought that they had gained a religion better than others. There are still fundamentalists who believe that the world will become heaven if everyone embraces their extreme views.

During the Great Cultural Revolution in China in late 1960s, young red guards heard, mistakenly, that serfdom was still practiced in Tibet and that the ruling monks maintained such serfdom by brainwashing the believers. Committing themselves to the noble cause of liberating the poor Tibetans, many red guards braved themselves against all the odds to march thousands of miles of treacherous terrains to Tibet. Many young boys and girls died along the way, taking their last breath to bid their comrades to continue their unfinished cause. Those who did reach Potala Palace immediately began to do cultural damage that is felt even today.

Plato believed that arrogance is the root cause of all misunderstanding and evil and illustrated his point brilliantly with his famous allegory of the cave. But we still live like the chained prisoners in the cave. We will not make progress unless we realize how ignorant and depraved we are.

# Chapter 7

## Hidden Markov Models and Protein Secondary Structure Prediction



### 1 Introduction

Hidden Markov models (HMMs) have been used in characterizing protein families and in deciding whether a new protein belongs to a particular protein family (Durbin 1998; Eddy 1996, 1998), in molecular phylogenetics (Felsenstein and Churchill 1996; Siepel and Haussler 2004a, b, 2005; Yang 1995), in structure-based gene finding (Baldi and Brunak 2001; Durbin 1998; Pevzner 2000), and in protein secondary structure prediction. The underlying HMM algorithm, however, is similar. In this chapter, I have chosen to illustrate the application of HMMs in predicting protein secondary structure, but you should be able to apply it elsewhere once you have finished this chapter.

Markov model is what is hidden in HMMs, so I will first give a brief introduction to Markov models sufficient for comprehending HMMs. In particular, we wish to know (1) two categories of parameters, the frequency parameters and the rate parameters in the transition probability matrix that characterize a first-order Markov model, also known as a Markov chain, (2) how to obtain the equilibrium frequencies, and (3) how to calculate the likelihood of a given sequence of events that follow a Markov model. These are the minimal requirement for a reasonable understanding of HMMs.

HMMs are illustrated numerically because most readers, just like me, cannot see the beauty of equations until they are rendered to numbers. You will learn the essential elements in an HMM, how to train an HMM and interpret a trained HMM, how to reconstruct the most probable path of hidden states by using the Viterbi algorithm, how to compute the likelihood of a particular sequence of events by using the forward algorithm, and how to estimate parameters in an HMM. If you are a programmer, you will be able to implement the algorithms associated with HMM once you have finished this chapter.

The simplest HMM would have two hidden states, e.g., my office has an “occupied by me” state and a “not occupied by me” state. You cannot push the

door open to check if I am inside, but you can see my office window to note if the light in my office is on. The light is motion-sensitive. If I am in, then the light is typically on, but it may be off if I take a nap or sit almost motionlessly in front of my computer. If I am not in, the light is typically off but occasionally may become on when a janitor comes in to clean the office. From your perspective, the two states are hidden, but each state has distinctive emission probabilities (i.e., the light is mostly on in the “occupied by me” state and mostly off in the “not occupied by me” state). So you can have a pretty good guess on which state my office is in. If the light is equally likely to be on or off between the two state (e.g., if janitors come in often or I take naps often), then your prediction will be poor. Alternatively, if I get in and out of office every 20 min, then the office light will always be on (because the light does not go off until about 30 min after I leave my office), and you again will not be able to use my office light to predict which state my office is in.

The example above shows that HMM could be quite problematic even in a very simple scenario. Consequently, many trained HMMs in published papers could be meaningless. By analogy, a linear regression model with the slope being close to 0 and nonsignificant is useless for prediction, although there is nothing preventing you from fitting the model and making meaningless predictions. Similarly, a trained HMM may also be uninformative and useless for prediction. You will know whether a HMM is useful or not by inspecting the transition probabilities and emission probabilities. A publication that claims to have used HMM for prediction but does not report the transition and emission probabilities is equivalent to one that claims to have used a linear regression model to describe data but report neither the values of the slope and intercept nor the sample size and the statistical significance associated with them. HMM for predicting protein secondary structure, or any observation that can be expressed as a sequence of string characters, is implemented in DAMBE (Xia 2013, 2017d) with a sample sequence.

## 2 Markov Models

A Markov model is typically used to model the dynamic change of a random variable over time. For example, the pitch of music from your stereo reaching your ears over time  $t_1, t_2, \dots, t_k$  can be represented as  $X_{t1}, X_{t2}, \dots, X_{tk}$ . The  $X_{ti}$  values constitute a realization of the random variable  $X$  over time. However, Markov models are equally applicable over a one-dimensional space. For example, along the linear DNA, nucleotides change over site 1, 2, ...,  $i$  and can be represented as  $X_1, X_2, \dots, X_i$ .

Suppose a DNA sequence of length  $L$ , with each site coded only as purine (R) and pyrimidine (Y). The proportions of R and Y are designated as  $p_R$  and  $p_Y$ , respectively. What is the probability that a nucleotide at site  $i$ , designated as  $X_i$ , is R (or Y)? Without any further information, our prediction of a site being occupied by R (or Y) is simply  $p_R$  (or  $p_Y$ ). For a fictitious sequence of alternating purine and pyrimidine triplets,

$$S = RRRYYYYRRRYYY \dots,$$

$p_R = p_Y = 0.5$ . For the partially sequenced human chromosome 22 with 10 contigs (accession NT\_028395, NT\_011519, NT\_011520, NT\_011521, NT\_011523, NT\_011525, NT\_019197, NT\_113818, NT\_011526, NT\_113961, dated 02-MAR-2006) with 35,017,877 base pairs (bp),  $p_R$  and  $p_Y$  are nearly equal, being 0.50047 and 0.49953, respectively. Without any further information, our prediction of a site being occupied by R and Y in human chromosome 22 is 0.50047 and 0.49953, respectively.

Now we consider a slightly more realistic case with the minimal site dependence, with the probability of  $X_{i+1}$  being either R or Y depending only on whether  $X_i$  is R or Y. We use  $P_{RR}$  and  $P_{RY}$  to designate the conditional probabilities of R and Y, respectively, at site  $i+1$  given R at site  $i$  and  $P_{YR}$  and  $P_{YY}$  to designate the conditional probabilities of R and Y at site  $i+1$  given Y at site  $i$ . Some authors (Higgs and Attwood 2005, p. 234; Weir 1990, p. 238) give the estimate of the conditional probabilities in the following form:

$$P_{RR} = \frac{N_{RR}}{N_R}; P_{RY} = \frac{N_{RY}}{N_R}; P_{YR} = \frac{N_{YR}}{N_Y}; P_{YY} = \frac{N_{YY}}{N_Y} \quad (7.1)$$

where  $N_X$  and  $N_{XK}$  are the number of X and XK doublets, respectively, in the sequence, with X and K being R or Y. What they have in mind is a very long sequence. Suppose we have only a short sequence  $S'$  equal to the first 12 nucleotides of S, i.e.,

$$S' = RRRYYYYRRRYYY.$$

Now Eq. (7.1) will result in weird estimates. For example,  $P_{YY}$  and  $P_{YR}$  would be 2/3 and 1/6, respectively, based on  $S'$ . These  $P_{YY}$  and  $P_{YR}$  values have two problems, one major and one minor. The major one is that they are probabilistically incorrect because the two do not even sum up to 1 as they should. The minor one is that given the regular alternating patterns of purine triplets and pyrimidine triplets in  $S'$ , our intuition suggests that  $P_{YR}$  should be equal to  $P_{RY}$ . The estimated  $P_{YR}$  ( $= 1/6$ ) being half of  $P_{RY}$  ( $= 2/6$ ) according to Eq. (7.1) makes us feel uncomfortable. A mathematically more consistent alternative for  $P_{YY}$  and  $P_{YR}$  (which is also a maximum likelihood estimate) is

$$P_{XK} = \frac{N_{XK}}{\sum_K N_{XK}}, \text{ e.g., } P_{YR} = \frac{N_{YR}}{N_{YR} + N_{YY}} = \frac{1}{5} \quad (7.2)$$

where the denominator is the summation of all dinucleotides starting with nucleotide X. This yields  $P_{YY} = 4/5$  and  $P_{YR} = 1/5$ . The two now do sum up correctly to 1, eliminating the major problem we mentioned above. Moreover, the difference between  $P_{YR}$  and  $P_{RY}$  is now somewhat smaller, alleviating the minor problem. A still more reasonable estimate of  $P_{XK}$ , assuming that the last nucleotide is followed by R with a probability  $p_R$  and by Y with a probability  $p_R$ , is

$$P_{YR} = \frac{N_{YR} + p_R}{1 + \sum_{K'} N_{YK'}}; \quad P_{YY} = \frac{N_{YY} + p_Y}{1 + \sum_{K'} N_{YK'}} \quad (7.3)$$

For sequence S',  $P_{RR} = 2/3$ ,  $P_{RY} = 1/3$ ,  $P_{YR} = 1.5/6$ , and  $P_{YY} = 4.5/6$  according to Eq. (7.3). Thus, the major problem is again solved, and the minor problem is further alleviated although  $P_{YR}$  and  $P_{RY}$  are still not the same as from our intuition.

To simplify the presentation of Markov models, let us just assume that we have a long sequence of alternative purine triplets and pyrimidine triplets, so that indeed  $P_{RR} = P_{YY} = 2/3$  and  $P_{YR} = P_{RY} = 1/3$ . These four values, arranged in a  $2 \times 2$  matrix (Table 7.1), constitute what is called the transition probability matrix for a first-order Markov model which is also known as a Markov chain. The four corresponding elements for human chromosome 22 are also included in Table 7.1. We note that a purine is more likely to be followed by a purine than by a pyrimidine and that a pyrimidine is more likely followed by a pyrimidine than by a purine and that this pattern, obvious enough for S, is also true for human chromosome 22 (Table 7.1). This helps us to predict the probability of  $X_{i+1}$  given  $X_i$ . Take S, for example, without information on  $X_i$ ; our prediction of  $X_{i+1}$  being R is  $p_R$  ( $= 0.5$ ). In contrast, with  $X_i = R$ , our prediction of  $X_{i+1}$  being R is  $P_{RR}$  ( $= 2/3$ ).

Other than the two transition probability matrices illustrated in Table 7.1, we need to know the frequency parameters, typically arranged in the form of a vector designated by  $p_i$ , to characterize the Markov chain. For our example with only two states (R and Y), the two frequency parameters are the probabilities of R and Y at site  $i$ , designated by  $p_{R,i}$  and  $p_{Y,i}$ , respectively. In other words, the vector  $p$  contains two elements,  $p_{R,i}$  and  $p_{Y,i}$ .

We now learn how to obtain equilibrium frequencies of R and Y given the transition probability matrix. Suppose  $X_{i-1} = R$ , so  $p_{R,i-1} = 1$  and  $p_{Y,i-1} = 0$ , and  $p_{R,i}$  and  $p_{Y,i}$  are naturally equal to  $P_{RR}$  and  $P_{RY}$ , respectively. In matrix algebra, and using the transition probability matrix for sequence S (Table 7.1), we have

$$\begin{aligned} p_i &= p_{i-1}M = [p_{R,i-1} \ p_{Y,i-1}] \begin{bmatrix} P_{RR} & P_{RY} \\ P_{YR} & P_{YY} \end{bmatrix} \\ &= [p_{R,i-1}P_{RR} + p_{Y,i-1}P_{YR} \ p_{R,i-1}P_{RY} + p_{Y,i-1}P_{YY}] \\ &= [1 \ 0] \begin{bmatrix} 2/3 & 1/3 \\ 1/3 & 2/3 \end{bmatrix} = [2/3 \ 1/3] \end{aligned} \quad (7.4)$$

**Table 7.1** Two transition probability matrices, one for S and one for human chromosome 22, for the Markov chain with two states (R and Y). Note that values in each row in a transition probability matrix are constrained with a row sum of 1

	S		Human Chr22	
	R	Y	R	Y
R	2/3	1/3	0.56683	0.43317
Y	1/3	2/3	0.43398	0.56602

Note that the probability of a site being R is the summation of two probabilities: (1) the probability that site  $i-1$  is R ( $p_{R,i-1}$ ) multiplied by the probability that this R at site  $i-1$  is followed by an R at site  $i$  ( $P_{RR}$ ) and 2) the probability that site  $i-1$  is Y ( $p_{Y,i-1}$ ) multiplied by the probability that this Y at site  $i-1$  is followed by an R at site  $i$  ( $P_{YR}$ ). This gives  $p_{R,i-1}P_{RR} + p_{Y,i-1}P_{YR}$  in Eq. (7.4). Similarly, the probability of a site being Y at site  $i$  is  $p_{R,i-1}P_{RY} + p_{Y,i-1}P_{YY}$ . At site  $i+1$ ,  $i+2$ , etc., we expect

$$\begin{aligned} p_{i+1} &= p_i M = [2/3 \quad 1/3] \begin{bmatrix} 2/3 & 1/3 \\ 1/3 & 2/3 \end{bmatrix} = [0.55556 \quad 0.44444] \\ p_{i+2} &= p_{i+1} M = [0.51852 \quad 0.48148] \\ &\dots \\ p_{i+9} &= p_{i+8} M = [0.50001 \quad 0.49999] \end{aligned} \quad (7.5)$$

If we continue the multiplication, eventually the two frequencies will approach  $p_R = p_Y = 0.5$  and do not change anymore with  $i$ . These  $p_R$  and  $p_Y$  that do not change with  $i$  are then called equilibrium frequencies and typically designated as  $\pi_R$  and  $\pi_Y$ . These equilibrium frequencies can be obtained more easily by solving the equation  $p_i = p_{i-1}M$  with the constraints that

$$\begin{aligned} p_{R,i} &= p_{R,i-1} \\ p_{Y,i} &= p_{Y,i-1} \end{aligned} \quad (7.6)$$

which implies that

$$\begin{aligned} p_{R,i} &= p_{R,i-1}P_{RR} + p_{Y,i-1}P_{YR} = p_{R,i}P_{RR} + p_{Y,i}P_{YR} \\ p_{Y,i} &= p_{R,i-1}P_{RY} + p_{Y,i-1}P_{YY} = p_{R,i}P_{RY} + p_{Y,i}P_{YY} \end{aligned} \quad (7.7)$$

Solving the resulting simultaneous equations with the constraints that  $p_R + p_Y = 1$ , we have

$$p_{R,i} = \frac{P_{YR}}{P_{RY} + P_{YR}}; \quad p_{Y,i} = \frac{P_{RY}}{P_{RY} + P_{YR}} \quad (7.8)$$

which are the equilibrium frequencies. For the fictitious sequence S,  $\pi_R = p_R = 0.5$  and  $\pi_Y = p_Y = 0.5$ . For human chromosome 22,  $\pi_R = 0.50047$  and  $\pi_Y = 0.49953$ .

You may ask whether first-order Markov model is any better than the zero-order Markov model. The latter assumes complete independence of nucleotides over sites, i.e., whether site  $i$  is R has nothing to do with what nucleotides are at other sites. Because the zero-order Markov model is a special case of the first-order Markov model, the question can be addressed by what is called a likelihood ratio test. For the fictitious sequence S' (the empirical observation) with only 12 nucleotides, the likelihoods according to the zero-order and first-order Markov models are, respectively,

$$\begin{aligned} L_0 &= p_R^3 p_Y^3 p_{RY}^3 p_{YY}^3 = 0.000244141 \\ L_1 &= p_R P_{RR}^2 P_{RY} P_{YY}^2 P_{YR} P_{RR}^2 P_{RY} P_{YY}^2 = 0.000722564 \end{aligned} \quad (7.9)$$

The larger the likelihood, the better the model fits the data. However, the absolute magnitude of a likelihood value is not important; what is important is the relative magnitude measured as a ratio of two likelihood values. In our case, what is important is the ratio of  $L_1/L_0$  which can be used in a likelihood ratio test to help us decide whether the first-order Markov model is significantly better than the zero-order Markov model. To carry out a likelihood ratio test, we calculate

$$X^2 = 2[\ln(L_1) - \ln(L_0)] = 2.170122511 \quad (7.10)$$

which follows approximately the  $\chi^2$ -distribution with one degree of freedom. The degree of freedom associated with a likelihood ratio test is the difference in the number of parameters between the two models. The zero-order Markov model has only two frequencies,  $p_R$  and  $p_Y$ . Because  $p_R = 1 - p_Y$ , and is therefore not free when  $p_Y$  is known, there is only one free parameter for the zero-order Markov model. For the first-order Markov chain, we have four elements in the transition probability matrix  $M$ . However, because the two values in each row have to add up to 1,  $M$  has only two free parameters. Thus, the degree of freedom associated with the likelihood ratio test involving the first-order and zero-order Markov models is  $(2-1) = 1$ . In general, for a  $k$ th-order Markov model with  $n$  categories of symbols (in our case we have only two categories of symbols, i.e., R and Y, so  $n = 2$ ), the number of parameters is

$$N_{\text{param}} = (n - 1)n^k \quad (7.11)$$

For example, if we have a nucleotide sequence with four symbols (A, C, G, and T), then the number of parameters for the first-order model would be 12. Similarly, the number of parameters for a third-order Markov model for an amino acid sequence ( $n = 20$ ) is  $19 \times 20^3 = 15,200$ . Given that a protein is generally only one thousand amino acid long, such a model may be meaningless.

The likelihood ratio test does not reject the zero-order Markov model in favor of the first-order Markov model for the given  $S'$  ( $p = 0.14$ ), although the site dependence, with purine triplets and pyrimidine triplets following each other, is quite obvious. The reason for the failure to reject the zero-order Markov model is the short length of  $S'$ . If  $S'$  is three times longer, i.e., if it is made of the first 36 nucleotides of  $S$ , then  $X^2 = 4.888507099$ , and we can reject the zero-order Markov model with  $p = 0.027036057$ .

$P_R$  and  $P_Y$  from  $S$  happen to be the same as  $\pi_R$  and  $\pi_Y$ . Usually  $P_R$  and  $P_Y$  from a short sequence will not be the same as  $\pi_R$  and  $\pi_Y$ . If one has already obtained  $\pi_R$  and  $\pi_Y$  from longer sequences, one should replace  $P_R$  with  $\pi_R$  in Eq. (7.9).

### 3 Hidden Markov Models

### 3.1 The Essential Elements in a Hidden Markov Model

The classic illustration of HMM involves a dishonest casino dealer stealthily switching between a fair die (F) and a loaded die (L). His switching pattern is characterized by the  $2 \times 2$  transition probability matrix (Table 7.2) detailed in the previous section. Now we introduce another kind of probability called emission probability. Tossing the fair die leads to numbers 1–6 appearing with an equal probability of  $1/6$ , but tossing the loaded die always results in number 6. These probabilities of observing different numbers conditional on the hidden state are termed emission probabilities (Table 7.2). For example,

because the casino dealer will not let others know when he switches, the two states (F and L) are hidden, hence the term “hidden Markov model.” What one can observe is just a series of numbers from which we can derive emission probabilities typically represented by  $e_S(x)$  which means the probability of observing  $x$  given hidden state S, e.g.,

$$e_F(1) = 1/6; \quad e_L(1) = 0; \quad e_L(6) = 1 \quad (7.12)$$

Observed symbols: 1435266634521334  
Hidden states: FFFFFL~~L~~FFF~~F~~FFF

To summarize, an HMM has three essential elements, the transition probability matrix ( $P_{ij}$  in the first three columns in Table 7.2) and state frequencies ( $\pi_F$  and  $\pi_L$  in our two-dice example), the number of emitted symbols (6 in our example) and the associated emission probability matrix (the last six columns in Table 7.2), and the observed sequence of events ( $O = "14,352\dots"$ ). The main task of HMM is to infer the number of states (e.g., how many dice is the casino dealer using?), estimate  $p_{ij}$  values, reconstruct the hidden state transition sequence (e.g., “FFFFFLLLFFFFFF” in our example), evaluate the probability of the sequence, and compute probability of the observed sequence given the reconstructed state transition sequence, i.e.,

$$\Pr(1435266634521334 | \text{FFFFFLLLFFFFFF})$$

**Table 7.2** HMM involving a dishonest casino dealer, with the transition probability matrix on the left and the emission probabilities on the right (the last six columns)

These tasks are obviously not trivial. Whether the completion of these tasks will lead to meaningful results depends mainly on two things. First, there should be site dependence, and the state transition is ideally rare, i.e.,  $P_{ii} \gg P_{ij}$ . In our example of the dishonest casino dealer,  $P_{ii} \gg P_{ij}$  means that the crook will keep using the same die for an extended number of throws so that we will see a string of 6 s followed by a string of random values between 1 and 6 and again followed by a string of 6 s. Crooks are never this dumb. Second, the emission probabilities have to be quite different between or among states. If the loaded die is only slightly loaded, with 6 appearing with only a slightly larger probability than the fair die, then we again have little chance of making a meaningful inference. For this reason, HMM attempts are aborted (wisely) when the training data yield small  $P_{ii}$  and large  $P_{ij}$  values and little difference among the vector of emission probabilities because the resulting inference/model will have little predictive power.

Suppose we use HMM for base calling in sequencing (the process of converting the four traces from the automatic sequencer to nucleotide sequences). In this case, the traces are the observed signals, and the bases are the hidden states. However, knowing one nucleotide at site  $i$  typically tells us little of what nucleotide will be in site  $i + 1$  (i.e., there is little site dependence). Fortunately, the association between a base and a particular peak in the trace is strong, i.e., a given hidden state has a strong and characteristic peak. In other words, emission probability matrix is highly informative, and base calling can generally be quite successful with this emission probability matrix only. In contrast, if nucleotide A tends to follow each other frequently, then we have state dependence which will contribute to improve base calling.

HMMs are generally associated with three objectives. The first is to estimate the parameters of HMM, e.g., the elements in the transition probability matrix and in the emission probability matrix, based on an observed sequence of events with known hidden states. This is also called HMM training. The second is to reconstruct the most probable path of hidden states by using the trained HMM and the Viterbi algorithm explained in detail later. The last is to obtain the probability of the observed sequence of events, also explained in detail later.

HMM has been used in predicting protein secondary structural elements such as  $\alpha$ -helix,  $\beta$ -sheet, and turns. These structural elements are hidden states that emit different amino acids with different frequencies. For example, some amino acids are frequently found in helices but rarely in sheets or turns (Xia and Xie 2002).

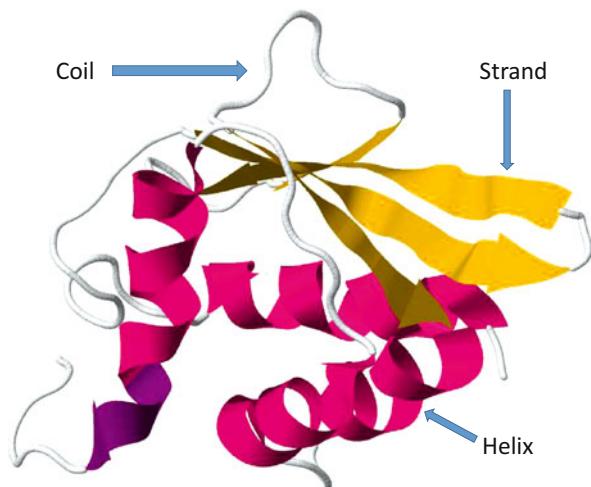
In what follows, I will use empirical data from the HIV1 proteins to illustrate the application of HMM in secondary structure prediction. In short, we will learn to (1) train a HMM, i.e., obtain the transition probability matrix  $M$  and emission probability matrix  $E$ , from a training sequence with known hidden states, (2) reconstruct the sequence of hidden states, known as the Viterbi path or the most probable path, by using the Viterbi algorithm (Viterbi 1967), and (3) compute the probability of the observed sequence of symbols. These are the three essential tasks closely associated with HMM (Rabiner 1989). The last two tasks require the transition probability matrix and the emission probability matrix from task 1.

### 3.2 Training HMM for Predicting Protein Secondary Structure

Three protein secondary structure elements were recognized by Critical Assessment of protein Structure Prediction (CASP) standard (Moult et al. 1999): helix, strand, and coil, illustrated with a mosquito protein from Protein Data Bank (PDB) with accession 2NBM (Fig. 7.1). Helices and strands are relatively easy to recognize, and all the rest that cannot be easily classified into helices or strands are lumped as coils. Early prediction of protein secondary structure is based on different amino acid frequencies in these three structural elements (Chou and Fasman 1978a, b; Fasman and Chou 1974).

Eight different secondary structure elements are currently recognized. The structural elements and their corresponding short-hand notations are listed in Table 7.3.

**Fig. 7.1** 3D structure of ligand free sterol carrier protein 2 like 2 from *Aedes aegypti* (Protein Data Bank accession 2NBM), illustrating the three basic elements of protein secondary structure: helix, strand, and coil. 3D rendering by software JSMOL



**Table 7.3** Notation of protein secondary structure elements following Protein Data Bank (PDB) and Critical Assessment of protein Structure Prediction (CASP)

Element	PDB	CASP
$\alpha$ -helix	H	H
$\beta_{10}$ helix	G	H
$\pi$ -helix	I	H
$\beta$ -strand	E	E
Bridge	B	E
$\beta$ -turn	T	C
Bend	S	C
Coil	C	C

The latter reduces eight different secondary structure elements to three

```

ST CCCCCCCEEEEEECCCCCCCCCCCCCCCCHHHHHHHHHHHHHHHHHHCCCEECCCC
RT PISPIETPVKLKPGMDGPVKQWPLTEEKIKALVEICTEMEKEGKISKIGPE

ST CCCCCCCEEEEEECCCCCCCCHHHHHHHHHHHHHHHHHHHEEECCCCCCCCCCCCCE
RT NPYNTPVFAIKKKDSTKWRKLVDLFRELNKSTQDFWEVQLGIPHAGLKKKSV

ST EEEEECCEEECCCCCCCCCEECCECCCCCCCCCCCCCECCCCCCCCCCCCHHH
RT TVLDVGDAYFSVPLDEDFRKYTAFTIIPSINNETPGIRYQYNVLPQGWKGSPA1

ST HHHHHHHHHHHHHHHCCCCEEEEEECCCCCCCCCCCCHHHHHHHHHHHHHHHHHHHHHH
RT FQSSMTKILEPFRKQNPDIVIYQYMDDLYVGSDELIGQHSTKIEELRQHLLRW

ST CCCCCCCCCCCCCCCCCCCCCCEECCCCCCEECCECCCCCCCCHHHHHHHHHHHHCC
RT GLTPDPDKHQKEPPFLWMGYELHPDKWTVQPIVLPEKDSWTVNDIQKLVGKLN

ST HHHHHCCCCHHHHHHHHCCCCCCCCCCCCHHHHHHHHHHHHHHHHHHHHCCCCCEE
RT WASQIYPGIKVRQLCKLLRGTKALTEVIPLTEEALELAENREILKEPVHGTVY

ST ECCCHHHHHHHHHHHHHCCCCCEEEEECCCCCCCCCCCCCCCCHHHHHHHHHHHHHHHH
RT YDPSKDYLIAEIQKQGQGQWTYQIYQEFPFKNLKTGKYARMRGAHTNDVKQLTEA

ST HHHHCCCCCEECCCCCCCCCCCCHHHHHHHHHHHHHHCCCCCCCCCCCCCCCCCCC
RT VQKITTESIVIWGKTPKFKLPIQKETWETWWTEYWQATWIPEWEFVNTPPLVK

ST EEEEECCCCCCCCCCCC
RT LWYQLEKEPIVGAETF

```

**Fig. 7.2** Example of a training sequence (RT) with known hidden states (ST) for protein secondary structure prediction. The hidden states are coil (C), strand (E), and helix (H). RT is a real HIV1 reverse transcriptase. ST is partially fictitious but treated here as real

The ss.txt file that can be downloaded from Protein Data Bank (<http://www.rcsb.org/pdb/files/ss.txt>) contains annotated secondary structure elements for proteins with 3D structures determined. This file typically serves as the training data set for programs aiming to predict protein secondary structure.

Suppose we are going to use a training sequence (Fig. 7.2) to train an HMM for predicting protein secondary structure defined to be either coil (C), strand (E), or helix (H). The sequence labeled as RT is a protein sequence of HIV1 reverse transcriptase, and the sequence labeled as ST is the partially fictitious sequence of known “hidden” states. In real HMM applications, we would need far more training data than just one sequence. If you want to train a Martian to recognize human males, presenting him with just a little baby boy is enough. For a proper training of HMM to predict protein secondary structure, you should download something like the ss.txt file mentioned above and use it as the training data instead of the RT sequence in Fig. 7.2.

Let  $N_{ij}$  be the number of transitions from state  $i$  to state  $j$ , which can be easily counted by moving along the ST sequence (Fig. 7.2). The maximum likelihood estimate of  $P_{ij}$ , i.e., the transition probability from state  $i$  to state  $j$ , is

**Table 7.4** Transition probability matrix estimated from the training data in Fig. 7.2

	C	E	H
C	0.88210	0.06987	0.04803
E	0.26154	0.73846	0.00000
H	0.06896	0.00690	0.92414

$$P_{ij} = \frac{N_{ij}}{\sum_k N_{ik}} \quad (7.13)$$

The  $P_{ij}$  values from data in Fig. 7.2, with relatively large  $P_{ii}$  values on the diagonal (Table 7.4), are typical of training data where each secondary structure element is made of a series of consecutive amino acids so that a state, be it C, E, or H, tends to stay in the same state for several consecutive amino acids before changing into another state (Fig. 7.2). If this is not the case, then HMM will typically be uninformative. Note that each row sum should be 1.

The three structure elements (hidden states) in the training sequence have empirical frequencies of 0.5227, 0.1477, and 0.3295 for states C, E, and H, respectively. These may be taken to approximate equilibrium frequencies  $\pi_C$ ,  $\pi_E$ , and  $\pi_H$ . If our data are from a comprehensive compilation of many accurately determined protein secondary structures, we may also assume the transition probabilities in Table 7.4 as accurate and obtain the equilibrium frequencies by solving

$$[\pi_C \ \pi_E \ \pi_H] \begin{bmatrix} P_{CC} & P_{CE} & P_{CH} \\ P_{EC} & P_{EE} & P_{EH} \\ P_{HC} & P_{HE} & P_{HH} \end{bmatrix} = [\pi_C \ \pi_E \ \pi_H] \quad (7.14)$$

with the constraint that  $\pi_C + \pi_E + \pi_H = 1$  and that each row of the transition probability matrix sums up to 1. This yields  $\pi_C = 0.5216$ ,  $\pi_E = 0.1481$ , and  $\pi_H = 0.3303$  which are very similar to the empirical frequencies.

For the emission probabilities, letting  $E_k(x_j)$  be the number of amino acid  $x_j$  ( $j = 1, 2, \dots, 20$  corresponding to the 20 amino acids) emitted from state  $k$ , the emission probability  $e_k(x_j)$  is

$$e_k(x_j) = \frac{E_k(x_j)}{\sum_{j=1}^{20} E_k(x_j)} \quad (7.15)$$

Note that the denominator is the sum of all amino acid in state  $k$ , and the numerator is the number of amino acid  $x_j$  in state  $k$ . So  $e_k(x_j)$  is simply the fraction of amino acid  $x_j$  in state  $k$ .

The  $e_k(x_j)$  values estimated from the training data in Fig. 7.2 reveal different amino acid frequency distribution among the three secondary structure features (Table 7.5). For example, amino acids proline (P) and glycine (G) are found frequently in coils but rarely in strands or helices. This should be obvious given

**Table 7.5** Emission probabilities estimated from the training data in Fig. 7.2

AA	C	E	H
A	0.0262	0.03077	0.05517
C	0	0	0.01379
D	0.05677	0.01538	0.03448
E	0.07424	0.03077	0.13793
F	0.02183	0.04615	0.02759
G	0.09607	0	0.01379
H	0.02183	0	0.01379
I	0.04803	0.13846	0.08276
K	0.11354	0.07692	0.11724
L	0.06987	0.07692	0.11034
M	0.0131	0.01538	0.01379
N	0.0393	0	0.02759
P	0.13974	0.01538	0.01379
Q	0.0393	0.07692	0.08276
R	0.02183	0	0.06207
S	0.0393	0.03077	0.02069
T	0.08297	0.06154	0.06207
V	0.0393	0.18462	0.04828
W	0.03057	0.04615	0.05517
Y	0.0262	0.15385	0.0069

The column heading “AA” stands for amino acid, and C, E, and H stands for coil, strand, and helix, respectively

the properties of proline and glycine. Proline introduces a bend in protein structure due to the cyclic binding of its three-carbon side chain to the nitrogen of the backbone. Glycine is the smallest amino acid, and only a small amino acid allows a sharp turn. While lysine (K) is frequent in state C, it is also frequent in state H (Table 7.5). Note that it is the relative distribution of an amino acid among the three states that are important. In this context, cysteine (C), although overall rare, is a good predictor because it occurs only in state H (although this could just be an artifact of limited training data). It is long known that different amino acids occur at different frequencies in different structural elements. Glu and Ala are good  $\alpha$ -helix formers, whereas some others such as Gly and Pro tend to disrupt the  $\alpha$ -helix structure. Similarly, Ile and Val are good, whereas Glu and Pro are poor  $\beta$ -sheet formers (Xia and Xie 2002).

Two other patterns in Table 7.5 are worth highlighting. Tyrosine (Y), valine (V), and isoleucine (I) are frequently found in strands but not in the other two structures, and leucine (L), glutamate (E), and cysteine (C) are frequently found in helices but rare elsewhere (Table 7.5 and Fig. 7.2). These differences in emission probabilities are encouraging. The larger the difference in emission probabilities among the three structural elements, the better the HMM will perform in predicting secondary structures. One could reach a crude prediction by just using emission probabilities,

i.e., one assigns state C upon seeing a stretch of amino acids P and G; state E upon seeing a stretch of amino acids T, V, and I; and state H upon seeing a stretch of L, E, and C. If the three columns of frequencies in Table 7.5 are similar, then the predictive power of the HMM will rest entirely on the information in the transition probability matrix.

### 3.3 The Viterbi Algorithm

Now that we have gone through the training process of obtaining the transition probability matrix (Table 7.4) and the emission probabilities (Table 7.5), we are ready to proceed with the prediction of protein secondary structure of unknown proteins with the Viterbi algorithm illustrated in this section. Because there are overlapping characters between the secondary structure notation (C, E and H) and the one-letter codes of amino acids, I will add the full notation in parenthesis to avoid confusion.

Suppose we have the following amino acid sequence:

$T = YVYVEEEEEVEEEEEEPGP$

How do we predict its secondary structure (or, in HMM parlance, decode the sequence of hidden states)? Of course, we can use only the emission probabilities (Table 7.5). Given the association of L (leucine) and E (glutamate) with helix (H), P (proline), and G (glycine) with coil (C) and Y (tyrosine) and V (valine) and I (isoleucine) with strand (E), we can write state E corresponding to Y and V, state H corresponding to E, and state C corresponding to P and G. This generates the secondary structure prediction known as the naïve path/prediction of hidden states (Naïve):

123456789012345678901  
 $T = YVYVEEEEEVEEEEEEPGP$   
 Naïve = EEEEHHHHHEHHHHHHCCCC

This seems to make secondary structure prediction really easy, and the approach has actually been taken before (Chou and Fasman 1978a, b; Fasman and Chou 1974). However, incorporating information on site dependence can improve the prediction. For example,  $P_{EH}$  in the transition probability matrix is 0.00000 (which again could be an artifact with our extremely limited training data), implying an extremely small probability of state transition from E (strand) to H (helix). Our naïve prediction above with an H (helix) at position 5 following an E (strand) at position 4 therefore represents an extremely unlikely event. Another example is at position 11 with  $T_{11} = V$  (valine). Our prediction of  $\text{Naïve}_{11} = E$  (strand) implies a transition of secondary structure from H (helix) at  $\text{Naïve}_{10}$  to E (strand) at  $\text{Naïve}_{11}$  and then

back from E (strand) at  $\text{Naïve}_{11}$  to H (helix) at  $\text{Naïve}_{12}$ . The transition probability matrix shows us that  $P_{\text{HE}}$  and  $P_{\text{EH}}$  are both very small. So  $T_{11}$  is very unlikely to be in state E (strand).

Let us see if the Viterbi algorithm in HMM can do better than the naïve prediction. The Viterbi algorithm (Viterbi 1967) is a dynamic programming algorithm. It generates a  $V$  matrix (Table 7.6) where each value incorporates the information from both the transition probability matrix and the emission probability matrix. The computation involves filling a Viterbi matrix and a backtrack matrix (referred to as  $V$  and  $B$ , respectively, hereafter), both of dimension  $n \times L$  where  $n$  is the number of states and  $L$  is the length of the sequence. In our example,  $n = 3$  and  $L = 21$ . Because the page is not wide enough for 21 columns,  $V$  and  $B$  are shown in  $L \times n$  tables (Table 7.6) with individual values in the  $V$  matrix designated as  $V_k(i)$  where  $k$  is the state C, E, or H and  $i$  is the site index. For example,  $V_C(2) = -7.54809$  (Table 7.6).

Recall that the  $V$  matrix is for decoding the sequence of hidden states. Each  $V_k(i)$  in Table 7.6 is the logarithm of probability of the most probable path up to site  $i$  in state  $k$  (it is the logarithm of the probability because the probabilities will become

**Table 7.6** The  $V$  and  $B$  matrices from running the Viterbi algorithm using the transition probability matrix in Table 7.4 and emission probability matrix in Table 7.5. The values in the  $V$  matrix are their natural logarithms. The values in the  $B$  matrix are pointers, with 0, 1, and 2, respectively, as pointers to C, E, or H state in the previous site. The last column (HS) is the reconstructed hidden states

V Matrix			B Matrix			
C	E	H	C(0)	E (1)	H(2)	HS
Y	-4.74057	-2.97041	-6.07535			E
V	-7.54809	-4.96308	-9.18506	1	1	2
Y	-9.94622	-7.13807	-14.24069	1	1	2
V	-11.71574	-9.13074	-16.01287	1	1	0
E	-13.07242	-12.91516	-16.73257	1	1	0
E	-15.79838	-16.69959	-18.08925	0	1	0
E	-18.52435	-20.48402	-20.14914	0	1	2
E	-21.25031	-24.26844	-22.20904	0	1	2
E	-23.97627	-27.39268	-24.26893	0	0	2
E	-26.70223	-30.11864	-26.32883	0	0	2
V	-30.06419	-31.05285	-29.43855	0	0	2
E	-32.79015	-34.83727	-31.49844	0	1	2
E	-35.51611	-38.62170	-33.55834	0	1	2
E	-38.24207	-41.65849	-35.61823	0	0	2
E	-40.89289	-44.07621	-37.67813	2	2	2
E	-42.95279	-46.13610	-39.73802	2	2	2
E	-45.01268	-48.19600	-41.79792	2	2	2
P	-46.44005	-50.94904	-46.16040	2	2	C
G	-48.90819	-237.91316	-50.52288	0	0	C
P	-51.00163	-55.74371	-54.88536	0	0	C
G	-53.46976	-242.47474	-58.32104	0	0	C

quite small). For example,  $V_H(5)$ , being  $-16.73257$ , is the logarithm of the probability of the most probable path up to amino acid E at site 5 in state H.

The output in Table 7.6 is from program DAMBE (Xia 2001; Xia and Xie 2001b) which implements the Viterbi algorithm as well as the forward algorithm (detailed later) for computing the probability of the observed sequence of events given the transition probability matrix and the emission probability matrix. Here we use manual computation so that you know how to get the output yourself.

The first amino acid  $Y$  has no site-dependent information because we do not know what goes before it. So the three values in the first row of  $V$  (Table 7.6) are calculated differently from the rest of the sites where both the transition probabilities and emission probabilities are used to calculate  $V$  values. The probability that the first  $Y$  is in state C, E, or H, without any other information, is  $1/3$ , i.e., the three states are equally likely. But we do have information on emission probability. If amino acid  $Y$  occurs very frequently in state E but rarely in C or H, then the likelihood of this first  $Y$  in state E should be greater than in C or H. Therefore, the  $V$  values for the first site are the emission probability for each state multiplied by  $1/3$ :

$$\begin{aligned} V_C(1) &= e_C(Y)/3 = 0.0262/3 = 0.0087336245 \\ V_E(1) &= e_E(Y)/3 = 0.15385/3 = 0.0512820513 \\ V_H(1) &= e_H(Y)/3 = 0.0069/3 = 0.0022988506 \end{aligned} \quad (7.16)$$

These  $V_k(1)$  values mean that, if we are asked which state (C, E, H) an amino acid  $Y$  is in, without giving us any other information, then our best guess is that it is in C, E, or H with probabilities  $V_C(1)$ ,  $V_E(1)$ , and  $V_H(1)$ , respectively. The three values in the first row in the  $V$  matrix in Table 7.6 are the logarithm of these three values. The reason for taking the logarithm is that the  $V_k(i)$  values will become quite small.

In more concise and more general mathematical terms, Eq. (7.16) is written as

$$V_k(1) = e_k(X_1)/n \quad (7.17)$$

where  $k = C, E$ , or  $H$  and  $n$  is the number of hidden states ( $=3$  in our example).

An alternative but more reasonable initialization of  $V_k(1)$  values is

$$V_k(1) = e_k(X_1)\pi_k \quad (7.18)$$

where  $\pi_k$  is the equilibrium frequency of state  $k$ . The empirical frequencies, as we have computed before, are  $\pi_C = 0.5216$ ,  $\pi_E = 0.1481$ , and  $\pi_H = 0.3303$ . Note that Eq. (7.17) assumes that the three states occur equally frequently.

The rest of the rows in  $V$  are filled with a more intimidating equation with both emission probability and transition probability:

$$V_l(i) = e_l(X_i)\max_k[V_k(i-1)P_{kl}] \quad (7.19)$$

where subscript  $l$  refers to the hidden state at site  $i$ ,  $k$  is the hidden state at site  $i-1$ , and  $P_{kl}$  is the transition probability from hidden state  $k$  to hidden state  $l$ .

Many intimidating mathematical expressions are in fact quite simple, and Eq. (7.19) is no exception, especially when it is rendered into numbers. For example, according to the equation, the second row of the  $V$  matrix is filled with the following three values:

$$\begin{aligned}
 V_C(2) &= e_C(V)\max[V_C(1)P_{CC}, V_E(1)P_{EC}, V_H(1)P_{HC}] \\
 &= 0.0393 \times \max(0.008733625 \times 0.8821, \\
 &\quad 0.051282051 \times 0.26154, 0.002298851 \times 0.06897) \\
 &= 0.0393 \times 0.013412308 = 0.000527104 \\
 V_E(2) &= e_E(V)\max[V_C(1)P_{CE}, V_E(1)P_{EE}, V_H(1)P_{HE}] \\
 &= 0.006991512 \\
 V_H(2) &= e_H(V)\max[V_C(1)P_{CH}, V_E(1)P_{EH}, V_H(1)P_{HH}] \\
 &= 0.000102569
 \end{aligned} \tag{7.20}$$

where  $V$  in  $e_C(V)$ ,  $e_E(V)$ , and  $e_H(V)$  is the amino acid  $V$  at the second site of the amino acid sequence. Don't confuse it with the  $V$  matrix. The three  $V_k(2)$  values are the probability of the most probable path of hidden states from the first two amino acids ending with state  $k$ . For example,  $V_E(2)$  is the probability of the most probable path for the first two amino acids ending with state E. Once you have learned the  $B$  matrix, you will know that  $V_E(2)$  is the probability of path EE. Any path that ends with E but not EE will have a smaller probability.

$V_l(i)$  values for  $i = 3, 4, \dots, 21$  are computed in exactly the same way. Each one depends on the  $V_l(i-1)$  values. This is typical of all dynamic programming algorithms.

Equation (7.19) and its numerical rendition in Eq. (7.20) are also very easy to understand and to remember. Take  $V_C(2)$  in Eq. (7.20), for example. The max function is to find, given the reconstructed hidden state at the second site being C, which of the three possible hidden states in the previous site is most likely to transit into C. In our case, it is the hidden state E that is most likely to transit into C, with its value of 0.013412308 being the maximum of the three. If the reconstructed hidden state is C, how likely is it to emit an amino acid  $V$ ? This is the emission probability  $e_C(V) = 0.0393$  (Table 7.5). So  $V_C(2)$  is the probability that the hidden state at the site 2 is C multiplied by the probability that the hidden state C emits an amino acid  $V$ . With this, the computation of  $V_E(2)$  and  $V_H(2)$ , as well as the rest of  $V_l(i)$ , is obvious. I encourage you to perform the computation by hand. Just in case you wish to have some values to check your computation, here are the three  $V_l(3)$  values:

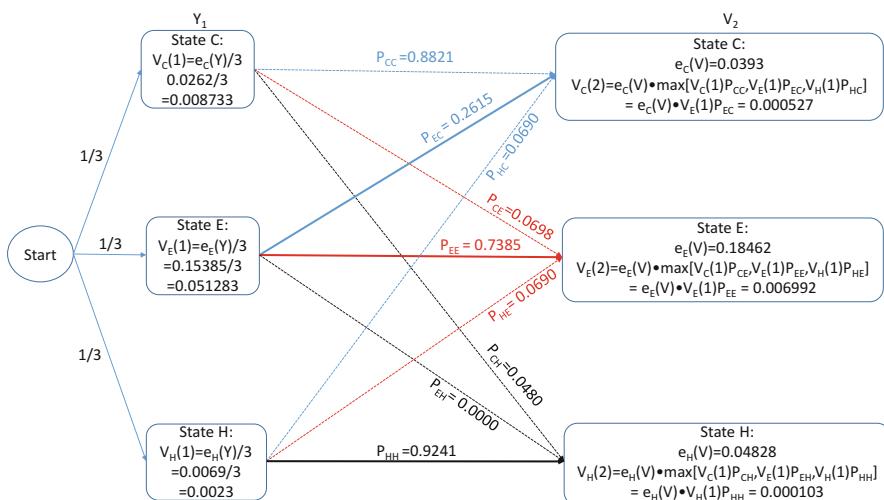
$$\begin{aligned}
 V_C(3) &= 0.0000479085 \\
 V_E(3) &= 0.0007942838 \\
 V_H(3) &= 0.0000006537
 \end{aligned} \tag{7.21}$$

Recall that  $V_k(i)$  is the probability of the most probable path of hidden states ( $i$ , e., C, E, H) from site 1 to site  $i$  ending state  $k$ . For example,  $V_E(3)$  is the probability of the most probable path of hidden states for the first three amino acids. Once we have learned the  $B$  matrix, we will see that  $V_E(3)$  is the probability of path EEE. Any path that ends with state E but is not EEE will not be as good, i.e., will have a smaller probability.

You might have noticed that the three  $V_l(2)$  values in Eq. (7.20) are substantially smaller than the three  $V_l(1)$  values in Eq. (7.16) and the three  $V_l(3)$  values in Eq. (7.21) are substantially smaller than the three  $V_l(2)$  values in Eq. (7.20), and so on. The Viterbi algorithm involves the multiplication of many small probabilities, and it takes only a short sequence for a computer to generate an arithmetic underflow error (and probably weird results long before this). In order to avoid arithmetic underflow problems in computation, any practical implementation of the Viterbi algorithm would have log-transformed the equations so that we do additions instead of multiplications. If you take the natural logarithm of  $V_l(1)$ ,  $V_l(2)$ , and  $V_l(3)$  values in Eqs. (7.16), (7.20), and (7.21), you should get the values in the first three rows in the  $V$  matrix in Table 7.6.

The computation for the first two rows of the  $V$  matrix (for the first two amino acids Y and V, represented as  $Y_1$  and  $V_2$ ) is also illustrated in Fig. 7.3 which seems to help some of the students who are used to visual learning. At the  $Y_1$  site, we do not know what state goes before it, so states C, E, and H each gets 1/3 for a sum of 1 (Fig. 7.3). An alternative is to assign the probability of the three states according to their empirical frequencies, i.e.,  $\pi_C = 0.5216$ ,  $\pi_E = 0.1481$ , and  $\pi_H = 0.3303$ .  $V_k(1)$  is the corresponding emission probability,  $e_k(Y)$ , multiplied by 1/3 (Fig. 7.3) or  $\pi_k$ . For amino acid at site  $i$  starting from  $i = 2$ ,  $V_k(i)$  is computed according to Eq. (7.19) and illustrated with amino acid  $V_2$  (Fig. 7.3).

How likely is  $V_2$  (Fig. 7.3) in state C? This likelihood is measured by  $V_C(2)$ . If V occurs frequently in state C, i.e., a large  $e_C(V)$ , if  $P_{kC}$  is large, and if  $V_k(1)$  is large,



**Fig. 7.3** Illustration of computing the  $V$  matrix for the first two amino acids Y and V labeled as  $Y_1$  and  $V_2$ .  $V_k(1)$  (where k stands for state C, E, or H) depends only on the emission probability  $e_k(Y)$  weighted by the probability of which of the three states (C, E, H)  $Y_1$  may be in (1/3). An alternative is to replace 1/3 by the equilibrium frequencies which we have computed to be  $\pi_C = 0.5216$ ,  $\pi_E = 0.1481$ , and  $\pi_H = 0.3303$ .  $V_k(2)$  values are computed with  $V_k(1)$ , emission probability  $e_k(V)$  and transition probabilities  $P_{ki}$ .

then  $V_C(2)$  will be large. However,  $e_C(V)$  is only 0.0393 (the average is  $1/20 = 0.05$ ). While  $P_{CC}$  is large ( $= 0.8821$ ),  $V_C(1)$  is quite small ( $= 0.008733$ ), so that the product of the two is also small. This means that, if  $V_2$  is in state C, it is not because the previous amino acid is in state C. In contrast, while  $P_{EC} (= 0.2615)$  is smaller than  $P_{CC}$ ,  $V_E(1)$  is quite large (Fig. 7.3), so the product of the two is also relatively large (the largest of the three). Thus, if  $V_2$  is in state C, it is largely due to site 1 amino acid being in state E. Hence we draw a thick line connecting state E at site 1 and state C at site 2 (Fig. 7.3).

The three lines of the same color in Fig. 7.3 represent the three  $V_k(i-1)P_{Ik}$  values, with the thick line representing the path that is the maximum of the three values. The thick lines also represent backtrack directions. If  $V_2$  is in state C, then  $Y_1$  should be in state E (blue thick line guiding us to backtrack from state C site 2 to E at site 1, coded by 1 in the  $B$  matrix in Table 7.6); if  $V_2$  is in state E, then  $Y_1$  should also be in state E (red thick line for backtracking from E at site 2 to E at site 1); if  $V_2$  is in state H, then  $Y_1$  should be in state H (black thick line for backtracking from H at site 2 to H at site 1.).

We need more explanation about the  $B$  matrix (the backtrack matrix), which should be filled concurrently with the  $V$  matrix. Just as the backtrack matrix in the dynamic programming algorithm for sequence alignment is for the actual reconstruction of aligned sequences, the backtrack matrix in the Viterbi algorithm is for reconstructing the most probable hidden path, also known as the Viterbi path. In our example, the Viterbi path is the reconstructed secondary structure of the peptide T.

Each cell in  $B$  in Table 7.6 is, in fact, a pointer (or arrow) to a cell in the previous site. The first row of  $B$  in Table 7.6 is empty for the obvious reason that the first site has no previous site to point to (Table 7.6). The first three values in the second site are 1, 1, and 2 (Table 7.6). We will first explain how we get these values and what they mean and finally learn how to reconstruct the Viterbi path by following the pointers in the  $B$  matrix.

The entries in the  $B$  matrix are obtained from the max function in Eqs. (7.19) and (7.20). Take  $V_C(2)$  in Eq. (7.20), for example. The three values within the max function are  $V_C(1)P_{CC}$ ,  $V_E(1)P_{EC}$ , and  $V_H(1)P_{HC}$ , with the maximum being  $V_E(1)P_{EC}$ . So we should have an arrow pointing from C at the second site to E at the first site. Because it is not convenient to store graphic arrows in digital computers, we coded the three hidden states C, E, and H to 0, 1, and 2, respectively. To represent an arrow from C at the second site to E at the first site, we simply put a number 1 under column C ( $B$  matrix in Table 7.6) at the second site. Because we only have three hidden states, a cell in  $B$  will contain either a 0 (for C) or 1 (for E) or 2 (for H).

Yes, I could have put real arrows to the  $B$  matrix in Table 7.6, but I was afraid of spoiling future programmers if everything is made too visual. Sometimes it is better to see things with our mind's eye. However, there is nothing preventing you from replacing the numbers by arrows in the  $B$  matrix in Table 7.6.

The second value in the second row of  $B$  is 1, meaning an arrow pointing from state E at the second site to state E at the first site (i.e., a vertical arrow pointing up). This is obtained in exactly the same way from  $V_E(2)$  in Eq. (7.20). Among the three values within the matrix function,  $V_C(1)P_{CE}$ ,  $V_E(1)P_{EE}$ , and  $V_H(1)P_{HE}$ ,  $V_E(1)P_{EE}$  is

the largest. So we again have a value of 1 representing an arrow from state E at the second site to state E at the first site.

For  $V_H(2)$  in Eq. (7.20), the last of the three values within the max function is the largest, yielding a value of 2 representing an arrow from state H in the second site to H in the first site. That is, you have another vertical arrow pointing upward.

Once the  $V$  and  $B$  matrices are complete, we can backtrack along the  $B$  matrix to reconstruct the hidden states. We first look at the very last row in  $V$  (Table 7.6) and find the largest value, which is  $-53.46976$  under column C. This means that the last amino acid (i.e., G) should be in state C (i.e., in a coil). This brings us to the value in the last row of  $B$  under column C in Table 7.6. This value is 0 (representing C). Recall that values in the  $B$  matrix are pointers. A value of 0 in a cell in the  $B$  matrix means that the second last amino acid (i.e., P) is also in a coil. Similarly, we know the third and fourth last amino acids (G and P, respectively) are also in a coil. We record these reconstructed hidden states in the last column in Table 7.6, i.e., a stretch of four C's from the bottom.

Now we are at the cell containing a value of 2 (for H or helix) under column C (the fourth row from the bottom in the  $B$  matrix). This means that the next amino acid (i.e., amino acid E at site 17) is no longer in state C but in state H. So now we move to the value of 2 (the fifth from the bottom) under column H in Table 7.6. This cell, together with the 11 cells in proceeding sites, contains a value of 2. This means a stretch of amino acids in state H all the way to the sixth amino acid (i.e., amino acid E). We again record this stretch of H's in the last column (HS) in Table 7.6.

The last value of 2 in this stretch of 2's is in the seventh row of the  $B$  matrix in Table 7.6, corresponding to amino acid E. The cell right above this 2 is 0. This means that the amino acid at fifth site (i.e., E) is no longer in state H, but in state C. This brings us back to column C corresponding to the fifth amino acid (amino acid E), where we find a value of 1. This value of 1 means that the previous amino acid (the fourth one, amino acid V) is in structure E. This brings us to column E at the fourth amino acid. This cell has a value of 1 and a stretch of 1's above it all the way to the top. This means that the four amino acids at sites 1–4 are all in structure E. If you find yourself confused, then just replace those numbers in the  $B$  matrix in Table 7.6 by real arrows and then follow the arrows to get the reconstructed secondary structure.

The final reconstructed sequences are shown below (Viterbi), together with the naïve reconstruction (Naïve) we have derived by using only information in the emission probability matrix:

```

123456789012345678901
T = YVYVEEEEEVEEEEEEPGGP
Viterbi = EEECHHHHHHHHHHHHHCCCC
Naive = EEEEHHHHHHHEHHHHHHCCCC

```

Two hidden states, at site 5 and 11, were reconstructed differently between the naïve path and the Viterbi path. A hidden state of H at site 5 in the naïve reconstruction implies a transition from hidden state E directly into hidden state H, which

has an extremely small probability  $P_{EH} = 0.00000$  (Table 7.4). The Viterbi path shows first a transition from E to C and then from C to H. This is a more likely path than the naïve reconstruction because  $P_{EC}$  and  $P_{CH}$  are both much larger than 0.00000 (Table 7.4). Another difference is at site 11. The naïve reconstruction of state E at this site implies a transition of secondary structure from H to E, and then back from E to H. The transition probability matrix shows us that  $P_{HE}$  and  $P_{EH}$  are both very small. So a hidden state of E at this site is very unlikely. The transition probability matrix shows a large  $P_{HH}$  value (Table 7.4). The reconstructed hidden state H at this site in the Viterbi path implies that the helix structure is more likely to continue across this site instead of switching to some other secondary structures.

We have now covered two of the three major tasks associated with HMM, i.e., train a HMM and reconstruct the sequence of hidden states. We now deal with the last task, i.e., computing the probability of the observed sequence of symbols by the forward algorithm (Rabiner 1989).

### 3.4 Forward Algorithm

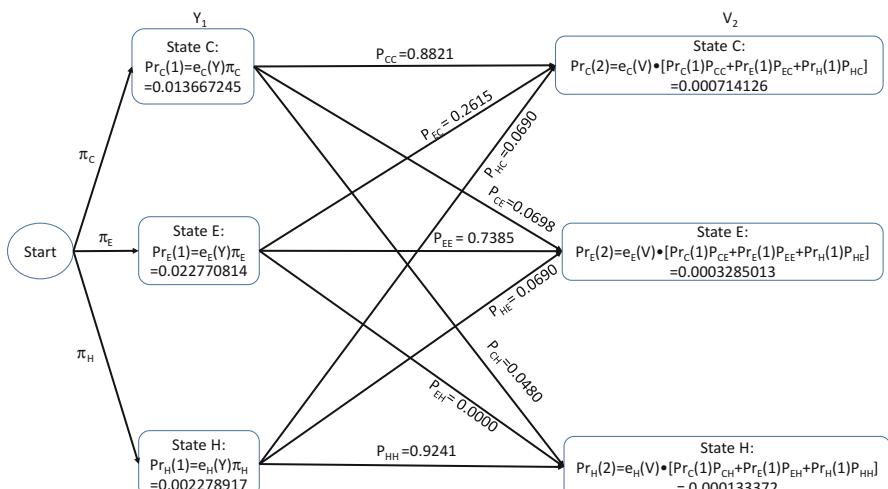
The forward algorithm is for computing the probability of the observed sequence of events. Given an observed amino acid sequence T, the probability is designated  $P(T)$ . This probability is useful in different context. For example, HMM is often used to decide whether an amino acid sequence belongs to a protein family and  $P(T)$  plays an important role in making such a decision. In such cases, the HMM is derived from a set of aligned protein sequences known to belong to a particular protein family. We classify a new sequence T into the protein family when  $P(T)$  is large. In the case of protein secondary structure prediction, the  $P(T)$  value indicates the reliability of the inferred secondary structure of protein T, assuming that our training data is comprehensive, i.e., including a large number of diverse proteins with accurately determined secondary structure.

The forward algorithm involves dynamically completing a  $L \times n$  matrix, where  $L$  is sequence length and  $n$  is the number of hidden states. We will actually illustrate the computation of two matrices, one is forward probability matrix (or  $P$  matrix) and the other is typically referred to as the  $F$  matrix ( $F$  for forward) which is simply a re-scaled  $P$  matrix. They are two representations of the same thing, and both serve the same purpose of giving us  $P(T)$ . If there is no computer overflow or underflow, then the  $P$  matrix is sufficient and is conceptually easier to understand. However, computer overflow/underflow is frequent in computing  $P(T)$ , so we need to learn how F matrix is obtained.

Table 7.7 is the  $P$  matrix for our data, with each element designated as  $Pr_k(i)$  where  $k$  is either C, E, or H and  $i$  is the site index.  $Pr_k(i)$  is the probability of having observed 1 to  $i$  amino acid with the  $i$ th amino acid in state  $k$ . For example,  $Pr_E(3)$  is the probability of observing the first three amino acids with amino acid Y at site 3 being in state E. We now learn the details of filling in values in Table 7.7, with Fig. 7.4 as a summary of the calculation below.

**Table 7.7** The P matrix from the forward algorithm. The first column is the amino acid sequence (AA). The summation of the last row is the probability of observing the sequence in the inferred secondary structure, being  $3.00171 \cdot 10^{-23}$

AA	$Pr_C$	$Pr_E$	H
Y	0.013667245	0.022780814	0.002278917
V	0.000714126	0.003285013	0.000133372
Y	3.92552E-05	0.000381035	1.08712E-06
V	5.28027E-06	5.2456E-05	1.39533E-07
E	1.36503E-06	1.20331E-06	5.27664E-08
E	1.13026E-07	3.02879E-08	1.5769E-08
E	8.07059E-09	9.34558E-10	2.75879E-09
E	5.60792E-10	3.91721E-11	4.0512E-10
E	3.95596E-11	2.18174E-12	5.53544E-11
E	2.91644E-12	1.46376E-13	7.31791E-12
V	1.22443E-13	6.68985E-14	3.33269E-13
E	1.10238E-14	1.85409E-15	4.32919E-14
E	9.79587E-16	7.5021E-17	5.5913E-15
E	9.42363E-17	4.99778E-18	7.19194E-16
E	9.95081E-18	4.68854E-19	9.22975E-17
E	1.13335E-18	5.16427E-20	1.18308E-17
E	1.358E-19	6.12186E-21	1.51554E-18
P	3.15696E-20	3.76292E-22	1.94038E-20
G	2.81334E-21	0	2.68189E-22
P	3.4937E-22	3.05168E-24	5.28114E-24
G	2.97184E-23	0	2.98702E-25



**Fig. 7.4** Computing the P matrix for the first two amino acids Y and V (labeled  $Y_1$  and  $V_2$ ). For  $Y_1$ , we do not know the secondary structure state before it, so we assign either 1/3 or the corresponding equilibrium frequencies ( $\pi$ ) as shown. P matrix values for the first amino acid site then depends on the emission probability  $e_k(Y)$  and  $\pi$  values. For subsequent sites,  $Pr_k(i)$  values are computed from  $Pr_k(i-1)$  and emission and transition probabilities.

The three values in the first row, corresponding to the first amino acid (i.e., Y), are filled as

$$\begin{aligned} \text{Pr}_C(1) &= e_C(Y)\pi_C = 0.013667245 \\ \text{Pr}_E(1) &= e_E(Y)\pi_E = 0.022770814 \\ \text{Pr}_H(1) &= e_H(Y)\pi_H = 0.002278917 \end{aligned} \quad (7.22)$$

where  $\pi_C$ ,  $\pi_E$ , and  $\pi_H$  are equilibrium frequencies that have been computed previously as  $\pi_C = 0.5216$ ,  $\pi_E = 0.1481$ , and  $\pi_H = 0.3303$  and  $e_C(Y)$ ,  $e_E(Y)$ , and  $e_H(Y)$  are emission probabilities being 0.0262, 0.15385, and 0.0069, respectively (Table 7.5). Sometimes these equilibrium frequencies are replaced simply by  $1/n$  (where  $n$  is the number of hidden states and is 3 in our case).

The  $\text{Pr}_l(i)$  values with  $i > 1$  are computed according to the following equation:

$$\text{Pr}_l(i) = e_l(X_i) \sum_k [\text{Pr}_k(i-1)P_{kl}] \quad (7.23)$$

where  $X_i$  is the amino acid at position  $i$  and the subscript  $l$  represents states C, E, or H (I use both  $l$  and  $k$  to represent states). You may have noticed the similarity between Eq. (7.23) and Eq. (7.19); changing the max function in Eq. (7.19) to  $\sum$  will give you Eq. (7.23). Again, this seemingly intimidating equation is quite easy to understand and simple to compute, especially when it is rendered to numbers. Take the second amino acid site (amino acid V), for example. The three  $\text{Pr}_l(2)$  values, shown in the second row in Table 7.7, are obtained as

$$\begin{aligned} \text{Pr}_C(2) &= e_C(V)[\text{Pr}_C(1)P_{CC} + \text{Pr}_E(1)P_{EC} + \text{Pr}_H(1)P_{HC}] \\ &= 0.0393(0.013667245 \times 0.8821 + 0.022780814 \times 0.26154 \\ &\quad + 0.002278917 \times 0.06897) \\ &= 0.000714126 \\ \text{Pr}_E(2) &= e_E(V)[\text{Pr}_C(1)P_{CE} + \text{Pr}_E(1)P_{EE} + \text{Pr}_H(1)P_{HE}] = 0.003285013 \\ \text{Pr}_H(2) &= e_H(V)[\text{Pr}_C(1)P_{CH} + \text{Pr}_E(1)P_{EH} + \text{Pr}_H(1)P_{HH}] = 0.000133372 \end{aligned} \quad (7.24)$$

We continue until we obtain the three values for the last amino acid at site 21 (i.e., the terminating G). The summation of these last three values is the probability of the observed amino acid sequence (T) given the transition probability matrix and the emission probability matrix, i.e.,

$$\begin{aligned} P(T) &= \sum_{l=1}^n \text{Pr}_l(L) = 2.97184 \cdot 10^{-23} + 0 + 2.98702 \cdot 10^{-25} \\ &= 3.00171 \times 10^{-23} \\ \ln [P(T)] &= -51.86027353 \end{aligned} \quad (7.25)$$

where  $l$  is the index of hidden states (E, C, or H in our example),  $n$  is the number of hidden states ( $= 3$  in our example), and  $L$  is the sequence length of T. However, Eq. (7.25) is almost never used in actual computation. You may have already noticed the decreasing values from top to bottom in Table 7.7. The forward algorithm involves the multiplication of many small probabilities, so that product can very quickly approach zero. If we have longer sequences, we will simply have three 0 values at the last row (or way before the last row). This is why we are not yet done, but we have to get familiar with the F matrix which avoids the underflow/overflow problem in computation (underflow means that a non-zero number has become so small that it becomes the same as zero when represented by a computer).

You might suggest using the logarithm as we did before with the Viterbi algorithm, and you are encouraged to try it after you have finished this section, but I will not. Instead, I will introduce the F matrix and how to compute it by a scaling method with a scaling factor  $s(i)$ . This is the standard methods for computing  $P(T)$  in HMM and is also implemented in DAMBE (Xia 2001, 2013; Xia and Xie 2001b). By convention, the scaling factor is

$$s(i) = \frac{1}{\sum_l F_l(i)} \quad (7.26)$$

We will need to compute  $F_l(i)$  and  $F_l'(i)$  values for each site. For site 1, the three  $F_l(1)$  values are the same as the three  $P_{rl}(1)$  values in Eq. (7.22):

$$\begin{aligned} F_C(1) &= e_C(Y)\pi_C = 0.013667245 \\ F_E(1) &= e_E(Y)\pi_E = 0.022770814 \\ F_H(1) &= e_H(Y)\pi_H = 0.002278917 \end{aligned} \quad (7.27)$$

These  $F_l(i)$  values are scaled to  $F_l'(i)$  values by

$$F'_k(i) = s(i)F_l(i) \quad (7.28)$$

For example, the three  $F_l(1)$  values in Eq. (7.27) are re-scaled to

$$\begin{aligned} s(1) &= \frac{1}{F_C(1) + F_E(1) + F_H(1)} = 25.824 \\ F'_C(1) &= F_C(1)s(1) = 0.35291279 \\ F'_E(1) &= F_E(1)s(1) = 0.588241482 \\ F'_H(1) &= F_H(1)s(1) = 0.058845729 \end{aligned} \quad (7.29)$$

Note that after scaling, the three  $F'_l(1)$  values add up to 1. These three re-scaled values are shown in the first row in Table 7.8. These  $F'_l(1)$  values are then used to compute  $F_l(2)$  values.  $F_l(i)$  and  $F_l'(i)$  with  $i > 1$  are computed by

**Table 7.8** The F matrix from the forward algorithm with the scaling method

AA	C	E	H	<i>s</i>
Y	0.35294	0.58824	0.05882	25.824
V	0.17282	0.79491	0.03226	9.371
Y	0.09317	0.90426	0.00258	9.807
V	0.09124	0.90635	0.00241	7.281
E	0.52078	0.45909	0.02013	22.082
E	0.71047	0.19040	0.09914	16.477
E	0.68601	0.07944	0.23455	13.523
E	0.55790	0.03897	0.40313	11.704
E	0.40736	0.02247	0.57018	10.351
E	0.28088	0.01410	0.70502	9.353
V	0.23427	0.12798	0.63775	19.865
E	0.19622	0.03300	0.77078	9.304
E	0.14736	0.01129	0.84135	8.452
E	0.11512	0.00610	0.87878	8.120
E	0.09685	0.00456	0.89858	7.968
E	0.08706	0.00397	0.90898	7.892
E	0.08192	0.00369	0.91439	7.853
P	0.61471	0.00733	0.37797	32.278
G	0.91292	0.00000	0.08708	16.665
P	0.97669	0.00853	0.01477	8.615
G	0.99004	0.00000	0.00996	11.917

The first column is the amino acid sequence (AA), and the last column is the scaling factor  $s(i)$  explained in the text

$$\begin{aligned}
 F_l(i) &= e_l(X_i) \sum_k F'_k(i-1) P_{kl} \\
 s(i) &= \frac{1}{\sum_l F_l(i)} \\
 F'_l(i) &= s(i) F_l(i)
 \end{aligned} \tag{7.30}$$

Applying Eq. (7.30) to site 2 with amino acid V, we have

$$\begin{aligned}
 F_C(2) &= e_C(V) [F'_C(1)P_{CC} + F'_E(1)P_{EC} + F'_H(1)P_{HC}] \\
 &= 0.0393 (0.35291279 \times 0.8821 + 0.588241482 \times 0.26154 \\
 &\quad + 0.058845729 \times 0.06897) \\
 &= 0.018440017 \\
 F_E(2) &= e_E(V) \times [F'_C(1)P_{CE} + F'_E(1)P_{EE} + F'_H(1)P_{HE}] = 0.084824925 \\
 F_H(2) &= e_H(V) \times [F'_C(1)P_{CH} + F'_E(1)P_{EH} + F'_H(1)P_{HH}] = 0.003443913
 \end{aligned} \tag{7.31}$$

$$\begin{aligned}
s(2) &= \frac{1}{\sum_i F_i(2)} = \frac{1}{0.018440017 + 0.084824925 + 0.003443913} \\
&= 9.371293444 F'_C(2) = s(2) F_C(2) = 0.018440017 \times 9.371293444 \\
&= 0.172806814 F'_E(2) = s(2) F_E(2) = 0.084824925 \times 9.371293444 \\
&= 0.794919262 F'_H(2) = s(2) F_H(2) = 0.003443913 \times 9.371293444 \\
&= 0.032273924
\end{aligned} \tag{7.32}$$

Note that  $s(1)^*s(2) = 241.9836119$ , which is the same as

$$\begin{aligned}
\frac{1}{Pr_C(2) + Pr_E(2) + Pr_H(2)} &= \frac{1}{0.000714126 + 0.003285013 + 0.000133372} \\
&= 241.9836119
\end{aligned} \tag{7.33}$$

In other words, the information in the  $P$  matrix in Table 7.7 and the  $F$  matrix in Table 7.8 are the same if there is no overflow/underflow problems typically associated with the  $P$  matrix.

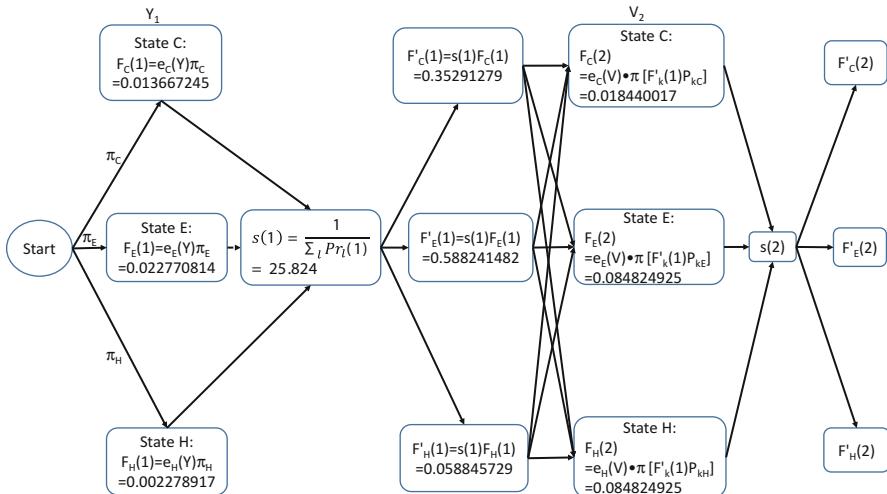
The last column in Table 7.8 is the scaling factor in Eq. (7.26) that can be used to compute  $P(T)$

$$\begin{aligned}
P(T) &= \frac{1}{L} = 3.00171 \cdot 10^{-23} \\
&\quad \prod_{i=1}^L s_i \\
\ln [P(T)] &= -\sum_{i=1}^L \ln (s_i) = -51.86027353
\end{aligned} \tag{7.34}$$

where  $L$  is the sequence length. Typically only  $\ln[P(T)]$  is computed because  $P(T)$  is effectively zero with long sequences. The calculation of the  $F$  matrix for the first two amino acids is illustrated in Fig. 7.5.

### 3.5 HMM and Gene Prediction

Given a nucleotide sequence, you may take two approaches to find if it is a gene or not. The first is to use BLAST and FASTA suite of programs to check against known “gene dictionaries” for matches. The other approach is to see if the sequence has characteristic signatures of functional protein or RNA genes. These signatures most likely will be different among classes of genes or among different evolutionary lineages. For example, microRNAs (miRNA) have their characteristic hairpins, a tRNA will typically fold into its cloverleaf structure, and processed mRNAs genes are associated with an open reading frame terminated by a start codon and ribosomal



**Fig. 7.5** Computation of the  $F$  matrix for the first two amino acids  $Y$  and  $V$  (labeled  $Y_1$  and  $V_2$ ). We first obtain  $F_k(i)$  values, generate  $s(i)$ , and then compute  $F'_k(i)$  values as a product of  $s(i)F_k(i)$

binding sites at the 5' end and a stop codon at the 3' end. Highly expressed *Escherichia coli* genes often have a strong and well-positioned Shine-Dalgarno (SD) sequence relative to the anti-SD (aSD) sequence at the 3' tail of small subunit (ssu) rRNA, as well as strongly biased codon usage. In contrast, highly expressed mammalian protein-coding genes will typically have a strong Kozak consensus and splice sites (if they contain introns). These sequence signatures are typically flanked by sequences that do not form secondary structure embedding the signature.

Many methods have been used for gene prediction in prokaryotic genomes (Borodovsky and McIninch 1993; Krogh et al. 1994; Salzberg et al. 1998) and eukaryotic genomes (Besemer and Borodovsky 2005; Burge and Karlin 1997, 1998). The application of HMM in gene prediction involves defining the structure of HMM with selected hidden states, training the defined HMM with genomic sequences of known hidden states to estimate the parameters in the transition probability matrix and the emission probability matrix, and finally applying the method to predict genes (i.e., reconstruct hidden states) by using the Viterbi and forward algorithms.

For eukaryotic genomes, exons, introns, 5'UTR, and 3'UTR are the hidden states, and they emit different triplet probabilities because exons typically have biased codon usage, especially for highly expressed genes. An exon is almost always preceded by a 5'UTR and is followed by either an intron or 3'UTR. An intron is almost always followed by an exon (although there are introns in 5'UTR and 3'UTR), so the dependence of neighboring states is obviously strong. These different emission probabilities and transition probabilities between neighboring hidden states allow us to reconstruct the hidden states of exons and introns.

It is not trivial to define the model structure of HMM in gene prediction. Current methods for gene prediction in eukaryotic genomes, e.g., GENSCAN (Burge and Karlin 1997), focus on the prediction of coding exons and exon-intron junctions. The first coding exon is the one containing the initiation codon ATG and ending at the first 5' splice junction. The last coding exon is the one containing the termination codon and extending into the 3-UTR. HMMs used in gene prediction typically involve many hidden states, e.g., GENSCAN involve 17 hidden states.

## Postscript

We see informal applications of HMM in our daily life. By making a telephone call, parents with their ears trained from many years of caring for their children can often detect hidden troubles of their children based only on the voice of the latter. In contrast, people unfamiliar to each other often find it frustratingly difficult to make sense of each other's behavior, and misunderstanding ensues. The most agonizing moment for me watching the movie "Waterloo Bridge" is when Lady Margaret Cronin failed to detect the distress experienced by Myra.

I once heard a story about the late Stephen Jay Gould giving a talk on evolution to the congregation of an All Souls Church in New York. When the guest and hosts were having lunch together, someone suggested that they should go around the table to introduce themselves. At that point Gould said something that seemed to be extraordinarily rude, something to the effect that he did not really care who the hosts were as he would never see them again. The name of Gould instantly became synonymous to rudeness among the church members.

However, soon after the incident, the members of the church learned from the newspaper that Gould had died of cancer and that his lecture in the church was in fact Gould's last public engagement – he reserved all the rest of his time to finish his 1464-page magnum opus entitled "The Structure of Evolutionary Theory." They realized that, at that moment when the seemingly rude remark erupted, Gould must have felt melancholy, as everyone would, knowing that his days were numbered, and that he was merely stating a heartbreakingly true truth that he would never see anyone around the table again.

In the HMM parlance, the remark by Gould is the emitted event from which the listeners should ideally be able to infer his hidden melancholy state of mind. An inference they did make, but it is wrong. Worse, they did not realize that it was wrong, otherwise they could have prayed more for Gould.

Stephen Jay Gould had spent all his life fighting two kinds of fundamentalists, the religious fundamentalists who believe that God is a micromanager of everything and that the Bible literally encompasses all truths in nature, and the evolutionary fundamentalists who believe that every bit of biodiversity manifests adaptation and results from natural selection. All Souls Church is perhaps the equivalent of Gould in the religious field. I would have expected Gould to have an easy time with members of this very liberal church. Yet misunderstanding still arose, and the

misunderstanding could have lasted for a long time if Gould's death had not been so well publicized.

It is truly enigmatic and paradoxical that, with the advanced computational algorithms helping us to infer the hidden, we still do not seem to make any progress in understanding each other and in understanding ourselves. The ancient Greek sage, Plato, had discovered the root cause of all misunderstanding and evil. It is called arrogance or the illusion that we are better than others. Plato illustrated his point with his famous allegory of the cave.

Imagine prisoners chained inside a cave since childhood, with their heads immobilized in such a way that their eyes were fixed on a gigantic wall. Immediately behind the prisoners was a road along which men, animals, and other things traveled. Behind the road was an enormous fire that projected the shadow of the travelers to the wall that the prisoners were facing. Also, the voice of the travelers was echoed from the wall in such a way that the prisoners believed that the words came from the shadows. Gradually, the prisoners became quite good at identifying the travelers by their shadows and voices. The shadows and the voices, as well as the interpretation of the shadows and voices by the prisoners, constituted the world of reality in the mind of the prisoners.

Now suppose a prisoner was freed and went outside the cave. Gradually he would comprehend a new reality from what he could sense. Once thus enlightened, he naturally would want to return to the cave to convey the new reality to his fellow prisoners. Unfortunately, once back in the cave, he found himself much less able to identify the travelers by their shadows than his fellow prisoners. Being thus perceived as inferior and stupid by his fellow prisoners, he failed completely in communicating the new reality to his fellow prisoners who believed to know better. The fellow prisoners were too arrogant to listen.

It is the arrogance in the mind of the prisoners that prevents them from comprehending the new reality hidden from them. It is the arrogance in the mind of the religious fundamentalists and the evolutionary fundamentalists that prevents them from understanding each other. It is the arrogance in the mind of the presidents and prime ministers that prolongs the misunderstanding among nations. Arrogance is Satan in Christianity.

I have had the privilege of meeting some of the religious and evolutionary fundamentalists. What is particularly ironical is that they all know Plato's allegory of the cave quite well, but all point to themselves as the enlightened who has seen the real world and sneer at the other party as the chained prisoners with restricted vision.

None of us is omnipresent and eternal, and our view of the world is consequently the same as that of the chained prisoners. Without grasping this painful but basic truth, we will misinterpret what we see or hear, either with HMM or not.

# Chapter 8

## Bioinformatics and Translation Initiation



### 1 Introduction

The objective of this chapter is not on the molecular details of translation but instead on the application of bioinformatic methods to solve biological problems in translation, especially those tools that have already been introduced in previous chapters such as RNA-Seq data analysis in Chaps. 4 and 5. I will illustrate a few practical research problems in the field of translation and show how they can be addressed bioinformatically. The main point, in the middle of the book, is to bring up the idea that there would be little bioinformatics without molecular biology.

Translation is one of the three essential processes in all forms of life, the other two being genome replication and transcription. Proteins are the workhorses in living cells, and their production is typically the bottleneck of biosynthesis. Signal transduction typically takes seconds, transcription minutes, and accumulation of proteins hours or days. Translation efficiency is related not only to the fitness of an organism but also to the profit of pharmaceutical industry where many of their products are protein in nature. Because many cellular processes are controlled by the right amount of proteins, abnormal abundance of certain proteins is often associated with diseases. For this reason, many drug agents and targets are proteins or protein inhibitors (Xia 2017b).

Translation efficiency depends on the efficiency and cooperation of three sub-processes: initiation, elongation, and termination. We focus on translation initiation in this chapter. Initiation is often the limiting step (Bulmer 1991; Kudla et al. 2009; Liljenstrom and von Heijne 1987; Prabhakaran et al. 2015). Efficient initiation partly depends on how efficient for the initiation tRNA to find the start codon. How does the cellular translation machinery juxtapose the start codon against the anticodon of the initiation tRNA? Do prokaryotes and eukaryotes achieve this in different ways? How to increase translation initiation efficiency so that more proteins can be produced in biopharmaceutical industry to increase profit? How can this be done with bioinformatic approaches?

Differential transcription efficiency impacts codon usage optimization and translation elongation efficiency (Tuller et al. 2010; Xia 2015). When translation initiation rate is low, increasing translation elongation rate by optimizing codon and amino acid usage will not increase protein production. However, when translation initiation is high, then elongation becomes rate limiting, and tRNA-mediated selection will shape codon usage (Prabhakaran et al. 2015; Xia 2015). For this reason, translation initiation and elongation are two closely linked processes; studying translation elongation in isolation of translation initiation can often lead to wrong conclusions (Xia 2015).

Translation initiation is not only linked to elongation; it is linked to termination as well. In prokaryotes, the junction of many neighboring genes exhibits the configuration of UAAUG (where UAA is the stop codon of the upstream gene and AUG is the start codon of the downstream gene) and AUGA (where UGA is the stop codon of the upstream gene and AUG is the start codon of the downstream gene). Thus, some sequence features thought to be associated with start codon are in fact associated with the stop codon. In eukaryotes, the 3' end of mRNA is looped back to the 5' end to facilitate ribosome recycling, with translation factor eIF4G as a scaffold between 5'UTR and the poly(A) tail coated with poly(A)-binding proteins (PABPs). Thus, PABP can affect sequence features of both 3'UTR and 5'UTR (Xia et al. 2011).

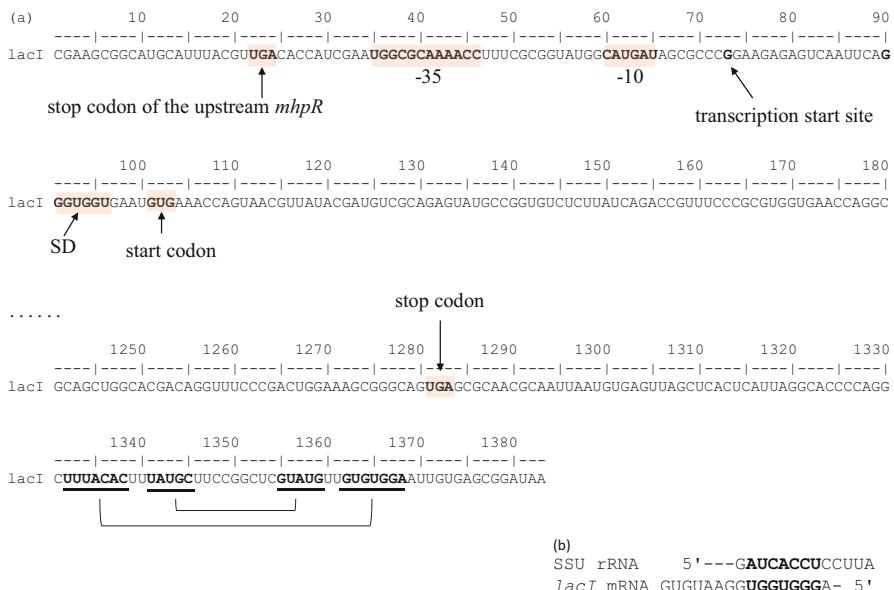
This chapter will also show how to characterize the stability of RNA secondary structure by minimum folding energy (MFE) using DAMBE (Xia 2013, 2017d) which uses the Vienna RNA library (Hofacker 2003; Hofacker et al. 2002) to measure MFE and to visually display RNA secondary structure. mRNA secondary structure especially those at the 5' end strongly affects translation efficiency. For example, a start codon as part of a translation initiation signal could be embedded in a secondary structure and become unavailable for decoding by the anticodon of the initiation tRNA. One could use a bacterial colony as a thermometer based on this observation. That is, one can incorporate into the bacterial genome a green fluorescent protein gene with a weak secondary structure that will melt at 20 °C, a yellow fluorescent protein gene with a stronger secondary structure that will melt at 37 °C, and a red fluorescent protein gene with a secondary structure that will melt at 40 °C. This would enable the bacterial colony to display green, yellow, and red fluorescence at 20 °C, 37 °C, and 40 °C, respectively. I will use MFE to study the relationship between RNA secondary structure stability and the activity of internal ribosome entry sites, but this function of DAMBE has been used to address other problems (Prabhakaran et al. 2015; Zid et al. 2009).

## 2 Translation Initiation in Bacteria

The translation initiation process in bacteria involves mRNA, tRNA<sup>fMet</sup>, and small subunit (SSU) ribosome. These constitute structural basis of translation initiation signals and their decoders. We will review them before explaining bioinformatic tools used to characterize these signals and their decoders.

### 2.1 Linear Structure of a Bacterial Gene and Its mRNA

Bacterial genes are organized in operons which may contain one or more genes transcribed in a single unit but translated separately. The *lacI* gene in *Escherichia coli* (Fig. 8.1) represents a typical linear structure of a bacterial gene. The stop codon of the upstream gene, which belongs to another operon, is UGA followed by nucleotide C. This is a rare arrangement. As we will learn in a later chapter, UGAC makes a poor stop signal in *E. coli* and is typically avoided by highly expressed *E. coli* genes (Wei et al. 2016). It is occasionally used as a regulatory switch. For example, the release factor RF2 coded by *prfB* gene in *E. coli* has an inframe UGA followed by C. When RF2 is abundant, the inframe UGA is recognized and a short nonfunctional RF2 is produced. When RF2 is rare, the inframe UGA is not



**Fig. 8.1** The linear structure of *lacI* gene in *Escherichia coli* (a), together with an illustration of SD/aSD base pairing in (b). The underlined and connected bases pair to form stems in RNA secondary structure that may facilitate the rho-independent transcription termination

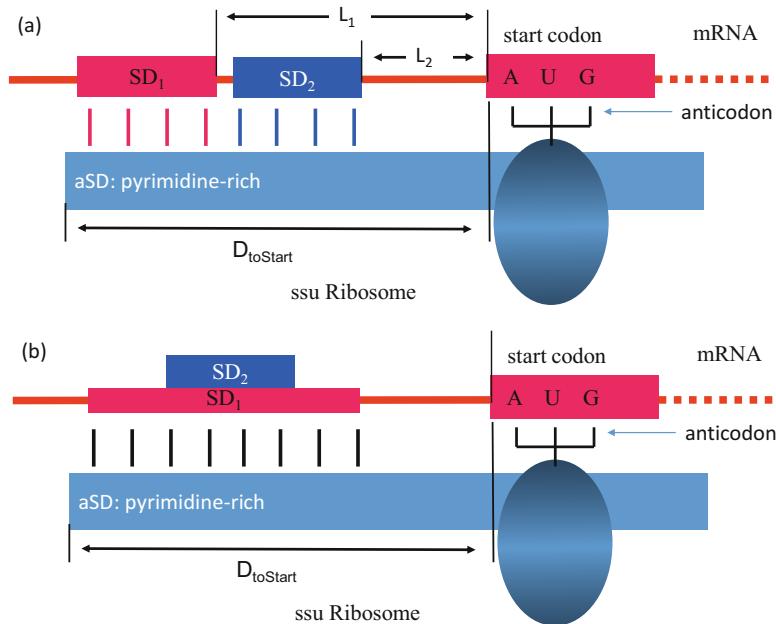
readily recognized, and the translation machinery will skip U (a 1-nucleotide frame-shift) and decode the GAC codon (Craigen et al. 1985; Donly et al. 1990).

Bacterial operons typically have RNA polymerase-binding sites at  $-10$  and  $-35$  positions (Fig. 8.1a), and the transcription start site is nucleotide G (bolded in Fig. 8.1a). However, translation start and termination are not as accurate as translation start and stop and do not always start or stop at the same site. Downstream of the transcription start site, we will find the translation initiation motif which is typically made of both a Shine-Dalgarno (SD) sequence (Shine and Dalgarno 1974a, b, 1975; Steitz and Jakes 1975; Taniguchi and Weissmann 1978) which forms base pair with the anti-SD (aSD) sequence at the 3' end of the small subunit ribosomal RNA (SSU rRNA) which will be referred hereafter as 3'tail. The model of base pairing between SD and aSD (Abolbaghaei et al. 2017; Prabhakaran et al. 2015) illustrates how the start codon is positioned against the anticodon of tRNA<sup>fMet</sup> (Fig. 8.2).

The start codon for *lacI* is GUG which still means methionine. Bacterial species frequently use non-AUG start codons such as GUG and UUG. In contrast to eukaryotes where the start codon itself is a key component of the initiation signal, bacterial start codon localization mainly depends on SD/aSD pairing (Fig. 8.1b). At the start codon, translation machinery transits from initiation to elongation which eventually brings us to the stop codon UGA which, being followed by a G (Fig. 8.1), represents a reasonable strong termination signal for RF2 (Wei et al. 2016). Further downstream we find the secondary structure typical of rho-independent termination (Farnham and Platt 1981; Wilson and von Hippel 1995).

## 2.2 A Model of SD/aSD Interaction and Other Factors Contributing to Translation Initiation

How does bacterial species translation machinery manage to juxtapose the start codon against the anticodon of initiation tRNA? The mechanism is revealed partially by bioinformatic approaches (Shine and Dalgarno 1974a, b, 1975). In short, known ribosome-binding sites were compared and found to share a purine-rich segment (now known as Shine-Dalgarno or SD sequence). This purine-rich segment can base pair against the 3' end of 16S rRNA (now known as anti-SD or aSD sequence). This led to the proposal that the juxtaposition of start codon and anticodon of initiation tRNA is facilitated in most bacterial species by base pairing of SD sequence located upstream of the start codon (Hui and de Boer 1987; Shine and Dalgarno 1974a, b, 1975; Steitz and Jakes 1975; Taniguchi and Weissmann 1978) and aSD located at the free 3' end of the small ribosomal rRNA, sometimes assisted by ribosomal protein S1 (RPS1) (Duval et al. 2013; Vellanoweth and Rabinowitz 1992). A well-positioned SD/aSD pairing, an AUG as a start codon, and reduced secondary structure in sequences flanking the start codon and SD are three prominent features of highly expressed genes in *Escherichia coli* and *Staphylococcus aureus* as well as their phages (Prabhakaran et al. 2015). This little historical digression shows that



**Fig. 8.2** Schematic model of SD/aSD interaction, illustrating  $D_{\text{toStart}}$  and the leash length ( $L$ ) for two SDs labeled SD<sub>1</sub> and SD<sub>2</sub>. (a) SD<sub>1</sub> and SD<sub>2</sub> have the same  $D_{\text{toStart}}$  but differ in leash length ( $L_1 \neq L_2$ ). (b) SD<sub>1</sub> and SD<sub>2</sub> have the same  $D_{\text{toStart}}$  but differ in both leash distance and base pairing strength ( $S_{\text{BP}}$ ). Modified from Wei et al. (2017)

bioinformatic approaches were already quite successful in early 1970s, with just a few ribosome-binding sites and the 3' end of 16S rRNA. The model also presents  $D_{\text{toStart}}$  as a measure for optimal positioning of SD/aSD pairs (Fig. 8.2).  $D_{\text{toStart}}$  is constrained within a narrow range (Abolbaghaei et al. 2017; Prabhakaran et al. 2015), especially for highly expressed genes. This suggests that  $D_{\text{toStart}}$  is under strong stabilizing selection.

SD/aSD is a key mechanism used by many bacterial genes for start codon localization. Recent studies (Abolbaghaei et al. 2017; Prabhakaran et al. 2015; Wei et al. 2017) proposed a new model of SD/aSD interaction (Fig. 8.2) that integrates three factors that contribute to the efficiency of SD/aSD interaction in localizing the start codon: (1) the position of the SD/aSD annealing measured by  $D_{\text{toStart}}$  (Fig. 8.2a, b), (2) the “leash length” shown as  $L_1$  and  $L_2$  for SD<sub>1</sub> and SD<sub>2</sub>, respectively (Fig. 8.2a), and (3) the strength of SD/aSD base pairing ( $S_{\text{BP}}$ ).  $D_{\text{toStart}}$  is measured by the number of bases from the 3' end of the SSU rRNA to the nucleotide just before the start codon. It is the major determinant of proper juxtaposition between the start codon and the anticodon of the initiation tRNA (or rather between the start codon and the A site of ribosome) and is consequently constrained within a narrow range, indicating its importance in determining translation initiation efficiency. The “leash length” ( $L$ ) is the number of bases between the 3' end of SD/aSD

base pairing and the start codon, i.e.,  $L_1$  and  $L_2$ , respectively, for SD<sub>1</sub> and SD<sub>2</sub> (Fig. 8.2).  $S_{BP}$  is typically measured by the number of consecutive pairs or by a weighted score. The simplest weight score accumulates three points for a G/C pair, two points for an A/U pair, and one point for a G/U pair. Figure 8.2b shows SD<sub>1</sub> and SD<sub>2</sub> that differ in  $S_{BP}$ .

For the SD/aSD pairing in *lacI* illustrated in Fig. 8.1b,  $D_{toStart} = 17$ , leash length  $L = 5$ , and  $S_{BP}$  following the simplest weight scheme is  $2 + 1 + 3 + 2 + 3 + 3 + 1 (= 15)$ . According to the model in Fig. 8.2, translation initiation efficiency ( $E_{TI}$ ) contributed from SD/aSD pairing can be expressed as

$$E_{TI} = F(D_{toStart}, S_{BP}, L) \quad (8.1)$$

We have illustrated how to measure the three independent variables, and program DAMBE (Xia 2013, 2017d) can automate the computation of these three variables in three simple steps. Take *Escherichia coli* genes, for example. First, download the *E. coli* genome in GenBank format, open the file in DAMBE to extract 20 or 30 nucleotides upstream of the start codon, and save these upstream sequences to a file, say UpstreamSeq.fas. Second, use DAMBE to extract the SSU rRNA and to obtain the free 3' end (3'tail). The sequences in UpstreamSeq.fas and the 3'tail are the only information needed to compute the three variables.

How to measure the dependent variable  $E_{TI}$ ? Without a reasonable measure of  $E_{TI}$ , we cannot build a quantitative model specified in Eq. (8.1). Theoretically, an mRNA with a high  $E_{TI}$  should be efficient in loading ribosomes onto its 5' end. Thus, operationally, we could use the density of ribosomes ( $D_r$ ) on the mRNA as a proxy of  $E_{TI}$ . One complication is that if translation elongation is slow, then even an mRNA without a high  $E_{TI}$  may have high ribosome density due to ribosome traffic jam. This problem can be eliminated by regressing  $D_r$  on  $I_{TE}$  (index of translation elongation, Xia 2015). Deviation from the regression line can then be used as a proxy of  $E_{TI}$ . For example, if  $D_r$  and  $I_{TE}$  are linearly related so that  $D'_r = \alpha + \beta I_{TE}$ , then an approximation of  $E_{TI}$ , designated by  $E'_{TI}$ , is simply  $(D_r - D'_r)$ .

The conceptual framework above should eliminate confusions in textbooks in molecular biology that take a simple-minded definition of SD as AGGAGG. Such a definition, without associated information on the position of AGGAGG, is meaningless. One can find AGGAGG upstream of the start codon that does not position properly the start codon against anticodon of the initiation tRNA. Such AGGAGG is not an SD. One can also find non-AGGAGG that can pair with the 3'tail to properly position the start codon against AC<sub>i</sub>. Many genes do not have AGGAGG, e.g., *lacI* does not, but has GGGUGGU which, as will be shown later, positions the start codon against AC<sub>i</sub> nearly optimally. Thus, defining SD as AGGAGG (or AGGAGGU) is misleading.

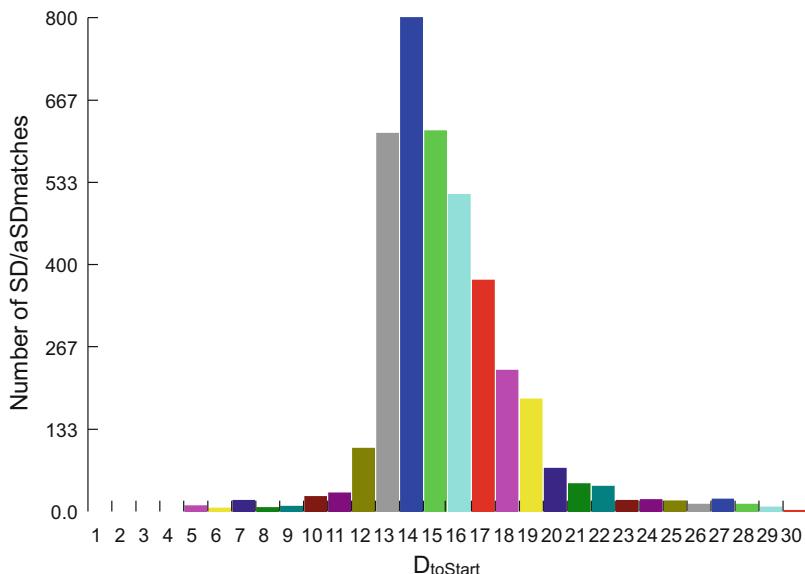
Occasionally a textbook will also specify that there should be a distance of about seven nucleotides between AGGAGG and the start codon. Such a specification defines only one specific SD, i.e., an AGGAGG spaced about seven nucleotides upstream of the start codon is an SD. For example, we have to redefine the SD in *lacI* as GGGUGGU spaced four nucleotides upstream of the start codon. Different SDs

with their respective optimal distances to the start codon are illustrated in Fig. 8.1c. For example, an AGGAGG with  $D = 7$  is about optimal, but this optimal  $D = 7$  is specific AGGAGG. With the *lacI* SD of GGGUGGU, the optimal  $D = 4$  is about optimal. Thus, without referring to the 3' tail, it is often meaningless to say what the optimal distance is.

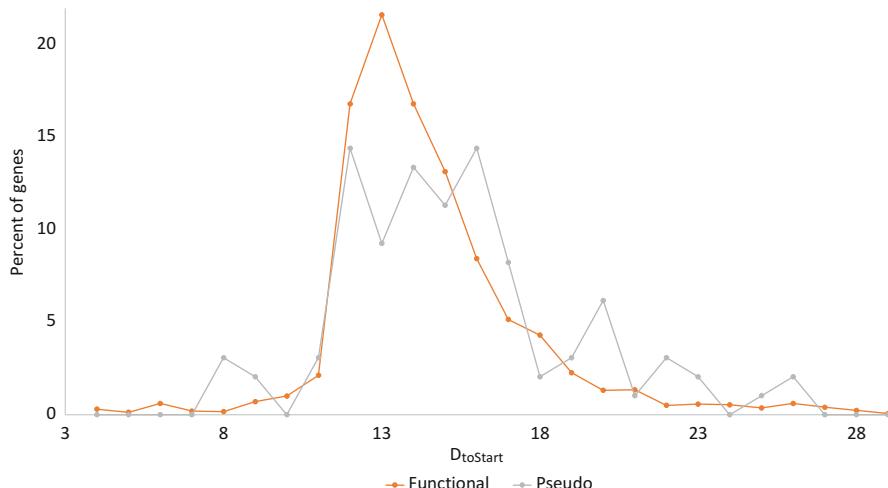
### 2.3 $D_{\text{toStart}}$ Is Constrained in a Narrow Range

As we can visualize from the model in Fig. 8.2, changing  $D_{\text{toStart}}$  will change the juxtaposition between the start codon and the anticodon of the initiation tRNA and affect the accuracy and efficiency of start codon localization. We consequently expect  $D_{\text{toStart}}$  to be strong constrained, and it is (Fig. 8.3). This pattern has been reported before (Abolbaghaei et al. 2017; Prabhakaran et al. 2015; Wei et al. 2017). The pattern is general, not only for the gram-negative *E. coli* but also for the gram-positive *B. subtilis* and a variety of other bacterial species.

To confirm that the pattern is maintained by natural selection, one may contrast the pattern between the functional genes and pseudogenes. The *E. coli* genome (NC\_000913) features 179 pseudogenes whose CDSs have not yet experienced frameshifting mutations and are presumably relatively young pseudogenes. If the peak is maintained by natural selection, presumably for optimal efficiency in translation initiation, then we expect the distribution for the pseudogenes to be less peaky



**Fig. 8.3** Frequency distribution of  $D_{\text{toStart}}$  for putative SD/aSD pairs in *E. coli* genes, obtained with the last 13 nucleotide at the 3' end of the small subunit rRNA of *E. coli*



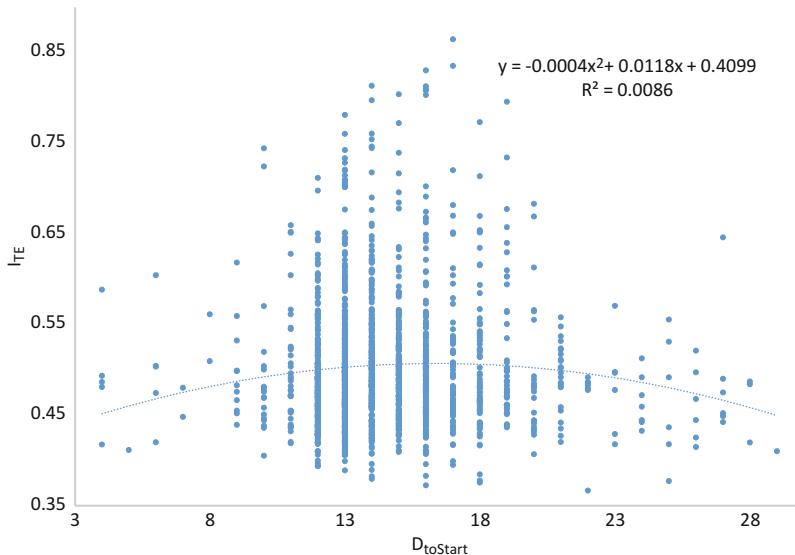
**Fig. 8.4** Contrasting the frequency distribution of  $D_{\text{toStart}}$  for putative SD/aSD pairs between functional genes (functional) and pseudogenes (pseudo) in *E. coli*

or off-peak. The contrast between the distribution pattern between the functional genes and the pseudogenes (Fig. 8.4) suggests strongly that  $D_{\text{toStart}}$  is under selection, because the peak is flattened in the distribution for pseudogenes. Thus, if one is to design a gene to be expressed in *E. coli*, one should make sure that  $D_{\text{toStart}}$  should be 14 or very close to it.

Not all genes have identifiable SDs. Among the 4140 functional genes in *E. coli* genome (NC\_000913), 19.9% do not have an identifiable SD. Among the 179 pseudogenes, 44.1% do not have an identifiable SD. Results in Fig. 8.4 are generated with functional and pseudogenes with an identifiable SD, defined as four consecutive base pairs between the putative SD and the aSD.

## 2.4 Genes with High D<sub>toStart</sub> Tend to Have Better Codon Adaptation

It has often been emphasized and empirically demonstrated (Xia 2015) that codon adaptation is most visible in genes with efficient initiation. For genes with inefficient translation initiation, optimizing codon usage would contribute little to protein production because translation initiation would limit protein production. For genes with highly efficient translation, translation elongation efficiency would become limiting, and such genes should, and have been shown to, exhibit strong codon adaptation (Xia 2015). Although  $D_{\text{toStart}}$  represents only one aspect of translation initiation, we could still predict that, given everything else being equal, genes with  $D_{\text{toStart}}$  near the peak in Fig. 8.3 should have better codon adaptation.

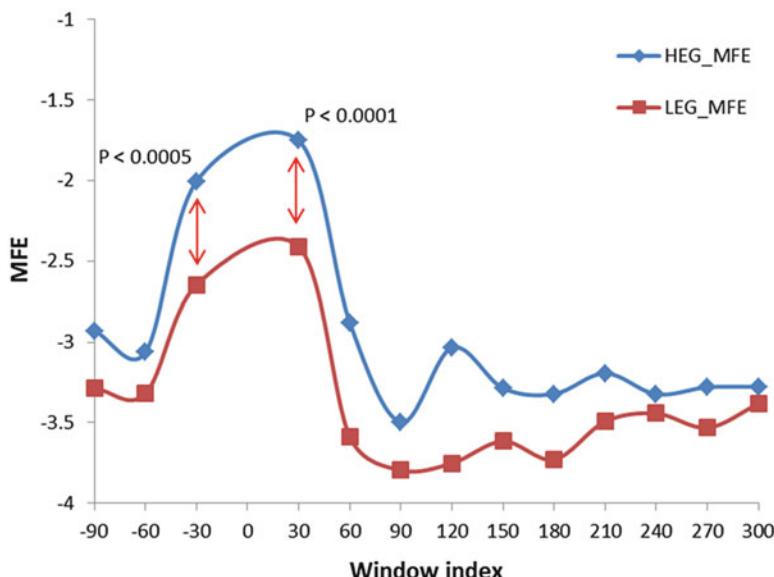


**Fig. 8.5** Relationship between  $I_{TE}$  values and  $D_{toStart}$  for *E. coli* genes, with the fitted order-2 polynomial relationship. The  $y$  and  $x$  variables in the fitted equation are  $I_{TE}$  and  $D_{toStart}$ , respectively. The slopes for  $x$  and  $x^2$  are both highly significant ( $p = 0.000273182$  for  $x$  and  $p = 0.000235339$  for  $x^2$ )

We may use the index of translation elongation ( $I_{TE}$ ) as a measure of codon adaptation (Xia 2015).  $I_{TE}$  takes background mutation bias into consideration and therefore is expected, and has been shown (Xia 2015), to perform better than CAI (codon adaptation index) which does not consider background mutation. We will cover  $I_{TE}$  in detail in the next chapter. The relationship is statistically significant (Fig. 8.5), but only a very small amount of variation in  $I_{TE}$  is explained by the model. However, this is expected because of the simplicity of the model. The next chapter will study factor contribution to codon adaptation.

## 2.5 Secondary Structure and Translation Initiation Efficiency

Secondary structure can either increase or decrease translation initiation efficiency (Kozak 1980b; Osterman et al. 2013; Prabhakaran et al. 2015). The minimum requirement for translation initiation is that the start codon is accessible (Nakamoto 2006). If a start codon is embedded in a stable secondary structure, then it will not be accessible. The classical example is the start codon AUG in the replicase gene of MS2 phage, which is embedded in a stem formed by part of the replicase gene and the upstream coat gene. Only when the coat genes are translated, separating the secondary structure as a consequence, can the downstream replicase gene be translated (Min Jou et al. 1972). Stable secondary structure in sequences flanking the start



**Fig. 8.6** Mean MFE (minimum folding energy, computed with DAMBE) of sliding windows along *E. coli* protein-coding genes, with window size of 40 nt. Window index 0 corresponds to the first 40 nt in CDS. Secondary structure stability decreases as MFE becomes less negative. *HEG* highly expressed genes, *LEG* lowly expressed genes

codon has been experimentally shown to inhibit translation initiation (Osterman et al. 2013), and mRNAs in bacterial species and unicellular eukaryotes tend to have much weaker secondary structure near the start codon than elsewhere, especially those from highly expressed (Fig. 8.6). MFE can be computed by using DAMBE process in protein production which implements the RNA folding library from Vienna RNA package (Hofacker 2003). Note that many *E. coli* genes have the configuration of UAAUG (where the first UAA is the stop codon of the upstream gene and the last AUG is the start codon of the downstream gene) or AUGA (where the first AUG is the start codon of the downstream gene and the last UGA is the stop codon of the upstream gene). For these genes, the flanking sequences may be interpreted as flanking either the start codon or the stop codon. Figure 8.6 is generated from nonoverlapping genes with an intergenic sequence of at least 100 nt.

mRNA secondary structure has also been experimentally shown to enhance translation initiation efficiency (Kozak 1980b; Osterman et al. 2013), especially when the secondary structure is positioned in such a way that the small ribosome subunit will spend more time scanning for the start codon. However, exactly where the secondary structure should be located remains uncertain.

## 2.6 Translation Initiation and Phage Host Specificity

*E. coli* possesses a more permissible translation machinery than *B. subtilis*, but there is one rare exception (Vellanoweth and Rabinowitz 1992). Gene 6 (*gp6*) of the *B. subtilis* phage  $\phi$ 29 can be translated efficiently in *B. subtilis* but not in *E. coli*. Among the 16 non-hypothetical genes in phage  $\phi$ 29, *gp6* is the only one that cannot form a well-positioned SD/aSD in *E. coli* (Table 8.1). Because *gp6* is an essential gene, its failure to form a proper SD/aSD in *E. coli* may explain its host specificity (Abolbaghaei et al. 2017). That is, even if it gains entry into an *E. coli*-like host, it will not be able to survive and reproduce successfully.

Another case of host specificity that may be explained by SD/aSD binding is *E. coli* phage PRD1 which has codon usage deviating much from that of its host, in contrast to the overwhelming majority of *E. coli* phages whose codon usage exhibits high concordance with that of the host (Chithambaram et al. 2014b). Phage PRD1 belongs to the peculiar *Tectiviridae* family whose other members parasitize gram-positive bacteria. Phage PRD1 is the only species in the family known to parasitize a variety of gram-negative bacteria including *Salmonella*, *Pseudomonas*, *Escherichia*, *Proteus*, *Vibrio*, *Acinetobacter*, and *Serratia* species (Bamford et al. 1995; Grahn et al. 2006). It is thus quite likely that the ancestor of phage PRD1 parasitizes gram-positive bacteria. The lineage leading to phage PRD1 may have switched to gram-

**Table 8.1** SD/aSD binding of non-hypothetical genes in *B. subtilis* phage  $\phi$ 29 in *E. coli* and *B. subtilis*. Gene *gp6* cannot form a well-positioned SD/aSD in *E. coli*

Gene	<i>E. coli</i>		<i>B. subtilis</i>	
	$D_{\text{toStart}}^{\text{a}}$	SD	$D_{\text{toStart}}^{\text{b}}$	SD
<i>gp2</i>	14	AAGGA	17	AAAGGA
<i>gp3</i>	17	AAGGAG	20	GAAAGGAG
<i>gp4</i>	18	AGGAGGU	21	AGGAGGU
<i>gp5</i>	15	AAGGA	18	AAAGGA
<i>gp6</i>			19	UAGAAAG
<i>gp7</i>	16	GAGGUGA	18,19	UAGAAAG,GAGGUGA
<i>gp8</i>	18	GAGGU	21,21	AGAAA,GAGGU
<i>gp8.5</i>	20	GGAGGUG	23	GGAGGUG
<i>gp9</i>	16,19	UAAGG,AGGUG	22	AGGUG
<i>gp10</i>	15	GAGGUGA	18	GAGGUGA
<i>gp11</i>	16	GGUGA	19	GGUGA
<i>gp12</i>	15	UAAGGAGG	18	AAGGAGG
<i>gp13</i>	17	GAGGU	20	GAGGU
<i>gp14</i>	17	AAGGAG	20	AAAGGAG
<i>gp15</i>	17	UAAGGAGG	20	AAGGAGG
<i>gp16</i>	16	GAGGUG	19	GAGGUG

<sup>a</sup>The optimal  $D_{\text{toStart}}$  is within the range of 10–21 in *E. coli*, with 3'tail being 5'GAUCACCUCCUUA3'

<sup>b</sup>3'AUCUUUCCUCCACUAG is used as 3'tail for *B. subtilis*, with the optimal  $D_{\text{toStart}}$  within the range of 15–25

negative bacterial hosts only recently and still has its codon usage similar to its ancestral gram-positive bacterial host, which is indeed the case (Chithambaram et al. 2014b). However, one non-hypothetical gene in phage PRD1 (*PRD1\_09*) has evolved an *E. coli*-specific SD (UAAG), and it does not have alternative SD that can form well-positioned SD/aSD with *B. subtilis* 3' tail. This may have contributed to the host limitation of phage PRD1 within *E. coli*-like species.

### 3 Translation Initiation in Eukaryotes

#### 3.1 Models of Eukaryotic Translation Initiation

##### 3.1.1 The Scanning Model of Translation Initiation

As I have mentioned in the section on prokaryotic translation initiation, bioinformatics, in the form of simple sequence comparison, was successful in discovering the SD/aSD pairing in prokaryotic translation initiation (Shine and Dalgarno 1974a, b, 1975). After much sequence comparisons, Kozak (1978, 1980a) became convinced that eukaryotic translation initiation is not just an extension of the prokaryotic model involving SD/aSD pairing and proposed the scanning model for eukaryotic translation initiation. That is, translation initiation factors recruit the 40S ribosomal subunit to the 5' cap; the ribosomal subunit scans down the mRNA to find the first AUG as the start codon. This is supported by the observation that all initiation (from 22 eukaryotic mRNAs) is at the first AUG, in spite of very heterogeneous 5'UTRs.

Unfortunately for this first version of the scanning model, counterexamples almost immediately appeared in which the first AUG is not used as the start codon. This prompted Kozak (1981) to examine the possibility of nucleotides flanking the start codon contributing to the signal for a start codon, the possibility that she casually dismissed before (Kozak 1980a). She compiled 153 mRNA sequences, aligned them by their start codon, and examined nucleotide frequencies of flanking nucleotides. This led to the proposal of Kozak consensus of RxxAUGG, i.e., -3R and +4G are far more frequent than other nucleotides flanking the start codon. The scanning model was then revised to state that the 40S ribosome subunit scans from the cap site in a 5' to 3' direction and find the first AUG flanked by -3R or +4G or both as the start codon. Thus, four factors affect the efficiency of eukaryotic translation initiation according to the scanning model: (1) the nature of start codon, with AUG typically better than alternatives, (2) proximity of the start codon to the m7G cap, (3) sequences flanking the start codon, and (4) secondary structure not embedding the start codon but immediately downstreaming the start codon to slow down the scanning of small ribosomal subunit. We may have noted that it is mainly bioinformatic analysis of mRNA sequences that had led to this revised scanning model.

### 3.1.2 Four Different Models of Eukaryotic Translation Initiation

Experimental and bioinformatic studies have suggested a much more complex picture in eukaryotic translation. First, complete ribosomes have been found in the 5' untranslated region (5'UTR), suggesting that it is not just the small ribosomal subunit that does the scanning for the start codon. Second, internal ribosomal entry has been documented in many cases (Gilbert 2010; Lacerda et al. 2016; Liberman et al. 2015; Pyronnet et al. 2000; Svitkin et al. 2001), suggesting that the scanning does not need to start from the m7G cap, although this remains somewhat controversial (Kozak 2005, 2007). If internal ribosomal entry is frequent, then long mRNA would be expected to be effectively polycistronic because the chance of harboring an internal ribosome entry site increases with the length of CDS. Such internally initiated translation would lead to multiple proteins translated from the same mRNA.

Over the years, these studies have contributed much to the understanding of translation in addition to the conventional cap-dependent scanning model which has gradually changed. There are now four major models of translation initiation in eukaryotes, cross-classified by two variables. The first is whether the translation machinery starts scanning for the start codon from the 5' end of mRNA (Jackson et al. 2010; Kozak 1980a) or from internal ribosome entry sites (Doudna and Sarnow 2007; Elroy-Stein and Merrick 2007; Gilbert et al. 2007; Sonenberg and Meerovitch 1990; Yu et al. 2011). The second is whether the small ribosomal subunit does the scanning for the start codon or a fully formed ribosome can also perform the scan. While there is little controversy now on the occurrence of internal ribosome entry, only recent ribosome profiling data have offered strong empirical support for fully formed ribosomes along 5'UTR of mRNAs (Ingolia et al. 2014), suggesting that fully formed ribosomes may also scan for the start codon.

In contrast to eukaryotic internal ribosomal entry sites (IRESs) whose IRES activity decreases with the stability of secondary structure (Xia and Holcik 2009), many viral IRESs have strong secondary structure. Cricket paralysis virus (CrPV) has an IRES located at the intercistronic region that is capable of directly interacting with the ribosome via its complex secondary structure without any translation initiation factors (Jan and Sarnow 2002; Jan et al. 2001; Pestova et al. 2004; Schuler et al. 2006). The hepatitis C virus (HCV) has an IRES that can mimic the translation initiation complex so that it does not need initiation factors essential for cap-dependent translation (Boehringer et al. 2005; Pestova et al. 1998). The IRES mechanism of translation initiation allows viruses to carry on their translation, while the host cap-dependent translation has been shut down, and viral IRESs, especially those with relatively rigid secondary and tertiary structure such as in HCV, have consequently been recognized as promising drug targets (Komar and Hatzoglou 2005).

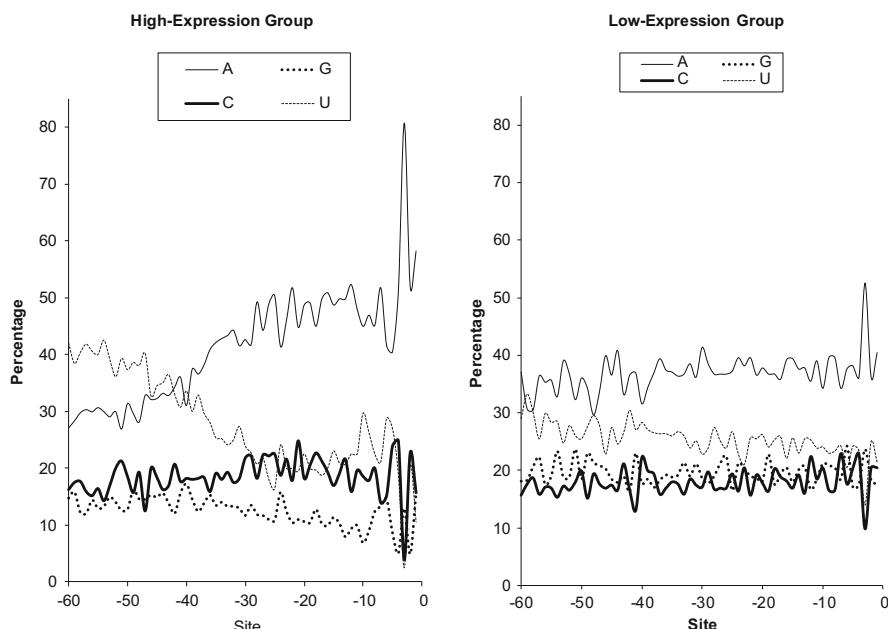
Translation regulation represents an important cellular mechanism capable of responding to extracellular environment. In the yeast *Saccharomyces cerevisiae*, a dozen or so genes are transcribed but not normally translated; they are translated when the surface nutrients have been depleted and their products enable yeast cells to burrow down into the culture medium to extract nutrients for growth (Gilbert et al. 2007). Ribosome profiling data can reveal the translation status of these translation

regulated messages and consequently help us understand how organisms use translation regulation in response to environmental changes.

Ribosome profiling is the ultimate tools to discover new protein-coding genes, many of which could be drug targets. That many protein-coding genes may remain unannotated is highlighted by the finding that even the extensively studied phage lambda may have unannotated protein-coding genes (Liu et al. 2013). In human and mouse, ribosomes are frequently found on transcripts not annotated as coding sequences, with the consequent production of polypeptides (Yoon et al. 2014). Given that the majority of the human genome are in fact transcribed (Birney et al. 2007), many new protein-coding genes may be discovered by bioinformatic analysis of ribosome profiling data (Popa et al. 2016).

### 3.2 Effect of Poly(A) in 5'UTR on Translation Initiation in *Saccharomyces cerevisiae*

5'UTR of the yeast (*S. cerevisiae*) immediately upstream of the start codon have increase occurrence of nucleotide A, especially in highly expressed genes (Fig. 8.7).



**Fig. 8.7** Contrasting site-specific nucleotide frequencies in 5'UTR between highly expressed and lowly expressed genes, with their CDS starting at site 1. Gene expression can be approximated by codon adaptation indices such as index of translation elongation ( $I_{TE}$ ) or by protein abundance. The resulting pattern remains the same using either method to characterize gene expression

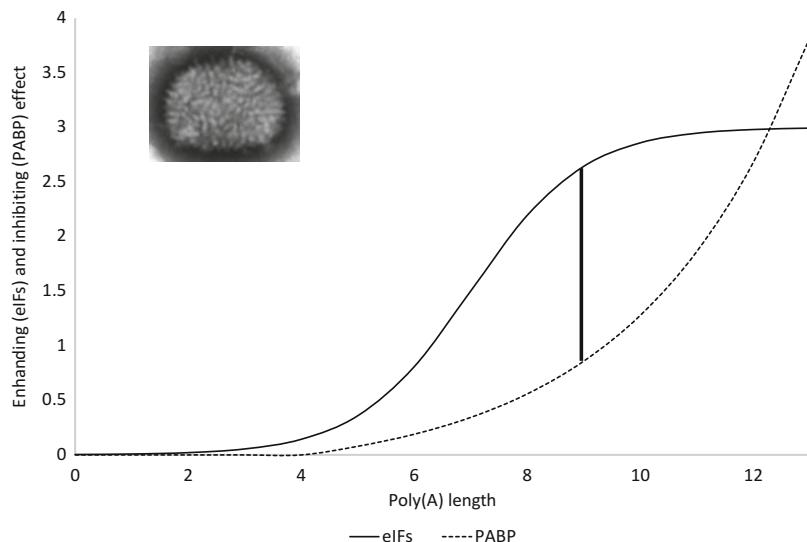
Poly(A) tracts are overrepresented in yeast 5'UTR. Given (1) the observation that highly expressed genes are particularly A-rich in 5'UTR and (2) that 5'UTR is essentially a loading dock for ribosomes to get on the mRNA, one naturally would ask if the poly(A) tract contributes to translation initiation efficiency (Xia et al. 2011).

Poly(A) tracts in 5'UTR have been recognized recently as important sites for translation regulation. These poly(A) tracts, referred previously (Xia et al. 2011) as pre-AUG  $A_N$  where N stands for the number of consecutive A nucleotides, can interact with translation initiation factors or poly(A)-binding proteins (PABP) to either increase or decrease translation efficiency. Pre-AUG  $A_N$  can enhance internal ribosomal entry both in the presence of PABP and eIF-4G in the yeast, *Saccharomyces cerevisiae* (Gilbert et al. 2007), and in the complete absence of PABP and eIF-4G (Shirokikh and Spirin 2008). Translation initiation factors eIF-4B and eIF-4F can bind to poly(A) tracts (Gallie and Tanguay 1994), and exogenous poly(A) added to an in vitro translation system can inhibit translation initiation (Grossi de Sa et al. 1988; Jacobson and Favreau 1983; Lodish and Nathan 1972), most probably by sequestering the translation initiation factors (Gallie and Tanguay 1994) and PABP (Gilbert et al. 2007). The inhibiting effect of exogenous poly(A) on translation can be removed by addition of either translation initiation factors eIF-4B and eIF-4F (with eIF-4A) in combination (Gallie and Tanguay 1994) or PABP (Gilbert et al. 2007; Grossi de Sa et al. 1988) which presumably would bind to the exogenous poly(A) and free translation initiation factors sequestered by the exogenous poly(A).

While pre-AUG  $A_N$  may improve translation efficiency, a few studies (Bag 2001; Bag and Bhattacharjee 2010; Ma et al. 2006; Melo et al. 2003a, b; Patel and Bag 2006; Patel et al. 2005; Wu and Bag 1998) suggest an inhibitory effect of PABP when it binds to a long pre-AUG  $A_N$  and presumably interferes with the scanning mechanism of translation initiation. The binding site of Pab1p in the yeast is about 12 consecutive A nucleotides, with the binding affinity decreasing rapidly with shorter poly(A) until 8 below which there is little affinity (Sachs et al. 1987). This suggests that mRNAs with pre-AUG  $A_N$  of different lengths may interact differently with yeast Pab1p and have different translation efficiencies.

A previous study (Shirokikh and Spirin 2008) characterized the effect of pre-AUG  $A_N$  on translation initiation efficiency, with  $N$  equal to 5, 12, and 25 in separate experiments. Surprisingly, the translation initiation efficiency is  $A_{25} > A_{12} > A_5$  (Shirokikh and Spirin 2008). One would have expected  $A_{25}$  and  $A_{12}$  to bind to PABP tightly and inhibit translation initiation and consequently should not have translation initiate efficiency higher than  $A_5$ . With this puzzle in mind, I contacted the authors and found that the translation system used does not include any PABP. The lack of PABP would explain  $A_{25} > A_{12} > A_5$  because longer  $A_N$  would be more effective in recruiting eukaryotic translation initiation factors.

These studies leads to the prediction that pre-AUG  $A_N$  should not have  $N > = 12$  because the binding size for PABP in the yeast is 12. Short pre-AUG  $A_N$  can recruit eukaryotic translation initiation factors and enhance translation initiation, but long pre-AUG  $A_N$  would bind to PABP and inhibit translation initiation (Fig. 8.8). How to test this prediction? One could use previously used translation system (Shirokikh and Spirin 2008) by simply adding PABP to check if  $A_{25}$  and  $A_{12}$



**Fig. 8.8** Schematic drawing illustrating the consequence of increasing length of pre-AUG poly(A) tract on translation initiation efficiency, mediated by eukaryotic translation initiation factors (eIFs) and poly(A)-binding protein PABP. Longer poly(A) will be more effective in recruiting eIFs, but this would be more than offset by the binding of poly(A) by PABP with long poly(A). The optimal length of poly(A) is marked by the vertical bar. The inset is vaccinia virus which has a molecular peculiarity that can be explained by the model (see text)

would now exhibit decreased translation initiation efficiency. However, such experiments typically involve only one or a few genes. Because each gene has its own peculiarities, the results from one or a few genes often cannot be generalized.

A large-scale study involving many genes would require the characterization of (1) pre-AUG  $A_N$  and (2) translation initiation efficiency for each gene. Several recent technological breakthroughs have contributed to the characterization of these two variables in *S. cerevisiae*. First, the transcription start site (TSS) necessary for the accurate delineation of 5'UTR and pre-AUG  $A_N$  has been characterized for thousands of yeast genes by direct mapping of the capped yeast mRNA sequences (Miura et al. 2006). Second, recent proteomic study (Ghaemmaghami et al. 2003) and ribosome profiling studies (Arava et al. 2003; Ingolia et al. 2009; MacKay et al. 2004) provide a measure of translation efficiency in the form of ribosomal density and the protein synthesis rate for thousands of yeast genes.

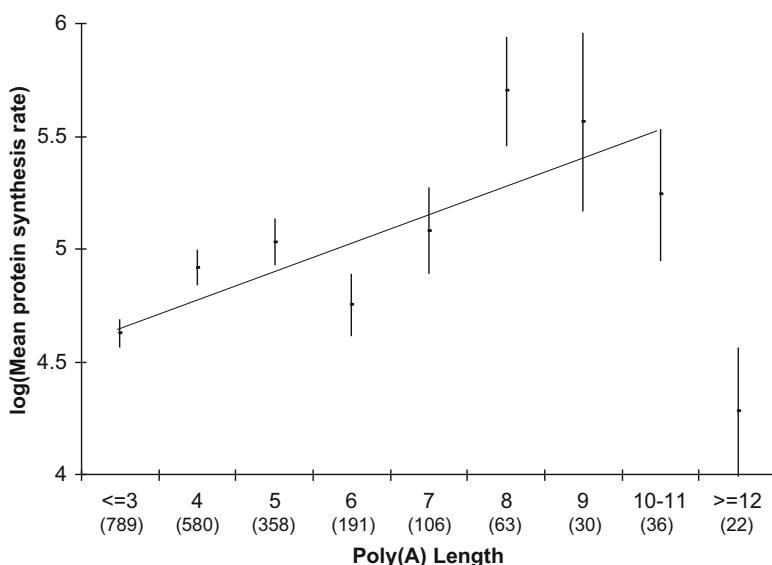
There are two approaches to obtain ribosome profiling data. The first (Arava et al. 2003; MacKay et al. 2004) is to isolate mRNA with various numbers of ribosomes attached to them. Careful centrifugation in a density gradient will have free mRNA floating on top and mRNA with most ribosomes at the bottom. These mRNAs can then be separated and identified, so that we know which mRNA has no ribosome attached and which has one, two, or many ribosomes attached. mRNAs from a gene with no ribosome attached are not actively translated, whereas mRNA from another

gene with many ribosomes attached is actively translated. This gives a proxy of translation activity for mRNA in different genes. This approach is now replaced by the deep-sequencing approach (Ingolia et al. 2009, 2011), which sequences all mRNA fragments protected by the binding of ribosomes (about 30 nt). These fragments are then mapped onto the genes so that we know exactly which mRNA has how many ribosomes translating it and where these ribosomes are positioned along the mRNA.

In short, there are sequence information for yeast 5'UTRs for characterizing AN, and there are proxies for translation efficiency derived from ribosome profiling data. So it is easy to test the prediction based on the conceptual framework in Fig. 8.8. That is, pre-AUG  $A_N$  should increase translation initiation, but long AN will decrease translation efficiency because it will recruit PABP at 5'UTR and interfere with the scanning of ribosome along the 5' end of mRNA.

The key finding, presented in Fig. 8.9, is consistent with the prediction that the optimal length of pre-AUG  $A_N$  is shorter than 12. Before  $A_N$  begins to bind to PABP, longer  $A_N$  increases protein synthesis rate. However the rate decreases dramatically when  $A_N$  is of length 12 or longer (Fig. 8.9). Similar patterns were observed when the rate is replaced by ribosome occupancy (percent of mRNA in polysome state) or ribosome density (Xia et al. 2011).

The finding that the length of pre-AUG  $A_N$  is strongly associated with ribosomal loading and protein synthesis sheds light on a peculiar molecular phenomenon in



**Fig. 8.9** Protein synthesis rate (lnMeanRate) increases with the length of pre-AUG poly(A) but decreases dramatically when the poly(A) length reaches 12. The straight line indicates the regression line for genes with poly(A) length shorter than 12. The vertical bars show one standard error above and below the mean for each poly(A) length category. Parenthesized numbers indicate number of genes in each poly(A) length category. Modified from Xia et al. (2011)

vaccinia virus which takes a great trouble to synthesize pre-AUG A<sub>N</sub> (which is not encoded in the genome). The length of the synthesized pre-AUG A<sub>N</sub> in vaccinia virus genes differs between early and late genes. The early viral genes have a pre-AUG A<sub>N</sub> with 4–14 A residues (Ahn et al. 1990; Ink and Pickup 1990), but the late genes often have pre-AUG A<sub>N</sub> with around 35 A residues (Bertholet et al. 1987; Schwer and Stunnenberg 1988; Schwer et al. 1987). The early viral genes are translated in the presence of abundant PABP which would repress the translation of mRNAs with a long pre-AUG A<sub>N</sub>. This implies that the viral early mRNAs should have only short A<sub>N</sub> to avoid repression. In contrast, late viral genes are translated when the cellular protein production has been much reduced, i.e., when PABP is expected to be less abundant. So mRNAs from late viral genes can have long pre-AUG A<sub>N</sub> without suffering much from translation repression mediated by PABP. It has been experimentally demonstrated that, in the absence of PABP, the translation enhancing effect of pre-AUG A<sub>N</sub> increases with its length (Shirokikh and Spirin 2008).

There is some controversy concerning whether the PABP level is reduced during the infection cycle of vaccinia virus. The degradation of host mRNA appears nearly complete 6 h after the viral infection as no host poly(A) mRNA is detectable at/after this time (Katsafanas and Moss 2007). Furthermore, a large-scale characterization of mRNA of HeLa cells infected with vaccinia virus (Yang et al. 2010) showed that PABP mRNA was reduced to 50% by 4 h. Although no mRNA characterization is done after this time, intuition would suggest continued reduction, and such a suggestion is consistent with the finding that no host mRNA is detectable after 6 h after the viral infection (Katsafanas and Moss 2007).

The study by Katsafanas et al. (2007) also showed that the viral mRNAs are located in the cavities of viral factories (VFs), where they are transcribed and translated. A number of translation initiation factors such as eIF4E and eIF4G are also localized in these cavities (Katsafanas and Moss 2007; Walsh et al. 2008). In contrast, PABP is localized on the periphery of VF (Walsh et al. 2008) which suggests that PABP does not participate in translation of the viral genes. It is known that vaccinia virus produces poly(A) nontranslated small RNA sequences that selectively inhibit cap-dependent translation of host messages (Bablanian and Banerjee 1986; Bablanian et al. 1986, 1987 1993; Lu and Bablanian 1996), presumably by binding to PABP and preventing it from interacting with other translation initiation factors. Both rubella virus and Bunyamwera virus inhibit translation of host genes by producing a capsid protein that binds to PABP and prevents it from binding to other translation initiation factor (Blakqori et al. 2009; Ilkow et al. 2008).

### ***3.3 Internal Ribosome Entry Site Activity and Stability of Secondary Structure***

We have discussed briefly the conventional cap-dependent scanning model of eukaryotic translation initiation, the associated recruitment of translation initiation

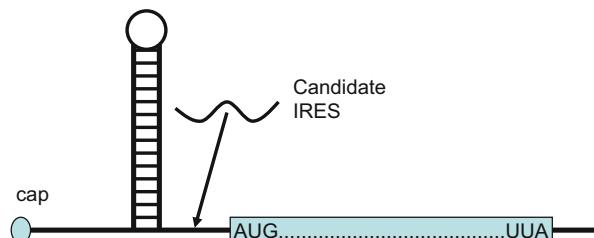
factors, and the effect of poly(A) tracts in yeast 5'UTR on translation initiation in the previous section. Recent studies have shown that many viral and eukaryotic genes have complex and stable secondary structure(s) in their 5'UTR which would block the scanning by the initiating ribosome along mRNA in search for the start codon in a favorable context. In particular, such mRNAs are often efficiently translated when the essential components of cap-dependent translation initiation complex (e.g., eIF4E) are compromised in the cell. The accumulation of these observations (for review, see Doudna and Sarnow 2007; Elroy-Stein and Merrick 2007) suggested mechanisms beyond the cap-dependent scanning model of translation initiation, leading to the proposal of internal ribosome entry mechanism of translation initiation which is facilitated by IRES (internal ribosome entry site).

Many cellular IRESs have been experimentally identified, mostly with a translation system in which an mRNA features a strong secondary structure, typically in the form of a long and stable hairpin, immediately upstream of the start codon (Fig. 8.10). The strong secondary structure prevents the scanning for start codon by ribosome, so little protein is produced from the downstream CDS. Various RNA segments are then inserted between the secondary structure and the start codon. An inserted RNA segment that allowed the production of proteins from the downstream CDS is then designated as an IRES. A strong IRES allows many proteins to be produced, and a weak IRES results in only trace amount of proteins produced from the downstream CDS.

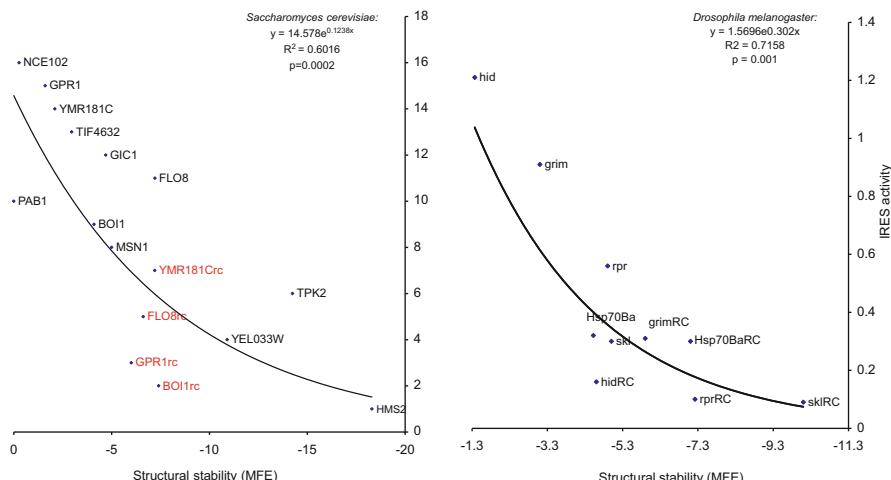
The experimental protocol above has generated many IRESs and IRES databases have been created. Biochemists and bioinformaticians instilled with the idea that a function is always associated with a structure immediately began the search of shared sequence or structure similarities among these identified IRESs. After all, many different types of RNA molecules depend on specific structure to perform their functions, especially tRNA, rRNA, siRNA, snRNA, etc. Surprisingly, while RNA structure plays a significant role in viral IRESs, there has been little sequence or structural similarity identified among the eukaryotic IRESs, leading to suggestions that RNA structure is not important for IRES activity (Dorokhov et al. 2002; Hernández 2008; Hernandez et al. 2004; Terenin et al. 2005; Vazquez-Pianzola et al. 2007). For example, mutations in the IRES element of *XIAP* that changed the secondary structure of this IRES had no impact on the *XIAP* IRES activity (Baird et al. 2007). Does RNA structure truly play no role in IRES activities?

Xia and Holcik (2009) asked a simple question. Is IRES activity associated with secondary structure stability? To address this question, only two variables are

**Fig. 8.10** Typical mRNA construct for identifying IRES



needed. One is the minimum folding energy (MFE) as a proxy of secondary structure stability, and the other is IRES activity. MFE is simple to compute. One only needs to start the software DAMBE (Xia 2013, d), click “FileRead in standard sequence file” to read in a file containing sequences of which we need to computer MFE, and click “StructurelRNA minimum folding energy.” The other variable, IRES activity, has been determined for two separate sets of IRESs, one for the yeast (Gilbert et al. 2007) and the other for the fruit fly (Hernandez et al. 2004; Vazquez-Pianzola et al. 2007). The yeast data includes 12 yeast genes (*NCE102*, *GPR1*, *YMR181C*, *GIC1*, *FLO8*, *BOI1*, *MSN1*, *PAB1*, *eIF4G2*, *TPS2*, *HMS2*, and *YEL033W*) whose 5'UTRs differ dramatically in IRES activity (Gilbert et al. 2007). The IRES activity in these genes was mapped to 60 nt immediately upstream of the initiation AUG (Gilbert et al. 2007). Also included in the analysis were the reverse complements of four of the experimentally identified yeast IRESs in *YMR181C*, *GPR1*, *FLO8*, and *BOI1* (designated as *YMR181Crc*, *GPR1rc*, *FLO8rc*, and *BOI1rc*, respectively). Thus, for each 60mer, IRES activity was measured experimentally and MFE calculated bioinformatically with DAMBE. The results, surprisingly, shows that the IRES activity of yeast IRESs is strongly associated with the lack of secondary structure measured by MFE, in the unit of kcal/mol of the 60 nt immediately upstream of the initiation AUG (left panel in Fig. 8.11,  $r = -0.7756$ ,  $p = 0.0002$ ), contrary to the conventional belief that IRESs should have complex and stable secondary structure (Baird et al. 2006; Boehringer et al. 2005; Komar and Hatzoglou 2005; Schuler et al. 2006). This result suggests that structureless RNA segments immediately upstream



**Fig. 8.11** Negative correlation between IRES activity, measured by the protein production of the downstream gene (protein/mRNA), and structural stability, measured by the minimum free energy (MFE, in kcal/mol). The MFE is shown in reverse order because greater stability is associated with more negative MFE values. The ranking of yeast IRES activity is based on Gilbert et al. (2007) and verified by the authors. The reverse complements of four IRES-containing genes are colored in red. TIF4632 is the name in GenBank for gene eIF4G2. The ranking of fruit fly IRES activity is based on Hernandez et al. 2004; Vazquez-Pianzola et al. 2007

of the initiation AUG can facilitate internal ribosome entry in yeast. It is remarkable that the reverse complements of four of the identified IRESs have little IRES activity (Gilbert et al. 2007) and also exhibit relatively stable secondary structure (Fig. 8.11).

What is remarkable is that the IRES data from the fruit fly (right panel in Fig. 8.11) exhibit the same pattern as in the yeast data. IRES activity of the 5'-UTR sequences in five *Drosophila melanogaster* protein-coding genes, *rpr*, *hsp70*, *hid*, *grim*, and *skl*, as well as the IRES activity of the reverse complement of the 5'UTR sequences were experimentally measured. The IRES activity of the fruit fly IRESs is also strongly associated with the lack of secondary structure measured by MFE (right panel in Fig. 8.11,  $r = -0.8461$ ,  $p = 0.001$ ), consistent with the pattern observed in the yeast data. The reverse complements of the 60 nt 5'UTR sequences in the five fruit fly genes have little IRES activity, and all exhibit relatively stable secondary structure. These results suggest the conclusion that structureless RNA segments immediately upstream of the initiation AUG can facilitate internal ribosome entry.

### 3.4 Two Alternative Hypotheses on Kozak Consensus

We have illustrated the application of bioinformatic methods to solve biological problems involving translation initiation, in particular the cap-dependent translation initiation and the internal ribosome entry. However, both the cap-dependent route and the internal ribosome entry route ultimately need to find the start codon. Mammalian mRNAs typically feature the Kozak consensus (Kozak 1981, 1991, 1999) in the general form of RCCaugGNN where “aug” is the start codon, “R” stands purine, and  $N$  stands for any nucleotide. The first R in the consensus is referred to as  $-3R$  and the G following “aug” as  $+4G$  (there is no site numbered 0). These two nucleotides ( $-3R$  and  $+4G$ ) are highly overrepresented in mammalian genes, suggesting their functional importance (Kozak 1997, 1999). Many experimental studies have demonstrated the importance of  $-3R$ . Changing the nucleotide to a pyrimidine always results in decrease protein production. In contrast, in spite of many studies aiming to demonstrate the importance of  $+4G$ , including many performed by Kozak herself (Kozak 1986, 1997), the results have been inconclusive.

An alternative hypothesis has been proposed with reference to  $+4G$  (Xia 2007a) claiming that the preponderance of  $+4G$  has nothing to do with translation initiation. This alternative hypothesis is built upon three key observations. First,  $N$ -terminal methionine excision (NME) of nascent peptides occurs in more than half of proteins in both prokaryotes and eukaryotes (Giglione et al. 2004; Giglione et al. 2003; Meinnel et al. 1993; Serero et al. 2003) as well as in mitochondria and plastids (Giglione et al. 2004). NME occurs soon after the  $N$ -terminal of the growing polypeptide chain emerges from the ribosome and is not only an important  $N$ -terminal modification in itself but also required for further posttranslational modifications. For example, it is required for myristylation which requires a glycine at the  $N$ -terminal, after the removal of the initiator methionine, to attach to a myristoyl

(C<sub>14</sub>H<sub>28</sub>O<sub>2</sub>) fatty acid side chain (Farazi et al. 2001). Second, the key requirement for NME is a small and nonpolar amino acid immediately following the first Met. NME is carried out by methionine aminopeptidase (MAP). A eubacterial species has only one type of MAP whereas a eukaryotic species typically has two types of MAP enzymes designated MAP 1 and MAP 2, respectively. According to experiments on eubacterial species such as *E. coli* (Chang et al. 1989) and *Salmonella typhimurium* (Miller et al. 1989), deletion mutants of the MAP gene is lethal. Deletion of both *MAP 1* and *MAP 2* genes in yeast is also lethal (Li and Chang 1995). Complete inactivation of MAP 1 and MAP 2 genes in *Arabidopsis thaliana* results in severe developmental abnormality (Ross et al. 2005). Such findings, together with the strong sequence and structural conservation of MAP proteins in prokaryotes, eukaryotes, mitochondria, and plastids (Giglione et al. 2004 and references cited therein), suggest that NME is an essential cellular function in living organisms (Frottin et al. 2006). NME in both prokaryotes and eukaryotes have the same requirement. NME is carried out efficiently as long as there is a small and nonpolar amino acid following the initiating Met. Third, the two smallest and nonpolar amino acids are glycine and alanine and are encoded by GGN and GCN codons. Taken together, these observations implies that more than half of the proteins need to have a small and nonpolar amino acid following the initiating Met which in turn means that the starting codon AUG should be followed mostly by GGN and GCN codons. This explains the preponderance of +4G simply because most proteins should have their mRNA starting with “AUG GGN” or “AUG GCN.” This new hypothesis is consequently named amino acid constraint hypothesis (Xia 2007a) but may be better termed NME constraint hypothesis.

The two hypotheses, i.e., the conventional translation initiation hypothesis and the alternative NME constraint hypothesis, are not mutually exclusive. However, they do have different predictions. The conventional translation initiation hypothesis predicts increased translation initiation efficiency with +4G. Thus, one predicts to have more +4G in efficiently translated genes than those rarely translated, and it does not matter if this +4G is supplied by GCN, GGN, GUN, or GAN following the initiation codon. In contrast, the new NME constraint hypothesis predicts that the preponderance of +4G is due to NME, and +4G is mainly supplied by GCN and GGN that encode Ala and Gly facilitating NME. Future studies may focus on mRNAs for non-NME proteins in discriminating between the two hypotheses.

## Postscript

We have seen that same overrepresentation of +4G in the Kozak consensus RxxAUGG can be interpreted in two different ways, one by the translation initiation hypothesis and one by the amino acid constraint hypothesis. Both are plausible and demand further study. This highlights the importance of different research perspectives, often brought about by researchers of different fields.

Having different perspectives can avoid blunders and embarrassment. I happen to have an ancient Islamic fable to illustrate this point. Afandi was a Turkish man of wisdom who rode his little donkey around the country to teach people to be wise. The king, driven by the desire to show himself smarter than Afandi, had designed a plot. He invited Afandi into the court to be seated next to him in a round table when honeydew melon was served. While they were eating, the king's courtiers would engage Afandi's attention with conversations, so that the king would stealthily push the melon rind onto Afandi's side. After eating, the king, pointing to the large pile of melon rind in front of Afandi, declared that Afandi must have been either starved or greedy, having devoured twice as much melon as everyone else. In the midst of ensuing jeering laughter, Afandi turned around, seeing no melon rind in front of the king, and calmly replied, "Your majesty is right that I am hungry today, but at least I know how to eat a melon. Someone among us has devoured the melon rind!"

The king must have regretted not to have thought the plot over in different perspectives.

# Chapter 9

## Bioinformatics and Translation Elongation



### 1 Introduction

We will first learn a few key definitions and notations on tRNA, its anticodon, and codon families. We will then outline the conceptual framework of codon adaptation, mediated by mutation and selection. This brings us to indices of codon usage bias, their calculation and interpretations, and factors that may confound their interpretations. There are codon-specific indices such as relative synonymous codon usage (RSCU, Sharp et al. 1986) or gene-specific indices such as index of translation elongation ( $I_{TE}$ , Xia 2015) and codon adaptation index (CAI, Sharp and Li 1987; Xia 2007c). All these indices are implemented in DAMBE (Xia 2013, 2017d).

$I_{TE}$  takes background mutation bias into consideration, while CAI does not.  $I_{TE}$  is reduced to CAI if there is no background mutation bias. I will illustrate the applications of these indices in practical research. Keep in mind that a codon adaptation index is just one variable which will not be particularly interesting until you relate it to other variables and understand their relationships.

Two additional topics are dealt with close to the end of the chapter. The first involves how to discriminate between selection for translation efficiency and accuracy (Akashi 1994). The second is on the effect of amino acid usage on translation elongation efficiency. The general prediction concerning amino acid usage is that highly expressed proteins should maximize the use of amino acids that are abundant and energetically cheap (Akashi and Gojobori 2002) to make and have many tRNAs to carry them (Xia 1998a). The same argument has been used for transcription, i.e., an mRNA with many A nucleotides will be transcribed faster than one with many C nucleotides because A is in general far more abundant than C and it takes extra ATP to make CTP (Xia 1996; Xia et al. 2006).

## 1.1 Basic Notations, Definitions, and Abbreviations

Notations, definitions, and abbreviations are essential in science. We are lucky enough to have almost all of them unambiguous. If you were studying social sciences, you would have to come to define what is man and what is woman, and the debate on a proper definition will last forever, eventually with all debaters losing their mind and being called jerks.

### 1.1.1 tRNA Notation and Identification of tRNA Anticodon

The simplest notation of a tRNA is  $tRNA^{AA}$ , where AA is a specific amino acid. For example,  $tRNA^{Gly}$  refers to all tRNAs that can be charged with amino acid glycine (Gly). A slightly more complicated notation is  $tRNA^{AA/AC}$ , where AC refers to tRNA anticodon. For example,  $tRNA^{Gly/GCC}$  refers specifically to  $tRNA^{Gly}$  with a GCC anticodon. The general notation of a tRNA is  $AA_2\text{-}tRNA^{AA_1/AC}$ , where  $AA_1$  is the amino acid the tRNA is supposed to carry,  $AA_2$  is the amino acid that is actually carried by the tRNA, and AC is the anticodon. In most cases,  $AA_1$  and  $AA_2$  are the same. However, there are two cases where  $AA_1$  and  $AA_2$  can be different. The first is modification of  $AA_2$  by a biochemist. The second occurs naturally in a number of species across all three domains of life (Sheppard et al. 2008; Yuan et al. 2008), where Gln-tRNA<sup>Gln</sup>, Asn-tRNA<sup>Asn</sup>, Cys-tRNA<sup>Cys</sup>, and Sec-tRNA<sup>Sec</sup> are formed indirectly by two steps. Take Gln-tRNA<sup>Gln</sup> and Asn-tRNA<sup>Asn</sup>, for example. Glu is first misacylated to tRNA<sup>Gln</sup>, and Asp to tRNA<sup>Asn</sup>, to form Glu-tRNA<sup>Gln</sup> and Asp-tRNA<sup>Asn</sup>, respectively. The resulting misacylated tRNAs are then converted to Gln-tRNA<sup>Gln</sup> and Asn-tRNA<sup>Asn</sup> by a group of tRNA-dependent modifying enzyme.

Isoacceptor tRNA is a somewhat confusing term as it may carry two slightly different meanings. It could refer to a single tRNA decoding different synonymous codons, e.g.,  $tRNA^{Gly/GCC}$  decoding GGC and GGU codons. Alternatively, it could refer to a set of different tRNAs that carry the same amino acid but decode different synonymous codons. For example,  $tRNA^{Gly/GCC}$ ,  $tRNA^{Gly/CCC}$ , and  $tRNA^{Gly/UCC}$  are isoacceptor tRNAs. They all carry amino acid Gly but with different anticodons decoding different synonymous Gly codons. Different isoacceptor tRNAs could decode the same codon. For example,  $tRNA^{Gly/CCC}$  decodes GGG, but  $tRNA^{Gly/UCC}$  decodes both GGA and GGG, so GGG is decoded by both  $tRNA^{Gly/CCC}$  and  $tRNA^{Gly/UCC}$ . Thus, isoacceptor tRNA refers to (1) one tRNA decoding different synonymous codons or (2) a set of tRNAs that carry the same amino acid but decode different sets of synonymous codons. The intersection of different sets of synonymous codons may not be empty. For example, the set of codons decoded by  $tRNA^{Gly/CCC}$  is {GGG}, and the set of codons decoded by  $tRNA^{Gly/UCC}$  is {GGA, GGG}. The intersection of the two sets is {GGG}.

Related to isoacceptor tRNA is another potentially confusing concept, near-cognate tRNA, which is defined in two ways. The first is based on empirical

evidence. If codon XYZ encoding amino acid AA1 can be misread by tRNA carrying amino acid AA2 ( $\text{AA1} \neq \text{AA2}$ ), then that tRNA is a near-cognate tRNA for codon XYZ. The second definition is based on nucleotide similarity among codons. A codon XYZ has nine XYZ-like codons which differ from XYZ by a single nucleotide. Some of these XYZ-like codons are synonymous to XYZ and some not. The set of tRNAs that can decode any of those nonsynonymous XYZ-like codons are near-cognate tRNAs for codon XYZ because they can “potentially” misread codon XYZ. For example,  $\text{tRNA}^{\text{Asp}}$  is a near-cognate for codons GAA and GAG because Asp is encoded by GAC and GAU which are GAA-like and GAG-like codons.

### 1.1.2 Genetic Codes and Associated Concepts and Definitions

It is through genetic code that the 64 codons are interpreted as encoding amino acids or translation stop. Nature is superfluous in her creation of genetic code. There are now 24 known genetic codes listed from 1 to 31 (Table 9.1). The standard genetic code is shown previously in Table 2.7.

Some codons do not change their meanings, e.g., Phe (UUY), Tyr (UAY), and Pro (CCN), whereas some others change their meaning frequently. Table 9.2 lists those codons with different meanings in different genetic codes. These codons tend to end with a purine, except for CUN. However, even within the CUR codon family, CUR codons are involved in recoding more often than CUY codons (Table 9.2).

We can build a distance tree from Table 9.2 by counting the pairwise number of reassignment events (i.e., when a codon for one amino acid is reassigned to a different amino acid or a stop codon). The only problem is how to treat reassignment between a sense codon and a stop. Such a change probably should occur less frequently than reassessments involving two sense codons. All pairwise comparisons among the 24 rows (24 genetic codes) generate 609 reassessments involving 2 sense codons and 445 reassessments between a sense codon and a stop codon. However, during the long evolutionary time, the more frequent reassessments will erase each other and the frequencies of their occurrences will be underestimated. So the actual difference between the two numbers must be much greater. If we count each reassignment between a sense codon and a stop codon as equivalent to four reassessments between two sense codons, we obtain a distance-based tree in Fig. 9.1. The topology remains the same if we treat each reassignment between a sense codon and a stop codon as equivalent to two, three, or five reassessments involving two sense codons.

Most bacteria use genetic code 11 which is the same as the standard code except for the difference in start codon usage. The wall-less bacteria including *Mycoplasma* and *Spiroplasma* use genetic code 4 which is identical to the mitochondrial genetic code used in a number of fungal lineages, red algae, and protozoa. The use of the same genetic code 4 by bacteria and mitochondria in eukaryotic lineages suggests two alternative hypotheses. First, it is convergence. Second, the ancestor of

**Table 9.1** The 24 genetic tables named after representative species and corresponding translation tables (TT)

Name	TT
Standard	1
Vertebrate mitochondrial	2
Yeast mitochondrial	3
Mold, protozoan, and coelenterate mitochondrial code and the <i>Mycoplasma/Spiroplasma</i>	4
Invertebrate mitochondrial	5
Ciliate, Dasycladacean, and <i>Hexamita</i> nuclear	6
Echinoderm and flatworm mitochondrial	9
Euplotid nuclear	10
Bacterial, archaeal, and plant plastid	11
Alternative yeast nuclear	12
Ascidian mitochondrial	13
Alternative flatworm mitochondrial	14
Chlorophycean mitochondrial	16
Trematode mitochondrial	21
<i>Scenedesmus obliquus</i> mitochondrial	22
Thraustochytrium mitochondrial	23
Pterobranchia mitochondrial	24
Candidate division SR1 and <i>Gracilibacteria</i>	25
<i>Pachysolen tannophilus</i> nuclear	26
Karyorelict nuclear	27
<i>Condyllostoma</i> nuclear	28
<i>Mesodinium</i> nuclear	29
Peritrich nuclear	30
<i>Blastocrithidia</i> nuclear	31

mitochondrial lineages in Cluster 3 (Fig. 9.1) is a *Mycoplasma*-like bacteria. This would imply multiple origin of mitochondrial lineages.

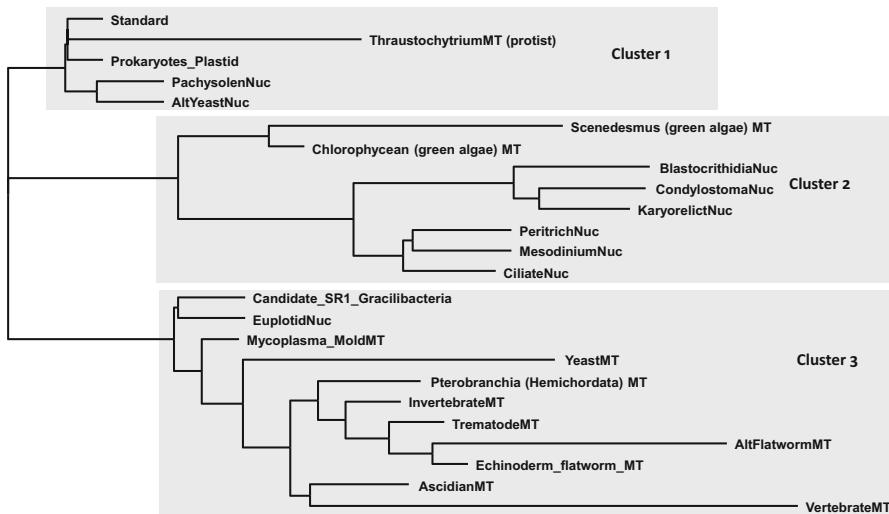
The main arguments for a single origin of mitochondria are (1) extensive phylogenetic reconstruction with rRNA sequences from diverse array of mitochondrial and bacterial lineages appears to recover mitochondrial lineages as a monophyletic taxon, with its closest phylogenetic relative being in *Alphaproteobacteria* lineages, especially *Rickettsiales* (Williams et al. 2007), and (2) all diverse mitochondrial genomes appear to represent reduced form of the mitochondrial genome from *Reclinomonas americana* (Lang et al. 1997). In particular, the closest phylogenetic relative for the mitochondrial genome from *R. Americana* among bacterial lineages is *Ehrlichia muris* strain AS145 within *Rickettsiales*. These lines of evidence, taken together, represent compelling evidence for the single-origin hypothesis of mitochondria.

Genetic codes also differ in start codons (Table 9.3). While AUG is used universally and dominantly as a start codon, other codons are used as well, although there has been no species in which a non-AUG codon is used as a start codon more

**Table 9.2** Codons with different meanings in different translation tables (TT)

TT	UUU	UUA	UCA	UAA	UAG	UGA	CUU	CUC	CUA	CUG	AUA	AAA	AGA	AGG
1	L	S	*	*	*	L	L	L	L	I	K	R	R	
2	L	S	*	*	W	L	L	L	M	K	*	*	*	
3	L	S	*	*	W	T	T	T	M	K	R	R	R	
4	L	S	*	*	W	L	L	L	I	K	R	R	R	
5	L	S	*	*	W	L	L	L	M	K	S	S	S	
6	L	S	Q	Q	*	L	L	L	I	K	R	R	R	
9	L	S	*	*	W	L	L	L	I	N	S	S	S	
10	L	S	*	*	C	L	L	L	I	K	R	R	R	
11	L	S	*	*	*	L	L	L	I	K	R	R	R	
12	L	S	*	*	*	L	L	L	S	I	K	R	R	
13	L	S	*	*	W	L	L	L	M	K	G	G	G	
14	L	S	Y	*	W	L	L	L	I	N	S	S	S	
16	L	S	*	L	*	L	L	L	I	K	R	R	R	
21	L	S	*	*	W	L	L	L	M	N	S	S	S	
22	L	*	*	L	*	L	L	L	I	K	R	R	R	
23	*	S	*	*	*	L	L	L	I	K	R	R	R	
24	L	S	*	*	W	L	L	L	I	K	S	K	K	
25	L	S	*	*	G	L	L	L	I	K	R	R	R	
26	L	S	*	*	*	L	L	A	I	K	R	R	R	
27	L	S	Q	Q	w	L	L	L	I	K	R	R	R	
28	L	S	q	q	w	L	L	L	I	K	R	R	R	
29	L	S	Y	Y	*	L	L	L	I	K	R	R	R	
30	L	S	E	E	*	L	L	L	I	K	R	R	R	
31	L	S	e	e	W	L	L	L	I	K	R	R	R	

A small-case letter, such as *q* in translation table 28, means that the corresponding codon can mean either amino acid Q or a stop codon



**Fig. 9.1** “Phylogenetic tree” of 24 genetic codes with their differences shown in Table 9.2, based on pairwise number of codon reassessments. A reassignment between a sense codon and a stop codon is treated as equivalent to four codon reassignment events between two nonsynonymous sense codons. Leaves labeled with a “MT”-ending are mitochondrial genetic codes

frequently than AUG. For eukaryotic species where AUG is part of translation initiation signal such as in the Kozak consensus RxxAUGG, non-AUG codons are rarely used. In bacterial species where start codon is localized by pairing of Shine-Dalgarno (SD) sequences and anti-SD sequences, the requirement for AUG as a start codon is less stringent.

A synonymous codon family refers to all codons coding the same amino acids. For example, GGA, GGC, GGG, and GGU codons all code Gly and are collectively referred to as the Gly codon family or just Gly family. I may use “family” for “synonymous codon family” when there is no confusion. A codon family such as Gly family that differs only at the third codon position is a simple family. The Gly codon family is a simple family. In contrast, a codon family that differs not only at the third codon position but also at other codon positions is a compound codon family. For example, in standard genetic code, Leu is coded by UUR (where R stands for purine) and CUN (where N stands for any nucleotide) codons. Therefore, Leu codon family is a compound family. Other compound families in the standard code include Ser (coded by UCN and AGY, where Y stands for pyrimidine) and Arg (coded by CGN and AGR). Compound families are often divided into subfamilies. For example, the Ser family is broken into UCN subfamily and AGY subfamily.

The phenomenon that one amino acid may be encoded by multiple codons is called codon degeneracy. This gives rise to 4-fold, 3-fold, 2-fold, and 1-fold (0-fold is a misnomer) degenerate sites. An  $n$ -fold site is one that can be occupied by  $n$  different nucleotides without changing the meaning of the encoded amino acid. For example, the third site in the four Gly codons above is fourfold degenerate. In the

**Table 9.3** The 24 translation tables (24) differ in start codon usage

TT	TTA	TTG	CTG	ATT	ATC	ATA	ATG	GTG
1	–	M	M	–	–	–	M	–
2	–	–	–	M	M	M	M	M
3	–	–	–	–	–	M	M	–
4	M	M	M	M	M	M	M	M
5	–	M	–	M	M	M	M	M
6	–	–	–	–	–	–	M	–
9	–	–	–	–	–	–	M	M
10	–	–	–	–	–	–	M	–
11	–	M	M	M	M	M	M	M
12	–	–	M	–	–	–	M	–
13	–	M	–	–	–	M	M	M
14	–	–	–	–	–	–	M	–
16	–	–	–	–	–	–	M	–
21	–	–	–	–	–	–	M	M
22	–	–	–	–	–	–	M	–
23	–	–	–	M	–	–	M	M
24	–	M	M	–	–	–	M	M
25	–	M	–	–	–	–	M	M
26	–	–	M	–	–	–	M	–
27	–	–	–	–	–	–	M	–
28	–	–	–	–	–	–	M	–
29	–	–	–	–	–	–	M	–
30	–	–	–	–	–	–	M	–
31	–	–	–	–	–	–	M	–

standard code, AUA, AUC, and AUU all encode amino acid Met, so that the third codon site is threefold degenerate. AAA and AAG both encode amino acid Lys, so that the third codon site is twofold degenerate. We may also have a twofold degenerate site at the first codon site. For example, both CUA and UUA encode amino acid Leu, so the first codon site is twofold degenerate. The second codon site of Gly codons is onefold degenerate because replacing it by any other nucleotide will change the meaning of the encoded amino acid.

A synonymous mutation refers to the change of a codon by another synonymous codon. A nonsynonymous mutation refers to codon replacement involving amino acid replacement. A substitution is a mutation that has spread to all individuals in the population. Synonymous substitutions occur often, but nonsynonymous substitutions occur rarely.

Throughout text, we will abbreviate highly and lowly expressed genes as HEGs and LEGs. Unless specified otherwise, HEGs and LEGs in this chapter pertain to protein expression, not mRNA expression. One may rank all proteins according to experimentally measured abundance and take the top and bottom 1/3 as HEGs and

LEGs, respectively. Non-HEGs are simply all genes from a genome that is not included in HEGs. Protein abundance values for most model species may be found in PaxDb (Wang et al. 2012).

## 1.2 *Elongation Efficiency Depends on Amino Acid and Codon Usage*

Many unicellular organisms, especially bacterial species, need to grow and replicate the cell rapidly in order not to be outcompeted by others. For example, an *E. coli* cell replicates once every 20 min with unlimited nutrients. To replicate a cell, not only the genome needs to be replicated, but a large amount of proteins have to be produced, with some proteins produced in nearly half a million copies in an *E. coli* cell. For such highly expressed proteins, it is very important for their coding genes to have efficient coding strategy to maximize the rate of translation. Translation involves three sub-processes, initiation, elongation, and termination. The previous chapter illustrates how natural selection can drive evolution toward more efficient translation initiation. This chapter addresses the question of how translation elongation can be improved through codon adaptation.

There are two obvious ways of increasing translation elongation efficiency for mass-produced proteins. The first is to optimize amino acid usage, i.e., to use energetically cheap and typically abundant amino acids as building blocks (Akashi and Gojobori 2002). The second is to maximize the usage of codons that match the anticodon of the most abundant cognate tRNA (Gouy and Gautier 1982; Ikemura 1992; Xia 1998a, 2005, 2009, 2015). For example, the amino acid glycine (Gly) can be coded by GGA, GGC, GGG, and GGU codons, but tRNA<sup>Gly</sup> species that decode GGY codons are more abundant than tRNA<sup>Gly</sup> species that decode GGR codons in *E. coli* cells. What codons should *E. coli* use to code glycine? Obviously natural selection should favor those that maximize the usage of GGY codons against GGR codons given the differential tRNA availability. However, selection and mutation may go in opposite directions, so any study of codon adaptation would be incomplete without considering both selection and mutation.

## 1.3 *Empirical Illustration of Codon-Anticodon Adaptation*

Ikemura's pioneering works established the relationship between differential tRNA abundance and its effect on codon usage in rapidly replicating bacterial species and unicellular eukaryotes (Ikemura 1981a, b, 1982, 1992). Many studies have since demonstrated a strong relationship not only between codon adaptation and gene expression (Coghlan and Wolfe 2000; Comeron and Aguade 1998; Duret and Mouchiroud 1999; Gouy and Gautier 1982; Xia 2007c) but also between

experimentally modified codon usage and protein production (Haas et al. 1996; Ngumbela et al. 2008; Robinson et al. 1984; Sorensen et al. 1989). These results have led to the explicit formulation of codon-anticodon coevolution and adaptation theory (e.g., Akashi 1994; Moriyama and Powell 1997; Ran and Higgs 2012; Xia 1998a, 2008) which states that (1) protein production is rate-limited by both translation initiation and elongation efficiency; (2) codon usage and tRNA anticodon coevolve to adapt to each other, resulting in increased production of correctly translated proteins; and (3) the increased elongation efficiency and accuracy represent the driving force for the HEGs to acquire a high degree of codon-anticodon adaptation.

### 1.3.1 Empirical Illustration of Codon-Anticodon Adaptation in Yeast

The baker's yeast, *Saccharomyces cerevisiae*, replicates rapidly and is expected to use codons with many decoding tRNAs and avoid codons with few decoding tRNAs. The earliest association between tRNA and codon usage was empirically demonstrated by Ikemura (1981a, b, 1992). Tables 9.4 and 9.5 show the association between tRNA gene copy number (T in Tables 9.4 and 9.5) in the genome and codon usage in highly expressed yeast genes (F in Tables 9.4 and 9.5). T is a good proxy for tRNA abundance (Percudani et al. 1997).

The association between T and F is obvious in Tables 9.4 and 9.5. Take the two Arg codons AGA and AGG in Table 9.4, for example. There are 11 tRNA<sup>Arg/UCU</sup> genes in the yeast genome that form perfect Watson-Crick base pair with AGA but

**Table 9.4** Copy number of tRNA genes in the yeast *Saccharomyces cerevisiae* genome (T) and codon counts (F) in highly expressed yeast protein-coding genes, compiled in the Eyeastcai.cut file distributed with EMBOSS (Rice et al. 2000)

AA <sup>a</sup>	Codon <sup>b</sup>	T	F	AA <sup>a</sup>	Codon <sup>b</sup>	T	F
Arg	AGA	11	314	His	CAC	7	102
Arg	AGG	1	1	His	CAU	0	25
Asn	AAC	10	208	Leu	UUA	7	42
Asn	AAU	0	11	Leu	UUG	10	359
Asp	GAC	16	202	Lys	AAA	7	65
Asp	GAU	0	112	Lys	AAG	14	483
Cys	UGC	4	3	Phe	UUC	10	168
Cys	UGU	0	39	Phe	UUU	0	19
Gln	CAA	9	153	Ser	AGC	2	6
Gln	CAG	1	1	Ser	AGU	0	4
Glu	GAA	14	305	Tyr	UAC	8	141
Glu	GAG	2	5	Tyr	UAU	0	10

Only twofold codon families are included

<sup>a</sup>Amino acid carried by tRNA

<sup>b</sup>Codons forming Watson-Crick base pair with the anticodon of tRNA

**Table 9.5** Copy number of tRNA genes in the yeast *Saccharomyces cerevisiae* genome (T) and codon counts (F) in highly expressed yeast protein-coding genes, compiled in the Eyeastcai.cut file distributed with EMBOSS (Rice et al. 2000)

AA	Codon	T	F	AA	Codon	T	F
Ala	GCA	5	6	Pro	CCA	10	211
Ala	GCG	0	0	Pro	CCG	0	0
Ala	GCC	0	130	Pro	CCC	0	2
Ala	GCU	11	411	Pro	CCU	2	10
Arg	CGA	0	0	Ser	UCA	3	7
Arg	CGG	1	0	Ser	UCG	1	1
Arg	CGC	0	0	Ser	UCC	0	133
Arg	CGU	6	43	Ser	UCU	11	192
Gly	GGA	3	1	Thr	ACA	4	2
Gly	GGG	2	2	Thr	ACG	1	1
Gly	GGC	16	9	Thr	ACC	0	164
Gly	GGU	0	459	Thr	ACU	11	151
Ile	AUA	2	0	Val	GUA	2	0
Ile	AUC	0	181	Val	GUG	2	5
Ile	AUU	13	149	Val	GUC	0	231
Leu	CUA	3	14	Val	GUU	14	278
Leu	CUG	0	1				
Leu	CUC	1	1				
Leu	CUU	0	2				

Only threefold and fourfold codon families are included. Symbols as in Table 9.4

only one tRNA<sup>Arg/CCU</sup> with AGG. So we expect yeast genes, especially highly expressed ones, to use AGA and avoid AGG, which is true (Table 9.4). The same applies to all other synonymous codon families or subfamilies, except for the Cys codon family. Why the rarely used Cys codon family should be exceptional remains unknown. It is possible that Cys codon UGC may happen to be followed by a GNN codon, leading to methylation of C at the third codon position which then changes to T via spontaneous deamination. Whether the yeast genome has cytosine methylation remains controversial, with both evidence for (Tang et al. 2012) and against (Capuano et al. 2014) the existence of methylation in *S. cerevisiae*. However, there is significant CpG deficiency and TpG and CpA surplus in genome, which is consistent with CpG-specific DNA methylation.

One can obtain tables similar to Tables 9.4 and 9.5 by downloading the yeast genome from GenBank and then using DAMBE to compile the data in three steps. First, read the GenBank files for yeast chromosome sequences into DAMBE (Xia 2013, 2017d) to extract the coding sequences (CDSs) and tRNA genes. Second, compute  $I_{TE}$  (Xia 2015) as a proxy of gene expression, and choose a subset of CDSs with highest  $I_{TE}$  as HEGs. Third, use DAMBE to obtain codon usage of these HEGs. In this way, a table similar to Table 9.4 can be generated in minutes.

### 1.3.2 Codon Usage Changes When tRNA Abundance Changes

An evolutionary change in tRNA composition or relative abundance is expected to alter codon-anticodon adaptation. This is not controversial theoretically, but empirically difficult to demonstrate. However, recent studies (Xia 2012c; Xia et al. 2007) have documented that changes in tRNA<sup>Met</sup> genes (where Met is the amino acid carried by the tRNA) in animal mitochondrial DNA (mtDNA) are associated with changes in Met codon usage.

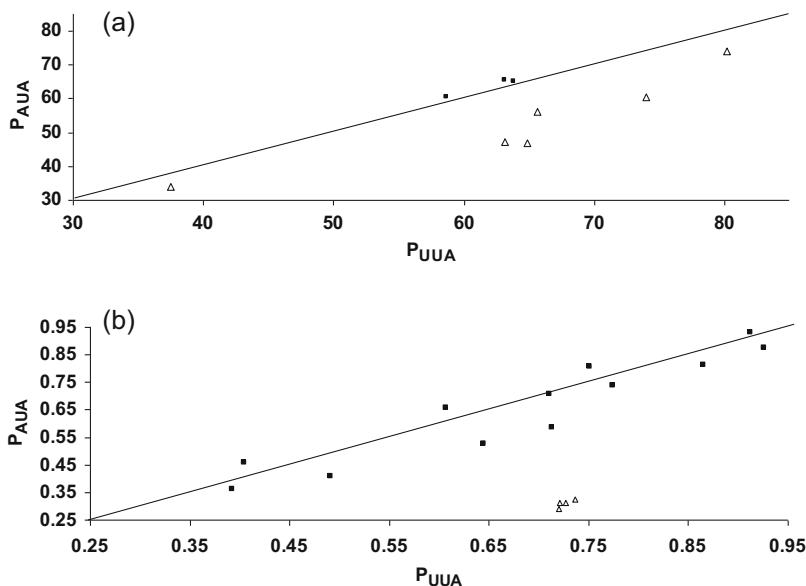
In mtDNA of most animal species, Met is coded by AUA and AUG codons. In some animal species, e.g., vertebrates, these two codons are translated by a single tRNA<sup>Met/CAU</sup> species (where CAU is the anticodon in the 5' to 3' orientation) with a modified C (i.e., f<sup>5</sup>C) at the first anticodon position (Grosjean et al. 2010) to allow C/A pairing. In other animal species, e.g., tunicates, an additional tRNA<sup>Met/UAU</sup> gene is present in the mtDNA. One would expect that, when tRNA<sup>Met/UAU</sup> is absent, Met should be preferably coded by AUG with a reduced AUA usage. The gain of tRNA<sup>Met/UAU</sup> would favor more Met to be coded by AUA.

In addition to tunicates, MtDNA in bivalve species also have two tRNA<sup>Met</sup> genes. In some bivalve species (e.g., *Acanthocardia tuberculata*, *Crassostrea gigas*, *C. virginica*, *Hiatella arctica*, *Placopecten magellanicus*, and *Venerupis philippinarum*), both tRNA<sup>Met</sup> genes have a CAU anticodon forming Watson-Crick base pair with codon AUG. In some other bivalve species (e.g., *Mytilus edulis*, *Mytilus galloprovincialis*, and *Mytilus trossulus*), one tRNA<sup>Met</sup> has a CAU anticodon, and the other has a UAU anticodon forming Watson-Crick base pair with the AUA codon. One would predict that the latter should be more likely to code Met by AUA than the former, i.e., the proportion of AUA codon within the AUR codon family, designated P<sub>AUA</sub>, should be greater in the latter with both a tRNA<sup>Met/CAU</sup> and a tRNA<sup>Met/UAU</sup> gene than in the former with tRNA<sup>Met/CAU</sup> gene only (Xia et al. 2007).

One complication in testing the prediction is that AUA usage will increase with genomic AT%. To control for this effect, one may use another A-ending codon, such as UUA as a reference. Thus, given the same P<sub>UUA</sub> (the proportion of UUA codon in the UUR codon family), P<sub>AUA</sub> in the three *Mytilus* mtDNA with both a tRNA<sup>Met/CAU</sup> and a tRNA<sup>Met/UAU</sup> gene should be higher than that in the six bivalve species without a tRNA<sup>Met/UAU</sup> gene. This is supported by empirical evidence (ANCOVA test,  $p = 0.0111$ , Fig. 9.2a). Thus, the presence of tRNA<sup>Met/UAU</sup> increases AUA usage significantly.

A similar comparison can be performed between the urochordates (tunicates, with both tRNA<sup>Met/CAU</sup> and tRNA<sup>Met/UAU</sup> genes in their mtDNA) and cephalochordates (lancelets, with only a tRNA<sup>Met/CAU</sup> gene in their mtDNA). Figure 9.2b shows that P<sub>AUA</sub> is much smaller in lancelets than in tunicates at the same P<sub>UUA</sub> level. Thus, AUA usage is consistently increased by the gain of a tRNA<sup>Met/UAU</sup> gene (or consistently decreased by the loss of a tRNA<sup>Met/UAU</sup> gene) in animal mtDNA.

A gain of a tRNA<sup>Met/UAU</sup> gene is also associated with a surplus of AUG→AUA substitutions in animal mitochondrial coding sequences (results not shown). Similar associations can also be observed with other gain/loss of tRNA genes in animal

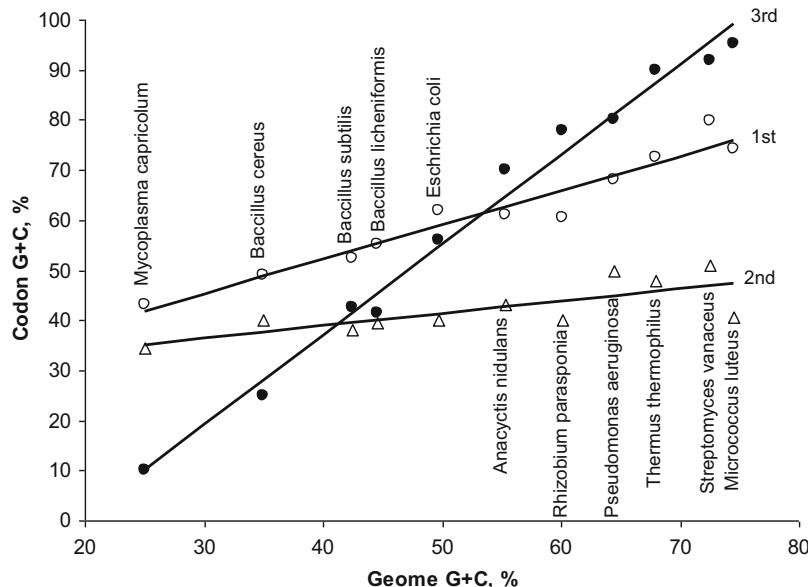


**Fig. 9.2** Relationship between PAUA and PUUA, highlighting the observation that PAUA is greater when both a tRNA<sup>Met/CAU</sup> and a tRNA<sup>Met/UAU</sup> are present than when only tRNA<sup>Met/CAU</sup> is present in the mtDNA, for bivalve species (a) and chordate species (b). The filled squares are for mtDNA containing both tRNA<sup>Met/CAU</sup> and tRNA<sup>Met/UAU</sup> genes, and the open triangles are for mtDNA without a tRNA<sup>Met/UAU</sup> gene

mitochondrial. In contrast, a gain/loss of tRNA genes in plant mtDNA appears to have little effect on nucleotide substitutions or codon usage, presumably because such gain/loss events do not significantly alter the tRNA pool in plant cells where nuclear tRNAs are mass-imported into plant mitochondria.

#### 1.4 Effect of Biased Mutation on Codon Usage and Some Misconceptions

Biased mutation has long been known to affect codon usage (Muto and Osawa 1987; Sueoka 1964; Xia and Yuen 2005; Xia et al. 2002). The third codon position is the most amenable to mutation bias (Fig. 9.4) because most nucleotide substitutions at the third codon position are synonymous. Nucleotide substitutions are synonymous at some first codon positions but nonsynonymous at all second codon position. Furthermore, all nucleotide substitutions at the second codon positions typically involve rather different amino acids and therefore should be subject to strong purifying selection (Xia 1998b; Xia and Li 1998). One therefore would predict that the third codon position should increase more rapidly with the genomic GC%



**Fig. 9.3** Correlation of GC% between genomic DNA and first, second, and third codon positions (Muto and Osawa 1987). While the actual position of the points may be substantially revised with new genomic data (e.g., the GC% for the first, second, and third codon positions for *Mycoplasma capricolum* is 35.8%, 27.4%, and 8.8% based on all annotated CDSs in the genomic sequence), the general trend remains the same

than the first codon position which in turn should have its GC% increase more rapidly with the genomic GC% than the second codon position. The empirical results (Fig. 9.3) strongly support the prediction (Muto and Osawa 1987).

However, the pattern in Fig. 9.3, while consistent with the mutation hypothesis, has resulted in two misconceptions. First, the pattern shown by the third codon position is often interpreted to reflect mutation bias. This interpretation is incorrect because the third codon position is subject to selection by differential availability of tRNA species (Carullo and Xia 2008; Xia 1998a, 2005, 2008; Xia et al. 2007). We may contrast a GC-rich *Streptomyces coelicolor* and a GC-poor *Mycoplasma capricolum* as an illustrative example. *M. capricolum* has no tRNA with a C or G at the wobble site for fourfold codon families (Ala, Gly, Pro, Thr, and Val), i.e., the translation machinery would be inefficient in translating C-ending or G-ending codons. This implies selection in favor of A-ending or U-ending codons and will consequently reduce GC% at the third codon position. This most likely has contributed to the low GC% at the third codon position in *M. capricolum*. In contrast, most of the tRNA genes translating the five fourfold codon families in the GC-rich *S. coelicolor* have G or C at the wobble site, and should favor the use of C-ending or G-ending codons. This most likely has contributed to the high GC% at the third codon position in *S. coelicolor*. In these two cases, mutation bias and tRNA-mediated

selection are in the same direction to drive up or down GC% at the third codon position. The same pattern is observed for twofold codon families. The most conspicuous one is the Gln codon family (CAA and CAG). There is only one tRNA<sup>Gln</sup> gene in *M. capricolum* with a UUG anticodon favoring the CAA codon. In contrast, there are two tRNA<sup>Gln</sup> in *S. coelicolor*, both with a CUG anticodon favoring the CAG codon. Thus, the high slope for the third codon position in Fig. 9.3 is at least partially attributable to the tRNA-mediated selection. Relative contribution of mutation and tRNA-mediated selection to codon usage has been evaluated in several recent studies (Carullo and Xia 2008; Xia 2005, 2008; Xia et al. 2007).

The second misconception arising from Fig. 9.3 is that the frequency of G-ending and C-ending codons will increase and A-ending and U-ending codons decrease, with genomic GC% or GC-biased mutation (Kliman and Bernal 2005). This is not generally true (Palidwor et al. 2010). Take the arginine codons, for example. Given the transition probability matrix for the six synonymous codons shown in Table 9.6, the equilibrium frequencies ( $\pi$ ) for the six codons are

$$\begin{aligned}\pi_{AGA} &= \frac{1}{2k^2 + 3k + 1} \\ \pi_{AGG} = \pi_{CGA} = \pi_{CGT} &= \frac{k}{2k^2 + 3k + 1} \\ \pi_{CGC} = \pi_{CGG} &= \frac{k^2}{2k^2 + 3k + 1}\end{aligned}\tag{9.1}$$

The three solutions correspond to the number of GC in the codon, with AGA having one, AGG, CGA and CGT having two, and CGC and CGG having three G or C. One may note that the G-ending codon AGG has the same equilibrium frequency as that of the A-ending CGA and the T-ending CGT. Thus, we should not expect A-ending or T-ending codons to always decrease or G-ending and C-ending codons always increase, with increasing genomic GC% or GC-biased mutation. In fact, according to the solutions in Eq. (9.1),  $\pi_{AGG}$ ,  $\pi_{CGA}$ , and  $\pi_{CGT}$  will first increase with  $k$  until  $k$  reaches  $\sqrt{2}/2$  and will then decrease with  $k$  when  $k > \sqrt{2}/2$  (Palidwor et al. 2010).

## 1.5 Two Hypotheses on Translation Elongation Efficiency

It is controversial as to what degree is protein production limited by translation elongation. Early theoretical considerations (Andersson and Kurland 1983; Bulmer 1990, 1991; Liljenstrom and von Heijne 1987) tend to favor the argument that translation elongation is not rate-limiting in protein production, but translation initiation is. This hypothesis does not deny the existence of codon adaptation, but it asserts that codon-anticodon adaptation and increased elongation efficiency are not related to protein production. Instead, the benefit of codon adaptation and increased elongation efficiency is to increase ribosomal availability for global translation. This

**Table 9.6** Transition probability matrix for the six synonymous arginine codons, with  $\alpha$  for transitions ( $C \leftrightarrow T$  and  $A \leftrightarrow G$ ),  $\beta$  for transversions, and  $k$  modeling AT-biased mutation ( $0 \leq k \leq 1$ ) or GC-biased mutation ( $k > 1$ )

	CGT	CGC	CGA	CGG	AGA	AGG
CGT		$k\alpha$	$\beta$	$k\beta$	0	0
CGC	$\alpha$		$\beta$	$\beta$	0	0
CGA	$\beta$	$k\beta$		$k\alpha$	$\beta$	0
CGG	$\beta$	$\beta$	$\alpha$		0	$\beta$
AGA	0	0	$k\beta$	0		$k\alpha$
AGG	0	0	0	$k\beta$	$\alpha$	

We ignore nonsynonymous substitutions because nonsynonymous substitution rate is often negligibly low compared to synonymous rate. The diagonal is constrained by the row sum equal to 1

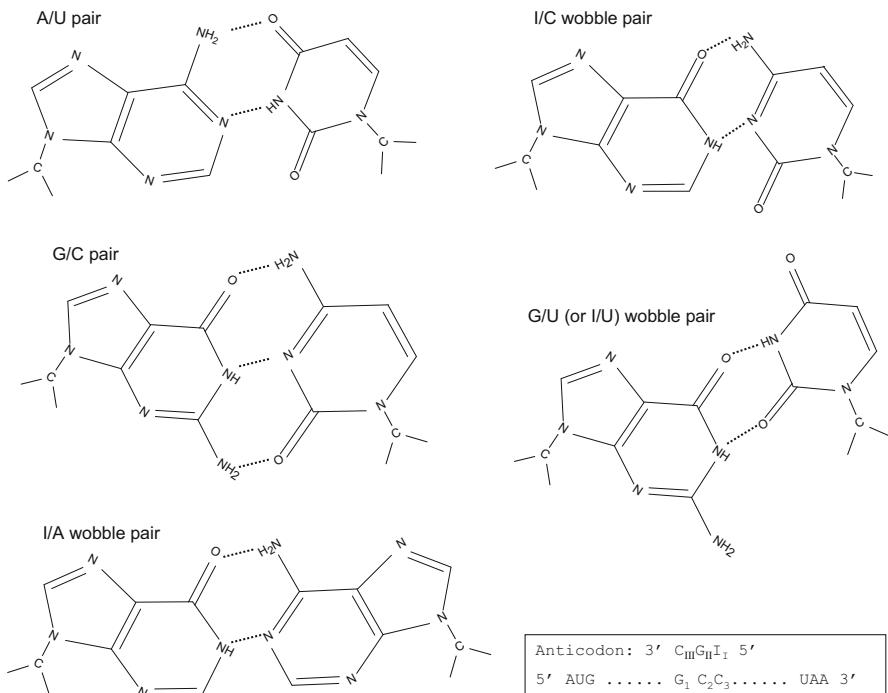
hypothesis was explicitly formulated only recently and empirically tested (Kudla et al. 2009).

We thus have two alternative hypotheses attributing different benefits to codon-anticodon adaptation. The first assumes that protein production is rate-limited by both initiation and elongation and codon-anticodon adaptation would result in higher elongation efficiency and more efficient and accurate protein production, especially for HEGs. The second claims that protein production is rate-limited only by initiation efficiency but improved codon adaptation and consequently increased elongation efficiency have the benefit of increasing ribosomal availability for global translation.

How should we go about testing these two hypotheses? Note that the two hypotheses make different predictions about the relationship among three variables: (1) translation initiation efficiency, (2) translation elongation efficiency, and (3) protein production. Before we can test these two hypotheses, we need to understand how these variables can be measured. The previous chapter outlines a few factors contributing to translation initiation efficiency. Here we first learn a few indices of codon usage bias as a proxy for translation elongation efficiency and then include them in the test of the two hypotheses in the section illustrating the application of index of translation elongation (Xia 2015).

## 1.6 Wobble Hypothesis and Its Extensions

The wobble hypothesis is proposed to explain how a set of tRNA molecules can decode all sense codons which are much larger in number. The wobble-pairing rules are specified in Fig. 9.4, together with the numbering system used here for individual codon and anticodon sites that is more precise than, but different from, the conventional one. The original wobble hypothesis (Crick 1966), with its extended codon-anticodon base pairs (Fig. 9.4), played a crucial role in understanding the working of the translation machinery. It explains why tRNA<sup>Ile/IAU</sup>, where I in IAU is inosine derived from A, is able to translate all three Ile codons (AUC, AUU, and, albeit



**Fig. 9.4** Base pairs between nucleotides at the first anticodon site (which can have I, G, C, U but rarely A) and the third codon site. The inset shows the site numbering system of codon and anticodon, with codon sites subscripted with 1, 2 and 3 and anticodon sites subscripted with I, II, and III, which is illustrated by the paring of II/C3, GII/C2, CIII/G1.

inefficiently, AUA), why a tRNA with a G<sub>I</sub> can translate Y-ending codons (where Y stands for C or U), and why a tRNA with a U<sub>I</sub> can translate R-ending codons (where R stands for A or G). The hypothesis also explains the lack of A<sub>I</sub> in tRNA genes for decoding twofold Y-ending codon family because such a tRNA, when its A<sub>I</sub> is modified to I<sub>I</sub>, would misread the near-cognate R-ending codons.

Wobble pairing reduces the number of tRNAs needed for translation and simplifies the translation machinery. As an example of parsimonious tRNA usage, the Y-ending codons, be they in twofold or fourfold codon families, are decoded by tRNAs with either a I<sub>I</sub> or a G<sub>I</sub>, but never both. This rule is obeyed in all three kingdoms of life. Almost all fourfold codon families in *Mycoplasma pulmonis* (including the Ser UCN codon family and Leu CUN codon family) are decoded by a single tRNA species with a U<sub>I</sub>, except for the Thr ACN and Arg CGN codon families which are each decoded by two tRNA species, one with a U<sub>I</sub> and other with a G<sub>I</sub>. The most dramatic simplification of tRNome is observed in vertebrate mitochondria, e.g., vertebrate mitochondrial genomes which contain only 22 tRNA genes, with each tRNA species decoding a codon family. Instead of separate initiation tRNA<sup>iMet/CAU</sup> and elongation tRNA<sup>eMet/CAU</sup> present in all nuclear genomes, a single

tRNA<sup>Met/CAU</sup>, with a modified C<sub>I</sub>, decodes both the initiation AUG codon and internal Met AUR codons. Each Y-ending codon family is decoded by a single tRNA species with a wobble G<sub>I</sub> and each R-ending codon family by a single tRNA with a wobble U<sub>I</sub> which is modified to prevent its pairing with U or C. All fourfold codon families are decoded by a tRNA with a wobble U<sub>I</sub> which is not modified.

Wobble pairing is not without cost as it often reduces translation efficiency and accuracy and is generally avoided (Xia 2008). For example, an I<sub>I</sub>/A<sub>3</sub> pair is bulky because it involves two purines (Fig. 9.4) in contrast to other base pairs which typically involve a large purine and a small pyrimidine. For this reason, Ile is rarely coded by AUA except for certain viruses with a strong A-biased mutation (van Weringh et al. 2011). Among a set of highly expressed genes in the yeast (*Saccharomyces cerevisiae*), AUA is not used at all (Table 9.5). Similarly, a tRNA with a U<sub>I</sub> can translate A-ending codons better than G-ending codons (Grosjean et al. 2010; Xia 2008). Most of the yeast tRNA<sup>Arg</sup> have a U<sub>I</sub>, and only one AGG codon is found in contrast to 314 AGA codons in highly expressed yeast genes (Table 9.4). Yeast genomic data also suggest that a tRNA with a G<sub>I</sub> can translate C-ending codons better than U-ending codons. For example, the yeast tRNA<sup>Asn</sup> genes translating the Asn AAY codon family all have a G<sub>I</sub>. Among 219 Asn codons in highly expressed yeast genes, only 11 are AAU codons, suggesting strong selection against AAU codons in favor of AAC codons (Table 9.4). Note that the yeast genome is strongly AT-biased. If there is no selection against AAU codons, we would expect more AAU codons than AAC codons, which is contrary to the observed frequencies. However, the selection against G<sub>I</sub>/U<sub>3</sub> pair is in general much weaker than that against U<sub>I</sub>/G<sub>3</sub> pair. In fungal mitochondrial genomes, there is no avoidance of G<sub>I</sub>/U<sub>3</sub> pair in favor of G<sub>I</sub>/C<sub>3</sub> pair, although U<sub>I</sub>/G<sub>3</sub> pair is strongly avoided in favor of U<sub>I</sub>/A<sub>3</sub> pair (Xia 2008). The weak, or lack of, selection against G<sub>I</sub>/U<sub>3</sub> can explain several puzzling counterexamples against the codon-anticodon adaptation theory (Bulmer 1991; Ikemura 1981b; Xia 1998a) which states that the most frequently used codon in each synonymous codon family should form Watson-Crick base paring with the anticodon of the most abundant tRNA species to reduce translation error and increase translation efficiency. For example, Cys codons (UGY) are translated by tRNA<sup>Cys/GCA</sup> in both cytoplasm and mitochondria in the yeast, yet most Cys codons have U<sub>3</sub>. If there is little selection against G<sub>I</sub>/U<sub>3</sub> pair (i.e., G<sub>I</sub>/U<sub>3</sub> pair is as efficient and accurate as G<sub>I</sub>/C<sub>3</sub> pair), then the frequencies of UGC and UGU will be mostly determined by AT-bias. Because the yeast nuclear and mitochondrial genomes are both AT-rich, we have more UGU codons than UGC codons, in spite of G<sub>I</sub> in tRNA<sup>Cys</sup>. The weak selection against G<sub>I</sub>/U<sub>3</sub> but strong selection against U<sub>I</sub>/G<sub>3</sub> also explains why Y-ending codons are typically translated by a tRNA with a G<sub>I</sub>, whereas R-ending codons are typically translated by two different tRNAs, one with a U<sub>I</sub> and the other with a C<sub>I</sub> (Xia 2008).

The wobble hypothesis points to the necessity of nucleotide modification in tRNA to either increase or decrease the wobble versatility to improve accuracy and efficiency of translation. The observation that an unmodified U<sub>I</sub> can pair with all N<sub>3</sub> in many mitochondrial genomes suggests that U<sub>I</sub> in tRNA for twofold R-ending codon families needs to be modified to restrict its wobble versatility to

avoid misreading the near-cognate Y-ending codons. Chemical modification of U<sub>I</sub> to restrict its pair versatility to R<sub>3</sub> in twofold R-ending codon family is universal in all three kingdoms of life and in organelles (Grosjean et al. 2010; Lim 1994). On the other hand, the tRNA<sup>Met/CAU</sup> in vertebrate mitochondria need to read both the initiation AUG codon and the internal AUG and AUA codons, and its C<sub>I</sub> is modified to f<sup>5</sup>C<sub>I</sub> to increase its wobble versatility so as to form a f<sup>5</sup>C<sub>I</sub>/A<sub>3</sub> pairing between the anticodon and the AUA codon. Nucleotide modification in tRNA has been extensively reviewed (Grosjean et al. 2010) and chemically detailed in MODOMICS (Czerwoniec et al. 2009).

Wobble pairing implies the theoretical possibility of adding new base pairs of novel nucleotides to protein-coding genes to increase the coding capacity (Hirao and Kimoto 2010). A single novel base pair, involving two novel nucleotides, would increase the number of codons from 64 to 216 (=6<sup>3</sup>), and one can then use these extra codons, together with engineered tRNAs to recognize these codons and to carry new amino acid analogs, to produce novel proteins.

The wobble hypothesis can be extended to explain the lack of UCG anticodon in Arg CGN codon family in a large number of evolutionary lineages. A tRNA species with a wobble U<sub>I</sub> is almost always present among tRNA species decoding fourfold codon families and twofold R-ending codon families, with most exceptions observed in the Arg CGN codon family. In the mitochondrial genomes of *Caenorhabditis elegans* (metazoan), *Marchantia polymorpha* (plant), *Pichia canadensis* (fungus), and *Saccharomyces cerevisiae* (fungus), there is no tRNA<sup>Arg/UCG</sup>, and Arg CGN codon family is decoded by tRNA<sup>Arg/ACG</sup> (Xia 2005). The lack of tRNA<sup>Arg/UCG</sup> in the mitochondrial genome of these diverse taxa suggests that the lack is an ancestral state and that the presence of tRNA<sup>Arg/UCG</sup> in vertebrate mitochondria is a derived state. This is substantiated by the fact that almost all eubacterial species, from which the mitochondrion was originally derived, lack tRNA<sup>Arg/UCG</sup> (Grosjean et al. 2010).

The expanded wobble hypothesis for the lack of tRNA<sup>Arg/UCG</sup> requires an extension of the wobble hypothesis by invoking wobble paring between the third anticodon site (N<sub>III</sub>) and the first codon site (N<sub>1</sub>), conditional on a C<sub>II</sub>/G<sub>2</sub> or G<sub>II</sub>/C<sub>2</sub> with three hydrogen bonds. Thus, the anticodon UCG would wobble-pair with stop codon UGA through a wobble G<sub>III</sub>/U<sub>1</sub> pair and should therefore be strongly selected against (Carullo and Xia 2008). This explains not only the absence of tRNA<sup>Arg/UCG</sup> in diverse evolutionary lineages but in particular why tRNA<sup>Arg/UCG</sup> is absent in most eubacterial species and ancestral mitochondrial lineages where UGA is used as a stop codon and why it is present in derived mitochondrial lineages such as vertebrate mitochondrial genomes where UGA is no longer used as a stop codon.

## 2 Commonly Used Codon Usage Indices

There are two key factors contributing to codon usage bias: the mutation bias (Osawa et al. 1987) and the tRNA-mediated selection (Ikemura 1981a, 1982, 1992; Xia 1998a, 2015). There are also two types of codon usage indices, but they do not

correspond to the two factors shaping codon usage. The first type of codon usage indices is codon-specific best represented by relative synonymous codon usage (RSCU, Sharp et al. 1986), which measures deviation of codon usage from equal usage. The second type of codon usage indices is gene-specific with several well-known representatives including codon adaptation index effective number of codons (ENC, Sun et al. 2013; Wright 1990), codon adaptation index (CAI, Sharp and Li 1987; Xia 2007c), codon bias index (CBI, Bennetzen and Hall 1982), frequency of optimal codons ( $F_{op}$ , Ikemura 1985), tRNA adaptation index (tAI, dos Reis et al. 2004), and index of translation elongation ( $I_{TE}$ , Xia 2015).

ENC aims to measure deviation of codon usage from equal usage and may be considered as the gene-specific equivalent of the codon-specific RSCU. They are both descriptive and do not distinguish between mutation bias or tRNA-mediated selection in their contribution to codon usage bias. All other gene-specific indices aim to measure the intensity of the tRNA-mediated selection on codon usage bias. A gene encoding a mass-produced (highly expressed) protein is expected to be under stronger selection to optimize its codon usage corresponding to differential tRNA availability than a gene encoding lowly expressed protein, and we expect CAI, CBI, tAI, and  $I_{TE}$  to be greater for the highly expressed gene than the lowly expressed gene. However, CAI, CBI, and tAI ignore background mutation bias.  $I_{TE}$  is a generalization of CAI, by incorporating background mutation, and is reduced to CAI when there is no background mutation bias (Xia 2015).

Codon indices that aim to measure tRNA-mediated selection (i.e., CAI, CBI,  $F_{op}$ , tAI, and  $I_{TE}$ ) all define a translationally optimal codon (TOC) within each codon family, and the codon usage index value will be the highest if all codons in a gene are TOCs. However, TOC is defined differently among these indices. CBI,  $F_{op}$ , and tRNA define a TOC mainly as one that corresponds to the most abundant isoacceptor tRNA, with CBI incorporating gene expression information as well. CAI defines a TOC as one in its codon family that is used most frequently in HEGs.  $I_{TE}$  defines a TOC as one in its codon family that is used most frequently in HEGs after adjustment of mutation bias reflected in LEGs. Comparative studies (Coghlan and Wolfe 2000; Comeron and Aguade 1998) suggest that CAI is better than ENC, CBI, and  $F_{op}$  in predicting gene expression levels, tAI is better than CAI (dos Reis et al. 2004; Tuller et al. 2010), and  $I_{TE}$  is better than CAI and tAI (Xia 2015). However, such comparison depends not only on the methods but also on the quality of the software that implements the methods. A good method could be conceptually sound but implemented erroneously and generate poor results. Moreover, the same index could be implemented differently. For example, one implementation could treat all synonymous codons into one family so that some codons could have six or even eight synonymous codons (trematode mitochondrial code has eight Ser codons: UCN and AGN), whereas another implementation would break all compound codon families, such as Leu, Ser, and Arg codon families, into separate fourfold and twofold codon families.

## 2.1 RSCU (Relative Synonymous Codon Usage)

RSCU measures codon usage bias for each codon within each codon family. It is essentially a normalized codon frequency so that the expectation is 1 when there is no codon usage bias. A codon is overused if its RSCU value is greater than 1 and underused if its RSCU value is less than 1. It is computed directly from input sequences.

### 2.1.1 Calculation of RSCU

The general equation for computing RSCU is

$$RSCU_{ij} = \frac{\text{CodFreq}_j}{\left( \sum_{j=1}^{\text{NumCodon}_i} \text{CodFreq}_j \right) / \text{NumCodon}_i} \quad (9.2)$$

where  $i$  refers to a codon family and  $j$  to a specific codon within the family. For example,  $i$  may refer to the alanine codon family with four codons (GCU, GCC, GCA, and GCG) and  $j$  to a specific codon such as GCU. In this case, the numerator is the frequency of GCU, and the denominator is the summation of the four codon frequencies divided by the number of codons in the codon family, i.e., 4.

For biology students, it is always easier to learn by numerical examples. Suppose we counted the codon frequencies of one particular protein-coding sequence and have obtained the codon frequencies (Table 9.7). The RSCU for the GCU codon is computed, according to Eq. (9.2), as

$$RSCU_{GCU} = \frac{52}{\frac{(52+91+103+2)}{4}} = 0.84 \quad (9.3)$$

which is displayed in Table 9.7. Biology students are recommended to cover up the last column in Table 9.7 and finish the computation of the rest of the RSCU values.

**Table 9.7** Data for illustrating the calculation of RSCU

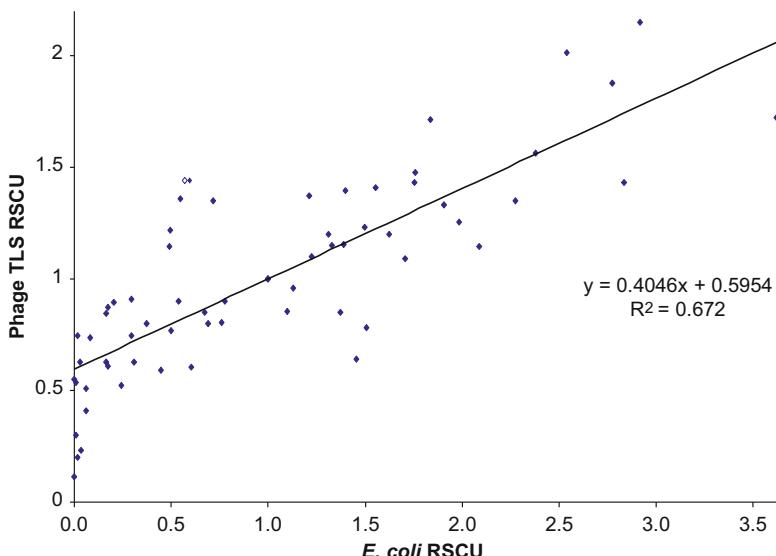
Codon	AA	N	RSCU
GCU	Ala	52	0.84
GCC	Ala	91	1.47
GCA	Ala	103	1.66
GCG	Ala	2	0.03
GAA	Glu	78	1.64
GAG	Glu	17	0.36
...	...	...	...

AA amino acid,  $T$  codon frequency

### 2.1.2 Illustration of RSCU Applications

As I mentioned earlier, a variable such as RSCU is often not interesting by itself, but it becomes more interesting when you relate the variable to some other variables. Figure 9.5 shows the correlation of RSCU for *Escherichia coli* genes and that for the *E. coli* double-stranded DNA (dsDNA) phage TLS. This strong and positive correlation suggests adaptation of host tRNA pool. This adaptation of the phage genes and the host genes to the same tRNA pool in *E. coli* cells and the evolution of the very similar codon usage patterns is an example of convergent evolution, i.e., phylogenetically remote organisms evolving similar features not due to coancestry, but in response to the same selection regime induced by the same environment.

What explanation would you offer if we find little correlation in RSCU between a phage and its host? There are in fact a large number of cases in which a virus and its host share little similarity in codon usage. Will such cases invalidate our convergent evolution explanation for the strong and positive correlation between phage TLS and its *E. coli* host? Science thrives in questions, and such questions immediately drive us to search for answers, and the answers enrich our explanatory conceptual framework. Ronald Fisher once said that “No aphorism is more frequently repeated in connection with field trials, than that we must ask Nature few questions, or ideally, one question at a time. The writer is convinced that this view is wholly mistaken. Nature, he suggests, will respond to a logical and carefully thought-out questionnaire; indeed, if we ask her a single question, she will often refuse to answer until some other topic has been discussed” (Fisher 1926).



**Fig. 9.5** Correlation in RSCU between *Escherichia coli* and its double-stranded DNA phage TLS

There are at least six factors that will weaken the correlation in RSCU between a virus and its host. First, some dsDNA phages carry many tRNA genes of their own genome, and the transcription of these tRNA genes would modify the host tRNA pool. For example, another dsDNA *E. coli* phage, enterobacteria phage WV8, carries 20 tRNA genes on its genome. In such cases, the phage genes would adapt to the modified tRNA pool which may be different from the tRNA pool where *E. coli* mRNAs are translated normally (i.e., without phage infection). Partly for this reason, the correlation in RSCU between enterobacteria phage WV8 and its *E. coli* host is much weaker than that shown in Fig. 9.5 (Chithambaram et al. 2014a). Phage TLS (Fig. 9.5) happens to have a genome that does not encode any tRNA genes of its own. So it depends entirely on the host tRNA pool to decode the codons of its genes.

Second, codon usage adaptation takes time. If a phage having adapted to one host has switched to a new host, and if the original host and the new host differ in their tRNA pools, then the phage codon usage will be more similar to that of the original host than the new host. This may be applicable to phage PRD1 which belongs to the peculiar *Tectiviridae* family with members parasitizing both gram-negative and gram-positive bacteria. Phage PRD1 is the only species in the family known to parasitize gram-negative bacteria, with other members of the family, i.e., phages PR3, PR4, PR5, L17, and PR772, parasitizing gram-positive bacteria (Bamford et al. 1995; Grahn et al. 2006). It is reasonably safe to assume that the phage PRD1 lineage has switched host from gram-positive to gram-negative bacteria. Furthermore, there is only one amino acid difference in the coat protein between phages PRD1 and PR4 (Bamford et al. 1995). This suggests that PRD1 is phylogenetically close to its relative parasitizing gram-positive, i.e., the host-switching may have occurred quite recently. In fact, codon usage in phage PRD1 is more similar to that in gram-positive bacteria than in gram-negative bacteria (Chithambaram et al. 2014b). Among 87 bacterial genomes covering major groups of bacterial species, the host species with codon usage most similar to that of phage PRD1 are strains in the gram-positive *Geobacillus* (NC\_014206, NC\_012793, NC\_014650, NC\_014915, NC\_013411).

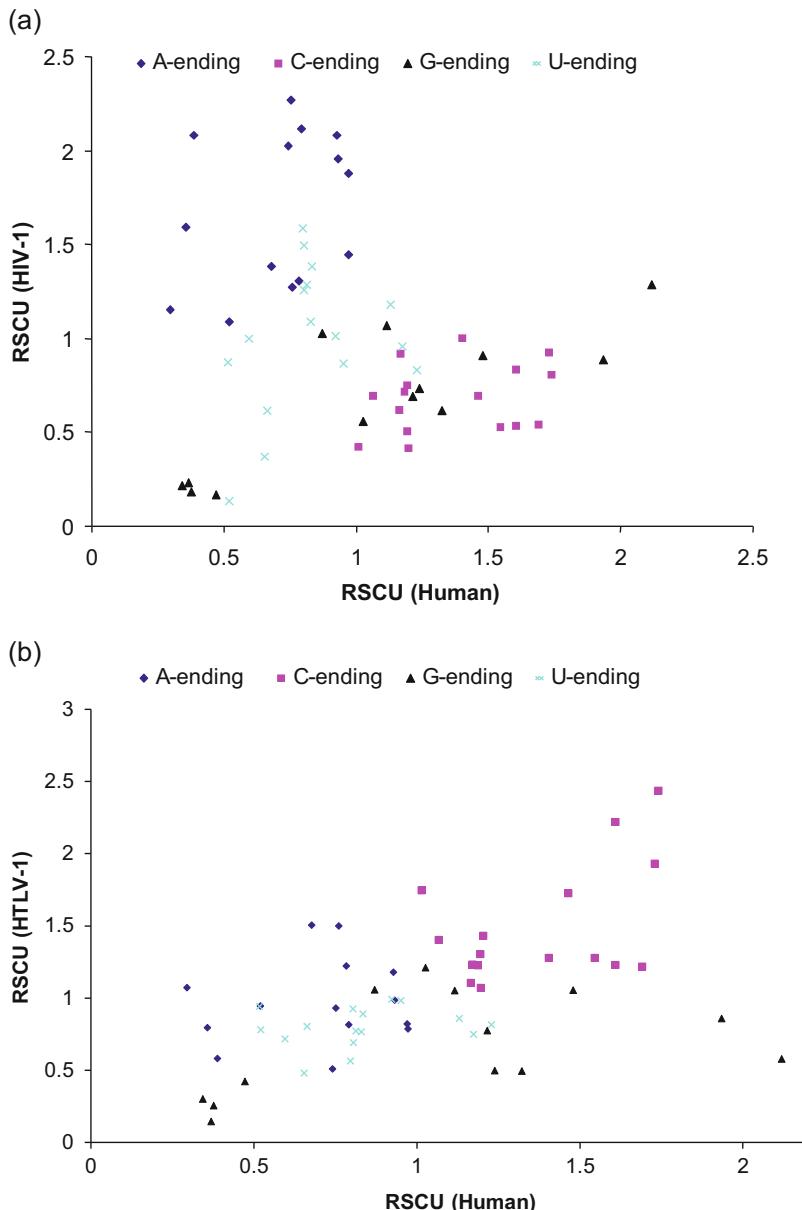
Third, a phage with a wide range of host species may imply diverse tRNA pools that would represent fluctuating selection with different optima. Phage PRD1 mentioned above does have a variety of gram-negative bacteria as hosts, including *Salmonella*, *Pseudomonas*, *Escherichia*, *Proteus*, *Vibrio*, *Acinetobacter*, and *Serratia* species (Bamford et al. 1995; Grahn et al. 2006). However, this diverse array of hosts actually have rather similar codon usage, so host variability is not a good explanation for the lack of similarity in codon usage between PRD1 and *E. coli* (Chithambaram et al. 2014b).

Fourth, the tRNA-mediated selection differs in its effectiveness between temperate phages (i.e., those with lysogeny) and virulent phages (i.e., those without lysogeny). The lysogenic phase effectively hides protein-coding genes of the phage from tRNA-mediated selection, and the phage codon usage will be at the mercy of mutation bias in the host genome. In contrast, virulent phages have their codon usage under tRNA-mediated selection every time they enter the host cell. For this reason, one would expect better codon usage adaptation in virulent phages than in temperate phages, which is true (Prabhakaran et al. 2015).

Fifth, mass translation of phage mRNA often occurs in the late infection phase when the host cellular environment has already been dramatically altered, presumably with a quite different tRNA pool in the late phase from that in the early phase. In vaccinia virus, the degradation of host mRNA appears nearly complete 6 h after the viral infection as no host poly(A) mRNA is detectable at/after this time (Katsafanas and Moss 2007). Shutdown or drastic alteration of host protein and RNA expression implies that many tRNA species are no longer sequestered for host translation, which would dramatically alter availability of different tRNA species. Many other viruses, including hepatitis C (Chan and Egan 2009), SARS (Minakshi et al. 2009), Japanese encephalitis virus (Su et al. 2002), and coxsackie B2 virus (Zhang et al. 2010), can induce stress responses such as the UPR (unfolded protein response) in late phase. URP often results in the shutdown of transcription of ribosomal RNAs as well as repression of translation via phosphorylation of eukaryotic translation initiation factor eIF-2 $\alpha$  (DuRose et al. 2009). All these suggest that the tRNA pool in the late phase differs from that in the normal cell. If codon usage of phage genes adapts to the altered tRNA pool in the late phase, whereas that of host genes adapts to the tRNA pool and normal cells, then we should not expect the parasite and the host share high similarity in codon usage. Interestingly, HIV-1 early genes have RSCU positively correlated with RSCU of human genes, but HIV-1 late genes have RSCU values negatively correlated with RSCU of human genes (van Weringh et al. 2011).

Sixth, if mutation bias is in different direction from tRNA-mediated selection, e.g., if tRNA-mediated selection favors Y-ending codons whereas mutation bias favors R-ending codons (where Y and R stand for pyrimidine and purine, respectively), then strong mutation bias will disrupt selection. This may well be the case for the poor codon adaptation in HIV-1. According to a recent compilation of tRNAs in human genome (Chan and Lowe 2009), the AUC codon can be translated by 17 tRNA<sup>Ile</sup> species (14 tRNA<sup>Ile/IAU</sup> and 3 tRNA<sup>Ile/GAU</sup>) and AUU can be translated by 14 tRNA<sup>Ile/IAU</sup> species, whereas AUA can be translated by only 5 tRNA<sup>Ile/UAU</sup> species. In agreement with the tRNA-mediated selection, human genes code Ile mostly by AUC and least by AUA. In contrast, HIV-1 genes code Ile mostly by AUA and least by AUC (Haas et al. 1996; Nakamura et al. 2000). The poor codon adaptation of HIV-1 (Fig. 9.6a) reduces the translation efficiency of HIV-1 genes. Modifying HIV-1 codon usage according to host codon usage has been shown to increase the production of viral proteins (Haas et al. 1996; Ngumbela et al. 2008). The high frequency of maladaptive AUA codons in HIV-1 genes is due to high A-biased mutation at the third codon position of HIV-1 genes (Jenkins and Holmes 2003). The A-bias is mediated by the error-prone reverse transcriptase (Martinez et al. 1994; Vartanian et al. 2002) and the human APOBEC3 protein (Yu et al. 2004). The frequency of A can reach up to 40% in some HIV-1 genomes (Vartanian et al. 2002), resulting in a preponderance of A-ending codons which are typically rarely used in the human HEGs (Kypr and Mrazek 1987; Sharp 1986).

One would predict a better correlation in RSCU between HIV-1 genes and highly expressed human genes. One viral species that may shed light on this prediction is HTLV-1 which infects the same type of host cell as HIV-1. Both HIV-1 and HTLV-1 are retroviruses with RNA genomes, but HTLV-1 is exceptional in that it does not



**Fig. 9.6** Relative synonymous codon usage (RSCU) of HIV-1 (a) and HTLV-1 (b) plotted against RSCU of highly expressed human genes. Modified from van Weringh et al. (2011)

have a strong A-biased mutation (Van Dooren et al. 2004; van Hemert and Berkhout 1995). HTLV-1 relies for the most part on the host polymerase to replicate through clonal expansion of infected cells rather than undergoing iterative replication cycles

like HIV-1 (Strelbel 2005). The substitution rate of HTLV-1 is consequently lower, about  $5.2 \times 10^{-6}$  substitutions/site/year (Hanada et al. 2004; Van Dooren et al. 2004), whereas that of HIV-1 is around  $2.5 \times 10^{-3}$  substitutions/site/year (Hanada et al. 2004). Thus, although HTLV-1 infects the same cells as HIV-1, i.e., human CD4+ T cells (Rimsky et al. 1988), and both viruses are therefore subject to the same selective pressures on codon usage by the host tRNA pool, mutations are less likely to disrupt codon-anticodon adaptation in HTLV-1 than in HIV-1 as they occur at a lower rate in the former. The positive correlation in RSCU between HTLV-1 and highly expressed human genes (Fig. 9.6b) is highly significant (Pearson  $r = 0.4982$ ,  $p < 0.0001$ , Spearman  $r = 0.4688$ ,  $p = 0.0002$ ).

## 2.2 CAI (*Codon Adaptation Index*)

CAI has been used extensively in biological research. Other than its primary use for measuring the efficiency of translation elongation, it has contributed to the finding that functionally related genes are conserved in their expression across different microbial species (Lithwick and Margalit 2005), to the prediction of protein production (Futcher et al. 1999; Gygi et al. 1999), and to the optimization of DNA vaccines (Ruiz et al. 2006).

### 2.2.1 Calculation of CAI

While RSCU characterizes codon usage bias in each codon family, CAI quantifies the codon usage bias in one gene. It is based on (1) the codon frequencies of the gene and (2) the codon frequencies of a set of known HEGs (often referred to as the reference set). The reference set of genes is used to generate a column of  $w$  values computed as

$$w_{ij} = \frac{\text{RefCodFreq}_{ij}}{\text{RefCodFreq}_{i,\max}} \quad (9.4)$$

where  $\text{RefCodFreq}_{ij}$  is the frequency of codon  $j$  in synonymous codon family  $i$  and  $\text{RefCodFreq}_{i,\max}$  is the maximum codon frequency in synonymous codon family  $i$ . For example, if the four alanine codons GCA, GCC, GCG, and GCU have frequencies 20, 4, 4, and 2, respectively, then their associated  $w$  value are 1, 0.2, 0.2, and 0.1, respectively. The codon whose frequency is  $\text{RefCodFreq}_{i,\max}$  is often referred to as the major codon (whose  $w$  is 1), and the other codons in the synonymous codon family are referred to as minor codons. The major codon is assumed to be the translationally optimal codon.

It is easy to see the relationship between  $w_{ij}$  and RSCU. The former is obtained by dividing each RSCU by the largest RSCU value within each codon family. With the

$w$  values for a particular species, we can now compute the CAI value of any protein-coding sequence from the species by using the following equation:

$$\text{CAI} = e^{\left( \frac{\sum_{i=1}^n [\text{CodFreq}_i \ln(w_i)]}{\sum_{i=1}^n \text{CodFreq}_i} \right)} \quad (9.5)$$

where  $n$  is the number of sense codons (excluding codon families with a single codon, e.g., AUG for methionine and UGG for tryptophan in the standard genetic code). Note that the exponent is simply a weighted average of  $\ln(w)$ . Because the maximum of  $w$  is 1,  $\ln(w)$  will never be greater than 0. Consequently, the exponent will never be greater than 0. Thus, the maximum CAI value is 1. The minimum CAI depends on the  $w$  values for minor codons in each codon family. If the minor codons all have  $w$  values close to zero, then the minimum CAI will also be very close to zero.

The calculation of CAI is numerically illustrated in Table 9.8 for a gene whose observed codon frequency is in column ObsFreq (Table 9.8). The codon frequency of the highly expressed reference set is in column “RefCodFreq.” The column “ $w$ ” is obtained by dividing RefCodFreq values by the largest value in the codon family. For example, the first  $w$  value in the table, 0.606, is obtained by dividing RefCodFreq value 195 by the largest RefCodFreq value in the alanine codon family, i.e., 322. We take a weight average of  $\ln(w)$  as shown in Eq. (9.5) and then exponentiate it to obtain CAI.

The way  $w$  is calculated implies that, if a protein contains only methionine and tryptophan, both encoded by a single codon (AUG and UGG, respectively, in

**Table 9.8** Illustration of CAI calculation for a gene whose observed codon frequencies are in column “ObsFreq”

Codon	AA	ObsFreq	RefCodFreq	$w$
GCA	A	1	195	0.606
GCU	A	15	322	1.000
GCG	A	0	81	0.252
GCC	A	8	242	0.752
UGC	C	3	123	1.000
UGU	C	3	112	0.911
GAU	D	9	69	1.000
GAC	D	11	40	0.580
GAG	E	11	289	0.863
GAA	E	14	335	1.000
UUU	F	3	118	0.554
UUC	F	9	213	1.000
...	...	...		...

The codon frequency of the highly expressed reference set is in column “RefCodFreq.” The column “ $w$ ” is obtained by dividing RefCodFreq values by the largest value in the codon family

standard code), then the gene will have the highest CAI value of 1 because  $w$  values are 1 for such codons. Similarly, a gene with many AUG and UGG codons would have high CAI values even if it is not under any tRNA-mediated selection. For this reason, a good implementation of CAI should exclude single-member codon families from CAI calculation.

I have previously mentioned that codon usage indices such as CAI can be implemented differently with different classification of codon families, so gene A could have a higher CAI value than gene B from one software, but the opposite from another software. I wish to illustrate this so that the reader can better interpret their results.

In highly expressed yeast genes (e.g., compiled in the Eyeastcai.cut in EMBOSS distribution), CGU is by far the most frequent codon in the CGN (coding for arginine) codon family. The overuse of CGT and the avoidance of CGG, CGA, and CGC codons in highly expressed yeast genes make sense because the yeast genome contains six tRNA<sup>Arg</sup> genes with anticodon ACG forming Watson-Crick base pairing with the CGT codon, but no other tRNA<sup>Arg</sup> gene forming Watson-Crick base pairing with the other three CGN codons (the nucleotide A in anticodon ACG is modified to inosine but still pairs with U better than with other nucleotides). While this illustrates well the codon-anticodon adaptation, it causes practical problems with computing CAI.

Suppose we now use a sequence consisting entirely of CGU codons and expect the resulting CAI to be 1 by using the Eyeastcai.cut reference set. The resulting CAI value from the EMBOSS.cai program is 0.140 instead of 1. It turns out that amino acid arginine is coded by two codon subfamilies, the CGN codon family we have mentioned and the AGR codon family. The largest codon frequency among these six codons is 314 (for AGA codon) in Eyeastcai.cut. So the  $w$  value for CGT is not 1 (43/43) as we have thought but is only 0.1369 (= 43/314). For this reason, some CAI-calculating programs, e.g., DAMBE (Xia 2013, 2017d), may separate compound codon families such as the arginine family into two separate families, one twofold and one fourfold.

### 2.2.2 Illustration of CAI Applications

The most obvious application of CAI or related codon usage indices is to optimize codon usage to optimize protein expression. Many experiments have demonstrated increased protein production by optimizing codon usage and decreased protein production if codons are replaced by rarely used ones (Haas et al. 1996; Kaishima et al. 2016; Ngumbela et al. 2008; Robinson et al. 1984; Sorensen et al. 1989). There are claims that codon optimization does increase protein production (e.g., Kudla et al. 2009), but these claims were found to be due to wrong data analysis (Tuller et al. 2010; Xia 2015) and will be dealt with on a later section on  $I_{TE}$  (Xia 2015). Below I list two less obvious applications of CAI.

### 2.2.2.1 Does High Mutation Rate Prevent HIV-1 Genes from Evolving Codon Adaptation?

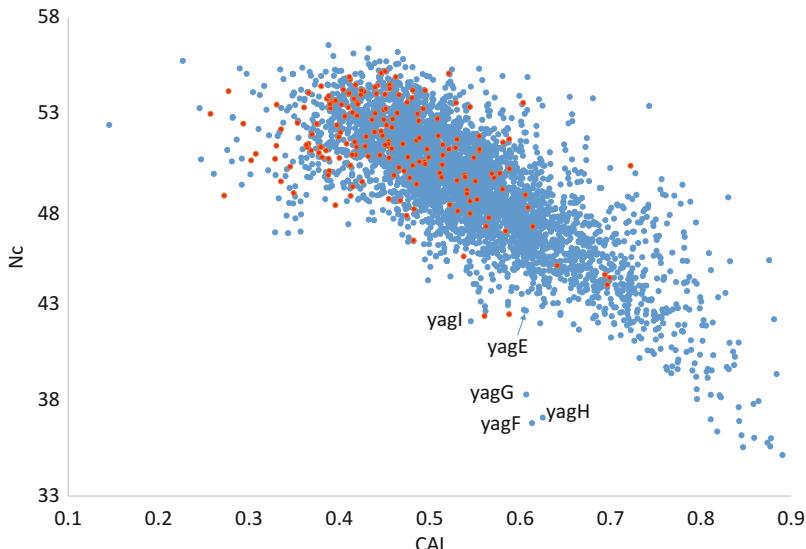
I have mentioned in the section on RSCU that the lack of concordance in codon usage between HIV-1 and human genes was conventionally explained by high mutation rate in HIV-1, based on the observation that (1) HIV-1 genome is known to experience strongly A-biased mutations, (2) usage of A-ending codons in HIV-1 genes is particularly different from that of the host genes, and (3) HTLV-1 that parasitizes the same human CD4+ T cells but has reduced mutation rate does have codon usage similar to human genes (Fig. 9.6b). Thus, the lack of concordance in codon usage between HIV-1 and human genes is interpreted as poor codon adaptation caused by high mutation rate disrupting codon adaptation.

However, van Weringh et al. (2011) objected to this interpretation. They argued that the lack of concordance in codon usage between HIV-1 and human genes is not due to poor codon adaptation in the part of HIV-1 genes, but because HIV-1 genes, especially the late genes, have adapted to a tRNA pool that is fundamentally different from that in a normal human CD4+ T cell. What originally prompted them to formulate this hypothesis is the observation that CAI for HIV-1 early genes are significantly greater than CAI for HIV-1 late genes when highly expressed human genes are used as reference genes. These late genes encode mass-translated HIV-1 structural proteins and are typically expected to have higher CAI than the relatively lowly expressed early genes. So it is thus a surprise to see late genes having smaller CAI than early genes, unless the mass-translated late genes adapt to a tRNA pool different from the early genes.

van Weringh et al. (2011) investigated experimentally measured tRNA abundance in the human cell when the late HIV-1 genes are translated and HIV-1 virions are produced. The tRNA pool for the late genes is indeed different in the expected direction, supporting their hypothesis that the lack of concordance in codon usage between HIV-1 and human genes is not due to poor codon adaptation in HIV-1 genes but because HIV-1 genes, especially the late genes, have adapted to a tRNA pool different from the one with which highly expressed human genes are translated (van Weringh et al. 2011).

### 2.2.2.2 Detecting Horizontally Transferred Genes

CAI has also been used jointly with a reformulated effective number of codons ( $N_c$ , Sun et al. 2013) to detect horizontally transferred genes. *E. coli* genes with a strong codon usage bias typically have high CAI values. However, three genes (*yagF*, *yagG*, and *yagH*) from the defective CP 4–6 prophages of *E. coli* (Wang et al. 2010) have strongly biased codon usage (small  $N_c$  values) but relatively small CAI values. This codon usage pattern sets the three genes apart from the rest of *E. coli* genes (Fig. 9.7) which highlight the value of using the “ $N_c$  versus CAI” plot to detect



**Fig. 9.7** Plot of CAI against a reformulated effective number of codons ( $N_c$ , Sun et al. 2013) for *E. coli* genes facilitates the detection of newly “immigrant” genes that exhibit codon usage bias different from the “native” genes. Three *E. coli* genes (*yagF*, *yagG*, and *yagH*) from the defective CP 4–6 prophages of *E. coli* (Wang et al. 2010) have strongly biased codon usage (relatively small  $N_c$ ) but relatively poor codon adaptation (mediocre CAI values). The red points represent 179 annotated *E. coli* pseudogenes (NC\_000913) that have not accumulated frameshifting mutations

recently horizontally transferred genes. These genes have been “naturalized” in *E. coli* genome and contribute to *E. coli* survival and growth (Wang et al. 2010).

The largest mucin gene (*mucin 14A*) in *Drosophila melanogaster* also exhibits strong codon usage bias ( $N_c = 38.6$ ), but in the direction opposite to those highly expressed *D. melanogaster* genes. Its CAI value is equal to 0.1277, which is the second smallest among all *D. melanogaster* genes. It is unknown how and why the gene has evolved to have such a peculiar feature.

The distribution of CAI values for the 179 annotated pseudogenes are indicated in red. These pseudogenes have not accumulated frameshifting mutations and presumably were pseudogenized only recently. They tend to be clustered on the lower end of CAI distribution, suggesting that genes with high CAI values require tRNA-mediated selection to maintain the high CAI values.

The gene with the smallest CAI is *mglL*, which has only 17 sense codons and is a bacterial mRNA leader that controls the expression of the downstream *mgtA* (Park et al. 2010). The low CAI is not due to stochastic fluctuation due to small number of codons but because almost all used codons are minor codons. This may represent a real case of a gene preferring minor codons to facilitate its regulatory function.

### 2.2.3 Problems with CAI and Other Gene-Specific Codon Usage Indices

There are major problems with CAI and other commonly used codon usage indices. While some minor problems have been addressed before (Xia 2007c), the key issue of properly inferring translationally optimal codons (TOCs) remains unresolved. These gene-specific codon usage indices all need to infer TOCs, by using two types of information. The first, represented by tAI (dos Reis et al. 2004), uses the most abundant tRNA and its anticodon to infer TOC within each codon family, i.e., the codon that base-pairs best with the most abundant tRNA is the TOC. The second, represented by CAI, considers the most frequent codon in HEGs as the TOC within each codon family. I will outline the problems to pave the way for the presentation of a new index of translation elongation in the next section ( $I_{TE}$ , Xia 2015).

#### 2.2.3.1 Problem with Codon Usage Indices Using tRNA Abundance to Infer TOCs

For indices such as tAI that use tRNA abundance information to define TOCs, the main problem is that TOCs cannot be inferred reliably from tRNA gene copy numbers or experimentally measured tRNA abundance. For example, inosine is expected to pair best with C and U, less with A (partly because of the bulky I/A pairing involving two purines), and not with G. However, tRNA<sup>Val/IAC</sup> from rabbit liver pairs better with GUG codon than with other synonymous codons (Jank et al. 1977; Mitra et al. 1977). No one would have identified GUG as the best codon for tRNA<sup>Val/IAC</sup> without actually seeing the experimental result.

Similarly, the *Bacillus subtilis* genome codes tRNA<sup>Ala/GGC</sup> for decoding GCY codons. One would have thought that GCC codon, which forms Watson-Crick base pairing with the anticodon, would be translationally more optimal than GCU. However, GCU is used much more frequently than GCC in HEGs than LEGs in *B. subtilis*. We have encountered a similar example in Table 9.4 involving Cys codon usage in HEGs. There are four tRNA<sup>Cys</sup> genes with the same anticodon GCA forming Watson-Crick base pair with UGC codon, but no tRNA<sup>Cys</sup> gene with anticodon forming Watson-Crick base pair with the alternative UGU codon. We would have taken UGC as the TOC. However, UGU is used far more frequently than UGC codon in highly expressed yeast genes relative to LEGs. In short, in all these cases we would be wrong to use the most abundant tRNA species and its matching codon to infer TOC.

There is one more reason for tRNA abundance not able to reliably predict TOCs. What matters in translation elongation is not the abundance of transcribed tRNAs but the availability of charged tRNAs. It is tedious to determine the level of charged tRNAs, and researchers typically would use transcriptionally determined tRNAs or even the number of tRNA genes in the genome as a proxy of charged tRNAs. Unfortunately, the abundance of tRNAs often do not reflect the abundance of charged tRNA (Elf et al. 2003).

Furthermore, codon-anticodon base pairing is known to be context-dependent (Lustig et al. 1989). For example, a wobble cmo<sup>5</sup>U in the anticodon of tRNA<sup>Pro</sup>, tRNA<sup>Ala</sup>, and tRNA<sup>Val</sup> can read all four synonymous codons in the respective codon family, but the same cmo<sup>5</sup>U in tRNA<sup>Thr</sup> cannot read C-ending codons (Nasvall et al. 2007). For this reason, the optimal codon usage is likely better approximated by the codon usage of HEGs than what we can infer based on codon-anticodon pairing. Consistent with this proposition, CAI, which is based on the codon usage of HEGs (HEGs), performs better in predicting protein production or abundance than other indices based on tRNAs (Coghlan and Wolfe 2000; Comeron and Aguade 1998; Duret and Mouchiroud 1999).

### 2.2.3.2 Problem with Using Codon Usage of HEGs to Infer TOCs

Codon usage indices such as CAI that use codon usage of HEGs to infer TOCs also have problems. Other than those previously outlined (Xia 2007c), it often leads to wrong interpretation of tRNA-mediated selection. I illustrate this problem here with the Ala codon subfamily GCR (where R stands for either A or G). The frequencies of GCA and GCG in *E. coli* HEGs, as compiled and distributed with EMBOSS (Rice et al. 2000), are 1973 and 2654, respectively, which may lead one to think that *E. coli* translation machinery prefers GCG over GCA. However, the codon frequencies of GCA and GCG for *E. coli* non-HEGs are 25,511 and 43,261, respectively. Thus, GCA is relatively more frequent in *E. coli* HEGs than in *E. coli* non-HEGs. This suggests that mutation bias favors GCG, but tRNA-mediated selection favors GCA. The battle between the mutation bias and tRNA-mediated selection leads to increased usage of GCA in *E. coli* HEGs relative to LEGs, although GCA is still not as frequent as GCG in HEGs. This interpretation is corroborated by the *E. coli* genome encoding three tRNA<sup>Arg</sup> genes for GCR codons, all with a UGC anticodon forming perfect Watson-Crick base pair with codon GCA.

The example above illustrates the point that mutation bias is reflected to codon usage of lowly expressed genes. This is what has driven the formulation, development, and implementation of a new codon usage index,  $I_{TE}$  (Xia 2015).

## 2.3 $I_{TE}$ (*Index of Translation Elongation*)

### 2.3.1 Illustration of $I_{TE}$ Calculation

$I_{TE}$  is implemented in DAMBE (Xia 2013, 2017d). There are in fact four different implementations of  $I_{TE}$  in DAMBE, depending on how one would classify codons into codon families. The first implementation is the most extreme (unconventional) and classifies all sense codons into NNR or NNY codon families or subfamilies. For example, the fourfold alanine codon is broken into GCR and GCY subfamilies. For such an NNR or NNY codon family or subfamily  $i$ , we first define  $P_{i,HEG}$  and  $P_{i,\text{non-}}$

HEG as the proportion of codon  $i$  within its R-ending or Y-ending family for *E. coli* HEGs and non-HEGs. Take data for codons GCA and GCG in Table 9.9, for example:

$$P_{\text{GCA.HEG}} = \frac{N_{\text{GCA.HEG}}}{N_{\text{GCR.HEG}}} = \frac{1973}{1973 + 2654} = 0.42641 \quad (9.6)$$

$$P_{\text{GCA.non-HEG}} = \frac{N_{\text{GCA.non-HEG}}}{N_{\text{GCR.non-HEG}}} = \frac{25511}{25511 + 43261} = 0.37095$$

$$S_{\text{GCA}} = \frac{P_{\text{GCA.HEG}}}{P_{\text{GCA.non-HEG}}} = 1.1495 \quad (9.7)$$

$$S_{\text{GCG}} = \frac{P_{\text{GCG.HEG}}}{P_{\text{GCG.non-HEG}}} = 0.9118$$

where  $S_{\text{GCA}}$  and  $S_{\text{GCG}}$  may be viewed as relative codon frequencies of HEGs corrected for the “background” non-HEGs. Codon  $i$  is considered selected for if  $S_i > 1$  and against if  $S_i < 1$ . Thus, codon GCA is considered selected for because, according to Eq. (9.7),  $S_{\text{GCA}} > 1$ . This insight would be obscured if we use codon frequency data from *E. coli* HEGs only which would have suggested that codon GCA is selected against. The  $S_i$  values for the four sense codons in *E. coli* are listed in Table 9.9.

We now compute  $w_i$  as follows:

$$w_i = \frac{S_i}{\text{Max}(S_i)}, \text{ e.g.,} \quad (9.8)$$

$$w_{\text{GCA}} = \frac{1.1495}{1.1495} = 1; w_{\text{GCG}} = \frac{0.9118}{1.1495} = 0.7932$$

The index of translation elongation ( $I_{\text{TE}}$ ) is then calculated in the same way as CAI except that, in this particular codon family classification, the computation is applied to NNR and NNY codon subfamilies:

$$I_{\text{TE}} = e^{\frac{\sum_{i=1}^{N_s} F_i \ln w_i}{\sum_{i=1}^{N_s} F_i}} \quad (9.9)$$

**Table 9.9** Codon frequency (CF) for *E. coli* highly expressed genes (HEGs) and non-HEGs, as well as the computed  $S_i$  values according to Eq. (9.7)

AA	Codon	CF <sub>HEG</sub>	CF <sub>non-HEG</sub>	$S_i$
A	GCA	1973	25,511	1.1495
A	GCG	2654	43,261	0.9118
A	GCC	1306	33,463	0.5646
A	GCU	2288	18,526	1.7865
...	...	...	...	...

where  $F_i$  is the frequency of codon  $i$  and  $N_s$  is the number of sense codons (excluding those in single-codon families). For example, AUG for methionine, AUA for isoleucine, and UGG for tryptophan in the standard genetic code are excluded from computing  $I_{TE}$ . Just like CAI, tAI, and  $N_c$ ,  $I_{TE}$  is a gene-specific index of codon usage bias.

One may note that CAI is a special case of  $I_{TE}$  when there is absolutely no codon usage bias in non-HEGs in all codon subfamilies. That is, when  $N_{GCA.\text{non-HEG}} = N_{GCG.\text{non-HEG}}$ ,  $N_{GCC.\text{non-HEG}} = N_{GCU.\text{non-HEG}}$ , and so on. The range of  $I_{TE}$  is the same as CAI, i.e., between 0 and 1.

Readers may demand a justification for the extreme classification of all sense codons into NNR and NNY codon families. The main reason is that, for genes encoded by the nuclear genome, the R-ending codons are typically decoded by two types of tRNA species (one with a wobble C and the other with a wobble U), whereas the Y-ending codons are decoded typically by a single type of tRNA species with either a wobble G or a wobble A modified to inosine, but never by both (Grosjean et al. 2007; Marck and Grosjean 2002). For this reason, the R-ending and Y-ending codons, even within a single fourfold codon family, are subject to different tRNA-mediated selection and therefore should be treated separately. Such implementation is also relevant for certain experimental settings that induce mutation almost exclusively in NNY codons, which is the case in Kudla et al. (2009). However, for comparative purposes, I have included two alternative  $I_{TE}$  implementations in DAMBE (Xia 2013, 2017d): (1) with compound sixfold and eightfold codon families broken into twofold and fourfold codon families and (2) lumping all synonymous codons into one codon family. One may access the function by clicking “Seq.Analysis|Codon usage|Index of translation elongation” and then choosing the desired implementation.

### 2.3.2 A Major Controversy Resolved by the Application of $I_{TE}$

Highly expressed genes in bacteria and unicellular eukaryotes overuse codons that match the anticodon of the most abundant tRNA (Ikemura 1981a, b, 1982, 1992). When such codons are replaced by rarely used codons, protein production is reduced (Robinson et al. 1984; Sorensen et al. 1989). Similarly, when codon usage is optimized, protein production is increased (Haas et al. 1996; Kaishima et al. 2016; Ngumbela et al. 2008). However, to what degree is translation elongation rate-limiting has been controversial. Early theoretical considerations (Andersson and Kurland 1983; Bulmer 1990, 1991; Liljenstrom and von Heijne 1987) tend to favor the argument that translation elongation is not rate-limiting in protein production, but translation initiation is. This hypothesis states that codon-anticodon adaptation and increased elongation efficiency are not related to protein production. Instead, the benefit of codon adaptation and increased elongation efficiency is to increase ribosomal availability for global translation and timely response to environmental perturbations.

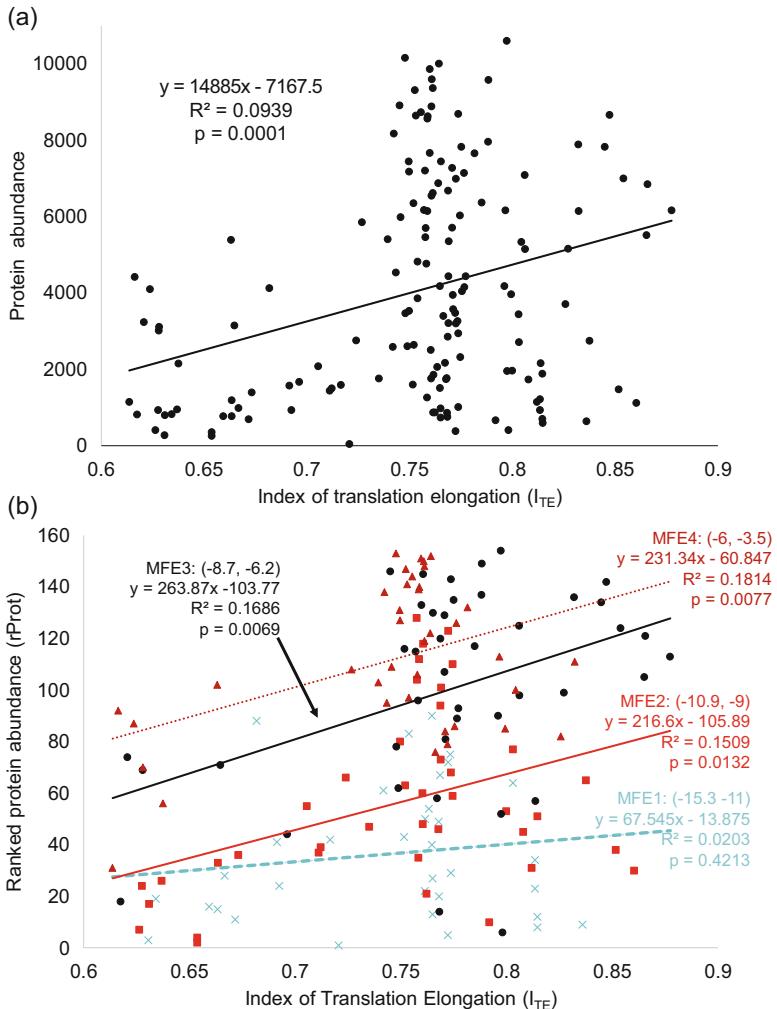
To test these two alternative hypotheses, Kudla et al. (2009) engineered a synthetic library of 154 genes, all encoding the same green fluorescent protein in *Escherichia coli*, but differing in synonymous sites (and consequently the degree of codon adaptation, as measured by codon adaptation index or CAI). All sequences share an identical 5' UTR of 144 nt long, so there is no variation in the Shine-Dalgarno sequence. Because the engineered genes all encode the same protein, it is justifiable to use protein abundance as a proxy for protein production (assuming that protein molecules sharing the same amino acid sequence have the same degradation rate).

Kudla et al. (2009) used minimum folding energy (MFE), computed from sites -4 to +37 (where ribosomes position themselves at the initiation codon), as a proxy for initiation efficiency. The rationale for using MFE as a measure of translation initiation is that an initiation codon would be inaccessible if it is embedded in a strong secondary structure and that accessibility of the initiation codon is a key determinant of translation initiation efficiency (Nakamoto 2006). Stable secondary structure in sequences positioned at or before the start codon has been experimentally shown to inhibit translation initiation (Osterman et al. 2013), presumably because it embeds SD and start codon in a structural stem and consequently hiding the SD and start codon signals from ribosomes. The previous chapter on translation initiation has already highlighted the point that mRNAs in bacteria and unicellular eukaryotes tend to have much weaker secondary structure near the start codon than elsewhere, especially those from highly expressed.

Kudla et al. interpreted CAI as a proxy of translation elongation. If both translation initiation and elongation contribute to translation efficiency, then protein production is expected to depend on both MFE and CAI. If only translation initiation is important, then protein production will depend on MFE only. They found that MFE accounts for 44% of the variation in protein production but CAI is essentially unrelated to protein production. They concluded consequently that “translation initiation, not elongation, is rate-limiting for gene expression.”

The conclusion by Kudla et al. (2009), however, is based on two critical assumptions. First, MFE and CAI are good proxies of translation initiation and elongation efficiencies, respectively. Second, the effect of translation elongation is independent on translation initiation. The problem with the second assumption has been pointed out recently (Supek and Smuc 2010; Tuller et al. 2010) who reanalyzed the data in addition to providing an overwhelming amount of additional empirical evidence to demonstrate the joint effect of both translation initiation and elongation on protein production. In short, protein production rate is expected to increase with elongation efficiency only when translation initiation is efficient. If translation initiation is slow, then increasing elongation rate is not expected to increase protein production. Kudla et al. (2009) ignored the dependence of elongation effect on translation initiation.

Xia (2015) reanalyzed the experimental data in Kudla et al. (2009) with two improvements, by replacing CAI by  $I_{TE}$  and by incorporating translation initiation and elongation into one model. Three points are worth highlighting in Fig. 9.8a. First, in contrast to a nonsignificant relationship between protein abundance and CAI, the protein abundance and  $I_{TE}$  are highly significantly correlated ( $p = 0.0001$ ,



**Fig. 9.8** Relationship between protein abundance (measured by GFP normalized fluorescence; data kindly provided by Dr. Plotkin) translation elongation efficiency ( $I_{TE}$ ). **(a)** Without considering translation initiation. **(b)** The relationship between protein abundance and  $I_{TE}$  is characterized separately for four groups of data, with MFE1, MFE2, MFE3, and MFE4 corresponding to groups of genes with increasing translation initiation efficiency. (Modified from Xia 2015)

Fig. 9.8a). Second, when  $I_{TE}$  is small (e.g.,  $I_{TE} < 0$ ), protein abundance is generally low, suggesting that translation elongation is limiting. Third, a large  $I_{TE}$  (efficient translation elongation) does not imply high protein production, e.g., when translation initiation is very slow. One expects a large  $I_{TE}$  to be associated with increase protein production only when translation initiation is efficient.

Xia (2015) binned MFE into four MFE categories, from strong secondary structure to weak secondary structure ( $-15.3, -11, -10.9, -9, -8.7, -6.2$ , and  $-6, -3.5$ ), representing translation initiation from the lowest to the highest, and designated as MFE1-MFE4 (Fig. 9.8b). The intervals are chosen in such a way that all MFE values fall into four roughly equal-sized groups with within-group MFE being as small as possible. The benefit of binning is that one can exclude the MFE variable so that the effect of  $I_{TE}$  can be modeled more explicitly. It is for the same reason that Tuller et al. (2010) also used binned analysis for this data set.

In the MFE1 group, translation initiation is the lowest, and we should expect little increase of protein production with translation elongation efficiency ( $I_{TE}$ ). This is consistent with the empirical result (Fig. 9.8b) where the relationship between  $I_{TE}$  and protein abundance is not statistically significant in the MFE1 group ( $b = 67.545, p = 0.4213$ , Fig. 9.8b), with  $I_{TE}$  accounting for only 2% of total variation in ranked protein abundance (rProt). In contrast, when translation initiation is more efficient in groups MFE2-MFE4, rProt increases significantly with  $I_{TE}$ , with the simple linear model consistently accounts for about 17% of the total variation in rProt (Fig. 9.8b, with  $b$  varying from 216.60 to 263.87). Thus, the contribution of translation elongation ( $I_{TE}$ ) to protein production is much greater than previously documented for this data set, i.e., absent (Kudla et al. 2009) or less than 3% of the total variation in protein production (Tuller et al. 2010). Readers may consult Xia (2015) for more explicit modeling of the protein abundance on translation initiation and elongation.

One might wonder why previous studies, although not taking translation initiation into consideration, almost always consistently show positive relationship between translation efficiency and codon adaptation. There are two explanations. First, previous experimental studies were carried out typically on highly expressed genes with efficient translation initiation efficiency. Such studies are equivalent to excluding the MFE1 group in Fig. 9.8b. Second, for correlational studies, nature generally does not generate bacterial genes with high translation initiation efficiency but poor codon adaptation or low translation initiation with high codon adaptation. However, the experiment by Kudla et al. (2009) generated both of these unnatural associations, leading to a lack of positive association between protein production and codon adaptation. This example highlights the point that a well-intended and well-done experiment can mislead us. It represents another illustration of Simpson's Paradox in which wrong conclusion is reached when one omits a contributing variable.

### 3 Translation Elongation Efficiency and Accuracy

Given a fixed translation initiation efficiency, our conceptual model for the relationship between codon adaptation (CA) and tRNA-mediated selection, in its simplest form, is

$$CA = \alpha + \beta S_E \quad (9.10)$$

where CA is tRNA-mediated codon adaptation often measured by CAI or  $I_{TE}$  (Xia 2015) and  $S_E$  is selection for translation efficiency (in unit of protein produced per mRNA molecule). The slope  $b$  is typically positive, i.e., stronger selection for translation efficiency leads to better codon adaptation. Many studies have demonstrated a strong relationship between codon adaptation and gene expression (Coghlan and Wolfe 2000; Duret and Mouchiroud 1999; Gouy and Gautier 1982).

One key deficiency in Eq. (9.10) is that it does not distinguish between selection due to translation efficiency or that due to translation accuracy (Akashi 1994). Take Asn codons AAC and AAU in *E. coli*, for example. AAC is a major codon (heavily used by highly expressed genes and decoded by the most abundant isoacceptor tRNA), whereas AAU is a rarely used minor codon. A major codon is typically translated faster than a minor codon, and highly expressed *E. coli* genes use AAC almost exclusively to code for Asn, so one could argue that the overuse of AAC is driven by  $S_E$ . However, AAC and AAU also differ in misreading rate, in particular by tRNA<sup>Lys</sup> which ideally should decode only AAA and AAG codons but does misread AAC and AAU, leading to Asn replaced by Lys. This misreading error rate is six times greater for AAU than for AAC, with the error ratio maintained in both Asn-starved and Asn-non-starved conditions (Johnston et al. 1984) or with streptomycin used to inhibit translation (Johnston and Parker 1985). Thus, the overuse of AAC could be driven either by selection for increased translation efficiency or increased translation accuracy or both. Designating  $S_A$  as selection for translation accuracy, we have three alternative hypotheses expressed, in the simplest form, as

$$CA = \alpha + \beta_1 S_E \quad (9.11)$$

$$CA = \alpha + \beta_1 S_A \quad (9.12)$$

$$CA = \alpha + \beta_1 S_E + \beta_2 S_A + \beta_3 S_E S_A \quad (9.13)$$

Akashi (1994) classified amino acid sites into conserved sites (assumed to be functionally important with high  $S_A$ ) and variable sites (assumed to experience low  $S_A$ ). He reasoned that, if codon adaptation is due to selection for translation efficiency, then all codons in the gene should be subject to similar selection regardless of whether the codon is in a functionally important or unimportant site. In contrast, if codon adaptation is driven by selection for translation accuracy, then the selection is stronger in functionally important sites than in functionally unimportant sites. So we should observe greater codon usage bias in functionally important codon sites than functionally unimportant codon sites. He found greater codon adaptation in conserved amino acid sites than in variable amino acid sites and concluded that this difference between the conserved and variable sites to have resulted from selection for accuracy.

There is a problem with the conclusion. Take lysine codons (AAA and AAG) and glutamate codons (GAA and GAG), for example. Suppose that AAA codon is favored by selection in lysine codon family and GAG favored in glutamate codon family. Also suppose that an ancestral gene has good codon adaptation with lysine coded by AAA and glutamate coded by GAG. Now some lysine sites experienced

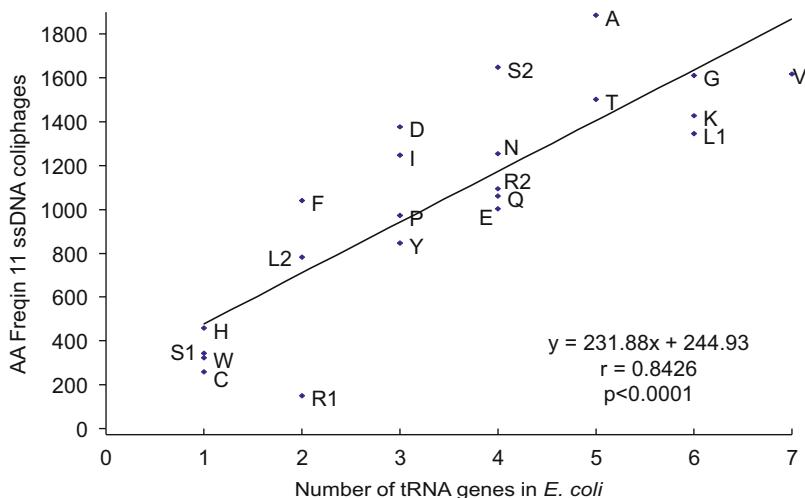
nonsynonymous substitutions from AAA to GAA. These sites are now designated as variable sites and are occupied by a minor codon GAA. This would result in an association between “poor codon adaptation” and variable sites that have little to do with translation accuracy. Akashi (1994) was aware of this problem but did not provide a definitive solution.

## 4 Amino Acid Usage and Translation Elongation Efficiency

There are at least four factors contributing to amino acid usage. The first two are related to selection for translation elongation efficiency, the third related to number of synonymous codons, and the fourth related to genomic mutation bias.

### 4.1 *Factors Related to Selection for Translation Elongation Efficiency*

Some amino acids are abundant and energetically cheap to make, i.e., consuming few ATPs in their production, whereas others are rare and energetically expensive, so mass-produced proteins should maximize the use of abundant and cheap amino acids (Akashi and Gojobori 2002). However, such a hypothesis, without considering other factors, often does not produce easily testable predictions. For example, we expect highly expressed proteins to maximize the use of energetically cheap amino acids and avoid the use of the expensive ones. However, many ribosome proteins are highly expressed, yet the need for many of them to bind to the negatively charged mRNA demands the usage of positively charged amino acids such as Lys and Arg that are typically energetically expensive to make in the cell. This would lead to an association between high expression and energetically expensive amino acid, thus confounding the prediction that highly expressed genes should maximize the use of cheap amino acids. Furthermore, amino acid availability changes with environment, and the same amino acid may be manufactured differently with different energy consumption in different organisms. So it is not easy to measure energetic cost of amino acids in different organisms. One could, however, turn the question around and ask how one can characterize energetic costs of amino acids by bioinformatic means. For example, in the ideal situation when all other factors affecting amino acid usage have been controlled for, we may infer that the avoided amino acid is perhaps rare or energetically expensive to make. This type of inference is of course not very satisfactory and is often derogatively termed the backdoor smuggling approach because one does not present direct evidence for energetic cost.



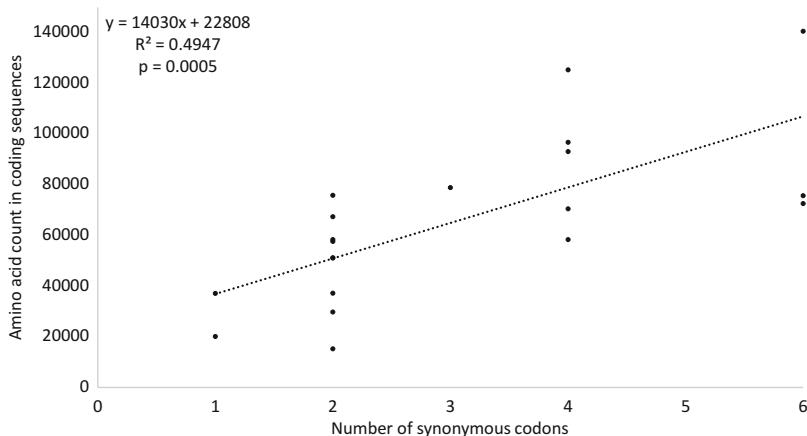
**Fig. 9.9** Amino acid usage in single-stranded DNA phages infecting *E. coli* increases with the abundance of isoaccepting tRNA

The other factor related to translation elongation is the tRNA abundance, and one expects mass-produced proteins to use amino acids with many tRNAs to carry them. Designating the proportion of tRNAs carrying amino acid  $i$  as  $P_i$ , and the frequency of amino acid  $i$  in highly expressed genes as  $N_i$ , Xia (1998a) analytically derived an equation with  $P_i$  linearly increasing with the square root of  $N_i$ . The relationship was well substantiated with data from *E. coli*, *Salmonella typhimurium*, and *Saccharomyces cerevisiae* (Xia 1998a).

Single-stranded DNA (ssDNA) bacteriophages do not carry their own tRNA and depend entirely on the host tRNA pool for decoding their codons. So one would predict that amino acid usage in these phages should be correlated with the abundance of tRNAs in the host cell. This prediction is tested in a study (Chithambaram et al. 2014b) of phages infecting *E. coli*, by using tRNA gene copy number in *E. coli* as a proxy of tRNA abundance (Fig. 9.9). An amino acid carried by more tRNA is used more frequently than another carried by few tRNAs.

## 4.2 Number of Synonymous Codons

In the lack of any selection, we would expect amino acid usage to increase with the number of synonymous codons (Fig. 9.10). However, this relationship is confounded with the number of tRNAs carrying each amino acid in the cell. If we designate the number of tRNA carrying amino acid  $i$  as  $N_{i,\text{tRNA}}$  and the number of synonymous codons for amino acid  $i$  as  $N_{i,\text{syn codon}}$ , then amino acid usage depends on both.  $N_{i,\text{tRNA}}$  and  $N_{i,\text{syn codon}}$  are also positively correlated.



**Fig. 9.10** Amino acid count in all coding sequences in *E. coli* I12 (NC\_000913) increases with number of synonymous codons

### 4.3 Genomic Mutation Bias

*E. coli* genomes have roughly equal nucleotide frequencies. A more AT-rich or GC-rich genome would tend to have more AT-rich or GC-rich codon and their encoded amino acids. For example, AT-rich genomes in bacterial pathogens tend to have many more lysine (encoded by AAA and AAG) than less AT-rich genomes (Xia and Palidwor 2005). This is highly visible even with mild difference in genomic AT content. For example, yeast (*Saccharomyces cerevisiae*) is only mildly AT-rich (0.3090, 0.1917, 0.1913, and 0.3080 for A, C, G, and T, respectively), but the yeast clearly uses more amino acids encoded by AT-rich codons and fewer amino acid encoded by GC-rich codons (Table 9.10).

In summary, amino acid usage ( $U$ ) is a function of four factors:

$$U = F(E, N_{\text{tRNA}}, N_{\text{syncodon}}, \text{GC}\%) \quad (9.14)$$

where  $E$  is energetic cost,  $N_{\text{tRNA}}$  and  $N_{\text{syncodon}}$  have been defined before, and  $\text{GC}\%$  is genomic GC% reflecting mutation bias. One needs to include all these factors in a model in order to reach a reasonable understanding of the determinants of amino acid usage.

### Postscript

I usually will share Simpson's Paradox with students after lecturing on the joint effect of translation initiation and elongation on protein production. If we do not take translation initiation into consideration, we may arrive at a wrong conclusion that

**Table 9.10** Amino acid usage in *E. coli* K12 (NC\_000913) and *S. cerevisiae* (NC\_001133-NC\_001148) coding sequences

AA	Codon	<i>E. coli</i>	Yeast	<i>E. coli</i> %	Yeast%
<i>Ala</i>	<i>GCT,GCC,GCA,GCG</i>	<i>125,332</i>	<i>160,810</i>	<b>9.5527</b>	<b>5.4966</b>
<i>Arg</i>	<i>CGT,CGC,CGA,CGG,AGA,AGG</i>	<i>72,502</i>	<i>130,068</i>	<b>5.5260</b>	<b>4.4458</b>
<b>Asn</b>	<b>AAT,AAC</b>	<b>51,075</b>	<b>179,836</b>	<b>3.8929</b>	<b>6.1469</b>
Asp	GAT,GAC	67,349	171,072	5.1333	5.8473
Cys	TGT,TGC	15,188	37,093	1.1576	1.2679
Gln	CAA,CAG	58,360	115,741	4.4481	3.9561
Glu	GAA,GAG	75,786	191,267	5.7763	6.5376
Gly	<i>GGT,GGC,GGA,GGG</i>	<i>96,701</i>	<i>145,433</i>	<b>7.3705</b>	<b>4.9710</b>
His	CAT,CAC	29,751	63,505	2.2676	2.1706
<b>Ile</b>	<b>ATT,ATC,ATA</b>	<b>78,845</b>	<b>191,677</b>	<b>6.0095</b>	<b>6.5516</b>
Leu	TTG,TTA,CTT,CTC,CTA,CTG	140,571	277,988	10.7142	9.5017
<b>Lys</b>	<b>AAA,AAG</b>	<b>57,620</b>	<b>214,842</b>	<b>4.3917</b>	<b>7.3434</b>
Met	ATG	37,093	60,672	2.8272	2.0738
<b>Phe</b>	<b>TTT,TTC</b>	<b>51,131</b>	<b>129,516</b>	<b>3.8972</b>	<b>4.4269</b>
<i>Pro</i>	<i>CCT,CCC,CCA,CCG</i>	<i>58,293</i>	<i>128,177</i>	<b>4.4430</b>	<b>4.3811</b>
Ser	TCT,TCC,TCA,TCG,AGT,AGC	75,661	263,096	5.7668	8.9927
Thr	ACT,ACC,ACA,ACG	70,494	173,084	5.3730	5.9161
Trp	TGG	20,060	30,387	1.5290	1.0386
<b>Tyr</b>	<b>TAT,TAC</b>	<b>37,134</b>	<b>98,746</b>	<b>2.8303</b>	<b>3.3752</b>
Val	GTT,GTC,GTA,GTG	93,061	162,642	7.0930	5.5592

Amino acids encoded by AT-rich codons are in bold, and those encoded by GC-rich codons are italicized

**Table 9.11** Success rate (in percentage) of two surgical treatments for removing kidney stone: “all open procedure” (AOS) or percutaneous nephrolithotomy (PN), taken from Table 2 of Charig et al. (1986)

Size	AOS	PN
Small stones	93% (81/87)	87% (234/270)
Large stones	73% (192/263)	69% (55/80)
Pooled	78% (273/350)	83% (289/350)

Values in parenthesis are in the format of “Number of successes/number of patients treated.” Kidney stone size (Size) is discretized into two categories as in the original paper

codon usage bias contributes little to the rate of protein synthesis, as did by Kudla et al. (2009). Simpson’s Paradox, illustrated with data in Table 9.11, presents a similar case in which one would reach a wrong conclusion when one factor is ignored.

Charig et al. (1986) summarized their findings in the abstract on the basis of the last row of “Pooled” data, stating that “Success was achieved in 273 (78%) patients after open surgery, 289 (83%) after percutaneous nephrolithotomy.” A reader would have thought that AOS is worse (78% success rate) than PN (83% success rate). However, taking kidney stone size into consideration allows us to immediately reach

an opposite (and correct) conclusion, i.e., AOS is better than PN for both small stones (93% vs. 87%) and large stones (73% vs 69%). We also note that both AOS and PN have much higher success rate for small stones than for large stones. Patients treated with PN had mostly small stones and patients treated with AOS had mostly large stones. It is this association between PN and small stone that leads to the misleading conclusion that PN is better than AOS when kidney stone size is ignored.

# Chapter 10

## Bioinformatics and Translation

### Termination in Bacteria



## 1 Introduction

Many molecular mechanisms can be understood in light of coevolution between motif signals and their decoders. In previous chapters, we have illustrated how intron splice sites are optimized by their decoding spliceosomes (Wei et al. 2016; Wei and Xia 2017), the Shine-Dalgarno (SD) sequences are optimized for the anti-SD sequences (Abolbaghaei et al. 2017; Prabhakaran et al. 2015; Wei et al. 2017), and sense codon usage is optimized by differential tRNA availability (Xia 1998a, 2005, 2008, 2015). Fundamental understanding of the translation termination process can also be gained in this conceptual framework of motif signals (the stop codon and flanking nucleotides) and their decoders (release factors). Most of the illustrative examples in this chapter will come from bacterial species. Bioinformatic analyses presented in this chapter can be easily carried out by using DAMBE (Xia 2013, 2017d).

### 1.1 The Release Factors

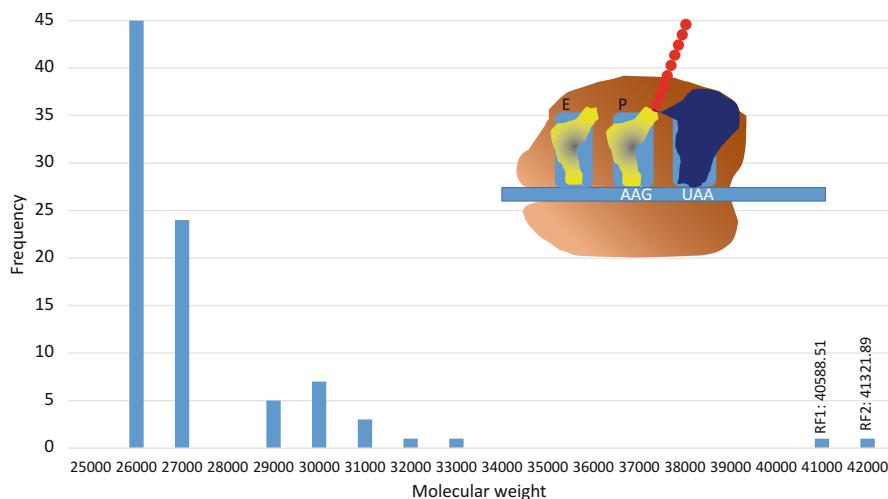
Release factors serve two functions, the decoding of translation termination signal (i.e., stop codon and flanking nucleotides that can enhance or attenuate the termination signal) and the release of the synthesized polypeptide from the peptidyl-tRNA. Associated with these two functions are two key structural elements, one acting as a peptide “anticodon” to decode the stop codon and the other with a landmark GGQ motif to help with peptidyl-tRNA hydrolysis.

### 1.1.1 The Discovery of Release Factors

When stop codons were found in T4 phage (Benzer and Champe 1962), it is natural to extrapolate from the knowledge accumulated since the discovery of tRNA (Hoagland et al. 1958) and assume that the stop codons were decoded by tRNA as well. There were also claims that stop codons might be recognized by rRNAs in similar manner as the SD/aSD pairing (Shine and Dalgarno 1974a). However, what sets science apart from fiction is that it critically distinguishes between the possible and the actual (Jacob 1982), and one key quality of scientist is to examine the existing model (i.e., tRNA decoding stop codons) critically.

What released the polypeptide chain in cell-free translation system for *Escherichia coli* was found in the supernatant. Capecchi (1967) fractionated the supernatant, added the different fractions to the system, and discovered the “release factor” that enabled the release of the polypeptide to be protein instead of RNA. The protein was determined to have a molecular weight of 40,000–50,000, larger than *E. coli* tRNAs (Fig. 10.1). As a side note, *E. coli* tRNAs do differ substantially in molecular weight, and those large ones would need to shrink more to fit the E, P, and A sites.

It is possible that the “release factor” may work with a tRNA-like element for stop codon recognition, or rRNA might be involved (Shine and Dalgarno 1974a). To check this possibility, RNase was added but was found to have no effect on the activity of the “release factor” (Capecchi 1967). So the “release factor” is a protein working alone without help from RNA. This finding was soon followed by the discovery that the “release factor” from *E. coli* B strain can be fractionated into two



**Fig. 10.1** Histogram of molecular weight of *E. coli* tRNAs, together with two release factors RF1 and RF2, and a schematic inset of two tRNAs occupying the E and P sites and a release factor occupying the A site

(R1 and R2), now known as RF1 and RF2, with specificities for different stop codons (Scolnick et al. 1968). RF1 decodes UAG and UAA, and RF2 decodes UGA and UAA (Milman et al. 1969; Scolnick et al. 1968; Scolnick and Caskey 1969).

These discoveries highlight one important point in scientific research. That is, rapid progress is often made after a plausible but false scientific hypothesis (e.g., tRNA decoding stop codons) is knocked down. A corollary of this is that a false hypothesis, if uncritically accepted, would impede progress in science.

While most tRNAs are similar in size and molecular weight, some are substantially larger. In *E. coli*, Leu, Ser, and Tyr tRNA sequences are longer and larger than others. This may have implications on decoding neighboring codons because a stretch of Leu, Ser and Tyr codons would usher in a set of large tRNAs that may cause local physical crowdedness in A, P and E sites.

The discoveries of release factors prompted biologists to ask a set of specific questions concerning the stop codon recognition and the specificity of recognition between RF1 and RF2. How do release factors recognize the stop codons? The observation that the release factors do not use a tRNA-like RNA element for anticodon recognition implies that the anticodon recognition is almost certainly done by a peptide motif. What is this motif and where in RF1 and RF2 is the motif located? What determines the specificity between the two release factors? In particular, what can bioinformatics help with finding the answers?

### 1.1.2 The Tripeptide “Anticodon” Model and Its Revision

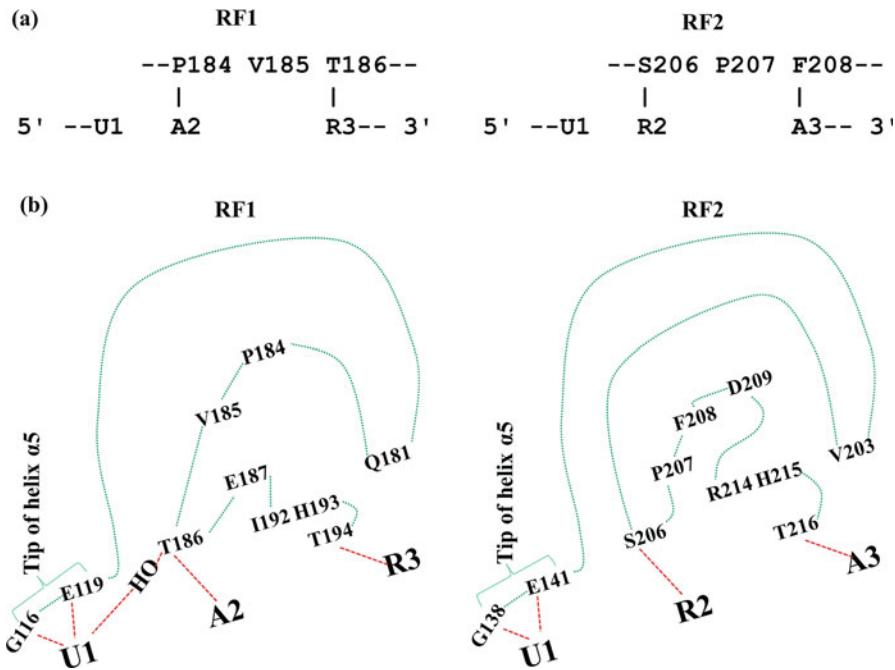
Bioinformatics did contribute to the answers concerning the stop codon recognition. By looking at RF1 and RF2 sequences, it becomes clear that they are paralogous genes (Fig. 10.2) with seven homologous domains designated A to G (Nakamura et al. 1996). They reasoned that the recognition motif must reside in one of these

V_cholerae_RF1	ESGGHRVQRVPATEAQGRIHTSACTVAVMPEIPEA-EIPEIKA-SDLKIDTFRSSGAGGQHVNTTD	CAIRITHLPT
E_colik12_RF1	EGCHHRVQRVPATEQGRRIHTACTAVAVMPEIPLDA-ELPDINPADLFRSSGAGGQHVNTTD	CAIRITHLPT
Th_maritima_RF1	EGGHRVQRVPTEQGRRIHTATAVAVLPEIEEK-DI-EIRPEDLKIETFRAGHGGQYVNVKTE	CAIRITHLPT
M_tuberculosis_RF1	EGGVHRVQRVPTEQGRVHTAAAGLVYYPEEEVGQV-QIDESDLRIDVFRSGKGQQGVNTTD	CAVRITHLPT
H_pylori_RF1	EAAGTHRVRQRVPETEQQGRRIHTAATVAIMPVEVDV-EV-INPSDLKIEVFRAGGHGGQCVNTTD	CAVRITHLPT
B_subtilis_RF1	ENGAHRVQRVPETEQQGRRIHTATVACLPPEAEEV-EV-DIHKEKD1RVDTFASCGPGQQ-VNTTD	CAVRITHLPT
Th_thermophilus_RF1	EGGVHRVQRVPVTEQQGRRIHTATAVALPKAEEE-DF-QLNMDDEIRIDVMRA-GPGQQGVNTTD	CAVRVHHLPT
E_colik12_RF2	ETGVHRLVLRSPFDGGRRHTSFSAFVYYPEVDDIDI-EINPADLIRDVYRTSGAGGQHVNRTE	CAVRITHIPT
V_cholerae_RF2	ETGVHRLVLRSPFDGGRRHTSFAAFIYYPEVDENIIDI-EINPADLIRDVYRASGAGGQHVNTTE	CAVRITHVPT
Th_maritima_RF2	EGGVHRLVRLSPFDAAARRHTFAVNVIPEIDDDVDI-EIRPEDLKIETFRAGHGGQYVNVKTE	CAVRITHLPT
H_pylori_RF2	ENGVHRLVRLSPFDANAKRHTFAVQTSPELDDIDI-EIDEKDVRDYYRSGAGGQHVNVKTE	CAVRITHPT
M_tuberculosis_RF2	EQGTHRLVRLSPFDNQRRQTFAVEVELPVPEVETTDHI-DIPEGDVRVVDYRSGPGQQ-VNTTD	CAVRTHIPS
B_subtilis_RF2	EKGVHRLVRLSPFDSSGRRHTFVCEVMPPEFNDEDI-DIRTEDIKVDTYRAGAGGQHVNTTD	CAVRVHPT
Th_thermophilus_RF2	EAGVHRLVRLSPFDAGRRHTFAGCVEVPEVDEEVEV-VLKPEEIRIDVMRA-GPGQQGVNTTD	CAVRVHPT
* * * * *	* * * * *	* * * * * * * * *

**Fig. 10.2** Alignment of a section of amino acid sequences for a set of highly diverged bacterial species. The first shaded tripeptide (PXT in RF1 and SPF in RF2) was proposed to be the tripeptide “anticodon” by Ito et al. (2000). The second shaded tripeptide (GGQ) mimics the CCA end of a tRNA. Sites identical in both RF1 and RF2 are indicated by “\*.” G116 and E119 in RF1 (not shown) are invariant among the seven species. In *Thermus thermophilus*, PXT in RF1 starts at site 184, SPF in RF2 starts at site 206, and GGQ starts at sites 228 in RF1 and 251 in RF2

domains and consequently designed experiments to swap these domains between RF1 and RF2 (Ito et al. 2000). If a domain from RF1 replaces the equivalent domain in RF2 and if the recognition motif resides within the domain, then RF2 will decode UAA and UAG as the wild-type RF1 does. Similarly, if a domain from RF2 replaces the equivalent domain in RF1 and if the recognition motif resides within the domain, then RF1 will decode UAA and UGA as the wild-type RF2 does. In this way, they identified domain D to be responsible for recognition and then swapped subsections within domain D.

The progressive swapping of smaller gene segments eventually led them to propose a tripeptide “anticodon” as PXT in RF1 and SPF in RF2 (Fig. 10.3a). For the tripeptide “anticodon” PXT in RF1, P184 interacts only with A2 (in stop codon), but T186 can interact with both A and G. So it can decode UAA and UAG. The amino acid at site 185 (Fig. 10.3a) does not contribute directly to the recognition and is therefore not conserved, in contrast to P184 and T186 which are highly conserved. For the tripeptide “anticodon” SPF in RF2, S206 can interact with either A2 or G2 in stop codon, but F208 can interact only with A3 in stop codon (Fig. 10.3a), so SPF



**Fig. 10.3** Schematic drawing of stop codon recognition by release factors RF1 and RF2, with the three codon sites numbered 1, 2, and 3, respectively, and amino acid residue numbered according to RF1 and RF2 in *Thermus thermophilus*. R stands for A or G. (a) Tripeptide anticodon PXT (where X is A, V, or E) in RF1 and SPF in RF2 proposed by Ito et al. (2000). (b) Interaction between stop codon and amino acids based on protein structure (Sund et al. 2010; Zhou et al. 2012)

can decode only UAA and UGA and is invariant in RF2 alignment among the seven divergent species (Fig. 10.2). Throughout this chapter, the amino acid numbering in RF1 and RF2 follows that in *Thermus thermophilus*, unless otherwise specified.

There are two shortcomings in the “anticodon” proposal above. First, the proposal is not substantiated with structural details. Second, it does not explain how U1 in the stop codon is recognized. Subsequent structural studies (reviewed in Sund et al. 2010; Zhou et al. 2012) confirmed the existence of a recognition motif but refined the tripeptide anticodon (Fig. 10.3b). For RF1, only T186 in the originally proposed PXT (Ito et al. 2000) interacts with A2. It is T194 that interacts with A3 and G3, aided by Q181 (Fig. 10.3). U1 in stop codon is decoded by G116 and E119 against C1 (so that CAA will not be misread by RF1 as a stop codon). G116 and E119 are invariant in RF1 in the alignment among the seven highly divergent bacterial species in Fig. 10.2, so are T186, T194, and Q181, suggesting that the interaction between amino acids in RF1 and nucleotides in stop codons is highly conserved.

For RF2 (Fig. 10.3), G138 and E141 interact with U1 in stop codon against C1. In the proposed SPF tripeptide (Ito et al. 2000), only S206 interacts with A2 or G2 in stop codon. T216 interacts with A3 and was helped by V203. All these involved amino acid sites, except for E141, are invariant in the RF2 alignment among the seven divergent species in Fig. 10.2. The E141 site is occupied by an aspartic acid (D in one-letter code) in *Mycobacterium tuberculosis*. Amino acids D and E are both negatively charged and often replace each other without affecting protein function.

One major problem with structural studies with release factors in complex with mRNA is that the mRNA is truncated at the stop codon with no downstream nucleotides. There is strong nucleotide bias in the nucleotides immediately downstream of the stop codon (Brown et al. 1990; Poole et al. 1995; Tate and Brown 1992; Wei and Xia 2017). It would be nice to see which amino acid residue is in contact with the downstream nucleotides and if that amino acid residue is strongly conserved.

### 1.1.3 The GGQ Motif

All tRNAs as adapters share two essential features, an anticodon for decoding sense codons and an acceptor stem with a CCA terminal group for attaching an amino acid. The success of identifying the peptide “anticodon” prompted researchers to search for the equivalent of CCA end that would help with releasing the polypeptide by peptidyl-tRNA hydrolysis. The GGQ motif shared in both RF1 and RF2 (Fig. 10.2) is a likely candidate, and it is conserved among all three kingdoms of life (Mora et al. 2003). Mutations of GGQ abolish the function of peptidyl-tRNA hydrolysis in human eEF1 (Frolova et al. 1999) and in *E. coli* RF1 and RF2 (Mora et al. 2003). This led to the molecular mimicry hypothesis that release factors mimic tRNA with a peptide “anticodon” mimicking tRNA anticodon and GGQ mimicking the CCA end of tRNA, first suggested by Nissen et al. (1995).

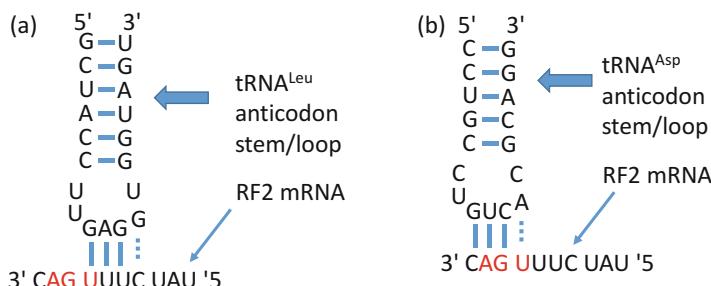
The distance between tRNA anticodon and the CCA terminal group is about 75 Å, so one expects the distance between the peptide “anticodon” and GGQ motif to

have a similar distance. Surprisingly, the distance from the first protein structure for RF2 is only 23 Å (Vestergaard et al. 2001), which is too close relative to the expected 75 Å (Mora et al. 2003; Vestergaard et al. 2001). Should we abandon the molecular mimicry hypothesis? Are there some alternatives to keep the hypothesis? This is where one needs to be creative and original. One possibility is that the RF2 structure when attached to ribosome could be different (Ehrenberg and Tenson 2002). This is somewhat supported by structural simulation (Ma and Nussinov 2004) which shows the distance to extend from 23 Å to 50 Å. The definite answer comes from structural of RF1 and RF2 in complex with ribosome, with GGQ and the tripeptide codon located almost exactly at the equivalent positions of the CCA end and anticodon of P-site tRNA (Petry et al. 2005; Vestergaard et al. 2005). Such studies highlight the shortcoming of a protein structure in isolation from its interacting partners.

## 1.2 Stop Codons as Part of the Stop Signal

The most significant signal for translation termination is the stop codon, with UAG and UAA decoded by RF1 and UGA and UAA decoded by RF2. However, two factors are known to contribute to the signal strength: the flanking nucleotide (Tate and Brown 1992; Tate and Mannerling 1996) and secondary structure that may embed the stop codon and render it inaccessible. The first hint that the efficiency of translation termination depends on nucleotides flanking the stop codon is the +1 frameshifting during RF2 mRNA translation in *E. coli* (Craigen and Caskey 1986; Craigen et al. 1985). *E. coli* RF2 mRNA has an inframe stop codon UGA at the 26th codon site, and a +1 shift to skip U in the UGA is needed to produce a functional RF2.

Empirical evidence (Curran and Yarus 1988; Weiss et al. 1988) implicates flanking nucleotides/codons in facilitating the +1 skipping. When RF2 concentration is low, the stop codon UGA is not immediately decoded by RF2, and tRNA<sup>Leu</sup> with anticodon GAG decoding the upstream codon CUU slips and wobble-pairs to CUUU (Fig. 10.4a) leading to +1 skip. Alternatively, when the stop codon UGA and the downstream C are decoded by tRNA<sup>Asp</sup> (Fig. 10.4b), also with tetranucleotide



**Fig. 10.4** Programmed +1 frameshift in translating *E. coli* RF2 mRNA mediated by tRNA. (a) Tetranucleotide decoding by tRNA<sup>Leu</sup>, (b) tetranucleotide decoding by tRNA<sup>Asp</sup>

base pairing. If the downstream C is replaced by a purine, then frameshifting is reduced. In general, C is strongly avoided immediately downstream of UGA stop codon (Brown et al. 1990; Poole et al. 1995; Tate and Brown 1992; Wei and Xia 2017).

The +1 frameshift provides RF2 with an autoregulation (Betney et al. 2010). When RF2 concentration is low, UGA is not immediately decoded, giving tRNA<sup>Leu</sup> and tRNA<sup>Asp</sup> time to mediate +1 frameshifting to produce a functional RF2. When RF2 is high, UGA is immediately decoded by RF2, and only a nonfunctional short peptide of length 25 is produced. It is interesting to note that polyamines enhanced +1 frameshift in RF2 in *E. coli* (Higashi et al. 2006) and that polyamines also enhances translation of certain mRNAs (Igarashi and Kashiwagi 2006).

It has now become well established that the flanking nucleotides, especially the nucleotide immediately downstream of the stop codon referred to as the +4 site, have significant effect on translation termination efficiency, not only for UGA but also for other stop codons (Betney et al. 2010; Wei and Xia 2017). For those of us taking a bioinformatic approach to study the effect of flanking nucleotides, it is important to keep in mind that the most frequently used flanking nucleotide may not necessarily have a positive impact on the stop codon recognition. It could be due purely to mutation bias but could be due to other constraints. In a previous chapter on translation initiation, we have already presented the case in which the preponderance of +4G in Kozak's consensus RxxAUGG may be due to either a positive effect on start codon recognition or to the requirement of the second amino acid being small and nonpolar (Xia 2007a). However, if we found a particular nucleotide such as +4U following UAA to be much more represented in highly expressed genes than lowly expressed genes (Tate and Brown 1992; Wei and Xia 2017), then it is a strong indication that UAAU is favored by natural selection as an efficient translation termination signal. However, a strong indication is often insufficient in science. The following sections outline the approaches for us to go further.

I wish to make a point before ending this section. In science, we often extrapolate from what is known to what is not known. Sometimes extrapolation is helpful and sometimes not. The extrapolation from what we know about tRNA decoding sense codons to the idea that the “release factor” should also be a tRNA would have hindered progress in molecular biology. On the other hand, the knowledge that tRNA as an adapter molecule should have an anticodon and an amino acid attachment site located about 75 Å has been very fruitfully extrapolated to infer the protein structure of release factors. Thus, in science, any bold extrapolation has to be followed by meticulous verification.

## 2 Three Key Factors Affecting Stop Codon Usage

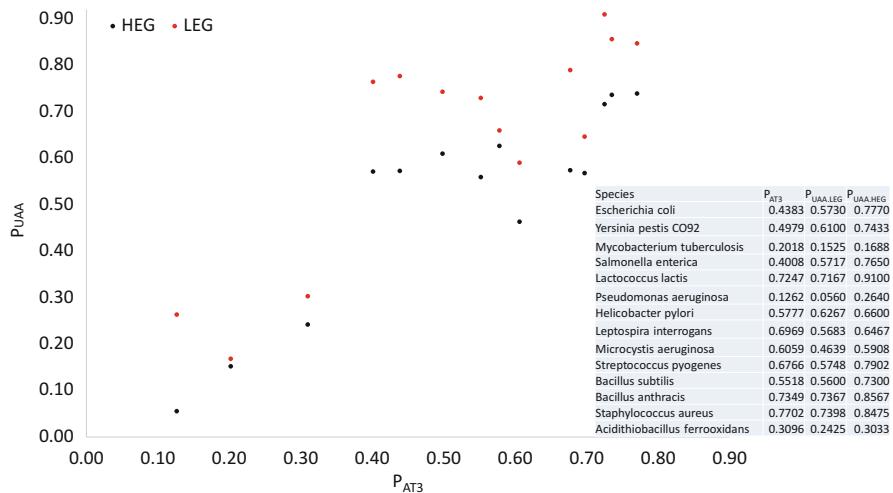
In a previous chapter on translation elongation, we learned of two key factors contributing to sense codon usage: (1) mutation bias and (2) tRNA-mediated selection. If a genome has been experiencing GC-biased mutation, then the third codon

position tends to be G-ending or C-ending. If a protein is mass-translated, its mRNA tends to code amino acids by codons decoded by the most abundant tRNA. One additional factor that is important but has not been studied in any detail is misread of sense codons by release factors. For example, high abundance of RF2 (decoding UAA and UGA but occasionally misreading UGG as a stop signal) may select against UGG usage. The effect of all these factors can be extrapolated to understanding stop codon usage.

## 2.1 Genomic Mutation Bias and Codon Usage

Stop codon usage, just like sense codon usage, depends partially on genomic mutation bias. One would expect that GC-rich genomes would have more UGA and UAG codons relative to UAA and the opposite with AT-rich genomes. Thus, if we define  $P_{UAA}$  as the proportion of UAA stop codons, then  $P_{UAA}$  indeed increases with genomic AT content or with proportion of AT at the third codon position of all protein-coding genes ( $P_{AT3}$ ), as shown in Fig. 10.5. The same trend is clear for both highly expressed and lowly expressed genes (HEGs and LEGs, respectively). However, HEGs always have higher usage of UAA than LEGs. As we will learn soon, UAA is preferred by highly expressed genes even in GC-rich genomes where stop codon UGA is far more frequently than UAA.

It is not surprising that UAA is the preferred stop codon by HEGs because all experimental evidence points to UAA as the most efficient and accurate translation



**Fig. 10.5** Proportion of UAA codons ( $P_{UAA}$ ) increases with AT-biased mutation measured as the proportion of AT in the third codon position of all protein-coding genes ( $P_{AT3}$ ). Highly expressed and lowly expressed genes (HEGs and LEGs, respectively) are plotted separately. The actual data for the plot, derived from 14 bacterial species with sequenced genomes, is included

termination signal. All three stop codons can be misread by tRNAs, and UGA appears to be the leakiest of the three, with a readthrough frequency of at least  $10^{-2}$  to  $10^{-3}$  in *Salmonella typhimurium* (Roth 1970) and *E. coli* (Sambrook et al. 1967; Strigini and Brickman 1973). UAA and UAG can also be leaky in bacteria (Davies et al. 1966; Ryden and Isaksson 1984), although their misreading has not been reported as frequently as UGA. Natural UAG readthrough frequency is mostly within the range of  $1.1 \times 10^{-4}$  to  $7 \times 10^{-3}$ , depending on the nature of the downstream nucleotides (Bossi 1983; Bossi and Ruth 1980; Miller and Albertini 1983; Ryden and Isaksson 1984). The readthrough of UAA seems to occur at frequencies from  $9 \times 10^{-4}$  to less than  $1 \times 10^{-5}$  (Ryden and Isaksson 1984). Overall, the available experimental data suggest that in bacteria species, particularly in *E. coli*, readthrough is most frequent for UGA, less for UAG, and least for UAA (Cesar Sanchez et al. 1998; Geller and Rich 1980; Jorgensen et al. 1993; Meng et al. 1995; Parker 1989; Strigini and Brickman 1973; Tate et al. 1999).

The mutation effect on stop codon usage is mainly studied through genomic GC content which has a strong effect on stop codon usage based on data from 736 species (Povolotskaya et al. 2012). The effect of genomic GC bias on stop codon usage has also documented in a study involving 4684 genomes (Korkmaz et al. 2014). However, the effect of mutation bias on stop codon usage is confounded by gene expression and relative abundance of RF1 and RF2 (Wei et al. 2016), as well as by nucleotides flanking the stop codons (Wei and Xia 2017). UGA is always avoided in HEGs relative to LEGs (Wei et al. 2016). This is true even in species where UGA accounts for an overwhelming majority of stop codons and RF2 is far more abundant than RF1. In such species, UAA is mostly found in HEGs.

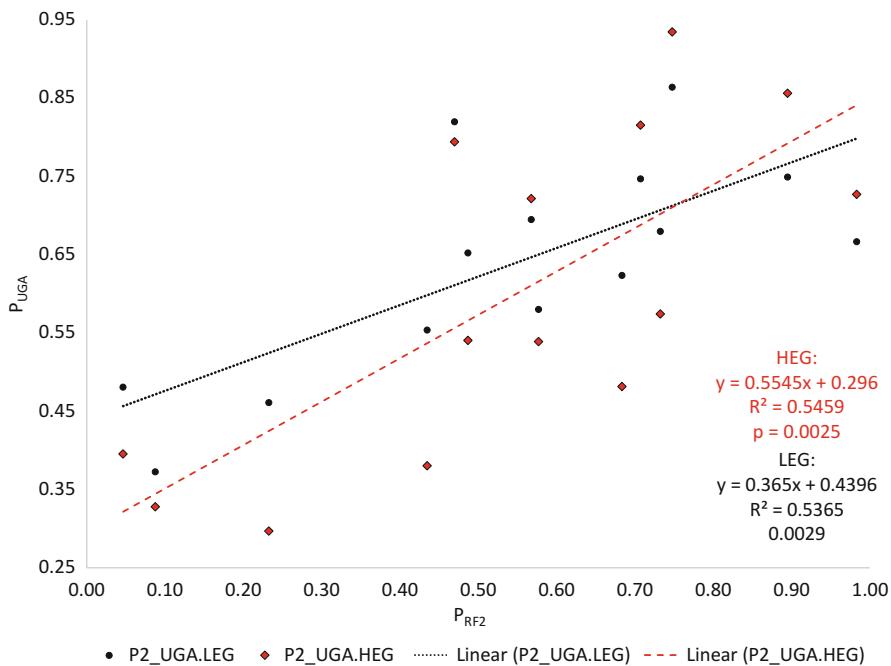
## 2.2 Differential Abundance of Release Factors and Stop Codon Usage

Because UGA is decoded only by RF2 and UAG only by RF1, one expects UGA to be used more than UAG when RF2 concentration is higher than RF1 (assuming that the two release factors have equal decoding efficiency on their respective codons). This is consistent with *E. coli* data where RF2 is about five times more frequent than RF1 (Adamski et al. 1994; Mora et al. 2007) and UGA is used much more frequently than UAG.

We may define

$$P_{\text{UGA}} = \frac{N_{\text{UGA}}}{N_{\text{UGA}} + N_{\text{UAG}}}, \quad P_{\text{RF2}} = \frac{N_{\text{RF2}}}{N_{\text{RF1}} + N_{\text{RF2}}} \quad (10.1)$$

where  $N_{\text{UGA}}$  and  $N_{\text{UAG}}$  are the number of genes with stop codons UGA and UAG, respectively, and  $N_{\text{RF1}}$  and  $N_{\text{RF2}}$  are protein abundance of RF1 and RF2, respectively,



**Fig. 10.6** Stop codon UGA usage, measured by  $P_{UGA}$ , increases with relative RF2 abundance measured by  $P_{RF2}$  (The data are from the 14 species in Fig. 10.5)

and can be obtained from PaxDb (Wang et al. 2012). The prediction, as an extension from the codon adaptation theory in a previous chapter on translation elongation, is that stop codon usage should increase with the abundance of its decoder. That is,  $P_{UGA}$  and  $P_{RF2}$  values have been obtained for the 14 species in Fig. 10.5 (Wei et al. 2016). The positive relationship between  $P_{UGA}$  and  $P_{RF2}$  (Fig. 10.6) is consistent with the hypothesized relationship.

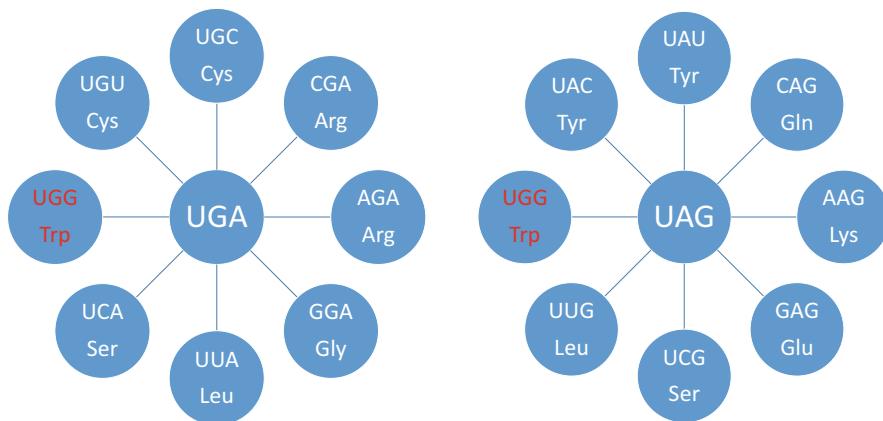
RF2 is more abundant than RF1 over a wide range of AT content but decreases rapidly toward zero at extreme AT richness (Wei et al. 2016), which implies that AT-rich genomes should have few UGA or not at all for two reasons. First, the AT-biased mutation would favor UAA against UGA. Second, the rapid decline of RF2 toward zero also selects against UGA usage. For AT-rich genomes such as those in *Mycoplasma* species, UGA is reassigned to encode amino acid Trp.

It is important to use protein abundance data to obtain  $P_{RF2}$  instead of using relative mRNA abundance data as done previously (Korkmaz et al. 2014) for two species, *B. subtilis* and *Mycobacterium smegmatis*. More *prfB* mRNA than *prfA* mRNA does not imply more RF2 than RF1 because RF2 is translationally regulated (Craigen et al. 1985; Donly et al. 1990), by the autoregulation mechanism that we have discussed already.

### 2.3 Near-Cognate tRNA and Stop Codon Usage

Through a single-nucleotide replacement, stop codons UGA and UAG can each mutate to UAA or to eight sense codons (Fig. 10.7). I will refer to these sense codons as UGA-like codons and UAG-like codons. A tRNA with an anticodon that can form perfect Watson-Crick base pair with a UGA-like codon or a UAG-like codon will be called a UGA-near-cognate or UAG-near-cognate tRNA. Anticodon in such a tRNA can base-pair with UGA and UAG by wobbling at one site and therefore can potentially misread a stop codon leading to a readthrough error. However, these near-cognate tRNAs don't have the same readthrough rate. For example, tRNA<sup>Trp</sup> (decoding UGG) results in readthrough much more frequently for stop codon UGA and UAG.

All three stop codons can be misread by tRNAs, and UGA appears to be the leakiest of the three. However, there are discrepancies in estimated readthrough rate. This discrepancy is largely attributable to the following five factors, with the first two involving sequence features and the last three involving experimental environment. First, stop codon readthrough, especially in UGA and UAG, depends on the downstream nucleotides (Bossi 1983; Bossi and Ruth 1980; Miller and Albertini 1983; Poole et al. 1998; Tate and Mannerling 1996). For example, UGA is particularly prone to be misread by tRNA<sup>Trp</sup> when followed by an A (Engelberg-Kulka 1981) in both *E. coli* and *S. cerevisiae* (Beznoskova et al. 2016) and is prone to frameshifting as a means of gene regulation (Craigen and Caskey 1987; Craigen et al. 1985, 1990; Donly et al. 1990) when followed by CU in *E. coli*. Codon context also affects usage of sense codons (Bossi and Ruth 1980; Carlini 2005; Eyre-Walker 1996; Eyre-Walker and Bulmer 1993; Gouy 1987; Jia and Higgs 2008; Shpaer 1986). Second, amino acid upstream of the stop codon or the peptidyl-tRNA occupying the P site can affect readthrough frequency in *E. coli* (Mottagui-Tabar and Isaksson 1997; Zhang et al. 1996). Third, near-cognate tRNAs and release factors compete for stop



**Fig. 10.7** Stop codons UGA and UAG each can mutation into eight sense codons through a single-nucleotide replacement. UGG is both a UGA-like and a UAG-like codon

codon recognition and their relative abundance affect readthrough frequencies (Nakamura et al. 1995; Tate et al. 1999). Fourth, experimental conditions such as temperature can affect readthrough frequency in both *B. subtilis* and *E. coli* (Gonzalez et al. 2003). Fifth, the same bacterial strain at different growth phases can have different readthrough rate (Wenthzel et al. 1998).

While early studies suspect that high-level suppression of stop codons is due to mutant tRNA genes, wild-type tRNA<sup>Trp</sup> was soon found to read UGA codons in vitro (Hirsh and Gold 1971). A natural protein from a coliphage was found to have tryptophan at the UGA site (Weiner and Weber 1973), and a large number of naturally occurring eukaryotic viral proteins with stop codon readthrough have been reported (Beier and Grimm 2001). Similar readthrough of UGA by tRNA<sup>Trp</sup> has also been reported in mammalian translation machinery (Geller and Rich 1980) and in eukaryotic viruses (Beier and Grimm 2001). Tryptophan (UGG), arginine (CGA), cysteine (UGY), or serine (UCA) can be incorporated at UGA stop codons in yeasts and higher eukaryotes (Blanchet et al. 2014; Eswarappa et al. 2014), suggesting that these near-cognate tRNAs can misinterpret a stop codon as a sense codon, with tRNA anticodon wobble pairing at the third (UGG, UGY), second (UCA), and first (CGA) codon sites. Wobbling at the second codon position is rare but does happen. In *E. coli*, arginine CGG codons were known to be misread as CAG by tRNA<sup>Gln</sup> (McNulty et al. 2003) and arginine AGA codons misread as AAA by tRNA<sup>Lys</sup> (Calderone et al. 1996; Seetharam et al. 1988; You et al. 1999).

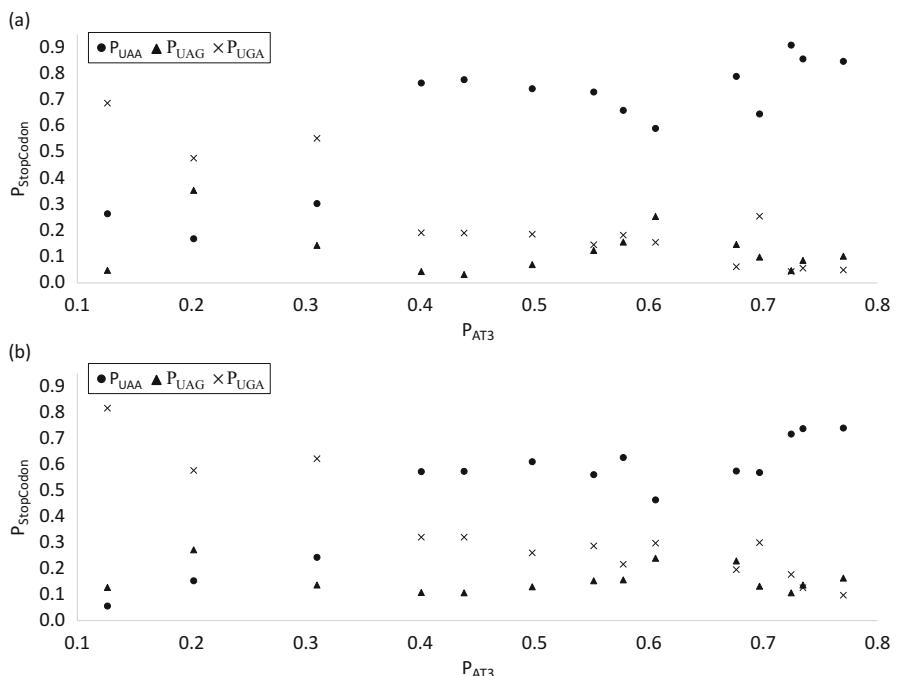
UAA and UAG can also be leaky in bacteria (Davies et al. 1966; Ryden and Isaksson 1984), and in vitro experiments suggest that tRNA<sup>Gln</sup> (decoding CAA and CAG) and tRNA<sup>Glu</sup> (decoding GAA and GAG) are two major culprits (Davies et al. 1966; Petrullo et al. 1983; Prival 1996; Yoshinaka et al. 1985), although other amino acids such as Leu, Ser, and Tyr can also be incorporated at the site of UAA or UAG (Winston et al. 1979). Misreading UAA for glutamine also occurs in the yeast, *Saccharomyces cerevisiae*, where overexpressed tRNA<sup>Gln/UUG</sup> suppresses UAA (Pure et al. 1985) and tRNA<sup>Gln/CUG</sup> suppresses UAG (Blanchet et al. 2014; Lin et al. 1986). The similarity between UAR and CAR codons is accentuated by the fact that all four are used to code Gln in Ciliate, Dasycladacean, and *Hexamita* (Keeling and Doolittle 1996). Natural UAG readthrough frequency is mostly within the range of  $1.1 \times 10^{-4}$  to  $7 \times 10^{-3}$ , depending on the nature of the downstream nucleotides (Bossi 1983; Bossi and Ruth 1980; Miller and Albertini 1983; Ryden and Isaksson 1984). The readthrough of UAA seems to occur at frequencies of  $9 \times 10^{-4}$  to less than  $1 \times 10^{-5}$  (Ryden and Isaksson 1984).

The available experimental data suggest that in bacteria species, particularly in *E. coli*, readthrough is most frequent for UGA, less for UAG, and least for UAA (Geller and Rich 1980; Jorgensen et al. 1993; Parker 1989; Strigini and Brickman 1973; Tate et al. 1999). The frequent use of UAA in *E. coli* and *B. subtilis* has been attributed to its accuracy as well as its property of being decoded by both RF1 and RF2 (Sharp and Bulmer 1988). In *Salmonella typhimurium*, the suppression of UGA and UAG stop codons is frequent but rare for UAA (Hughes 1987). When a human gene with stop codon UGA is expressed intracellularly in *E. coli*, readthrough occurs frequently, but replacing UGA by UAA eliminates readthrough (Cesar Sanchez et al. 1998). A similar

study involving a human gene expressed in *E. coli* shows that frequent readthrough associated with UGA is eliminated when UGA is replaced by either UAA or UAG (Meng et al. 1995). Given the same codon context, RF2 decoding UGA is less efficient than RF1 decoding UAG in *E. coli* (Bjornsson and Isaksson 1996).

## 2.4 Why UAG Is Rarely Used as Stop Codons?

One conspicuous pattern observed previously (Korkmaz et al. 2014; Wei et al. 2016) is that UAA usage increases and UGA usage decreases, with AT content, but UAG usage remains low and hardly changes with AT content. This pattern is also visible in the 14 species here (Fig. 10.8). Korkmaz et al. (2014) interpreted this pattern as consistent with UAG being a minor codon that has translation termination efficiency and accuracy problems and is therefore nearly universally avoided. This interpretation by Korkmaz et al. (2014) is somewhat far-fetched for two reasons. First, experimental evidence, as reviewed before, suggests that UAG is typically more efficient and accurate than UGA as a termination signal. Second, UAG is favored by



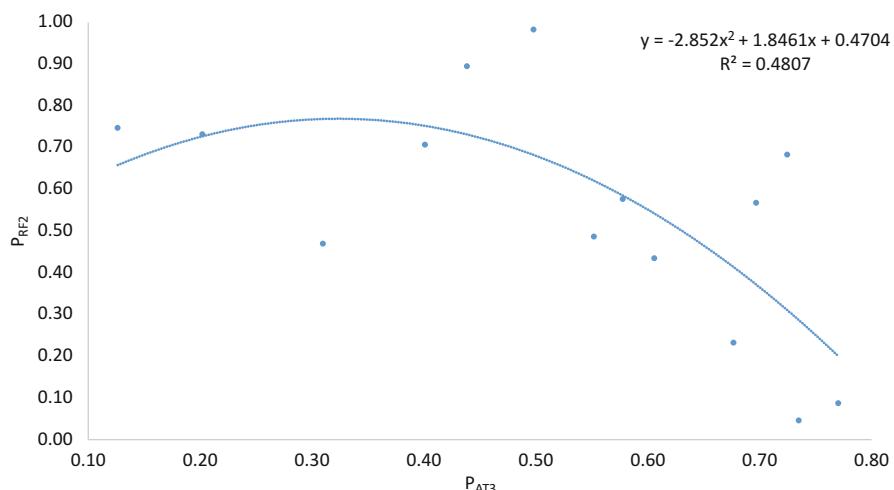
**Fig. 10.8** UAA usage increases and UGA usage decreases, with  $P_{AT3}$ , but UAG usage is low and changes little with  $P_{AT3}$ . The pattern is consistent in both highly expressed genes (a) and lowly expressed genes (b) (The data are from the 14 species in Fig. 10.5. From Wei et al. (2016))

HEGs in 3 of the 14 species, whereas UGA is avoided by HEGs in all 14 species (Wei et al. 2016).

A coherent explanation (Wei et al. 2016) has been advanced to explain the consistently low UAG usage over the entire range of AT content in bacterial genomes. At the low  $P_{AT3}$  range, mutation would have favored both UAG and UGA at the cost of UAA. However,  $P_{RF2}$  is high with low  $P_{AT3}$  (Fig. 10.9) which would favor UGA and select against UAG, keeping the latter at low frequency. At high  $P_{AT3}$ , mutation would favor UAA against UGA and UAG, and we expect low usage of both UGA and UAG. However,  $P_{RF2}$  approaches 0 in species with high  $P_{AT3}$  (Fig. 10.9), which selects strongly against UGA codons but little against UAG codons (as RF1 becomes the dominant release factor at high  $P_{AT3}$ ). This explains why, at high  $P_{AT3}$ , UAG does not decrease as much as UGA in Fig. 10.8a and tend to have its usage higher than that of UGA. In the midrange of  $P_{AT3}$ , UAA is overused because (1) it is favored by selection and (2) there is no mutation bias against it. Also in this range,  $P_{RF2}$  is still much greater than 0.5 (Fig. 10.9), favoring UGA against UAG and keeping the latter at low frequency. So UAG usage is kept low and changes little over the entire range of  $P_{AT3}$ .

The low usage of UAG as a stop codon led Korkmaz et al. (2014) to think that “TAG is truly a minor stop codon in all aspects.” Designating codons as major and minor codons are important not only in understanding the function of the translation machinery but also in biopharmaceutical industry as many experimental studies have shown that replacing minor codons by major codons increases protein production (Haas et al. 1996; Ngumbela et al. 2008; Robinson et al. 1984; Sorensen et al. 1989).

The term “major (or minor) codon” is often misunderstood. “Major codon” (or optimal codon) originally refers to sense codons preferred by highly expressed



**Fig. 10.9** Relative abundance of RF2 decreases rapidly at high range of AT content, measured by proportion of AT at the third codon site ( $P_{AT3}$ ) (The data are from the 14 species in Fig. 10.5)

genes and decoded by the most abundant tRNA. It is first used by McPherson (1988) in reference to a study (Kurland 1987) showing that highly expressed genes use codons to optimize decoding efficiency of the tRNA pool. A minor codon is the opposite. Major and minor codons are not necessarily the most frequent or least frequent codons when compilation is done for all genes.

Two criteria, one essential and one corroborative, have been used, sometimes implicitly, to identify a minor sense codon. The essential criterion is that a minor codon is the most strongly avoided in HEGs. The corroborative criterion is that a minor codon corresponds to the least abundant tRNA among synonymous codons. Without these two criteria, a minor codon could be identified incorrectly (Xia 2015). For example, if we compile the codon frequencies of Asp codon family for all genes in *E. coli* (NC\_000913), we will get 41,806 GAU and 25,015 GAC, which would mislead us to conclude that GAU is the major codon and GAC the minor. LEGs use more GAU than GAC, but HEGs use more GAC than GAU. Furthermore, these Asp codons are translated by three tRNA<sup>Asp</sup> genes all with the same GUC anticodon forming perfect base pair with GAC. Thus, both criteria support GAC as the major (optimal) codon and GAU as the minor.

We may use the same two criteria for identifying major and minor stop codons by replacing tRNA abundance by release factor abundance. Take *Microcystis aeruginosa*, for example. LEGs use more UGA than UAG as stop codons in this species, but HEGs use more UAG than UGA (Wei et al. 2016). This stop codon usage pattern is consistent with the relative RF1 and RF2 concentrations compiled in the integrated data set available in PaxDb (Wang et al. 2012). Protein abundance is 33.3 ppm (parts per million) for RF1 and 18.2 ppm for RF2 in that integrated data set. The average concentration of RF1 is also higher than RF2 based on multiple separate measurements (Wei et al. 2016). Thus, UAG has more decoders than UGA and is expected to be more preferred than UGA by HEGs. So UAG clearly is not a minor stop codon in *M. aeruginosa*.

### 3 Quantification of Translation Termination Efficiency

We have outlined many factors that could affect translation termination efficiency. In order to model the effect of these factors on translation termination efficiency, we need to have an objective measure of translation termination efficiency ( $E_{TT}$ ). Modern ribosomal profiling data (Ingolia 2010, 2014, 2016; Ingolia et al. 2009) based on deep sequencing allows us to “visualize” translation termination. About 30 nt of mRNA is protected by ribosome binding. Sequencing all these 30 mers and mapping them to genes let us know exactly where the translating ribosomes are located along individual mRNAs. If we see a lot of ribosomes traveling down an mRNA, but all these ribosomes abruptly disappeared at the point of the stop codon, then the mRNA has a high  $E_{TT}$ . In contrast, if we see many ribosomes travel down beyond the stop codon far into the 3'UTR of an mRNA, then the mRNA has a low  $E_{TT}$ .

We can use ribosome profiling data to objectively measure  $E_{TT}$  as a function of ribosome density on mRNA before the stop codon ( $D_b$ ) and after the stop codon ( $D_a$ ). Thus defined,  $E_{TT}$  can be expressed as

$$E_{TT} = 1 - \frac{D_a}{D_b} \quad (10.2)$$

## Postscript

It is important to keep in mind that our coverage of translation termination is applicable to most bacterial species, but not all. For example, some bacteria do not even have a functional RF2 gene. Blind extrapolation and hasty generalization could lead to absurdity. Jacques Monod asserted that “what is true for the colon bacillus is true for the elephant” (Jacob 1988, p. 290), but in reality what is true in colon bacillus is not even true in other bacteria.

This example reminded me of an anecdote involving the late Chinese premier Zhou Enlai that highlights the importance of distinguishing part and whole. After the establishment of the People’s Republic of China, the government launched a nationwide campaign against pornography and prostitution. The effort resulted in a complete eradication of prostitution in mainland China. In a news conference celebrating this landmark socialist achievement, a western reporter suddenly asked Premier Zhou if prostitution still existed in China. All Chinese officials present were surprised to hear Premier Zhou answering “Yes,” and all applauded when he added that “They are in Taiwan.” It turned out that the western reporter was hoping that Premier Zhou would answer “No” given the special occasion so that he could then claim that Chinese leadership did not consider Taiwan as part of China. A clear perception of part and whole by Premier Zhou saved the day.

# Chapter 11

## Genomic Features: Content Sensors, Nucleotide Skew Plot, Strand Asymmetry, and DNA Methylation



### 1 Introduction

Content sensors are typically frequencies of various kinds, e.g., nucleotide, dinucleotide, trinucleotide frequencies, etc. In hidden Markov models covered in a previous chapter, these frequencies given a hidden state are the emission probabilities given the underlying hidden states. In a genomic sequences, the hidden states could be exon, intron, intergenic sequences, etc. For example, triplet frequencies or nucleotide frequencies at the three different positions of the triplets are often different between exons and introns, and such differences in content sensor (or emission probability) contribute to the power of gene-predicting algorithms.

The term content sensor is used in contrast to signal sensor which is a motif whose signal depends on site-specific information, such as the yeast 5' splice site (GUAUGU) or branchpoint site (UACUAAC). The signal in 5' splice site or branchpoint site is lost in most cases if we randomly reorder the nucleotides. However, nucleotide frequencies (content sensors) do not change.

Once a sequence is compiled into frequency tables, the site-specific information is lost. Any sensor that does not include site-specific information (in cases where site specificity is not important) is a content sensor and is the topic in this chapter. Sometimes we may use a splicing window to see changes in nucleotide frequencies or strand biases along genomic or chromosome sequences. In this case, the site-specific information within the window is lost, but different windows still have their specific start site. Such a sliding-window approach reveals the change in content sensor along the genome.

This chapter is organized into three parts to illustrate content sensors and their applications in bioinformatic research: (1) the GC skew frequently used to characterize strand bias and to identify the origin of replication and (2) content sensors associated with DNA methylation. DNA methylation is a process that simply cannot be avoided in a book with the word “cell” in its title. In short, DNA methylation plays a key role in gene regulation in many vertebrate species and is a ubiquitous

biochemical process particularly pronounced in vertebrate genomes, with its main function being tissue-specific gene regulation (Bestor and Coxon 1993; Rideout et al. 1990; Sved and Bird 1990). A typical representative of the vertebrate methyltransferase is the mammalian DNMT1 (mainly for methylation pattern maintenance) with five domains of which the NIISD, ZnD, and CatD domains bind specifically to unmethylated CpG, methylated CpG, and hemimethylated CpG sites, respectively (Fatemi et al. 2001). Methylation of C (at its #5 carbon atom) in the CpG dinucleotide greatly elevates the mutation rate of C to T through spontaneous deamination of the resultant m<sup>5</sup>C (Brauch et al. 2000; Tomatsu et al. 2004), where the superscript 5 indicates the #5 carbon atom, generating strong footprints in both prokaryotic and vertebrate genomes. Various nucleotide skews and DNA methylation indices are implemented in DAMBE (Xia 2013, 2017d).

## 2 Genomic Strand Asymmetry and GC Skew

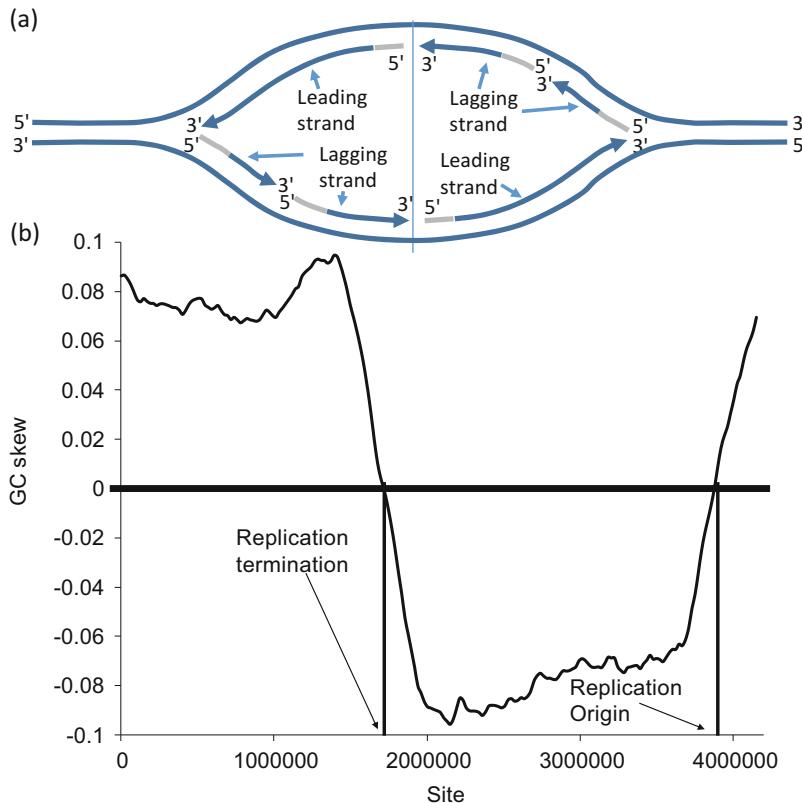
Most mutations occur during DNA replication, and different DNA replication mechanisms often leave distinct footprints in genomic strand asymmetric patterns because DNA polymerase for the leading and lagging strands differs in replication fidelity (Marin and Xia 2008; Xia 2012a). Strand asymmetry is typically measured by the GC skew (Lobry 1996; Marin and Xia 2008) defined as

$$S_G = \frac{P_G - P_C}{P_G + P_C} \quad (11.1)$$

A more general motif skew (Lopez et al. 1999) is defined as

$$S_m = \frac{N_m - N_{m_{rc}}}{N_m + N_{m_{rc}}} \quad (11.2)$$

where  $m$  is either a nucleotide (e.g., G or A) or a motif (e.g., ACG),  $m_{rc}$  is the reverse complement of  $m$  ( $m_{rc} = C$  if  $m = G$ , or  $m_{rc} = CGT$  if  $m = ACG$ ), and  $N_x$  is the number of  $x$  (where  $x$  is either  $m$  or  $m_{rc}$ ). GC skew and AT skew are special cases of  $S_m$  when  $m$  is equal to either G or A, respectively, i.e., GC skew is  $S_G$  and AT skew is  $S_A$ . Strand asymmetry represents a primary feature of bacterial genomes with single-origin replication (Fig. 11.1a). The leading strand is GT-rich and the lagging strand AC-rich, leading to high  $S_G$  values in sliding windows within the leading strand and low  $S_G$  values in sliding windows within the lagging strand (Fig. 11.1b). Because DNA is written in the 5' to 3' direction, the point where  $S_G$  changes from low to high values is the origin of replication. Similarly, the point where  $S_G$  changes from high to low values corresponds to the site of replication termination (Fig. 11.1b). The window size of the  $S_G$  plot, or GC skew plot, in Fig. 11.1b was chosen automatically so as to maximize the area enclosed between the  $S_G$  curve and the horizontal line.



**Fig. 11.1** Identifying the origin of replication (a) by GC skew plot illustrated with the *Bacillus subtilis* genome along a sliding window, with inferred replication origins (b). The *B. subtilis* pattern is shared among all eubacterial species known to have single-origin bi-directional replication. The gray segments in (a) represents RNA primer. Generated from DAMBE (Xia 2013, 2017d)

Bacterial species from *Bacillus subtilis* to *Escherichia coli* share the strand asymmetric pattern in Fig. 11.1a. Interestingly, mitochondrial DNA (mtDNA) in primitive forms of plants such as the liverwort *Marchantia polymorpha*, or of metazoans such as the sponge *Oscarella lobularis*, have strand asymmetric patterns that are similar to the pattern of single-origin replication in Fig. 11.1b. Thus, the GC skew plot in Fig. 11.1b represents an ancestral phenotype. This similarity in strand asymmetric patterns suggests similarity in replication mechanisms and may explain the extremely slow rate of evolution in primitive animal and plant mtDNA relative to mtDNA in higher metazoans. In other words, mitochondrial genomes in plants and primitive invertebrates may maintain the high-fidelity replication as in their bacterial ancestor and differ from more recent eukaryotic lineages with low-fidelity replication in their mtDNA. According to the strand-displacement model of mammalian mtDNA replication (Bogenhagen and Clayton 2003; Brown et al. 2005; Clayton 1982, 2000; Shadel and Clayton 1997), one strand is left in single-stranded state for

almost up to 2 h during mtDNA replication. Spontaneous deamination of both A and C (Lindahl 1993; Sancar and Sancar 1988) occurs frequently in vertebrate mtDNA (Tanaka and Ozawa 1994). Deamination of A leads to hypoxanthine that pairs with C, generating an A→G mutation. Deamination of C leads to U, generating C→U mutations. Among these two types of spontaneous deamination, C→U mutation occurs more frequently than A→G mutation (Lindahl 1993). In particular, the C→U mutation mediated by the spontaneous deamination occurs in single-stranded DNA more than 100 times as frequent as double-stranded DNA (Frederico et al. 1990). Note that these C→U sites will immediately be used as template to replicate the daughter L-strand, leading to a G→A mutation in the L-strand.

### 3 Content Sensors Related to DNA Methylation and Spontaneous Deamination

CpG deficiency has been documented in a large number of genomes covering a wide taxonomic distribution (Cardon et al. 1994; Josse et al. 1961; Karlin and Burge 1995; Karlin and Mrazek 1996; Nussinov 1984). DNA methylation is one of the many hypotheses proposed to explain differential CpG deficiency in different genomes (Bestor and Coxon 1993; Rideout et al. 1990; Sved and Bird 1990). This methylation hypothesis of CpG deficiency features a plausible mechanism as follows. Methyltransferases in many species, especially those in vertebrates, appear to methylate specifically the cytosine in CpG dinucleotides, and the methylated cytosine is prone to mutate to thymine by spontaneous deamination (Frederico et al. 1990; Lindahl 1993). This implies that CpG would gradually decay into TpG and CpA, leading to CpG deficiency, TpG and CpA surplus, and reduced genomic GC%, which has been substantiated numerous times by genomic analysis. Different genomes may differ in CpG deficiency because they differ in methylation activities, with genomes having high methylation activities exhibiting stronger CpG deficiency than genomes with little or no methylation activity. The methylation hypothesis of CpG deficiency was challenged in recent years in several genomic analyses (Cardon et al. 1994; Goto et al. 2000), in particular of *Mycoplasma genitalium* and *M. pneumoniae* genomes, but the controversy has been resolved by a phylogeny-based comparative genomic study (Xia 2003). A detailed recount of the controversy and its resolution will be presented later in the chapter.

#### 3.1 Indices of DNA Methylation

Here we will develop indices that help measure the methylation effect in coding and noncoding sequences. Designate the nucleotide frequencies of a sequence as  $p_A$ ,  $p_C$ ,  $p_G$ , and  $p_T$  and the sequence length as  $L$ . Consider both coding and noncoding

sequences as a linear sequence of consecutive nonoverlapping triplets and the nucleotide frequencies at the three sites of a triplet as  $p_{i1}$ ,  $p_{i2}$ , and  $p_{i3}$ , where  $i = 1, 2, 3$ , and 4 corresponding to nucleotide A, C, G, and T, respectively.

For noncoding sequences, there is no codon structure. So we expect  $p_{i1} \approx p_{i2} \approx p_{i3} \approx p_i$ , where  $p_i$  is the average of  $p_{i1}$ ,  $p_{i2}$ , and  $p_{i3}$ . For coding sequences, various mutation and selection processes will create heterogeneity in nucleotide frequencies among the three sites (Xia 1998b). Take NCG codon, for example, where N stands for any of the four nucleotides. DNA methylation and spontaneous deamination tend to change these NCG codons to NTG and NCA codons (the latter resulting from CpG → TpG mutations in the complementary strand), with the former change being nonsynonymous and the latter synonymous. Because nonsynonymous substitution is generally deleterious and consequently selected against by natural selection, they are much rarer than the synonymous substitutions in a large number of protein-coding genes in many organisms studied (Xia 1998b; Xia et al. 1996; Xia and Li 1998), we should expect NCG → NCA mutations more often than NCG → NTG mutations. This tends to increase the frequency of A at the third codon position. Thus, the presence of many ancestral NCG codons in a protein-coding gene in a methylated genome can lead to local strand bias, with the sense strand being AC-rich (i.e., having many NCA codons) and the template strand GT-rich (i.e., having many TpG dinucleotides).

Similarly, dicodons such as “NNC GNN” tend to mutate synonymously to “NNT GNN” with DNA methylation and spontaneous deamination, increasing the frequency of T at the third codon position. Thus, in contrast to noncoding sequences where we expect  $p_{i1} \approx p_{i2} \approx p_{i3} \approx p_i$ , we should expect  $p_{i1} \neq p_{i2} \neq p_{i3} \neq p_i$  in coding sequences. This suggests that the deviation of  $p_{ij}$  (where  $j = 1, 2$ , and 3 corresponding to the three triplet sites) from  $p_i$  can contribute to the discrimination between coding and noncoding sequences. A measure of this deviation that is independent of  $L$  is as follows:

$$\varphi_{\text{Nuc}} = \frac{\sum_{i=1}^4 \sqrt{\frac{\sum_{j=1}^3 (f_{ij} - f_i)^2}{N_i}}}{4} = \frac{\sum_{i=1}^4 \left( \frac{f_{i1}^2 + f_{i2}^2 + f_{i3}^2}{3f_i^2} - 1 \right)}{4} \quad (11.3)$$

where  $f_{ij}$  stands for the number of nucleotide  $i$  at triplet position  $j$ ,  $f_i$  is the mean number of nucleotide  $i$  averaged over the three codon (triplet) positions, and  $N_i$  is the sum of nucleotide  $i$  in the sequence. For noncoding sequences with the expectation of  $p_{i1} \approx p_{i2} \approx p_{i3} \approx p_i$ ,  $\varphi_{\text{Nuc}}$  will be close to 0. We expect  $\varphi_{\text{Nuc}}$  to be greater for coding sequences than for noncoding sequences.

Following a similar line of reasoning, we expect the dinucleotide frequencies at triplet positions (1,2), (2,3), and (3,1) to be similar to each other in noncoding sequences but different in coding sequences. Designate the number of dinucleotides as  $f_{ij,k}$ , where  $ij = \text{AA}, \text{AC}, \dots, \text{TT}$ , respectively, and  $k = 1, 2, 3$  corresponding to the triplet positions (1,2), (2,3), and (3,1), respectively. The deviation of  $f_{ij,k}$  from  $f_{ij}$ , which is the number of dinucleotide  $i$  averaged over the three triplet positions,

should also contribute to the discrimination between coding and noncoding sequences. A measure of this deviation that is independent of  $L$  is

$$\varphi_{\text{DiNuc}} = \frac{\sum_{i=1}^4 \sum_{j=1}^4 \left( \frac{f_{ij1}^2 + f_{ij2}^2 + f_{ij3}^2}{3f_{ij}^2} - 1 \right)}{16} \quad (11.4)$$

While we expect intron sequences to contain fewer CpG and more TpG and CpA than coding sequences, some other processes can also generate a similar pattern. For example, three codons (AUG for methionine and CAC and CAU for histidine) are generally avoided in coding sequences in vertebrate genomes, presumably because both methionine and histidine are essential amino acids that cannot be synthesized de novo in vertebrates. Intron sequences will not exhibit an avoidance of such AUG, CAC, or CAU triplets. This will result in relatively more TpG and CpA in intron sequences than in coding sequences, but it has little to do with DNA methylation.

### 3.2 Are These Indices Useful in Discriminating Between Coding and Noncoding Sequences?

Here we demonstrate the utility of these indices in discriminating between introns and coding sequences by using the annotated DNA sequence of human chromosome 22 (ref\_chr22.gbk) in GenBank. Human chromosome 22 is perhaps the best annotated human chromosome sequence being the first to be sequenced and having undergone many revisions. The CDSs, exons, and introns were extracted according to the sequence annotation in the FEATURES table, and their triplet/codon frequencies were computed, by using DAMBE (Xia 2013; Xia 2017d). The indices shown in Eqs. (11.3) and (11.4) were also computed by DAMBE for introns and CDSs. Whether the indices are useful can be assessed by how well they can discriminate between coding sequences and introns.

We will take a first look at the coding and noncoding sequences that are at least 2000 bases long. As I have mentioned before, the indices are likely to fluctuate widely with short sequences. Focusing on long sequences will allow us to extract the difference in these indices between the two groups more efficiently. The mean and standard deviation of the two indices (Table 11.1) reveal highly significant difference in these indices between the coding and noncoding (intron) sequences, in the direction as we have expected. For example, the mean  $\varphi_{\text{Nuc}}$  and  $\varphi_{\text{DiNuc}}$  are both much smaller in introns than in coding sequences (Table 11.1), with  $p = 0.0000$ .

I wish to illustrate the discriminating power of these indices by a particular “intron” that has its index values similar to coding sequences. If we apply the linear discriminant function analysis to the two groups of data in Table 11.1, then “intron” would be classified as a coding sequence. The “intron” belongs to a gene annotated as “LOC284861” in the ref.chr22.gbk file, starts with GT and ends with AG, and is “derived by automated computational analysis” according to the FEATURES table

**Table 11.1** Mean and standard deviation of the two indices defined in Eqs. (11.3) and (11.4) for coding and intron sequences in the human chromosome 22. For intron sequences, the indices differ little for the six different triplet frames (i.e., three on each strand), and the numerical results are presented only for the triplet frame starting with the first intron site

Seq. name	Seq. type	$\varphi_{\text{Nuc}}$	$\varphi_{\text{DiNuc}}$
CELSR1	Coding	0.3508	0.5257
MYO18B	Coding	0.2329	0.3169
EP300	Coding	0.1865	0.2780
PKDREJ	Coding	0.1525	0.2146
CACNAII	Coding	0.3670	0.5595
...	...	...	...
Mean		0.2507	0.3822
Std		0.0775	0.1090
LOC40205	Intron	0.0046	0.0110
SYN3	Intron	0.0033	0.0081
LARGE	Intron	0.0107	0.0178
OSBP2	Intron	0.0046	0.0124
SEZ6L	Intron	0.0034	0.0126
...	...	...	...
Mean		0.0318	0.0768
Std		0.0166	0.0302

in the GenBank file. However, it is annotated as part of the coding sequence in other cloned homologous human cDNA sequences (GenBank accession: AL117481, AL122069, AL133561).

There are three lines of evidence to suggest that this “intron” is not an intron. First, when the intron and its two flanking exons are treated as a single exon, there is no embedded stop codon. Second, it has at least four indels when aligned with the GenBank sequence XM\_375042, and all indels are inframe triplets. Such indel events are typical of coding sequences. Third and perhaps the most important, aligning the “intron” with other homologous human cDNA genes shows that its starting GT and ending AG are not conserved, which is not what we would expect if the starting GT and ending AG represent true donor and acceptor sites of an intron. All these suggest that the “misclassification” of the intron as a coding sequence by the discriminant function may in fact represent correct identification.

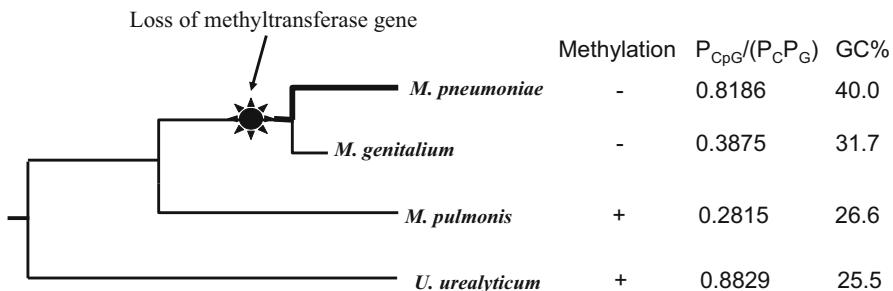
## 4 Do *Mycoplasma* Genomes Challenge the Association Between DNA Methylation and CpG Deficiency?

The seemingly well-established association between CpG deficiency and CpG-specific DNA methylation was recently challenged in a few genomic analyses (Cardon et al. 1994; Goto et al. 2000). For example, *Mycoplasma genitalium* does

not have any methyltransferase and exhibits no methylation activity, yet its genome shows strong CpG deficiency. Therefore, the CpG deficiency in *M. genitalium*, according to the critics of the methylation hypothesis of CpG deficiency, must be due to factors other than DNA methylation. A related species, *M. pneumoniae*, is also devoid of any DNA methyltransferase, but its genome exhibits only mild deficiency in CpG. Given the difference in CpG deficiency between the two *Mycoplasma* species, the methylation hypothesis of CpG deficiency would have predicted that the *M. genitalium* genome is more methylated than the *M. pneumoniae* genome, which is not true as neither has CpG-specific methyltransferase genes. Thus, the methylation hypothesis does not seem to have any explanatory power to account for the variation in CpG deficiency, at least in the two *Mycoplasma* species.

These criticisms are derived from phylogeny-free reasoning, and its fallacy is easy to see in a phylogenetic perspective (Xia 2003). First, several lines of evidence suggest that the common ancestor of *M. genitalium* and *M. pneumoniae* has CpG-specific methyltransferases and should have evolved strong CpG deficiency and low genomic GC% as a result of the specific DNA methylation. Methylated m<sup>5</sup>C exists in the DNA of a close relative, *Mycoplasma hyorhinis* (Razin and Razin 1980), suggesting the existence of methyltransferases in *M. hyorhinis*. Methyltransferases are also present in *Mycoplasma pulmonis* which contains at least four CpG-specific methyltransferase genes (Chambaud et al. 2001). Methyltransferases are also found in all surveyed species of a related genus, *Spiroplasma* (Nur et al. 1985). These lines of evidence suggest that methyltransferases are present in the ancestors of *M. genitalium* and *M. pneumoniae*.

Second, the methyltransferase-encoding *M. pulmonis* genome is even more deficient in CpG and lower in genomic GC% than *M. genitalium* or *M. pneumoniae*, consistent with the methylation hypothesis of CpG deficiency (Fig. 11.2). It is now easy to understand that, after the loss of methyltransferase in the ancestor of *M. genitalium* and *M. pneumoniae* (Fig. 11.2), both genomes would begin to accumulate CpG dinucleotides and increase their genomic GC%. However,



**Fig. 11.2** Phylogenetic tree of *Mycoplasma pneumoniae*, *M. genitalium*, and their relatives, together with the presence (+) or absence (−) of CpG-specific methylation,  $P_{CpG}/(P_C P_G)$  as a measure of CpG deficiency, and genomic GC%. *M. pneumoniae* evolves faster and has a longer branch than *M. genitalium*. Cytosine methylation in *U. urealyticum* is not CpG-specific, so it does not reduce CpG dinucleotide but does reduce GC% in the genome

the evolutionary rate is much faster in *M. pneumoniae* than in *M. genitalium* based on the comparison of a large number of protein-coding genes (Xia 2003). So *M. pneumoniae* regained CpG dinucleotide and genomic GC% much faster than *M. genitalium* whose slow rate of genomic evolution allows it to retain the ancestral low CpG phenotype better than *M. pneumoniae*. In short, the *Mycoplasma* data that originally seem to contradict the methylation hypothesis actually provide strong support for the methylation hypothesis when phylogeny-based genomic comparisons are made.

One might note that *Ureaplasma urealyticum* in Fig. 11.2 is not deficient in CpG because its  $P_{CpG}/(P_C P_G)$  ratio is close to 1, yet its genomic GC% is the lowest. Has its low genomic GC% resulted from CpG-specific DNA methylation? If yes, then why doesn't the genome exhibit CpG deficiency? It turns out that *U. urealyticum* has C-specific, but not CpG-specific, methyltransferase, i.e., the genome of *U. urealyticum* is therefore expected to have low CG% (because of the methylation-mediated C→T mutation) but not a low  $P_{CpG}/(P_C P_G)$  ratio. The methyltransferase gene from *U. urealyticum* is not homologous to that from *M. pulmonis*.

We have seen how phylogeny can help us in evolutionary inference, and most comparative genomic studies represent phylogeny-based inference. I hope that this will motivate you to study topics related to molecular phylogenetics covered in the next few chapters.

## 5 Epigenetics and New Perspectives on Human Diseases

Phenotype results from the interaction between genotype and the environment, but we often fail to appreciate environmental contributions to phenotype. The news media often flash photos of monozygotic twins to impress us with how similar they are, consequently misleading us to overestimate the effect of genotype and underestimate the effect of environment on phenotype. We may consider human adrenoleukodystrophy (ALD) to illustrate this point. ALD results from partial deletion of the 10-exon gene *ABCD1* whose protein participates in degradation of very long chain fatty acids (VLCFA). The deletion results in the accumulation of VLCFA (Krasemann et al. 1996; Morita et al. 2011). Monozygotic twins carrying the same deleterious deletion often differ much in phenotype (Korenke et al. 1996; Petronis 2004, 2006; Petronis et al. 2003).

### 5.1 DNA Methylation and Conventional View of Methylated Gene Silencing

Epigenetic modification includes two interrelated events, DNA methylation and histone modification. The maintenance of DNA methylation pattern in mammals is

accomplished by the mammalian DNA methyltransferase 1 (DNMT1) whose CatD domain recognizes hemimethylated CpG sites (Fatemi et al. 2001) so that DNA methylation pattern can be maintained from parental to daughter cells. In mammals, the methylated CpG recruits proteins with a methyl-CpG-binding domain such as MBD1, MBD2, MBD3, and MeCP2 which then recruit histone deacetylase to remove the acetyl group and restore the positive charge of lysine residues (or histone N-terminal) in histone so that the negatively charged backbone of DNA can wrap tightly around the positively charged histone to silence the gene (Wade and Wolffe 2001). A silenced gene is in many ways equivalent to a loss-of-function mutation. Because some cancers appear to be caused by permanent silencing of genes involved in apoptosis pathway through DNA methylation and histone deacetylation (Insinga et al. 2005a, b), histone deacetylase has been used as a drug target with its inhibitors aiming to reactivate the apoptosis pathway (Bolden et al. 2006). The main problem in this approach is specificity because deacetylase inhibitors often have profound effect on the regulation of many other genes, which may explain why such drugs often do not enter clinical trials (Voelter-Mahlknecht 2016). Methods for precise editing of the epigenome, involving components for DNA-binding and specific sequence recognition and modification, are currently being developed (Kungulovski and Jeltsch 2016).

## 5.2 *Epigenetics and New Perspective on Methylation-Mediated Gene Interaction*

The conventional view that DNA methylation and histone deacetylation mainly serve the purpose of permanent gene silencing has now been replaced by a more general conceptual framework of epigenetic modification and gene regulation. DNA methylation alters DNA/protein binding which in turn alters genome architecture, i.e., two DNA segments far apart along the linear DNA can be brought together through the interaction of their binding proteins. That gene expression depends on gene location on the genome is known since 1930 through studies of translocation (Muller and Altenburg 1930), but empirical evidence accumulated much later to demonstrate that protein/DNA interactions resulted in nucleosome reconfiguration and interaction between enhancer and promoter (Ito et al. 1997; Pazin et al. 1994, 1996, 1998; Sheridan et al. 1995). Mutations at enhancer alter the long-range DNA interaction and change gene expression. You may recall that lactase persistence phenotype (i.e., the phenotype of producing lactase into adulthood) results from mutations about 14,000 nt upstream from the lactase gene (reviewed in Segurel and Bon 2017).

## 5.3 Bioinformatics and Epigenetics

### 5.3.1 Three New Types of Data

The new epigenetic perspective implies three closely knit processes: DNA methylation, protein-DNA binding mediated by DNA methylation, and long-range interaction between DNA segments mediated by protein-DNA binding. We consequently have three closely related new types of data: (1) large-scale methylation pattern characterized by bisulfite sequencing (Grigg and Clark 1994; Grigg 1996), (2) DNA-protein binding data (cistrome) by ChIP-on-chip and ChIP-Seq (Robertson et al. 2007), and (3) genome architecture data from Hi-C (Lieberman-Aiden et al. 2009) or its derivatives, which gives us a list of DNA segment pairs that are physically close to each other. Bioinformatic analysis needs to integrate these three new types of high-throughput data (Xia 2017b).

### 5.3.2 Key Bioinformatic Questions

#### 5.3.2.1 What Constitutes a CpG Methylation Signal?

From a bioinformatics point of view, the key question concerns what is the methylation signal on DNA and whether it is possible to modulate such a signal to alter epigenetic modifications. I have mentioned before that some  $\beta$ -thalassemia patients have the correct version of the  $\beta$ -globin gene but the gene is not expressed because of mutations that occurred far away from it (Kioussis et al. 1983; Taramelli et al. 1986). One may formulate two hypotheses. First, the enhancer that controls the expression of  $\beta$ -globin gene is mutated or deleted in the patient (Forrester et al. 1990; Kioussis et al. 1983). Second, the enhancer that is brought close to the promotor of  $\beta$ -globin gene in normal genome architecture is relocated somewhere else due to abnormal epigenetic modifications and protein/DNA binding. Testing these hypotheses, which has become possible only with the availability of high-throughput data of genome architecture, methylation patterns, and cistromes (the set of all protein-/DNA-binding sites), would shed light on how we can reposition the enhancer and the  $\beta$ -globin promotor so that the gene will be expressed (Deng et al. 2012, 2014b; Hou et al. 2008). Similarly, if the  $\beta$ -globin gene is silenced through DNA methylation, then the knowledge of how to modulate the signal to modify the methylation pattern would bring us closer to reactivating the silenced  $\beta$ -globin gene. Along the same line of reasoning, if the fetal globin genes are silenced by methylation, and if reactivation of these fetal globin genes can alleviate the problem caused by mutations in adult globin genes, then the knowledge of site-specific demethylation would be highly desirable (Kungulovski and Jeltsch 2016).

Given that some CpG are methylated and some are not in mammalian genomes, one straightforward bioinformatic analysis would be to compare the flanking sites of these two groups of CpG dinucleotides to detect if flanking nucleotides contributes to methylation signals. Such an approach has been fruitful in study splicing signals (Ma and Xia 2011; Vlasschaert et al. 2016) or translation stop signals (Wei and Xia 2017). However, comparisons of flanking regions between methylated and unmethylated CpG, although done in a limited scale (Bibikova et al. 2011; Eckhardt et al. 2006; Shoemaker et al. 2010), have not yielded clear-cut results. Equally disappointing is that, while the concept of imprinting center (IC) has been known for many years (Ohta et al. 1999), the physical basis of IC, either at the sequence level or structural level, remains elusive.

### 5.3.2.2 What Environmental Factors Modulate DNA Methylation?

The observation that monozygotic twins carrying the same genetic defect often differ much in manifestation of the associated disease (Korenke et al. 1996; Petronis 2004, 2006; Petronis et al. 2003) suggests environmental contribution to the difference, such as diet (Chen et al. 2016; Sharma et al. 2016). As methylation needs S-adenosyl L-methionine (SAM) as the methyl donor, a deficiency in methionine most likely will, and indeed has been confirmed to, affect DNA methylation (Ingrosso et al. 2003; Ingrosso and Perna 2009). Similarly, one would predict that any major perturbation on methionine, such as the deletion of methylthioadenosine phosphorlyase (MTAP) crucial in the methionine salvage pathway, would also affect DNA methylation, gene regulation, and cancer. Indeed, MTAP deletion is common in cancer cells (Bigaud and Corrales 2016). Thus, all genes that affect methionine metabolism could be drug targets, and bioinformatics, with databases such as KEGG (Kanehisa 2013; Kanehisa et al. 2016; Tanabe and Kanehisa 2012), can identify such genes effectively.

### 5.3.2.3 What Is the Correct Methylation Pattern and What Molecules Encode It?

If a wrong DNA methylation pattern could be identified, then an ideal drug (or an epigenome-repairing nanomachine) should be able to specifically identify the wrong pattern and correct it (Kungulovski and Jeltsch 2016). To develop such a drug or nanomachine, we first have to know the correct methylation pattern or ideally discover a set of molecules that encode such a correct pattern. Experimental results have accumulated in support of RNA's role in epigenetic modification (Jin et al. 2004a). Given that DNA in the zygote undergoes demethylation to regain pluripotency (Clark 2015), the epigenomic code is perhaps not on DNA. As proteins do not seem to be good for writing code in and because most core histones are replaced by protamine in male germ cells (Bao and Bedford 2016), the epigenetic codes, especially the ones that specify de novo DNA methylation, are unlikely to be

found in proteins. However, such codes may exist in a set of highly conserved and structurally stable RNA molecules that might be present as early as the oocyte and sperm stage. Long noncoding RNAs (lncRNAs) can participate in epigenetic modification and regulate chromatin state. Characterization of lncRNAs bound to DNA and protein by the ChiRP-seq method (Chu et al. 2011, 2012) revealed numerous sequence-specific binding sites on DNA, and the binding of lncRNA such as HOTAIR (Chu et al. 2011; Rinn et al. 2007) and Kcnq1ot1 (Pandey et al. 2008) to such sites facilitates the recruitment of polycomb repressive complex 2 (PRC2) for mediating histone H3 lysine-27 trimethylation. Short RNAs can also modulate epigenetic changes. Mature sperm contain a number of small RNA species (Chen et al. 2016; Rodgers et al. 2015; Sendler et al. 2013; Sharma et al. 2016), and these small RNAs do affect offspring phenotype (Gapp et al. 2014; Rodgers et al. 2015). Furthermore, these small RNAs on offspring appear to contribute to epigenetic modification (Chen et al. 2016; Gapp et al. 2014; Rodgers et al. 2015; Sharma et al. 2016). The ENCODE pilot project shows that “the genome is pervasively transcribed, such that the majority of its bases can be found in primary transcripts” (Birney et al. 2007). Those noncoding transcripts may be a treasure trove for bioinformaticians to discover epigenome-modifying RNAs as drug targets.

### 5.3.2.4 How Do Epigenomes Evolve?

Epigenetic modification has an early origin. Many bacterial species modify their own DNA by methylation to protect against endogenous type II restriction endonucleases. Some bacteriophages have their own methyltransferase that can modify their own genome against host restriction digestion (Murphy et al. 2013), and human viral pathogens such as HIV-1 can induce profound alteration in host epigenetic pattern (Abdel-Hameed et al. 2016). It is now known that some of the host defense mechanisms against pathogens are implemented through epigenetic modifications (Arbibe and Sansonetti 2007; Bierne et al. 2012) and many pathogens can modify host epigenetic patterns in favor of their survival and reproduction in the host (Bierne et al. 2012). What is the eventual fate of such pathogen-mediated epigenetically modified host cells remains unclear. Do they defeat the pathogen invasion, restore the normal epigenetic pattern, and reassume normal function again, or do they initiate certain apoptosis pathway and perish? What epigeneticists need is a model organism or a cell line in which the epigenetic pattern can be perturbed by extrinsic factors and then restored back to normal.

## Postscript

“No aphorism is more frequently repeated in connection with field trials, than that we must ask Nature few questions, or ideally, one question at a time. The writer is convinced that this view is wholly mistaken. Nature, he suggests, will respond to a

logical and carefully thought-out questionnaire; indeed, if we ask her a single question, she will often refuse to answer until some other topic has been discussed” (Fisher 1926).

Researchers have attacked the methylation hypothesis of CpG deficiency without a phylogeny perspective. Nature gave them a wrong answer and landed them in an undesirable position.

Here comes another story for illustrating the value of thinking broadly. I have previously told a fable of Afandi in which the ancient Islamic sage rode his little donkey around the country teaching people to be wise. The king, craving to be recognized as the wisest, had always been wracking his brain for plots to ambush Afandi intellectually. One day he received the happy news that Afandi had recently suffered from double vision and jumped into action immediately by inviting Afandi to the palace to meet him and his courtiers. Upon Afandi’s arrival, the king greeted him, “Congratulations Afandi! I heard that you have just doubled your fortune. Isn’t it true that, through your wise and penetrating eyes, you now see two wives and two houses of your own? Even the little donkey behind you has now doubled itself in your eyes, isn’t it? What a smart way of doubling one’s fortune!” Afandi, realizing that the king was making fun of his double vision, was not pleased, but he replied calmly, among the jeering laughters of the courtiers, “Truly, Your Majesty. At this moment, my eyes are misleading me to think that Your Majesty has four legs.”

Just as researchers attacking the methylation hypothesis landed themselves in an undesirable position, so was the king attacking Afandi. Following Fisher’s sagely advice of thinking broadly may keep us safe from making the same mistake.

# Chapter 12

## Nucleotide Substitution Models and Evolutionary Distances



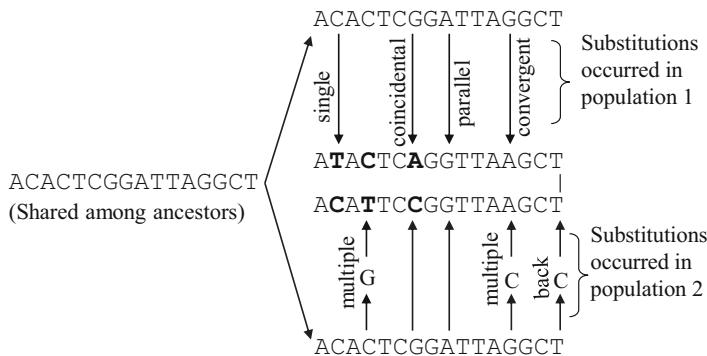
### 1 Introduction

This and the following chapters belong to the domain of molecular phylogenetics, including phylogeny-based comparative methods. Substitution models are the subject of this and the next chapter, which is followed by phylogenetic reconstruction methods. The objective is to trace evolutionary history as far back as possible so that we, in Aristotle's words, can all have the most advantageous view of things by viewing them from the very beginning.

It is often difficult to stretch our vision back to time immemorial. Substitutions occur over time and can overwrite each other at the same nucleotide or amino acid site. Such a process has been going on for millions or billions of years. When we compare two homologous nucleotide sequences and find differences in  $N$  sites, the actual number of substitutions (designated by  $M$ ) could be much greater than  $N$  because multiple substitutions could have happened at the same site, overwriting each other (Fig. 12.1). How can we infer the unobservable  $M$  from the observable  $N$ ?

Substitution models are used to correct for multiple hits. This chapter covers three related topics. The first is how to derive transition probabilities and evolutionary distances from an instantaneous rate matrix. Most of the chapter is taken up by my detailed illustration of three different approaches to do the derivation. The second is how far we can trace evolutionary history back in time if sequences do evolve according to a specified substitution model. The third is how to choose the best substitution models, i.e., likelihood ratio test for nested models and information theoretic indices (AIC, BIC, etc.) for nested or non-nested models.

There are three types of molecular sequences, i.e., nucleotide, AA, and codon sequences. Consequently, there are three corresponding types of substitution models. Many substitution models have been proposed for nucleotide, amino acid, and codon sequences. All substitution models used in molecular phylogenetics are Markov chain models characterized by (1) either a transition probability matrix ( $P$ ) with discrete time or a rate matrix ( $Q$ ) in continuous time where  $P$  can be derived



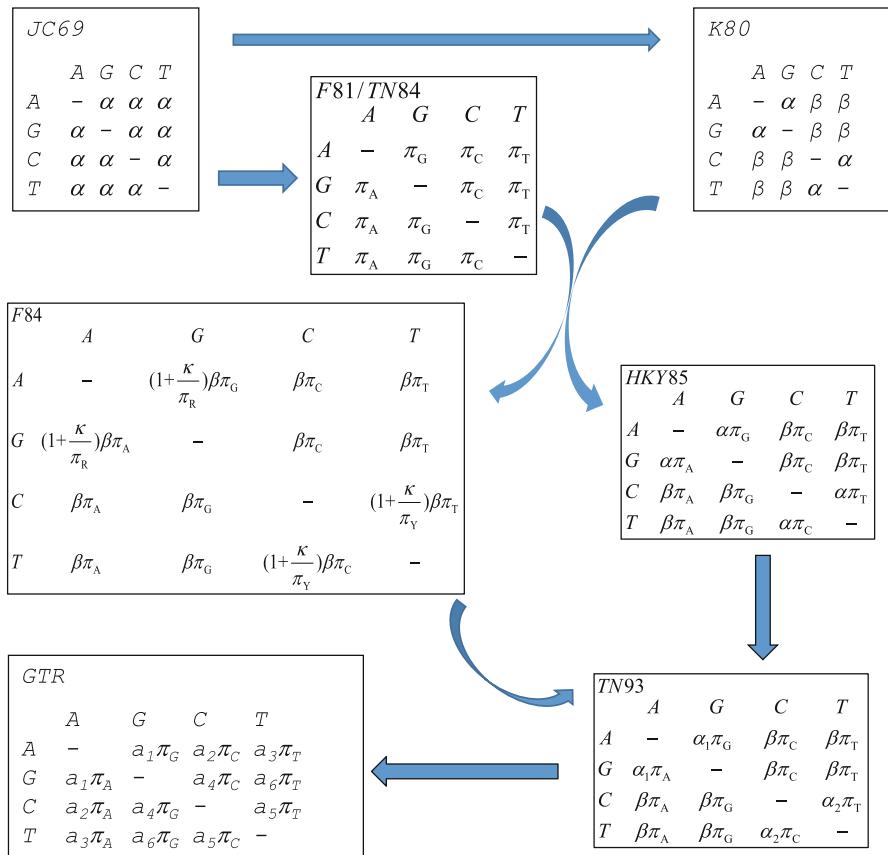
**Fig. 12.1** Illustration of nucleotide substitutions and the difficulty in correcting for multiple hits. The ancestor has diverged into northern and southern populations that have independently accumulated substitutions indicated by arrows. A total of 12 substitutions have happened but only three sites differ between the two extant sequences. After Li (1997)

from  $Q$  and vice versa and (2) equilibrium frequencies. Note that the phrase “transition probability” in transition probability matrix means the probability of remaining in one state or changing from one state to another (e.g., nucleotide A to one of the other three nucleotides) in one unit time. The general form of a rate matrix for nucleotide sequences is in the order of A, G, C, and T:

$$Q = \begin{bmatrix} A & - & a & b & c \\ G & g & - & d & e \\ C & h & i & - & f \\ T & j & k & l & - \end{bmatrix} \quad (12.1)$$

Frequently used nucleotide-based substitutions and their relationships are illustrated in Fig. 12.2, with the JC69 model (Jukes and Cantor 1969) being the simplest. The K80 model (Kimura 1980) generalizes the JC69 model by allowing transitions and transversions to have different rates ( $\alpha$  for transitions and  $\beta$  for transversions). F81/TN84 (Felsenstein 1981; Tajima and Nei 1984) generalizes JC69 by adding the frequency parameters. F84 (used in DNAML since 1984, Hasegawa and Kishino 1989; Kishino and Hasegawa 1989) and HKY85 (Hasegawa et al. 1985) may be viewed as extension from F81/TN84 with additional differential rates for transitions and transversions or from K80 by adding frequency parameters. TN93 (Tamura and Nei 1993) extended F84/HKY85 by allowing C↔T and A↔G to have different rates, and GTR (Lanave et al. 1984; Tavaré 1986) extended TN93 by allowing the four types of transversions to have different rates. These substitution models are used in DAMBE (Xia 2013, 2017d) for performing various molecular phylogenetic analyses. DAMBE also implements functions for selecting the best substitution models for specific sequences.

Many more substitution models and genetic distances have been proposed (Tamura and Kumar 2002), with the number of all possible time reversible models



**Fig. 12.2** Relationship among commonly used substitution models for nucleotide sequences specified by their respective Q matrices. The diagonals of each matrix are constrained by the summation of each row being equal to 0

of nucleotide substitution being 203 (Huelsenbeck et al. 2004). In addition, there are more complicated models underlying the LogDet and the paralinear distances (Lake 1994; Lockhart et al. 1994) that can presumably accommodate nonstationary substitution process. Such models have not been implemented in a maximum likelihood framework until very recently (Jayaswal et al. 2005). Different substitution models often lead to different trees produced and constitute a major source of controversy in molecular phylogenetics (Rosenberg and Kumar 2003; Xia 2000; Xia et al. 2003a).

Transition probability matrix, often referred to as the  $P$  matrix, specifies the probability of a nucleotide or amino acid changing into another one after time  $t$ . It is needed to calculate likelihood and to derive evolutionary distances. Thus, whether a substitution model can be implemented in a distance-based or maximum likelihood method essentially depends on whether the model's transition probabilities can be calculated. There are three ways to obtain transition probabilities from

the  $Q$  matrix. These three ways will be explained and numerically illustrated in great detail in the next section. The derivation of variance for a single parameter or variance-covariance matrix for multiple parameters by the delta method is covered in Appendix 1.

## 2 Three Methods to Obtain Transition Probabilities

Transition probabilities can be obtained by three approaches which will be covered in this section: (1) probability reasoning, (2) solving differential equations involving rates, and (3) taking the matrix exponential of the rate matrix. The last two require some mathematical background, i.e., solving differential equations in the second and matrix algebra in the third. The first, in contrast, demands hardly any mathematical skill, except for careful bookkeeping and solving simultaneous equations, and can also yield nice and clean mathematical expressions for transition probabilities and evolutionary distances for a variety of substitution models. Consequently, this approach is particularly relevant to biological students to gain a conceptual understanding of the substitution models. For example, new researchers often ask why we can derive evolutionary distances between two aligned sequences for the TN93 model but cannot for the simpler HKY85 model which is a special case of the TN93 model, yet another model, F84, which is also a special case of the TN93 model, can have its evolutionary distance readily derived. One can easily offer answers to such questions by taking the first approach. However, the first approach is not of general purpose and cannot handle very complicated substitution models. In contrast, the last one can be used with any substitution models specified by a rate matrix from which the matrix exponential can be obtained. The GTR model and its evolutionary distance are covered only with this last approach. In short, all these approaches need to be learned by anyone wishing to become a molecular phylogeneticist. This section aims to make this learning easy, at the cost of some repetition and redundancy. The cost of failing to understand is typically much greater than the cost of some repetition and redundancy.

### 2.1 *Probability Reasoning to Obtain Transition Probabilities and Evolutionary Distances*

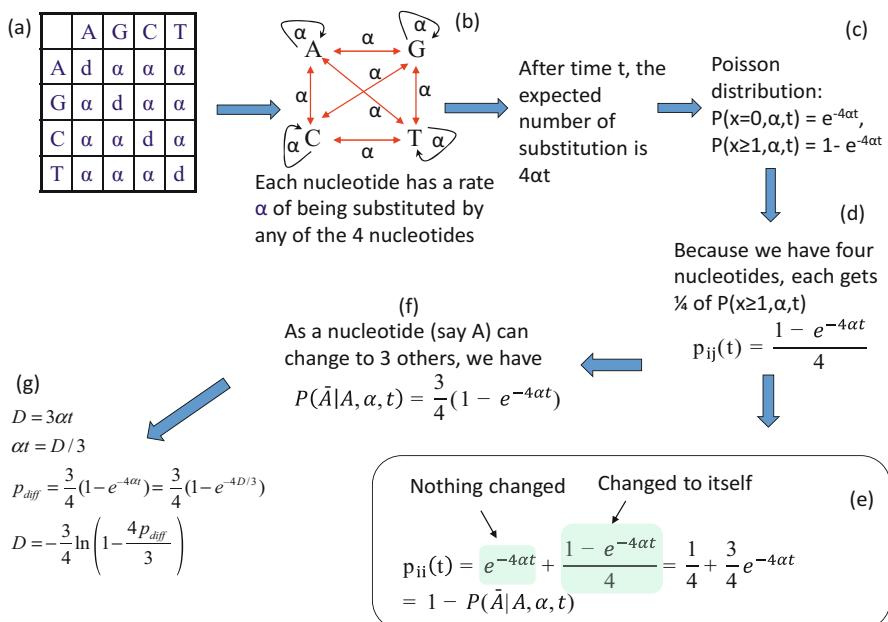
Felsenstein (2004) presented nice examples of probability reasoning to derive transition probabilities and evolutionary distances from rate matrices. This section presents the approach in a more systematic and more accessible way.

### 2.1.1 JC69 Model

Consider nucleotide A in the JC69 model (Fig. 12.3a). Imagine that the nucleotide has a rate  $\alpha$  of changing into any of the four nucleotides, i.e., including changing to itself (Fig. 12.3b). This is effectively the same specification as the JC69 model. After time  $t$ , the expected number of substitutions is  $4\alpha t$ , and the probability of no substitution is  $p(x = 0, \alpha, t) = e^{-4\alpha t}$  according to the Poisson distribution, and the probability of having at least one change is then  $p(x \geq 1, \alpha, t) = 1 - e^{-4\alpha t}$  (Fig. 12.3c). Because nucleotide A can change into any one of the four nucleotides (including nucleotide A itself), each nucleotide gets  $1/4$  of  $p(x \geq 1, \alpha, t)$ . We therefore have in Fig. 12.3d

$$p_{ij}(t) = \frac{p(x \geq 1, \alpha, t)}{4} = \frac{1}{4} - \frac{1}{4}e^{-4\alpha t} \quad (12.2)$$

The transition probability  $p_{ii}(t)$  is the summation of two probabilities: the probability of no change (which is  $e^{-4\alpha t}$ ) and the probability of changing to itself which is the same as specified in Eq. (12.2), as shown in Fig. 12.3e, i.e.,



**Fig. 12.3** Derivation of transition probabilities and the evolutionary distance ( $D$ ) based on the JC69 model. The  $d$  value in the diagonal of the rate matrix (a) is constrained by the row sum equal to 0, i.e.,  $d = -3\alpha$ .  $P(j|i, t)$  means the probability of changing from the original nucleotide  $i$  to nucleotide  $j$  after time  $t$  and is synonymous to  $p_{ij}(t)$  or simply  $p_{ij}$  in this chapter. See text for explanations on (b)–(g)

$$p_{ii}(t) = e^{-4\alpha t} + \frac{p(x \geq 1, \alpha, t)}{4} = \frac{1}{4} + \frac{3}{4}e^{-4\alpha t} \quad (12.3)$$

Thus, the transition probability matrix for the JC69 model is (in the order of A, G, C, and T)

$$P_{\text{JC69}} = \begin{bmatrix} \frac{1}{4} + \frac{3}{4}e^{-4\alpha t} & \frac{1}{4} - \frac{1}{4}e^{-4\alpha t} & \frac{1}{4} - \frac{1}{4}e^{-4\alpha t} & \frac{1}{4} - \frac{1}{4}e^{-4\alpha t} \\ \frac{1}{4} - \frac{1}{4}e^{-4\alpha t} & \frac{1}{4} + \frac{3}{4}e^{-4\alpha t} & \frac{1}{4} - \frac{1}{4}e^{-4\alpha t} & \frac{1}{4} - \frac{1}{4}e^{-4\alpha t} \\ \frac{1}{4} - \frac{1}{4}e^{-4\alpha t} & \frac{1}{4} - \frac{1}{4}e^{-4\alpha t} & \frac{1}{4} + \frac{3}{4}e^{-4\alpha t} & \frac{1}{4} - \frac{1}{4}e^{-4\alpha t} \\ \frac{1}{4} - \frac{1}{4}e^{-4\alpha t} & \frac{1}{4} - \frac{1}{4}e^{-4\alpha t} & \frac{1}{4} - \frac{1}{4}e^{-4\alpha t} & \frac{1}{4} + \frac{3}{4}e^{-4\alpha t} \end{bmatrix} \quad (12.4)$$

Note that the four diagonal elements are the same as specified in Eq. (12.3), and all the off-diagonal elements are the same as specified in Eq. (12.2). Each row in  $P$  adds up to 1 as a nucleotide can either stay the same or change into some other nucleotides.

There are some quick ways to check the derived transition probabilities. First, we note that when  $t$  approaches infinity, then all entries in matrix  $P$  approach  $\frac{1}{4}$  if  $\alpha > 0$ . This is what we have expected. Second, when  $t = 0$ , then all diagonal elements in matrix  $P$  are 1 and all off-diagonal elements are zero. This is again what we expected. Third, if  $\alpha$  is zero, then no change is possible, and we again expect all diagonal elements in matrix  $P$  to be 1 and all off-diagonal elements to be zero, which is also true.

We also notice that, because a nucleotide can change into three other nucleotides, the expected proportion of sites that are different between two aligned homologous sequences ( $p_{\text{diff}}$ ) is  $3 * p_{ij}(t)$ , i.e.,

$$p_{\text{diff}} = 3p_{ij}(t) = \frac{3}{4} - \frac{3}{4}e^{-4\alpha t} \quad (12.5)$$

Note that  $p_{\text{diff}}$  approaches  $\frac{3}{4}$  when  $t$  is infinitely large, which means that multiple substitutions can no longer be properly estimated. Equation (12.5) offers another way of deriving  $p_{ii}(t)$  in Eq. (12.3), i.e.,  $p_{ii}(t)$  is simply  $1 - p_{\text{diff}}$ .

Equation (12.5) allows us to derive the JC69 distance ( $D_{\text{JC69}}$ ) because a distance is defined as  $\mu t$  where  $\mu$  is the substitution rate which is equal to  $3\alpha$  in the JC69 model. This is the same as the distance that you have driven is the product of the speed (rate) and time. We can derive  $D_{\text{JC69}}$  (Fig. 12.3g) by substituting  $\alpha t = D_{\text{JC69}}/3$  into Eq. (12.5), i.e.,

$$D_{\text{JC69}} = -\frac{3}{4} \ln \left( 1 - \frac{4p_{\text{diff}}}{3} \right) \quad (12.6)$$

where  $p_{\text{diff}}$  (the expected number of sites that are different between the two homologous sequences) can be approximated by the observed proportion of sites ( $p_{\text{diff,obs}}$ )

differing between the two aligned sequences. Note that  $p_{\text{diff,obs}}$  may differ from  $p_{\text{diff}}$  even when the underlying substitution model indeed follows JC69 because of (1) stochastic factors due to limited aligned length of the two sequences and (2) distortion caused by suboptimal sequence alignment. Thus, although  $p_{\text{diff}}$  in Eq. (12.5) cannot be greater than 0.75,  $p_{\text{diff,obs}}$  could, even when sequences evolve strictly according to the JC69 model.  $D_{\text{JC69}}$  is not defined when  $p_{\text{diff}} \geq 0.75$  as there is no logarithm for 0 or negative values.

We can show that  $D_{\text{JC69}}$  in Eq. (12.6) is the maximum likelihood distance. For two aligned sequences of length  $N$ , designate the number of sites that differ between the two sequences as  $N_D$  and the number of sites identical between the two sites as  $(N - N_D)$ . Now the likelihood function is

$$\begin{aligned} L &= \left(\frac{1}{4}\right)^N p_{ii}^{(N-N_D)} (1-p_{ii})^{N_D} \\ \ln L &= N \ln \left(\frac{1}{4}\right) + (N - N_D) \ln(p_{ii}) + N_D \ln(1 - p_{ii}) \\ &= N \ln \left(\frac{1}{4}\right) + (N - N_D) \ln \left(\frac{1}{4} + \frac{3}{4}e^{-4D_{\text{JC69}}/3}\right) + N_D \ln \left(\frac{3}{4} - \frac{3}{4}e^{-4D_{\text{JC69}}/3}\right) \end{aligned} \quad (12.7)$$

where the constant term  $N^* \ln(1/4)$  can be dropped in maximizing  $\ln L$  to obtain the distance estimate but needs to be kept when performing likelihood ratio test for comparing different substitution models (e.g., JC69 against TN93). We take the derivative of  $\ln L$  with respect to  $D_{\text{JC69}}$ , set the derivative to 0, and solve for  $D_{\text{JC69}}$ . The resulting  $D_{\text{JC69}}$  is exactly the same as that in Eq. (12.6). I used  $D$  instead of  $D_{\text{JC69}}$  in the equations below:

$$\begin{aligned} \frac{d \ln L}{d D} &= -\frac{(N - N_D)e^{-4D/3}}{\frac{1}{4} + \frac{3}{4}e^{-4D/3}} + \frac{N_D e^{-4D/3}}{\frac{3}{4} - \frac{3}{4}e^{-4D/3}} = 0 \\ D &= -\frac{3}{4} \ln \left( \frac{3N - 4N_D}{3N} \right) = -\frac{3}{4} \ln \left( 1 - \frac{4p_{\text{diff}}}{3} \right) \end{aligned} \quad (12.8)$$

The variance of  $D_{\text{JC69}}$  (designated as  $V_{\text{JC69}}$ ) is obtained as the negative reciprocal of the second derivative of  $\ln L$ :

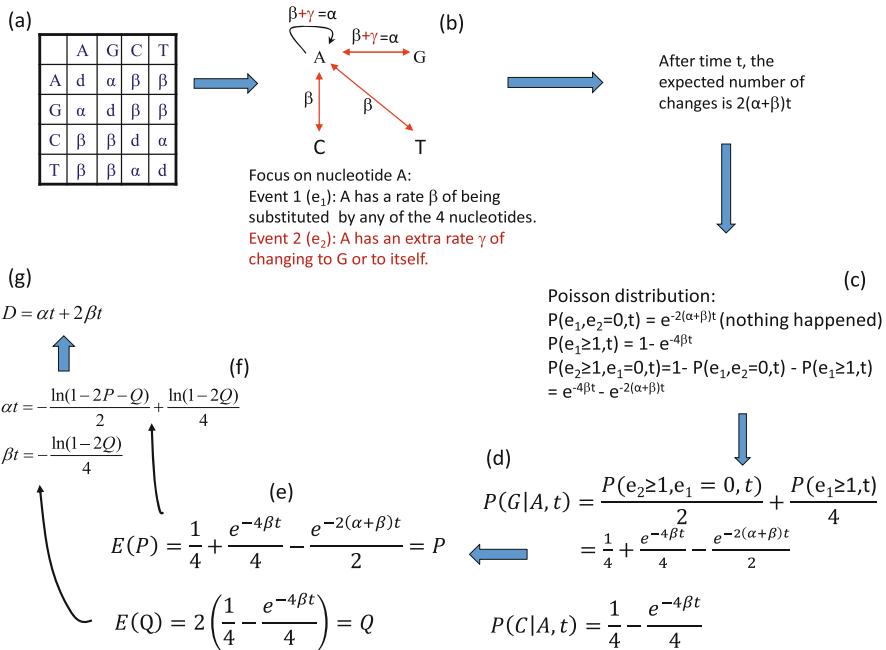
$$V_{\text{JC69}} = -\frac{1}{\frac{d^2 \ln L}{d D_{\text{JC69}}^2}} = \frac{p_{\text{diff}}(1-p_{\text{diff}})}{L \left(1 - \frac{4p_{\text{diff}}}{3}\right)^2} \quad (12.9)$$

Note that  $V_{\text{JC69}}$  decreases with sequence length  $L$  as one would have expected. We illustrate the application of Eqs. (12.6) and (12.9) by using the aligned sequences in Fig. 12.4 where  $N = 24$ ,  $N_D = 6$ , and  $p_{\text{diff}} = 6/24 = 0.25$ . So  $D_{\text{JC69}} = 0.3041$ , and  $\text{var.}(D_{\text{JC69}}) = 0.0176$ .

The equilibrium frequencies of the  $\pi$  vector can be derived by setting  $t = \infty$  in Eqs. (12.2) and (12.3) which leads to  $p_{ii} = p_{ij} = 1/4$ . This implies that the equilibrium

S1: AAG CCT CGG GGC CCT TAT TTT TTG  
 || | || | || | | | |  
 S2: AAT CTC CGG GGC CTC TAT TTT TTT

**Fig. 12.4** Two homologous sequences for illustrating computation of pairwise evolutionary distances



**Fig. 12.5** Derivation of transition probabilities and the evolutionary distance ( $D$ ) based on the K80 model. The rate matrix (a) has the diagonal elements ( $d$ ) constrained by the row sum equal to 0, i.e.,  $d = -\alpha - 2\beta$ .  $P$  and  $Q$  are the observed proportion of transitional and transversional changes between two aligned homologous sequences. Equating them to their respective expected values,  $E(P)$  and  $E(Q)$ , leads to the solution of  $\alpha t$  and  $\beta t$  shown, and the evolutionary distance  $D$ .  $P(j|i, t)$  means the probability of changing from the original nucleotide  $i$  to nucleotide  $j$  after time  $t$  and is synonymous to  $p_{ij}(t)$  or simply  $p_{ij}$  in this chapter. See text for explanations on (b)–(g)

frequencies of the four nucleotides will be equal for the JC69 model. This is not surprising because the frequencies did not even appear in the rate matrix (Fig. 12.3a)

### 2.1.2 K80 Model

The K80 model has a transition substitution rate  $\alpha$  and a transversion rate  $\beta$  (Fig. 12.5a). We will focus on nucleotide A and conceptualize the model with two events (Fig. 12.5b), in contrast to only one event in the JC69 model. The first event

$(e_1)$  occurs when nucleotide A changes into any of the four nucleotides (including to itself). In other words, the original A is replaced by a nucleotide randomly drawn from a nucleotide pool with equal nucleotide frequencies. This event occurs with a rate  $\beta$ . The second event  $(e_2)$  occurs when nucleotide A changes either to G or to itself, i.e., the original A is replaced by a nucleotide randomly drawn from a purine pool with equal number of A and G. This  $e_2$  occurs with a rate  $\gamma$ . Thus the transition rate  $\alpha$  equals  $\beta + \gamma$  according to this conceptualization. Note that, whenever  $e_1$  happens, the original nucleotide is replaced by any one of the four nucleotides with equal probability, no matter how many  $e_2$  events have occurred before or after the occurrence of  $e_1$ . It might help to think of a long sequence with L sites being A at time 0. If these L sites each has experienced at least one  $e_2$  event, then these sites will either be A or G with equal probability (i.e., 0.5), and we expect to have L/2 sites being A and the other L/2 sites being G. In contrast, if each of these L sites has experienced at least one  $e_1$  event, then the site will be replaced by either A, C, G, or T with equal probability, and we expect to observe A, C, G, and T in L/4 sites each. Any  $e_2$  events occurring before or after the  $e_1$  event do not change this expectation. This means that  $e_1$  erases  $e_2$ , but not vice versa. The probability that an  $e_2$  event happened is informative only when no  $e_1$  event has happened.

After time  $t$ , the expected number of substitutions is  $2(\alpha + \beta)t$ , i.e., the nucleotide A has two ways of change with a rate of  $\alpha$  (to A and to G) and another two ways of change with a rate  $\beta$  (to C and to T), so the probability of no change, according to Poisson distribution, is

$$p(e_1 = 0, e_2 = 0, t) = e^{-2(\alpha+\beta)t} \quad (12.10)$$

Note that  $\alpha$  is conceptualized as  $(\beta+\gamma)$  in Fig. 12.5, so  $e^{-2(\alpha+\beta)t}$  in Eq. (12.10) is equivalent to  $e^{-(4\beta+2\gamma)t}$ . The probability that at least one  $e_1$  event has occurred is

$$p(e_1 > 0, t) = 1 - e^{-4\beta t} \quad (12.11)$$

Thus, the probability that at least one  $e_2$  has occurred but  $e_1$  has not occurred is simply

$$\begin{aligned} p(e_2 > 0, e_1 = 0, t) &= 1 - p(e_1 = 0, e_2 = 0, t) - p(e_1 > 0, t) \\ &= 1 - e^{-2(\alpha+\beta)t} - (1 - e^{-4\beta t}) \\ &= e^{-4\beta t} - e^{-2(\alpha+\beta)t} \end{aligned} \quad (12.12)$$

These probabilities are also shown in Fig. 12.5c. The reason for the condition that “ $e_1$  has not occurred” is because  $e_1$  event can erase  $e_2$  event (see the discussion in the first paragraph of this section).

Now the probability of the starting nucleotide A changing to G during time  $t$ , designated as  $p(G|A,t)$ , is the summation of two probabilities. The first is 1/2 of the probability of  $p(e_2 > 0, e_1 = 0, t)$  in Eq. (12.12) because the other 1/2 is for A to itself. The second is 1/4 of  $p(e_1 > 0, t)$  in Eq. (12.11) because A→A, A→G, A→C, and A→T each gets 1/4, so only 1/4 of  $p(e_1 > 0, t)$  is for A→G. The summation of these

two probabilities (Fig. 12.5d) is  $p(\text{G|A}, t)$ . This probability is equal to  $p(\text{A|G}, t)$ ,  $p(\text{C|T}, t)$ , and  $p(\text{T|C}, t)$  in the K80 model. In other words, the summation of these two probabilities is the probability of a transition during time  $t(P_s)$ . Thus,

$$\begin{aligned} P_s &= \frac{p(e_2 > 0, e_1 = 0, t)}{2} + \frac{p(e_1 > 0, t)}{4} \\ &= \frac{e^{-4\beta t} - e^{-2(\alpha+\beta)t}}{2} + \frac{1 - e^{-4\beta t}}{4} \\ &= \frac{1}{4} + \frac{e^{-4\beta t}}{4} - \frac{e^{-2(\alpha+\beta)t}}{2} = P(\text{G|A}, t) = P(\text{A|G}, t) = P(\text{T|C}, t) = P(\text{C|T}, t) \end{aligned} \quad (12.13)$$

Similarly, the probability of the starting A changing to C (or to T) is 1/4 of  $p(e_1 > 0, t)$  in Eq. (12.11) because 1/4 is for A→A, 1/4 is for A→G, and 1/4 is for A→T, so only 1/4 is for A→C (Fig. 12.5d). This probability is the probability for a transversional change during time  $t$ :

$$P_v = \frac{p(e_1 > 0, t)}{4} = \frac{1 - e^{-4\beta t}}{4} \quad (12.14)$$

As a quick check of the derived transition probabilities, we note that  $P_s$  and  $P_v$  are zero when  $t = 0$  (or when  $\alpha = 0$  and  $\beta = 0$ ). This also implies that all diagonal elements in the transition probability matrix are equal to 1 and is what we have expected. When  $t = \infty$  with  $\alpha > 0$  and  $\beta > 0$ , all entries in matrix  $P$  approach 1/4 (the equilibrium frequency of the K80 model). This is also what we expected.

For two aligned homologous sequences,  $P_s$  can be approximated by the proportion of sites differing by a transition ( $P$ ), and  $2P_v$  by the portion of sites differing by a transversion ( $Q$ , Fig. 12.5e). Note that the expected  $Q$  is equal to  $2P_v$  because there are two ways of having a transversional change. Therefore,

$$P = \frac{1}{4} + \frac{e^{-4\beta t}}{4} - \frac{e^{-2(\alpha+\beta)t}}{2} \quad (12.15)$$

$$Q = 2P_v = 2\left(\frac{1 - e^{-4\beta t}}{4}\right) = \frac{1 - e^{-4\beta t}}{2} \quad (12.16)$$

We can now first solve for  $\beta t$  from Eq. (12.16) and then substitute the solution for  $\beta t$  into Eq. (12.15) to solve for  $\alpha t$ . This leads to

$$\begin{aligned} \alpha t &= -\frac{\ln(1 - 2P - Q)}{2} + \frac{\ln(1 - 2Q)}{4} \\ \beta t &= -\frac{\ln(1 - 2Q)}{4} \end{aligned} \quad (12.17)$$

Recall that evolutionary distance is defined as  $\mu t$ , where  $\mu$  is the substitution rate which is equal to  $(\alpha + 2\beta)$  in the K80 model. Thus, the evolutionary distance based on the K80 model ( $D_{\text{K80}}$ ) is  $(\alpha + 2\beta)t$ , which comes to

$$D_{K80} = \alpha t + 2\beta t = -\frac{\ln(1-2P-Q)}{2} - \frac{\ln(1-2Q)}{4} \quad (12.18)$$

where  $P$  and  $Q$  can be approximated by the observed proportion of sites differing by a transition or a transversion from two aligned sequences, designated as  $P_{obs}$  and  $Q_{obs}$ . Similar to what I have mentioned with reference to  $D_{JC69}$ ,  $P$  and  $Q$  may differ from  $P_{obs}$  and  $Q_{obs}$  even if the K80 model is followed during the sequence evolution. This is because (1) limited aligned length of the two sequences may result in stochastic variation in  $P_{obs}$  and  $Q_{obs}$ , and (2) the two observed proportions may be distorted by alignment errors. For example, for two homologous sequences that have diverged for an infinite length of time according to the K80 model, we would have expected  $P$  and  $Q$  to be 0.25 and 0.5, respectively. However, we may actually have  $P_{obs} > 0.25$  or  $Q_{obs} > 0.5$ , which would render  $D_{K80}$  inapplicable. On the other hand, after sequence alignment and deletion of indels (because evolutionary distances are typically calculated without using sites with indels),  $P_{obs}$  and  $Q_{obs}$  may well be much smaller than the expected 0.25 and 0.5 leading to severer underestimation of the true distance. It is also possible to have  $P_{obs}$  and  $Q_{obs}$  values that, when used to replace  $P$  and  $Q$  in Eq. (12.17), result in negative  $\alpha t$  or  $\beta t$  values that make no biological sense. The same applies to  $D_{JC69}$  (in fact to any evolutionary distances based on a substitution model). Methods for handling such situations are discussed later in the section on the GTR model.

$D_{K80}$  in Eq. (12.18) is also a maximum likelihood estimator of the distance based on the K80 model, just like the  $K_{JC69}$  distance in Eq. (12.6). To see this, it is simpler to re-parameterize the K80 model by replacing  $\alpha t$  and  $\beta t$  by  $D_{K80}$  and  $\kappa$  using the following relationship:

$$\begin{aligned} D_{K80} &= \alpha t + 2\beta t \\ \kappa &= \alpha t / \beta t \end{aligned} \quad (12.19)$$

Solving these two equations gives us

$$\begin{aligned} \alpha t &= \frac{D_{K80}\kappa}{\kappa+2} \\ \beta t &= \frac{D_{K80}}{\kappa+2} \end{aligned} \quad (12.20)$$

Substituting  $\alpha t$  and  $\beta t$  into Eqs. (12.15) and (12.16) so that  $P$  and  $Q$  will be functions of  $D_{K80}$  and  $\kappa$  and the likelihood function for deriving  $D_{K80}$  and  $\kappa$  is

$$\begin{aligned} L &= \left(\frac{1}{4}\right)^N P^{N_s} Q^{N_v} (1-P-Q)^{N-N_s-N_v} \\ \ln L &= N \ln \left(\frac{1}{4}\right) + N_s \ln P + N_v \ln Q + (N - N_s - N_v) \ln (1-P-Q) \end{aligned} \quad (12.21)$$

where the constant term  $N^* \ln(1/4)$  can be dropped in maximizing  $\ln L$  to obtain the distance estimate but needs to be kept when performing likelihood ratio test for

comparing different substitution models (e.g., K80 against TN93). Taking partial derivatives with respect to  $D_{\text{K80}}$  and  $\kappa$ , setting them to zero, and solving the simultaneous equations, we have

$$D_{\text{K80}} = -\frac{\ln(1 - 2P_{\text{obs}} - Q_{\text{obs}})}{2} - \frac{\ln(1 - 2Q_{\text{obs}})}{4} \quad (12.22)$$

$$\kappa = \frac{2 \ln(1 - 2P_{\text{obs}} - Q_{\text{obs}})}{\ln(1 - 2Q_{\text{obs}})} - 1 \quad (12.23)$$

where  $P_{\text{obs}} = N_s/N$ , and  $Q_{\text{obs}} = N_v/N$ . Using the two aligned sequences in Fig. 12.4, we have  $N = 24$  and  $P_{\text{obs}} = 4/24$  and  $Q_{\text{obs}} = 2/24$ . These lead to  $D_{\text{K80}} = 0.3151$ , and  $\kappa = 4.9126$ . It may be relevant to add that, while  $D_{\text{JC69}}$  and  $D_{\text{K80}}$  are maximum likelihood estimates, distance formulae for F84 and TN93 models, obtained in the same by equating the observed number of substitutions to expected number of substitutions, are generally not maximum likelihood estimates. This will become clear when we deal with these models.

We have previously derived the variance of  $D_{\text{JC69}}$  as the negative reciprocal of the second derivative of  $\ln L$  with respect to  $D_{\text{JC69}}$ . This can be used only when the log-likelihood function is used to estimate a single parameter. When there are multiple parameters (e.g.,  $D_{\text{K80}}$  and  $\kappa$ ), we cannot use the same approach unless the parameters are not correlated. There are two commonly used methods for deriving variances of parameters. The first is the delta method (Kimura and Ohta 1972; Waddell and Steel 1997a; Xia 2007b, pp. 256–262), and the second uses the Fisher information matrix to obtain both the variances of parameters and covariances between parameters. The delta method, which often yields nice and clean mathematical expressions for the variance, is illustrated elsewhere. The method using the Fisher information matrix is shown below.

To estimate variance involving multiple parameters such as  $D_{\text{K80}}$  and  $\kappa$ , we first take the second-order partial derivatives of  $\ln L$  with respect to  $D_{\text{K80}}$  and  $\kappa$ , substituting the estimated  $D_{\text{K80}}$  and  $\kappa$  in Eqs. (12.22) and (12.23) into the second-order partial derivatives, arranging them into what is called a Fisher information matrix ( $M_{\text{FI}}$ ) below, and compute the matrix inverse of  $M_{\text{FI}}$  (designated by  $M_{\text{FI}}^{-1}$ ):

$$M_{\text{FI}} = \begin{bmatrix} -\frac{\partial^2 \ln L}{\partial \kappa^2} & -\frac{\partial^2 \ln L}{\partial \kappa \partial D_{\text{K80}}} \\ -\frac{\partial^2 \ln L}{\partial D_{\text{K80}} \partial \kappa} & -\frac{\partial^2 \ln L}{\partial D_{\text{K80}}^2} \end{bmatrix} \quad (12.24)$$

The diagonal elements of  $M_{\text{FI}}^{-1}$  are the variances for  $\kappa$  and  $D_{\text{K80}}$ , and the off-diagonal elements of  $M_{\text{FI}}^{-1}$  are covariances. The mathematical expression for the variance of  $\kappa$  is tedious but that for the variance of  $D_{\text{K80}}$  is simpler:

$$V(D_{K80}) = \frac{a^2 P + c^2 Q - (aP + cQ)^2}{N}, \text{ where}$$

$$a = \frac{1}{1 - 2P - Q}, b = \frac{1}{1 - 2Q}, c = \frac{a + b}{2} \quad (12.25)$$

With the aligned sequences in Fig. 12.4, we have  $N = 24$  and empirical  $P = 4/24$  and  $Q = 2/24$ . These lead to  $D_{K80} = 0.3151$ , and  $\kappa = 4.9126$ . The  $M_{\text{FI}}$  and  $M_{\text{FI}}^{-1}$  are

$$M_{\text{FI}} = \begin{bmatrix} 0.047435 & -0.286795 \\ -0.286795 & 49.641105 \end{bmatrix}$$

$$M_{\text{FI}}^{-1} = \begin{bmatrix} 21.84451668 & 0.126204037 \\ 0.126204037 & 0.020873724 \end{bmatrix} \quad (12.26)$$

where the two parameters are in the order of  $\kappa$  and  $D_{K80}$ , i.e., the variance is 21.8445 for  $\kappa$  and 0.0209 for  $D_{K80}$ . The off-diagonal elements are covariances between the two parameters. Note that the variance (which measures uncertainty about the parameter value) for  $D_{K80}$  is larger than that for  $D_{JC69}$ . This is why we should always aim to find the simplest possible but sufficient model for our data because more parameters in the model increase uncertainty. The last section in this chapter explains methods for model selection.

### 2.1.3 F84 and HKY85 Model

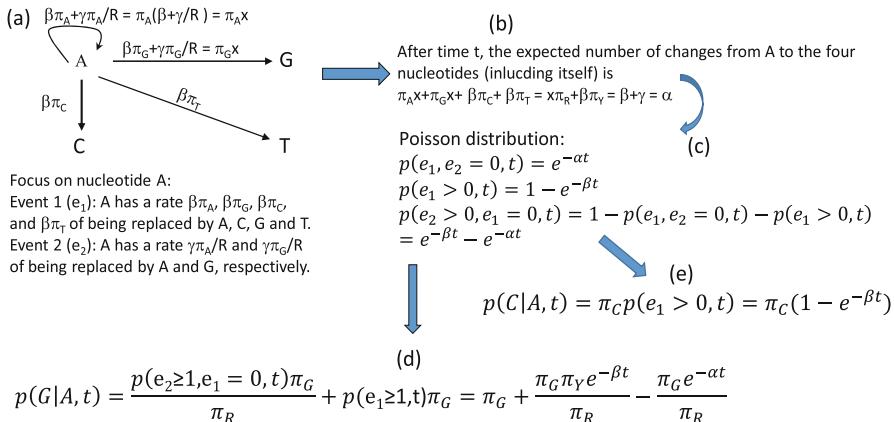
The F84 and HKY85 model accommodate not only the differential substitution rates between transitions and transversions but also different equilibrium nucleotide frequencies, in contrast to JC69 and K80 which assume equal equilibrium nucleotide frequencies. The same probabilistic reasoning used before can be applied to derive transition probabilities for the HKY80 model.

The rate matrix for the F84 model is

$$Q_{\text{F84}} = \begin{bmatrix} \text{A} & - & \beta\pi_G + \gamma\pi_G/\pi_R & \beta\pi_C & \beta\pi_T \\ \text{G} & \beta\pi_A + \gamma\pi_A/\pi_R & - & \beta\pi_C & \beta\pi_T \\ \text{C} & \beta\pi_A & \beta\pi_G & - & \beta\pi_T + \gamma\pi_T/\pi_Y \\ \text{T} & \beta\pi_A & \beta\pi_G & \beta\pi_C + \gamma\pi_C/\pi_Y & - \end{bmatrix} \quad (12.27)$$

where  $\pi_A$ ,  $\pi_G$ ,  $\pi_C$ , and  $\pi_T$  are equilibrium frequencies,  $\pi_R$  and  $\pi_Y$  are frequencies of purines and pyrimidines, and the diagonal elements are constrained by each row summing up to 0. Note that the  $Q_{\text{F84}}$  shown in Fig. 12.2 is parameterized differently, with  $\beta$  and  $\kappa$  instead of  $\beta$  and  $\gamma$ . If you replace  $\gamma$  in Eq. (12.27) by  $\kappa\beta$ , then  $Q_{\text{F84}}$  becomes identical to that in Fig. 12.2. It is easier to understand the F84 model by using  $Q_{\text{F84}}$  specified in Eq. (12.27).

We may view the F84 model as featuring two events ( $e_1$  and  $e_2$ ). Suppose we start with a nucleotide A. Event  $e_1$  occurs with rate  $\beta$ . When it occurs, the original A will



**Fig. 12.6** Derivation of transition probabilities based on the F84 model.  $\pi_A, \pi_G, \pi_C, \pi_T$ , and  $\pi_R, \pi_Y$  are equilibrium frequencies of A, C, G, T, purine (A + G), and pyrimidine (C + T). Event  $e_1$  occurs at a rate of  $\beta$  and leads to the original A being replaced by any of the four nucleotides according to their equilibrium frequencies, and event  $e_2$  occurs at a rate of  $\gamma$  and results in the original A being replaced by either A or G according to their frequencies in the purine pool, i.e.,  $\pi_A/\pi_R$  and  $\pi_G/\pi_R$ . I used  $x$  as a shorthand for  $\beta + \gamma/\pi_R$ . The rate  $\gamma$  does not appear in the final transition probabilities because it has been absorbed into  $\alpha$  which equals  $\beta + \gamma$  shown in (b).  $P(j|i, t)$  means the probability of changing from the original nucleotide  $i$  to nucleotide  $j$  after time  $t$  and is synonymous to  $p_{ij}(t)$  or simply  $p_{ij}$  in this chapter. See text for explanations on (a)–(g)

be replaced by a nucleotide drawn randomly from a nucleotide pool in which the nucleotide frequencies are the same as the equilibrium frequencies. This means that the original A has a rate of  $\beta\pi_A, \beta\pi_G, \beta\pi_C$ , and  $\beta\pi_T$  to change to A, G, C, and T, respectively, when  $e_1$  occurs. This is different from the K80 model where, when  $e_1$  occurs, the original A has a rate of 0.25 to change to any of the four nucleotides. Event  $e_2$  has a rate of  $\gamma$  to occur and will result in the original A being replaced by a purine drawn randomly from a purine pool with A and G frequencies specified as  $\pi_A/R$  and  $\pi_G/R$ . This means that the original A has a rate of  $\gamma\pi_A/\pi_R$  and  $\gamma\pi_G/\pi_R$  to change to A and G when  $e_2$  occurs. This is illustrated in Fig. 12.6a, where we use  $x$  to represent  $\beta + \gamma/\pi_R$ . Note that it is not a good idea to use  $\alpha$  to represent  $\beta + \gamma/\pi_R$  for two reasons. First, if we had started with a nucleotide C or T instead of A, then we would have  $\beta + \gamma/\pi_Y$  instead of  $\beta + \gamma/\pi_R$  which would force us to use  $\alpha_1$  and  $\alpha_2$  to distinguish between the two. A casual reader will then be misled to think that F84 has three rate parameters (i.e.,  $\beta$ ,  $\alpha_1$ , and  $\alpha_2$ ) without knowing that  $\alpha_1$  and  $\alpha_2$  are used as different functions of the same rate parameter  $\gamma$ . Second, I have reserved  $\alpha$  to represent  $\beta + \gamma$  in Fig. 12.6b which simplifies the derivation of transition probabilities illustrated in Fig. 12.6.

Note that whenever event  $e_1$  happens, the original A is replaced by A, C, G, and T with probabilities  $\pi_A, \pi_G, \pi_C$ , and  $\pi_T$ , no matter how many  $e_2$  events have occurred before or after the occurrence of  $e_1$ . This is similar to the scenario involving the K80 model, except that the K80 model assumes equal nucleotide frequencies. It might help to think of a long sequence with  $L$  sites being A at time 0. If these  $L$  sites each

have experienced at least one  $e_2$  event, then these sites will either be A or G with probabilities  $\pi_A$  and  $\pi_G$ , respectively, and we expect to have  $\pi_{AL}$  sites being A and  $\pi_{GL}$  sites being G. In contrast, if each of these  $L$  sites has experienced at least one  $e_1$  event, then the site will be replaced by A, C, G, or T with probabilities  $\pi_A, \pi_G, \pi_C$ , and  $\pi_T$ , and we expect to observe A, C, G, and T in  $\pi_{AL}$ ,  $\pi_{GL}$ ,  $\pi_{CL}$ , and  $\pi_{TL}$  sites, respectively. Any number of  $e_2$  events occurring before or after the  $e_1$  event does not change this expectation. This means that  $e_1$  erases  $e_2$ , but not vice versa. The occurrence of an  $e_2$  event is informative only when no  $e_1$  event has happened.

After time  $t$ , the total flow of the original A to the four nucleotides (including itself, Fig. 12.6a,b) is

$$\begin{aligned} \pi_Ax + \pi_Gx + \beta\pi_C + \beta\pi_T &= \pi_Rx + \pi_Y\beta = \pi_R(\beta + \gamma/\pi_R) + \pi_Y\beta = \beta + \gamma \\ &= \alpha \end{aligned} \quad (12.28)$$

so the probability that no substitution has happened during time  $t$  (Fig. 12.6c), according to Poisson distribution, is

$$p(e_1, e_2 = 0, t) = e^{-\alpha t} \quad (12.29)$$

The rate of A changing to A, G, C, and T through  $e_1$  is  $\beta\pi_A + \beta\pi_G + \beta\pi_C + \beta\pi_T = \beta$ , so the probability that at least one  $e_1$  has occurred during time  $t$  is

$$p(e_1 > 0, t) = 1 - e^{-\beta t} \quad (12.30)$$

The probability that  $e_2$  has happened but  $e_1$  has not is then

$$p(e_2 > 0, e_1 = 0, t) = 1 - p(e_1, e_2 = 0, t) - p(e_1 > 0, t) = e^{-\beta t} - e^{-\alpha t} \quad (12.31)$$

The reason for the condition that “ $e_1$  has not occurred” is because  $e_1$  event can erase  $e_2$  event. With these, it is easy to derive transition probability from A to G (Fig. 12.6d) as the summation of (1) a fraction of  $\pi_G$  of  $p(e_1 > 0, t)$ , which is the probability of  $e_1$  event that results in the original A being replaced by A, C, G, and T with probabilities  $\pi_A, \pi_G, \pi_C$ , and  $\pi_T$ , and (2) a fraction of  $\pi_G/\pi_R$  of  $p(e_2 > 0, e_1 = 0, t)$ , which is the probability that  $e_2$  events not erased by  $e_1$ . That is,

$$\begin{aligned} p(G|A, t) &= p(e_1 > 0, t)\pi_G + \frac{p(e_2 > 0, e_1 = 0, t)\pi_G}{\pi_R} \\ &= \pi_G + \frac{\pi_G\pi_Ye^{-\beta t}}{\pi_R} - \frac{\pi_Ge^{-\alpha t}}{\pi_R} \end{aligned} \quad (12.32)$$

From now on,  $p(j|i,t)$  will be written simply as  $p_{ij}$ , so  $p(G|A,t)$  is  $p_{AG}$ . With the same reasoning, we can derive transition probabilities for other A↔G and C↔T substitutions. Note that the two rate parameters in the F84 model  $\beta$  and  $\gamma$  have been re-parameterized into  $\alpha$  ( $=\beta + \gamma$ ) and  $\beta$  in Eq. (12.32). The transition probability from the original A to C (a transversion, Fig. 12.6e) is simply

$$p_{AC} = \pi_C p(e_1 > 0, t) = \pi_C (1 - e^{-\beta t}) \quad (12.33)$$

For other transversions, e.g.,  $p_{AT}$ , one just needs to replace  $\pi_C$  by  $\pi_T$ . The complete transition probability matrix for the F84 model is

$$P_{F84} = \begin{bmatrix} A & \pi_A + \pi_A \pi_Y x_1 + \pi_G x_2 & \pi_G + \pi_G \pi_Y x_1 - \pi_G x_2 & \pi_C (1 - e^{-\beta t}) & \pi_T (1 - e^{-\beta t}) \\ G & \pi_A + \pi_A \pi_Y x_1 - \pi_A x_2 & \pi_G + \pi_G \pi_Y x_1 + \pi_A x_2 & \pi_C (1 - e^{-\beta t}) & \pi_T (1 - e^{-\beta t}) \\ C & \pi_A (1 - e^{-\beta t}) & \pi_G (1 - e^{-\beta t}) & \pi_C + \pi_C \pi_R x_3 + \pi_G x_4 & \pi_T + \pi_T \pi_R x_3 - \pi_T x_4 \\ T & \pi_A (1 - e^{-\beta t}) & \pi_G (1 - e^{-\beta t}) & \pi_C + \pi_C \pi_R x_3 - \pi_C x_4 & \pi_T + \pi_T \pi_R x_3 + \pi_C x_4 \end{bmatrix} \quad (12.34)$$

where

$$x_1 = \frac{e^{-\beta t}}{\pi_R}, x_2 = \frac{e^{-\alpha t}}{\pi_R}, x_3 = \frac{e^{-\beta t}}{\pi_Y}, x_4 = \frac{e^{-\alpha t}}{\pi_Y} \quad (12.35)$$

As a quick check of the transition probabilities, we first note that when  $t = 0$  (or when  $\alpha = 0$  and  $\beta = 0$ ), then the diagonal elements are 1, and all off-diagonal elements are 0, which is what we expected. Second, when  $t = \infty$  with  $\alpha > 0$  and  $\beta > 0$ , then the transition probabilities will approach the equilibrium frequencies, which is also what we expected.

To obtain the distance for the F84 model ( $D_{F84}$ ), recall that a distance is defined as  $\mu t$  where  $\mu$  is the average substitution rate, i.e., substitution rates in Eq. (12.27) weighted by the equilibrium frequencies:

$$D_{F84} = 2\pi_A \pi_G (\beta t + \gamma t / \pi_Y) + 2\pi_T \pi_C (\beta t + \gamma t / \pi_Y) + 2\pi_Y \pi_R \beta t \quad (12.36)$$

Now we need to obtain  $\beta t$  and  $\gamma t$  in order to calculate  $D_{F84}$ . We can obtain  $\alpha t$  and  $\beta t$  and then obtain  $\gamma t = \alpha t - \beta t$ , remembering  $\alpha = \beta + \gamma$  [Fig. 12.6b and Eq. (12.28)]. The method we will use is the same as that for the K80 model, i.e., we obtain the expected transitions and transversions, designated  $E(S)$  and  $E(V)$ , respectively, from transition probabilities and equate them to the observed  $S$  and  $V$  to solve for  $\alpha t$  and  $\beta t$ . With the property of time reversibility, we have

$$\begin{aligned} E(S) &= 2\pi_A p_{AG} + 2\pi_C p_{CT} \\ E(V) &= 2\pi_A p_{AT} + 2\pi_A p_{AC} + 2\pi_G p_{GC} + 2\pi_G p_{GT} \end{aligned} \quad (12.37)$$

Equating  $E(S)$  and  $E(V)$  to the observed  $S$  and  $V$ , and solving these two equations with the two unknowns ( $\alpha t$  and  $\beta t$ ), we have

$$\alpha t = \ln \left( \frac{-2(\pi_A \pi_G \pi_R \pi_Y^2 + \pi_C \pi_T \pi_R^2 \pi_Y)}{S\pi_R^2 \pi_Y^2 - 2\pi_A \pi_G \pi_R \pi_Y^2 - 2\pi_C \pi_T \pi_R^2 \pi_Y + (\pi_A \pi_G \pi_Y^2 + \pi_C \pi_T \pi_R^2)V} \right) \quad (12.38)$$

$$\beta t = -\ln \left( 1 - \frac{V}{2\pi_R\pi_Y} \right) \quad (12.39)$$

Substitute  $\beta t$  and  $\gamma t$  ( $= \alpha t - \beta t$ ) into Eq. (12.36), and, after some algebraic manipulation, we have a more useful form of  $D_{F84}$ :

$$D_{F84} = \frac{2}{\pi_R\pi_Y} \left[ -(\pi_A\pi_G + \pi_C\pi_T)\pi_R\pi_Y \ln(x_1) + \pi_C\pi_T\pi_R \ln\left(\frac{x_2}{x_3}\right) + \pi_A\pi_G\pi_Y \ln\left(\frac{x_2}{x_3}\right) - \pi_R^2\pi_Y^2 \ln(x_1) \right] \quad (12.40)$$

where

$$\begin{aligned} x_1 &= 1 - \frac{V}{2\pi_R\pi_Y}; \\ x_2 &= (\pi_A\pi_G\pi_Y + \pi_C\pi_T\pi_R)(2\pi_R\pi_Y - V) \\ x_3 &= -S\pi_R^2\pi_Y^2 + 2\pi_A\pi_G\pi_R\pi_Y^2 + 2\pi_C\pi_T\pi_Y\pi_R^2 - \pi_A\pi_G\pi_Y^2V - \pi_C\pi_T\pi_R^2V \end{aligned} \quad (12.41)$$

To illustrate the calculation of  $D_{F84}$ , we may use the two aligned sequences in Fig. 12.4 which gives us  $\pi_A = 6/48$ ,  $\pi_C = 12/48$ ,  $\pi_G = 10/48$ ,  $\pi_T = 20/48$ ,  $S = 4/24$ ,  $V = 2/24$ ,  $\alpha t = 0.5778363341$ ,  $\beta t = 0.2076393648$ ,  $\gamma t = \alpha t - \beta t = 0.3701969693$ , and  $D_{F84} = 0.3198867427$ . The variance of the  $D_{F84}$  can be obtained by either the delta method or the method using Fisher information matrix.

A substitution model similar to the F84 model is the HKY85 model, with its rate matrix specified as

$$\mathcal{Q}_{HKY85} = \begin{bmatrix} A & - & (\beta + \gamma)\pi_G & \beta\pi_C & \beta\pi_T \\ G & (\beta + \gamma)\pi_A & - & \beta\pi_C & \beta\pi_T \\ C & \beta\pi_A & \beta\pi_G & - & (\beta + \gamma)\pi_T \\ T & \beta\pi_A & \beta\pi_G & (\beta + \gamma)\pi_C & - \end{bmatrix} \quad (12.42)$$

where  $(\beta + \gamma)$  is often written as  $\alpha$  and the diagonal elements are constrained by each row summing up to 0. The HKY85 model and the F84 model differ only in the specification of rates involving transitions.  $q_{AG}$  and  $q_{CT}$  for the HKY85 model are  $\pi_G(\beta + \gamma)$  and  $\pi_T(\beta + \gamma)$  as specified in Eq. (12.42), in contrast to  $\pi_G(\beta + \gamma/\pi_R)$  and  $\pi_T(\beta + \gamma/\pi_Y)$ , respectively, in the F84 model specified in Eq. (12.27). By comparing these rates, it becomes obvious that the F84 model would be equivalent to the HKY85 model if  $\pi_R = \pi_Y$ .

We can obtain the transition probabilities for the HKY85 model in the same way as that for the F84 model. In short, we again start with nucleotide A and envision two events  $e_1$  and  $e_2$ . Event  $e_1$  occurs with rate  $\beta$  and results in the original A replaced by any of the four nucleotides with probabilities equal to their respective equilibrium frequencies. Event  $e_2$  occurs with a rate  $\gamma$  and results in the original A being replaced by either A or G with the probabilities equal to their respective equilibrium frequencies. Fictionalized in this way, the expected number of substitutions after time  $t$  is

$\beta(\pi_A + \pi_G + \pi_C + \pi_T) + \gamma(\pi_A + \pi_G) = \beta + \gamma R$ . According to the Poisson distribution, the probability that no substitution has happened during time  $t$  is

$$p(e_1, e_2 = 0, t) = 1 - e^{-(\beta + \gamma R)} \quad (12.43)$$

The probability that at least one  $e_1$  occurred after time  $t$  is

$$p(e_1 > 0, t) = 1 - e^{-\beta t} \quad (12.44)$$

The probability that  $e_2$  has occurred but  $e_1$  has not is

$$\begin{aligned} p(e_2 > 0, e_1 = 0, t) &= 1 - p(e_1, e_2 = 0, t) - p(e_1 > 0, t) \\ &= e^{-\beta t} - e^{-(\beta + \gamma R)} \end{aligned} \quad (12.45)$$

The transition probability  $p(\text{G} \rightarrow \text{A}, t)$ , abbreviated as  $p_{AG}$ , is

$$\begin{aligned} p_{AG} &= \pi_G p(e_1 > 0, t) \\ &+ \frac{\pi_G}{\pi_R} p(e_2 > 0, e_1 = 0, t) = \pi_G + \frac{\pi_G \pi_Y e^{-\beta t}}{\pi_R} - \frac{\pi_G e^{-(\beta + \gamma R)t}}{\pi_R} \end{aligned} \quad (12.46)$$

In the same way, we can derive other transition probabilities which are shown below:

$$P_{\text{HKY}} = \begin{bmatrix} A & \pi_A + \pi_A x_1 + \pi_G x_2 & \pi_G + \pi_G x_1 - \pi_G x_2 & \pi_C(1 - e^{-\beta t}) & \pi_T(1 - e^{-\beta t}) \\ G & \pi_A + \pi_A x_1 - \pi_A x_2 & \pi_G + \pi_G x_1 + \pi_A x_2 & \pi_C(1 - e^{-\beta t}) & \pi_T(1 - e^{-\beta t}) \\ C & \pi_A(1 - e^{-\beta t}) & \pi_G(1 - e^{-\beta t}) & \pi_C + \pi_C x_3 + \pi_T x_4 & \pi_T + \pi_T x_3 - \pi_T x_4 \\ T & \pi_A(1 - e^{-\beta t}) & \pi_G(1 - e^{-\beta t}) & \pi_C + \pi_C x_3 - \pi_T x_4 & \pi_T + \pi_T x_3 + \pi_C x_4 \end{bmatrix} \quad (12.47)$$

where

$$x_1 = \frac{\pi_Y e^{-\beta t}}{\pi_R}; x_2 = \frac{e^{-(\beta + \gamma R)t}}{\pi_R}; x_3 = \frac{\pi_R e^{-\beta t}}{\pi_Y}; x_4 = \frac{e^{-(\beta + \gamma Y)t}}{\pi_Y} \quad (12.48)$$

As a quick check of the transition probabilities, we first note that when  $t = 0$  (or when  $\alpha = 0$  and  $\beta = 0$ ), then the diagonal elements are 1, and all off-diagonal elements are 0, which is what we expected. Second, when  $t$  approaches infinity with  $\beta > 0$  and  $\gamma > 0$ , then the transition probabilities will approach the equilibrium frequencies, which is also what we expected.

We cannot derive the distance for the HKY85 model by following the same approach as that for the F84 model. Hasegawa et al. (1985) have tried this approach but were not successful because there is no explicit solution for  $\beta t$  and  $\gamma t$ . However, if we treat the  $A \leftrightarrow G$  transition and  $C \leftrightarrow T$  transition separately, then we can solve

for  $\beta t$  and  $\gamma t$  (Rzhetsky and Nei 1995). In other words, we obtain one set of  $\beta t$  and  $\gamma t$  from observed A↔G transitions and transversions, and another set of  $\beta t$  and  $\gamma t$  from observed C↔T transitions and transversions.  $\beta t$  in the two sets are the same as that in Eq. (12.39), but  $\gamma t$  is different between the two sets of estimates. We can then take a weighted average of  $\gamma t$ . Admittedly, this does sound mathematically clumsy and explains why HKY85, while commonly implemented in a likelihood approach or Bayesian inference, is almost never used in a distance-based phylogenetic analysis.

Here is the protocol to get  $\beta t$  and  $\gamma t$  from HKY85. The expected numbers of A↔G and C↔T transitions, designated  $S_R$  and  $S_Y$ , respectively, and transversions are

$$\begin{aligned} E(S_R) &= 2\pi_A p_{AG} \\ E(S_Y) &= 2\pi_C p_{CT} \\ E(V) &= 2\pi_A p_{AT} + 2\pi_A p_{AC} + 2\pi_G p_{GC} + 2\pi_G p_{GT} \end{aligned} \quad (12.49)$$

Setting  $E(S_R)$  and  $E(V)$  to their observed  $S_R$  and  $V$ , and solving for  $\beta t$  and  $\gamma t$ , we have

$$\begin{aligned} \beta t &= -\ln \left( 1 - \frac{V}{2\pi_R \pi_Y} \right) \\ \gamma_R t &= \frac{1}{\pi_R} \ln \left( \frac{\pi_A \pi_G (2\pi_R \pi_Y - V)}{2\pi_A \pi_G \pi_R \pi_Y - S_R \pi_Y \pi_R^2 - \pi_A \pi_G \pi_Y V} \right) \end{aligned} \quad (12.50)$$

where  $\beta t$  is the same as that in Eq. (12.39), and  $\gamma_R t$  in Eq. (12.50) is  $\gamma t$  estimated from observed  $S_R$  and  $V$ .

We now obtain another set of solutions for  $\beta t$  and  $\gamma t$  by setting  $E(S_Y)$  and  $E(V)$  to their observed  $S_Y$  and  $V$ , and by solving for  $\beta t$  and  $\gamma t$ . We get the same  $\beta t$  but a different  $\gamma t$ :

$$\gamma_Y t = \frac{1}{\pi_Y} \ln \left( \frac{\pi_C \pi_T (2\pi_R \pi_Y - V)}{2\pi_C \pi_T \pi_R \pi_Y - S_Y \pi_R \pi_Y^2 - \pi_C \pi_T \pi_R V} \right) \quad (12.51)$$

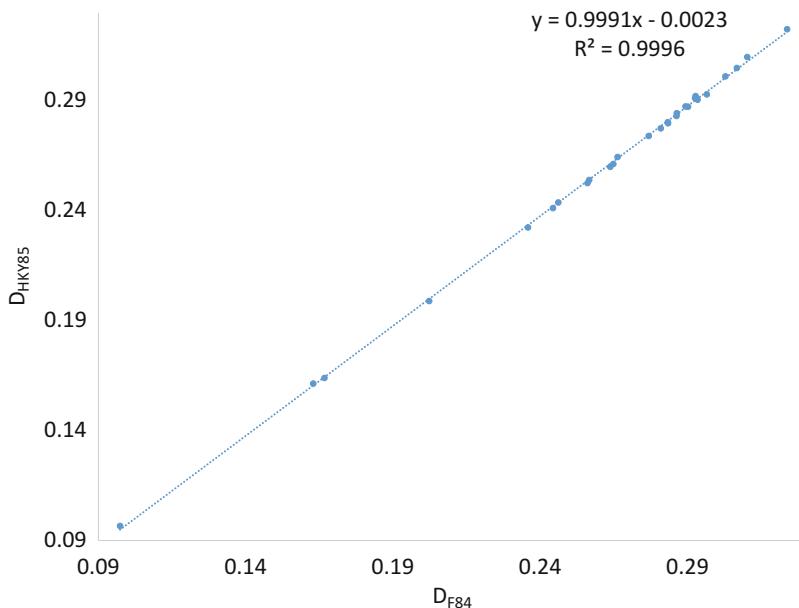
A weighted average of  $\gamma t$  could be

$$\gamma t = \pi_R \gamma_R t + \pi_Y \gamma_Y t \quad (12.52)$$

The distance for the HKY model is

$$D_{HKY85} = \mu t = 2\pi_A \pi_G (\beta t + \gamma t) + 2\pi_T \pi_C (\beta t + \gamma t) + 2\pi_Y \pi_R \beta t \quad (12.53)$$

To compute  $D_{HKY85}$  using the two aligned sequences in Fig. 12.4, we have  $\pi_A = 6/48$ ,  $\pi_C = 12/48$ ,  $\pi_G = 10/48$ ,  $\pi_T = 20/48$ ,  $S_Y = 4/24$ ,  $S_R = 0$ ,  $V = 2/24$ ,  $\beta t = 0.2076393648$ ,  $\gamma_R t = -0.2223239164$ ,  $\gamma_Y t = 1.047432870$ , weighted  $\gamma t = 0.624180608$ , and  $D_{HKY85} = 0.308904$ . For comparison, the same two sequences yield  $D_{F84} = 0.319887$ . One may be curious about the  $\gamma_R t$  value because one would have expected its minimum to be  $-\beta t$  so that  $q_{AG}$  and  $q_{GA}$  in Eq. (12.42)



**Fig. 12.7** Evolutionary distances from the HKY85 and F84 models are nearly identical

are 0. The  $\gamma_{Rt}$  value that is smaller than  $-\beta t$  may be viewed as follows. When event  $e_1$  happens that leads to an  $A \leftrightarrow G$  change, there is a very strong tendency (e.g., mediated by DNA repair or selection) to restore the new nucleotide back to the original.

In general,  $D_{HKY85}$  is slightly smaller than  $D_{F84}$ . I used the eight vertebrate COI sequences in the FASTA file VertCOI.fas that comes with DAMBE (Xia 2013) to compute both  $D_{HKY85}$  and  $D_{F84}$  (Fig. 12.7). The difference is minor, although  $D_{HKY85}$  is consistently but slightly smaller than  $D_{F84}$  (Fig. 12.7).

The HKY85 model itself may not carry much biological significance given the existence of the F84 model. However, the twists involved in computing the evolutionary distance, i.e., the separate estimation of  $\gamma_{A \leftrightarrow G}$  and  $\gamma_{C \leftrightarrow T}$ , lead very naturally to a very useful TN93 model that we will cover next.

#### 2.1.4 TN93 Model

We have come far, so far that we need hardly any extra effort to derive transition probabilities for the TN93 model. There are two equivalent specifications of the rate matrix for the TN93 model. The first is

$$\mathcal{Q}_{\text{TN93}} = \begin{bmatrix} A & - & \beta\pi_G + \gamma_R\pi_G/\pi_R & \beta\pi_C & \beta\pi_T \\ G & \beta\pi_A + \gamma_R\pi_A/\pi_R & - & \beta\pi_C & \beta\pi_T \\ C & \beta\pi_A & \beta\pi_G & - & \beta\pi_T + \gamma_Y\pi_T/\pi_Y \\ T & \beta\pi_A & \beta\pi_G & \beta\pi_C + \gamma_Y\pi_C/\pi_Y & - \end{bmatrix} \quad (12.54)$$

where the diagonal elements are constrained by each row summing up to 0. The second specification, shown in Fig. 12.2, simply replaces  $(\beta + \gamma_R/\pi_R)$  by  $\alpha_1$  and  $(\beta + \gamma_Y/\pi_Y)$  by  $\alpha_2$ . We see that TN93 is reduced to F84 if  $\gamma_R = \gamma_Y$  and to HKY85 if  $\gamma_R/\pi_R = \gamma_Y/\pi_Y$ .

The similarity between TN93 and F84 allows us to reuse Fig. 12.6 for deriving transition probabilities for TN93. We only need to add a subscript  $R$  to  $\gamma$  and  $\alpha$  in Fig. 12.6 so that we have  $\gamma_R$  and  $\alpha_R$  as rates for purine, keeping everything else the same, and we instantly obtain the transition probabilities for transitional substitutions between purines and for transversional substitutions as shown in Fig. 12.6. To get transition probabilities between pyrimidines, we can just replace the original nucleotide A in Fig. 12.6 by nucleotide C or T and rename  $\gamma$  and  $\alpha$  in Fig. 12.6 to  $\gamma_Y$  and  $\alpha_Y$ . Note that our  $\alpha_R = \beta + \gamma_R$ , and  $\alpha_Y = \beta + \gamma_Y$ .

The transition probability matrix for the TN93 model is

$$\mathcal{P}_{\text{TN93}} = \begin{bmatrix} A & \pi_A + \pi_A\pi_Yx_1 + \pi_Gx_2 & \pi_G + \pi_G\pi_Yx_1 - \pi_Gx_2 & \pi_C(1 - e^{-\beta t}) & \pi_T(1 - e^{-\beta t}) \\ G & \pi_A + \pi_A\pi_Yx_1 - \pi_Ax_2 & \pi_G + \pi_G\pi_Yx_1 + \pi_Ax_2 & \pi_C(1 - e^{-\beta t}) & \pi_T(1 - e^{-\beta t}) \\ C & \pi_A(1 - e^{-\beta t}) & \pi_G(1 - e^{-\beta t}) & \pi_C + \pi_C\pi_Rx_3 + \pi_Tx_4 & \pi_T + \pi_T\pi_Rx_3 - \pi_Tx_4 \\ T & \pi_A(1 - e^{-\beta t}) & \pi_G(1 - e^{-\beta t}) & \pi_C + \pi_C\pi_Rx_3 - \pi_Cx_4 & \pi_T + \pi_T\pi_Rx_3 + \pi_Cx_4 \end{bmatrix} \quad (12.55)$$

where  $x_1$  and  $x_3$  are the same as those in Eq. (12.35), but  $x_2$  has  $\alpha$  replaced by  $\alpha_R$  and  $x_4$  has  $\alpha$  replaced by  $\alpha_Y$ , i.e.,

$$x_1 = \frac{e^{-\beta t}}{\pi_R}, x_2 = \frac{e^{-\alpha_R t}}{\pi_R}, x_3 = \frac{e^{-\beta t}}{\pi_Y}, x_4 = \frac{e^{-\alpha_Y t}}{\pi_Y} \quad (12.56)$$

To obtain the distance for the TN93 model ( $D_{\text{TN93}}$ ), recall that a distance is defined as  $\mu t$  where  $\mu$  is the average substitution rate, i.e., substitution rates in Eq. (12.54) weighted by the equilibrium frequencies, so

$$D_{\text{TN93}} = 2\pi_A\pi_G(\beta t + \gamma_R t/\pi_R) + 2\pi_T\pi_C(\beta t + \gamma_Y t/\pi_Y) + 2\pi_Y\pi_R\beta t \quad (12.57)$$

Now we need to obtain  $\alpha_R t$ ,  $\alpha_Y t$ , and  $\beta t$ . The method we will use is the same as that for the K80 and F84 models, i.e., we obtain the expected numbers of  $A \leftrightarrow G$  transitions,  $C \leftrightarrow T$  transitions, and transversions, designated  $E(S_R)$ ,  $E(S_Y)$ , and  $E(V)$ , respectively, from transition probabilities, and equate them to the observed  $S_R$ ,  $S_Y$ , and  $V$  to solve for  $\alpha_R t$ ,  $\alpha_Y t$ , and  $\beta t$ :

$$\begin{aligned} E(S_R) &= 2\pi_A p_{AG} = S_R \\ E(S_Y) &= 2\pi_C p_{CT} = S_Y \\ E(V) &= 2\pi_A p_{AT} + 2\pi_A p_{AC} + 2\pi_G p_{GC} + 2\pi_G p_{GT} = V \end{aligned} \quad (12.58)$$

The resulting  $\alpha_R t$ ,  $\alpha_Y t$ , and  $\beta t$  are

$$\alpha_R t = \ln \left( \frac{2\pi_A \pi_G \pi_R \pi_Y}{2\pi_A \pi_G \pi_R \pi_Y - \pi_R^2 \pi_Y S_R - \pi_A \pi_G \pi_Y V} \right) \quad (12.59)$$

$$\alpha_Y t = \ln \left( \frac{2\pi_C \pi_T \pi_R \pi_Y}{2\pi_C \pi_T \pi_R \pi_Y - \pi_Y^2 \pi_R S_Y - \pi_C \pi_T \pi_R V} \right) \quad (12.60)$$

$$\beta t = -\ln \left( 1 - \frac{V}{2\pi_R \pi_Y} \right) \quad (12.61)$$

If one wishes to express  $D_{TN93}$  in SR, SY, and V, then one may just substitute  $\gamma_R t$ ,  $\gamma_Y t$ , and  $\beta t$  into Eq.(12.57), which yields

$$\begin{aligned} D_{TN93} &= \frac{2\pi_A \pi_G [\pi_Y \ln(x_1) + \ln(x_2)]}{\pi_R} + \frac{2\pi_C \pi_T [\pi_R \ln(x_1) + \ln(x_3)]}{\pi_Y} \\ &\quad - 2\pi_R \pi_Y \ln(x_1) \end{aligned} \quad (12.62)$$

where

$$\begin{aligned} x_1 &= 1 - \frac{V}{2\pi_R \pi_Y}; \\ x_2 &= \frac{2\pi_A \pi_G \pi_R \pi_Y}{2\pi_A \pi_G \pi_R \pi_Y - S_R \pi_R^2 \pi_Y - \pi_A \pi_G \pi_Y V}; \\ x_3 &= \frac{2\pi_C \pi_T \pi_R \pi_Y}{2\pi_C \pi_T \pi_R \pi_Y - S_Y \pi_Y^2 \pi_R - \pi_C \pi_T \pi_R V}. \end{aligned} \quad (12.63)$$

To illustrate the application of  $D_{TN93}$  with the two aligned sequences in Fig. 12.4, we have  $\pi_A = 6/48$ ,  $\pi_C = 12/48$ ,  $\pi_G = 10/48$ ,  $\pi_T = 20/48$ ,  $S_Y = 4/24$ ,  $S_R = 0$ ,  $V = 2/24$ ,  $\alpha_R t = 0.13353$ ,  $\alpha_Y t = 0.90593$ ,  $\beta t = 0.20764$ ,  $\gamma_R t = \alpha_R t - \beta t = -0.07411$ ,  $\gamma_Y t = \alpha_R t - \beta t = 0.69829$ , and  $D_{TN93} = 0.35299$ . The variance of the  $D_{TN93}$  can be obtained by either the delta method or the method using Fisher information matrix. Note that  $S_R = 0$  means no information for estimating  $\alpha_R t$  properly.

In short, the approach of deriving transition probabilities by probability reasoning can go a long way if one can do good bookkeeping. In particular, the probability reasoning approach is very useful for conceptual understanding. However, the approach becomes increasingly difficult with more complicated substitution models, and we discuss alternatives in the following sections.

## 2.2 Obtaining Transition Probabilities by Solving Differential Equations

To avoid too much redundancy, I will only illustrate this approach of deriving transition probabilities by solving (partial) differential equations with the JC60, K80, and TN93 models. One may use symbolic computation software such as MAPLE (<http://www.maplesoft.com>), which is a beautiful Canadian product, to solve equations. MAPLE essentially puts a capable mathematical assistance by your side.

### 2.2.1 JC69 Model

Suppose we start with a site occupied by a nucleotide A. Over time the site will change, and we need to model the probability that it is still occupied by A after time  $t$ , designated as  $P_{A(t)}$ . The JC69 model has only one rate  $\alpha$ . The rate of nucleotide A flowing away from A to the other nucleotides is  $3\alpha P_{A(t)}$ , and the rate flowing into A from the other three nucleotide is  $\alpha$ , so we have

$$\frac{dP_{A(t)}}{dt} = -(3\alpha)P_{A(t)} + \alpha[P_{G(t)} + P_{C(t)} + P_{T(t)}] \quad (12.64)$$

One may add  $[-\alpha P_{A(t)} + \alpha P_{A(t)}]$ , which is 0, to Eq. (12.64) to facilitate its simplification to

$$\frac{dP_{A(t)}}{dt} = -4\alpha P_{A(t)} + \alpha \quad (12.65)$$

You can solve this equation by hand, but if you want to use MAPLE to solve the equation, just type in the following MAPLE statement:

```
dsolve({diff(PA(t),t) = -4*a*PA(t) + a, PA(0) = 1}, PA(t));
```

where dsolve is for solving differential equations, the curly brackets enclose the differential equation and initial condition.

Solving the ordinary differential equation yields  $P_{A(t)}$  which is the probability that the original nucleotide A will remain as A after time  $t$  and equals the diagonal elements of the transition probability matrix for the JC69 model:

$$P_{A(t)} = \frac{1}{4} + \frac{3}{4}e^{-4at} = p_{AA} = p_{GG} = p_{CC} = p_{TT} \quad (12.66)$$

The off-diagonal elements are all equal in the JC69 model. Because each row sums to 1, the three off-diagonal elements are simply

$$p_{ij} = \frac{1 - p_{ii}}{3} = \frac{1}{4} - \frac{1}{4}e^{-4\alpha t} \quad (12.67)$$

To obtain the evolutionary distance between two aligned sequences, one simply equate  $3p_{ij}$  to  $p_{\text{diff}}$  (the proportion of sites differing between the two sequences), solve for  $\alpha t$ , and obtain the distance as  $3\alpha t$ . This leads to previously derived distance in Eq. (12.6).

The equilibrium frequencies are obtained nucleotide frequencies which do not change with time, e.g., when  $dP_{A(t)}/dt = 0$ , i.e.,  $-4\alpha P_{A(t)} + \alpha = 0$ . This yields  $P_{A(t)} = 1/4$ .

This approach can be generalized as

$$\begin{aligned} \left[ \frac{dP_{A(t)}}{dt} \frac{dP_{G(t)}}{dt} \frac{dP_{C(t)}}{dt} \frac{dP_{T(t)}}{dt} \right] &= P(t)Q = [P_{A(t)} \quad P_{G(t)} \quad P_{C(t)} \quad P_{T(t)}] \\ \times \begin{bmatrix} -a - b - c & a & b & c \\ g & -g - d - e & d & e \\ h & i & -h - i - f & f \\ j & k & l & -j - k - l \end{bmatrix} & \end{aligned} \quad (12.68)$$

where  $P(t)$  is the frequencies of the four nucleotides at time  $t$  and  $Q$  is the rate matrix. One can obtain analytical solutions from the JC69 model all the way to the TN93 model.

## 2.2.2 K80 Model

The rate matrix of Kimura's two-parameter model is in Fig. 12.2. Substituting it into Eq. (12.68) and solving the equations with the initial condition that  $P_{A,0} = 1$  and  $P_{G,0} = P_{C,0} = P_{T,0} = 0$  (i.e., we start with a nucleotide A) and the constrain of  $P_A + P_G + P_C + P_T = 1$ , we have

$$P_{A,t} = \frac{1}{4} + \frac{1}{4}e^{-4\beta t} + \frac{1}{2}e^{-2(\alpha+\beta)t} = p_{ii} \quad (12.69)$$

$$P_{G,t} = \frac{1}{4} + \frac{1}{4}e^{-4\beta t} - \frac{1}{2}e^{-2(\alpha+\beta)t} = P_s \quad (12.70)$$

$$P_{C,t} = P_{T,t} = \frac{1}{4} - \frac{1}{4}e^{-4\beta t} = P_v \quad (12.71)$$

where  $P_s$  and  $P_v$  are the probability of observing a transition or a transversion, respectively, at a site, and are often approximated by the proportion of sites with a transition or a transversion difference.

One can obtain the results above by issuing the following statements in the MAPLE interface:

```
eq1:=diff(Pa(t),t) = -Pa(t)*(a + 2*b) + Pg(t)*a + Pc(t)*b + Pt(t)*b;
eq2:=diff(Pg(t),t) = -Pg(t)*(a + 2*b) + Pa(t)*a + Pc(t)*b + Pt(t)*b;
eq3:=diff(Pc(t),t) = -Pc(t)*(a + 2*b) + Pa(t)*b + Pg(t)*b + Pt(t)*a;
eq4:=diff(Pt(t),t) = -Pt(t)*(a + 2*b) + Pa(t)*b + Pg(t)*b + Pc(t)*a;
sols:=dsolve({eq1,eq2,eq3,eq4,Pa(0) = 1,Pc(0) = 0,Pg(0) = 0,Pt(0) = 0},{Pa(t),
Pg(t),Pc(t),Pt(t)});
```

The MAPLE statements include specification of the four partial differential equations as eq1 to eq4, with the frequencies of nucleotides A, C, G, and T at time t specified as  $Pa(t)$ ,  $Pc(t)$ ,  $Pg(t)$ , and  $Pt(t)$ , respectively. Rate parameters  $\alpha$  and  $\beta$  are specified as  $a$  and  $b$ . We start with nucleotide A, so the initial conditions are  $Pa(0) = 1$  and  $Pc(0) = 0$ ,  $Pg(0) = 0$ , and  $Pt(0) = 0$ . Executing the statements generates Eqs. (12.69), (12.70), and (12.71).

The other transition probabilities can be obtained in the same way. Once we have the transition probabilities, we can obtain the evolutionary distance between two aligned sequences by (1) equating the  $P_s$  to the proportion of sites differing by a transition ( $P$ ) and  $2P_v$  to the proportion of sites differing by a transversion ( $Q$ ), (2) solving for  $\alpha t$  and  $\beta t$ , and 3) obtaining the distance as  $(\alpha t + 2\beta t)$ . This yields the previously derived distance in Eq. (12.18).

### 2.2.3 TN93 Model

We will use the rate matrix specified in Fig. 12.2 for the TN93 model. Substituting the  $Q$  matrix into Eq. (12.68) and solving the partial differential equations will generate transition probabilities. For example, we can use the following MAPLE statements to obtain transition probabilities  $p_{AA}$ ,  $p_{AG}$ ,  $p_{AC}$ , and  $p_{AT}$  by specifying the initial condition with a nucleotide A, i.e.,  $P_A(0) = 1, P_C(0) = 0, P_G(0) = 0, P_T(0) = 0$ :

```
eq1:= diff(Pa(t),t) = Pa(t)*(-G*a1-C*b-T*b) + Pg(t)*A*a1 + Pc(t)*A*b + Pt(t)
      *A*b;
eq2:= diff(Pg(t),t) = Pa(t)*G*a1 + Pg(t)*(-A*a1-C*b-T*b) + Pc(t)*G*b + Pt(t)
      *G*b;
eq3:= diff(Pc(t),t) = Pa(t)*C*b + Pg(t)*C*b + Pc(t)*(-A*b-G*b-T*a2) + Pt(t)
      *C*a2;
eq4:= diff(Pt(t),t) = Pa(t)*T*b + Pg(t)*T*b + Pc(t)*T*a2 + Pt(t)*(-A*b-G*b-
      C*a2);
sols:=dsolve({eq1,eq2,eq3,eq4,Pa(0) = 1,Pc(0) = 0,Pg(0) = 0,Pt(0) = 0},{Pa(t),
Pg(t),Pc(t),Pt(t)});
```

The MAPLE statements above include specification of the four partial differential equations as eq1 to eq4, with the frequencies of nucleotides A, C, G, and T at time t specified as  $Pa(t)$ ,  $Pc(t)$ ,  $Pg(t)$ , and  $Pt(t)$ , respectively. Rate parameters  $\alpha_1$ ,  $\alpha_1$ , and  $\beta$  are specified as  $a1$ ,  $a2$ , and  $b$ . We start with nucleotide A, so the initial conditions are

$P_A(0) = 1$  and  $P_C(0) = 0$ ,  $P_G(0) = 0$ , and  $P_T(0) = 0$ . Executing the statements will generate  $P_A(t)$ ,  $P_G(t)$ ,  $P_C(t)$ , and  $P_T(t)$ , which are the probabilities that the initial A will remain A or become G, C, or T. They are, after some algebraic manipulation,

$$\begin{aligned} P_A(t) &= \frac{\pi_A \pi_R + \pi_A \pi_Y e^{-\beta t} + \pi_G e^{-(R\alpha_1+Y\beta)t}}{\pi_R} = p_{AA} \\ P_G(t) &= \frac{\pi_G \pi_R + \pi_G \pi_Y e^{-\beta t} - \pi_G e^{-(R\alpha_1+Y\beta)t}}{\pi_R} = p_{AG} \\ P_C(t) &= \pi_C (1 - e^{-\beta t}) = p_{AC} \\ P_T(t) &= \pi_T (1 - e^{-\beta t}) = p_{AT} \end{aligned} \quad (12.72)$$

Note that the rate parameter  $\alpha_1$  appears only in the expression of  $p_{AA}$ ,  $p_{GG}$ ,  $p_{AG}$ , and  $p_{GA}$  and the rate parameter  $\alpha_2$  appears only in  $p_{CC}$ ,  $p_{TT}$ ,  $p_{CT}$ , and  $p_{TC}$ . All other  $p_{ij}$  expressions not shown in Eq. (12.72) can be easily obtained in the same way by changing the initial conditions. Once we have the transition probabilities, the distance can be obtained as shown previously in Eq. (12.57), (12.58), (12.59), (12.60), (12.61), and (12.62).

## 2.3 Obtain Transition Probabilities from the Rate Matrix by Using Matrix Exponential

The relationship between the transition probability matrix ( $P$ ) and the rate matrix ( $Q$ ) can be written as (Lanave et al. 1984)

$$P = e^{Qt} \quad (12.73)$$

This is a general approach, particularly useful when analytical solution is not available or tedious to write out. We will first illustrate this method with JC69 and K80 in which simple analytical solutions for  $P$  are available, but we will pretend not to have them. After gaining the familiarity with this approach, we will apply the method to derive  $P$  and evolutionary distance ( $D_{GTR}$ ) for the GTR model for which we do not have an analytical solution for  $P$ . The basic skill we need is to obtain eigenvalues and eigenvectors from the rate matrix and the inverse of a matrix, but there are many programs such as  $R$ , MAPLE, MatLab, etc., which can do it for us with one statement or two.

We will use the same set of sequence data throughout all these exercises. The two aligned sequences are 860 nt long, with site information shown in Table 12.1. By visual inspection of the values in Table 12.1, we see that A↔G transitions are the most frequent, followed by C↔T transitions, with the transversions least frequent. You will find  $D_{GTR}$  similar to  $D_{TN93}$  for these two sequences.

**Table 12.1** Empirical substitution pattern between two aligned sequences (S1 and S2) of 860 nucleotides long

	A	G	C	T
A	100	40	8	12
G	35	200	11	8
C	9	9	250	20
T	11	12	15	120
$\pi$	0.18314	0.299419	0.332558	0.184884

The first value in the first row, 100, means 100 sites have A in both sequences; the second value in the first row, 40, means 40 sites with A in S1 and G in S2; and so on

### 2.3.1 JC69 Model

Students often ask why JC69 has no rate to estimate. This is because (1) the evolutionary distance is defined as  $D = \mu t$  where  $\mu$  is the substitution rate equal to  $3\alpha$  in the case of JC69 model, i.e.,  $D = 3\alpha t$ , and (2) we cannot estimate  $\alpha$  and  $t$  separately, so that rate is scaled to 1, i.e.,  $3\alpha = 1$ , so that  $t$  is measured by  $D$ , i.e.,  $D = 3\alpha t = 1*t = t$ . So  $\alpha = 1/3$ , i.e.,

$$Q_{JC69} = \begin{bmatrix} -1 & 1/3 & 1/3 & 1/3 \\ 1/3 & -1 & 1/3 & 1/3 \\ 1/3 & 1/3 & -1 & 1/3 \\ 1/3 & 1/3 & 1/3 & -1 \end{bmatrix} \quad (12.74)$$

The sorted (in descending order) eigenvalues ( $\lambda$ ) and eigenvectors ( $U$ ) from  $Q_{JC69}$  are

$$\lambda = \begin{bmatrix} 0 \\ -4/3 \\ -4/3 \\ -4/3 \end{bmatrix}; U = \begin{bmatrix} 1 & -1 & -1 & -1 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix} \quad (12.75)$$

Now the transition probabilities for the JC69 model can be numerically calculated as

$$P = e^{QD} = \begin{bmatrix} 1 & -1 & -1 & -1 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} e^{0D} & 0 & 0 & 0 \\ 0 & e^{-4D/3} & 0 & 0 \\ 0 & 0 & e^{-4D/3} & 0 \\ 0 & 0 & 0 & e^{-4D/3} \end{bmatrix} \times \begin{bmatrix} 1/4 & 1/4 & 1/4 & 1/4 \\ -1/4 & 3/4 & -1/4 & -1/4 \\ -1/4 & -1/4 & 3/4 & -1/4 \\ -1/4 & -1/4 & -1/4 & 3/4 \end{bmatrix} \quad (12.76)$$

where the first matrix is  $\mathbf{U}$ , the second diagonal matrix has diagonal elements being  $e^{-\lambda D}$ ,  $D$  is  $D_{JC69}$  that we wish to find, and the last matrix is the inverse of  $\mathbf{U}$ , i.e.,  $\mathbf{U}^{-1}$ . I will use  $D$  for  $D_{JC69}$  to save some typing. Equation (12.76) simplified the calculation of  $P$  (by matrix exponential) from  $Q$  for different  $D$  values. To obtain  $P$  for a different  $D$ , we only need to change the middle diagonal matrix.

As we have mentioned several times in previous sections, the  $P$  matrix is needed to compute evolutionary distance ( $D$ ) between two sequences (e.g., the two aligned sequences with data summarized in Table 12.1) or the log-likelihood in tree construction. We can obtain  $D$  in two ways, similar to what we have already learned in previous sections with analytical solutions of the  $P$  matrix. We will repeat these two ways as I found this repetition helpful to enhance understanding in students.

The first way is based on the fact that the  $P$  matrix represents our expected sequence divergence between two sequences with a given  $D$ , and we want to find a  $P$  matrix that ideally matches our observed sequence divergence. For the JC69 model, the observed sequence divergence is measured by the proportion of sites that differ between the two sequences ( $p_{\text{diff}}$ ). From Table 12.1,  $p_{\text{diff}} = 0.22093$ . If we put  $D = 0.22093$  into Eq. (12.76), then the resulting  $P$  matrix is

$$P_{D=0.22093} = \begin{bmatrix} 0.808636992 & 0.063787669 & 0.063787669 & 0.063787669 \\ 0.063787669 & 0.808636992 & 0.063787669 & 0.063787669 \\ 0.063787669 & 0.063787669 & 0.808636992 & 0.063787669 \\ 0.063787669 & 0.063787669 & 0.063787669 & 0.808636992 \end{bmatrix} \quad (12.77)$$

Note that the  $P$  matrix has only two distinct values, the diagonal value of 0.808636992 and the off-diagonal value of 0.063787669. The expected divergence from the  $P$  matrix given  $D = 0.22093$ , designated by  $E(p_{\text{diff}})$ , is the summation of the off-diagonal elements in the  $P$  matrix above divided by 4, i.e.,

$$E(p_{\text{diff}}) = \frac{\sum_{i=1}^4 \sum_{j=1, j \neq i}^4 p_{ij}}{4} = \frac{0.063787669 \times 12}{4} = 0.191363007 \quad (12.78)$$

Apparently,  $E(p_{\text{diff}})$  ( $=0.191363007$ ) is smaller than the observed  $p_{\text{diff}}$  ( $=0.22093$ ), so we need to try different  $D$  values larger than 0.22093 so as to find one that will result in  $E(p_{\text{diff}})$  equal to the observed  $p_{\text{diff}}$ . There are efficient algorithms to automate this (Press et al. 1992), but suppose at this moment the god of mathematics happens to whisper  $D = 0.2617147$  into our ears (otherwise we may have to guess various  $D$  values for a long time without using a computer). Replacing  $D$  in Eq. (12.76) with this holy value, we get a new  $P$  matrix with diagonal values being 0.779069767 and off-diagonal values being 0.073643411. With this new  $P$  matrix,  $E(p_{\text{diff}}) = p_{\text{diff}} = 0.22093$ , so we conclude that the best  $D$  is 0.2617147 for the JC69 model. For those who do not believe in the god of mathematics, consult the bible of numerical recipes (Press et al. 1992), and you will get the same answer.

The second way of estimating  $D$  is by the likelihood method, i.e., we want to find a  $D$  (and the associated  $P$  matrix) so that the following likelihood function is maximized:

$$\ln L = \sum_{i=1}^4 \sum_{j=1}^4 N_{ij} \ln (\pi_i p_{ij}) \\ = N_{A,A} \ln (\pi_1 p_{1,1}) + N_{A,G} \ln (\pi_1 p_{1,2}) + \dots + N_{T,T} \ln (\pi_4 p_{4,4}) \quad (12.79)$$

where  $N_{ij}$  is the number of sites where one sequence is occupied by nucleotide  $i$  and the other sequence by nucleotide  $j$  (e.g.,  $N_{AA} = 100$  in Table 12.1),  $\pi_i$  values are equilibrium frequencies, and  $p_{ij}$  values are entries in the  $P$  matrix (i.e., transition probabilities from nucleotide  $i$  to nucleotide  $j$  during  $D$  which measures time when the average rate is scaled to be 1). With the JC69 model, we could collapse the 16 terms in  $\ln L$  into 2 terms because (1) all  $\pi_i = 1/4$  and (2) there are only two distinct  $p_{ij}$  values, with the 4 diagonal elements being the same and the 12 off-diagonal elements being the same. However, I choose to write them out in Eq. (12.79) because we eventually have to write out all individual terms with the GTR model.

What  $D$  value should we try first? As a distance corrected for multiple substitutions will always be greater than  $p_{\text{diff}}$ , the value of  $p_{\text{diff}}$  ( $=0.22093$ ) can serve as the low bound for  $D$ . If we do set  $D = 0.22093$ , then the  $P$  matrix has already been shown in Eq. (12.77), so that  $\ln L$  is

$$\ln L = 860 \ln (1/4) + 100 \ln (0.808637) \\ + 40 \ln (0.0637877) + \dots + 120 \ln (0.8086370) = -665.2286 \quad (12.80)$$

where  $1/4$  is the equilibrium frequencies for the JC69 model. The constant term  $860 * \ln(1/4)$  can be omitted in finding DJC69 to maximize  $\ln L$ .

We need to try various  $D$  values so as to find one that maximizes  $\ln L$ . There are efficient numerical algorithms to do the guessing for us (Press et al. 1992), but again suppose at this moment the god of mathematics happens to whisper  $D = 0.2617147$  into our ears. Replacing  $D$  in Eq. (12.76) with this holy value, we get a new  $P$  matrix with diagonal values being 0.779069767 and off-diagonal values being 0.073643411. This new  $P$  matrix, Eq. (12.79), leads to a larger  $\ln L$  value of  $-1855.1007$ . In fact this is the maximum  $\ln L$  we can get (rounded to 4 decimal points), so we declare that  $D_{\text{JC69}}$  is 0.2617147 which is exactly the same by the first method. It is also the same as that from Eq. (12.6) which, as we have shown before, is a likelihood estimate.

### 2.3.2 K80 Model

The K80 model has two rate parameters  $\alpha$  and  $\beta$  and is often replaced by and is often re-parameterized by  $\kappa = \alpha/\beta$  so that  $\alpha = \kappa\beta$ . Because we cannot separate rate from time, time is measured by distance with the average rate scaled to one. This implies that  $\alpha + 2\beta = 1$ , so

$$\begin{aligned}\alpha + 2\beta &= \kappa\beta + 2\beta = \beta(\kappa + 2) = 1 \\ \beta &= \frac{1}{\kappa + 2}\end{aligned}\quad (12.81)$$

Thus, the K80 model, although having two rates, ends up with only one rate ratio ( $\kappa$ ) to be estimated. A given  $\kappa$  determines both  $\beta$  and  $\alpha$ . For example,  $\kappa = 2$  implies  $\beta = 1/4$  and  $\alpha = 1/2$  according to Eq. (12.81).

As transitions do appear to occur more frequently than transversions in Table 12.1, we will start with a guess of  $\kappa = 2$ . This leads to the rate matrix  $Q$  as follows:

$$Q = \begin{bmatrix} A & -1 & 1/2 & 1/4 & 1/4 \\ G & 1/2 & -1 & 1/4 & 1/4 \\ C & 1/4 & 1/4 & -1 & 1/2 \\ T & 1/4 & 1/4 & 1/2 & -1 \end{bmatrix} \quad (12.82)$$

We again obtain the sorted eigenvalues ( $\lambda$ ) and eigenvectors ( $U$ ) and again arrange them in the following form as we have done before with Eq. (12.76). Note that eigenvectors are not unique, but the resulting  $P$  matrix will be the same as if all roads do lead to Rome:

$$\begin{aligned}P = e^{QD} &= \begin{bmatrix} 1 & -1 & 0 & -1 \\ 1 & -1 & 0 & 1 \\ 1 & 1 & -1 & 0 \\ 1 & 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} e^{0D} & 0 & 0 & 0 \\ 0 & e^{-1D} & 0 & 0 \\ 0 & 0 & e^{-1.5D} & 0 \\ 0 & 0 & 0 & e^{-1.5D} \end{bmatrix} \\ &\times \begin{bmatrix} 1/4 & 1/4 & 1/4 & 1/4 \\ -1/4 & -1/4 & 1/4 & 1/4 \\ 0 & 0 & -1/2 & 1/2 \\ -1/2 & 1/2 & 0 & 0 \end{bmatrix}\end{aligned}\quad (12.83)$$

where the first matrix is  $U$ , the second diagonal matrix has diagonal elements being  $e^{\lambda D}$ ,  $D$  is  $D_{\text{K80}}$  that we wish to find, and the last matrix is the inverse of  $U$ , i.e.,  $U^{-1}$ . Now we have two parameters ( $D$  and  $\kappa$ ) to estimate, in contrast to the JC69 model where we have only  $D$  to estimate.

Again, we have the same two methods for parameter estimation. The first is to find a pair of  $\{D, \kappa\}$  that will maximize  $\ln L$  defined in Eq. (12.79), and the second is to find a pair of  $\{D, \kappa\}$  so that the observed proportion of sites with a transition ( $S$ ) and the proportion of sites with a transversion ( $V$ ) are the same as their expected values  $E(S)$  and  $E(V)$  derived from the  $P$  matrix:

$$\begin{aligned}E(S) &= \frac{p_{AG} + p_{GA} + p_{CT} + p_{TC}}{4} = \frac{4p_{AG}}{4}; \\ E(V) &= \frac{p_{AC} + p_{AT} + \dots + p_{TA}}{4} = \frac{8p_{AC}}{4}\end{aligned}\quad (12.84)$$

Note that the four  $p_{ij}$  values involving A↔G and C↔T are identical, so are the eight transversional  $p_{ij}$  values. Also, we have used  $S$  and  $V$  for the proportion of sites being transitions and transversions, respectively, instead of using  $P$  and  $Q$  as we did in previous sections. This is because we have already used  $P$  for transition probability matrix and  $Q$  for rate matrix. The two methods lead to the same estimate for the JC69 and K80 models.

From Table 12.1, the observed  $S$  and  $V$  are

$$\begin{aligned} S &= \frac{40 + 35 + 20 + 15}{860} = 0.127906977; V \\ &= \frac{8 + 12 + 11 + 8 + 9 + 9 + 11 + 12}{860} = 0.093023256 \end{aligned} \quad (12.85)$$

If we fix  $\kappa = 2$ , then no matter what  $D$  values we try, we can't get  $E(S)$  and  $E(V)$  equal to the observed  $S$  and  $V$ . In fact, the closest we can get is when  $D = 0.260446116$  (fixing  $\kappa = 2$ ). The  $P$  matrix for this pair of  $\{D = 0.260446116, \kappa = 2\}$  is

$$P_{\kappa=2, D=0.260446116} = \begin{bmatrix} 0.7809789 & 0.104374942 & 0.057323079 & 0.057323079 \\ 0.104374942 & 0.7809789 & 0.057323079 & 0.057323079 \\ 0.057323079 & 0.057323079 & 0.7809789 & 0.104374942 \\ 0.057323079 & 0.057323079 & 0.104374942 & 0.7809789 \end{bmatrix} \quad (12.86)$$

Note that there are only three distinct values in the  $P$  matrix above, with the four diagonal values being 0.7809789, the four transitions being 0.104374942, and eight transversions being 0.057323079. From this  $P$  matrix,  $E(S) = 0.104374942$  (smaller than observed  $S$ ) and  $E(V) = 0.114646158$  (greater than the observed  $V$ ). The difference between  $E(S)$  and  $S$  and between  $E(V)$  and  $V$  suggests that  $\kappa$  has to be larger than 2. So we have to try different  $\{D, \kappa\}$  combinations to find the optimal pair. Again, suppose the god of mathematics whispered  $D = 0.265960816$  and  $\kappa = 3.167999267$  into our ears or, alternatively, we have obtained the magic  $D$  and  $\kappa$  values by using the bible of numerical recipes (Press et al. 1992). This new  $\kappa$  value implies the following  $Q$  matrix:

$$Q = \begin{bmatrix} A & -1 & 0.613003041 & 0.193498479 & 0.193498479 \\ G & 0.613003041 & -1 & 0.193498479 & 0.193498479 \\ C & 0.193498479 & 0.193498479 & -1 & 0.613003041 \\ T & 0.193498479 & 0.193498479 & 0.613003041 & -1 \end{bmatrix} \quad (12.87)$$

This  $Q$  matrix gives us  $\lambda$  and  $U$  as 0.265960816:

$$\lambda = \begin{bmatrix} 0 \\ -0.77399 \\ -1.613 \\ -1.613 \end{bmatrix}; U = \begin{bmatrix} -0.5 & 0.5 & 0 & 0.70711 \\ -0.5 & 0.5 & 0 & -0.70711 \\ -0.5 & -0.5 & -0.70711 & 0 \\ -0.5 & -0.5 & 0.70711 & 0 \end{bmatrix} \quad (12.88)$$

With this set of  $\lambda$  and  $U$ , if we try  $D = 0.265960816$ , then we will end up with the following  $P$  matrix:

$$P = \begin{bmatrix} 0.779069767 & 0.127906977 & 0.046511628 & 0.046511628 \\ 0.127906977 & 0.779069767 & 0.046511628 & 0.046511628 \\ 0.046511628 & 0.046511628 & 0.779069767 & 0.127906977 \\ 0.046511628 & 0.046511628 & 0.127906977 & 0.779069767 \end{bmatrix} \quad (12.89)$$

Now  $E(S)$  and  $E(V)$  as specified in Eq. (12.84) are 0.127906977 and 0.093023256, respectively. They are exactly the same as the observed  $S$  and  $V$ . So we conclude that the combination of  $D = 0.265960816$  and  $\kappa = 3.167999267$  are the best estimates. To use the maximum likelihood approach, we will also find the same  $D$  and  $\kappa$  combination to yield the highest  $\ln L$  ( $= -1831.1357$ ). That is,  $D = 0.265960816$  and  $\kappa = 3.167999267$  are also maximum likelihood estimates.

There are two awkward situations associated with deriving transition probabilities and estimating distances. The first is that substitution rates such as  $\kappa$  in the K80 model may need to be negative for  $E(S) = S$  and  $E(V) = V$ . A negative  $\kappa$  (which implies a negative evolutionary rate) makes no biological sense, so computer programs such as DAMBE(Xia 2013) sometimes would constrain  $\kappa \geq 0$ . Unfortunately, users often are not aware that, among the estimated distances in a distance matrix, some distance may be subject to such constraints and some are not. The second awkward situation is when the distance formula is inapplicable, e.g., when  $1-2P-Q \leq 0$  or  $1-2Q \leq 0$  for the K80 model. Such cases are discussed in the next section with the GTR model. For this reason, simultaneously estimated distances (Tamura et al. 2004; Xia 2009), which use information from all pairwise comparisons, are much more robust and typically generate better distance estimates and better trees than independently estimated distances which use only information from two sequences. Simultaneously estimated distances are covered in the chapter on distance-based phylogenetic methods.

### 2.3.3 GTR Model

We will illustrate two parameter estimation approaches, by maximizing likelihood and by minimizing  $\chi^2$ . After applying them to regular cases where a unique and biologically meaningful solution of parameters can be obtained, we will explore the approximate method for dealing with the two awkward cases we have mentioned before, one with negative rates and the other inapplicable cases.

The GTR model, as shown in Fig. 12.2, has six rate parameters  $a_1$  to  $a_6$ . Because we cannot separate rate and time, the average rate is scaled to be 1 so that the time is

measured by the distance, and the six rate parameters are typically re-parameterized as ratios over  $a_6$  (the rate for G↔T) by the following:

$$\begin{aligned} 2\pi_A\pi_G a_1 + 2\pi_A\pi_C a_2 + 2\pi_A\pi_T a_3 + 2\pi_C\pi_G a_4 + 2\pi_C\pi_T a_5 + 2\pi_G\pi_T a_6 &= 1 \\ \kappa_1 = a_1/a_6, \kappa_2 = a_2/a_6, \kappa_3 = a_3/a_6, \kappa_4 = a_4/a_6, \kappa_5 = a_5/a_6 \end{aligned} \quad (12.90)$$

where the first equation shows the scaling of average rate to 1 and equations in the second line defines the five rate ratio parameters. We will use the empirical nucleotide frequencies in Table 12.1 as equilibrium frequencies  $\pi_i$  so we don't have too many parameters to estimate. Solving these six equations for  $a_1$ – $a_6$ , we have

$$a_1 = \frac{\kappa_1}{2x}; a_2 = \frac{\kappa_2}{2x}; a_3 = \frac{\kappa_3}{2x}; a_4 = \frac{\kappa_4}{2x}; a_5 = \frac{\kappa_5}{2x}; a_6 = \frac{1}{2x} \quad (12.91)$$

where

$$x = \pi_A\pi_G\kappa_1 + \pi_A\pi_C\kappa_2 + \pi_A\pi_T\kappa_3 + \pi_C\pi_T\kappa_4 + \pi_G\pi_C\kappa_5 \quad (12.92)$$

This re-parameterization is similar to what we have done with the K80 model where two rate parameters ( $\alpha$  and  $\beta$ ) are re-parameterized to a single rate ratio parameter  $\kappa$ . Note that  $a_6$  is not free if  $\kappa_1$  to  $\kappa_5$  are fixed.

Now we have six parameters to estimate ( $\kappa_1$  to  $\kappa_5$  and the distance  $D$ ) if we use the empirical nucleotide frequencies for equilibrium nucleotide frequencies  $\pi_i$ . The computation is the same as before, i.e., we guess various parameter values to see which set of  $\kappa_i$  and  $D$  is the best by using either of the two criteria: (1) the observed transition probabilities matching the expected transition probabilities and (2) maximum likelihood. We will use both criteria simultaneously to evaluate and progressively improve the estimation of the six parameters.

### 2.3.3.1 Regular Cases

Regular cases in which  $\kappa_i$  and  $D$  have unique solutions are typically characterized by an empirical substitution matrix (Table 12.1) with diagonal values substantially greater than off-diagonal values. The observed transition probability matrix ( $P_{\text{obs}}$ ) can be obtained by first averaging the two corresponding off-diagonal values in Table 12.1 (because time reversibility implies that we cannot distinguish between A→G and G→A events and the two corresponding numbers should be averaged to generate a symmetrical matrix in columns 2–5 under  $C_{\text{obs}}$  in Table 12.2) and then divide each value in  $C_{\text{obs}}$  by their respective row sums. The resulting observed transition probabilities are shown under  $P_{\text{obs}}$  in Table 12.2.

We note that A↔G and C↔T transitions differ, but the four transversions have roughly similar  $C_{\text{obs}}$  values in Table 12.2. As a first approximation, we will just use  $\kappa_1 = 4$ ,  $\kappa_2 = 1$ ,  $\kappa_3 = 1$ ,  $\kappa_4 = 1$ ,  $\kappa_5 = 2$ , and  $D = 0.2$  as our initial guesses. These initial parameter values yield, according to Eqs. (12.91) and (12.92),

**Table 12.2** Obtaining the empirical transition probability matrix from observed substitution data in Table 12.1. Each pair of corresponding off-diagonal values in Table 12.1 is first averaged to give values in columns 2–5 designated as  $C_{\text{obs}}$  (observed counts)

$C_{\text{obs}}$	$P_{\text{obs}}$				$C_{\text{exp}}$				
	A	G	C	T	Sum	A	G	C	T
A	100	37.5	8.5	11.5	157.5	0.634921	0.238095	0.055968	0.073016
G	37.5	200	10	10	257.5	0.145631	0.776699	0.038835	0.38835
C	8.5	10	250	17.5	286	0.02972	0.034965	0.874126	0.061189
T	11.5	10	17.5	120	159	0.072327	0.062893	0.110063	0.754717

These values are then divided by the “sum” column to give the observed  $P$  matrix ( $P_{\text{obs}}$ ) in the last four columns. The last columns represent expected counts ( $C_{\text{exp}}$ , see text) corresponding to  $C_{\text{obs}}$

$$\begin{aligned}x &= 0.5920065 \\a_1 &= 3.378341, a_2 = 0.8445853; \\a_3 &= 0.8445853; a_4 = 0.8445853; \\a_5 &= 1.689171; a_6 = 0.8445853\end{aligned}\quad (12.93)$$

The rate matrix  $Q$  for GTR (Fig. 12.2) is determined by  $a_i$  and  $\pi_i$ , e.g.,  $Q_{AG} = a_1 * \pi_G = 1.0115383$ , and the diagonal elements are constrained by the row sum equal to 0:

$$Q = \begin{bmatrix} A & -1.4485621 & 1.0115383 & 0.2808737 & 0.1561501 \\ G & 0.6187079 & -1.0557317 & 0.2808737 & 0.1561501 \\ C & 0.154677 & 0.2528846 & -0.7198617 & 0.3123002 \\ T & 0.154677 & 0.2528846 & 0.5617475 & -0.969309 \end{bmatrix} \quad (12.94)$$

With our initial  $D = 0.2$ , the  $Q$  matrix above leads to expected transition probability matrix  $P$  as

$$P = \begin{bmatrix} A & 0.75935986 & 0.160218784 & 0.051686534 & 0.028734822 \\ G & 0.097997904 & 0.82158074 & 0.051686534 & 0.028734822 \\ C & 0.028463741 & 0.046535953 & 0.871009116 & 0.05399119 \\ T & 0.028463741 & 0.046535953 & 0.097116226 & 0.82788408 \end{bmatrix} \quad (12.95)$$

which differs much from  $P_{obs}$  shown in Table 12.2. The  $\ln L$  value, defined in Eq. (12.79), is  $-1796.492$  and is not the maximum as a slight increase in  $D$  will generate a higher  $\ln L$  value. If we do not change  $\kappa_i$ , but optimize the  $D$  value, we will find  $\ln L$  reaching its maximum of  $-1789.729$  when  $D = 0.269279$ . But the resulting expected  $P$  matrix still does not match  $P_{obs}$  in Table 12.2, although somewhat closer than with  $D = 0.2$ . This means that our initial  $\kappa_i$  values are not good enough and need to be optimized as well.

We can proceed by trying various combination of the  $\kappa_i$  and  $D$  values to see which combination will give us the highest  $\ln L$ . Efficient algorithms (Press et al. 1992) exist to find these  $\kappa_i$  and  $D$  values, but suppose the god of mathematics has again whispered the following into our ears:  $\kappa_1 = 5.2352836$ ,  $\kappa_2 = 0.8195326$ ,  $\kappa_3 = 2.4238604$ ,  $\kappa_4 = 0.5354983$ ,  $\kappa_5 = 1.8115580$ , and  $D = 0.2785347$ . These values would give the  $Q$  and  $P$  matrices as

$$Q = \begin{bmatrix} A & -1.7899718 & 1.2292351 & 0.2130239 & 0.3477128 \\ G & 0.7518622 & -1.0347866 & 0.1391862 & 0.1437382 \\ C & 0.1173121 & 0.1253162 & -0.5056578 & 0.2630294 \\ T & 0.3444325 & 0.2327835 & 0.4731222 & -1.0503382 \end{bmatrix} \quad (12.96)$$

$$P = \begin{bmatrix} A & 0.63492903 & 0.23810462 & 0.05395282 & 0.07301354 \\ G & 0.14563681 & 0.77668322 & 0.03883880 & 0.03884118 \\ C & 0.02971178 & 0.03496850 & 0.87410775 & 0.06121197 \\ T & 0.07232472 & 0.06290316 & 0.11010454 & 0.75466759 \end{bmatrix} \quad (12.97)$$

The  $P$  matrix, together with  $N_{ij}$  values in Table 12.1 and equilibrium frequencies, leads to  $\ln L = -1779.257$ . If you try to use any other combination of  $\kappa_i$  and  $D$  values, the  $\ln L$  will get smaller. Thus, the values whispered into our ears by the god of mathematics are the maximum likelihood estimates. The  $P$  matrix in Eq. (12.97), expected on the basis of  $D$ ,  $\pi_i$ , and the  $Q$  matrix and hereafter referred to as  $P_{\text{exp}}$ , is also nearly identical to  $P_{\text{obs}}$  in Table 12.2.  $P_{\text{obs}}$  and  $P_{\text{exp}}$  can be made more similar by increasing the precision of iteration.

We have not been explicit about the approach of matching the observed and the expected  $P$  matrix. With the substitution models from JC69 to TN93, we have analytical solutions for transition probabilities expressed as functions of  $D$  and substitution rates, so we simply equate the observed transition probabilities to their expected counterparts to obtain  $D$  and substitution rates. In the case of GTR, we do not have a handy analytical solution for the transition probabilities, so we aim to find a combination of  $D$  and  $\kappa_i$  values that will minimize the difference between  $P_{\text{obs}}$  and  $P_{\text{exp}}$ . One may note that if we multiply the values in  $P_{\text{exp}}$  in Eq. (12.97) by their corresponding row sum of counts (the “Sum” column in Table 12.2), we obtain the expected counts ( $C_{\text{exp}}$ ) corresponding to observed counts ( $C_{\text{obs}}$ ) in Table 12.2 (rounded to one decimal point). Thus one may simply use  $C_{\text{obs}}$  and  $C_{\text{exp}}$  in Table 12.2 to compute  $\chi^2$  as an operational criterion for matching  $P_{\text{obs}}$  and  $P_{\text{exp}}$ , i.e.,

$$\chi^2 = \sum_{i=1}^4 \sum_{j=i}^4 \frac{(C_{\text{obs},ij} - C_{\text{exp},ij})^2}{C_{\text{exp},ij}} \quad (12.98)$$

Note that Eq. (12.98) has 10 terms instead of 16 for the GTR model because the  $C_{\text{obs}}$  and  $C_{\text{exp}}$  are symmetrical due to the fact that we do not know the direction of nucleotide substitutions, e.g., we cannot distinguish between A→G and G→A, and consequently averaged the off-diagonal counts (Table 12.2). For  $i \neq j$ ,  $C_{\text{obs}}$  and  $C_{\text{exp}}$  values in Table 12.2 would be multiplied by 2. Ideally,  $\chi^2 = 0$ , which means perfect match between  $P_{\text{obs}}$  and  $P_{\text{exp}}$ . Thus, we have two similar ways to estimate  $D$  and  $\kappa_i$ . The likelihood method will find  $D$  and  $\kappa_i$  that maximize  $\ln L$  defined in Eq. (12.79), and the  $\chi^2$  method will find  $D$  and  $\kappa_i$  that minimizes  $\chi^2$  defined in Eq. (12.98), both with constraints of  $\kappa_i \geq 0$ .

Note that parameter estimation by minimizing  $\chi^2$  in Eq. (12.98) is a special case of weight least-squares method. The general form of WLS is to minimize the residual sum of squares (RSS):

$$\text{RSS} = \sum \left[ \frac{(o_i - e_i)^2}{w_i^m} \right] \quad (12.99)$$

where  $o_i$  and  $e_i$  are the observed and expected values,  $w_i$  is the weight typically equal to  $o_i$  or  $e_i$ , and  $m$  is an integer typically taking the value of 0, 1, or 2. If  $m = 0$ , Eq. (12.99) becomes ordinary least-squares (OLS) method. If  $m > 0$ , then RSS represents weighted least-squares (WLS) method. If  $m = 1$  and  $w_i = e_i$ , RSS in Eq. (12.99) becomes the same as  $\chi^2$  in Eq. (12.98). The Fitch-Margoliash method (Fitch and Margoliash 1967) for reconstructing phylogenies from a distance matrix has  $m = 2$  and  $w_i = o_i$  (where  $o_i$  is the estimated evolutionary distance for species pair  $i$ , and  $e_i$  is the patristic distance linking species pair  $i$  along the tree.)

In addition to the likelihood and the  $\chi^2$  methods, we can also obtain  $D$  and the  $Q$  matrix from  $P_{\text{obs}}$  in Table 12.2, as has been well illustrated by Felsenstein (Felsenstein 2004, pp. 208–209). Note that

$$P = e^{Qt} = e^{QD} \quad (12.100)$$

where  $t = D$  because the average rate is scaled to 1, so time is measured by distance. Thus, the matrix logarithm of  $P_{\text{obs}}$  in Table 12.2 gives us matrix  $QD$ :

$$\begin{aligned} QD &= \log(P_{\text{obs}}) \\ &= \begin{bmatrix} A & -0.49908268 & 0.2096211 & 0.03272184 & 0.09603549 \\ G & 0.34271386 & -0.28848954 & 0.03493475 & 0.0648886 \\ C & 0.05941871 & 0.03880131 & -0.14096078 & 0.13185534 \\ T & 0.09695011 & 0.04006713 & 0.07330419 & -0.29277942 \end{bmatrix} \end{aligned} \quad (12.101)$$

$D$  can then be obtained as

$$D = -\sum_{i=1}^4 QD_{i,i}\pi_i = 0.278789 \quad (12.102)$$

The rate matrix  $Q$  can then be obtained as

$$\begin{aligned} Q &= \frac{QD}{D} \\ &= \begin{bmatrix} A & -0.139138616 & 0.058439996 & 0.00912248 & 0.02677361 \\ G & 0.095544754 & -0.080427626 & 0.009739414 & 0.018090209 \\ C & 0.016565265 & 0.010817367 & -0.039298274 & 0.03675978 \\ T & 0.027028596 & 0.011170263 & 0.02043638 & -0.081623596 \end{bmatrix} \end{aligned} \quad (12.103)$$

which would lead to  $\kappa_i$  values similar to those from the maximum likelihood or  $\chi^2$  methods.

### 2.3.3.2 Two Awkward/Irregular Scenarios

The two awkward scenarios associated with independently estimated distances are (1) the best-fitting model has one or more negative substitution rates which make no

**Table 12.3** Empirical substitution pattern between two aligned sequences illustrating parameter estimation when one or more substitution rates are negative

	A	G	C	T
A	10	10	13	13
G	10	20	13	13
C	13	13	25	10
T	13	13	10	12

biological sense and (2) inapplicable cases (where distance formulae are not applicable). Two approaches have been used to handle them. The first is to constrain substitution rates to be nonnegative. The second is to use simultaneously estimated distances.

The first scenario with negative substitution rates may be simpler to illustrate with the K80 model than with the GTR model. Suppose we have empirical substitution pattern shown in Table 12.3 between two aligned sequences with 211 aligned sites, of which 40 sites differ by a transition, 104 sites by a transversion, and 67 sites are identical. So the observed proportion of transitions ( $S$ ) and transversions ( $V$ ) are 0.18957346 and 0.492890995, respectively. From these, we can obtain a  $D_{K80} = 2.091322$ , a negative  $\kappa = -0.033198953$ , and the expected  $S$  and  $V$ ,  $E(S)$  and  $E(V)$ , will be exactly equal to  $S$  and  $V$  leading to  $\chi^2 = 0$ . The  $D_{K80}$  and  $\kappa$  values also result in the highest likelihood ( $\ln L = -581.551246$ ).

However, because a negative  $\kappa$  implies a negative substitution rate which makes no biological sense, we may want to constrain  $\kappa \geq 0$ . With such a constraint, we may not have a unique solution. You may imagine a likelihood surface as a hill with a single peak (with maximum  $\ln L$ ) which requires a negative  $\kappa$  to reach it. If we prevent ourselves from having a negative  $\kappa$ , then we are condemned to stay somewhere below the peak, and there could be a circle around the peak with every point on the circle having the same  $\ln L$ . It would seem that we should sample this circle to get parameter values and then get an average of parameter values. However, in our particular case, a solution that reached convergence has an  $\ln L$  equal to  $-581.551753$  which is very close to the GTR model allowing a negative  $\kappa$ . This new constrained solution has  $\kappa = 0$ ,  $\beta = 0.5$ , and  $D = 2.056021268$ . Note that we tend to underestimate the distance whenever we force rates to be nonnegative.

The above treatment implies two models, one general model allowing negative  $\kappa$  and a constrained model forcing  $\kappa = 0$  which eliminates one parameter to estimate. A likelihood ratio test between the constrained and non-constrained models gives us  $\chi^2 = 2*(-581.551753 + 581.551246)$  with one degree of freedom, and the associated  $p = 0.974596409$ . Thus, there is no evidence that the constrained model with  $\kappa = 0$  is worse than the model allowing a negative  $\kappa$ .

We can also force  $\kappa = 0$  and estimate  $D$  by minimizing  $\chi^2$  following Eq. (12.98), but this has the same problem as the maximum likelihood method. One of the converged solutions has the same  $D = 2.056021268$ , with resulting  $E(S)$  and  $E(V)$  equal to 0.190112531 and 0.491812852, respectively, which are slightly different from the observed  $S$  and  $V$ . We have three observed  $C_{obs}$  values, being 40, 104, and 67 for transitions, transversions, and identical sites, respectively, and three

corresponding  $C_{\text{exp}}$  values, being  $40.11374408 [=E(S)*200]$ ,  $103.7725119 [=E(V)*211]$ , and  $67.11374407 (=211-40.11374408 - 103.7725119)$ . The resulting  $\chi^2$  value is

$$\begin{aligned}\chi^2 &= \frac{(40 - 40.11374408)^2}{40.11374408} + \frac{(104 - 103.7725119)^2}{103.7725119} + \frac{(67 - 67.11374407)^2}{67.11374407} \\ &= 0.001013994\end{aligned}\quad (12.104)$$

With such a small  $\chi^2$  value and one degree of freedom (one free parameter  $D$ ),  $p = 0.9746$ .

If we fit the data in Table 12.3 to the GTR model allowing negative  $\kappa$  values, then we will get  $\kappa_1 = -0.3016516$ ,  $\kappa_2 = 1.0981328$ ,  $\kappa_3 = 3.1294315$ ,  $\kappa_4 = 0.3936567$ ,  $\kappa_5 = -0.4275744$ , and  $D = 3.4131381$ , with  $\ln L = 577.4771$ . If we force  $\kappa_i \geq 0$ , then we will have multiple solutions with the same  $\ln L$ , and all with the estimated distance smaller than that estimated by allowing negative  $\kappa_i$ . There has been no comprehensive evaluation of whether constraining  $\kappa_i$  leads to better phylogenetic resolution.

The second awkward scenario is when the distance cannot be estimated. The transition probability matrix from GTR ( $P_{\text{exp}}$ ) is expected to have eigenvalues between 0 and 1 (Waddell and Steel 1997b). However,  $P_{\text{obs}}$  could have negative eigenvalues, making it impossible to take a matrix logarithm of  $P_{\text{obs}}$ . Such inapplicable cases occur often with highly diverged sequences. For example, we cannot have DJC69 if  $p_{\text{diff}}$  in Eq. (12.6) is equal to or greater than 0.75. Distance calculation is typically more likely to become inapplicable for more complicated models. For example, the distance formula in Eq. (12.18) for the K80 model becomes inapplicable when  $P = 0.2$  and  $Q = 0.5$ , but this is fine with the JC69 model because  $p_{\text{diff}}$  ( $=P + Q$ ) is smaller than 0.75. For the GTR model, Felsenstein (Felsenstein, pp. 209–210) included a fictitious data set (reproduced in Table 12.4) from which one cannot obtain  $D$  with the above method because  $P_{\text{obs}}$  has negative eigenvalues and therefore cannot have matrix logarithm. For such a data set, Felsenstein (Felsenstein, pp. 210) considers the correct distance as infinite, and Waddell and Steel (1997b) suggest to set the distance for such inapplicable cases as twice as large as the maximum computable distance when computing evolutionary distances for a set of aligned sequences. The reason for doing this is that inapplicable cases often arise not because sequences have really diverged for an infinitely long period of time but because of random sampling effect. The data in Table 12.4 has a  $D_{\text{K80}}$  of only 0.799964455, which is not suggestive of an infinitely large distance.

With the likelihood method or the  $\chi^2$  method, we can constrain  $\kappa_i \geq 0$  and try to find  $D$  and  $\kappa_i$  that will maximize  $\ln L$  or minimize  $\chi^2$ . One of the point that has one of the largest  $\ln L$  and the smallest  $\chi^2$  has  $D = 1.404823107$ ,  $\kappa_1 = 9862.39307$ ,  $\kappa_2 = 485.5764748$ ,  $\kappa_3 = 2484.380521$ ,  $\kappa_4 = 459.8089364$ , and  $\kappa_5 = 469.2547688$ . These  $\kappa_i$  values do look extraordinarily large. However the set of parameters does offer a good fit to the observed data (Table 12.4). With the resulting  $C_{\text{exp}}$  values shown in Table 12.4, the  $\chi^2$  value is

**Table 12.4** Observed substitutions between two sequences ( $C_{\text{obs}}$ ) that would make the GTR distance inapplicable.  $P_{\text{exp}}$  is the expected transition probability matrix derived with  $D = 1.404823107$ ,  $\kappa_1 = 9862.39307$ ,  $\kappa_2 = 485.5764748$ ,  $\kappa_3 = 2484.380521$ ,  $\kappa_4 = 459.8089364$ , and  $\kappa_5 = 469.2547688$ .  $C_{\text{exp}}$  is the expected substitutions based on  $P_{\text{exp}}$

$C_{\text{obs}}$				$P_{\text{exp}}$				$C_{\text{exp}}$				
A	G	C	T	A	G	C	T	A	G	C	T	
A	32	40	8	20	0.358602	0.371038	0.080386	0.189974	35.9	37.1	8.0	19.0
G	40	40	8	12	0.371038	0.420328	0.079525	0.129109	37.1	42.0	8.0	12.9
C	8	8	76	8	0.080386	0.079525	0.760303	0.079786	8.0	8.0	76.0	8.0
T	20	12	8	60	0.189974	0.129109	0.079786	0.601132	19.0	12.9	8.0	60.1

$$\chi^2 = \sum_{i=1}^4 \sum_{j=i}^4 \frac{(O - E)^2}{E} = \frac{(32 - 35.9)^2}{35.9} + \frac{(2 \times 40 - 2 \times 37.1)^2}{2 \times 37.1} \\ + \dots + \frac{(60 - 60.1)^2}{60.1} = 1.889137 \quad (12.105)$$

The associated  $p$  value is 0.9296 given  $\chi^2 = 1.889137$  and six degrees of freedom. Thus, the set of  $D$  and  $\kappa_i$  values offer a very good fit between the model and the data, suggesting  $D = 1.404823107$  as a statistically more reasonable alternative than an infinitely large distance or a distance twice as large as the largest applicable distance. It is odd to have a pairwise distance not depending on the two associated aligned sequences but on the most divergent pair of sequences with a definable distance.

An alternative approach that is better than constraining  $\kappa_i \geq 0$  is to use simultaneously estimated (SE) distances which uses all sequence pairs for parameter estimation (Tamura et al. 2004; Xia 2009) instead of using the independently estimated (IE) distances. SE distances eliminate or at least alleviate three serious problems with the IE approach for distance estimation: (1) inapplicable cases where the distance often cannot be computed for highly diverged sequences (Rzhetsky and Nei 1994; Tajima 1993; Zharkikh 1994); (2) internal inconsistency, with the substitution process between sequences A and B having  $\kappa_{AB}$  but that between sequences A and C having  $\kappa_{AC}$  (Felsenstein 2004, p. 200; Yang 2006, pp. 37–38); and (3) insufficient use of information because the computation of pairwise distances ignores information in other sequences that should also contain information about the divergence between the two compared sequences (Felsenstein 2004, p. 175; Yang 2006, p. 37). Both DAMBE (Xia 2013, 2017d) and MEGA (Kumar et al. 2016) implement SE distances. SE distances will be numerically illustrated in the chapter on distance-based phylogenetic methods.

All the models we have discussed above are time-reversible models. There are cases in which time reversibility is violated. For example, vertebrate genomes are heavily methylated and, as a consequence, are associated with strong C→T substitutions. In contrast, most invertebrate genomes are little methylated and not associated with strong C→T substitutions. Thus, the balance between C and T is different between vertebrate and invertebrates. This violates the time reversibility and cautions against analyzing vertebrate and invertebrate sequences in a single phylogenetic analysis with a single time-reversible model.

### 3 How Far Can We Trace Back the Evolutionary History?

Suppose we are dealing with rapidly evolving retroviruses whose per-generation transition probabilities are specified as

$$P = \begin{pmatrix} & A & G & C & T \\ A & 0.997 & 0.001 & 0.001 & 0.001 \\ G & 0.001 & 0.997 & 0.001 & 0.001 \\ C & 0.001 & 0.001 & 0.997 & 0.001 \\ T & 0.001 & 0.001 & 0.001 & 0.997 \end{pmatrix} \quad (12.106)$$

Note that each row of a transition probability matrix sums up to 1, i.e., after one generation, a nucleotide either stays the same with a probability of 0.997 or changes into another nucleotide with a probability of 0.001. This is equivalent to the simplest JC69 model (Jukes and Cantor 1969). Thus, if we start with a nucleotide A, which corresponds to a frequency vector [1,0,0,0], then in two generations the frequency vector will become [0.994,0.002,0.002,0.002]. This means that the starting nucleotide A, after two generations, will remain the same with a probability of 0.994 and become nucleotide G, C, or T each with a probability of 0.002:

$$\begin{aligned} [1 & 0 & 0 & 0] \begin{bmatrix} 0.997 & 0.001 & 0.001 & 0.001 \\ 0.001 & 0.997 & 0.001 & 0.001 \\ 0.001 & 0.001 & 0.997 & 0.001 \\ 0.001 & 0.001 & 0.001 & 0.997 \end{bmatrix} = [0.997 & 0.001 & 0.001 & 0.001] \\ [0.997 & 0.001 & 0.001 & 0.001] \begin{bmatrix} 0.997 & 0.001 & 0.001 & 0.001 \\ 0.001 & 0.997 & 0.001 & 0.001 \\ 0.001 & 0.001 & 0.997 & 0.001 \\ 0.001 & 0.001 & 0.001 & 0.997 \end{bmatrix} = [0.994 & 0.002 & 0.002 & 0.002] \end{aligned} \quad (12.107)$$

If we continue this multiplication for 3000 generations, the  $\pi$  vector will become [0.25,0.25,0.25,0.25] which means that all original historical information has been lost and we will have no information to infer what is the nucleotide to start with. In short, if a nucleotide has a probability of 0.001 to change into one of the three other nucleotides in each generation, then we cannot use the nucleotide sequences to reconstruct evolutionary history beyond 3000 generations because the sequences would have already reached full substitution saturation. Severe substitution saturation decreases the accuracy of phylogenetic reconstruction (Ritland and Clegg 1990; Xia and Lemey 2009; Xia et al. 2003b).

Human genomic mutation rate is  $\sim 2.5 \times 10^{-8}$  (Nachman and Crowell 2000), much lower than 0.001 in the previous fictitious example. This allows us to trace evolutionary history back much further. Assuming that most mutations in human genome are neutral so that we can use this mutation rate as a proxy of substitution rate all the way back to the origin of life, we will find the starting frequency vector of [1,0,0,0] reaching [0.250,0.250,0.250,0.250] in fewer than 100,000,000 generations. If the average generation time from the beginning of life to human is 1 year, then the DNA sequences would be in full substitution saturation after 100,000,000 year, and there is no way to trace evolutionary history beyond 100,000,000 years. It is likely that early organisms may have a generation time much shorter than 1 year (*Escherichia coli* replicate once every 20 min when nutrients are unlimited), so sequences may

reach full substitution saturation much earlier than 100,000,000 years. Thus, one should be cautious in reporting molecular dating results back to billions of years. One may be able to reconstruct phylogenetic relationships among very ancient lineages because stabilizing selection can preserve co-ancestral information. However, such selection-preserved characters should not be used for dating.

## 4 Selecting the Best-Fitting Substitution Model

There are two approaches for model selection (Burnham and Anderson 2002), one by the likelihood ratio test (LRT) for nested models and the other by information theoretic indices. Nested models refer to one general and one special model with the latter being a special case of the former. For example, the JC69 model and the K80 model are nested because the former is a special case of the latter which is reduced to the former if transition rates and transversion rates are all equal. In the framework of LRT, the null hypothesis is that the special model fits the data just as well as the general model. Rejecting the null hypothesis means that the special model performs significantly worse than the general model. One disadvantage of LRT, as in any other statistical significance test, is that we would be left with no decision when the null hypothesis is not rejected. However, the approach with information theoretic indices such as AIC or BIC is not immune to this problem. Although we always choose a model with smaller AIC or BIC as if we always arrive at a decision unambiguously, our decision obviously would be weak if AIC or BIC values are similar between the two models.

We will use the aligned sequences in Fig. 12.3 to illustrate the two model selection method. The advantage of using sequences in Fig. 12.3 is that different site patterns are easily counted. However, the aligned sequences are short (only 24 nt long). A very limited sample has two problems. First, it is expected to be subject to strong stochastic effect, and the resulting statistics may not be reliable. For example, LRTs assume that the sample is large enough for the resulting likelihood ratio chi-square to follow the  $\chi^2$  distribution. Small samples may result in wrong  $p$  values. Second, small samples typically do not provide enough statistical power for discriminating between alternative hypotheses. For selecting substitution models, ideally one should have aligned sequence lengths of a few hundred or longer. This requirement is not only for alleviating the two problems above but also to avoid the estimation based on no direct data. For example, TN93 allows  $A \leftrightarrow G$  and  $C \leftrightarrow T$  transitions to have different rates, so we need to estimate  $\alpha_R t$  and  $\alpha_Y t$ . However, if we observe only  $C \leftrightarrow T$  transitions but no  $A \leftrightarrow G$  transitions, then our estimation of  $\alpha_R t$  becomes weak. For GTR, we ideally should have data with not only observed  $A \leftrightarrow G$  and  $C \leftrightarrow T$  transitions but also the four distinct types of transversions. Missing some of them in observed data weakens parameter estimation. The short sequence alignment in Fig. 12.3 highlights such problems.

## 4.1 Likelihood Ratio Test for Alternative Substitution Models

The test statistic for likelihood ratio test (LRT) is the log-likelihood ratio chi-square defined as  $2*(\ln L_G - \ln L_S)$  where  $\ln L_G$  and  $\ln L_S$  are the maximum log-likelihood of the general and the special models, respectively. It is sometimes abbreviated as  $2\Delta\ln L$ . This statistic, if derived from a large amount of data, follows approximately the  $\chi^2$  distribution with the degree of freedom being the difference in the number of parameters between the two models (Wilks 1938). That is, if the special and the general models have  $N_S$  and  $N_G$  parameters, respectively, then the degree of freedom ( $DF$ ) is  $(N_G - N_S)$ , which is sometimes written as  $DF = \Delta p$ .

That a substitution model has  $N$  parameters is equivalent to saying the model has  $N$  unknowns in its transition probability matrix ( $P$  matrix) that needs to be estimated from the sequence data. A model can be expressed in different ways. For example, the  $P$  matrix for the JC69 model can be expressed either as functions of  $\alpha t$  or as functions of  $D_{JC69}$  ( $= 3\alpha t$ ). Similarly, the  $P$  matrix of K80 can be expressed either as functions of  $\alpha t$  and  $\beta t$  or as functions of  $D_{K80}$  ( $= \alpha t + 2\beta t$ ) and  $\kappa$  ( $= \alpha/\beta$ ). In the same way, the  $P$  matrix for TN93 can be expressed as three rate parameters (Table 12.5) plus frequency parameters or as  $\kappa_R$ ,  $\kappa_Y$ , and  $D_{TN93}$  plus frequency parameters. The  $P$  matrix of GTR can be expressed as functions of six rate parameters plus frequency parameters or as five rate ratio parameters ( $\kappa_1$  to  $\kappa_5$ ) and  $D_{GTR}$  plus frequency parameters. However, the number of parameters and  $\ln L$  will remain the same no matter how we re-parameterize the model.

**Table 12.5** Parameter values and  $\ln L$  for JC69, K80, TN93, and GTR models. The equilibrium frequencies for A, G, C, and T are 0.1250, 0.2083, 0.2500, and 0.41667, respectively, and are needed for TN93 and GTR (i.e., three additional parameters as the four frequencies need to sum up to 1). AIC and BIC values for each model are also listed

Model	Parameter <sup>a</sup>	Value	$\ln L$	AIC	BIC
JC69	$D$	0.3041	-53.3588	108.718	109.896
K80	$D$	0.3151	-51.9725	107.945	110.301
	$\kappa$	4.9126			
TN93	$\alpha_R t^b$	0.13354	-47.4428	106.886	113.954
	$\alpha_Y t^b$	0.88225			
	$\beta t^b$	0.20765			
GTR	$a_{A \leftrightarrow G}^c$	0	-40.0172	98.034	108.637
	$a_{A \leftrightarrow C}^c$	0			
	$a_{A \leftrightarrow T}^c$	0			
	$a_{G \leftrightarrow C}^c$	-0.0612			
	$a_{C \leftrightarrow T}^c$	0.4515			
	$a_{G \leftrightarrow T}^c$	0.2464			

<sup>a</sup>D: evolutionary distance

<sup>b</sup>Defined in Eqs. (12.59), (12.60), and (12.61). The three rate parameters in TN93 can be re-parameterized to  $\kappa_R$ ,  $\kappa_Y$ , and  $D_{TN93}$

<sup>c</sup>These six rate parameters can be re-parameterized into five rate ratio parameters and  $D_{GTR}$

Suppose we want to test whether K80 provides better fit to the aligned sequence pair in Fig. 12.3 than JC69. For the JC69 model,  $\ln L$  is specified in Eq. (12.7), which contains only one distance parameter ( $D_{JC69}$ ). Maximum  $\ln L_{JC69}$  is reached when  $D_{JC69} = 0.3041$  and resulting  $\ln L_{JC69} = -53.3588$ . Similarly,  $\ln L$  for the K80 model is specified in Eq. (12.21) with a distance ( $D_{K80}$ ) and a  $\kappa$  parameter. Maximum  $\ln L_{K80}$  ( $= -51.9725$ ) is reached with  $D_{K80} = 0.3151$  and  $\kappa = 4.9126$ . So  $2\Delta\ln L = 2.7726$  and  $DF = \Delta p = 1$ . The  $p$  value, given  $2\Delta\ln L = 2.7726$  and  $DF = 1$ , is 0.0959, which is not significant at 0.05 significance level.

It is relevant here to mention that all statistical significance tests confer two key pieces of information: (1) the effect size which is independent of sample size and (2) the  $p$  value which depends on sample size. The effect size is the deviation of observation from expectation based on the null hypothesis. For example, in a t-test between two group means ( $\bar{x}_1$  and  $\bar{x}_2$ ) with the null hypothesis of ( $\bar{x}_1 = \bar{x}_2$ ), the observed difference ( $d_o$ ) is ( $\bar{x}_1 - \bar{x}_2$ ), and the expected difference ( $d_e$ ) based on the null hypothesis is 0. So the effect size is  $d_o - d_e = d_o = (\bar{x}_1 - \bar{x}_2)$ . This effect size does not change with sample size. In contrast, the  $t$  statistic and the resulting  $p$  value depend on sample size. Similarly, in bivariate linear regression, the expected slope ( $b_e$ ) under the null hypothesis of no linear relationship between the two variables is  $b_e = 0$ , so the effect size is the difference between the observed (fitted from the observed data) slope ( $b_o$ ) and  $b_e$ , i.e.,  $b_o - b_e = b_o$ . This effect size does not change with sample size. In contrast, in the t-test testing the significance of the slope, the  $t$  statistic and the associated  $p$  value depend on sample size.

In our LRT above contrasting K80 and JC69, the null hypothesis is that JC69 is just as good as K80. Under this null hypothesis,  $\kappa = 1$  (which is assumed by the JC69 model). The observed  $\kappa$  is 4.9126 (estimated from the observed data). So the effect size is (4.9126–1), and the  $p$  value is 0.0959. If the aligned sequences are twice as long, everything else being equal, then we would still have  $\kappa = 4.9126$ , but  $2\Delta\ln L$  will become 5.5452 which is twice as large as before (the statistic increases linearly with sample size, everything else being equal), and the  $p$  value would then be 0.0185 (which would lead to rejection of the JC69 model at 0.05 significance level).

Table 12.5 lists parameters, their estimated values, and  $\ln L$  for JC69, K80, TN93, and GTR so that we can contrast any of the two nested hypotheses. For example, applying LRT to JC69 and TN93, we will have  $2\Delta\ln L$  equal to 11.8320 with five degrees of freedom (TN93 has three rate parameters and three frequency parameters in contrast to JC69 with only one parameter, Table 12.5) and  $p$  equals 0.0372, and we reject JC69 and conclude that TN93 is a better model for the aligned sequences in Fig. 12.3. Similarly, contrasting between GTR and JC69 leads to  $2\Delta\ln L = 26.6832161$ ,  $DF = 8$ , and  $p = 0.0008$ .

The accuracy and validity of LRT has often been questioned (Pinheiro and Bates 2000, pp. 82–93), mainly with the concern of whether the statistic  $2\Delta\ln L$  follows the  $\chi^2$  distribution. Unfortunately, there has not yet been a simple alternative for biologists. However, one can take some comfort from the observation that if one simulates sequence evolution according to a specific substitution model, the specific substitution model used to simulate the sequences is almost always identified by LRT as the best model.

## 4.2 *Information-Theoretic Indices for Substitution Model Selection*

Several information-theoretic indices have been formulated for model selection. The Akaike information criterion or AIC (Akaike 1973; 1974) is defined as

$$\text{AIC} = -2 \ln L + 2p \quad (12.108)$$

where  $p$  is the number of parameters. The smaller the AIC value, the better the model. The AIC values for JC69, K80, TN93, and GTR (Table 12.5) show that AIC favors the more complicated model, with GTR having the smallest AIC value (Table 12.5).

Bayesian information criterion or BIC (Schwarz 1978) is defined as

$$\text{BIC} = -2 \ln L + p \ln (n) \quad (12.109)$$

where  $n$  is the sample size (or sequence length in our case). AIC and BIC differ only in the second term, with BIC penalizing parameter-rich models more than AIC. GTR is again the best model because it has the smallest BIC value (Table 12.5), but JC69 is better than K80 which is in turn better than TN93.

## Postscript

It is thought-provoking that, while a nice mathematical expression of evolutionary distance based on the HKY85 model is difficult to obtain, it comes easy for the more general TN93 model with one additional parameter. This is somewhat analogous to an ancient Arabic fable of the 18th camel in which a father, upon his deathbed, instructed his three sons to divide his property of 17 camels in the following way: 1/2 to the oldest, 1/3 to the second, and 1/9 to the youngest. As 17 cannot be divided by 2, 3, or 9, the three sons went to consult a wise man. The wise man instructed the sons to imagine that their father actually left 18 camels for them, so the oldest son got 9 ( $= 1/2 \times 18$ ), the second son got 6 ( $= 1/3 \times 18$ ), and the youngest got 2 ( $= 1/9 \times 18$ ). The total number of camels they took away ended up to be exactly 17.

Thus, one additional (virtual) camel simplified the solution of dividing the inheritance, and one more parameter in the TN93 model simplified the solution of evolutionary distance. More can be simpler! May you always have your 18th camel handy!

# Chapter 13

## Protein Substitution Model and Evolutionary Distance



### 1 Introduction

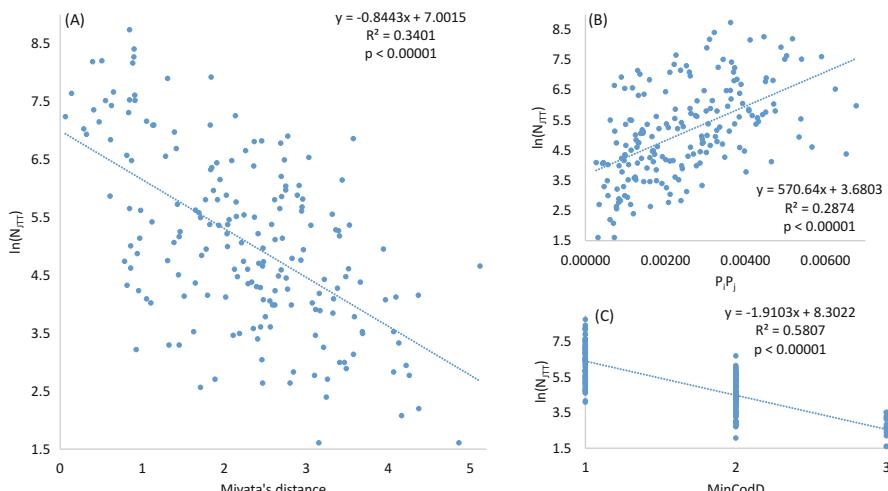
The availability of protein sequence alignment almost immediately led to the recognition of two patterns. First, synonymous substitutions occur much more frequently than nonsynonymous substitutions (Kimura 1968; King and Jukes 1969; Miyata and Yasunaga 1980) which eventually resulted in the formulation of the neutral theory of molecular evolution (Kimura 1977, 1983). Second, amino acid replacement occurs more frequently between similar amino acids than between different amino acids (Grantham 1974; Miyata et al. 1979). For example, amino acid replacement involving Asp↔Glu (both negatively charged and of similar size), Arg↔Lys (both positively charged and of similar size), and Ala↔Gly (both small and non-polar) occurs much more frequently than those involving dissimilar ones, e.g., Gly-Arg. This remains true when codon differences are controlled for (Xia and Li 1998).

Amino acid similarity is often measured by two indices such as Grantham's (Grantham 1974) and Miyata's (Miyata et al. 1979) distances. Grantham's distance combines the volume, the polarity, and the side chain composition, and Miyata's distance incorporates only volume and polarity. We expect the observed replacement frequencies to decrease with these amino acid distances, which have been confirmed in numerous occasions involving diverse array of organisms (Grantham 1974; Miyata et al. 1979; Palidwor et al. 2010; Xia 1998b; Xia and Li 1998; Xia and Xie 2002). Table 13.1 shows partial data for such confirmation. The column headed “ $N_{JTT}$ ” is the observed frequency of pairwise replacement inferred from amino acid sequence alignment (Jones et al. 1992), and “MinCodD” is the minimum codon distance defined as the number of site differences between two most similar codons, one from each codon family. For example, Ala is coded by codon family GCN (where N is any nucleotide), and Lys is coded by codon family AAR (where R is a purine). The two most similar codons, one from each codon family, could be GCA and AAA, or GCG and AAG, both with MinCodD = 2.

**Table 13.1** Observed frequencies of amino acid replacement ( $N_{JTT}$ ) compiled by Jones et al. (1992), together with Miyata's distance (Miyata) and minimum codon distance (MinCodD, defined as the number of site differences between two most similar codons, one from each codon family)

AA1	AA2	Miyata	MinCodD	$P_i P_j$	$N_{JTT}$
Arg	Ala	2.9152	2	0.003934	426
Asn	Ala	1.7729	2	0.003295	333
Asn	Arg	2.0357	2	0.002309	185
Asp	Ala	2.3613	1	0.003858	596
Asp	Arg	2.3352	2	0.002703	80
Asp	Asn	0.6498	1	0.002264	2134
Cys	Ala	1.3888	2	0.001438	159
Cys	Arg	3.0567	1	0.001007	214
Cys	Asn	2.8202	2	0.000844	54
Cys	Asp	3.4674	2	0.000988	20
...	...	...	...	...	...
Val	Trp	2.5036	2	0.000916	44
Val	Tyr	1.5137	2	0.002111	63

$P_i$  is the frequency of amino acid  $i$



**Fig. 13.1** Amino acid replacement frequencies depend on amino acid dissimilarity (a), residue frequencies (b), and codon similarity (b). Note that  $N_{JTT}$  has been log-transformed in the Y-axis. Output from DAMBE (Xia 2013. 2017d)

There is strong correlation between  $N_{JTT}$  and Miyata's distance (Fig. 13.1a), reflecting strong purifying selection against amino acid replacement involving very different amino acids. Replacing Miyata's distance by Grantham's distance will generate similar results. The relationship in Fig. 13.1a strongly suggests that any

mechanical model aims to explain amino acid replacement or nonsynonymous codon substitutions should include amino acid dissimilarity measures. That is, the substitution rate should be a function of amino acid dissimilarity, as well as amino acid frequencies (more frequent amino acids should be observed more often in amino acid replacement than rare amino acids).

While the negative relationship between  $\ln(N_{JTT})$  and Miyata's distance in Fig. 13.1a is strong, many points do deviate substantially from the predicted value. One of the possible reasons is that purifying selection is not homogenous over different sites of proteins (Xia 1998b). Important sites may be subject to strong purifying selection with few amino acid replacements allowed, whereas amino acid replacement, even those involving a large Miyata's distance, could occur frequently in functionally unimportant sites. For this reason, it is often difficult to derive mechanical models for protein evolution. Instead, empirical models are used to derive evolutionary distances between homologous proteins.

The general procedure of obtaining evolutionary distances from protein sequences is of four steps. First, obtain empirical substitution counts between closely related sibling species. For example, if one wants to compute evolutionary distances for a protein among primates, then one would compile protein pairs from closely related primate sibling species, perform pairwise alignment, and obtain a 20 by 20 matrix of empirical substitution counts. Second, compute the transition probability matrix from the counts ( $P$  matrix) and the substitution rate matrix ( $Q$  matrix). Third, use the  $Q$  matrix and pairwise empirical count to obtain pairwise evolutionary distances. The next section details these steps.

## 2 Deriving $P$ and $Q$ Matrices for Computing Evolutionary Distances

Suppose we have chosen a set of closely related sibling sequences from HIV-1 and obtained an empirical substitution matrix by checking matched and mismatched amino acids along the aligned sequences (Table 13.2). There are three reasons for choosing closely related species. The first is to minimize the need for correction for multiple substitutions at the same site. The second is to avoid uncertainties concerning indels. The third is to ensure that the  $A$  matrix has large values in the diagonal and small values off-diagonal, which increases the chance that, from the  $A$  matrix, we can derive a  $P$  matrix with its matrix logarithm in real domain and a  $Q$  matrix that has real eigenvalues ( $\lambda$ ) and corresponding eigenvectors ( $U$ ). These mathematical necessities will become clear later.

There are 792 sites at which both sequences are Ala, 3 sites with one sequence being Ala and the other being Arg, and so on (Table 13.2). Because we cannot distinguish between substitutions such as Ala→Arg and Arg→Ala, the two corresponding off-diagonal elements are averaged. This resulting symmetrical matrix is often referred as

**Table 13.2** Empirical substitution matrix A from HIV-1 gag protein sequences of closely related sibling taxa

	Ala	Arg	Asn	Asp	Cys	Gln	Glu	Gly	His	Ile	Leu	Lys	Met	Phe	Pro	Ser	Thr	Trp	Tyr	Val
Ala	792	1.5	5.5	4	0.5	1.5	5.5	9	0.5	4	0.5	6	1	1.5	6	10.5	24	0.5	0.5	12
Arg	1.5	670	6.5	1	1	8.5	6	12.5	2	1	1.5	55.5	2	0.5	2	5.5	7.5	1	0.5	2
Asn	5.5	6.5	645	16.5	0.5	2	6.5	6.5	5.5	1.5	14	2	1	1.5	34.5	20	0.5	1.5	2.5	
Asp	4	1	16.5	426	0.5	0.5	30.5	8.5	2	1	1	5	0.5	1	7	4.5	0.5	1	1.5	
Cys	0.5	1	0.5	0.5	346	0.5	0.5	0.5	0.5	0.5	0.5	0.5	1.5	0.5	1	0.5	1	2	0.5	
Gln	1.5	8.5	2	0.5	0.5	715	6.5	1	6.5	0.5	5	17.5	1	0.5	9.5	2.5	2.5	0.5	1	0.5
Glu	5.5	6	6.5	30.5	0.5	6.5	724	16	1	1.5	18	0.5	0.5	2	3	6.5	0.5	1.5	3.5	
Gly	9	12.5	6.5	8.5	0.5	1	16	865	1	1	0.5	3.5	3	0.5	1	13	3.5	3.5	1.5	
His	0.5	2	5.5	2	0.5	6.5	1	1	255	0.5	2	1.5	0.5	1	1.5	1.5	0.5	7	0.5	
Ile	4	1	5.5	1	0.5	0.5	1	1	0.5	741	31.5	5.5	13	3.5	1	5	17.5	0.5	1.5	
Leu	0.5	1.5	1	0.5	5	1.5	0.5	2	31.5	1031	1	10	8.5	6	4	3.5	3	1	11	
Lys	6	55.5	14	5	0.5	17.5	18	3.5	1.5	5.5	1	695	2.5	0.5	0.5	4.5	7	0.5	0.5	
Met	1	2	2	0.5	0.5	1	0.5	3	0.5	13	10	2.5	344	0.5	0.5	1.5	4	0.5	0.5	
Phe	1.5	0.5	1	0.5	1.5	0.5	0.5	0.5	1	3.5	8.5	0.5	372	0.5	2.5	1	1	9	1	
Pro	6	2	1.5	1	0.5	9.5	2	1	1.5	1	6	0.5	0.5	0.5	636	7.5	5.5	0.5	0.5	
Ser	10.5	5.5	34.5	7	1	2.5	3	13	1.5	5	4	4.5	1.5	2.5	7.5	627	22	1	1	
Thr	24	7.5	20	4.5	0.5	2.5	6.5	3.5	1.5	17.5	3.5	7	4	1	5.5	22	677	0.5	1	
Trp	0.5	1	0.5	0.5	1	0.5	0.5	3.5	0.5	0.5	3	0.5	0.5	1	0.5	1	0.5	367	1.5	
Tyr	0.5	0.5	1.5	1	2	1	1.5	7	1.5	1	0.5	0.5	9	0.5	1	1	1.5	299	1	
Val	12	2	2.5	1.5	0.5	3.5	1	0.5	47	11	1.5	1.5	1	0.5	1	6	0.5	1	645	
$\pi_i$	6.54	5.82	5.75	3.79	2.65	5.78	6.16	7.03	2.16	6.51	8.30	6.21	2.88	3.01	5.05	5.58	6.02	2.84	2.46	

Because we cannot distinguish between substitutions such as Ala→Arg and Arg→Ala, the two corresponding off-diagonal values have been averaged. Amino acid frequencies, shown at the last row, are assumed to be equilibrium frequencies ( $\pi_i$ )

the  $A$  matrix. We also designate  $N_i$  as the sum of row  $i$  in matrix  $A$ ,  $N = \Sigma N_i$  ( $i = \text{Ala}, \text{Arg}, \dots, \text{Val}$ ). Let  $\pi_i$  be the frequencies of amino acid  $i$ ,  $\pi_i = N_i/N$ .

The empirical  $P$  matrix is obtained by dividing every  $A_{ij}$  by  $N_i$ . Note that the resulting  $P$  matrix (Table 13.3) satisfies time reversibility, i.e.,  $\pi_i P_{ij} = \pi_j P_{ji}$ . Such a  $P$  matrix derived from empirical substitution data is sometimes subscripted as  $P_{\text{obs}}$  or  $P_{\text{emp}}$  to distinguish it from a mathematically derived  $P$  matrix based on an explicit substitution model. I will use  $P$  or  $P_{\text{obs}}$  interchangeably when there is no confusion.

Note that

$$\begin{aligned} P &= e^{Qt} = e^{QD} \\ QD &= \log(P) \end{aligned} \quad (13.1)$$

where  $t = D$  because the average substitution rate is scaled to 1 so time is measured by distance. Note that the empirical  $P$  matrix is from all pairwise comparisons between sibling species, so  $D$  represents the “average” distance among all sequence pairs of sibling species that contributed data to the  $A$  matrix. Given Eq. (13.1) the matrix logarithm of  $P$  in Table 13.3 gives us a  $QD$  matrix (Table 13.4).

$D$  can then be obtained as

$$D = - \sum_{i=1}^{20} QD_{i,i}\pi_i = 0.134704653 \quad (13.2)$$

Note that the  $A$  matrix derived from two highly diverged homologous sequences may not have matrix logarithm in real domain, so we will not have matrix  $QD$  in Eq. (13.1) and consequently will not be able to calculate  $D$  according to Eq. (13.2). For this reason we want to have an  $A$  matrix with a strong diagonal which increases the chance for us to obtain a matrix logarithm of the  $P$  matrix.

If we have only two aligned sequences and if the  $A$  matrix in Table 13.2 is in fact obtained from the pairwise comparison between these two aligned sequences, then  $D$  above would be the evolutionary distance between the two sequences, and there is no need to compute a  $Q$  matrix. However, in molecular phylogenetics with  $N$  sequences, we want to have all  $N(N-1)/2$  pairwise distances, and we want to have the same  $Q$  matrix for computing all these distances. So we need to derive the  $Q$  matrix as

$$Q = \frac{QD}{D} \quad (13.3)$$

which is shown in Table 13.5. With this  $Q$  matrix, we can compute  $D$  for any pairs of homologous sequences, assuming that the  $Q$  matrix is indeed applicable for all sequences.

The next step is to obtain eigenvalues and eigenvectors from the  $Q$  matrix. Because our  $A$  matrix is derived from closely related species with diagonal values much greater than the off-diagonal values, a  $Q$  matrix derived from such an  $A$  matrix is almost guaranteed to have real eigenvalues ( $\lambda$ ) and corresponding eigenvectors ( $U$ ). The 20 eigenvalues, sorted in ascending order, are  $-2.00123, -1.83984,$

**Table 13.3** Empirical P matrix ( $P_{\text{obs}}$ ) derived from A matrix in Table 13.2 (rounded to three decimal points)

	Ala	Arg	Asn	Asp	Cys	Gln	Glu	Gly	His	Ile	Leu	Lys	Met	Phe	Pro	Ser	Thr	Tyr	Trp	Val
Ala	0.893	0.002	0.006	0.005	0.001	0.002	0.006	0.010	0.001	0.005	0.001	0.007	0.002	0.007	0.012	0.027	0.001	0.001	0.014	
Arg	0.002	0.850	0.008	0.001	0.001	0.011	0.008	0.016	0.003	0.001	0.002	0.070	0.003	0.001	0.007	0.010	0.001	0.001	0.003	
Asn	0.007	0.008	0.828	0.021	0.001	0.003	0.008	0.008	0.007	0.007	0.002	0.018	0.003	0.001	0.002	0.044	0.026	0.001	0.002	0.003
Asp	0.008	0.002	0.032	0.830	0.001	0.001	0.059	0.017	0.004	0.002	0.002	0.010	0.001	0.001	0.002	0.014	0.009	0.001	0.002	0.003
Cys	0.001	0.003	0.001	0.001	0.962	0.001	0.001	0.001	0.001	0.001	0.001	0.004	0.001	0.001	0.003	0.001	0.003	0.006	0.001	
Gln	0.002	0.011	0.003	0.001	0.001	0.913	0.008	0.001	0.008	0.001	0.006	0.022	0.001	0.001	0.012	0.003	0.003	0.001	0.001	
Glu	0.007	0.007	0.008	0.037	0.001	0.008	0.867	0.019	0.001	0.001	0.002	0.022	0.001	0.001	0.002	0.004	0.008	0.001	0.002	0.004
Gly	0.009	0.013	0.007	0.009	0.001	0.001	0.017	0.909	0.001	0.001	0.004	0.003	0.001	0.001	0.014	0.004	0.004	0.002	0.001	
His	0.002	0.007	0.019	0.007	0.002	0.022	0.003	0.003	0.873	0.002	0.007	0.005	0.002	0.003	0.005	0.005	0.005	0.002	0.024	0.002
Ile	0.005	0.001	0.006	0.001	0.001	0.001	0.001	0.001	0.001	0.840	0.036	0.006	0.015	0.004	0.001	0.006	0.020	0.001	0.002	0.053
Leu	0.000	0.001	0.001	0.000	0.004	0.001	0.000	0.000	0.002	0.028	0.917	0.001	0.009	0.008	0.005	0.004	0.003	0.003	0.001	0.010
Lys	0.007	0.006	0.017	0.006	0.001	0.021	0.021	0.004	0.002	0.007	0.001	0.827	0.003	0.001	0.005	0.008	0.001	0.001	0.001	0.002
Met	0.003	0.005	0.005	0.001	0.001	0.003	0.001	0.008	0.001	0.033	0.026	0.006	0.883	0.001	0.001	0.004	0.010	0.001	0.001	0.004
Phe	0.004	0.001	0.002	0.001	0.004	0.001	0.001	0.001	0.002	0.009	0.021	0.001	0.001	0.913	0.001	0.006	0.002	0.002	0.022	0.002
Pro	0.009	0.003	0.002	0.001	0.001	0.014	0.003	0.001	0.002	0.001	0.009	0.001	0.001	0.001	0.930	0.011	0.008	0.001	0.001	0.001
Ser	0.014	0.007	0.046	0.009	0.001	0.003	0.004	0.017	0.002	0.007	0.005	0.006	0.002	0.003	0.010	0.830	0.029	0.001	0.001	0.001
Thr	0.029	0.009	0.025	0.006	0.001	0.003	0.008	0.004	0.002	0.021	0.004	0.009	0.005	0.001	0.007	0.027	0.830	0.001	0.001	0.007
Trp	0.001	0.003	0.001	0.001	0.003	0.001	0.001	0.009	0.001	0.001	0.008	0.001	0.001	0.003	0.001	0.953	0.004	0.001	0.001	
Tyr	0.002	0.002	0.005	0.003	0.006	0.003	0.005	0.005	0.021	0.005	0.003	0.002	0.002	0.027	0.002	0.003	0.003	0.005	0.898	0.003
Val	0.016	0.003	0.003	0.002	0.001	0.001	0.005	0.001	0.001	0.064	0.015	0.002	0.002	0.001	0.001	0.008	0.001	0.001	0.001	0.872

$P_{\text{obs}}$  satisfies time reversibility with  $\pi_i P_{ij} = \pi_j P_{ji}$

**Table 13.4**  $QD$  matrix resulting from the matrix logarithm of  $P_{\text{obs}}$  in Table 13.3, with diagonal elements rounded to two decimal points and the off-diagonal elements to three decimal points

	Ala	Arg	Asn	Asp	Cys	Gln	Glu	Gly	His	Ile	Leu	Lys	Met	Phe	Pro	Ser	Thr	Trp	Tyr	Val
Ala	-0.11	0.001	0.007	0.008	0.001	0.002	0.007	0.010	0.002	0.004	0.000	0.008	0.002	0.004	0.009	0.015	0.034	0.001	0.018	
Arg	0.001	-0.17	0.009	0.001	0.003	0.011	0.007	0.015	0.007	0.001	0.001	0.078	0.005	0.001	0.003	0.008	0.010	0.003	0.001	0.003
Asn	0.006	0.008	-0.19	0.038	0.001	0.002	0.008	0.007	0.022	0.007	0.001	0.019	0.005	0.002	0.002	0.054	0.028	0.001	0.005	0.003
Asp	0.005	0.001	0.025	-0.19	0.001	0.000	0.043	0.010	0.008	0.001	0.001	0.006	0.001	0.001	0.001	0.010	0.006	0.001	0.003	0.002
Cys	0.001	0.001	0.001	-0.04	0.001	0.001	0.001	0.002	0.001	0.000	0.001	0.004	0.001	0.001	0.001	0.001	0.003	0.003	0.006	0.001
Gln	0.002	0.011	0.002	0.001	0.001	-0.09	0.008	0.001	0.025	0.000	0.005	0.023	0.003	0.001	0.015	0.003	0.003	0.001	0.003	0.001
Glu	0.006	0.007	0.008	0.070	0.001	0.009	-0.15	0.018	0.003	0.001	0.024	0.001	0.001	0.003	0.004	0.009	0.001	0.005	0.005	0.005
Gly	0.011	0.018	0.009	0.018	0.001	0.001	0.021	-0.10	0.003	0.001	0.000	0.004	0.008	0.001	0.001	0.019	0.004	0.010	0.005	0.001
His	0.001	0.003	0.008	0.004	0.001	0.009	0.001	0.001	-0.14	0.000	0.002	0.002	0.001	0.002	0.002	0.002	0.001	0.024	0.001	0.001
Ile	0.004	0.001	0.008	0.002	0.001	0.000	0.001	0.001	-0.18	0.031	0.007	0.038	0.009	0.001	0.007	0.025	0.001	0.005	0.074	
Leu	0.000	0.002	0.002	0.002	0.001	0.007	0.002	0.000	0.007	0.040	-0.09	0.001	0.028	0.022	0.009	0.006	0.004	0.008	0.003	0.015
Lys	0.007	0.084	0.021	0.010	0.001	0.025	0.025	0.003	0.005	0.007	0.001	-0.19	0.007	0.001	0.000	0.006	0.009	0.001	0.001	0.002
Met	0.001	0.003	0.003	0.001	0.001	0.001	0.003	0.002	0.017	0.010	0.003	-0.13	0.001	0.001	0.002	0.005	0.001	0.002	0.002	
Phe	0.002	0.001	0.001	0.004	0.001	0.001	0.000	0.003	0.004	0.008	0.001	0.001	-0.09	0.001	0.004	0.001	0.003	0.030	0.001	
Pro	0.007	0.003	0.002	0.002	0.001	0.013	0.002	0.001	0.005	0.001	0.006	0.000	0.001	0.001	-0.07	0.011	0.007	0.001	0.001	0.001
Ser	0.013	0.007	0.053	0.015	0.003	0.003	0.015	0.005	0.006	0.004	0.005	0.004	0.004	0.007	0.012	-0.19	0.031	0.003	0.003	0.001
Thr	0.031	0.011	0.030	0.009	0.001	0.003	0.008	0.003	0.005	0.023	0.003	0.009	0.011	0.002	0.009	0.034	-0.19	0.001	0.003	0.008
Trp	0.001	0.001	0.001	0.001	0.003	0.001	0.001	0.004	0.002	0.001	0.003	0.001	0.001	0.003	0.001	0.001	0.001	-0.05	0.005	0.001
Tyr	0.001	0.001	0.002	0.006	0.001	0.002	0.002	0.002	0.027	0.002	0.001	0.001	0.024	0.001	0.001	0.001	0.004	-0.11	0.001	
Val	0.015	0.003	0.003	0.003	0.001	0.005	0.001	0.002	0.062	0.010	0.002	0.003	0.002	0.001	0.001	0.007	0.001	0.003	-0.14	

**Table 13.5** Instantaneous rate matrix ( $\mathcal{Q}$ ), computed according to Eq. (13.3), rounded to save space

	Ala	Arg	Asn	Asp	Cys	Gln	Glu	Gly	His	Ile	Leu	Lys	Met	Phe	Pro	Ser	Thr	Trp	Tyr	Val
Ala	-0.84	0.010	0.052	0.061	0.010	0.013	0.051	0.075	0.011	0.031	0.001	0.058	0.018	0.029	0.069	0.112	0.249	0.009	0.010	0.134
Arg	0.009	-1.23	0.064	0.009	0.022	0.083	0.052	0.108	0.055	0.005	0.010	0.583	0.040	0.009	0.023	0.058	0.075	0.020	0.010	0.022
Asn	0.045	0.063	-1.42	0.280	0.010	0.017	0.058	0.052	0.160	0.049	0.008	0.142	0.040	0.018	0.014	0.402	0.209	0.009	0.034	0.025
Asp	0.035	0.006	0.184	-1.40	0.011	0.002	0.317	0.072	0.056	0.007	0.007	0.046	0.009	0.009	0.011	0.075	0.043	0.010	0.023	0.015
Cys	0.004	0.010	0.005	0.008	-0.28	0.005	0.004	0.004	0.013	0.004	0.003	0.004	0.010	0.028	0.005	0.010	0.004	0.020	0.047	0.005
Gln	0.012	0.083	0.017	0.004	0.010	-0.68	0.062	0.006	0.183	0.002	0.035	0.173	0.019	0.009	0.111	0.025	0.023	0.010	0.022	0.004
Glu	0.048	0.055	0.062	0.516	0.010	0.066	-1.08	0.136	0.024	0.006	0.010	0.182	0.008	0.009	0.022	0.026	0.064	0.009	0.036	0.038
Gly	0.081	0.131	0.063	0.133	0.010	0.008	0.156	-0.72	0.025	0.007	0.002	0.026	0.062	0.008	0.010	0.142	0.030	0.072	0.035	0.009
His	0.004	0.020	0.060	0.032	0.010	0.068	0.008	0.008	-1.01	0.004	0.014	0.013	0.010	0.018	0.017	0.015	0.014	0.010	0.175	0.005
Ile	0.031	0.006	0.056	0.013	0.010	0.003	0.006	0.007	0.011	-1.32	0.232	0.055	0.282	0.068	0.009	0.052	0.185	0.009	0.035	0.549
Leu	0.002	0.014	0.012	0.014	0.010	0.050	0.013	0.003	0.054	0.296	-0.65	0.007	0.205	0.167	0.069	0.042	0.030	0.061	0.020	0.112
Lys	0.055	0.622	0.154	0.076	0.010	0.186	0.183	0.023	0.037	0.052	0.005	-1.44	0.051	0.009	0.002	0.043	0.067	0.009	0.010	0.013
Met	0.008	0.020	0.020	0.007	0.011	0.010	0.004	0.025	0.013	0.125	0.071	0.024	-0.93	0.008	0.005	0.015	0.040	0.010	0.011	0.012
Phe	0.013	0.005	0.009	0.007	0.032	0.004	0.004	0.004	0.025	0.032	0.061	0.004	0.009	-0.68	0.005	0.027	0.009	0.020	0.221	0.009
Pro	0.053	0.020	0.012	0.014	0.010	0.097	0.018	0.007	0.040	0.007	0.042	0.002	0.009	-0.54	0.082	0.054	0.010	0.011	0.004	0.004
Ser	0.096	0.055	0.390	0.111	0.022	0.024	0.024	0.113	0.038	0.045	0.028	0.039	0.028	0.050	0.090	-1.40	0.232	0.020	0.022	0.007
Thr	0.229	0.078	0.219	0.068	0.010	0.024	0.063	0.026	0.039	0.171	0.022	0.065	0.083	0.018	0.064	0.250	-1.40	0.009	0.023	0.061
Trp	0.004	0.010	0.004	0.007	0.021	0.005	0.004	0.029	0.013	0.004	0.021	0.004	0.010	0.019	0.005	0.010	0.004	-0.36	0.035	0.005
Tyr	0.004	0.015	0.015	0.044	0.009	0.014	0.012	0.200	0.013	0.006	0.004	0.010	0.181	0.005	0.010	0.009	0.031	-0.81	0.010	0.010
Val	0.112	0.020	0.024	0.022	0.011	0.004	0.034	0.007	0.013	0.460	0.074	0.011	0.022	0.017	0.004	0.007	0.055	0.010	0.023	-1.04

$-1.78497, -1.69732, -1.56214, -1.17191, -1.05933, -0.99382, -0.94424, -0.92587, -0.89523, -0.85784, -0.76272, -0.63056, -0.54285, -0.51662, -0.39716, -0.35996, -0.27922$ , and  $-0.00015$ , with the associated eigenvector  $\mathbf{U}$  in Table 13.6.

Now the transition probability matrix for a given distance  $D$  (designated  $P_D$ ) is

$$P_D = \mathbf{U} \Lambda \mathbf{U}^{-1} \quad (13.4)$$

where  $\Lambda$  is a diagonal matrix with values being  $e^{D\lambda_1}, e^{D\lambda_2}, \dots, e^{D\lambda_{20}}$ ,  $\mathbf{U}$  the eigenvectors, and  $\mathbf{U}^{-1}$  the matrix inverse of  $\mathbf{U}$ .

To estimate the evolutionary distance between two aligned sequences, we will first compile an  $A$  matrix similar to that in Table 13.2 by comparing the two aligned sequences. Table 13.7 displays the result of such a compilation between two aligned HIV-1 gag protein sequences.

The best  $D$  is one that maximizes the log-likelihood ( $\ln L$ ):

$$\ln L = A_{ij} \ln (\pi_i P_{D,ij}) \quad (13.5)$$

where  $A_{ij}$  is in Table 13.7,  $\pi_i$  is the equilibrium frequencies typically approximated by the empirical amino acid frequencies, and matrix  $P_D$  is specified in Eq. (13.4). The maximum  $\ln L$  is  $-3450$ , achieved when  $D = 0.2679$ . Thus, to estimate  $D$  for a pair of sequences, we do not need to change  $\mathbf{U}$  or  $\mathbf{U}^{-1}$  in Eq. (13.4). We only need to replace  $D$  in the diagonal matrix  $\Lambda$  and find the best  $D$  that maximizes  $\ln L$  in Eq. (13.5). The final  $P_D = 0.2679$  is shown in Table 13.8.

In short, we have a dilemma in computing evolutionary distances between aligned protein sequences. In order to obtain a  $P$  matrix with a matrix logarithm in real domain and a  $Q$  matrix with real eigenvalues ( $\lambda$ ), we need to have an  $A$  matrix with large diagonal values and small off-diagonal values. So we would compile the  $A$  matrix from closely related sibling species. However, such an  $A$  matrix and the derived  $P$  and  $Q$  matrices may not be appropriate when we use the  $Q$  matrix to compute distances for highly diverged sequences.

**Table 13.6** Eigenvectors ( $\mathbf{U}$ ) from the instantaneous rate matrix ( $\mathcal{Q}$ ), corresponding to the 20 eigenvalues sorted in ascending order, multiplied by 10 and rounded to save space

Ala	Arg	Asn	Asp	Cys	Gln	Glu	Gly	His	Ile	Lys	Leu	Met	Phe	Pro	Ser	Thr	Trp	Tyr	Val
-0.59	-0.01	1.226	0.047	1.78	-2.16	-5.11	1.649	-3.63	-1.57	3.607	2.189	4.582	-3.54	1.805	-1.29	-0.37	-1.03	1.211	-2.76
5.44	-2.52	-1.54	-0.4	0.259	-1.03	-1.14	3.85	0.096	4.335	-2.32	3.261	-0.16	2.639	-0.51	-0.56	-2.01	-1.15	0.934	-2.46
-2.74	6.566	1.11	-0.66	3.151	2.242	4.223	-0.59	-1.85	-0.98	-2.29	-0.16	0.747	-0.78	-0.24	-1.07	-0.98	-0.88	0.878	-2.43
0.848	-2.93	-1.1	-5.61	2.556	-0.15	-0.35	-1.93	1.15	-2.82	-1.8	-0.5	-0.63	-0.6	-0.41	-1.14	-1.14	-0.64	0.598	-1.6
0.012	0.033	0.012	-0.01	-0.02	-0.1	0.046	0.015	0	-0.11	0.065	0.113	-0.09	0.157	1.355	-0.2	-0.2	-2.31	-9.08	-1.11
-0.58	-0.19	-0.15	-0.33	0.159	2.516	0.793	-2.34	0.09	-2	4.65	-0.22	1.389	6.011	-0.98	4.104	-1.8	-1.4	0.998	-2.44
-1.53	1.081	0.782	4.878	-2.66	-1.22	-3.68	-3.06	3.651	-4.37	-2.45	-0.12	-1.5	0.012	-0.7	-1.68	-2.17	-1.18	1.102	-2.6
0.493	0.758	0.069	-0.62	-0.37	-0.2	1.18	1.529	0.675	2.267	5.802	-2.66	-5.76	-2.32	-0.54	-3.33	-3.03	-0.23	1.085	-2.97
0.166	-0.27	0.051	0.252	-0.41	-6.41	1.367	0.83	-0.98	-0.37	-0.51	-3.79	0.729	0.456	-1.96	0.676	-0.16	-0.09	-0.07	-0.91
-1.38	-1.65	7.262	-2.67	-2.59	0.425	1.399	-0.73	2.671	1.431	0.171	-1.2	1.234	0.534	2.2	-1.3	3.624	0.33	0.792	-2.75
0.225	0.458	-1.25	0.374	0.521	0.799	-1.52	4.726	-3.18	-4.25	-0.7	0.262	-4.8	2.67	2.058	0.753	6.399	1.879	0.594	-3.51
7.308	2.187	1.89	-0.1	0.471	-0.49	-1.38	2.231	0.335	2.549	-2.36	2.983	0.421	2.792	-0.52	-0.47	-1.93	-1.31	1.1	-2.62
0.07	0.092	-0.77	0.47	0.709	-0.77	-1.69	-5.36	-2.6	3.237	-0.54	-1.28	-0.85	0.762	0.961	-0.54	1.338	0.166	0.287	-1.22
0.002	0.104	-0.1	-0.04	-0.02	-1.87	1.019	-1.56	0.643	0.115	0.909	5.572	-0.48	-1.35	-5.85	0.647	1.902	0.834	-0.99	-1.27
0.034	0.321	0.123	-0.23	0.155	-0.48	-0.37	-0.01	1.403	1.299	-1.05	-0.12	-1.36	-3.94	3.527	7.561	-0.52	-1.1	0.943	-2.13
1.466	-5.57	0.632	5.155	1.307	3.434	4.287	-0.21	-2.38	-0.59	-0.99	0.319	0.463	-1.51	0.241	-0.61	-0.76	-0.72	0.764	-2.36
0.872	0.755	-4.01	-2.4	-7.57	2.096	1.726	-0.29	-1.97	-0.19	-0.63	0.627	1.967	-1.29	0.982	-0.84	-0.06	-0.78	0.979	-2.54
0.002	0.02	0.027	-0.01	-0.02	-0.08	0.031	-0.19	0.083	-0.08	-0.19	0.257	0.506	0.148	1.119	0.274	-1.7	8.886	-1.83	-1.19
0	0.014	-0.02	-0.03	0.113	4.257	-2.04	1.118	0.207	0.926	-0.62	-4.12	0.851	-0.83	-4.9	0.602	0.58	0.531	-1.01	-1.04
0.779	0.767	-4.25	1.923	2.483	-0.8	1.208	0.322	5.587	1.185	1.227	-1.46	2.74	-0.02	2.386	-1.56	3.007	0.14	0.748	-2.31

**Table 13.7** Empirical substitution matrix  $A$  from comparing two aligned HIV-1 gag protein sequences (B.TH.90.BK132.AY173951 and D.CM.01.01CM\_4412HAL.AY371157)

	Ala	Arg	Asn	Asp	Cys	Gln	Glu	Gly	His	Ile	Leu	Lys	Met	Phe	Pro	Ser	Thr	Trp	Tyr	Val
Ala	39	0.5	0	0	0	1	0	0	1.5	0	0	0	0.5	1	5	0	0	0	39	
Arg	0.5	37	0.5	0	0	2.5	0.5	1.5	0	0	5.5	1	0	0.5	0	0	0	0	0.5	
Asn	0	0.5	43	2.5	0	0	1.5	2	1	0	0	1.5	0	0	0	3.5	2.5	0.5	0	
Asp	0	0	2.5	21	0	0.5	2.5	0.5	0	0	0	0.5	0	0	0	0.5	0	0	0	
Cys	0	0	0	0	21	0	0	0	0	0	0	0	0	0	0	0	0.5	0	0	
Gln	0	2.5	0	0.5	0	31	2	1	0	0	0.5	2.5	0	0	0	0.5	0	0	0	
Glu	1	0.5	1.5	2.5	0	2	32	1	0	0.5	0	2	0	0	0	1	0	0	1	
Gly	0	1.5	2	0.5	0	1	1	47	0	0.5	0	0	0	0.5	0.5	1	2.5	0	0	
His	0	0	1	0	0	0	0	0	10	0	1.5	0	0	0	0	0	0	0	0	
Ile	1.5	0	0	0	0	0.5	0.5	0	54	2.5	1	2	0	0	0.5	3	0	0	1.5	
Leu	0	0	0	0	0	0.5	0	0	1.5	2.5	76	0.5	0.5	2	0	1	0	0.5	0	
Lys	0	5.5	1.5	0.5	0	2.5	2	0	0	1	0.5	23	0.5	0	0	1	2	0	0	
Met	0	1	0	0	0	0	0	0	2	0.5	0.5	12	0	0	0	1	0	0	0	
Phe	0	0	0	0	0	0	0.5	0	0	2	0	0	20	0	0.5	0	0	0.5	0	
Pro	0.5	0.5	0	0	0	0	0.5	0	0	0	0	0	0	29	0	0	0	0	0.5	
Ser	1	0	3.5	0.5	0	0.5	1	1	0	0.5	1	1	0	0.5	0	37	3.5	0	0.5	
Thr	5	0	2.5	0	0.5	0	0	2.5	0	3	0	2	1	0	0	3.5	37	0.5	5	
Trp	0	0	0.5	0	0	0	0	0	0.5	0	0	0	0	0	0.5	27	0	0	0	
Tyr	0	0	0.5	0	0	0	0	0	0	0	0.5	0	0	0.5	0	0	18	0	0	
Val	2.5	0	1.5	0.5	0	0	0.5	0	0	4.5	0.5	0	1	0	0	1	0	0	2.5	

Because we cannot distinguish between substitutions such as Ala→Arg and Arg→Ala, the two corresponding off-diagonal values have been averaged

**Table 13.8** Transition probability matrix  $P$  derived from Eq. (13.4) with  $D = 0.2679$

	Ala	Arg	Asn	Asp	Cys	Gln	Glu	Gly	His	Ile	Leu	Lys	Met	Pho	Pro	Ser	Thr	Tyr	Tip	Tyr	Val
Ala	0.801	0.004	0.014	0.015	0.003	0.004	0.013	0.018	0.004	0.01	0.001	0.013	0.005	0.007	0.016	0.025	0.052	0.003	0.003	0.029	
Arg	0.004	0.73	0.016	0.005	0.005	0.021	0.014	0.024	0.013	0.003	0.111	0.01	0.003	0.006	0.014	0.017	0.005	0.005	0.003	0.005	
Asn	0.012	0.016	0.691	0.055	0.003	0.005	0.015	0.013	0.033	0.012	0.003	0.029	0.01	0.005	0.005	0.077	0.043	0.003	0.009	0.007	
Asp	0.008	0.003	0.036	0.694	0.003	0.002	0.062	0.017	0.012	0.002	0.011	0.003	0.002	0.003	0.017	0.011	0.003	0.006	0.004	0.004	
Cys	0.001	0.002	0.001	0.002	0.927	0.001	0.001	0.003	0.001	0.001	0.001	0.002	0.007	0.001	0.002	0.001	0.005	0.011	0.001	0.001	
Gln	0.004	0.021	0.005	0.003	0.003	0.836	0.015	0.002	0.04	0.001	0.008	0.037	0.005	0.003	0.026	0.006	0.006	0.003	0.006	0.001	
Glu	0.012	0.015	0.016	0.101	0.003	0.015	0.756	0.031	0.007	0.003	0.003	0.038	0.003	0.003	0.006	0.008	0.015	0.003	0.009	0.009	
Gly	0.019	0.029	0.016	0.031	0.003	0.003	0.035	0.828	0.007	0.003	0.001	0.009	0.014	0.003	0.003	0.031	0.009	0.017	0.009	0.003	
His	0.001	0.005	0.012	0.007	0.003	0.015	0.002	0.002	0.765	0.001	0.003	0.004	0.005	0.004	0.004	0.004	0.004	0.003	0.037	0.001	
Ile	0.001	0.003	0.013	0.004	0.003	0.002	0.003	0.002	0.004	0.713	0.05	0.012	0.059	0.016	0.003	0.012	0.037	0.003	0.009	0.109	
Leu	0.002	0.004	0.004	0.004	0.003	0.012	0.004	0.001	0.013	0.064	0.843	0.003	0.047	0.039	0.016	0.01	0.009	0.015	0.007	0.029	
Lys	0.012	0.119	0.031	0.018	0.003	0.04	0.038	0.008	0.01	0.011	0.002	0.691	0.012	0.003	0.002	0.012	0.016	0.003	0.003	0.004	
Met	0.002	0.005	0.005	0.002	0.003	0.003	0.001	0.006	0.003	0.026	0.016	0.006	0.782	0.003	0.002	0.004	0.009	0.003	0.003	0.005	
Phe	0.003	0.001	0.003	0.002	0.008	0.001	0.001	0.001	0.007	0.014	0.001	0.003	0.835	0.001	0.006	0.003	0.005	0.005	0.049	0.003	
Pro	0.013	0.005	0.004	0.004	0.003	0.022	0.005	0.002	0.01	0.002	0.001	0.003	0.003	0.866	0.018	0.013	0.003	0.003	0.002	0.002	
Ser	0.022	0.013	0.075	0.025	0.005	0.006	0.008	0.025	0.01	0.011	0.007	0.011	0.008	0.011	0.02	0.694	0.047	0.005	0.006	0.003	
Thr	0.047	0.017	0.045	0.017	0.003	0.006	0.014	0.008	0.01	0.034	0.006	0.016	0.019	0.005	0.015	0.05	0.694	0.003	0.006	0.016	
Tip	0.001	0.002	0.001	0.002	0.005	0.001	0.001	0.007	0.003	0.001	0.005	0.001	0.003	0.005	0.001	0.002	0.001	0.909	0.008	0.001	
Tyr	0.001	0.001	0.004	0.004	0.011	0.003	0.003	0.003	0.042	0.003	0.002	0.001	0.003	0.04	0.001	0.003	0.002	0.007	0.808	0.003	
Val	0.024	0.005	0.006	0.006	0.003	0.001	0.008	0.002	0.004	0.091	0.019	0.004	0.009	0.005	0.002	0.003	0.014	0.003	0.006	0.765	

# Chapter 14

## Maximum Parsimony Method in Phylogenetics



### 1 Introduction

The maximum parsimony (MP) method uses a parsimonious criterion to choose the best tree, i.e., the tree that can account for the sequence variation with the smallest number of substitutions or the smallest substitution cost is the best tree. The method became immensely popular due to the excellent program PAUP (Swofford 1993). MP does not specify an explicit substitution model or make any explicit effort in correcting for multiple hits. For this reason, it has the inherent assumption of a slow rate of evolution (so slow that multiple hits are negligible). The Fitch algorithm (Fitch 1971) further assumes that all nucleotides or amino acids replace each other with equal frequencies. The Sankoff algorithm (Sankoff 1975) makes use of a step matrix to accommodate different substitution rates among different nucleotides or among different amino acids. The step matrix is sometimes also called a cost matrix if we measure the cost as the number of steps (of nucleotide or amino acid changes) that have been taken place during the evolution along the tree.

The step matrix has dimension  $4 \times 4$  for nucleotide sequences and  $20 \times 20$  for amino acid sequences. Its diagonal elements are zero and off-diagonal elements increases with the rarity of substitutions between nucleotides and amino acids. If all off-diagonal elements of a step matrix are 1, it means that all 4 nucleotides or all 20 amino acids replace each other with the same frequencies. However, transitions typically occur more frequently than transversions during the evolution of nucleotide sequences and are consequently given a lower value in the step matrix than transversions. A step matrix represents an attempt of the MP method to approximate a substitution model used in model-based phylogenetic methods such as the distance-based, maximum likelihood, or Bayesian methods.

MP remains the best entry point for computer science students to learn molecular phylogenetics and for biology students to gain a foothold in computational biology. The method was, and still is, highly popular among phylogeneticists working on plants but much less so among those working on animals. There is a good reason

for this difference. Historically, most phylogenies were reconstructed with mitochondrial sequences. Animal mitochondrial genomes evolve very rapidly (Xia 2012c) and violate a key assumption of MP (i.e., slow evolution rate). In contrast, plant mitochondrial genomes evolve extraordinarily slowly, even slower than plant nuclear genomes (Drouin et al. 2008; Wolfe et al. 1987). Thus, MP is bad for animal mitochondrial sequences, but is not bad for plant mitochondrial sequences. MP is particularly powerful in cladistics analysis of morphological, physiological, embryological, or paleontological characters (Brooks and McLennan 1991) whose changes are slow because of functional constraints and whose rates of change cannot be rendered into a substitution model.

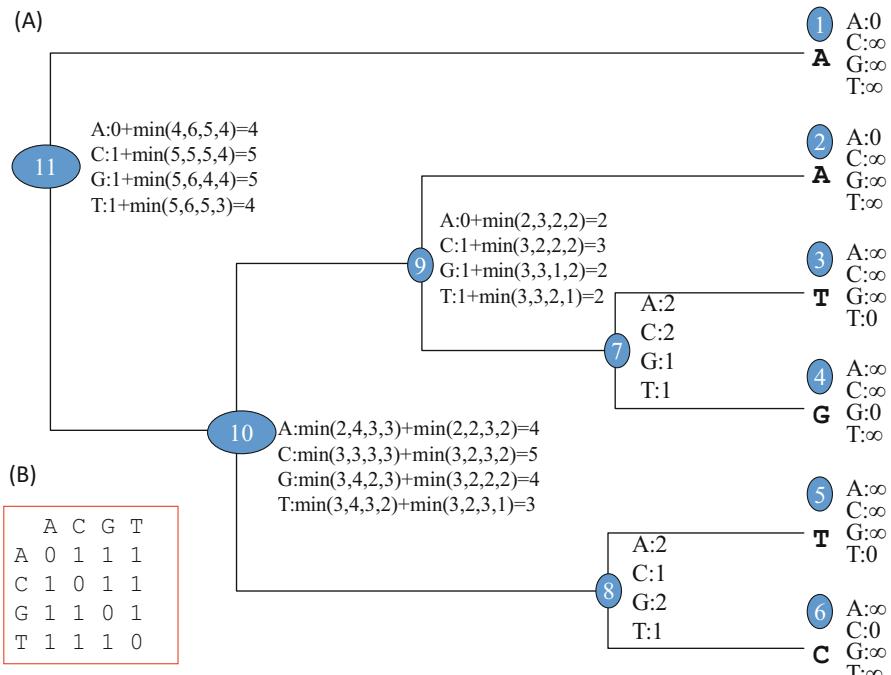
Because the Fitch algorithm has already been numerically illustrated in a previous chapter on sequence alignment, this chapter focuses on the Sankoff algorithm which, through the step matrix, is advantageous over the Fitch algorithm in two ways. First, it uses more information. For example, in a four-taxon case with a site having A, C, G, and T for the four taxa, respectively, the site is not informative for the Fitch algorithm, but is informative if transitions ( $A \leftrightarrow G$  or  $C \leftrightarrow T$ ) occur at a different rate from transversions, i.e.,  $\{A, G\} \leftrightarrow \{C, T\}$ . Take, for example, four sequences (S1 to S4) with nucleotides A, G, C, and T, respectively, at site  $i$ . If transitions occur much more frequently than transversions, then the former will have a lower substitution cost in the step matrix, and site  $i$  will favor the grouping of {S1, S2} and {S3, S4} with the Sankoff algorithm, but such a site pattern is not informative with the Fitch algorithm. Second, the step matrix alleviates the long-branch attraction problem (Felsenstein 1978a) because frequent substitutions can be assigned a smaller cost than rare substitutions. Impossible changes will have substitute cost equal to  $\infty$ .

We will first give a few numerical illustrations of the Sankoff algorithm, followed by a brief introduction on the branch-and-bound algorithm which is guaranteed to find the best solution without exhaustive searching. Exhaustive searching and branch-and-bound algorithm are possible with MP because MP is fast, especially in comparison with computation-intensive maximum likelihood or Bayesian method. Of course, given the rapid increase of possible topologies with the number of OTUs (operational taxonomic units which could be a group of related species or a group of homologous sequences or any phylogenetically related entities positioned as leaves on a phylogenetic tree), the exhaustive and the branch-and-bound approaches can be applied to only a small number (<20) of OTUs even with MP. This is followed by an illustration of the long-branch attraction problem which has plagued MP when the method is misapplied to molecular sequences that have experienced a high degree of substitution saturation. An advocate of the MP method can argue that it is unfair to criticize MP with the long-branch attraction problem because MP does not correct for multiple hits and therefore is not supposed to be used where long branches (which implies substitution saturation) exist. However, phylogeneticists do frequently run MP on highly diverged sequences, and it is only fair to highlight the long-branch attraction problem. The last section of the chapter details MP-related statistical tests (both parametric and nonparametric) for alternative topologies. These tests are implemented in DAMBE (Xia 2013, 2017d).

## 2 The Sankoff Algorithm

The Sankoff algorithm will take as input (1) a step matrix ( $C$ , whose elements are  $c_{ij}$  with  $i$  and  $j$  taking the value of A, C, G, and T for nucleotide sequences or the 20 amino acids for the amino acid sequences), (2) a topology, and (3) a multiple sequence alignment (Fig. 14.1, which show only a single site of the alignment) and will evaluate the substitution cost (number of steps) of the topology. After all alternative topologies have been evaluated, the MP criterion picks the tree with the smallest substitution cost as the MP tree. In reality, most published MP trees are not real MP trees because they are typically the result of a heuristic search which does not guarantee the final tree is in fact the MP tree.

The step matrix ( $C$ ) in Fig. 14.1b has  $c_{ij} = 1$  (which means that a nucleotide substitution is counted as one step) and  $c_{ii} = 0$ . For such a  $C$  matrix, the results from the Sankoff algorithm are exactly the same as the Fitch algorithm. For this reason, Sankoff algorithm, which is slower than the Fitch algorithm, should be used only when we give  $c_{ij}$  different values, e.g., when a transition such as A $\leftrightarrow$ G is given  $c_A, G = 1$  and a transversion such as A $\leftrightarrow$ T is given  $c_{A,T} = 2$  (which means that a transversion is counted as two steps and a transition as one step). For amino acid



**Fig. 14.1** Computing the minimum number of changes for one site of a multiple sequence alignment using the Sankoff algorithm (a) with step matrix  $C$  in (b). All nodes are numbered sequentially, with a four-element step vector (S) for each node

sequences, a Trp $\leftrightarrow$ Gly substitution is very rare and should have a large  $c_{\text{Trp}, \text{Gly}}$ , whereas a Asp $\leftrightarrow$ Glu substitution is very common, and  $c_{\text{Asp}, \text{Glu}}$  should be small. We will first illustrate the Sankoff algorithm with the  $C$  matrix in Fig. 14.1b and then with a  $C$  matrix of unequal  $c_{ij}$  ( $i \neq j$ ).

The Sankoff algorithm, just as the Fitch algorithm, is site-based, i.e., the evaluation is done to obtain the minimum number of steps (of changes) for each site. For the Fitch algorithm, the minimum number of steps at site  $j$  (designated by  $L_j$ ) is the minimum number of substitutions needed to account for the nucleotide (or amino acid) variation at site  $j$ . After evaluating all sites, the total number of steps for the tree, often designated as tree length (TL), is

$$\text{TL} = \sum_{j=1}^N L_j \quad (14.1)$$

where  $N$  is the number of sites in the multiple alignment. These designations are also used for the Sankoff algorithm, although here a step is typically not equivalent to a nucleotide or amino acid substitution unless the step matrix is the same as that in Fig. 14.1b.

In contrast to the Fitch algorithm where each node can be represented by a single character, the Sankoff algorithm represents each node (either a leaf node or an internal node) by a cost vector ( $S$ ).  $S$  would have 4 elements for nucleotide sequences or 20 for amino acid sequences. For simplicity, I will illustrate the algorithm with nucleotide sequences with the four elements in  $S$  in the order of A, C, G, and T (Fig. 14.1). For terminal nodes (leaves) with character  $i$ , the element corresponding to character  $i$  is assigned a cost of 0, and other elements are assigned  $\infty$  (Fig. 14.1). This means, in a probabilistic sense, that the probability of the character remaining the same is 1 in a time interval of 0 (i.e., there should be no cost/penalty). Similarly, the probability of a nucleotide becoming a different one is 0 (impossible) when there is no time to change and should be given the maximum cost. In short, higher costs are associated with less likely substitutions.

For an internal node (say  $a$ , where  $a = 7, 8, \dots$ , or 11 in Fig. 14.1) with two daughter nodes, each element in the cost vector is the summation of two parts, one from each of the daughter nodes, according to the following equation:

$$S_a(i) = \min_j [c_{ij} + S_l(j)] + \min_k [c_{ik} + S_r(k)] \quad (14.2)$$

where  $c_{ij}$  is the cost between characters  $i$  and  $j$  specified in Fig. 14.1b and the subscripted  $l$  and  $r$  represent the two daughter nodes of node  $a$ , following my own programming convention of representing two daughter nodes as left ( $l$ ) and right ( $r$ ) nodes. According to Eq. (14.2), to get  $S_7(\text{A})$  with node 7 (which has daughter nodes 3 and 4 with characters “T” and “G,” respectively, Fig. 14.1), we first compute four values based on  $S_3$ :

$$\begin{aligned}
 c_{AA} + S_3(A) &= 0 + \infty = \infty \\
 c_{AC} + S_3(C) &= 1 + \infty = \infty \\
 c_{AG} + S_3(G) &= 1 + \infty = \infty \\
 c_{AT} + S_3(T) &= 1 + 0 = 1
 \end{aligned} \tag{14.3}$$

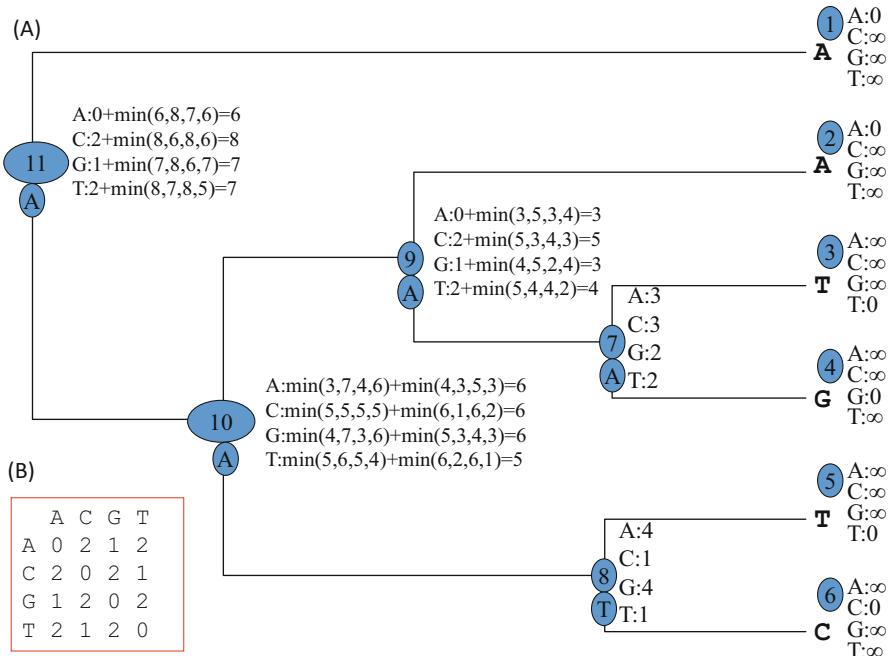
The minimum of the four values in Eq. 14.3 is 1 which simply means that it will incur a cost of 1 to go from an A at node 7 to a T at node 3. Similarly, we compute the four values based on the leaf node 4. The minimum value out of these four values is also 1, obtained as  $c_{AG} + S_4(G)$ . Summing up the two minimum values gives us  $S_7(A) = 2$ , which means that, if we have A at node 7, it will take a cost of 2 to account for the two daughter nodes with T and G, respectively (Fig. 14.1). We do the same to obtain  $S_7(C) = 2$ ,  $S_7(G) = S_7(T) = 1$ . The vector for node 8 is obtained in exactly the same way.

For  $S_9(A)$  with daughter nodes 2 and 7 in Fig. 14.1, the minimum of the four values based on node 2 is 0. The four values based node 7 are

$$\begin{aligned}
 c_{AA} + S_7(A) &= 0 + 2 = 2 \\
 c_{AC} + S_7(C) &= 1 + 2 = 3 \\
 c_{AG} + S_7(G) &= 1 + 1 = 2 \\
 c_{AT} + S_7(T) &= 1 + 1 = 2
 \end{aligned} \tag{14.4}$$

You must have noted that the two terms in each equation above are made of two terms, the first being the cost to go from nucleotide  $i$  in the parental node 9 to nucleotide  $j$  in the daughter node 7 and the second is the cost already accumulated at the daughter node 7. So  $S_9(A) = 0 + 2 = 2$ , which is obtained in three ways, with nucleotides at nodes 9 and 7 being (1) A and A, (2) A and G, and (3) A and T (Fig. 14.1). In the same way, we obtain  $S_9(C) = 3$ ,  $S_9(G) = S_9(T) = 2$  (Fig. 14.1). We progress all the way to node 11 to obtain the score vector [4, 5, 5, 4].  $L_j$  in Eq. 14.1 for this site is the minimum of these four values (= 4). For a C matrix in Fig. 14.1b,  $L_j$  is the minimum substitutions required to account for the nucleotide variation at the site in Fig. 14.1. In other words, mapping the first nucleotide site to the topology requires a minimum of four nucleotide substitutions. We will do the same for all other sites and then add all  $L_j$  values to obtain TL for the topology. The most parsimonious tree is the one with the smallest TL.

The maximum parsimony criterion cannot identify the root, although the tree in Fig. 14.1 is presented in a rooted form. We can take any internal node as the root node, and TL will be the same, i.e., topologies rerooted with any alternative internal node are equally parsimonious trees. For this reason, a phylogenetic tree is often rooted by an outgroup that is known not belong to the ingroup. An ingroup is a group of OTUs whose phylogenetic relationship needs to be resolved. An ideal outgroup is one that not only does not belong to the ingroup but also is the closest phylogenetically to the ingroup. If an ingroup consists of primate species, we should not use a bacterial species as an outgroup because it would be too distant to identify the root of



**Fig. 14.2** Computing substitution cost for the save nucleotide site as in Fig. 14.1 using the Sankoff algorithm (a) with the cost for transversions being 2 and that for the transitions being 1 (b). All nodes are numbered sequentially, with an S vector associated with each node. The circled nucleotides at internal nodes represent one of several possible ancestral reconstructions

the ingroup, although a bacterial species is a valid outgroup to the ingroup of mammalian species.

The strength of the Sankoff algorithm is that it allows a variety of cost matrices to be used. In some mammalian mitochondrial sequences where transitions occur about 40 times higher than transversions (Xia et al. 1996), one can specify a much high cost for transversions than transitions. In single-stranded DNA viruses where  $C \leftrightarrow T$  mutations occur more frequently than  $A \leftrightarrow G$  mutations (Chithambaram et al. 2014a, b; Frederico et al. 1990; Kreutzer and Essigmann 1998), one can assign a higher cost to  $A \leftrightarrow G$  substitutions than  $C \leftrightarrow T$  substitutions. Such a  $C$  matrix can alleviate the problem of substitution saturation and the long-branch attraction problem that is particularly pronounced in the maximum parsimony method.

The computational details involving a  $C$  matrix with a higher cost for transversions than for transitions are illustrated in Fig. 14.2. The final vector for node 11 is [6,8,7,7]. The  $L_j$  value in Eq. 14.1 for this site is the minimum of the vector (= 6) when the character at node 11 is A. This cost does not have the same straightforward interpretation as when all  $c_{ij} = 1$  (in which case the cost is the number of nucleotide substitutions). A cost of 6, in this case, could result from six transitional substitutions or two transitional plus two transversional substitutions or

$T_1$	12345678	$T_2$	12345678	$T_3$	12345678
S1 ATGTGTTA		S1 ATGTGTTA		S1 ATGTGTTA	
S2 ACGTACTG		S3 ATCTCTGA		S4 ACCTTCGG	
S3 ATCTCTGA		S2 ACGTACTG		S3 ATCTCTGA	
S4 ACCTTCGG		S4 ACCTTCGG		S2 ACGTACTG	
$L_{jF}$ 02103212		$L_{jF}$ 01203121		$L_{jF}$ 02203222	
$L_{js}$ 02204222		$L_{js}$ 01405141		$L_{js}$ 02405242	
$TL_{1F} = 11$		$TL_{2F} = 10$		$TL_{3F} = 13$	
$TL_{1S} = 14$		$TL_{2S} = 15$		$TL_{3S} = 19$	

**Fig. 14.3** Maximum parsimony evaluation of tree lengths (TL) for the three possible topologies ( $T_1$ ,  $T_2$ , and  $T_3$ ) with four aligned sequences (S1 to S4).  $L_{jF}$  and  $L_{js}$  are substitution cost at site  $j$  evaluated with the Fitch algorithm and Sankoff algorithm, respectively.  $TL_{iF}$  and  $TL_{is}$  are tree length for tree  $i$  from the Fitch algorithm and from Sankoff algorithm, respectively

just three transversional substitutions. One of the possible reconstruction of ancestral nodes, shown in Fig. 14.4, requires two transitions and two transversions. Note that such a reconstruction of ancestral nodes is not unique. For example, node 7 could be A or G or T, and node 8 could be C or T without increasing the total cost of 6.

The benefit of accommodating differential substitution rates may be illustrated with a practical example. Here, I compare the Fitch and Sankoff algorithms with sequence data in Fig. 14.3 which was part of simulated data along topology  $T_1$  (Fig. 14.3) based on the K80 model (Kimura 1980) with  $\alpha/\beta = 4$  (where  $\alpha$  and  $\beta$  stand for transitional and transversional rates, respectively). Because the Fitch algorithm assumes equal rates among all substitutions, we expect it to be less likely to recover the true tree (which is topology  $T_1$  that is used to simulate the sequence data). In contrast, the Sankoff algorithm allows us to give a higher cost to transversional substitutions than to transitional substitutions and therefore should be more appropriate for analyzing the data set than the Fitch algorithm. Indeed, the Fitch algorithm recovered topology  $T_2$  as the most parsimonious tree ( $TL_{2F} = 10$ , smaller than  $TL_{1F}$  and  $TL_{3F}$ , Fig. 14.3), whereas the Sankoff algorithm with the step matrix in Fig. 14.2b recovered  $T_1$  ( $TL_{1S} = 14$ , smaller than  $TL_{2S}$  and  $TL_{3S}$ , Fig. 14.3). Analogously, if we use either the distance-based methods or maximum likelihood with the JC69 model (Jukes and Cantor 1969) that does not accommodate the transition bias, then topology  $T_2$  will be recovered as the best tree. However, if a substitution model that accommodates the transition bias, such as K80, F84 (Felsenstein and Churchill 1996; Kishino and Hasegawa 1989), or TN93 (Tamura and Nei 1993), is used, then both the distance-based or maximum likelihood methods will recover tree  $T_1$  (the true tree) as the best tree. Numerous studies have shown that a judicious use of a step matrix in MP can often alleviate the long-branch attraction problem and result in improved accuracy in phylogenetic analysis.

The illustration above suggests the paramount importance of a good substitution model in phylogenetic analysis. Because values in a step matrix are quite subjective and does not have the statistical rigor of a substitution model, it is impossible to

know if a step matrix provides a good approximation to the substitution process or to evaluate statistically different step matrices to see which one is the best. In contrast, standard model selection criteria can be used to choose a best model for the data. One can use likelihood ratio test for nested models or information-theoretic indices such as AIC (Akaike 1973), BIC (Schwarz 1978), or Bayes factor (Kass and Raftery 1995) in other situations.

Another shortcoming which is MP relative to maximum likelihood (ML) method is in the handling of missing characters. Missing data occur when we concatenate sequences to create a large data matrix. However, one gene may be represented in all species, but another gene may only be sequenced for a subset of species. As will be shown later, ML has no problem in phylogenetic analysis with such data, but MP does.

### 3 The Uphill Search and Branch-and-Bound Search Algorithms

The number of possible topologies (Felsenstein 1978b) increases very quickly with the increase in the number of OTUs ( $n$ ), with the number of rooted and unrooted topologies, designated as  $N_R$  and  $N_U$ , respectively, being

$$\begin{aligned} N_R &= \frac{(2n - 3)!}{2^{n-2}(n - 2)!} \\ N_U &= \frac{(2n - 5)!}{2^{n-3}(n - 3)!} \end{aligned} \quad (14.5)$$

Searching through all possible topologies is called exhaustive search. It is often computationally impossible to search all possible trees with a large  $n$ . Two faster alternatives are commonly used. The first is the uphill search which does not guarantee that the resulting tree is the most parsimonious. The second is the branch-and-bound approach which does guarantee the finding of the most parsimonious tree. Obviously, the uphill search is much faster than the exhaustive search which evaluates all possible topologies. The shortcoming of the uphill search is that it may miss the most parsimonious tree (which may be among those discarded without being evaluated).

The branch-and-bound algorithm for the MP method (Hendy and Penny 1982) starts by generating an initial tree with fast algorithms such as the uphill search. Designate the tree length of this initial tree as  $TL_i$ , which is now the upper bound of  $TL$  for the true MP tree. One may also use the neighbor-joining method (Saitou and Nei 1987) to generate a topology and then evaluate the topology to obtain  $TL_i$ . The resulting distance-based tree also allows us to know which OTUs exhibit the highest divergence. This information is crucial for an efficient branch-and-bound search.

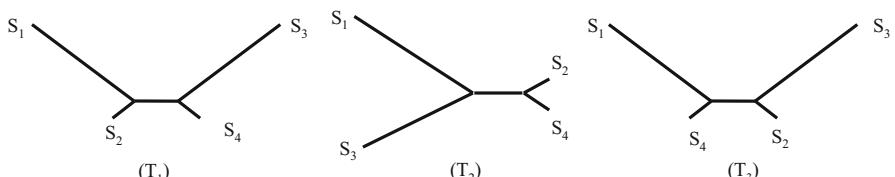
The next step is to start with four most divergent sequences and evaluate their TL. If only one topology is shorter than  $TL_i$ , then we can safely discard the two other topologies with  $TL > TL_i$  because TL will only get longer with more OTUs added. This is why we always add more divergent OTUs to the growing tree first because this increases the chance of having subtrees with a  $TL > TL_i$  so that such subtrees get eliminated early. If we take four closely related species to start, then none of the three topologies will have  $TL > TL_i$ , and we would have to keep them all, leading to a very inefficient branch-and-bound algorithm. If no 4-OTU topology is eliminated (i.e., their TL are all shorter than  $TL_i$ ), then we have to evaluate all 5-OTU topologies derived from them. Again we evaluate all these 5-OTU topologies and eliminate anyone that has  $TL > TL_i$ . This process is continued until we have added all OTUs and declare with confidence that our final tree is the true MP tree.

The efficiency of the branch-and-bound algorithm depends much on the initial topology. If the initial topology is good (i.e., with  $TL_i$  being very close to TL of the true MP tree), then many non-MP alternatives get eliminated early. If  $TL_i$  is unduly long, then the algorithm will be nearly as slow as the exhaustive search.

## 4 The Long-Branch Attraction Problem

The MP method is known to be inconsistent (Felsenstein 1978a; Takezaki and Nei 1994) as a result of long-branch attraction. Consistency is an important criterion for a good estimator, the others being unbiased and efficient (with small variance of the parameter). A consistent estimator is one that, biased or not, will converge to the true parameter (e.g., the true tree) with increasing probability when sample size increases (e.g., longer and longer sequences). An inconsistent estimator is one that will become increasingly more likely to converge to a wrong parameter value with increasing sample size. Thus, an estimator that is inconsistent is deemed unacceptable. I will provide a simple demonstration here by using trees in Fig. 14.4. With four species, we have three possible unrooted topologies, designated  $T_k$  ( $k = 1, 2, 3$ ), with  $T_1$  being the correct topology.

Let  $X_{ij}$  be nucleotide at site  $j$  for species  $S_i$  and  $L$  be the sequence length. For simplicity, assume that nucleotide frequencies are all equal to 0.25. Suppose that the lineages leading to  $S_1$  and  $S_3$  have experienced full substitution saturation, so that



**Fig. 14.4** The long-branch attraction problem in the maximum parsimony methods

$$p(X_{1j} = X_{ij, i \neq 1}) = p(X_{3j} = X_{ij, i \neq 3}) = 0.25 \quad (14.6)$$

where  $p$  stands for probability. The lineages leading to  $X_2$  and  $X_4$  have not experienced substitution saturation and should have  $p(X_{2j} = X_{4j}) > 0.25$ . Let's set  $p(X_{2j} = X_{4j}) = 0.8$  and sequence length  $L = 1000$ .

We now consider the expected number of informative sites, designated by  $n_k$  ( $k = 1, 2, 3$ ), favoring  $T_k$ . By definition, site  $j$  is informative and favoring  $T_1$  if it meets the following three conditions:  $X_{1j} = X_{2j}$ ,  $X_{3j} = X_{4j}$ , and  $X_{1j} \neq X_{3j}$ . Similarly, site  $j$  favors  $T_2$  if  $X_{1j} = X_{3j}$ ,  $X_{2j} = X_{4j}$ , and  $X_{1j} \neq X_{2j}$ . Thus, the expected numbers of informative sites favoring  $T_1$ ,  $T_2$ , and  $T_3$ , respectively, are

$$\begin{aligned} E(n_1) &= p(X_{1j} = X_{2j})p(X_{3j} = X_{4j})p(X_{1j} \neq X_{3j})p(X_{1j} \neq X_{4j}) \\ &\quad p(X_{2j} \neq X_{3j})p(X_{2j} \neq X_{4j})L \\ &= 0.25 \times 0.25 \times 0.75 \times 0.75 \times 0.75 \times 0.20 \times 1000 \approx 5 \\ E(n_2) &= p(X_{1j} = X_{3j})p(X_{2j} = X_{4j})p(X_{1j} \neq X_{2j})p(X_{1j} \neq X_{4j}) \\ &\quad p(X_{3j} \neq X_{2j})p(X_{3j} \neq X_{4j})L \\ &= 0.25 \times 0.8 \times 0.75 \times 0.75 \times 0.75 \times 0.75 \times 1000 \approx 63 \\ E(n_3) &= E(n_1) \approx 5. \end{aligned} \quad (14.7)$$

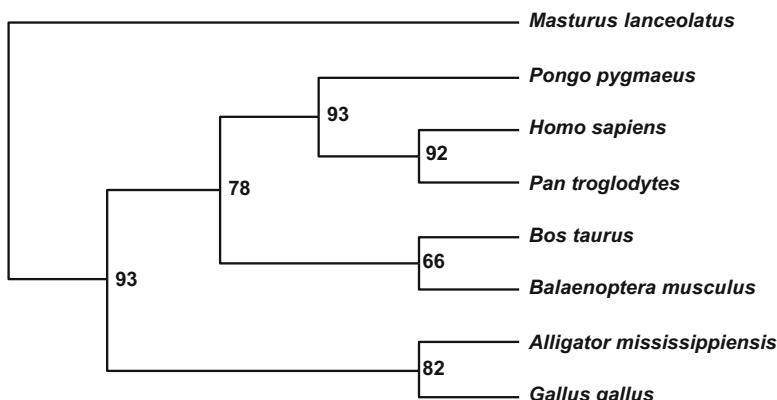
Equation 14.7 is contributed by Yongxin Ye when I was teaching a summer course at Peking University where Yongxin was a PhD student. Yongxin sent me the equation to correct an error I made. In spite of  $T_1$  being the true topology, we should have, on average, only about 5 informative sites favoring  $T_1$  and  $T_3$ , but 63 sites supporting the wrong tree  $T_2$ . This is one of the several causes for the familiar problem of long-branch attraction (Hendy and Penny 1989) or short-branch attraction (Nei 1996). Because it is the two short branches that contribute a large number of informative sites supporting the wrong tree, “short-branch attraction” seems a more appropriate term for the problem than “long-branch attraction.” However, because the problem is mainly caused by the failure of correcting for multiple hits in long branches, naming it long-branch attraction makes sense. I wish to emphasize again that long-branch attraction problem is not unique in MP. It occurs in distance-based, maximum likelihood, and Bayesian inference methods as well when multiple hits are not corrected properly. For example, if sequences evolve with a strong transition bias, but we choose the JC60 model which will not correct multiple substitutions as a consequence of strong transition bias, then all model-based methods will also exhibit long-branch attraction problem.

Inconsistency in statistical estimation is never a good thing. Some advocates of the MP method have gone so far as to claim that long-branch attraction problem could be a good thing because lineages with long branches tend to be sister lineages, so a bias favoring their grouping together will, in fact, give MP greater efficiency to arrive at the true tree. I will not name names but will simply point out that such great efficiency is purchased with illegal statistical currency.

## 5 Bootstrapping and Delete-Half Jackknifing

It is desirable to express our confidence in a tree or its subtrees. One frequently used approach for this purpose is by the resampling methods such as bootstrapping and jackknifing (Efron 1982), first introduced to phylogenetics by Felsenstein (Felsenstein 1985). The bootstrap resampling involves the site-wise resampling of the aligned sequences with replacement, so that sequences of  $N$  sites long will be resampled  $N$  times to generate a new set of aligned sequences of the same length (a pseudosample), with some sites in the original sequences sampled one or more times, while some other sites do not get sampled at all. In short, to obtain each pseudosample, one generates  $N$  random integers between 1 and  $N$ , e.g., 3, 1, 101, . . . , and put sites 3, 1, 101, . . . , in the new pseudosample. In molecular phylogenetics, one typically will generate at least 500 pseudosamples, so we will reconstruct 500 trees, each from a pseudosample, and obtain a consensus tree expressing our confidence in the subtrees (Fig. 14.5). A bootstrap value of 93 for the three great apes (*Pongo pygmaeus*, *Homo sapiens*, and *Pan troglodytes*) means that 93% of the trees group the three species together as a monophyletic taxon. Note that the bootstrapping values will be similar but not identical if one repeats the procedure to regenerate another bootstrapped tree.

The delete-half jackknifing technique will randomly purge off half of the sites from the original sequences to generate a pseudosample, so that the new set of aligned sequences in the pseudosample will be half as long as the original. The rest is the same as with bootstrapping. Both resampling methods give us information on data consistency. If all sites contain similar phylogenetic signals, then the support values will be high. If different sites contain conflicting phylogenetic signals, then the support value will be low.



**Fig. 14.5** A maximum parsimony tree of eight vertebrate species with bootstrap support values. Generated from DAMBE (Xia 2013, 2017d)

Although the bootstrap and the jackknife generally produce similar results, there are some subtle differences. Suppose that we have a set of aligned sequences of  $N$  sites long. For the bootstrap resampling, the probability of each site being sampled is  $1/N$ , and the mean number of times a site gets sampled in each bootstrapping resampling is simply one. Thus, a site gets sampled 0, 1, 2, ...,  $N$  times follows a Poisson distribution with a mean equal to one. The proportion of sites that will not be sampled will them be  $p(0) = e^{-1} = 0.368$  which means that 36.8% of the sites will not be sampled, while 63% of the sites will be sampled at least once. In jackknifing, we have 50% of the sites not sampled and the other 50% of the sites sampled just once.

## 6 Statistical Tests of Alternative MP Topologies

It is often useful to test if alternative topologies are significantly different from each other. In order to have a powerful test, it is crucial to consider (1) what sites are relevant to the test, (2) what OTUs are relevant to the test, and (3) when to include sites as a random effect.

### 6.1 What Sites Are Relevant to the Test?

Suppose I have four species and wish to evaluate relative statistical support for the three alternative topologies. I will use cow (*Bos taurus*), orangutan (*Pongo pygmaeus*), chimpanzee (*Pan troglodytes*), and human (*Homo sapiens*) mitochondrial COX1 gene to illustrate the test. These are taken from the VertCOI.fas file that comes with software DAMBE (Xia 2013) and contains 1512 aligned sites of mitochondrial COI genes with no indels (i.e., no inferred indel events during their evolution). We can use either the Fitch algorithm or Sankoff algorithm to evaluate  $L_j$  (minimum number of steps for site  $j$ ) for each of the three possible topologies ( $T_1$ ,  $T_2$ , and  $T_3$ ) to obtain the site-specific steps in a table (Fig. 14.6). This table can then

The figure shows three phylogenetic trees (T<sub>1</sub>, T<sub>2</sub>, T<sub>3</sub>) for four species: Cow, Human, Orangutan, and Chimp. Tree T<sub>1</sub> has a total length of 540, with branches connecting Cow-Orangutan, Orangutan-Human, and Human-Chimp. Tree T<sub>2</sub> has a total length of 563, with branches connecting Cow-Human, Human-Orangutan, and Orangutan-Chimp. Tree T<sub>3</sub> has a total length of 563, with branches connecting Cow-Chimp, Chimp-Human, and Human-Orangutan.

Site	T <sub>1</sub>	T <sub>2</sub>	T <sub>3</sub>
L1	0	0	0
L2	0	0	0
L3	1	1	1
...	...	...	...
L1512	0	0	0

**Fig. 14.6** Three topologies for illustrating statistical tests of alternative topologies. Tree length is 540, 563, and 563 for  $T_1$ ,  $T_2$ , and  $T_3$ , respectively. It would be a mistake to include all sites in the statistical test

be used to evaluate relative statistical support of alternative topologies, with the null hypothesis being that the three column means for  $T_1$ ,  $T_2$ , and  $T_3$ , respectively, in the table in Fig. 14.6 are equal. What is important is to consider which site should be included in the test.

It makes no sense to include all 1512 sites in the significance tests. For sequences with little divergence, most sites will be identical and have  $L_j = 0$  for all three topologies. A large number of shared 0's will obscure true differences among the three topologies. For example, if we subject the table in Fig. 14.6 to a one-way ANOVA, we will see no significant difference among the three topologies ( $F = 0.2947$ ,  $DF1 = 2$ ,  $DF2 = 4533$ ,  $p = 0.7448$ ). You can use alternative statistical tests other than ANOVA, but the point here is to illustrate that, regardless of what statistical test you use, including irrelevant sites will reduce the power of the test.

If we exclude all identical sites, then the  $p$  value becomes 0.1505. However, this is still not appropriate. For example, a large number of sites could be autapomorphic with only one species differing from the other three species. Such sites will have  $L_j = 1$  for all three topologies. A large number of shared 1's will also obscure true differences among the three topologies. Excluding all these autapomorphic sites leaves us with only 145 sites, but these sites yield a highly significant difference among the 3 topologies ( $F = 6.8246$ ,  $DF1 = 2$ ,  $DF2 = 432$ ,  $p = 0.001207758$ ).

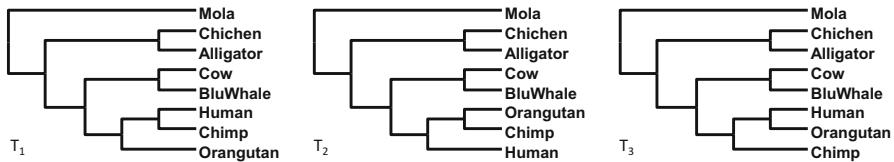
We can refine the test even further. The tabled  $L_j$  values in Fig. 14.6 are evaluated by the Fitch algorithm for which only informative sites matter. An informative site contains at least two nucleotides with each nucleotide represented by at least two OTUs. Thus, for these  $L_j$  values generated from the Fitch algorithm, we should use only informative sites in the significance tests. There are 83 informative sites. Applying one-way ANOVA to these sites gives us  $F = 10.2307$ ,  $DF1 = 2$ ,  $DF2 = 246$ , and  $p = 0.000054$ . Both  $T_2$  and  $T_3$  are highly significantly worse than  $T_1$  ( $p = 0.0001$  for both comparisons between  $T_1$  and  $T_2$  and between  $T_1$  and  $T_3$ ).

There is no point to consider sites as a random effect with only four species because, when we use only informative sites, all sites will have the same steps. An informative site will have  $L_j = 1$  for the topology that it supports and  $L_j = 2$  for the two other topologies that it does not support. So all sites will have exactly the same mean  $L_j$ .

## 6.2 *What OTUs to Include?*

Suppose that we now add more species to Fig. 14.6 but keep the same biological question of testing the three alternative topologies featuring different relationships among the three hominids (Fig. 14.7). The three topologies are the same except for the different relationships among the three hominids. We again use the Fitch algorithm to evaluate the 473 informative sites, and I have made available the  $L_j$  values for the 3 topologies at <http://dambe.bio.uottawa.ca/teach/VertCOI.txt>.

We now subject these  $L_j$  values to a one-way ANOVA. To our surprise, there is no significant difference among the three topologies ( $F = 0.2804$ ,  $DF1 = 2$ ,



**Fig. 14.7** Three topologies for illustrating the effect of adding irrelevant species in testing alternative topologies. Tree length is 1427, 1446, and 1445 for T<sub>1</sub>, T<sub>2</sub>, and T<sub>3</sub>, respectively

DF2 = 1416,  $p = 0.7555$ ), in contrast to our previous results of highly significant differences among the three topologies in Fig. 14.6, also on the informative sites. The reason for this discrepancy is that, when we add more species, we added 390 (=473–83) informative sites that, while contributing to the resolution of other parts of the tree, contribute nothing to resolving the 3 topologies in Fig. 14.7. That is, these 390 informative sites all add exactly the same number of steps to the 3 topologies. In short, while there are informative sites that supports T<sub>1</sub> against T<sub>2</sub> and T<sub>3</sub> (Fig. 14.7), the effect of these sites is diluted by the additional of a large number of sites that support the three topologies equally.

The rule of thumb in phylogenetics is that, if you wish to resolve phylogenetic controversies among great apes, you can include the lesser apes as outgroup species but not remote relatives such as fish or worms or even bacteria. The outgroup provides you with a perspective to see which ingroup species is close to the root. If the outgroup is very far from the ingroup, then all ingroup species will become blurry. If you really have to include more distant relatives, then the next section will offer you a binocular.

### 6.3 Include Sites as a Random Effect in a Mixed Model

We mentioned that there is no point to include sites as a random effect when we use informative sites with only four OTUs because all sites will have the same mean  $L_j$  values. With more than four OTUs, including sites as a random effect will increase the power of statistical tests if there is heterogeneity among sites, i.e.,  $L_j$  depends on both topology (TREE) and sites (SITE). There are specialized programs for mixed models, but in our case, just a two-way ANOVA with no replication (which is an extension of the paired-sample t-test, just as a one-way ANOVA is an extension of the unpaired two-sample t-test) would suffice. Note that we cannot test TREE\*SITE interactions. In R, one would specify  $\text{lm}(L_j \sim \text{TREE} + \text{SITE})$ . The result shows highly significant differences in tree length among the three topologies ( $p = 0.002129$ ) and among sites ( $p < 0.000001$ ). If one uses the lmer function in R, fit a model with both TREE and SITE and a reduced model without TREE but with SITE, then a

likelihood ratio test will yield a  $p = 0.002129$ , i.e., the tree lengths of the three topologies are significantly different from each other. This is a demonstration that a mixed model with SITE as a random effect could dramatically increase the power in phylogenetic tests. I expect such models to be implemented in all phylogenetic analysis that are site-based (e.g., maximum parsimony and maximum likelihood).

## Postscript

(Warning!! Adult content)

Many computational or statistical methods have assumptions. Many mathematical models are also derived with assumptions. The methods and models would be wrong if the assumptions are not met. Applying a method or model in situations where the assumptions are violated is termed misapplication. The MP method strongly assumes slow evolution with negligible multiple substitutions at the same site and should be used only with sequences of slow evolution. MP is very popular among early plant phylogeneticists working on plant mitochondrial genes because plant mitochondrial genomes evolve very slowly. In contrast, animal mitochondrial genomes, except for ancient and primitive lineages such as sponges and hydra, evolve very quickly. It would be wrong to apply MP to resolving deep divergence among animal lineages using animal mitochondrial DNA.

One colleague shared with me a Chinese story to illustrate the evil of misapplying a method. The story could have originated elsewhere and translated into Chinese. In spite of a little bit of adult content and being somewhat insensitive to the modern culture, I decided to include it here for two reasons: (1) it makes a good point and (2) out of respect for my colleague who shared the story with me.

Once there was a watercolor painter who had been mediocre for many years. Then, as if through a Midas touch, he suddenly began to produce highly characteristic paintings of butterflies. His fame spread far and wide and won him a title of a grand master. One female fellow painter, intrigued by his mysterious painting technique, came to ask for enlightenment. The grand master was very generous and explained that his technique was in fact very simple and easy to learn: “I just fetch a portable wash basin, fill it half-full with water, mix water color in it, take pants off and dip my bottom in the wash basin, and finally sit on Xuan paper. That is all what it takes to produce a painting of a beautiful but mystic butterfly.”

The fellow painter went home and practiced for a few months but came back to complain that she was not able to sell any of her butterfly paintings. The grand master, upon seeing her butterfly paintings, burst out laughing: “Your butterflies have no head!”

# Chapter 15

## Distance-Based Phylogenetic Methods



### 1 Introduction

Phylogenetics has two main objectives. The first is to build phylogenetic relationships among operational taxonomic units (OTUs which could be a group of species or a gene family with duplicated genes). The second is to date speciation or gene duplication events. There are four categories of methods that can contribute to the first objective, maximum parsimony, distance-based, maximum likelihood method, and Bayesian inference, but only the last three can contribute to the second objective.

This chapter is on distance-based phylogenetic reconstruction and dating. For the first objective, we need a distance matrix and a tree-building algorithm making use of the distance matrix. For dating, we need calibration points to convert branch lengths to geological time. There are two commonly used calibration methods, one by using fossil record to calibrate internal nodes and the other by differential sampling times for rapidly evolving viral lineages, e.g., RNA viruses such as HIV-1. This latter calibration method is commonly known as tip-dating (Drummond et al. 2003a, b).

Distance-based phylogenetic methods, especially those based on the least-squares criterion, are widely used in biomedical research and featured in major textbooks on molecular phylogenetics (Felsenstein 2004; Li 1997; Nei and Kumar 2000; Yang 2006). The least-squares method for phylogenetic reconstruction is generally consistent when the distance is estimated properly (Felsenstein 2004; Gascuel and Steel 2006; Nei and Kumar 2000). However, even when the distance is over- or underestimated, the resulting bias is generally quite small (Xia 2006).

Distance-based method has one advantage not shared by other phylogenetic methods, i.e., it can be done with pairwise alignment only. Phylogenetic reconstruction becomes difficult with deep phylogenies, mainly due to the difficulty in obtaining reliable MSA (Blackburne and Whelan 2013; Edgar and Batzoglou 2006; Herman et al. 2014; Kumar and Filipski 2007; Lunter et al. 2008; Wong et al. 2008). In contrast to pairwise sequence alignment (PSA) by dynamic

programming which is guaranteed to generate, for a given scoring scheme, the optimal alignment or at least one of the equally optimal alignments, MSAs of highly divergent sequences are often poor, especially those obtained from a progressive alignment with a guide tree. Although an iterative approach (Feng and Doolittle 1990; Hogeweg and Hesper 1984; Katoh et al. 2009; Thompson et al. 1994) is typically used for MSA in which a guide tree is used to generate an alignment which is then used to construct a new guide tree to guide the next round of progressive alignment, such approach still often produces poor alignment for deeply diverged sequences, and a poor alignment typically leads to bias and inaccuracy in phylogenetic estimation (Blackburne and Whelan 2013; Kumar and Filipski 2007; Wong et al. 2008).

One way to avoid problems associated with poor MSA is simply not to do MSA, i.e., by phylogenetic analysis based on PSA only as pioneered by Thorne and Kishino (1992). However, Thorne and Kishino (1992) did not actually implement the method and evaluate it against the maximum likelihood (ML) method. It is only recently that the method, termed PhyPA for phylogenetics from pairwise alignment, has been implemented in DAMBE (Xia 2016). PhyPA outperforms maximum likelihood methods for highly diverged sequences.

This chapter has three sections. I first introduce the simultaneously estimated distances and the paralinear/LogDet distances. While a previous chapter on substitution models covers the conceptual framework of deriving evolutionary distances, all those distances are independently estimated distances with inherent problems. This is followed by numerical illustration of phylogenetic algorithms building trees from distances. The last section of this chapter illustrates the distance-based dating methods. All the distance-based phylogenetic methods in this chapter, as well as independently estimated and simultaneously estimated distances, are implemented in DAMBE (Xia 2013, 2017d).

## 2 Simultaneously Estimated Distances and Paralinear/LogDet Distances

### 2.1 Simultaneously Estimated Distances

The conceptual framework of deriving evolutionary distances from substitution models has been covered extensively in a previous chapter on substitution models, illustrated with commonly used substitution models such as JC69 (Jukes and Cantor 1969), K80 (Kimura 1980), F84 (used in DNAML since 1984, Hasegawa and Kishino 1989; Kishino and Hasegawa 1989), HKY85 (Hasegawa et al. 1985), TN93 (Tamura and Nei 1993), and GTR (Lanave et al. 1984; Tavaré 1986) models. Distances derived from these models are, respectively, referred to as JC69 distances, K80 distances, and so on.

All those distances derived in the chapter on substitution models are independently estimated (IE) distances. Each IE distance is based on information from a single pair of sequences and suffers from three problems. The first involves inapplicable cases where the distance often cannot be computed for highly diverged sequences (Rzhetsky and Nei 1994; Tajima 1993; Zharkikh 1994). For example, the K80 distance cannot be computed when  $(1 - 2Q \leq 0)$  or  $(1 - 2P - Q \leq 0)$ . The second is internal inconsistency, with the substitution process between sequences A and B having rate ratios  $\kappa_{AB}$  but that between sequences A and C having rate ratios  $\kappa_{AC}$  (Felsenstein 2004, p. 200; Yang 2006, pp. 37-38). These two problems are exacerbated by limited sequence length with greater stochastic effects. The third problem is insufficient use of information because the computation of pairwise distances ignores information in other sequences that should also contain information about the substitution process involving the two compared sequences (Felsenstein 2004, p. 175; Yang 2006, p. 37). Because of these problems, distance-based phylogenetic methods are generally considered as quick and dirty methods, used either in situations where high phylogenetic accuracy is not particularly important or as a first step to generate preliminary candidate trees for subsequent more rigorous phylogenetic evaluation by maximum likelihood methods (Ota and Li 2000; 2001). However, these problems can be eliminated, or at least dramatically alleviated, by simultaneously estimated (SE) distances.

SE distances use information from all pairs of sequences and generally produce much more robust trees than IE distances. The first part of this chapter covers two approaches, the LS approach and the ML (maximum likelihood) approach, for simultaneously estimating distances, as well as two distances that have not been covered before: the paralinear (Lake 1994) and LogDet (Lockhart et al. 1994) distances which are similar to each other. These distances have been implemented in my program DAMBE (Xia 2001, 2013; Xia and Xie 2001b).

There are two approaches to derive SE distances. The first is the quasi-likelihood approach (Tamura et al. 2004), referred to as the maximum composite likelihood distance in MEGA (Tamura et al. 2007) and MLComposite in DAMBE (Xia 2009, 2013, 2017d), respectively. MEGA implemented the distance only for the TN93 model (Tamura and Nei 1993), whereas DAMBE implemented it for both the TN93 and the F84 models, referred as MLCompositeTN93 and MLCompositeF84, respectively. The second approach for deriving SE distances is the least-squares (LS) or weighted least-squares (WLS) approach that has been implemented in DAMBE for F84, TN93, and GTR models.

### 2.1.1 Simultaneous Estimation of Distances with the Quasi-likelihood Approach

I will use the K80 model (Kimura 1980) to illustrate the principle of SE distance estimation based on the likelihood framework. The K80 model, as we have learned from the chapter on nucleotide substitution models, has two parameters, which can be expressed either as  $\alpha t$  and  $\beta t$  or as  $D$  and  $\kappa$ . The expected proportions of

**Table 15.1** Observed number of sites with transitional and transversional differences ( $N_s$  and  $N_v$ , respectively) from pairwise comparisons among three sequences (S1, S2, and S3) of length 100

Seq. Pair	$N_s$	$N_v$	$D_{IE}$	$D_{SE,ML}$	$D_{SE,LS}$
S1 vs S2	9	4	0.1451	0.1464	0.1473
S1 vs S3	40	30	Inapplicable	2.0914	2.0725
S2 vs S3	20	10	0.4024	0.4116	0.4131

Independently estimated K80 distances ( $D_{IE}$ ) and simultaneously estimated K80 distances ( $D_{SE}$ ) from the maximum likelihood (ML) and least-squares (LS) frameworks are included. Note that K80 distance cannot be computed for S1 and S3 using independent estimation method (labeled as “Inapplicable”)

transitions and transversions, designated by  $E(P)$  and  $E(Q)$  and expressed in  $D$  and  $\kappa$ , respectively, are

$$\begin{aligned} E(P) &= \frac{1}{4} + \frac{1}{4}e^{-\frac{4D}{\kappa+2}} - \frac{1}{2}e^{-\frac{2D(\kappa+1)}{\kappa+2}} \\ E(Q) &= \frac{1}{2} - \frac{1}{2}e^{-\frac{4D}{\kappa+2}} \end{aligned} \quad (15.1)$$

Suppose we have three aligned sequences, S1, S2, and S3, with sequence length of  $L$  ( $= 100$ ). The observed number of sites with transitional and transversional differences are shown in Table 15.1, used for contrasting the IE and SE estimation of distances. For IE estimation, one simply substitutes  $E(P)$  and  $E(Q)$  in Eq. (15.1) by the observed  $P$  and  $Q$  (which equal  $N_s/L$  and  $N_v/L$ , respectively) and then solves for  $D$  and  $\kappa$ . You will encounter the frustration that you cannot estimate  $D$  between S1 and S3 (Table 15.1).

For the SE method, we assume the same  $\kappa$  but different  $D_{ij}$  (i.e.,  $D_{12}$ ,  $D_{13}$ , and  $D_{23}$ ) and maximize the following  $\ln L$ :

$$\begin{aligned} \ln L = & N_{s,12} \ln [E(P_{12})] + N_{v,12} \ln [E(Q_{12})] + N_{i,12} \ln [1 - E(P_{12}) - E(Q_{12})] \\ & + N_{s,13} \ln [E(P_{13})] + N_{v,13} \ln [E(Q_{13})] + N_{i,13} \ln [1 - E(P_{13}) - E(Q_{13})] \\ & + N_{s,23} \ln [E(P_{23})] + N_{v,23} \ln [E(Q_{23})] + N_{i,23} \ln [1 - E(P_{23}) - E(Q_{23})] \end{aligned} \quad (15.2)$$

We take partial derivative of  $\ln L$  with respect to  $D_{12}$ ,  $D_{13}$ ,  $D_{14}$ , and  $\kappa$ , set them to zero, and solve the four simultaneous equations for the four unknowns. This leads to  $\kappa = 6.2382$  and the three  $D_{ij}$  values shown in Table 15.1. There is no problem for the SE method to estimate  $D_{13}$  which was not possible with the IE method.

### 2.1.2 Simultaneous Estimation of Distances with the Least-Squares (LS) Approach

The LS method uses one of a class of models differing in the form of weight. The general form is to minimize the following residual sum of squares (RSS):

$$\text{RSS} = \sum \left[ \frac{(o_{ij} - e_{ij})^2}{w_{ij}^m} \right] \quad (15.3)$$

where  $o_{ij}$  and  $e_{ij}$  are the observed and expected values,  $w_{ij}$  is the weight typically equal to  $o_{ij}$  or  $e_{ij}$ , and  $m$  is an integer typically taking the value of 0, 1, or 2. If  $m = 0$ , Eq. (15.3) becomes ordinary least-squares (OLS) method. If  $m > 0$ , then RSS represents weighted least-squares (WLS) method. If  $m = 1$  and  $w_{ij} = e_{ij}$ , RSS is the familiar chi-square ( $\chi^2$ ). Thus, parameter estimation by minimizing the chi-square function is a special case of the WLS method. The Fitch-Margoliash method (Fitch and Margoliash 1967) for reconstructing phylogenies from a distance matrix has  $m = 2$  and  $w_{ij} = o_{ij}$  (where  $o_{ij}$  is the estimated evolutionary distance between species  $i$  and  $j$ , and  $e_{ij}$  is the patristic distance along the path linking species  $i$  and  $j$  on the tree.)

I illustrate the WLS method for estimating  $D$  and  $\kappa$  by specifying  $m = 1$  and  $w_{ij} = e_{ij}$ , i.e., RSS in Eq. (15.3) is  $\chi^2$ . For IE estimation of  $D_{12}$ , we would minimize the following  $\chi^2$ :

$$\begin{aligned} \chi_{12}^2 = & \frac{[P_{12} - E(P_{12})]^2}{E(P_{12})} + \frac{[Q_{12} - E(Q_{12})]^2}{E(Q_{12})} \\ & + \frac{\{1 - P_{12} - Q_{12} - [1 - E(P_{12}) - E(Q_{12})]\}^2}{1 - E(P_{12}) - E(Q_{12})} \end{aligned} \quad (15.4)$$

where  $P_{12} = N_{s,12}/N$  and  $Q_{12} = N_{v,12}/N$  (values in Table 15.1). Minimizing  $\chi_{12}^2$  would result in  $D_{12} = 0.1451$  and  $\kappa = 4.9596$ .

For the SE method with  $N$  species and  $N^*(N - 1)/2$  pairwise distances, we will again assume the same  $\kappa$  but different  $d_{ij}$  values and minimize the following:

$$\begin{aligned} \chi^2 &= \sum_{i=1}^{N-1} \sum_{j=i+1}^N \chi_{ij}^2 \\ \chi_{ij}^2 &= \frac{[P_{ij} - E(P_{ij})]^2}{E(P_{ij})} + \frac{[Q_{ij} - E(Q_{ij})]^2}{E(Q_{ij})} \\ &+ \frac{\{1 - P_{ij} - Q_{ij} - [1 - E(P_{ij}) - E(Q_{ij})]\}^2}{1 - E(P_{ij}) - E(Q_{ij})} \end{aligned} \quad (15.5)$$

which leads to  $\kappa = 6.276273961$  and  $D_{12}$ ,  $D_{13}$ , and  $D_{23}$  values shown in Table 15.1. These four parameters are close to those estimated in the likelihood framework (also shown in Table 15.1).

## 2.2 Paralinear and LogDet Distances

All substitution models that I have covered are time-reversible stationary models in which the change of a nucleotide or an amino acid to another is balanced by the reverse change so that the nucleotide or amino acid frequencies remain the same

along the lineages. However, substitution rates may also change over time. For example, some bacterial lineages have several CpG-specific methyltransferases, with consequent high rate of methylated C changing to T leading to low frequency of C as well as CpG deficiency (Xia 2003). However, genes encoding such CpG-specific methyltransferases may lose during evolution, with descendent lineages regaining C and CpG dinucleotides (Xia 2003). This is a case of nonstationary substitution process. Phylogeneticists are all aware of such processes but generally avoided the topic because it is too complicated to implement nonstationary substitution process in phylogenetics. It is easy to say that substitution rates are functions of time, but no one knows what relationship is between rates and time.

Paralinear (Lake 1994) and LogDet (Lockhart et al. 1994) distances are suggested to be relevant to nonstationary process. For two nucleotide or amino acid sequences, the paralinear distance (Lake 1994) between sequences 1 and 2 is defined as

$$D_{12} = -\frac{1}{N} \ln \frac{|J_{12}|}{\sqrt{\prod_{i=1}^N F_{1i} \prod_{i=1}^N F_{2i}}} \quad (15.6)$$

where  $N$  is 4 for nucleotide sequences and 20 for amino acid sequences,  $J_{12}$  is the observed substitution matrix, and  $F_1$  and  $F_2$  are nucleotide frequencies (counts, not proportions) for sequences 1 and 2, respectively, derived from  $J_{12}$ . For data in Table 15.2, the  $|J_{12}| = 544202788$ , denominator = 1843167240, and  $D_{12} = 0.30498$ .

The LogDet distance (Lockhart et al. 1994) is except for two minor details. It defines  $J_{12}$  as proportions summing up to 1, i.e., with each value in the 4 by 4  $J_{12}$  matrix in Table 15.2 divided by the sum (= 860), and replaces  $F_1$  and  $F_2$  by  $p_1$  and  $p_2$  which are proportions summing up to 1. LogDet distance would be the same as paralinear distance if  $p_1$  and  $p_2$  are calculated from  $F_1$  and  $F_2$  as are done in Table 15.2. However, there are some reasons for one to use  $p_1$  and  $p_2$  different from those in Table 15.2. One of the reasons for 18S rRNA sequences goes as follows. First, both substitution and indel events have occurred almost exclusively in just a few variable domains of the 18S rRNA sequences (Van de Peer et al. 1993), but the variable domains have nucleotide frequencies different from those of the

**Table 15.2** Observed nucleotide substitution patterns from comparing two aligned sequences 1 and 2, for illustrating the calculation of paralinear and LogDet distances

	A	G	C	T	$F_2$	$p_2$
A	<b>100</b>	<b>40</b>	<b>8</b>	<b>12</b>	160	0.186047
G	<b>35</b>	<b>200</b>	<b>11</b>	<b>8</b>	254	0.295349
C	<b>9</b>	<b>9</b>	<b>250</b>	<b>15</b>	283	0.32907
T	<b>11</b>	<b>12</b>	<b>20</b>	<b>120</b>	163	0.189535
$F_1$	155	261	289	155	860	1
$p_1$	0.180233	0.303488	0.336047	0.180233		

The  $J_{12}$  matrix is in bold.  $F_1$  and  $F_2$  are the column and row sums of  $J_{12}$

conserved domains in the 18S rRNA gene and in the 28S rRNA gene (Xia et al. 2003a). In phylogenetic analyses involving distance and maximum likelihood methods, frequency parameters most appropriate for the underlying substitution model should be used. The most appropriate estimate of the frequency parameters should be from the sites where substitution occurs, i.e., from the variable domains. However, variable domains in the 18S rRNA sequences are poorly represented in the aligned sites because of the presence of many indels in these domains. Thus,  $p_1$  and  $p_2$ , as calculated in Table 15.2, are dominated by conservative domains and consequently are not appropriate for phylogenetic reconstruction. For this reason, one might use  $p_1$  and  $p_2$  calculated from all sequences instead of those site pairs entered in  $J_{12}$  (Xia et al. 2003a). Obviously, if one computes LogDet distances by using  $p_1$  and  $p_2$  that are different from those in Table 15.2, then such distances will be different from paralinear distances.

### 2.3 *Other Distances Not from Aligned Sequences*

Evolutionary distances can be computed from a variety of data other than molecular sequence data. Conventional data includes 1D and 2D gel protein electrophoresis, DNA hybridization, restriction fragment length polymorphism, gene frequency data (especially microsatellite data which accumulate rapidly in human biology and molecular ecology), and molecular sequence data based on various substitution models. In recent years, the availability of genomic data for a variety of species has resulted in the development of new types of distances derived from whole genomes for molecular phylogenetic reconstruction. This latter category includes genome BLAST distances (Auch et al. 2006; Deng et al. 2006; Henz et al. 2005), breakpoint distances based on genome rearrangement (Gramm and Niedermeier 2002; Herniou et al. 2001), distances based on the relative information between unaligned/unalignable sequences (Otu and Sayood 2003), distances based on the sharing of oligopeptides (Gao and Qi 2007), and composite distances incorporating several whole-genome similarity measures (Lin et al. 2009). Some of the whole-genome-based distances are necessary for constructing phylogenies of bacterial species because of three complications. The first is the rampant occurrence of horizontal gene transfer leading to difficulties in identifying orthologous genes. The second is that the leading strand and lagging strand in bacterial genomes typically have very different mutation patterns (Marin and Xia 2008), yet bacterial genes frequently switch between strands and consequently would experience different substitution patterns. The third is the frequent loss or gain of genomic DNA methylation affecting both genomic CpG dinucleotides and genomic GC content (Xia 2003). Both the second and third complications lead to heterogeneity in the evolutionary process even among orthologous gene lineages. All these genome-based distances mentioned above have been used in molecular phylogenetic reconstruction, but whether they are proportional to divergence, time has never been

studied. This hinders their use in molecular phylogenetics in general and dating speciation events or gene duplication events in particular (Xia 2009).

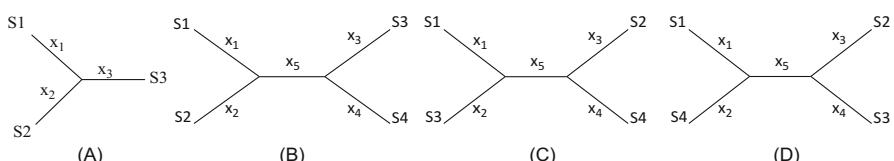
### 3 Distance-Based Phylogenetic Algorithms

As I mentioned before, distance-based phylogenetic reconstruction has two components, the first being the distance and the second being the algorithm of building trees from a distance matrix. Representatives of tree-building algorithms include UPGMA (Sneath 1962), the neighbor-joining (NJ) method (Saitou and Nei 1987), and FastME (Desper and Gascuel 2002). The UPGMA method is an average linkage clustering method and has already been numerically illustrated in a previous chapter on self-organizing maps and clustering algorithms in transcriptomic data analysis. Most frequently used distance-based tree-building algorithms are NJ and LS-based method with global optimization. I will numerically illustrate these methods in detail in this chapter. These algorithms have also been implemented in my program DAMBE (Xia 2013, 2017d).

#### 3.1 Least-Squares (LS) Methods and Minimum Evolution (ME) Criterion

Distance-based tree reconstruction involves two steps. The first is branch-length evaluation given a topology, which is typically done with least-squares (LS) method, i.e., finding branch lengths that will minimize the residual sum of squares (RSS). The second is to choose the best tree among all trees based on a criterion. The criterion could be minimum RSS or the shortest tree length. All these will become quite clear after numerical illustration below.

Let's start with a tree with only three OTUs (operational taxonomic units). There is only one unrooted tree with three OTUs (S1, S2, and S3 in Fig. 15.1a), so we only need to evaluate branch lengths by using the following equation:



**Fig. 15.1** Topologies for illustrating the distance-based methods. (A) An unrooted 3-OTU tree that does not require a least-squares (LS) method for branch evaluation. (B-D) three alternative unrooted topologies for four OTUs, where a LS method is needed for evaluating branch lengths

$$\begin{aligned} d_{12} &= x_1 + x_2 \\ d_{13} &= x_1 + x_3 \\ d_{23} &= x_2 + x_3. \end{aligned} \tag{15.7}$$

where  $d_{ij}$  is the distance between OTUs  $i$  and  $j$  and can be computed given a set of aligned sequences, and  $x_1$ ,  $x_2$ , and  $x_3$  are branch lengths (Fig. 15.1a). We have three equations with three unknowns, so  $x_1$ ,  $x_2$ , and  $x_3$  can be easily obtained by solving the simultaneous equations, i.e.,

$$x_1 = \frac{d_{12} + d_{13} - d_{23}}{2}; x_2 = \frac{d_{12} + d_{23} - d_{13}}{2}; x_3 = \frac{d_{13} + d_{23} - d_{12}}{2} \tag{15.8}$$

For the four-OTU tree in Fig. 15.1b, we can write down the equations in the same way as in Eq. (15.7), i.e.,

$$\begin{aligned} d_{12} &= x_1 + x_2 \\ d_{13} &= x_1 + x_5 + x_3 \\ d_{14} &= x_1 + x_5 + x_4 \\ d_{23} &= x_2 + x_5 + x_3 \\ d_{24} &= x_2 + x_5 + x_4 \\ d_{34} &= x_3 + x_4 \end{aligned} \tag{15.9}$$

where  $d_{ij}$  is the observed distance between OTUs  $i$  and  $j$  (i.e., distance computed from aligned sequences), and the right is the expected distances ( $d'_{ij}$ ) as the summation of branch lengths linking OTUs  $i$  and  $j$ .

Unfortunately, now we have six equations for five unknowns ( $x_1$ – $x_5$ ), so we will not be able to solve for  $x_1$ – $x_5$  exactly. The ordinary LS (OLS) method finds the  $x_i$  values that minimize the residual sum of squared deviations (RSS):

$$\text{RSS} = \sum \left( d_{ij} - d'_{ij} \right)^2 = [d_{12} - (x_1 + x_2)]^2 + \dots + [d_{34} - (x_3 + x_4)]^2. \tag{15.10}$$

Recall that OLS is a special case of weighted least-squares (WLS) method which is expressed as

$$\text{RSS} = \sum \frac{\left( d_{ij} - d'_{ij} \right)^2}{w_{ij}^m} \tag{15.11}$$

where  $w_{ij}$  is the weight, typically equal to either  $d_{ij}$  or  $d'_{ij}$ , and  $m$  is an integer typically equal to either 0, 1, or 2. When  $m = 0$ , we have OLS. When  $m = 1$  and  $w_{ij} = d'_{ij}$ , we have  $\text{RSS} = \chi^2$ . For numerical illustration, we will stay with OLS.

By taking the partial derivatives of RSS in Eq. (15.10) with respect to  $x_i$ , setting the derivatives to zero and solving the resulting simultaneous equations, we get

$$\begin{aligned}
x_1 &= d_{13}/4 + d_{12}/2 - d_{23}/4 + d_{14}/4 - d_{24}/4 \\
x_2 &= d_{12}/2 - d_{13}/4 + d_{23}/4 - d_{14}/4 + d_{24}/4 \\
x_3 &= d_{13}/4 + d_{23}/4 + d_{34}/2 - d_{14}/4 - d_{24}/4 \\
x_4 &= d_{14}/4 - d_{13}/4 - d_{23}/4 + d_{34}/2 + d_{24}/4 \\
x_5 &= -d_{12}/2 + d_{23}/4 - d_{34}/2 + d_{14}/4 + d_{24}/4 + d_{13}/4.
\end{aligned} \tag{15.12}$$

If we have  $d_{12} = 0.3$ ,  $d_{13} = 0.4$ ,  $d_{23} = 0.5$ ,  $d_{14} = 0.4$ ,  $d_{24} = 0.6$ , and  $d_{34} = 0.6$ , then  $x_1 = 0.075$ ,  $x_2 = 0.225$ ,  $x_3 = 0.275$ ,  $x_4 = 0.325$ , and  $x_5 = 0.025$ . The summation of all branch lengths is termed tree length (TL):

$$\text{TL} = \sum_{i=1}^{2n-3} x_i \tag{15.13}$$

where  $n$  is the number of OTUs. For our topology in Fig. 15.1, TL = 0.925. With four OTUs, there are three unrooted trees, so we need to repeat the above procedure to obtain  $x_1$  to  $x_5$  for the other two trees in Fig. 15.1c, d. We will designate TL for topologies in Fig. 15.1b–d as  $\text{TL}_B$ ,  $\text{TL}_C$ , and  $\text{TL}_D$ , respectively.

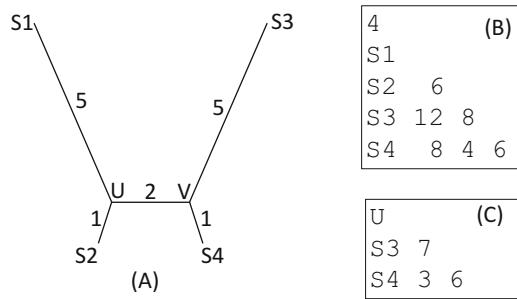
Once we have evaluated the branch lengths and computed  $\text{TL}_B$ ,  $\text{TL}_C$ , and  $\text{TL}_D$ , the minimum evolution (ME) criterion will choose the tree with the shortest TL as the best tree. This is analogous to the maximum parsimony criterion, i.e., the best tree is the one that requires the simplest explanation. The Fitch-Margoliash method uses the same criterion of minimum RSS for both branch length evaluation and tree selection. That is, the branch lengths are estimated as those that minimize RSS, and the tree associated with the smallest RSS is the best tree. As specified in Eq. (15.11), the Fitch-Margoliash method sets  $w_{ij} = d_{ij}$  and  $m = 2$ .

### 3.2 Neighbor-Joining (NJ) Method

Neighbor-joining (NJ) method and UPGMA are related. Recall that UPGMA will cluster two OTUs with the shortest distance in the distance matrix. In contrast, NJ takes into consideration two pieces of information: (1) the distance between the two OTUs and (2) how far these two OTUs are from other species. Two OTUs should be clustered if they have a short distance between them and jointly are far from the rest of the OTUs. This makes good sense.

Suppose we have a tree with four species (S1–S4) and labeled branch lengths shown in Fig. 15.2a. We cannot observe this tree, but we can estimate the distance between each species pair from aligned molecular sequences. Suppose we have estimated the distances shown in Fig. 15.2b. How would NJ infer the tree from the distance matrix?

The first step of NJ is to compute the “total distance” for species  $i$  ( $r_i$ ) as



**Fig. 15.2** An unobservable tree (a) with branch lengths labeled, together with an observable distance matrix (b), for illustrating neighbor-joining method that takes the matrix (b) to infer the tree (a). The matrix (b) is directly derived from the tree (a), which is equivalent to having perfectly accurate distance estimates. The two internal nodes are labeled U and V. The NJ method progressively reduces the distance matrix to smaller ones, e.g., from the  $4 \times 4$  matrix in (b) to  $3 \times 3$  matrix in (c)

$$r_i = \sum_{j=1, j \neq i}^N d_{ij} \quad (15.14)$$

For our four species,

$$\begin{aligned} r_1 &= d_{12} + d_{13} + d_{14} = 26 \\ r_2 &= d_{12} + d_{23} + d_{24} = 18 \\ r_3 &= d_{13} + d_{23} + d_{34} = 26 \\ r_4 &= d_{14} + d_{24} + d_{34} = 18 \end{aligned} \quad (15.15)$$

Now for each pair of species  $i$  and  $j$ , we calculate an index ( $M_{ij}$ ) that will help us decide which two species should be clustered:

$$M_{ij} = d_{ij} - \frac{r_i + r_j}{N - 2} \quad (15.16)$$

where  $N$  is the number of OTUs, i.e., 4.

For our four species with six species pairs,

$$\begin{aligned} M_{12} &= d_{12} - \frac{r_1 + r_2}{N - 2} = 6 - \frac{26 + 18}{2} = -16 \\ M_{13} &= M_{14} = M_{23} = M_{24} = -14, M_{34} = -16 \end{aligned} \quad (15.17)$$

Note that  $M_{12}$  or  $M_{34}$  and the other four  $M_{ij}$  values is 2 which means that if we cluster S1 and S2, or S3 and S4, then the branch length between the two ancestors (i.e., the two internal nodes) is 2. If the branch length between the two internal nodes is zero (i.e., a star tree), then all six  $M_{ij}$  values will be equal.

Now we choose the smallest  $M_{ij}$  and cluster species  $i$  and  $j$  together. Note that we would have UPGMA if  $M_{ij} = d_{ij}$  and would have wrongly clustered species 2 and 4 because  $d_{24} (= 4)$  is the smallest distance, leading to failure to recover the true tree in Fig. 15.2a. With NJ, we found  $M_{12}$  and  $M_{34}$  being the smallest, so we may cluster either species 1 and 2 together or species 3 and 4 together. This is a non-conflicting situation, and we may choose either  $M_{12}$  or  $M_{34}$ . There are conflicting situations, e.g., when  $M_{12}$  and  $M_{13}$  are both the smallest.

Suppose we have chosen  $M_{12}$  to cluster species 1 and 2, which has a common ancestor  $U$  (Fig. 15.2). We need to compute the branch lengths linking species 1 and  $U$  and species 2 and  $U$ :

$$\begin{aligned} b_{1U} &= \frac{d_{12}}{2} + \frac{r_1 - r_2}{2(N-2)} = 5 \\ b_{2U} &= d_{12} - b_{1U} = 1 = \frac{d_{12}}{2} + \frac{r_2 - r_1}{2(N-2)} \end{aligned} \quad (15.18)$$

Recall that UPGMA would have  $b_{1U} = d_{12}/2$ . In NJ, a species with a large  $r$  will have a longer branch to the ancestor than the sister species with a small  $r$ . This again makes intuitive sense. Note that the numerator is  $(r_1 - r_2)$  for computing  $b_{1U}$  but  $(r_2 - r_1)$  for computing  $b_{2U}$ .

Now that we have node  $U$  representing S1 and S2, we need to compute the distances between  $U$  and S3 and between  $U$  and S4:

$$\begin{aligned} d_{U3} &= \frac{d_{13} + d_{23} - d_{12}}{2} = 7 \\ d_{U4} &= \frac{d_{14} + d_{24} - d_{12}}{2} = 3 \end{aligned} \quad (15.19)$$

which should be obvious and should not need any explanation. These two distances, together with the original  $d_{34}$ , are shown in Fig. 15.2c where S1 and S2 have been replaced by  $U$ .

Now we have only three OTUs ( $U$ , S3, and S4) with their pairwise distances shown in Fig. 15.2c. We repeat the process by computing  $r_U$ ,  $r_3$ , and  $r_4$  as well as  $M_{U3}$ ,  $M_{U4}$ , and  $M_{34}$ :

$$\begin{aligned} r_U &= d_{U3} + d_{U4} = 10, \quad r_3 = 13, \quad r_4 = 9 \\ M_{U3} &= d_{U3} - \frac{r_U + r_3}{N-2} = -16 = M_{U4} = M_{34} \end{aligned} \quad (15.20)$$

where  $N$  again is the number of OTUs but now reduced from 4 to 3.

pt?>Note that  $r_3$  and  $r_4$  in Eq. (15.20) are different from those in Eq. (15.15). Also note that, with three OTUs ( $N = 3$ ), we will always have equal  $M_{ij}$  values which are equal to

$$M_{12} = M_{13} = M_{23} = -(d_{12} + d_{13} + d_{23}) \quad (15.21)$$

This is because we have only one unrooted tree with three OTUs, and we will always get to this same tree no matter which two OTUs are clustered.

Now we need to compute the branch lengths from  $S_3$  and  $S_4$  to their common ancestor (internal node  $V$  in Fig. 15.2a):

$$\begin{aligned} b_{UV} &= \frac{d_{U3}}{2} + \frac{r_U - r_3}{2(N-2)} = \frac{7}{2} + \frac{10-13}{2(3-2)} = 2 = \frac{d_{U4}}{2} + \frac{r_U - r_4}{2(N-2)} \\ b_{3V} &= d_{U3} - b_{UV} = 5 = \frac{d_{U3}}{2} + \frac{r_3 - r_U}{2(N-2)} = \frac{d_{34}}{2} + \frac{r_3 - r_4}{2(N-2)} \\ b_{4V} &= d_{34} - b_{3V} = 1 = \frac{d_{U4}}{2} + \frac{r_4 - r_U}{2(N-2)} = \frac{d_{34}}{2} + \frac{r_4 - r_3}{2(N-2)} \end{aligned} \quad (15.22)$$

where  $N = 3$ . Now we have fully reconstructed the tree in Fig. 15.2a from the distance matrix in Fig. 15.2b. We can apply the LS method to the distance matrix in Fig. 15.2b and obtain exactly the same result. However, as we have mentioned before, UPGMA will fail to reconstruct the tree because it will start by cluster  $S_2$  and  $S_4$  together because  $d_{24}$  ( $= 4$ ) is the smallest of all pairwise distances.

You might be wondering why, in computing  $M_{ij}$  in Eqs. (15.17) and (15.20), the denominator in the second term is  $(N - 2)$ . Why not  $(N - 1)$  or  $(N - 3)$ ? Let's focus on the three-OTU case and pretend that we do not know if the denominator should be  $(N - 1)$ ,  $(N - 2)$ , or  $(N - 3)$ . We will just use  $(N - x)$  and then infer value of  $x$ .

We now have expressions of  $M_{ij}$  as

$$M_{12} = d_{12} - \frac{r_1 + r_2}{N-x}; \quad M_{13} = d_{13} - \frac{r_1 + r_3}{N-x}; \quad M_{23} = d_{23} - \frac{r_2 + r_3}{N-x} \quad (15.23)$$

We know that there is only one tree with three OTUs, which implies that clustering  $S_1$  and  $S_2$ , or  $S_1$  and  $S_3$ , or  $S_2$  and  $S_3$  together will all arrive at the same tree. This implies that  $M_{12} = M_{13} = M_{23}$ , which is satisfied only  $x = 2$ .

We can also use the three-OTU case to explain why, in computing branch lengths in Eq. (15.18), we have  $(N - 2)$  in the denominator of the second term. Take the three-OTU tree in Fig. 15.1a, for example. We have already obtained  $x_1$ ,  $x_2$ , and  $x_3$  in Eq. (15.8). Suppose we now need to compute  $x_1$  using NJ and not sure if the denominator should have  $(N - 1)$ ,  $(N - 2)$ , or  $(N - 3)$ . So we just use  $(N - x)$ :

$$x_1 = \frac{d_{12}}{2} + \frac{r_1 - r_2}{2(3-x)} \quad (15.24)$$

We can set this  $x_1$  in Eq. (15.24) equal to the  $x_1$  in Eq. (15.8) and solve for  $x$ . This leads to  $x = 2$ .

## 4 Dating Speciation and Gene Duplication Events

Although the molecular clock concept was proposed on the basis of evolutionary distances (Zuckerkandl and Pauling 1965), and the conceptual framework for distance-based dating has been in place for many years (Chakraborty 1977; Drummond and Rodrigo 2000; Takezaki et al. 1995), it is only recently (Xia 2013, 2017d; Xia and Yang 2011) that statistical methods and software for dating speciation and gene duplication events by using evolutionary distances have become available. This is partly due to the existence of more sophisticated methods based on the likelihood and Bayesian inference, but there are still cases where distance-based methods represent a good contender in dating, just as there are still cases where distance-based methods are good choice in phylogenetic reconstruction. This section outlines the rational of the distance-based dating, numerically illustrated with both fossil-calibrated internal nodes for conventional dating or with tip-dating (Drummond et al. 2003a, b).

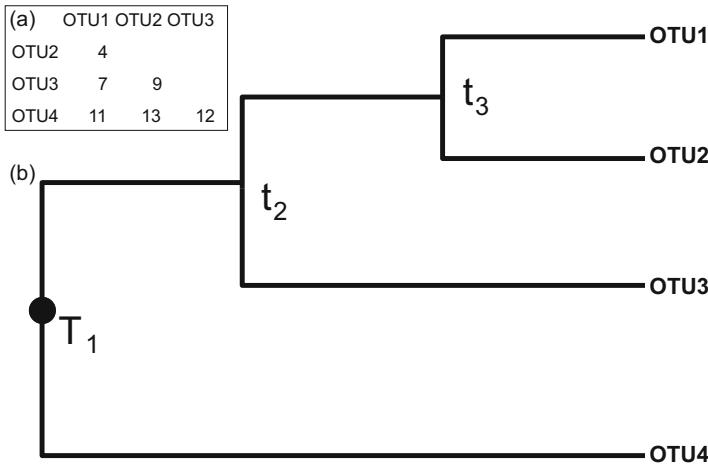
I will first detail the approach involving a single gene with one or more calibration points and with global and two versions of local clocks. This is followed by approaches for dating with multiple genes and estimating the variation of the divergence time by resampling methods such as bootstrapping and jackknifing (Felsenstein 2004, pp. 335–363). The accuracy of the distance-based dating method is illustrated by applying it to dating with two data sets, one with seven great ape species (Rannala and Yang 2007) and the other with 35 mammalian species including major mammalian lineages (Yang and Yoder 2003).

### 4.1 *The Least-Squares Method for Conventional Internal Node-Calibrated Dating*

The statistical framework of the least-squares dating method has been presented independently in matrix form twice before (Chakraborty 1977; Drummond and Rodrigo 2000). Here we illustrate the mathematical rationale as well as the extensions including multiple calibration points and tip-dating, two versions of local clocks, and the computation of confidence limits by resampling methods. The method is implemented in DAMBE (Xia 2013, 2017d; Xia and Yang 2011).

#### 4.1.1 **Dating with One Calibration Point**

Given the evolutionary distances  $d_{ij}$  and the topology in Fig. 15.3, with the time to the root known as  $T_1$ , we need to estimate  $t_2$ ,  $t_3$ , and  $r$  (the substitution rate). Assuming a global clock, we minimize the following residual sum of squares (RSS) based on the ordinary least-squares method:



**Fig. 15.3** A distance matrix (a) and a rooted tree (b) with four OTUs (numbered 1–4) for illustrating the distance-based least-squares method for dating speciation events.  $T_1$  is known and used to calibrate the molecular clock, and  $t_2$  and  $t_3$ , as well as the substitution rate  $r$ , are to be estimated

$$\text{RSS} = (d_{12} - 2rt_3)^2 + (d_{13} - 2rt_2)^2 + (d_{23} - 2rt_2)^2 + \dots + (d_{34} - 2rT_1)^2 \quad (15.25)$$

Note that Eq. (15.25) is expressed as squared residuals between the observed distance ( $d_{ij}$ ) and the expected distance ( $2rt_i$ ). One can also express it as squared residuals in time such as

$$\text{RSS} = \left( \frac{d_{12}}{r} - 2t_3 \right)^2 + \left( \frac{d_{13}}{r} - 2t_2 \right)^2 + \left( \frac{d_{23}}{r} - 2t_2 \right)^2 + \dots + \left( \frac{d_{34}}{r} - 2T_1 \right)^2 \quad (15.26)$$

To derive formulae for  $t_2$ ,  $t_3$ , and  $r$ , we take the partial derivatives of RSS with respect to  $r$ ,  $t_2$ , and  $t_3$ , equate the three partial derivatives to zero, and solve the three resulting simultaneous equations. This will give us

$$\begin{aligned} r &= \frac{d_{14} + d_{24} + d_{34}}{6T_1} \\ t_2 &= \frac{3(d_{13} + d_{23})T_1}{2(d_{14} + d_{24} + d_{34})} = \frac{d_{13} + d_{23}}{4r} \\ t_3 &= \frac{3d_{12}T_1}{d_{14} + d_{24} + d_{34}} = \frac{d_{12}}{2r} \end{aligned} \quad (15.27)$$

In general, when there is only one calibration point ( $T$ ) for an internal node, then  $r$  is expressed as

$$r = \frac{\sum_{i=1}^n \sum_{j=1}^m d_{ij}}{2nmT} \quad (15.28)$$

where  $n$  is the number of children in one descendent clade of the node with calibration time  $T$ ,  $m$  is the number of children in the other descendent clade of the node, and  $d_{ij}$  is the evolutionary distance from  $i$ th leaf in one descendent clade to  $j$ th leaf in the other descendent clade. In the four-OTU case with  $T_1$  known (Fig. 15.3),  $n = 1$  (OTU 4) and  $m = 3$  (OTUs 1, 2, and 3), and  $d_{ij}$  values are  $d_{14}$ ,  $d_{24}$ , and  $d_{34}$ .

#### 4.1.2 Dating with Multiple Calibration Points

With multiple calibration points, the method will be essentially the same except that we have fewer parameters to estimate. For example, if both  $T_1$  and  $T_3$  are known, then we only need to estimate  $r$  and  $t_2$ , which are

$$\begin{aligned} r &= \frac{T_3 d_{12} + T_1 d_{14} + T_1 d_{24} + T_1 d_{34}}{2(T_3^2 + 3T_1^2)} \\ t_2 &= \frac{(d_{13} + d_{23})(T_3^2 + 3T_1^2)}{2(T_3 d_{12} + T_1 d_{14} + T_1 d_{24} + T_1 d_{34})} = \frac{d_{13} + d_{23}}{4r} \end{aligned} \quad (15.29)$$

When  $N_c$  calibration points are available, then the LS estimate of  $r$  is

$$r = \frac{\sum_{k=1}^{N_c} T_k \sum_{i=1}^{n_k} \sum_{j=1}^{m_k} d_{ijk}}{2 \sum_{k=1}^{N_c} n_k m_k T_k^2} \quad (15.30)$$

For example, with the tree in Fig. 15.3, but with both  $T_1$  and  $T_3$  known, then  $r$  is

$$r = \frac{T_1(d_{14} + d_{24} + d_{34}) + T_3 d_{12}}{2(3T_1^2 + T_3^2)} \quad (15.31)$$

The method above with multiple calibration points provides the flexibility for the user to further optimize the time estimates. This is done with three steps. The first is to construct a tree with an imposed clock and the LS criterion, without reference to the calibration time. This results in a set of internal nodes with estimated path lengths ( $D$ ) to descendent leaves. The second step is to minimize the following residual sum of squares (RSS) after constructing a tree with an imposed clock:

$$\text{RSS} = \sum_{i=1}^{N_c} (D_i - rT_i)^2 \quad (15.32)$$

where  $N_c$  is the number of nodes having calibration time  $T_1, T_2, \dots, T_{N_c}$  and  $D_i$  is the distance from the node with calibration time  $T_i$  to the tip, i.e., the path length from the node with calibration time  $T_i$  to a descendent leaf (Note that the node has equal path length to any of its descendent leaves when a global clock is assumed). Solving for  $r$  leads to

$$r = \frac{\sum_{i=1}^{N_c} D_i T_i}{\sum_{i=1}^{N_c} T_i^2} \quad (15.33)$$

The third, and final, step is to rescale all  $D_i$  values by  $r$ , i.e., converting  $D_i$  to divergence time. This rescaling includes the nodes with calibration time  $T_i$ . The approach of rescaling fossil dates by the LS criterion is termed soft-calibrated dating.

Note that  $r$  is an unbiased estimate of the true evolutionary rate ( $\gamma$ ) only when  $T_i$  is an unbiased estimate of the true divergence time  $\tau_i$  and  $D_i$  is an unbiased estimate of  $\gamma\tau_i$ . While  $D_i$  could arguably be an unbiased estimate of  $\gamma\tau_i$  for molecular sequence data when the substitution model is correct and stochastic effect negligible,  $T_i$  is typically an underestimate of  $\tau_i$ , i.e.,  $T_i = \tau_i - \varepsilon_{\text{fossil},i}$ , where  $\varepsilon_{\text{fossil}}$  is the bias in the fossil date. This implies that the estimated  $r$  is typically an overestimate of  $\gamma$ , with the bias (designated by  $h_{\text{fossil}}$ ) being

$$h_{\text{fossil}} = \frac{\gamma - r}{\gamma} = -\frac{\sum_{i=1}^{N_c} \varepsilon_{\text{fossil},i} T_i}{\sum_{i=1}^{N_c} T_i^2} \quad (15.34)$$

When  $D_i$  is also uncertain, e.g., due to limited data or due to substitution saturation in molecular sequences (which typically leads to  $D_i$  underestimating  $\gamma\tau_i$ ), we have  $D_i = \gamma\tau_i - \varepsilon_{\text{data},i}$ , and the bias in the estimated  $r$ , designated by  $h_{\text{fossil+data}}$ , becomes

$$h_{\text{fossil+data}} = \frac{\gamma - r}{\gamma} = \frac{\sum_{i=1}^{N_c} \varepsilon_{\text{data},i} T_i - \gamma \sum_{i=1}^{N_c} \varepsilon_{\text{fossil},i} T_i}{\gamma \sum_{i=1}^{N_c} T_i^2} \quad (15.35)$$

Equation (15.35) shows clearly that the estimated  $r$  (as well as the estimated divergence time) contains two sources of uncertainty, one due to  $\varepsilon_{\text{fossil}}$  and  $\varepsilon_{\text{data}}$ . These two sources of uncertainty have not been distinguished in published papers on dating. It is important to keep in mind that, while unlimited amount of good sequence data for estimating  $D_i$  can reduce  $\varepsilon_{\text{data}}$  to 0, no amount of sequence data can reduce  $\varepsilon_{\text{fossil}}$ .

### 4.1.3 Dating with Multiple Genes with One or More Calibration Points

The method can be easily extended to perform a combined analysis with multiple distance matrices, e.g., when there are two or more genes, when each distance is obtained from each of the three codon positions in a protein-coding gene, or when one has one distance matrix from sequence data and another from DNA hybridization data. With two genes A and B and two corresponding distance matrices whose individual elements are represented by  $d_{A,ij}$  and  $d_{B,ij}$ , respectively, we can perform a combined analysis to estimate jointly  $t_2$ ,  $t_3$ ,  $r_A$ , and  $r_B$  (where  $r_A$  and  $r_B$  are the substitution rate for genes A and B, respectively) in two steps. First, we obtain  $k = r_B/r_A$  by using a simple linear regression with the regression model  $d_B = k \cdot d_A$  (i.e., forcing the intercept to 0). Second, we rewrite Eq. (15.25) as follows:

$$\begin{aligned} \text{RSS}_1 &= (d_{A,12} - 2r_A t_3)^2 + (d_{A,13} - 2r_A t_2)^2 + \cdots + (d_{A,34} - 2r_A T_1)^2 \\ \text{RSS}_2 &= (d_{B,12} - 2kr_A t_3)^2 + (d_{B,13} - 2kr_A t_2)^2 + \cdots + (d_{B,34} - 2kr_A T_1)^2 \\ \text{RSS} &= \text{RSS}_1 + \text{RSS}_2 \end{aligned} \quad (15.36)$$

Now  $r_A$ ,  $t_2$ , and  $t_3$  can be estimated just as before and  $r_B$  can be estimated as  $k \cdot r_A$ . With the topology in Fig. 15.3, the LS solutions for the unknowns are

$$\begin{aligned} r_A &= \frac{C}{6T_1(1+k^2)} \\ t_2 &= \frac{3(d_{A,13} + d_{A,23} + kd_{B,13} + kd_{B,23})T_1}{2C} \\ t_3 &= \frac{3(d_{A,12} + kd_{B,12})T_1}{C} \\ C &= d_{A,14} + d_{A,24} + d_{A,34} + kd_{B,14} + kd_{B,24} + kd_{B,34} \end{aligned} \quad (15.37)$$

If the two genes evolve at the same rate so that  $d_{A,ij} = d_{B,ij}$  and  $k = 1$ , then  $r_A$ ,  $t_2$ , and  $t_3$  are reduced to the same expressions as those in Eq. (15.27).

One potential problem with this approach is that if  $k > 1$  (i.e., when gene B evolves much faster than gene A), the estimation will depend on  $d_{B,ij}$  much more than  $d_{A,ij}$ . Similarly, if  $k < 1$ , the estimation will depend on  $d_{A,ij}$  much more than  $d_{B,ij}$ . For example, the third codon position evolves much faster than codon positions 1 and 2. Applying Eq. (15.37) will result in estimates dominated by the distance matrix from the third codon position.

An alternative approach is to first scale  $\text{RSS}_2$  in Eq. (15.36) by dividing values within each parenthesis in  $\text{RSS}_2$  by  $k$ , so Eq. (15.36) becomes

$$\begin{aligned} \text{RSS}_1 &= (d_{A,12} - 2rt_3)^2 + (d_{A,13} - 2rt_2)^2 + \cdots + (d_{A,34} - 2rT_1)^2 \\ \text{RSS}_2 &= (d_{B,12}/k - 2rt_3)^2 + (d_{B,13}/k - 2rt_2)^2 + \cdots + (d_{B,34}/k - 2rT_1)^2 \\ \text{RSS} &= \text{RSS}_1 + \text{RSS}_2 \end{aligned} \quad (15.38)$$

Neither this scaled approach nor the unscaled approach specified in Eq. (15.36) is well-reasoned. A better approach is to see which gene or which site partition conforms to the molecular clock better than others and then weigh them accordingly. Two approaches can be used, preferably jointly, in evaluating the suitability of genes for dating. The first is by testing the clock hypothesis based on distances (Xia 2009), and the second, when multiple high-quality calibration points are available, is to test which gene or site partition are consistent with the calibration points. The third codon position, less constrained by natural selection, should provide better estimates of evolutionary time as long as substitution saturation (Xia and Lemey 2009; Xia et al. 2003b) is not an issue. In addition, the third codon position exhibits little heterogeneity in substitution rate over sites relative to the first and second codon positions, which is a highly desirable property in molecular phylogenetic reconstruction (Xia 1998b). In other words, the third codon position may deserve a greater weight than codon positions 1 and 2 for dating evolutionary events and should not be scaled to have the same weight as codon positions 1 and 2. On the other hand, if the third codon position have all experienced multiple substitution, then it will have little information left for dating. An ideal case is when distances estimated from the three codon positions are all highly positively and linearly correlated.

In general, for the same period of evolutionary time, the fast-evolving gene (i.e., the one generating large pairwise distances) is expected to conform to neutral evolution better than a slow-evolving gene subject to functional constraints. For this reason, the unscaled method seems more justifiable. Following this reasoning, we can perform a simple combined analysis involving  $N_g$  genes by generating a new distance matrix with  $d_{ij}$  computed as a weighted average:

$$d_{ij} = \frac{\sum_{k=1}^{N_g} d_{ijk} \bar{d}_k}{\sum_{k=1}^{N_g} \bar{d}_k} \quad (15.39)$$

where  $\bar{d}_k$  is the mean of all the pairwise distances from gene k.

#### 4.1.4 Dating with Local Clocks

Molecular sequence data violating a global clock have long been known (Britten 1986; Li and Tanimura 1987; Li et al. 1987; Li and Wu 1987; Wu and Li 1985), and it is rare for a large tree to have a global clock operating along all lineages (Pereira and Baker 2006; Smith et al. 2006; Tinn and Oakley 2008). Relatively fast evolving lineages will have overestimated divergence times if a global clock is imposed. Although protocols are available to eliminate offending lineages that do not conform to the global clock (Rambaut and Bromham 1998; Takezaki et al. 1995) and to generate linearized trees, such treatments lead to inefficient use of data and are

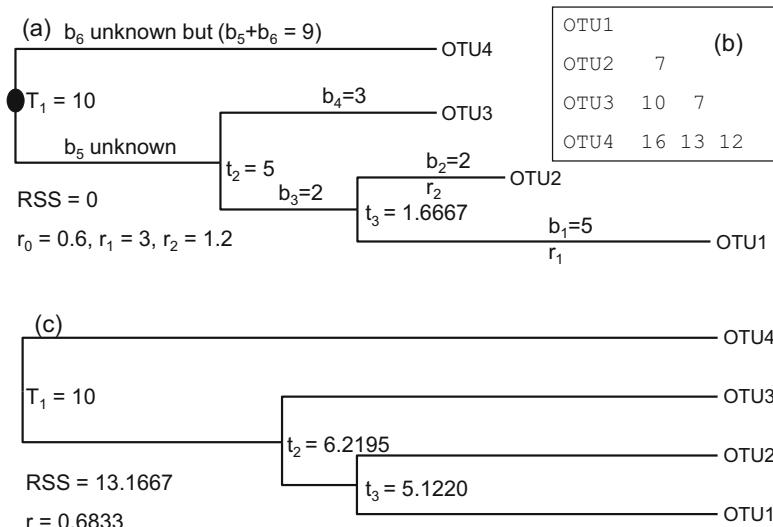
practical only when the majority of the lineages conform to the global clock. For this reason, local clocks are often necessary for practical dating.

There are two general approaches for local clock dating. The first is when specific lineages are *a priori* known to evolve differently from others and can therefore be explicitly modeled. Several approaches have been proposed to solve this local clock dating problem (Kishino and Hasegawa 1990; Yoder and Yang 2000).

The second approach to local clock dating is the rate smoothing pioneered by Sanderson (1997), based on the inference that the evolution rate is autocorrelated along lineages (Gillespie 1991). This constraint of rate autocorrelation will penalize dramatic changes in evolutionary rate along lineages. Thus, for rapidly evolving lineages, this approach will result in a divergence time smaller than that from the global clock approach but larger than that from the first approach without the constraint of rate autocorrelation. We will illustrate both approaches for comparative purposes.

#### 4.1.4.1 Local Clock with Lineages Known *A Priori* to Evolve Differently

Suppose we have four lineages with very different evolutionary rates (Fig. 15.4a), with the lineages leading to OTU1 and OTU2 expected *a priori* to evolve at different rates from lineages leading to OTU3 and OTU4. So we will assign a background rate to OTU3 and OTU4 and different rates for OTU1 and OTU2. Note that, although we



**Fig. 15.4** A four-OTU tree with lineage-specific evolutionary rates (a). The branch lengths are indicated on the branch, together with the distance matrix with each distance being the path length from OTU  $i$  to OTU  $j$  (b). In practice, branch lengths are unknown and the distance matrix needs to be estimated from the data.  $T_1$  is the calibration point, and  $t_2$  and  $t_3$  are dated with either three rates ( $r_0$ ,  $r_1$ , and  $r_2$ ), i.e., a local clock model (a) or a single rate  $r$ , i.e., a global clock (c)

labeled branch lengths ( $b_i$ ) on the tree, in practice both branch lengths and pairwise distances are unknown and need to be estimated from the data. Thus, the input for the local clock dating is a distance matrix, a topology, and a specification of which lineages have different rates. One may ask how to know which lineages have different rates. A simple approach is to construct an unrooted tree with a known outgroup and inspect the branch lengths. If some branches are unduly long relative to their sister lineages, then we may allow them to have an evolutionary rate different from the background rate. We will then test whether these additional rates improve the fit of the model to the data significantly relative to the model with fewer rates or just one rate.

Let's designate evolutionary rate from  $t_3$  to OTU1 as  $r_1$  and from  $t_3$  to OTU2 as  $r_2$ . The rest of the lineages are assumed to evolve at the rate  $r_0$ . Given the evolutionary distances (Fig. 15.4b) and calibration time  $T_1$  (Fig. 15.4a), we can obtain  $r_0$ ,  $r_1$ , and  $r_2$  as well as  $t_2$  and  $t_3$  by minimizing the following RSS:

$$\begin{aligned} \text{RSS} = & (d_{12} - r_1 t_3 - r_2 t_3)^2 + (d_{13} - r_1 t_3 - r_0(t_2 - t_3) - r_0 t_2)^2 \\ & + (d_{14} - r_1 t_3 - r_0(t_2 - t_3) - r_0(T_1 - t_2) - r_0 T_1)^2 \\ & + (d_{23} - r_2 t_3 - r_0(t_2 - t_3) - r_0 t_2)^2 \\ & + (d_{24} - r_2 t_3 - r_0(t_2 - t_3) - r_0(T_1 - t_2) - r_0 T_1)^2 \\ & + (d_{34} - r_0 t_2 - r_0(T_1 - t_2) - r_0 T_1)^2 \end{aligned} \quad (15.40)$$

Note that the local clock model specified in Eq. (15.40) is reduced to the global clock model specified in Eq. (15.25) when  $r_0 = r_1 = r_2$ . Thus, the global clock model is a special case of the local clock model. Such models are termed nested models. In the likelihood framework, such models are typically tested either by the likelihood ratio test or by information-theoretic indices such as AIC and BIC.

Taking partial derivatives of RSS in Eq. (15.40) with respect to  $r_0$ ,  $r_1$ ,  $r_2$ ,  $t_2$ , and  $t_3$ , setting the derivatives to 0, and solving the resulting simultaneous equations, we obtain

$$\begin{aligned} r_0 &= \frac{d_{34}}{2T_1} \\ r_1 &= \frac{(2d_{12} + d_{13} + d_{14} - d_{23} - d_{24})d_{34}}{A} \\ r_2 &= \frac{(2d_{12} + d_{23} + d_{24} - d_{13} - d_{14})d_{34}}{A} \\ t_2 &= \frac{(d_{13} + 2d_{34} + d_{23} - d_{14} - d_{24})T_1}{2d_{34}} \\ t_3 &= \frac{(d_{12} + 2d_{34} - d_{14} - d_{24})T_1}{d_{34}} \\ A &= 4(d_{12} + 2d_{34} - d_{14} - d_{24})T_1 \end{aligned} \quad (15.41)$$

With the actual  $d_{ij}$  values in Fig. 15.4b, we have  $r_0 = 0.6$ ,  $r_1 = 3$ ,  $r_2 = 1.2$ ,  $t_2 = 5$ , and  $t_3 = 1.6667$ . Because the  $d_{ij}$  values we used are the actual path lengths from the

branch lengths shown in Fig. 15.4a, i.e.,  $d_{ij}$  values are accurate, the resulting RSS is 0, i.e., the fit of the distance matrix to the tree is perfect.

In contrast, if we assume a single evolutionary rate (i.e., a global clock), then we will have  $r = 0.6833$ ,  $t_2 = 6.2195$ ,  $t_3 = 5.2122$ , and RSS = 13.1667 (Fig. 15.4c). In other words, the increased evolutionary rates along lineages leading to OTU1 and OTU2 resulted in poor fit of the distance matrix to the tree (i.e., a larger RSS) and the inflated estimates of  $t_2$  and  $t_3$  (especially  $t_3$  due to the much faster evolutionary rate along the lineage leading to OTU1). Whether the two parameters in the local clock model (i.e.,  $r_1$  and  $r_2$ ) justify the decrease in RSS can be tested in the framework of model selection based on differences in RSS and the number of parameters (Xia 2009), given that the rate differences are expected a priori.

#### 4.1.4.2 The Rate-Smoothing Approach for Local Clock Dating

The rate-smoothing approach (Sanderson 1997) involves two steps. The first is to evaluate the tree to obtain the branch lengths, which can be done either by distance-based or maximum likelihood methods. The second is to use the estimated branches to estimate divergence time with the constraint of rate autocorrelation.

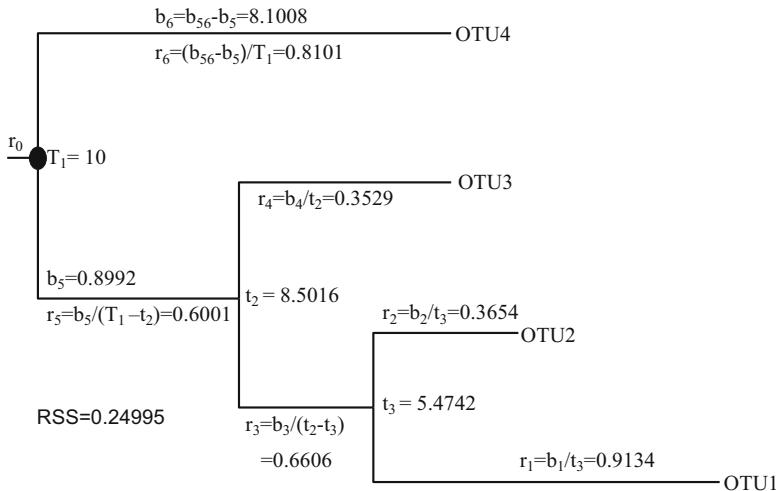
With the distance matrix in Fig. 15.4b, the branch lengths ( $b_i$ ) can be evaluated by either neighbor-joining or FastME and are shown in Fig. 15.4a. Branch lengths  $b_5$  and  $b_6$  cannot be separately evaluated by the distance-based methods without assuming a molecular clock, and consequently only their summation, designated by  $b_{5+6}$  ( $= b_5 + b_6$ ), is shown.

The second step in the rate-smoothing approach is to estimate local rates, which are  $r_1$  ( $= b_1/t_3$ ),  $r_2$  ( $= b_2/t_3$ ),  $r_3$  [ $= b_3/(t_2-t_3)$ ], and so on (Fig. 15.5). The method of rate-smoothing is to obtain  $t_2$  and  $t_3$  (with  $T_1$  as the calibration time) as well as  $b_5$  that minimize the following sum of squares:

$$\begin{aligned} \text{RSS} = & (r_1 - r_3)^2 + (r_2 - r_3)^2 + (r_3 - r_5)^2 + (r_4 - r_5)^2 \\ & + (r_5 - r_0)^2 + (r_6 - r_0)^2 \text{ where } r_0 = \frac{b_{5+6}}{2T_1 - t_2}; \\ & t_2 < T_1; t_3 < t_2 \end{aligned} \quad (15.42)$$

The minimization results in  $t_3 = 5.4742$ ,  $t_2 = 8.5016$ , and  $b_5 = 0.8992$  (which leads to  $b_6 = b_{5+6} - b_5 = 8.1008$ ), with minimized SS equal to 0.2500. All local rates were shown in Fig. 15.5. Note that RSS in Eq. (15.42) is not comparable to RSS in other equations.

The rate-smoothing approach implies that evolutionary rate of the ancestral lineage will always be between the evolutionary rates of the child lineages, which reminds us of the dating approaches assuming a Brownian motion model. Theoretically, there is no strong reason to believe that the two child lineage cannot both evolve faster than the ancestral lineage. However, with no external information available, the best guess of the evolutionary rate of the ancestral lineage should be some sort of average of the evolutionary rates of child lineages.



**Fig. 15.5** The estimated rates and divergence times from the rate-smoothing approach for local clock dating.  $T_1$  is the calibration point

The application of Eq. (15.42) requires  $T_1$  to be fixed because, if  $T_1$  is bounded with a minimum and maximum, then RSS will always be the smallest when  $T_1$  equals the maximum. Specifically, when  $T_1$  is increased  $n$  times, RSS will decrease by  $n^2$  times. This suggests that a modified version of Eq. (15.42), i.e.,  $RSS_m = RSS * T_1^2$ , might allow  $T_1$  to be bounded. Unfortunately, such a modification only makes the model unidentifiable because  $RSS_m$  can be identical for any  $T_1$ , i.e., if we obtain  $RSS_m$  and rates  $r_i$  with  $T_1 = T$ , we can obtain the same  $RSS_m$  (but rates  $r_i/n$ ) with  $T_1 = n \cdot T$ , where  $n$  is any positive real number.

It has been proposed that the rate-smoothing approach can incorporate fossil uncertainty by using the fossil date as a minimum age (Sanderson 1997). For the reason in the previous paragraph, this proposed approach is impossible. Molecular sequences can be used to estimate branch lengths but not time and rate separately. If we increase the calibration time 10 times, then all estimated node times will be 10 times greater, and the resulting rates will simply be 10 times smaller. This is the problem shared by all dating approaches, including the likelihood and the Bayesian (Yang 2006). Multiplying the calibration time by  $n$  and simultaneously dividing rate by  $n$  will not change the likelihood (or  $RSS_m$  in the least-squares approach). This invariance of likelihood with respect to calibration time due to the confounding of time and rate also causes problems in Bayesian inference. Because the joint probability in the Bayesian inference is the product of the prior and the likelihood, the invariance of likelihood means that the prior for the calibration time used in Bayesian dating will not be modified by the sequence data and will essentially be regurgitated in the posterior in an undigested form.

With the least-squares approach, we can also replace the calibration time  $T$  by a distribution (the equivalent of a Bayesian prior), e.g., a normal or exponential

distribution with mean  $T$ , and repeatedly sample from this distribution to obtain a set of estimated divergence times so that each internal node will have, instead of a single estimate of divergence time, a set of estimated divergence times that form a distribution. This “posterior” will have the same shape as the “prior” and does not lead to better inference. This criticism is also applicable to the Bayesian approach that sets a prior on the calibration time.

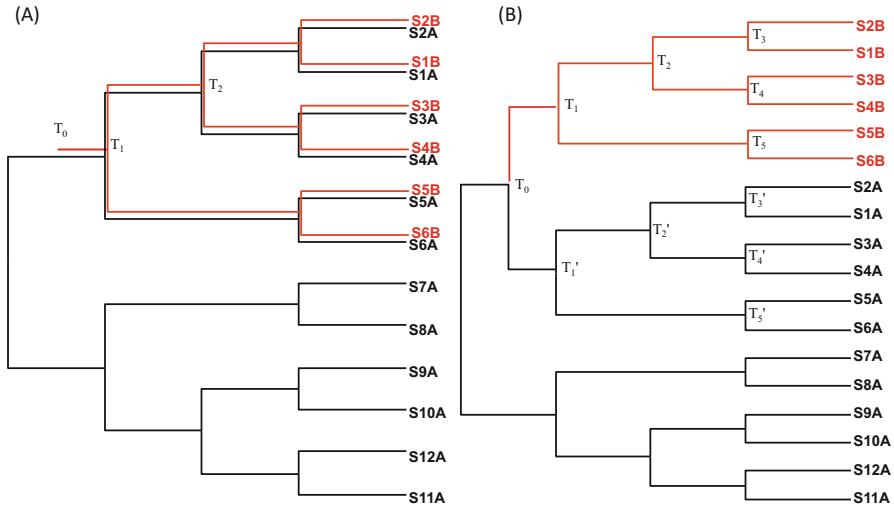
One main problem with the constraint of rate autocorrelation is whether the rate autocorrelation assumption is valid. If the assumption is false, then much estimation error will be introduced. For example, if one terminal lineage evolves very rapidly leading to a long branch length ( $b$ ), then the only way to minimize RSS in the rate-smoothing approach is to increase the associated  $t$  because  $r = b/t$ . This implies that all ancestral nodes (parent, grandparent, etc.) of this lineage will tend to have overestimated divergence times. This problem is quite obvious when we contrast estimates in Fig. 15.2a (with no constraint of rate autocorrelation, and  $r_1 = 3$ ) and Fig. 15.3 (with the constraint of rate autocorrelation, and  $r_1 = 0.9134$ ). Constraining  $r_1$  to a small value necessitates a much larger  $t_3$  ( $= 5.4742$ ) in Fig. 15.3 relative to a much smaller  $t_3$  ( $= 1.6667$ ) in Fig. 15.2a.

#### 4.1.5 Notes on Dating Gene Duplication Events

Gene duplication and subsequent subfunctionalization (i.e., the function of the parental gene is divided between the duplicated genes) and neofunctionalization (i.e., the duplicated gene acquires a new function) are important evolutionary events (Vlasschaert et al. 2015). Phylogenetic relationship and timing of gene duplication can help understand how duplicated genes evolve under which constraints (Vlasschaert et al. 2017). I wish to outline here a few caveats in dating gene duplication events as well as some special relationships among between duplicated lineages that we can take advantage in following evolutionary trajectories of duplicated genes.

Suppose we have 12 species (labeled S1–S12) related by the phylogeny in Fig. 15.6a with gene A present in each species. A gene duplication event occurred at time  $T_0$  (Fig. 15.6), giving rise to paralogous gene B represented in species S1–S6 (Fig. 15.6a). While estimating  $T_0$  is important, we will leave it aside momentarily and focus on another research problem. That is, which of the paralogous lineages (A lineage or B lineage) evolve faster. We can estimate  $T_1$  to  $T_5$  by using gene B sequences in species S1 to S6 and  $T_1'$  to  $T_5'$  by using gene A sequences in species S1 and S6. As depicted in Fig. 15.6b, we expect  $T_i = T_i'$ . Deviation from this expectation may be associated with the functional constraints in these two gene lineages.

Now come back to  $T_0$  in Fig. 15.6. We can estimate  $T_0$  if (1) all duplicated gene lineages are preserved, and (2) the duplicated genes have evolved in conformation to the molecular clock hypothesis. If these two assumptions are met, then we just need to obtain gene sequences from S1A to S12A and S1B to S6B, build a tree, and then date  $T_0$  (Fig. 15.6b). Unfortunately, duplicated genes typically do not evolve in a clocklike manner, so we cannot use these gene sequences to date the duplication



**Fig. 15.6** A gene duplication event occurred at time  $T_0$  (A) with an alternative view in (B). We now have two copies (labeled A and B) for genes S1 to S6

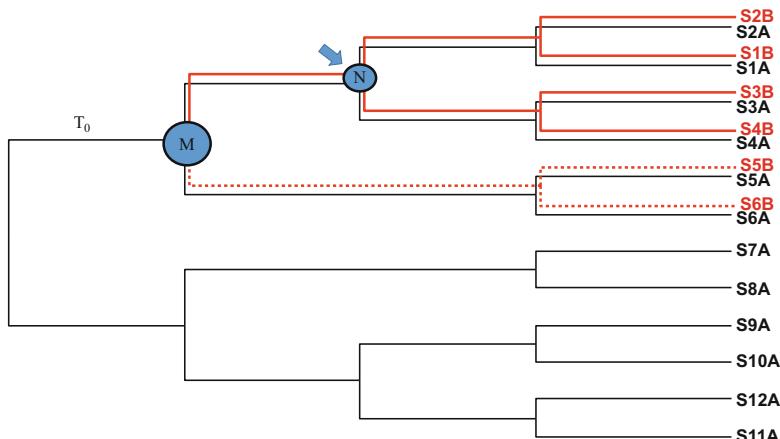
events. However, if we use a single-copy gene for dating, we can date only  $T_1$  in Fig. 15.6 but not  $T_0$ .

What could be worse is when some paralogous lineages become lost during evolution (e.g., S5B and S6B in Fig. 15.7). Differential loss of paralogous genes in different lineages is long known to affect not only dating accuracy but also phylogenetic accuracy in general (Hudson 1992).

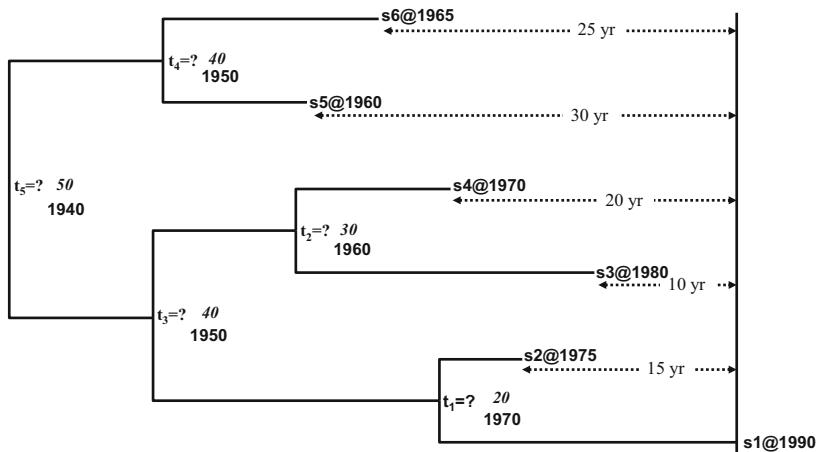
## 4.2 Tip-Dating for Rapidly Evolving Viruses

Tip-dating (Drummond et al. 2003a, b) is used for rapidly evolving viral lineages and is based on different sampling times for calibration (Fig. 15.8). Distance-based methods for dating with serial samples have already been developed (Drummond et al. 2001; Drummond and Rodrigo 2000; O'Brien et al. 2008; Yang et al. 2007) and implemented in DAMBE (Xia 2013, 2017d; Xia and Yang 2011). We can easily write down the residual sum of squares (RSS) for ordinary least-squares method:

$$\begin{aligned} \text{RSS} = & \left( \frac{d_{12}}{r} + 15 - 2t_1 \right)^2 + \left( \frac{d_{13}}{r} + 10 - 2t_3 \right)^2 + \dots \\ & + \left( \frac{d_{56}}{r} + 30 + 25 - 2t_4 \right)^2 \end{aligned} \quad (15.43)$$



**Fig. 15.7** Illustration of lineage sorting (differential loss of paralogous genes in different lineages). The loss of paralogous gene B in lineages leading to species S5 and S6 would result in underestimation of gene duplication time. Rather than identifying gene duplication time at  $T_0$ , we may think that gene duplication occurred somewhere along the branch connection nodes M and N



**Fig. 15.8** A viral sequence tree with sampling time embedded in sequence name (which is represented as species@sampling\_time). The estimated  $t_i$  values are in italics, with converted year under each  $t_i$  value

where  $d_{ij}/r$  is the time needed to generate a sequence divergence corresponding to  $d_{ij}$ . Note that, as I have shown before with the conventional internal node-calibrated dating, we can also write RSS as squared differences between the observed distance ( $d_{ij}$ ) and the expected distance:

$$\text{RSS} = (d_{12} + 15r - 2rt_1)^2 + (d_{13} + 10r - 2rt_3)^2 + \dots \\ + (d_{56} + 30r + 25r - 2rt_4)^2 \quad (15.44)$$

To obtain  $r$  and  $t_i$  values, we take partial derivatives of RSS with respect to  $r$  and  $t_i$ , set the six partial derivatives to zero, and solve the equation. The most recent sampling time (i.e., 1990) is taken as “present,” and the estimated  $t_i$  values are the number of years from the respective internal node to the “present.” These  $t_i$  values are shown as italicized number next to their respective internal node in Fig. 15.8, together with the actual year.

For tip-dating in DAMBE, the sequence name should be in one of the two following formats. The first is “Name@Year,” e.g., S1@1980, which means that sequence S1 was sampled in the year 1980. If there are multiple “@” in the sequence name, only the last “@” is used. The second format is “Name/Year,” e.g., “A/Brisbane/10/2007.” Again, only the last “/” is used. DAMBE will look for “/” only if it does not find “@” in the sequence name. The “Year” part can take three forms, e.g., “2007,” “2007.56,” and “02062007” (DDMMYYYY). Thus, sequence names such as “Brisbane/10/2007,” “H3N2Brisbame@2007.5,” and “Brisbane/10/02062007” are all valid sequence names for tip-dating in DAMBE. One should avoid using the format of “Name@00” or “Name/00” for sequences sampled in the year 2000 because it is ambiguous.

### 4.3 Obtaining Confidence Intervals by Using Bootstrapping or Jackknifing

While some dating results are published occasionally without an estimate of the variability of the estimated divergence time, such results are generally difficult to interpret with any confidence. A simple method to estimate the standard error of the time estimates is to use a resampling method such as the bootstrap or jackknife which have been used widely in molecular phylogenetics (see Felsenstein 2004 for an extensive review). The method is applicable not only to aligned sequence data but also to other genetic data such as allele frequency data with multiple loci.

For each resampled data set  $i$  and a fixed topology with  $N_n$  internal nodes, one evaluates branch lengths and obtains a set of estimated divergence time ( $T_{ij}$ , where  $j = 1, 2, \dots, N_n$ ). One can then obtain the standard error (SE) of  $T_j$  (designated by  $s_{Tj}$ ) as

$$s_{Tj} = \sqrt{\frac{\sum_{i=1}^N (T_{ij} - \bar{T}_j)^2}{N-1}}; \bar{T}_j = \frac{\sum_{i=1}^N T_{ij}}{N} \quad (15.45)$$

where  $N$  is the number of resampled data sets. The 95% confidence interval is  $\bar{T}_j \pm 1.96s_{Tj}$ . DAMBE (Xia 2013, 2017d; Xia and Yang 2011) uses this method

to obtain SE of  $T_i$  values in both tip-dating and the conventional dating with calibrated internal nodes.

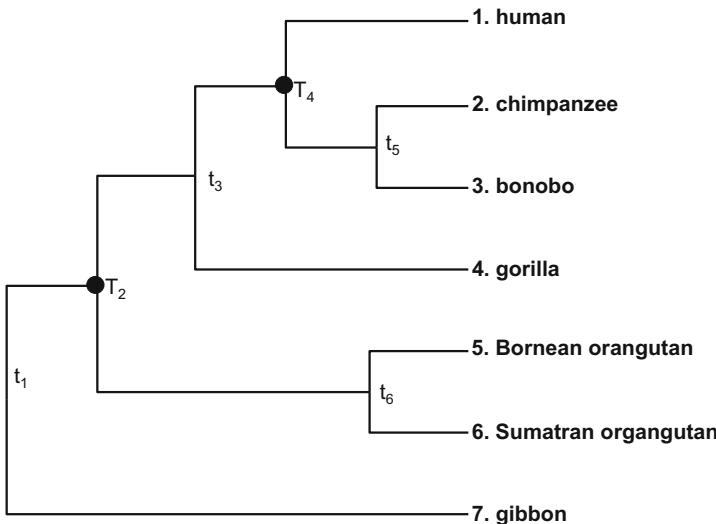
One might be wondering why we treat  $s_{Tj}$  as SE, given that  $s_{Tj}$  in Eq. (15.45) looks like an equation for standard deviation. This is because SE is “standard error of means.” Suppose you take a random sample of ten students from the first-year student population and obtain a mean body height ( $\bar{x}_1$ ). You repeat the sampling 100 times to obtain  $\bar{x}_1, \bar{x}_2, \dots, \bar{x}_{100}$ . The standard deviation of these  $\bar{x}_j$  values is the SE of  $x$ . In practice, we typically cannot afford to take many samples, and we estimate SE simply by (1) taking one sample of ten individuals, (2) computing the standard deviation (sd) from these 1 values, and (3) obtaining SE of  $x$  as  $sd/\sqrt{10}$ . When we take 100 bootstrapping samples to obtain  $T_{i,1}, T_{i,2}, \dots, T_{i,100}$ , these  $T_{i,j}$  values are equivalent to  $\bar{x}_1, \bar{x}_2, \dots, \bar{x}_{100}$ . So the standard deviation of these means is SE. You may resample 100 times or 1000 times;  $s_{Tj}$  will be roughly the same.

As we have emphasized in Eq. (15.35), there are two sources of random error in dating, one from limited sequence data and one from limited fossil data. If the amount of sequence data is infinite, then the resampled distance matrices will be identical, leading to no variation in the estimated divergence time (Thorne and Kishino 2005). This would give us false confidence in the estimated time with SE approaching 0. It is therefore important to keep in mind that the SE from resampling method estimates only uncertainty of sequence data, without any consideration of uncertainty in fossil data. Therefore, an  $SE = 0$  from resampling does not mean that the time estimates are precise. The confidence here pertains specifically to  $\varepsilon_{\text{data}}$  in Eq. (15.35). No amount of sequence data (or other data used to estimate branch lengths) can reduce uncertainty associated with fossil dates, i.e.,  $\varepsilon_{\text{fossil}}$  in Eq. (15.35), which can be estimated only from additional dated fossil data. However, if one can characterize the uncertainty in calibration time  $T$  by a distribution, then one can repeatedly sample from this distribution to obtain a set of time estimates for each internal node. In this case, when sequence data is infinite, the variation in the estimated divergence time will be all due to  $\varepsilon_{\text{fossil}}$ .

## 4.4 Application of the Dating Methods

### 4.4.1 Dating the Divergence Time of the Great Apes

The set of aligned mitochondrial sequences for seven ape species contains 9993 sites from 12 protein-coding genes (Cao et al. 1998). I chose this set of data to illustrate the LS-based dating method for comparison with results from a previous study based on Bayesian inference with the Markov chain Monte Carlo method (Rannala and Yang 2007). I also performed dating with BEAST (Drummond and Rambaut 2007) on the same data set. The same topology (Fig. 15.9) as in Rannala and Yang (2007) is used. Two fossil calibration points are indicated on the topology by  $T_2 = 14$  million years (Myr) and  $T_4 = 7$  Myr (Fig. 15.9), so we need to estimate only  $t_1, t_3, t_5$ ,



**Fig. 15.9** Topology for seven ape species.  $T_2$  and  $T_4$  are calibration points, and  $t_1, t_3, t_5, t_6$ , and  $r$  are to be estimated. OTUs are numbered so that  $d_{ij}$  in the text refers to the evolutionary distance between OTUs  $i$  and  $j$ , e.g.,  $d_{25}$  is the distance between chimpanzee and Bornean orangutan

$t_6$ , and the substitution rate ( $r$ ). However,  $T_2$  and  $T_4$  can be further refined by using the least-squares criterion.

The first and second codon positions are highly conserved in this set of sequences, with most of the substitutions observed at the third codon position. In the first part of the application, I will first use the 3331 third codon positions to illustrate the LS method with a single distance matrix. Choosing the third codon position is mainly because the third codon position is expected to evolve more in a clocklike manner than the first and second codon positions that are subject to strong purifying selection (Xia 1998b; Xia et al. 1996). Although the third codon position is also under selection pressure mediated by differential abundance of tRNA species (Carullo and Xia 2008; Xia 2005, 2008), such selection is generally weak (Higgs and Ran 2008) and expected to be much weaker than the purifying selection at the first and second codon positions.

The second part of the application illustrates the combined analysis involving more than one distance matrix. The combined analysis is performed on two distance matrices, one from codon positions 1 and 2 and the other from the third codon positions.

The evolutionary distance ( $d_{ij}$ , where  $i$  and  $j$  correspond to the taxon numbering in Fig. 15.9, i.e.,  $d_{12}$  is the distance between human and chimpanzee) is computed by using the simultaneous estimation method (Tamura et al. 2004) implemented in DAMBE (Xia 2013, 2017d) for the F84 substitution model which was used in

**Table 15.3** Distance matrix for the seven ape species

Species							
Human		0.03377	0.03298	0.04369	0.08152	0.07789	0.07964
Chimpanzee	0.35614		0.01504	0.04288	0.07899	0.07589	0.07468
Bonobo	0.34434	0.11419		0.04207	0.07845	0.07604	0.07483
Gorilla	0.49710	0.46341	0.44526		0.07895	0.07840	0.07707
OrangutanB <sup>a</sup>	0.95933	0.94465	0.93699	0.99102		0.03050	0.08896
OrangutanS <sup>b</sup>	0.93121	0.94003	0.94296	0.98467	0.20216		0.08806
Gibbon	1.33905	1.34517	1.31364	1.37386	1.42659	1.38938	

Values in the lower triangle are from the third codon position and those in the upper triangle are from the first and second codon positions

<sup>a</sup>Bornean orangutan

<sup>b</sup>Sumatran orangutan

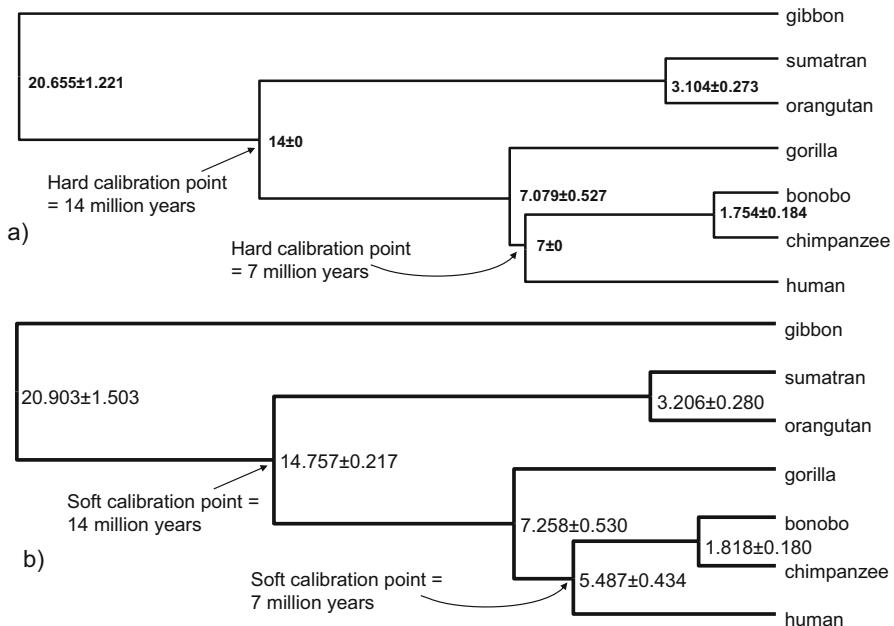
Rannala and Yang (2007). Distances from codon positions 1 and 2 are in the upper triangle in Table 15.3 and those from the third codon positions are in the lower triangle in Table 15.3.

#### 4.4.1.1 Dating with a Single Distance Matrix

With the tree topology (Fig. 15.9) and the two calibration points ( $T_2$  and  $T_4$ ) indicated on the topology, the LS solution of the substitution rate ( $r$ ) and the divergence time ( $t_1$ ,  $t_3$ ,  $t_5$ , and  $t_6$ ) is

$$\begin{aligned}
 r &= \frac{A}{4B} \\
 t_1 &= \frac{(d_{17} + d_{27} + d_{37} + d_{47} + d_{57} + d_{67})B}{3A} = \frac{(d_{17} + d_{27} + d_{37} + d_{47} + d_{57} + d_{67})}{12r}^{t_3} \\
 &= \frac{2(d_{14} + d_{24} + d_{34})B}{3A} = \frac{(d_{14} + d_{24} + d_{34})}{6r} t_5 = \frac{2d_{23}B}{A} = \frac{d_{23}}{2r} t_6 \\
 &= \frac{2d_{56}B}{A} = \frac{d_{56}}{2r} A \\
 &= T_4(d_{12} + d_{13}) \\
 &\quad + T_2(d_{15} + d_{16} + d_{25} + d_{26} + d_{35} + d_{36} + d_{45} + d_{46})B \\
 &= 4T_2^2 + T_4^2
 \end{aligned} \tag{15.46}$$

These LS estimates are appropriate when the fossil dates are accurate, i.e.,  $T_2$  and  $T_4$  (equal to 14 and 7 million years, respectively) are true divergence times of their respective nodes. The residual sum of squares (RSS) is 0.04339 when the calibration points  $T_1$  and  $T_2$  are fixed, with  $r$  and  $t_i$  values estimated by using Eq. (15.46).

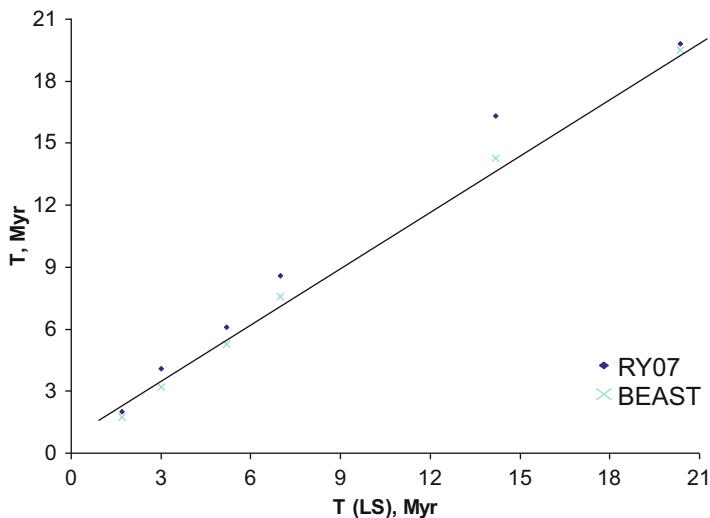


**Fig. 15.10** Dating the divergence of the great apes with the LS-based method with fixed (hard) calibration points (a) and soft calibration points (b). Each node is labeled by the mean  $\pm$  SE (standard error) estimated from 100 bootstrap samples. Two soft calibration points shown in the figure were used in dating

The divergence times estimated, together with the standard error of the estimates, are shown in Fig. 15.10a, with the evolutionary rate ( $r$ ) equal to 0.0326 per million year or 3.26 per 100 million years as in Rannala and Yang (2007).

When the calibration times  $T_1$  and  $T_2$  are allowed to change to minimize RSS, RSS is reduced from 0.04339 to 0.0149 with the estimate of  $r$  equal to 3.105 per 100 million years which is similar to that in Rannala and Yang (2007) where they obtained  $r = 3.11$  when a global clock is imposed and with soft bounding of the divergence time. The dating details, together with the bootstrap-estimated standard error of the estimates, are shown in Fig. 15.10b. These time estimates are also similar to those from Rannala and Yang (2007) using the Bayes MCMC method (Fig. 15.11).

For comparison, we have also estimated the divergence time by using BEAST (Drummond and Rambaut 2007) which is now a leading method for estimating evolutionary rates and divergence times. Setting options with the HKY85 model, with no rate heterogeneity over site, with the clock model being “Relaxed clock: uncorrelated lognormal,” with tree prior set to “Speciation: Yule process,” and with  $T_2$  set to have a mean of 14 million years and standard error of 1.3 million years in a normal distribution,  $T_2$  set to have a mean of 7 million years and standard error of 1.3 million years in a normal distribution, chain length equal to 1,000,000, and



**Fig. 15.11** Comparing the LS-based dating (horizontal axis) and the dating based on Bayesian inference with Markov chain Monte Carlo (BI-MCMC) from Rannala and Yang (2007) and from BEAST (Vertical axis), all assuming a global clock

pre-burnin of 10,000, we obtained time estimates very close to those from the LS method (Fig. 15.11).

One major difference between the Bayesian dating method and distance-based dating method is that the former can specify, albeit rather arbitrarily but explicitly, uncertainty about the calibration time, but the latter cannot. Thus, the variation in the estimated dates may be greater in the Bayesian method than in the distance-based method. This is not because the distance-based method is more efficient, but because the variation of the distance-based estimates includes uncertainty only from the sequences, whereas the variation from the Bayesian method includes uncertain in both sequences and the calibration time.

#### 4.4.1.2 Dating with Multiple Distance Matrices

Here we use two distance matrices to illustrate combined analysis with multiple distance matrices. The first distance matrix is from codon positions 1 and 2 of the ape mitochondrial sequences and the second distance matrix is from codon position 3 (upper and lower triangular matrices in Table 15.3, respectively). Because the distances from the third codon position are much greater than those from codon positions 1 and 2, we used both unscaled and scaled analyses for comparison. We should mention at the very beginning that it is not a good idea to combine highly heterogeneous genes or site partitions. So it is not a good approach to combine the third codon position with first and second codon positions. We used the two matrices only to illustrate the method.

**Table 15.4** Dating results for data at codon position 3 (CP3Only), for combined analysis of two matrices (one from codon positions 1 and 2 and the other from codon position 3) using unscaled approach (Unscaled) and scaled approach (Scaled)

Time	CP3Only	Unscaled	Scaled
$t_1$ (gibbon-hominid)	21.558	20.312	17.239
$T_2$ (orangutan-human+chimp+gorilla)	14.750	14.200	14.200
$t_3$ (gorilla-human+chimp)	7.314	6.991	7.388
$T_4$ (human-chimp)	5.500	5.200	5.600
$t_5$ (chimp-bonobo)	1.830	1.709	2.243
$t_6$ (Bornean orangutan-Sumatran orangutan)	3.244	3.029	4.345
$r_{12}^a$		0.241	0.259
$r_3^b$	3.105	3.356	3.603

Initial values for  $T_2$  and  $T_4$  are 14 and 7 million years, respectively. Substitution rate  $r$  is measured by the expected number of substitutions per site per 100 Myr

<sup>a</sup>Substitution rate at codon positions 1 and 2

<sup>b</sup>Substitution rate at codon positions 3

Designate the substitution rate and evolutionary distance at the first and second codon positions as  $r_A$  and  $d_{A,ij}$ , respectively, and those at the third codon position  $r_B$  and  $d_{B,ij}$ , respectively. The  $k$  value, estimated by the linear regression of  $d_B = k \cdot d_A$ , is 13.9. An unscaled analysis analogous to that specified in Eq. (15.36), combining the two distance matrices, results in estimates (under column heading “Unscaled” in Table 15.4) very similar to those obtained with the third codon position alone (under column heading “CP3Only” in Table 15.4). This is expected because the estimation is dominated by the distance matrix with greater values. A scaled approach, analogous to that specified in Eq. (15.38), has slightly different results (under column heading “Scaled” in Table 15.4). For comparison with the estimates from a combined analysis with site partitions or multiple genes in the likelihood or Bayes framework, we should use the estimates from the unscaled method.

Dating with a new distance matrix generated by using Eq. (15.39) produced results almost identical to that with the third codon position alone. This is understandable because the new  $d_{ij}$  is almost identical to  $d_{ij}$  based on the third codon position alone.

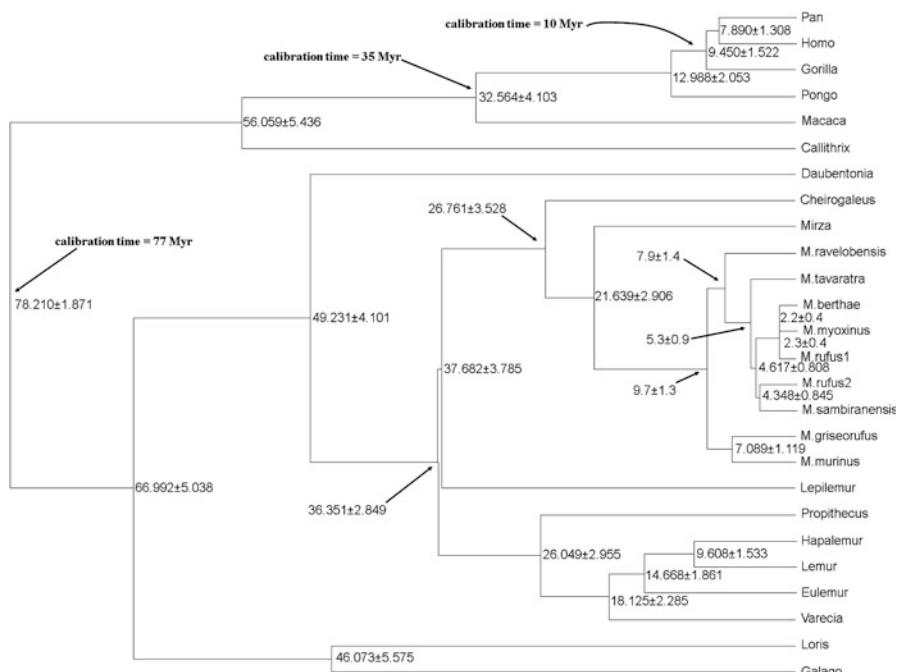
We have also performed dating and bootstrapping with the three site partitions (i.e., the three codon positions) as follows. Each site partition was bootstrapped separately, so each resampled data set will lead to three separate distance matrices for first, second, and third codon positions, respectively. The three matrices are then combined into one matrix according to Eq. (15.39). The new matrix is then used for dating. This is repeated 100 times, and the mean divergence time and the associated standard error are estimated in the same way as in Eq. (15.45). The results are similar to those in Fig. 15.10, but the standard error is slightly larger, which is understandable because the second codon position violates the molecular clock hypothesis (likelihood ratio test. With the F84 model,  $\ln L$  is  $-6381.9048$  and  $-6388.4284$ , respectively, for a tree without a clock and with a global clock,  $2\Delta\ln L = 13.0471$ ,  $DF = 5$ ,  $p = 0.0229$ ). Combining third codon positions from different mitochondrial protein-coding genes invariably leads to reduced standard error.

#### 4.4.2 Dating the Divergence Time of the Mouse Lemurs

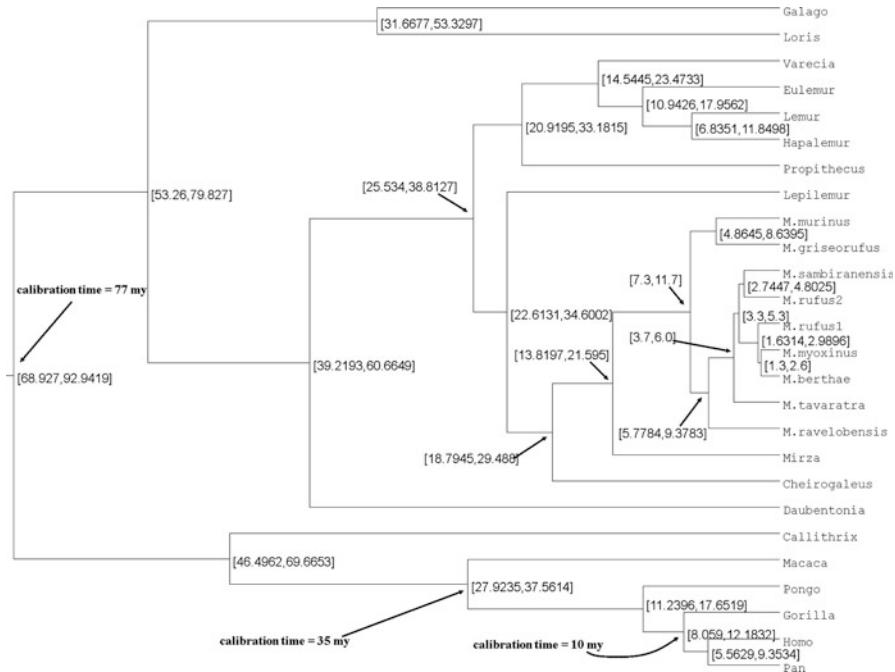
Here I compare the dating results between the LS method and BEAST (Drummond and Rambaut 2007) by using the mouse lemur data set (Yang and Yoder 2003). The data set consists of two mitochondrial genes (COII and Cyt-b) from 35 mammalian species, of which 26 are primate species. I used only the 604 third codon positions of the primate species because the third codon position evolves in a more clocklike manner than the other two codon positions (Yang and Yoder 2003).

Three calibration points for primates and four calibration points for non-primates were used in Yang and Yoder (2003). However, the calibration points for non-primate species are somewhat doubtful as expressed in the original publications cited in Yang and Yoder (2003). So I used only the three calibration points for the primates. I used BEAST with the settings identical to those for analyzing the great ape data except that the calibration points which are 77 million years for the root of primates, 35 million years for monkey/ape divergence, and 10 million years for human/gorilla divergence, i.e., the same as those used in Yang and Yoder (2003).

I first performed dating with BEAST and the LS-based method by using only the primate species. The dating results from the LS-based method (Fig. 15.12) are shown with each node labeled with the mean divergence time and the standard error



**Fig. 15.12** Dating the divergence of primates with the LS-based method. Each node is labeled with the mean divergence time and standard error (mean±s) estimated from 100 bootstrap samples. A global clock and the F84 substitution model were used. Three calibration points used are shown



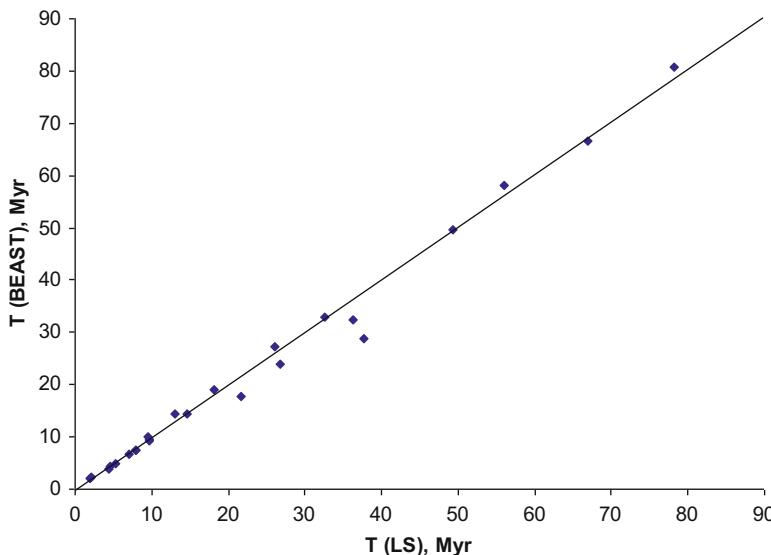
**Fig. 15.13** Dating the divergence of primates with BEAST. Each node is labeled with a 95% highest posterior density (HPD) interval of the estimated divergence time. A global clock and the HKY85 substitution model were used. Three calibration points used are shown

estimated by 100 bootstrap samples. The results are nearly identical to those from BEAST (Fig. 15.13) where each node is labeled with a 95% high posterior density (HPD) interval of the estimated divergence time. The mean divergence time from the LS-based method consistently falls right in the middle of the time interval from BEAST (Figs. 15.12 and 15.13).

To check whether there might be discordance with deep or shallow divergence times, I have plotted all corresponding divergence times from BEAST and from the LS-based method. The points effectively fall on a straight line (Fig. 15.14).

While the performance of distance-based methods in dating speciation and gene duplication events have not been evaluated extensively, the similarity between the estimates from the distance-based dating and those from Bayesian inference (Rannala and Yang 2007) and from BEAST suggests that the distance-based method is not only very simple and extremely fast but also accurate.

The cause of the minor difference between estimated divergence time in this paper and those in Rannala and Yang (2007) can be attributed mainly to the two calibration points  $T_2$  and  $T_4$ . Applying the LS criterion, the distance-based method



**Fig. 15.14** Concordance in dating results between the LS-based method, designated as T (LS) and BEAST, designated as T (BEAST). Results are from 26 primate species

fine-tuned  $T_2$  to 14.20–14.75 Myr in the three separate estimations (Table 15.4) and  $T_4$  to 5.2–5.6 Myr in the three separate estimations (Table 15.4). In Rannala and Yang (2007),  $T_2$  was fine-tuned to ~16 Myr and  $T_4$  to 6.1–6.2 Myr. A recent study with extensive data analysis found  $T_4$  to be 4.1 Myr (Hobolth et al. 2007), suggesting that the LS estimate (5.2–5.6 Myr) may be closer to the truth than in Rannala and Yang (2007) with  $T_4 > 6$  Myr. Also, the current consensus on  $T_2$  among paleoanthropologists is 14 Myr or earlier (Raam et al. 2005), again suggesting that our estimate here (14.20–14.75 Myr) may be closer to the truth than in Rannala and Yang (2007) with  $T_4$  ranging from 15.8 to 16.3 with different clock models.

The dating method presented here should be useful for many new genome-based distances proposed in recent years. These include genome BLAST distances (Auch et al. 2006; Deng et al. 2006; Henz et al. 2005), breakpoint distances based on genome rearrangement (Gramm and Niedermeier 2002; Herniou et al. 2001), distances based on the relative information between unaligned/unalignable sequences (Otu and Sayood 2003), distances based on the sharing of oligopeptides (Gao and Qi 2007), the composite vector distance (Xu and Hao 2009), and composite distances incorporating several whole-genome similarity measures (Lin et al. 2009). However, how well do these distances conform to the molecular clock have not been evaluated thoroughly.

## Postscript

1983 is the year when Motoo Kimura published his book entitled “The neutral theory of molecular evolution.” Molecular phylogenetics, as well as chapters on substitution models, are largely based on neutral theory. Only when an overwhelming majority of substitutions are neutral or nearly neutral will the number of substitutions between two homologous genes be roughly proportional to time, which is essential for constructing phylogenetic trees and dating speciation events. Substitutions resulting from natural selection often result in a molecular version of “punctuated equilibrium” and often not occur in a rate proportional to time. The confirmation of numerous numbers of predictions derived from neutral theory attests to the success of the theory.

1983 is also the year when Peter Medawar and Jean Medawar published a book entitled “Aristotle to zoos: a philosophical dictionary of biology” claiming that “in a sense all evolution is adaptation” (Medawar and Medawar 1983, p. 1), when the neutral theory had become so successful, so well-established, so widely accepted, and its validity so obvious. Adaptation results from natural selection, but much of evolution at the molecular level is attributable to fixation of neutral alleles. What was wrong with the Medawars failing to see this? Peter Medawar was obviously a very smart person, and he understood evolution well – his thinking on the evolution of aging is insightful. However, the claim is silly. It seems paradoxical for a person to be smart and silly at the same time.

But we are all smart and silly at the same time. We don’t see all things, just like a little chick that has scratched the surface in a corner of a vast terrain (to paraphrase Darwin), but we all have a tendency to extrapolate our fragmented observation to the whole world. Our limited interaction with a biased sample of people from a race or an ethnic group often results in racist views, and our limited understanding of other subjects often leads to academic bigotry. Think of claims such as “what is true for the colon bacillus is true for the elephant” (attributed to Jacques Monod, Jacob 1988, p. 290), or “Nothing in biology makes sense except in the light of evolution” (Dobzhansky 1973), or “All science is either physics or stamp collecting” attributed to Ernest Rutherford.

What is particularly amazing to me is that grandiosity, either racial or academic, is often much valued and treasured. Dobzhansky’s assertion has given evolutionary biologists a tremendous feeling of pride, and some of them never seem to get tired of repeating it. Rutherford’s claim is said to have inspired many young physicists. Racial or religious grandiosity had driven huge crowds of people to a narcissistic and exalted state of mind that they feel happiness beyond description.

If grandiosity blinds our views and blocks our senses, why does it often remain so captivating and delicious?

# Chapter 16

## Maximum Likelihood in Molecular Phylogenetics



### 1 Introduction

Both MP and ML methods operate on a set of aligned sequences typically represented as an  $N \times L$  character matrix, where  $N$  is the number of sequences and  $L$  is the aligned sequence length. The core algorithms of the two methods take the character matrix and a topology as input, assign a value to each of the  $L$  sites, and sum up the values as a criterion for choosing the best topology. The site-specific value is the substitution cost ( $c_i$ ) in the MP method and log-likelihood ( $\ln L_i$ ) in the ML method. The best topology is one with the smallest  $\sum c_i$  in MP or the largest  $\sum \ln L_i$  in ML, where  $i = 1, 2, \dots, L$ .

While the key algorithm in for the MP method is the Fitch (1971) and the Sankoff (1975) algorithm, the key algorithm in the ML approach is the pruning algorithm for computing the likelihood based on equilibrium frequencies and transition probabilities in phylogenetic reconstruction. The pruning algorithm not only speeds up computation but also offers a natural and statistically valid way to handle missing data.

This chapter deals only with tree reconstruction by the ML method based on sequence data. For the ML method for computing pairwise evolutionary distances, please review the chapter on substitution models and the chapter on distance-based phylogenetic methods. For comparative methods using the likelihood approach based on Brownian motion model, with a post hoc modification for characterizing directional changes, please read the chapter on comparative methods. For the likelihood method and likelihood ratio test used to characterize association between discrete characters, please read the chapter on large-scale characterization on association between discrete characters.

We will first present simple examples of likelihood-based estimation to illustrate the rationale of the maximum likelihood approach and then detail the likelihood calculation given a topology and a set of aligned sequences. The basic knowledge needed for this chapter is tree traversal and transition probabilities derived from

substitution models. Readers who do not know transition probabilities should review a previous chapter on substitution models. We will illustrate the likelihood method with the simplest scenario with four sequences labeled S1 to S4 and three possible unrooted trees. We need to compute the log-likelihood ( $\ln L$ ) for each tree and choose the tree with the maximum likelihood as the ML tree. We will first take a brute-force approach to computing the  $\ln L$  of the tree and then introduce the pruning algorithm (Felsenstein 1973, 1981, 2004, pp. 253–255). The last section illustrates a bias in the likelihood method when missing data is associated with rate heterogeneity among sites (Xia 2014). Likelihood-based phylogenetic methods, as well as the associated statistics for alternative tree topologies, are implemented in DAMBE (Xia 2013, 2017d).

## 2 The Rationale of Maximum Likelihood Approach

Maximum likelihood (ML) is a criterion in model selection and in statistical estimation of model parameters, i.e., the best model and the best parameter values given a model are those that maximize the likelihood. Recall that we always need to have a criterion whenever we need to choose one from several alternatives. Two sequences could have different pairwise alignments, and our criterion is the alignment score given a scoring scheme (specified by gap penalties and a match/mismatch matrix). Alignment A is better than Alignment B if the former has higher alignment score than the latter. In the chapter on Gibbs sampler, we could have different sets of putative motifs, and our criterion is the Kullback-Leibler information (F). Any set of putative motifs that gives us the largest F is the best set. In the chapters on substitution models and distance-based phylogenetic methods, we have used the least-squares criterion (both the ordinary and the weighted least-squares). A set of branch lengths that minimize the residual sum of squares is the best branch length estimates. For choosing the best tree among different alternatives, we have used the minimum evolution criterion, i.e., whichever tree has the shortest tree length is the best tree. Maximum likelihood is just one of these criteria for making a choice among alternatives.

Let us illustrate the ML approach with a few examples. Suppose we wish to estimate the proportion of males ( $p$ ) of a fish population in a large lake. A random sample of  $N$  fish contains  $M$  males. With the binomial distribution, the likelihood, which is the probability mass function for discrete variables, is

$$L = \frac{N!}{M!(N-M)!} p^M (1-p)^{N-M}. \quad (16.1)$$

The maximum criterion states that the best  $p$  should maximize the likelihood value given the observation. This maximization process is simplified by maximizing the natural logarithm of  $L$  instead:

$$\begin{aligned}\ln L &= A + M \ln(p) + (N - M) \ln(1 - p) \\ \frac{\partial \ln L}{\partial p} &= \frac{M}{p} - \frac{N - M}{1 - p} = 0 \\ p &= \frac{M}{N}.\end{aligned}\tag{16.2}$$

The likelihood estimate of the variance of  $p$  is the negative reciprocal of the second derivative,

$$\text{Var}(p) = -\frac{1}{\frac{\partial^2 \ln(L)}{\partial p^2}} = -\frac{1}{-\frac{M}{p^2} - \frac{N-M}{(1-p)^2}} = \frac{p(1-p)}{N}.\tag{16.3}$$

Note that the likelihood method needs a model (binomial distribution in our example) to formulate the likelihood function, which is in Eq. (16.1) for our example. For this reason, the likelihood method is always model-based.

Let us have just one more example to illustrate the likelihood approach in model selection. Suppose we wish to know whether body height differs between male and female students. We randomly sampled seven male and six female students and measured body height (shown in the first two columns in Table 16.1). We consider three hypotheses (or models). The first (M1) assumes that male and female students do not differ in body height and all 13 values represent sample values from the same normal distribution specified by mean  $\mu$  and standard deviation  $\sigma$ . The likelihood for continuous variables is the probability density function. For example, the likelihood for body height of 170:

$$L(170|M1) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{170-\mu}{\sigma}\right)^2}\tag{16.4}$$

The log-likelihood function for the 13 sample values is

**Table 16.1** Body height (in cm) measured from seven male ( $M$ ) and six female ( $F$ ) students

Height		M1 ( $\mu, \sigma$ )		M2 ( $\mu_M, \sigma_M, \mu_F, \sigma_F$ )		M3 ( $\mu_M, \mu_F, \sigma$ )	
M	F	lnLM	lnLF	lnLM	lnLF	lnLM	lnLF
170	170	-2.92876	-2.92876	-3.92540	-2.49898	-3.92540	-2.58489
175	165	-2.86487	-3.54639	-2.63108	-2.49893	-2.63108	-2.58485
175	172	-2.86487	-2.83676	-2.63108	-2.93880	-2.63108	-2.92317
180	170	-3.35471	-2.92876	-2.54483	-2.49898	-2.54483	-2.58489
181	168	-3.51913	-3.10936	-2.67254	-2.31048	-2.67254	-2.43991
179	160	-3.21244	-4.71774	-2.46543	-4.06956	-2.46543	-3.79288
185		-4.39828		-3.66665		-3.66665	

Three models (M1, M2, and M3) are evaluated, assuming normal distribution. M1: male and female students do not differ in body height, and all 13 values represent sample values from the same population. M2: the seven male and six female sample values are from two populations differing in both mean and standard deviation. M3: the seven male and six female sample values are from two populations differing in mean but having the same standard deviation

$$\ln L_{M1} = \ln [L(170|M1)] + \ln [L(175|M1)] + \cdots + \ln [L(160|M1)] \quad (16.5)$$

which is a function of two unknowns ( $\mu$  and  $\sigma$ ). The likelihood criterion states that the best  $\mu$  and  $\sigma$  should maximize  $\ln L$ . We thus take partial derivatives of  $\ln L_{M1}$  with respect to  $\mu$  and  $\sigma$ , set the partial derivatives to zero, and solve the two simultaneous equations. This gives us  $\mu = 173.0769$  and  $\sigma = 6.7192$  and  $\ln L_{M1} = -43.2108$ .  $\ln L$  for individual observations are shown in Table 16.1, in the two columns under the heading of M1. This M1 is equivalent to the null hypothesis of no difference in body height between male and female students.

Suppose we now have a second hypothesis (M2) that sample values from males and females belong to two different normal distributions, with the male population specified by  $\mu_M$  and  $\sigma_M$ , and the female population specified by  $\mu_F$  and  $\sigma_F$ . Now the same sample value (e.g., 170) for a male and a female student will have different likelihoods, denoted  $L_{M,170}$  and  $L_{F,170}$  below

$$L(170, \text{male}|M2) = \frac{1}{\sigma_M \sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{170-\mu_M}{\sigma_M}\right)^2}; \quad L(170, \text{female}|2) = \frac{1}{\sigma_F \sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{170-\mu_F}{\sigma_F}\right)^2} \quad (16.6)$$

Designating body height for males and females as MH and FH, respectively, the log-likelihood ( $\ln L$ ) for the seven male and six female sample values is

$$\ln L_{M2} = \sum_{i=1}^7 \ln L(MH_i|M2) + \sum_{i=1}^6 \ln L(FH_i|M2) \quad (16.7)$$

Now the function  $\ln L_{M2}$  has four parameters. The maximum likelihood estimates of these parameters (i.e., the parameters that maximize  $\ln L_{M2}$ ) are  $\mu_M = 177.8570$ ,  $\sigma_M = 4.5491$ ,  $\mu_F = 167.4998$ , and  $\sigma_F = 3.9896$ , with the resulting  $\ln L_{M2} = -37.3527$ . The log-likelihood of these individual sample values are shown in Table 16.1 in the two columns under M2.

Which model (M1 or M2) is the more preferable? M1 is simpler, with only two parameters, but is it too simple as to fail to describe nature adequately? What criterion should we use to discriminate between these two models? Just as  $R^2$  is a measure of how well a model fits the data in a least-squares context,  $\ln L$  is a measure of how well a model fits the data in a likelihood context. However, just as  $R^2$  is not a good criterion for model selection, so is  $\ln L$ . A model will increase its fit to the data if we keep adding parameters. If we use  $R^2$  or  $\ln L$  as a criterion for model selection, then complicated models will be favored against simple models.

A series of models are termed nested models if the simpler model is a special case of the more complicated model. For example, the following are nested models because the second can be reduced to the first if  $b = 0$  and the third can be reduced to the second if  $b_2 = 0$ :

$$\begin{aligned}y &= a \\y &= a + bx \\y &= a + b_1x + b_2x^2\end{aligned}\tag{16.8}$$

In our case, M1 and M2 are nested model because M2 is reduced to M1 if  $\mu_M = \mu_F$  and  $\sigma_M = \sigma_F$ .

Nested models can be tested by the likelihood ratio test (LRT). Any statistical significance test will have two essential quantities: a statistic that measures the difference between the two models (two hypotheses) and a known distribution of the statistic. The statistic in LRT is  $2\Delta\ln L$  (twice the difference in  $\ln L$  between the two nested models, which is often referred to as the likelihood ratio chi-square) which follows approximately the  $\chi^2$  distribution with the degree of freedom being  $\Delta p$  (the difference in the number of parameters between the two nested models). In our example,  $2\Delta\ln L = 2*(\ln L_{M2} - \ln L_{M1}) = 11.7162$ . With two degrees of freedom,  $p = 0.0029$ . Note that the null hypothesis being tested in LRT is that M1 and M2 are equally good, and this null hypothesis is rejected at  $p = 0.0029$ , so we adopt M2 and conclude that male and female students differ highly significantly in body height.

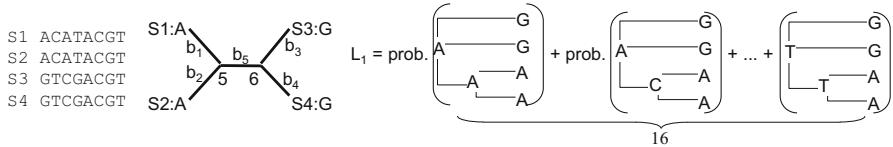
Now suppose we have a third hypothesis (M3) stating that the male and female populations differ in means but not in standard deviation, i.e.,  $\mu_M \neq \mu_F$  but  $\sigma_M = \sigma_F = \sigma$ . Is this M3 as good as M2? We can do the same calculation to obtain  $\mu_M = 177.8570$ ,  $\mu_F = 167.4998$ , and  $\sigma = 4.5491$ , with  $\ln L_{M3} = -37.4476$ . The  $\ln L$  for individual sample values are also shown in Table 16.1 in the two columns under M3. The likelihood ratio chi-square  $2\Delta\ln L = 0.1897$ . With one degree of freedom,  $p = 0.6631$ , so we cannot reject the null hypothesis that M3 and M2 are equally good.

### 3 Likelihood for a Phylogenetic Tree

Students often find it a steep learning curve to proceed from the sections above to the sections below. The first difficulty is the tree traversal, the second is the pruning algorithm, and the third is that they have forgotten about substitution models when the classes come to this point. It is important that students should review the substitution models. We will first go slowly with tree traversal with a brute-force approach without the pruning algorithm, which is followed by a detailed numerical illustration of the pruning algorithm, together with missing data handling.

#### 3.1 The Brute-Force Approach

Given a site in a set of aligned sequences and a topology (Fig. 16.1), we have two internal nodes with unknown states, leading to 16 possible nucleotide configurations



**Fig. 16.1** Computing likelihood with a brute-force approach involving one site (site 1) in a set of four aligned sequences. Nodes 5 and 6 have unknown states that could be A, C, G, or T, generating 16 different combinations. We need to compute likelihood for each site designated  $L_1$  to  $L_{16}$ . The log-likelihood of a tree is  $\ln L = \sum \ln(L_i)$ , and the maximum likelihood tree is the one with the highest  $\ln L$  among all possible topologies

with the same topology. At first sight, it seems that we need to sum up 16 terms to obtain the likelihood of each site (Fig. 16.1). The likelihood for site 1 is

$$\begin{aligned} L_1 = & \pi_A P_{AA}(b_1)P_{AA}(b_2)P_{AA}(b_5)P_{AG}(b_3)P_{AG}(b_4) \\ & + \pi_C P_{CA}(b_1)P_{CA}(b_2)P_{CA}(b_5)P_{AG}(b_3)P_{AG}(b_4) \\ & + \dots + \pi_T P_{TA}(b_1)P_{TA}(b_2)P_{TT}(b_5)P_{TG}(b_3)P_{TG}(b_4) \end{aligned} \quad (16.9)$$

where  $\pi_A$ ,  $\pi_C$ ,  $\pi_G$ , and  $\pi_T$  are equilibrium frequencies,  $b_1$  to  $b_5$  are branch lengths, and  $P_{AA}$ ,  $P_{AG}$ , ...,  $P_{TT}$  are transition probabilities derived from a given substitution model. One can express the likelihood for other sites in the same way. For example, the likelihood for site 5 is

$$\begin{aligned} L_5 = & \pi_A P_{AA}(b_1)P_{AA}(b_2)P_{AA}(b_5)P_{AA}(b_3)P_{AA}(b_4) \\ & + \pi_C P_{CA}(b_1)P_{CA}(b_2)P_{CA}(b_5)P_{AA}(b_3)P_{AA}(b_4) \\ & + \dots + \pi_T P_{TA}(b_1)P_{TA}(b_2)P_{TT}(b_5)P_{TA}(b_3)P_{TA}(b_4) \end{aligned} \quad (16.10)$$

For illustration, we may take equal nucleotide frequencies and the simplest JC69 model with only two distinct transition probabilities:

$$P_{ii}(b) = \frac{1}{4} + \frac{3}{4}e^{-4b/3}, P_{ij}(b) = \frac{1}{4} - \frac{1}{4}e^{-4b/3} \quad (16.11)$$

where  $b$  is branch length between neighboring nodes.

Because the JC69 model assumes that the four nucleotides replace each other with equal rate, the four sequences in Fig. 16.1 have only two site patterns, one shared among sites 1–4 and the other shared among sites 5–8. Given the simple JC60 model, the 16 terms for the first site pattern in Eq. (16.9) can be reduced to seven terms because many terms are identical. For example, seven terms are identical and equal to  $P_{ij}(b_1)*P_{ij}(b_2)*P_{ij}(b_3)*P_{ij}(b_4)*P_{ij}(b_5)*0.25$  when internal nodes 5 and 6 are occupied by nucleotide pairs (C, A), (C, T), (G, A), (G, C), (G, T), (T, A), and (T, C), where the first nucleotide within parenthesis is at node 5 and second at node 6. Similarly, the 16 terms for the second pattern in Eq. (16.10) can be reduced to five terms. The tree  $\ln L$  given the four sequences in Fig. 16.1 is

$$\ln L = 4 \ln L_1 + 4 \ln L_5 \quad (16.12)$$

Maximizing  $\ln L$  results in  $b_1 = b_2 = b_3 = b_4 = 0$  and  $b_5 = 0.823959217$ , with  $\ln L = -21.029981488111$ . That  $b_1 = b_2 = b_3 = b_4 = 0$  is as expected because S1 = S2 and S3 = S4 (Fig. 16.1), so there are really just two sequences instead of four. Note that if we do not use  $\ln L$  but use  $L$  instead, then  $L = 7.358598123 \times 10^{-10}$ , which is already small. A larger tree with many OTUs will result in an  $L$  so small that computers will not distinguish it from zero.

We have evaluated just one topology in Fig. 16.1 for the four OTUs. There are two other unrooted topologies for four OTUs, one with OTUs S1 and S3 clustered together and the other with OTUs S1 and S4 clustered together.  $L_5$  in Eq. (16.10) is the same for all three topologies, but we need to recompute  $L_1$ . The resulting  $\ln L$ , given the JC69 model, for these two topologies are the same and equal to  $-30.96960809$ . Thus, the topology in Fig. 16.1, with S1 and S2 clustered together and  $b_1 = b_2 = b_3 = b_4 = 0$  and  $b_5 = 0.823959217$ , is the best tree of the three, because it has the largest  $\ln L$  ( $= -21.029981488111$ ).

### 3.2 The Pruning Algorithm

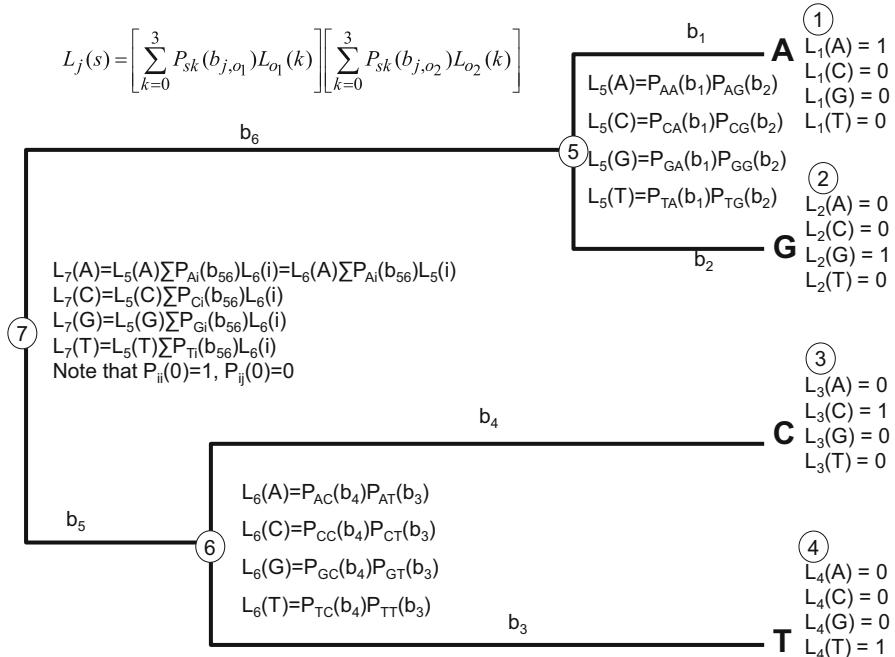
The brute-force approach for computing  $\ln L$  is unnecessary and is not used in practice other than classroom or textbook illustration. The pruning algorithm (Felsenstein 1973, 1981, 2004, pp. 253–255), illustrated below, economizes the computation substantially.

As in the maximum parsimony method, we only need to illustrate the application of the pruning algorithm for a single site because all sites are computed the same way. For any given topology, e.g., the four-species topology in Fig. 16.1, we first define a likelihood vector ( $L_j$ ) for each of the nodes including the leaf nodes. The vector contains four elements for nucleotide sequences, 20 for amino acid sequences or the number of sense codons for codon sequences with codon-based models. We will use nucleotide sequences for illustration, but the computation is the same for amino acid or codon sequences.

We have four sequences with the first site being A, G, C, and T for species 1, 2, 3, and 4, respectively (Fig. 16.2). Our task is to compute the  $\ln L_1$  for this first site with the pruning algorithm. The computation is the same for all other sites.

For a leaf node  $j$  with a nucleotide  $s$  (where  $s$  is either A, C, G, or T),  $L_j(s) = 1$ , and  $L_j(\bar{s}) = 0$  (Fig. 16.2). For example, for the first sequence with nucleotide A,  $L_1(A) = 1$ , and  $L_1(C) = L_1(G) = L_1(T) = 0$ . For an internal node  $j$  with two offspring ( $o_1$  and  $o_2$ ),  $L_j$  is recursively defined as

$$L_j(s) = \left[ \sum_{k=0}^3 P_{sk}(b_{j,o_1}) L_{o_1}(k) \right] \left[ \sum_{k=0}^3 P_{sk}(b_{j,o_2}) L_{o_2}(k) \right] \quad (16.13)$$



**Fig. 16.2** Likelihood computation with the pruning algorithm on a four-species tree. Node  $j$  is represented by a vector ( $L_j$ ) of four elements for nucleotide sequences or 20 for amino acid sequences.  $L_j$  is computed according to equation on the upper-left, i.e., Eq. (16.13).  $b_5$  and  $b_6$  cannot be estimated separately without assuming a molecular clock, and only their summation ( $b_{56}$ ) is used in likelihood calculation

where  $s$  is either A, C, G, or T;  $k$  takes value of 0, 1, 2, and 3 corresponding to nucleotides A, C, G, and T;  $b_{j,o1}$  is the branch length between internal node  $j$  and its offspring  $o_1$ ; and  $P_{sk}$  is the transition probability from state  $s$  to state  $k$  (where  $s$  and  $k$  are A, C, G, or T for nucleotide sequences). Transition probabilities  $P_{sk}$  and its derivation from various substitution models have been detailed in the chapter on substitution models.

The application of Eq. (16.13) is straightforward. Take, for example, internal node 5 with its two offspring nodes 1 and 2,

$$L_5(A) = \left[ \sum_{k=0}^3 P_{Ak}(b_1)L_1(k) \right] \left[ \sum_{k=0}^3 P_{Ak}(b_2)L_2(k) \right] \quad (16.14)$$

Because  $L_1(A) = 1$ ,  $L_1(C) = L_1(G) = L_1(T) = 0$ ,  $L_2(G) = 1$ ,  $L_2(A) = L_2(C) = L_2(T) = 0$ ,  $L_5(A)$  becomes

$$L_5(A) = P_{AA}(b_1)P_{AG}(b_2) \quad (16.15)$$

The other three elements, as well as  $L_j$  vectors for other internal nodes, are listed in Fig. 16.2.

Internal node 7 is special in that we cannot estimate  $b_5$  and  $b_6$  separately because the substitution model is time-reversible and the resulting tree is consequently unrooted. We simply move node 7 to the location of node 6 (or node 5), so that either  $b_5$  or  $b_6$  is 0 and the other is then equal to  $(b_5 + b_6)$  represented as  $b_{56}$  in  $L_7$  in Fig. 16.2. If  $b_5$  is 0, then  $P_{ii}(b_5) = 1$  and  $P_{ij}(b_5) = 0$ , i.e., no time for anything to change. This leads to the simplified equations for computing  $L_7(i)$  in Fig. 16.2. The final likelihood for the tree is

$$L = \sum_{i=0}^3 \pi_i L_7(i) \quad (16.16)$$

where  $\pi_i$  is the equilibrium frequency of nucleotide  $i$  and reflects the assumption that sufficient time has elapsed for the frequencies to reach equilibrium. Note that, for the JC69 model,  $\pi_i = 1/4$ .

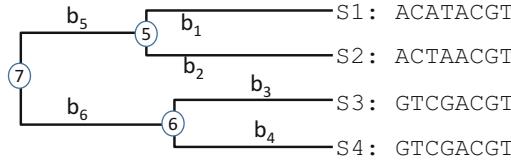
The application of the pruning algorithm to the aligned sequences and the topology in Fig. 16.1 with the JC69 model will also result in  $b_1 = b_2 = b_3 = b_4 = 0$  and  $b_5 = 0.823959217$ , with  $\ln L = -21.029981488111$ , just as we have obtained with the brute-force approach. The benefit of the pruning algorithm is the reduction of repeated calculation.

## 4 Calculating Likelihood by Imposing a Molecular Clock

Recall that molecular phylogenetics has two main objectives: (1) defining the branching pattern and (2) dating evolutionary events such as speciation events or gene duplication events. A good molecular clock is important for dating. To test the molecular clock hypothesis, we need to compute  $\ln L$  by imposing a molecular clock and  $\ln L$  without a molecular clock and then use the likelihood ratio test to see if the two fit the data equally well. If we have long sequence alignment but the molecular clock hypothesis is not rejected, then we can calibrate the clocked tree to perform dating.

This section illustrates the calculation of likelihood given the sequence alignment and topology in Fig. 16.3. The pruning algorithm is the same as before except that we constrain  $b_1 = b_2$ ,  $b_3 = b_4$ , and  $(b_1 + b_5) = (b_3 + b_6)$ . For simplicity, we will again use the JC69 model, which allows us to reduce the eight aligned sites (Fig. 16.3) to three site patterns: pattern 1 for sites 1–2, pattern 2 for sites 3–4, and pattern 3 for sites 5–8. Thus, we only need to compute the likelihood for sites 1, 3, and 5.

For site 1 (site pattern 1), the likelihood vector  $L$  for the internal nodes 5, 6 and 7 are



**Fig. 16.3** Sequence alignment and topology for illustrating likelihood calculation with a molecular clock. Imposing a molecular clock implies  $b_1 = b_2$ ,  $b_3 = b_4$ , and  $(b_1 + b_5) = (b_3 + b_6)$ . So there are only three branch lengths to estimate. Without a clock we would need to estimate five branch lengths:  $b_1$ ,  $b_2$ ,  $b_3$ ,  $b_4$ , and  $b_{56}$

$$\begin{aligned} L_{5.1}(A) &= P_{AA}(b_1)P_{AA}(b_2) = P_{ii}^2(b_1) \\ L_{5.1}(C) &= P_{CA}(b_1)P_{CA}(b_2) = P_{ij}^2(b_1) = L_{5.1}(G) = L_{5.1}(T) \end{aligned} \quad (16.17)$$

where  $L_{5.1}$  denotes the  $L$  vector for internal node 5 with site pattern 1.

$$\begin{aligned} L_{6.1}(A) &= P_{AG}(b_3)P_{AG}(b_4) = P_{ij}^2(b_3) = L_{6.1}(C) = L_{6.1}(T) \\ L_{6.1}(G) &= P_{GG}(b_3)P_{GG}(b_4) = P_{ii}^2(b_3) \end{aligned} \quad (16.18)$$

$$\begin{aligned} L_{7.1}(A) &= \left[ \sum_{k=0}^3 P_{Ak}(b_5) L_{5.1}(k) \right] \left[ \sum_{k=0}^3 P_{Ak}(b_6) L_{6.1}(k) \right] \\ L_{7.1}(C) &= \left[ \sum_{k=0}^3 P_{Ck}(b_5) L_{5.1}(k) \right] \left[ \sum_{k=0}^3 P_{Ck}(b_6) L_{6.1}(k) \right] \\ L_{7.1}(G) &= \left[ \sum_{k=0}^3 P_{Gk}(b_5) L_{5.1}(k) \right] \left[ \sum_{k=0}^3 P_{Gk}(b_6) L_{6.1}(k) \right] \\ L_{7.1}(T) &= \left[ \sum_{k=0}^3 P_{Tk}(b_5) L_{5.1}(k) \right] \left[ \sum_{k=0}^3 P_{Tk}(b_6) L_{6.1}(k) \right] \end{aligned} \quad (16.19)$$

For site 3 (site pattern 2), the  $L$  vectors for internal nodes 5 and 6 are specified below, and  $L_{7.1}$  is specified the way as that in Eq. (16.19) except that  $L_{5.1}$  and  $L_{6.1}$  are, respectively, replaced by  $L_{5.2}$  and  $L_{6.2}$  specified below:

$$\begin{aligned} L_{5.2}(A) &= L_{5.2}(T) = P_{ii}(b_1)P_{ij}(b_1) \\ L_{5.2}(C) &= L_{5.2}(G) = P_{ij}^2(b_1) \end{aligned} \quad (16.20)$$

$$\begin{aligned} L_{6.2}(A) &= L_{6.2}(G) = L_{6.2}(T) = P_{ij}^2(b_3) \\ L_{6.2}(C) &= P_{ii}^2(b_3) \end{aligned} \quad (16.21)$$

For site 5 (site pattern 3), the  $L$  vectors for internal nodes 5 and 6 are shown below, and  $L_7$  is specified the way as that in Eq. (16.19) except that  $L_{5.1}$  and  $L_{6.1}$  are, respectively, replaced by  $L_{5.3}$  and  $L_{6.3}$  specified below:

$$\begin{aligned} L_{5.3}(A) &= P_{ii}^2(b_1) \\ L_{5.3}(C) &= L_{5.3}(G) = L_{5.3}(T) = P_{ij}^2(b_1) \end{aligned} \quad (16.22)$$

$$\begin{aligned} L_{6.3}(A) &= P_{ii}^2(b_3) \\ L_{6.3}(C) &= L_{6.3}(G) = L_{6.3}(T) = P_{ij}^2(b_3) \end{aligned} \quad (16.23)$$

The likelihood for the three site patterns, designated as  $L_1$ ,  $L_2$ , and  $L_3$ , are

$$L_1 = \frac{1}{4} \sum_{i=0}^3 L_{7.1}(i); \quad L_2 = \frac{1}{4} \sum_{i=0}^3 L_{7.2}(i); \quad L_3 = \frac{1}{4} \sum_{i=0}^3 L_{7.3}(i) \quad (16.24)$$

where 1/4 is the equilibrium frequencies ( $\pi_i$ ) for the JC69 model. The log-likelihood ( $\ln L$ ) given the topology and the sequence alignment in Fig. 16.3 is

$$\ln L = 2 \ln(L_1) + 2 \ln(L_2) + 4 \ln(L_3) \quad (16.25)$$

which has three unknown branch lengths ( $b_1$ ,  $b_3$ , and  $b_5$ ). The  $b_1$ ,  $b_3$ , and  $b_5$  values that maximize  $\ln L$ , subject to the constraints that branch lengths cannot be negative, are  $b_1 = 0.1597105$ ,  $b_3 = 0$ , and  $b_5 = 0.3011361$ . Given the clock constraint that  $(b_1 + b_5) = (b_3 + b_6)$ , we have  $b_6 = b_1 + b_5 - b_3 = 0.460846$ . The L-BFGS-B algorithm (Zhu et al. 1997) is often used for such constrained optimization, e.g., with the lower bounds for branch lengths set to zero. The resulting  $\ln L = -27.63046$ , with  $L_1 = 0.02970894$ ,  $L_2 = 0.003665145$ , and  $L_3 = 0.09584556$ . Note that variable sites in general will have smaller likelihood than conserved sites.

Before one calibrates the clocked tree with dated fossils, it is customary to test the clock hypothesis by likelihood ratio test (LRT). Note that a tree without a clock is more general than a tree with a clock. With the former we need to estimate  $(N-2)$  more branch lengths than with the latter, where  $N$  is the number of OTUs. With four OTUs, a tree without clock will have five branch lengths to estimate, in contrast to only three in a tree without a clock. The procedure of LRT is to estimate  $\ln L$  for the tree with and without the clock, designated  $\ln L_{\text{clock}}$  and  $\ln L_{\text{no.clock}}$ , respectively. The likelihood ratio chi-square is  $2(\ln L_{\text{no.clock}} - \ln L_{\text{clock}})$  with  $(N-2)$  degrees of freedom. To discriminate between the two models, one typically needs to have much longer sequences than those in Fig. 16.3, for two reasons. First, there would be little statistical power to reject the null hypothesis if we have little data, even if the null hypothesis is false. Second, the likelihood ratio chi-square may not follow chi-square distribution when there is little data, so that resulting  $p$  value will not be accurate.

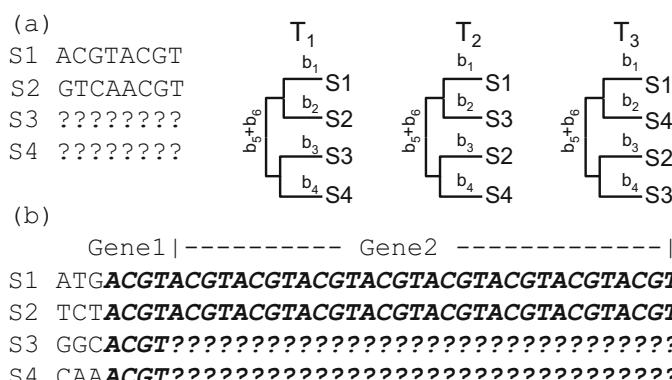
The sequences in Fig. 16.3 cannot be used to illustrate the test of molecular clock because S3 and S4 are identical and S1 and S2 diverge equally from S3/S4, so we know a priori that the sequences conform to the clock hypothesis. In other words, the clock model and the non-clock model will have the same  $\ln L$ . If we change the third site of S3 from C to T, then, again applying the JC69 model and the pruning algorithm as illustrated before for the clock and no-clock hypotheses, we will find  $\ln L$  equal to  $-31.2924$  with clock and  $-30.4874$  without clock. This gives us the likelihood ratio chi-square of  $2\Delta\ln L$  equal 1.6099. With two degrees of freedom,

$p = 0.4471$ , so we do not reject the molecular clock hypothesis. Keep in mind that this illustration has two problems because of the short sequences (sequence length of 8). First, there is little power to reject the null hypothesis. Second, approximate of  $2\Delta \ln L$  distribution by  $\chi^2$  distribution may not be accurate with small samples. Fortunately, we always have more data in real research.

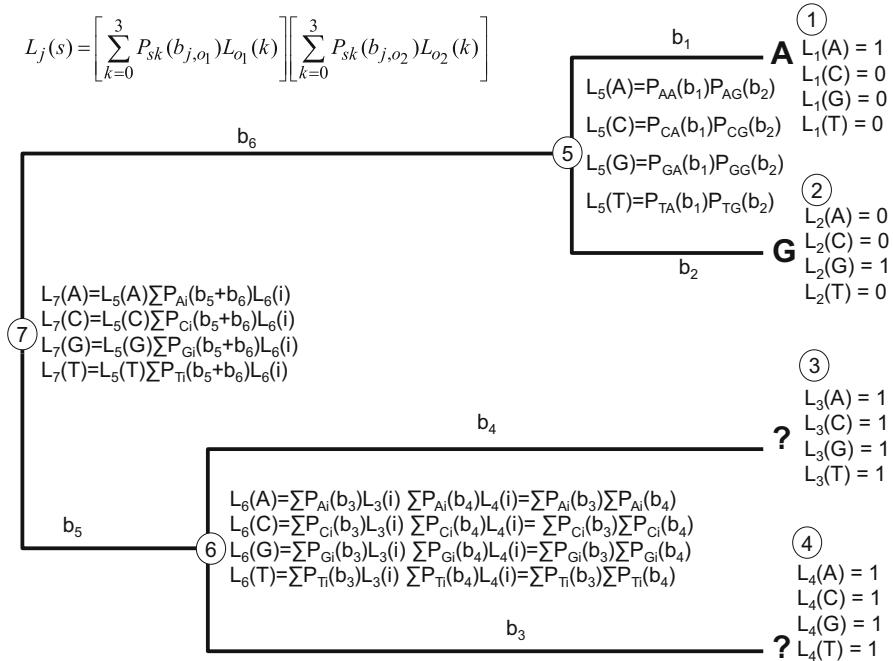
## 5 Handling of Missing Data with the Pruning Algorithm (and the Potential Bias)

The pruning algorithm facilitates the handling of missing data. The key requirement for handling missing data is that what is missing will not contribute anything to the choice of the best tree. The maximum likelihood method, when implemented properly, is not biased with missing data. However, phylogenetic bias may be induced by missing data in conjunction with rate heterogeneity over sites. I will illustrate the handling of missing data by the pruning algorithm and the potential phylogenetic bias based on a previous publication (Xia 2014).

Suppose we have data in Fig. 16.4a and need to evaluate the three possible unrooted trees ( $T_1$ ,  $T_2$ , and  $T_3$  in Fig. 16.4). It is obvious that the only information we have for the data set is the distance between S1 and S2, so we should not be able to discriminate among the three topologies. We also note that, given the JC69 model, the sequences in Fig. 16.4a have two site patterns, with the first four sites sharing one site pattern (i.e., with the same site-specific likelihood) and the last four sites sharing the other site pattern. We therefore need to compute the log-likelihood for only the first site ( $\ln L_1$ ) and the fifth site ( $\ln L_5$ ) and multiply them by 4 to get  $\ln L$  for the entire alignment.



**Fig. 16.4** Aligned sequences with missing data (a-b) for illustrating missing data handling by the likelihood method and the potential bias induced by missing data in conjunction with rate heterogeneity



**Fig. 16.5** Likelihood computation with the pruning algorithm and missing data

The computation of  $\ln L_1$ , given Topology  $T_1$ , is illustrated in Fig. 16.5. For an unknown or missing nucleotide, we simply set  $L_j(A) = L_j(C) = L_j(G) = L_j(T) = 1$ . The likelihood calculation then proceeds exactly as before. The log-likelihood ( $\ln L$ ) for all eight sites, given topology  $T_1$  in Fig. 16.4, is

$$\ln L = 4 \ln L_1 + 4 \ln L_5 \quad (16.26)$$

which, upon maximization, leads to  $b_1 + b_2 = 0.8239592165$  and  $\ln L = -21.02998149$ . Terms containing  $b_3$ ,  $b_4$ , and  $b_5 + b_6$  all cancel out, i.e., the sequences in Fig. 16.4a have no information for estimating  $b_3$ ,  $b_4$ , and  $b_5 + b_6$ , which again is what we would have expected. The resulting distance between S1 and S2 ( $= b_1 + b_2$ ) is the same if we just use the distance formula for two sequences.

If we perform the computation again with topology  $T_2$  in Fig. 16.4, we will have exactly the same  $\ln L$ , but  $b_5 + b_6$  will be 0 and  $b_1 + b_3 = 0.8239592165$  (i.e., the distance between OTUs S1 and S2 is 0.8239592165 as before). This again is perfectly consistent with our common sense. Topology  $T_3$  in Fig. 16.4 will lead to the same  $\ln L$  and the same conclusion with distance between S1 and S2 being 0.8239592165.

Our happy feeling with the likelihood method, however, does not last forever. Suppose now we have sequence data in Fig. 16.4b, with Gene1 being variable but Gene2, which is missing in S3 and S4, is so conservative as to be invariant. In

practice, Gene1 and Gene2 could be different segments within the same gene, e.g., the conserved and variable domains in ribosomal RNAs with no clear boundary between them. Note that the three variable sites at the 5'-end could be scattered over different sites in the data instead of clumping together to be as easily recognizable as in Fig. 16.4a.

The sequences are intentionally made not to favor any one of the three possible topologies in Fig. 16.4). For Gene1, the four OTUs are exactly equally divergent from each other given the JC69 or more complicated models, i.e., each pair of sequences differ in exactly one transition and two transversions so that no particular topology is favored over the other two. Gene2 is extremely conservative and no substitution has been observed, so it also should not favor any topology over the other two. If Gene2 is not missing in S3 and S4, then all three topologies will be equally supported.

With the sequence data in Fig. 16.4b and topology T<sub>1</sub> in Fig. 16.4, we can apply the pruning algorithm and the JC69 model to compute the likelihood. There are only three different site patterns with the JC69 model, i.e., sites 1–3 sharing the first site pattern, sites 4–7 sharing the second, and sites 8–39 sharing the third. Maximizing the likelihood leads to  $\ln L = -83.56464029$  which is reached when  $b_1 = b_2 = 0.04153005797$ ,  $b_3 = b_4 = 0.3787544804$ , and  $(b_5 + b_6) = 0.3511004094$ .

The maximum  $\ln L$  value for topology T<sub>2</sub> in Fig. 16.4 is  $-83.96663731$ , reached when  $b_1 = b_3 = 0.04184900$ ,  $b_2 = b_4 = 0.60765526$ , and  $(b_5 + b_6) = 0.000947018$ . The maximum  $\ln L$  value for topology T<sub>3</sub> is the same as that for T<sub>2</sub> and both are significantly smaller ( $p < 0.001$ ) than that for T<sub>1</sub> (Fig. 16.4) based on either the Kishino-Hasegawa test or RELL test (Kishino and Hasegawa 1989) or Shimodaira and Hasegawa test (Shimodaira and Hasegawa 1999).

We have previously mentioned that the most fundamental criterion for missing data handling methods is that the missing data should not contribute phylogenetically relevant information. The demonstration above shows clearly that missing data do contribute such information, even with the likelihood approach. If the data are not missing, then the three topologies will be equally supported. So the bias in favor of topology T<sub>1</sub> in the presence of missing data can only be attributed to the presence of missing data.

While the phylogenetic bias in the sequence configuration in Fig. 16.4b favors the grouping S<sub>1</sub> and S<sub>2</sub> together, one can easily envision scenarios in which S<sub>1</sub> and S<sub>2</sub> would repulse each other, e.g., when the last 32 sites in Fig. 16.4b are far more variable than the first seven sites. Thus, the direction of the bias cannot be predicted before data analysis.

## Postscript

We have covered the maximum likelihood framework in molecular phylogenetics in depth, but this book does not cover the Bayesian approach which extended the likelihood framework to incorporate prior knowledge. The Bayesian framework can not only help us with molecular phylogenetics but also reduce our tendency to develop prejudice and social bias.

Suppose we live in a multiracial society and need to decide whom our family should interact with. We implicitly would want to estimate the proportion of good people ( $P_{\text{good}}$ ) in a race (or an ethnic group), with “good people” defined as those whom we have pleasant experience interacting with. Naturally one wants to interact with people in a race whose  $P_{\text{good}}$  is high and avoid people in a race whose  $P_{\text{good}}$  is low.

Now suppose we have interacted with a small number of people, say three, in one race and our experiences are all bad. A likelihood estimate of  $P_{\text{good}}$  is then 0 because it is based on data only. If we take this estimated  $P_{\text{good}}$  seriously in spite of the small sample size of three, then we become a racist.

With the Bayesian approach, we would first conceive a prior for  $P_{\text{good}}$  before any interaction with people of different races. If we are fair-minded, our prior of  $P_{\text{good}}$  will be the same for all races to start with. If we are unfortunate to have a bad experience with a member of one race, we would reduce  $P_{\text{good}}$  for that race a bit. If our second encounter with people of this race is also bad, then we reduce  $P_{\text{good}}$  still further for that race. Eventually these different  $P_{\text{good}}$  values for different races constitute our private model of racial differences, and the model, correct or wrong, will affect our behavior.

The model of racial differences thus developed in our mind may be quite different from models in other people’s mind, because different people often interact with different samples from different races. Because few of us could claim to have a representative sample of people to interact with,  $P_{\text{good}}$  is almost always biased. However, it may not be as biased as what one gets from a likelihood framework.

In this context of unrepresentative samples from differences, racism, as well as other kinds of prejudices, is almost inevitable. What is important to keep in mind is that much of the differences in  $P_{\text{good}}$  among races or ethnic groups are due to historical differences in racial environment. If a little boy is driven by poverty to steal a loaf of bread for his sick and hungry mother, then it is the ruler of the society, not the boy, who is bad. May the joint effort of mankind lead to a monotonic increase in  $P_{\text{good}}$  in all races.

# Chapter 17

## Protein Isoelectric Point and *Helicobacter pylori*



Proteins are the workhorses in a living cell. They can perform a variety of tasks because of their diverse properties. Isoelectric point is just one of the many properties of proteins. Although we focus on isoelectric point ( $pI$ ) in this chapter, the same large-scale comparative studies illustrated here can be performed on other properties as well. DAMBE (Xia 2013, 2017d) can take a set of protein sequences and compute  $pI$  for each sequence. It also implements many other functions for describing and characterizing protein sequences.

### 1 What Is Protein Isoelectric Point and Why It Is Important to Know Its Computation

Proteins have ionizable groups such as carboxyl groups and amino groups. A carboxyl group tends to act as an acid by donating its proton ( $H^+$ ) and consequently become negatively charged. In contrast, an amino group tends to be protonated and carry positive charge. Since the electric charge of these groups depends on pH, a protein molecule can have different electric charges at different pH. The isoelectric point ( $pI$ ) of a protein is the pH at which the protein carries no net charge. The protein is positively charged if its  $pI$  is greater than the pH and negatively charged if its  $pI$  is smaller than the pH.

There are several reasons why we should understand protein  $pI$ . First,  $pI$  is important in understanding enzyme-substrate interactions. An enzyme and its substrate should not be both positively charged or both negatively charged because the two will repulse each other, although some proteins may have positively and negatively charged domains that interact with different cellular components. For example, FOXL2 protein, which is probably the earliest known sex dimorphic marker of ovarian determination/differentiation in vertebrates (Baron et al. 2004; Cocquet et al. 2003), has a positively charged DNA-binding domain to facilitate

electrostatic interaction with the negatively charged DNA backbone, but other parts are not charged or weakly charged. To know whether an enzyme (or its individual domains) is positively charged or negatively charged at a given ambient pH, we need to know how to compute *pI* of the protein.

Second, the stability of a protein often depends on the electrostatic interaction between its positively charged and negatively charged groups on the surface of protein at its physiological condition. When the pH deviates substantially from the physiological pH, the electrostatic interaction is disrupted and the protein will denature. For example, at extremely low pH (acidic), even the carboxyl group is protonated and negative charges are decreased, whereas more of the amino groups are positively charged. Such a protein will tend to lose its stability, become less compact, and finally denature.

Third, if a highly expressed protein happens to have its *pI* equal to the cytoplasmic pH, then there is no electrostatic repulsion among different copies of this protein when it is mass produced. Because the protein is not charged, its solubility is the lowest, and different copies of this protein may aggregate and precipitate, which is often bad for the cell. The “amyloid precursor protein” causing Alzheimer’s disease and the prion protein causing the mad cow disease are examples of the undesirable protein aggregation and precipitation. From an evolutionary point of view, one should expect directional selection driving the protein *pI* away from the physiological pH, and the directional selection should be strong in highly expressed proteins than in lowly expressed proteins. There are known cases where natural selection has shaped protein *pI*. For example, *Helicobacter pylori*, a bacterial species colonizing mammalian stomach, features a set of membrane proteins that are positively charged, and this positively charged membrane is likely important in alleviating the influx of protons ( $H^+$ ) in the acidic stomach fluid into the bacterial cytoplasm (Sachs et al. 2003; Xia and Palidwor 2005).

Fourth, the isoelectric point of a protein can often determine the cellular localization and function of the protein. Positively charged peptides and proteins under physiological pH can be readily taken up by the liver and kidney. Thus, a protein drug intended for liver or kidney diseases should be designed to be positively charged. Positively charged peptides and proteins can also cross the blood-brain barrier by a mechanism related to receptor-mediated transcytosis (Tamai et al. 1997; Terasaki et al. 1991). Thus, a protein drug intended for the brain should be positively charged, but should not be taken up or degraded by the liver or the kidney.

Fifth, understanding protein *pI* is also important for proteomics. Large-scale proteomic research often starts with sodium dodecyl sulfate polyacrylamide gel electrophoresis or SDS-PAGE (Laemmli 1970). Subsequent perfection of isoelectric focusing leads to the development of 2D-SDS-PAGE. While large-scale peptide analysis methods have been developed by John Yates and colleagues recently (Washburn et al. 2001; Yates 2004a, b), 2D-SDS-PAGE remains the most frequently used proteomic method. Indeed, 2D-SDS-PAGE has almost become synonymous to proteomic research (Liebler et al. 2002, p. 36). Understanding 2D-SDS-PAGE

requires an understanding of *pI*. For example, if the *pI* values of your target proteins in a cell ranges from 2 to 14 and you intend to use an isoelectric-focusing strip with a fixed pH range between 3 and 7 in your 2D-SDS-PAGE, then you know that you will miss many proteins. It is now routine for researchers to extract all annotated coding sequences in a genome, translate them in silico into proteins, obtain their molecular mass and theoretical *pI* values, and generate an in silico 2D-SDS-PAGE to improve the experimental design of a real 2D-SDS-PAGE. This in silico 2D-SDS-PAGE can also be used as the expected pattern to compare with the observed pattern on a real 2D-SDS-PAGE. Those proteins found at the same location in both the in silico gel and the real gel may be assumed to have undergone no charge-related posttranslational modification (e.g., phosphorylation and acetylation), whereas those whose coordinates do not match between the in silico gel and the real gel are good candidates for studying posttranslational modification.

There are two kinds of *pI* in literature, the theoretical *pI* and empirical *pI*. Theoretical *pI* is computed with the assumption of no posttranslational modification. Posttranslational modifications such as phosphorylation (which increases negative charge) and acetylation (which reduces positive charge) can change *pI* of a protein. A protein could have a number of phosphorylate sites that may be phosphorylated or not. For this reason, theoretical *pI* is not sufficient to predict *pI* of such proteins, and we need empirical *pI* which is experimentally determined.

This chapter begins with a brief review of the basic concepts of biochemistry related to computing theoretical protein *pI*. The method of computing theoretical protein *pI* based on computer iterations is then presented. In what follows, *pI* refers to theoretical protein *pI* unless specified otherwise. The key component of this chapter is an application of bioinformatics analysis using *pI* to understand the proteome of an acid-resistant gastric pathogen, *Helicobacter pylori*.

## 2 Computation of Protein Isoelectric Point

Computation of protein's theoretical *pI* can be done by DAMBE (Xia 2013, 2017d), with one or a set of amino acid sequences as input. However, it is beneficial to know how the theoretical *pI* is computed. The iterative computation method is detailed below.

We need to know the acid ionization constant ( $K_a$ ), also known as acid dissociation constant, and a few associated concepts. The larger the  $K_a$  value, the more readily for an acid to give up its  $H^+$ . With an ionizing reaction below involving a weak acid:



where R represents the chemical without the carboxyl (-COOH),  $K_a$  is expressed as

$$K_a = \frac{[\text{RCOO}^-][\text{H}^+]}{[\text{RCOOH}]} \quad (17.2)$$

Take the base-10 logarithm ( $\lg$ ) of both sides of Eq. (17.2), we have

$$\lg K_a = \lg [\text{H}^+] + \lg \frac{[\text{RCOO}^-]}{[\text{RCOOH}]} \quad (17.3)$$

Recall that, in chemistry, pH is defined as  $-\lg[\text{H}^+]$  and  $pK_a$  as  $-\lg K_a$ . Now the equation above becomes the well-known Henderson-Hasselbalch equation:

$$pH = pK_a + \lg \frac{[\text{RCOO}^-]}{[\text{RCOOH}]} \quad (17.4)$$

Thus, the larger the  $pK_a$  value, the less likely for a chemical (or a functional group) to give up  $\text{H}^+$ . It is important not to confuse  $pK_a$  and  $K_a$ . High  $K_a$  values imply high tendency of giving up  $\text{H}^+$ , but high  $pK_a$  values imply the opposite. My teacher told us that  $K_a$  measures the propensity of giving up  $\text{H}^+$  and that  $pK_a$  measures the propensity of retaining  $\text{H}^+$ .

Equation (17.4) is valid for other chemicals that may give up a proton, i.e., any chemical with two forms, one with, and the other without, the proton. We just need to put the form with proton in the denominator and the form without proton in the numerator as in Eq. (17.4). Thus, for weak bases such as amines, the Henderson-Hasselbalch equation is of the same form:

$$pH = pK_a + \lg \frac{[\text{NH}_2]}{[\text{NH}_3^+]} \quad (17.5)$$

Note that Eqs. (17.4) and (17.5) are the same equation, with protonated form in the denominator and the deprotonated form in the numerator. In Eq. (17.4),  $pK_a$  measures propensity of  $\text{RCOOH}$  to retain  $\text{H}^+$ , and  $pK_a$  in Eq. (17.5) measures the propensity of  $\text{NH}_3^+$  to retain  $\text{H}^+$ .

Table 17.1 lists the  $pK_a$  values for the seven ionizable amino acid residues together with the amino and carboxyl groups at the N-terminal and C-terminal, respectively. Note that these  $pK_a$  values refer to different functional groups. The  $pK_a$  value for arginine refers to its amino group ( $\text{NH}_3^+$ ) in the side chain. The high  $pK_a$  value (12.5 in Table 17.1) means a high propensity for  $\text{NH}_3^+$  to retain  $\text{H}^+$  and stay positively charged. In contrast, the  $pK_a$  value for tyrosine is associated with its hydroxyl group ( $-\text{OH}$ ) in the side chain. The high value (10.95 in the third column of Table 17.1) means that the hydroxyl group has a high propensity of retaining  $\text{H}^+$  and stay as  $-\text{OH}$  and uncharged (it would be negatively charged if it loses  $\text{H}^+$  and becomes  $-\text{O}^-$ ). Thus, arginine residues are highly likely to contribute to positive charges, but tyrosine residues are very unlikely to contribute to negative charges, although both have high  $pK_a$  values.

**Table 17.1** The ionizable residues in proteins and their approximate  $pK_a$  values. The last four columns illustrate the calculation of  $pI$  for protein polA2 from *Halobacteria* NRC1

Amino acid group	$pK_a^a$	$pK_a^b$	N	pH = 3	pH = 4.2227	pH = 5
Arginine	12.5	12.50	95	95.0000	95.0000	95.0000
Lysine	10.0	10.79	31	31.0000	31.0000	30.9999
Histidine	6.0	6.50	38	37.9880	37.8004	36.8352
Tyrosine	10.4	10.95	40	0.0000	0.0000	0.0000
Cysteine	8.3	8.30	18	-0.0001	-0.0015	-0.0090
Glutamic acid	4.1	4.25	115	-6.1226	-55.6905	-97.6374
Aspartic acid	4.1	3.91	161	-17.6374	-108.2869	-148.8972
N-terminal $\alpha$ -amino	8.0	8.56	1	1.0000	1.0000	0.9997
C-terminal $\alpha$ -carboxyl	3.1	3.56	1	-0.2159	-0.8214	-0.9650

<sup>a</sup>From Berg et al. (2002)<sup>b</sup>The average of 16 sets of values that I collected from journal papers, books, and web pages

Protein  $pI$  is typically computed by an iterative method illustrated in Table 17.1, with the polA2 protein from *Halobacteria* NRC1 and the  $pK_a$  values in the third column in Table 17.1. First, we count the frequencies of the seven ionizable amino acid residues in the protein (shown in the column headed with ‘N’ in Table 17.1). Next, for each amino acid, we compute the proportion of positively charged and negatively charged residues. For amino acids with an amino group or a carboxyl group, we may designate these proportions by  $P_{NH_3^+}$  and  $P_{RCOO^-}$ , respectively:

$$P_{RCOO^-} = \frac{[RCOO^-]}{[RCOO^-] + [RCOOH]} \quad (17.6)$$

$$P_{NH_3^+} = \frac{[NH_3^+]}{[NH_3^+] + [NH_2]}$$

It is simpler to first derive the reciprocals of the proportions, i.e.,

$$\frac{1}{P_{RCOO^-}} = \frac{[RCOO^-] + [RCOOH]}{[RCOO^-]} = 1 + \frac{[RCOOH]}{[RCOO^-]} \quad (17.7)$$

$$\frac{1}{P_{NH_3^+}} = \frac{[NH_3^+] + [NH_2]}{[NH_3^+]} = 1 + \frac{[NH_2]}{[NH_3^+]}$$

From Eqs. (17.4) and (17.5), we have:

$$\lg \frac{[RCOO^-]}{[RCOOH]} = pH - pK_a \quad (17.8)$$

$$\frac{[RCOO^-]}{[RCOOH]} = 10^{pH-pK_a}$$

$$\frac{[RCOOH]}{[RCOO^-]} = \frac{1}{10^{pH-pK_a}}$$

$$\lg \frac{[\text{NH}_2]}{[\text{NH}_3^+]} = pH - pK_a$$

$$\frac{[\text{NH}_2]}{[\text{NH}_3^+]} = 10^{pH - pK_a} \quad (17.9)$$

Now we finally have:

$$\frac{1}{P_{\text{RCOO}^-}} = 1 + \frac{[\text{RCOOH}]}{[\text{RCOO}^-]} = 1 + \frac{1}{10^{pH - pK_a}} = \frac{1 + 10^{pH - pK_a}}{10^{pH - pK_a}}$$

$$P_{\text{RCOO}^-} = \frac{10^{pH - pK_a}}{1 + 10^{pH - pK_a}} \quad (17.10)$$

$$\frac{1}{P_{\text{NH}_3^+}} = 1 + \frac{[\text{NH}_2]}{[\text{NH}_3^+]} = 1 + 10^{pH - pK_a}$$

$$P_{\text{NH}_3^+} = \frac{1}{10^{pH - pK_a} + 1} \quad (17.11)$$

$P_{\text{RCOO}^-}$  and  $P_{\text{NH}_3^+}$  in Eqs. (17.10) and (17.11) can be interpreted in two equivalent ways. For example, with N Glu residues,  $P_{\text{RCOO}^-}$  means the expected proportion of the residues that carry the negative charge at a given pH. For a single Glu residue,  $P_{\text{RCOO}^-}$  means the probability that the residue will be in a negatively charged state. Similarly, for N Arg residues,  $P_{\text{NH}_3^+}$  means the expected proportion of the residues that are positively charged at a given pH. For a single Arg residue,  $P_{\text{NH}_3^+}$  means the probability that the residue will be in a positively charged state. Note we use the same formula as in Eq. (17.10) to compute the proportion of tyrosine that carries a negative charge.

With the 95 Arg residues in Table 17.1, the number of positively charged Arg residues, given  $\text{pH} = 3$  is

$$N_{\text{Arg}^+} = N_{\text{Arg}} P_{\text{NH}_3^+} = 95 \times \frac{1}{10^{3-12.50} + 1} = 95 \quad (17.12)$$

Similarly, the number of negatively charged Asp residues out of a total of 161 in Table 17.1, given  $\text{pH} = 3$ , is

$$N_{\text{Asp}^-} = N_{\text{Asp}} P_{\text{RCOO}^-} = 161 \times \frac{10^{3-3.91}}{1 + 10^{3-3.91}} = 17.63745 \quad (17.13)$$

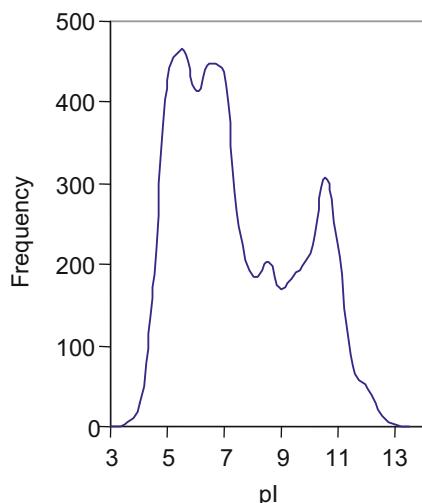
Such calculations are done for each of the amino acids to generate the fifth column in Table 17.1 headed by “ $\text{pH} = 3$ .” The negative sign is used to indicate those that carry negative charges. The summation of the column is 141.0119, which means that the protein is positively charged at  $\text{pH} = 3$  so that  $pI$  must be greater than 3. The iterative procedure then finds a pH value at which the protein is negatively charged in order to bracket the  $pI$  value between this new pH and the old pH of

3. Suppose the next value is  $\text{pH} = 5$ , and we repeat the procedure to generate the last column in Table 17.1 headed by “ $\text{pH} = 5$ .” Now the summation becomes  $-83.6737$ , i.e., the protein is negatively charged at  $\text{pH} = 5$ . Now we know that the  $\text{pH}$  at which the protein carries no net charge must lie between 3 and 5. There are efficient algorithms (Press et al. 1992) for us to try values within the  $\text{pH}$  range of (3, 5), but the most straightforward one is bisection search. That is, we get the middle  $\text{pH}$  value which is 4. The resulting summation of charges turns out to be positive, so we narrow the  $\text{pH}$  range of (3, 5) to (4, 5). Continuing this process, we will arrive at a  $\text{pH}$  value of 4.222657 to find the summation of charge to be very close to 0 (the second last column in Table 17.1 headed by “ $\text{pH} = 4.222657$ ”). Therefore,  $pI = 4.222657$ . The software DAMBE (Xia 2013, 2017d; Xia and Xie 2001b) implements this iterative procedure to compute protein  $pI$ .

### 3 Genomic *pI* Profiling

Genomic *pI* profiling refers to the computation and graphic display of *pI* for all genome-derived proteins. Fig. 17.1 is a genomic *pI* profiling for *Escherichia coli*. The saddle-shaped distribution is typical of most species from prokaryotes to eukaryotes. There is a good reason for such a saddle-shaped distribution. *E. coli* cytoplasmic  $\text{pH}$  is about eight. If there are many proteins with *pI* equal to eight, then these proteins will carry no net charge and may tend to clump together and precipitate. Thus, for proteins to stay in the cytoplasm, it is better to avoid having *pI* near cytoplasmic  $\text{pH}$ . We will deal with this issue in more detail in a later section.

**Fig. 17.1** Frequency distribution of *pI* values of genome-derived proteins from *E. coli*



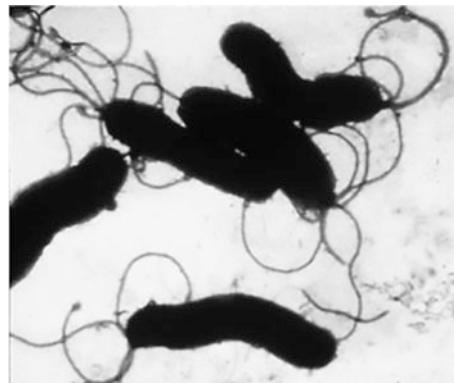
## 4 A Case Study with the Gastric Pathogen *Helicobacter pylori*

*H. pylori* (Fig. 17.2) is one of the terminal lineages in the highly invasive *Helicobacter* complex. It thrives in the acidic environment of mammalian stomach, causing gastric and duodenal ulcers and gastric cancer in human (Correa 1997; Hamajima et al. 2004; Hunt 2004; Menaker et al. 2004; Siavoshi et al. 2004).

Being an acid-resistant neutralophile (Bauerfeind et al. 1997; Rektorschek et al. 2000; Sachs et al. 1996; Scott et al. 2002), *H. pylori* is capable of surviving for at least 3 h at pH 1 with urea (Stingl et al. 2001) and maintaining a nearly neutral cytoplasmic pH between pH 3.0 and 7.0 (Matin et al. 1996; Scott et al. 2002; Stingl et al. 2002b). These properties allow it to survive and reproduce in the human stomach where the gastric fluid has a pH averaging about 1.4 over a 24-h period (Sachs et al. 2003).

The buffering action of the gastric epithelium and limited acid diffusion through the gastric mucus were previously thought to protect the bacterium against stomach acidity, but both empirical studies (Allen et al. 1993) and theoretical modeling (Engel et al. 1984) have suggested that the protection is rather limited (Matin et al. 1996; Sachs et al. 2003). Recently it has also been shown that mucus does not hinder proton diffusion, and a trans-mucus pH gradient is abolished when the luminal pH drops to <2.5 (Baumgartner and Montrose 2004). It is therefore necessary for *H. pylori* to have acid resistance mechanisms to colonize the gastric mucosa successfully (Sachs et al. 2003).

**Fig. 17.2** Microscopic image of *H. pylori* (From Paul Stokes Hoffman, University of Virginia)

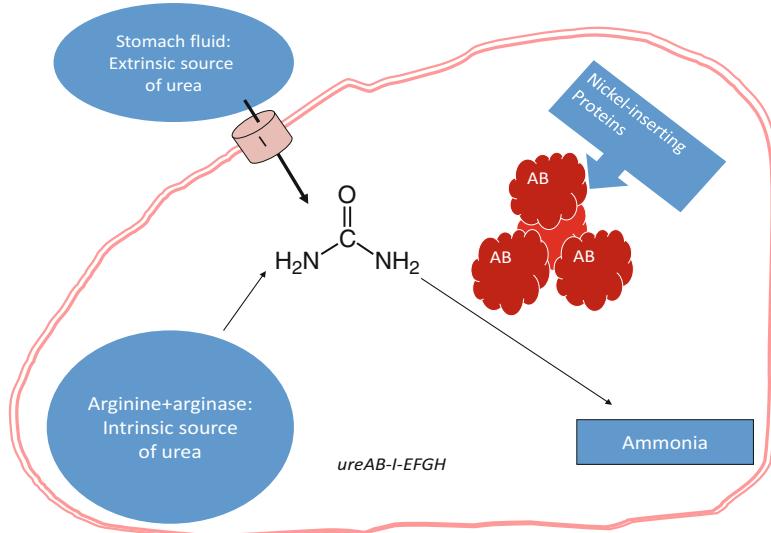


## 4.1 Two Hypothesized Mechanisms of Acid Resistance in *H. pylori*

### 4.1.1 Acid Resistance Conferred by the Urease Gene Cluster

Two mechanisms have been hypothesized to protect *H. pylori* against the acidic environment in the mammalian stomach. The first involves the urease gene cluster *ureABIEFGH* (Fig. 17.3). The constitutively expressed cytoplasmic urease, coded by *ureAB*, is made of four identical monomers each consisting of proteins A and B. The enzyme catalyzes urea to generate  $2\text{NH}_3 + \text{CO}_2$  to buffer against the  $\text{H}^+$  influx into either the periplasm or the cytoplasm (Mobley et al. 1991; Rektorschek et al. 2000; Sachs et al. 2003; Stingl et al. 2002a) and to facilitate the extrusion of  $\text{H}^+$  from the cytoplasm in the form of  $\text{NH}_4^+$  (Stingl et al. 2002a). However, urease is an apoenzyme requiring a nickel to be active. The *ureEFGH* gene cluster, whose expression is acid-induced, codes for nickel-sequestrating proteins that insert nickel into the urease, leading to increased and sustained urease activity (Sachs et al. 2003; Wen et al. 2003; Williams et al. 1996).

The urease, once activated, naturally needs a constant supply of urea as its substrate, and the cell has two sources of urea supply, one intrinsic and one extrinsic. The extrinsic source refers to urea present in saliva and stomach fluid. The exposure to gastric acid results in a large increase in urea influx into the cell due to the pH gating of the urea channel protein UreI (Bury-Mone et al. 2001; Weeks et al. 2000). The intrinsic source comes from efficient conversion of arginine to urea in the cytoplasm by the highly expressed arginase in *H. pylori* (Mendz and Hazell 1996).



**Fig. 17.3** Schematic illustration of the acid resistance mechanisms in *H. pylori* mediated by genes in the urease gene cluster *ureAB-I-EFGH*

For this reason, arginine is underused in *H. pylori* proteins, and the positively charged membrane in *H. pylori* is mainly maintained by a surplus of positively charged lysine residues (Xia and Palidwor 2005).

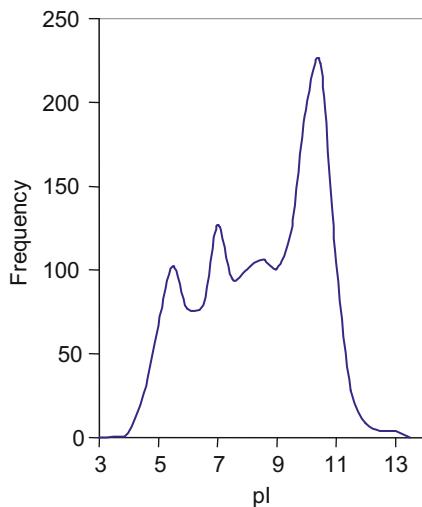
#### 4.1.2 A Positively Charged Shell to Alleviate Proton Influx

While the urease gene cluster appears to work well, researchers were puzzled by the observation that urease-negative strains of *H. pylori* can colonize gerbil stomach and induce ulcer (Mine et al. 2005). Gerbil is a model organism for studying acid resistance in *H. pylori*, and its stomach, like the stomach in other mammalian species, is also acidic. We thus are in need of a second mechanism of acid resistance which remains elusive.

One possible alternative mechanism that has been proposed is the restriction of acute proton entry across its membranes by having a high frequency of positively charged amino acids in the inner and outer membrane proteins (Sachs et al. 2003; Scott et al. 1998; Valenzuela et al. 2003). The membrane proteins have long been suspected to play an important role in acid resistance in *H. pylori* (Alm et al. 2000; Huynen et al. 1998; Solnick et al. 2004; Tomb et al. 1997; Yamaoka et al. 2002). In a recent characterization of 34 membrane proteins of *H. pylori* STR 26695 (Baik et al. 2004), four proteins (HP0243, HP0072, HP0512, and HP1563) have *pI* values ranging from 5.86 to 6.25, whereas the rest 30 have *pI* greater than 7. The average *pI* is 8.9221 for these 34 membrane proteins, whereas the average calculated *pI* value for the other 1542 proteins annotated in the genomic sequence (NC\_000915) is 8.2147. The two average *pI* values are significantly different by a two-sample t-test ( $DF = 1572$ ,  $t = 2.075$ ,  $p = 0.0382$ , two-tailed test). Thus, membrane proteins are significantly more basic than the rest of the proteins in *H. pylori* (Xia and Palidwor 2005).

*H. pylori* does have an extraordinarily high number of positively charged proteins (Fig. 17.4). The difference in the *pI* profile between *E. coli* (Fig. 17.1) and *H. pylori* (Fig. 17.4) is obvious. The conspicuous peak of basic proteins has been interpreted as a mechanism protecting the organism against its acidic environment in the mammalian stomach (Sachs et al. 2003; Xia and Palidwor 2005). Note that a protein is acidic if its *pI* is smaller than 7 and basic if its *pI* is greater than 7. However, whether the protein is positively or negatively charged depends on its environmental pH. A protein will be positively charged when the environmental pH is lower than its *pI* and negatively charged when the environmental pH is higher than its *pI*. The cytoplasmic pH in *H. pylori* can be maintained around pH near 5. Most of its proteins have their *pI* greater than its cytoplasmic pH and are consequently positively charged. In particularly, the stomach fluid where *H. pylori*'s membrane proteins are exposed has pH near 1, so almost all membrane proteins should be positively charged.

**Fig. 17.4** Frequency distribution of *pI* values of genome-derived proteins from *H. pylori*



## 4.2 Where Do Positively Charged Proteins in *H. pylori* Come From?

The positively charged proteins in *H. pylori* could arise through two mechanisms, one by modifying existing genes to increase positively charged amino acids and the other through horizontal gene transfer. *H. pylori* populations are large and have high mutation rate (Bjorkholm et al. 2001; Wang et al. 1999) and high recombination rate (Bjorkholm et al. 2001; Kersulyte et al. 1999; Suerbaum et al. 1998). These, together with the obvious directional selection mediated by the acidic environment, suggest that the evolution of existing protein-coding genes could occur fast to acquire positively charged amino acids such as Lys and Arg. On the other hand, strains of *H. pylori* are naturally competent for uptake of chromosomal DNA (Wang et al. 1999), leading to horizontal gene transfer (Alm et al. 1999; Alm and Trust 1999; Axon 1999; Censini et al. 1996; Covacci et al. 1997). Thus, the basic proteome in *H. pylori* can result either from modification of ancestrally inherited genes or from horizontal gene transfer.

To evaluate the relative importance of these two alternative but not exclusive hypotheses, we can compare genome between *H. pylori* and its sister lineage *H. hepaticus*, which is a liver pathogen, to identify homologous and nonhomologous genes. Homologous genes having higher *pI* in *H. pylori* than in *H. hepaticus* are consistent with evolution of genes inherited from their common ancestor. Nonhomologous genes having higher *pI* in *H. pylori* than in *H. hepaticus* are consistent with horizontal gene transfer. This approach to evaluate the two hypotheses is made possible by the availability of the relevant genomes. The genome of *H. pylori* strain J99 (NC\_000921) has 1492 annotated protein-coding sequences (CDSs). The related *H. hepaticus* genome (NC\_004917) has 1875 CDSs. I will refer to *H. pylori* strain J99 simply as *H. pylori* in the following sections unless otherwise specified.

#### 4.2.1 Contrast of *pI* from Homologous Genes Between *H. pylori* and *H. hepaticus*

There are 503 pairs of genes between the *H. pylori* genome and *H. hepaticus* with the same gene name and clear homology (e.g., BLAST e-value <0.000000001). The mean *pI* for these 503 genes is 7.82751 for *H. pylori* and 7.35306 for *H. hepaticus*. The difference, while small, is highly significant (paired-sample t-test:  $t = 8.5941$ ,  $DF = 502$ ,  $p < 0.00001$ ). We may conclude that a basic protein in *H. pylori* is at least partially through the evolution of ancestrally inherited genes, mainly through the increase of Lys. There is no increase in Arg because Arg is in general rarely used in *H. pylori* proteins, presumably because it is catalyzed to generate urea (Fig. 17.3).

It is important to keep in mind that the contrast above may underestimate the *pI* divergence in homologous proteins between *H. pylori* and *H. hepaticus*. For example, if we set the e-value so small that BLAST only report identical sequences, then there would be no differences in *pI* between the two species. If one increases BLAST e-value, then there will be more homologous genes. For example, out of the 1576 predicted protein-coding genes in the genomes of *H. pylori* strain 26,695, 938 found matches in the genome of *H. hepaticus*. Similarly, 941 out of 1492 predicted protein-coding genes in the genome of *H. pylori* strain J99 (NC\_000921) have matches in the genome of *H. hepaticus* (Suerbaum et al. 2003). The differences in mean *pI* increases with more divergent homologous are included.

#### 4.2.2 Contrast of *pI* from Nonhomologous Genes Between *H. pylori* and *H. hepaticus*

There are 175 genes in *H. pylori* that exhibit little homology to genes in *H. hepaticus*, and 182 genes in *H. hepaticus* that show little homology to genes in *H. pylori*. The 175 *H. pylori* genes have a mean *pI* of 8.25034 in contrast to the 182 *H. hepaticus* genes with a mean *pI* of 7.04456. The difference between the two means ( $=1.20578$ ) is much greater than that for homologous genes ( $=0.47445$ ). As I mentioned before, the difference in *pI* from homologous genes between the two species increases as more divergence homologues are included. However, the difference is never as large as that from nonhomologous sequences between the two species. One may conclude that *H. pylori* have integrated preferentially horizontally transferred genes encoding high-*pI* proteins.

The result above is corroborated by an analysis of *pI* values (Xia and Palidwor 2005) from 34 membrane proteins isolated from *H. pylori* STR 26695 by Baik et al. (2004). The average *pI* for the 34 proteins is very high ( $=8.9221$ ). What is particularly remarkable is that four proteins (HP0243, HP0072, HP0512, and HP1563) have low *pI* values ranging from 5.86 to 6.25, and they all have homologues in the *H. hepaticus* genome. In contrast, among the remaining 30 membrane proteins with  $pI > 7$ , only one has a homologue in the *H. hepaticus* genome. Thus, horizontal gene transfer has contributed much to the basic proteome in *H. pylori*.

### 4.3 Is the Basic Proteome in *H. pylori* Really an Adaptation to the Acidic Environment?

Given that *H. pylori* has proteins with higher *pI* values than *H. hepaticus* that does not live in acidic environment, one is naturally tempted to conclude that the high *pI* values in the *H. pylori* proteins represent an adaptation to the acidic environment. However, there are at least four possible hypotheses for the origin of the basic proteome in *H. pylori*.

The first hypothesis invokes natural selection and adaptation, i.e., *H. pylori* inhabits the acidic environment in the mammalian stomach with a high concentration of H<sup>+</sup>. These H<sup>+</sup> may squeeze their way into the cytoplasm and reduce cellular pH. So *H. pylori* needs a lot of positively charged proteins (especially membrane proteins) to alleviate the influx of H<sup>+</sup> into cytoplasm. It is therefore beneficial for the organism to accumulate positively charged amino acid residues in its proteins, especially in its membrane proteins. This hypothesis is known as the acid-adaptation hypothesis or AAH (Xia and Palidwor 2005), i.e., *H. pylori* acquired its high-*pI* proteins as an adaptation in response to the acidic environment.

The second hypothesis argues that parasitic bacterial genomes typically evolve toward AT richness because spontaneous mutations are generally AT-biased based on comparisons between pseudogenes and their functional counterparts (Gojobori et al. 1982; Li 1983; Li et al. 1981). *H. pylori* has a relatively AT-rich genome, e.g., the genomic GC% of *H. pylori* 26,695 is only 38%, in contrast to the genomic GC% of 50% in *E. coli* substr DH10B. This potentially mutation-mediated AT richness will lead to an increase in A-rich codons such as the lysine codon AAA and AAG (and a consequently increased usage of lysine). The increased lysine usage in proteins then increases protein *pI*. Because *H. pylori* and its sibling species are all parasites, their most recent common ancestor might have already practiced parasitism and acquired AT richness and increased frequency of lysine codons before it became a parasite in the mammalian stomach. Therefore, with an overrepresentation of its lysine residues in its proteins, it is already preadapted to acidic environment (if positively charged residues indeed contribute to acid resistance). This hypothesis is termed exaptation hypothesis (Xia and Palidwor 2005) following the terminology by Gould and Vrba (1982), i.e., the process in which an originally neutral trait has subsequently acquired a beneficial function. A well-known example of exaptation is the brain-specific RNA gene BC200 resulting from the exaptation of a presumably neutral SINE repeat (Smit 1999).

The third hypothesis states that nucleotide C is rare in mammalian cells and a mammalian parasite should therefore minimize the usage of C as a building block of its RNA and DNA. Minimizing C in an organism with a DNA genome has the necessary consequence of reduced G, with a consequent increase in A and T. This will also contribute to increase AT and increased lysine codon. Thus, originally an adaptation to a C-rare environment predisposed the organism to tolerate an acidic environment (if positively charged residues indeed contribute to acid resistance). Such a mechanism is called preadaptation in evolution (Xia and Palidwor 2005), i.e.,

a trait originally selected for one function but that subsequently gained a different function beneficial to the carrier of the trait. An often cited example of preadaptation is the rudimentary feather that presumably has been selected for thermoregulation in nonavian dinosaurs but preadapted their carriers to subsequent evolution of flight.

The fourth hypothesis is more complicated. As mentioned previously, a protein in a solution with a pH equal to the protein  $pI$  does not carry net electric charge. If highly expressed proteins happen to have their  $pI$  equal to the cytoplasmic pH, then there is no electrostatic repulsion among these proteins when they are mass produced. Because the proteins are not charged, their solubility is at the lowest, and they may aggregate and precipitate, which is often harmful to the cell. The “amyloid precursor protein” causing Alzheimer’s disease and the prion protein causing the mad cow disease are examples of the undesirable protein aggregation and precipitation. *H. pylori* living in an acidic environment maintains its cytoplasmic pH around 5. This suggests that *H. pylori* proteins should avoid having  $pI = 5$ . It is biochemically and evolutionarily easier for *H. pylori* proteins to shift their  $pI$  upward than downward to avoid having  $pI = 5$ , hence the increase in protein  $pI$  overall. This hypothesis is termed precipitation avoidance hypothesis (PAH).

The first three hypotheses have already been tested and AAH (acid-adaptation hypothesis) was declared the winner (Xia and Palidwor 2005). However, PAH has not been considered as an alternative before.

#### 4.4 Discriminate Between AAH and PAH

If AAH is correct, i.e., *H. pylori* needs a lot of membrane proteins carrying positively charged residues to alleviate influx of  $H^+$ , then we expect membrane proteins to carry more positively charged residues than cytoplasmic proteins. In contrast, PAH predicts that highly expressed proteins should evolve  $pI$  away from cytoplasmic pH and become positively charged (a protein would need to have an inordinately large number of negatively charged residues to reduce  $pI$  below the cytoplasmic pH of ~5).

To test the prediction from AAH, we can classify proteins into membrane proteins and cytoplasmic proteins by using programs such as PSORT (Nakai and Horton 1999; Peabody et al. 2016) which uses experimentally verified protein cellular localization data for prediction. The cytoplasmic proteins have substantially smaller mean  $pI$  than the three groups of membrane proteins (Table 17.2). This is consistent with AAH which predicts that membrane proteins should be driven to increase positively charged residues more than the cytoplasmic proteins.

Results in Table 17.2 alone are insufficient to reach a conclusion concerning AAH. For example, it is possible that membrane proteins in general have high  $pI$  than cytoplasmic proteins in bacterial species, regardless of whether the species lives in acidic environment or not. For example, it is possible that bacterial species that inhabit an environment with neutral pH may also have high  $pI$  for membrane proteins than cytoplasmic proteins. For this reason it is essential to contrast the

**Table 17.2** *pI* statistics among four groups of proteins, together with ANOVA output

Groups		Count	Mean <i>pI</i>	Variance	
Cytoplasmic membrane		296	9.1167	2.4680	
Outer membrane		43	9.3281	1.8902	
Periplasmic		16	9.1500	2.8344	
Cytoplasmic		807	7.6426	3.4820	
ANOVA					
Source of variation	SS	DF	MS	F	P-value
Between groups	557.2659	3	185.7553	58.8286	0.0000
Within groups	3656.4625	1158	3.1576		
Total	4213.7284	1161			

**Table 17.3** *pI* statistics among four groups of proteins in *H. hepaticus*

Groups		Count	Mean <i>pI</i>	Variance	
Cytoplasmic membrane		356	8.0315	3.2819	
Outer membrane		26	8.3919	2.2434	
Periplasmic		25	8.1044	4.1021	
Cytoplasmic		896	7.2124	3.4181	
ANOVA					
Source of variation	SS	DF	MS	F	P-value
Between groups	203.7982	3	67.9327	20.1526	0.0000
Within groups	4378.8106	1299	3.3709		
Total	4582.6088	1302			

pattern in *H. pylori* against that in *H. hepaticus* (Table 17.3). Indeed cytoplasmic proteins have lower mean *pI* than membrane proteins in *H. hepaticus* as well. However, the difference in mean *pI* between cytoplasmic proteins and membrane proteins is smaller in *H. hepaticus* than in *H. pylori*, and this difference is significant statistically ( $p < 0.05$  for “species by groups” interaction). The results are therefore consistent with the prediction of AAH.

## Postscript

Organisms can survive in challenging environment such as the extreme acidity in mammalian stomach because of the harmonic interaction of their genes and gene products shaped by natural selection. The importance to understand the relationships among genes and their products, or among interacting partners in general, has been highlighted by using the Greek legend of the Delphi boat (Danchin 2002). Delphi is the location where Apollo, the son of the Greek god Zeus, established a priesthood. The female priests could often enter into a mental state in which they were prone to

murmur an encoded prophecy of the future called an oracle. For this reason the word Delphi and Oracle have become associated with prophecy of the future. Both terms have been used as names of software by companies that are confident enough to project their software as the prophecy of the future. The female priests might also ask passersby questions, and one of the question is whether a boat made of wooden planks remains the same boat after all of its original planks have gone rotten over time and been sequentially replaced by new planks.

From the owner's point of view, the boat is the same boat because the planks still have the same relationship to each other. It is not the individual planks that define the boat but the relationships among planks that define the boat. While having a list of planks is an essential step in building a boat, it is only after we have identified the relationship among planks can we actually build the boat.

The relevance of the Delphi boat to proteomics and genomics is that, even if we have a complete list of genes (or proteins), and perfect description of each of them, we will still not be able to build the boat until we know the relationship among the proteins and genes. In fact, we still know little of alternative splicing and consequently can hardly predict mature mRNAs from their precursors. Biologists roughly know that there are boats (organisms) made of similar set of planks (genes) and that some planks taken from one of these boats can often replace an equivalent plank in another boat without evil effect, but they are still far from building a boat de novo.

# Chapter 18

## Bioinformatics and In Silico 2D Gel Electrophoresis



2D-SDS-PAGE remains arguably the most frequently used proteomic method (Liebler et al. 2002, p. 36). The 2D separation process can be simulated on computer so that one can obtain an *in silico* gel with expected pattern of protein separation. This simulation is implemented in DAMBE (Xia 2013, 2017d), and this chapter presents methodological details of the implementation.

### 1 Scientific Rationale Behind the 2D-SDS-PAGE

2D-SDS-PAGE separates proteins by their differences in isoelectric point (pI) and molecular mass. It first separates proteins by isoelectric focusing (IEF), i.e., by using immobilized pH gradient (IPG) strips that are available commercially. When loaded onto the strip under an electric field, proteins will migrate to the location of the strip with the pH equal to their pI values. At that location proteins do not carry any net charge and consequently will not move in the electric field. A protein wandering out of the location will carry charges again and will be forced back to the location by the electric field.

The second dimension of protein separation in 2D-SDS-PAGE is by molecular mass. Proteins are denatured by mercaptoethanol or DDT or the like. SDS (sodium dodecyl sulfate) then binds and imparts negative charge to proteins in roughly constant proportion to the length of the protein. This implies that each amino acid residue will be pulled roughly by the same electric force, and longer proteins, being clumsier in passing through porous materials, migrate more slowly than shorter ones. This resolves the protein mixture by molecular mass.

The DNA sequence in a living cell typically codes for many proteins. Prokaryotes typically have hundreds or thousands of protein-coding genes in their genomes. The genome of the yeast, *Saccharomyces cerevisiae*, which is a unicellular eukaryote, contains about 6000 protein-coding genes. Human genome contains about 30,000 protein-coding genes. However, the limited number of genes in the genome of

multicellular eukaryotes can generate a huge number of different proteins through alternative splicing (Ast 2004). Even a single *Dscam* gene in *Drosophila melanogaster* can generate 38,016 protein variants through alternative splicing (Graveley 2005; Schmucker et al. 2000). Displaying and detecting these different proteins on a gel is by no means trivial. It is valuable for one to have an in silico gel with expected protein separation patterns as a guide for experimental optimization and as a reference to compare against the observed separation pattern on a 2D gel.

## 2 In Silico Gel with Expected Separation Pattern

Given a set of protein sequences, we can compute pI and molecular mass for each protein and place them at proper location of an in silico gel. Such a set of protein sequences can be obtained from a well-annotated genome. For example, one can extract all protein-coding genes in the yeast genome, translate them into protein sequences, and generate an in silico gene in seconds by using software DAMBE (Xia 2013, 2017d).

We have already learned how to compute the pI value of a known protein sequence. Here we learn how to compute the molecular mass of an amino acid sequence. It is important to know that the molecular mass of an amino acid sequence is not the summation of the molecular mass of all constituent amino acids, because a peptide is formed by the end-to-end condensation of amino acids with loss of water. The molecular mass of the resulting amino acid residues (Table 18.1) in a peptide, taken from Caltech's Protein/Peptide MicroAnalytical Laboratory at <http://www.its.caltech.edu/~ppmal/>, is consequently smaller than the molecular mass of intact amino acids by about 18 (i.e., molecular mass of H<sub>2</sub>O).

One may note that Ile and Leu have identical residue mass, and Gln and Lys have very similar residue mass. They cause problems in de novo sequencing of proteins by mass spectrometry (Carroll et al. 2003). Bioinformatic tools to alleviate such problems will be presented in a latter chapter on proteomics.

The molecular mass of a peptide is the summation of the molecular mass of the constituent residues plus an extra proton (with a molecular mass of 1) at the N-terminal and an extra –OH (with a molecular mass of 17) at the C-terminal. Thus, the molecular mass of an amino acid sequence AACAGGRQD is 847.893.

The migration distance (*D*) can be expressed as a function of protein molecular mass (*M*). The following equation appears general enough to fit the relationship between *D* and *M* very well:

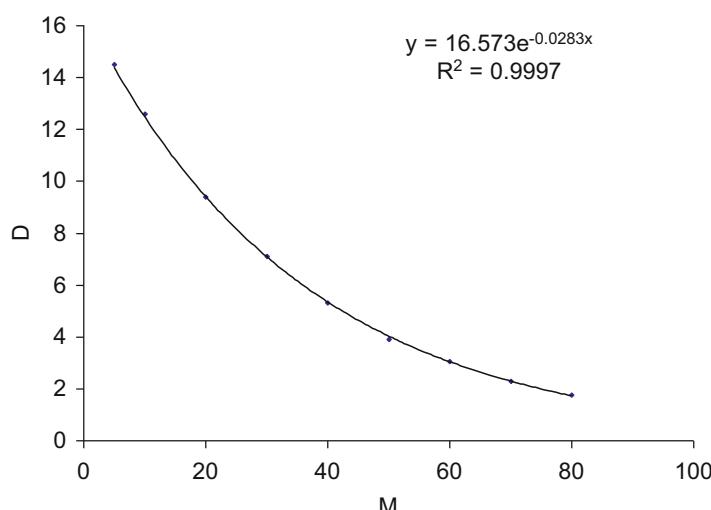
$$D = ae^{bM} \quad (18.1)$$

where *a* and *b* are constants that can be estimated by a simple linear regression of observed *D* on *M* values, after log transformation of the two sides of the equation. Figure 18.1 shows the relationship between *D* and *M*, with *a* and *b* estimated to be

**Table 18.1** Molecular mass of amino acid residues

AA3	AA1	AA mass	Residue mass
Gly	G	75.07	57.052
Ala	A	89.10	71.079
Ser	S	105.10	87.078
Pro	P	115.13	97.117
Val	V	117.15	99.133
Thr	T	119.12	101.105
Cys	C	121.20	103.144
Ile	I	131.17	113.160
Leu	L	131.18	113.160
Asn	N	132.10	114.104
Asp	D	133.10	115.089
Gln	Q	146.15	128.131
Lys	K	146.19	128.174
Glu	E	147.10	129.116
Met	M	149.21	131.198
His	H	155.16	137.142
Phe	F	165.19	147.177
Arg	R	174.20	156.188
Tyr	Y	181.19	163.170
Try	W	204.23	186.213

AA3 and AA1 refer to the three-letter and one-letter notation of amino acids

**Fig. 18.1** Relationship between migration distance (*D*) and molecular mass (*M*)

16.573 and  $-0.0283$ , respectively, based on my sampling of a subset of secreted proteins of the gastric pathogen *Helicobacter pylori* (Bumann et al. 2002).

Now that we have protein pI, molecular mass, and migration distance based on the molecular mass, the location of the protein on the 2D gel is defined. However, there is still one thing missing. A protein dot in a real 2D-SDS-PAGE gel always features at least three types of information. That is, not only does it show the protein pI and the migration distance, it also reveals the protein abundance (i.e., big dots for highly expressed proteins and small dots for lowly expressed proteins). While we already know how to draw a protein dot on the 2D graph by using protein pI as the X-coordinate and D as the Y-coordinate, we need a measure of protein abundance so that different proteins will have different dot size. The in silico gel would not look professional if all dots were of the same size.

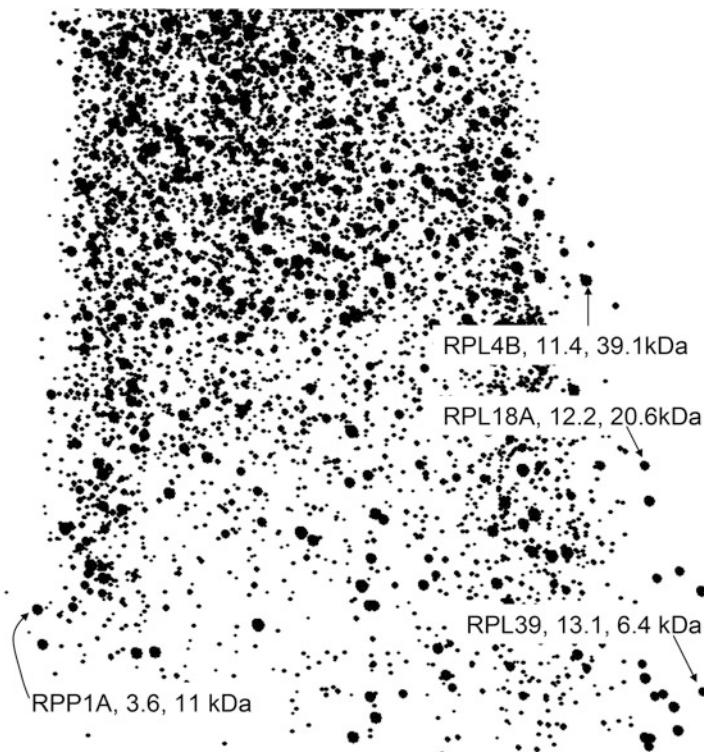
We know that highly expressed proteins exhibit strong codon usage bias. Thus, we can use an index of codon usage bias as an approximate measure of the dot size. This is not ideal, but is used for the lack of anything better. We have learned a number of codon usage indices for measuring codon usage bias, such as codon adaptation index or CAI (Sharp and Li 1987) or  $I_{TE}$  (Xia 2015). The in silico 2D-SDS-PAGE in Fig. 18.2 is produced in this way by using CAI as a measure of protein abundance, i.e., the protein-coding gene with a large CAI value will have a large dot. It includes all protein-coding genes longer than 100 in the budding yeast, *Saccharomyces cerevisiae*.

The in silico 2D-SDS-PAGE (Fig. 18.2) reveals a set of highly expressed, positively charged, and relatively small proteins on the lower right of the gel. Many of these proteins contain a positively charged binding domain that may interact electrostatically with DNA and RNA (recall that DNA and RNA, with the phosphate backbone, are negatively charged and therefore exhibit affinity to positively charged proteins).

The protein pI values in *Saccharomyces cerevisiae* range from 3.2 to 13.1. This suggests that one should use an isoelectric focusing strip covering this range in a real 2D-SDS-PAGE. The in silico 2D-SDS-PAGE for *E. coli* exhibits a similar pattern with a number of small and positively charged proteins.

One may think that, if the yeast proteins already generate a rather crowded in silico 2D-SDS-PAGE, then it would be totally messy to display a much large number of possible proteins in a multicellular eukaryote. This concern seems unnecessary. Recent gene expression studies (e.g., Velculescu et al. 1999) have demonstrated that tissue-specific genes in multicellular organisms are relatively few. By taking advantage of such gene expression studies, we can produce tissue-specific in silico 2D-SDS-PAGE much simpler than that in Fig. 18.2.

Proteins are first separated by isoelectric focusing and then by protein molecular mass. Recall that a protein will carry no net charge at pH equal to the protein's pI and consequently will not move in an electric field. Thus, if we do electrophoresis of proteins along a pH gradient, then proteins will migrate into the location where the pH matches their pI values, so proteins with different pI will become separated along the pH gradient.



**Fig. 18.2** In silico 2D-SDS-PAGE of genome-derived proteins in the budding yeast, *Saccharomyces cerevisiae*, with four protein dots labeled. The annotations are in the form of “Gene name, pI, molecular mass.” RPL4B, RPL18A, and RPL39 are protein components of the large (60S) ribosomal subunit, all with high pI values and positively charged, which indicates the possibility of their electrostatic interaction with the negatively charged ribosomal RNA. RPP1A (acidic ribosomal protein P1 $\alpha$ ) is a component of the ribosomal stalk involved in the interaction between translational elongation factors and the ribosome. Produced with DAMBE (Xia 2013, 2017d)

In Chap. 10 we have briefly mentioned how to generate an expected pattern of proteins separated by their differences in isoelectric point and molecular mass, by assuming no posttranslational modification. This can then be compared with the actual separation pattern obtained by using such bioinformatic tools that are valuable for identifying proteins that are the products of alternative splicing or that have undergone posttranslational modification.

Alternative splicing has now been recognized as the most fundamental mechanism in generating the complexity of multicellular eukaryotes. A limited number of protein-coding genes in multicellular eukaryotic genomes can generate a huge number of different proteins through alternative splicing (Ast 2004). Even a single *Dscam* gene in *Drosophila melanogaster* can generate 38,016 protein variants through alternative splicing (Graveley 2005; Schmucker et al. 2000). While alternative splicing is long known for generating the diversity of immunoglobulins, recent

studies have shown that it affects the expression of many other genes (Kazan 2003; Lipscombe 2005; Stamm et al. 2005), with an estimate of up to 60% of human genes (Kornblhtt 2005; Lee and Wang 2005) being involved in alternative splicing. Large-scale transcriptomic approaches such as microarray (Diehn et al. 2000; Epstein and Butow 2000; Gaasterland and Bekiranov 2000; Holstege et al. 1998; Schena 1996, 2003) or SAGE (Madden et al. 1997; Saha et al. 2002; Velculescu et al. 1995, 1997; Zhang et al. 1997) experiments are rather limited in detecting or predicting protein products resulting from alternative splicing, and direct proteomic methods are needed to characterize the diversity of protein products in multicellular eukaryotic cells.

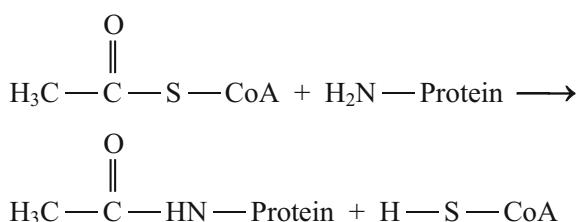
Protein posttranslational modifications represent another biochemical mechanism contributing to the diversity of proteins in living cells. They typically involve changes in molecular mass or in charge. Such modified proteins will migrate to locations on the gel different from what we expect assuming no modification. Therefore, those proteins found at the same location in both the in silico gel and the real gel may be assumed to have undergone no posttranslational modification, whereas those whose coordinates do not match between the in silico gel and the real gel are good candidates for studying posttranslational modification.

The scientific rationale of the 2D-SDS-PAGE is outlined first in this chapter for readers not familiar with the method. This is followed by the simple computation needed to generate the in silico 2D-SDS-PAGE. We have already learned how to obtain the isoelectric point (pI) for each protein in the previous chapter. We only need to learn how to obtain the values for the other dimension (i.e., the molecular mass of the protein) and how to graphically present the results.

### 3 Differences Between the In Silico and the Actual 2D Gel

Many proteins undergo posttranslation modifications (PTM). Some PTMs will alter charge, e.g., phosphorylation and acetylation (Fig. 18.3). Many proteins are not functional until they have undergone posttranslational modification. Some enzymes, in their nascent state, carry the same positive charge as their substrates and cannot function well because of electric repulsion. Phosphorylation adds negative charges to such enzymes and allows them to electrostatically interact with positively charged substrate. Such changes in charge will alter pI and consequently will change the gel

**Fig. 18.3** Acetylation.  
Acetyl coenzyme A (acetyl-CoA) attaches to the amino group of an amino acid residue (e.g., lysine, whose amino group typically exist in the form of  $\text{--NH}_3^+$ )



location of the protein. Such deviation from the expected location can be used to identify proteins that have undergone PTMs.

Some PTMs change molecular mass, such as the eukaryote-specific glycosylation which involves the attachment of sugar molecules to specific amino acids in a polypeptide chain. This modification changes the molecular mass resulting in a change of the protein in the location of the 2D gel. Because small sugar molecules are water soluble, glycosylated proteins are also more soluble in water. Glycosylation exists in two forms. The N-glycosylation involves the attachment of sugars to the nitrogen in the side chain of the amino acid asparagine (Asn). It requires a characteristic sequence of Asn-Xaa-Ser or Asn-Xaa-Thr where Xaa is any amino acid except proline. It is interesting that some N-glycosylation sites are never used, indicating that amino acids flanking the N-glycosylation sites might also be important. One can use the perceptron algorithm detailed in Chap. 5 to investigate this possibility, by collecting two sets of peptide sequences containing the N-glycosylation sites together with, say, 10 flanking amino acids. One set (the positive group) would include all those known to have undergone N-glycosylation in some tissue or at certain developmental stage, and the other set (the negative group) would include all those “N-glycosylation sites” that have never been N-glycosylated. Running the perceptron algorithm or its multilayer derivatives should shed light on the differences between the two groups. The other form of glycosylation, called O-glycosylation, involves the attachment of sugars to the oxygen in the side chain of the amino acids serine or threonine. No specific sequence motif is associated with this form of glycosylation.

Glucagon production serves as another example of mass-altering PTMs. The gene coding for glucagon is transcribed and translated into proglucagon in both pancreas and intestine. However, proglucagon is cut to produce glucagon in the pancreas, but glucagon-like peptides in the intestine. A similar example is the differential production of different forms of somatostatin. Somatostatin SS-14 is secreted from pancreas, whereas SS-28 is the predominant form produced in the intestine. Both SS-14 and SS-28 are derived from the same prosomatostatin which in turn is derived from preprosomatostatin. Such mass-changing PTMs, just like pI-changing PTMs, will result in gel locations of proteins different from the expected patterns in an in silico gel.

Most PTMs change both protein pI and molecular mass. For example, the amino group in the lysine is normally positively charged, but acetylation (Fig. 18.3) results in the loss of the positive charge and a consequent decrease in pI. The modification also decreases migration distance because of the increased molecular mass. Acetylation plays a crucial role in gene expression. Eukaryotic genomic DNA typically wraps itself around a group of proteins in the nucleus called histones to form nucleosomes. Histones are rich in lysine residues and are consequently positively charged. These positively charged lysine residues interact electrostatically with the negatively charged DNA backbone to facilitate the formation of nucleosomes. RNA polymerases, which transcribe RNA from DNA, generally cannot access transcription start site when DNA is tightly wrapped around histones. Acetylation of the lysine residues removes the positive charge and facilitates nucleosome melting to

allow gene transcription. Acetylation is observed only in eukaryotes. Aside from histones in the nucleus, about 59% and 90% of proteins in the cytoplasm are acetylated, mostly at the N-terminus. Proteins with its N-terminus acetylated are more resistant to degradation, with their lifetime extended from between seconds and hours up to days. Prokaryotic proteins, mitochondrial proteins, and chloroplast proteins are not acetylated.

In vertebrates, when DNA is methylated at the 5-carbon of nucleotide C in CpG dinucleotides, these methylated CpG will attract methyl-CpG-binding domain (MBD) of a number of proteins such as MBD1, MBD2, MBD3, and MeCP2. The resulting protein-DNA complex will recruit histone deacetylase which will remove acetyl-CoA from lysine residues to restore the positive charge of the amino group. These positively charged lysine residues then allow histones to bind tightly to DNA to prevent transcription.

# Chapter 19

## Fundamentals of Proteomics



### 1 Introduction

The terms “proteome” and “proteomics” were coined by Marc Wilkins and colleagues in 1994 (Ezzell 2002), with proteomics referring specifically to studies of proteins using the method of mass spectrometry (MS). Proteomics has since become one of the two major components of what is now known as protein science (e.g., Lesk 2004), with the other component being protein structure determination, typically by X-ray crystallography and nuclear magnetic resonance.

Mass spectrometry used in combination with affinity purification and/or chemical cross-linking has made significant contributions to protein interaction networks (Figeyns 2003a, b; Vasilescu and Figeyns 2006). Protein arrays have recently been developed to directly assess protein-protein interactions (Figeyns 2002; Sloane et al. 2002; Wilson and Nock 2002). Ultimately, all proteins, isolated either by conventional 2D gel, protein arrays, or other affinity purification methods, have to go through MS for protein identification. For this reason, protein identification is the most fundamental in proteomics.

Large-scale research of proteins started with sodium dodecyl sulfate (SDS) polyacrylamide gel electrophoresis, SDS-PAGE (Laemmli 1970). Subsequent perfection of isoelectric focusing leads to the most frequently used protein separation method, the 2D-SDS-PAGE. Large-scale peptide analysis methods have been developed by John Yates and colleagues (Washburn et al. 2001; Yates 2004a, b). Mass spectrometry used in combination with affinity purification and/or chemical cross-linking has made significant contributions to protein interaction networks (Figeyns 2003a, b; Vasilescu and Figeyns 2006). Protein arrays have recently been developed to directly assess protein-protein interactions (Figeyns 2002; Sloane et al. 2002; Wilson and Nock 2002).

Protein abundance data for thousands of proteins are now available for a number of model organisms through databases such as PaxDB (Wang et al. 2012), paving the way for comparative proteomic analysis. For example, relative abundance of

bacterial release factors (RF1 and RF2) differs much among bacterial species and correlates well with stop codon usage, i.e., stop codon UGA usage increases with the relative abundance of RF2 which decodes UGA (Wei et al. 2016).

Methodologically, proteomics involves separating proteins, digesting individual proteins into small peptides, performing mass spectrometry (MS) to obtain the molecular mass of the peptides, identifying the proteins based on the molecular mass data, and quantifying protein abundance based on the number of peptide fragments matched to the genome-encoded proteins. Sometimes the entire protein sample is digested into peptides without first separating the proteins.

Protein identification through peptide molecular mass and charge is made possible by the availability of many genomic sequences or large collections of known protein sequences. Protein identification used to require protein sequencing, which is tedious. Fortunately, modern computational methods can link the peptide mass data to proteins quite easily and accurately, by using what is now known as the peptide mass fingerprinting (Cottrell 1994; Cottrell and Sutton 1996; James et al. 1994; Pappin et al. 1993; Sutton et al. 1995). With the availability of many well-annotated genomic sequences, it is now also easy to map the identified proteins to locations on the genomic sequence (Sommerer et al. 2006; Webster and Oxley 2005). The method is traditionally associated with the first proteomic approach, i.e., separating the proteins and analyze individual proteins by MS. However, large-scale peptide analysis methods have been developed by John Yates and colleagues (Washburn et al. 2001; Yates 2004a, b).

## 2 Protein Mass Spectrometry

All MS instrumentations include two essential components, an ionizer that generates gaseous ions from a sample and a mass analyzer that generates the number as well as the mass/charge ratios, i.e.,  $m/z$  values of each type of ions. A computational protocol called charge deconvolution (or just deconvolution) uses the  $m/z$  values to produce the estimated molecular mass of the molecule of interest, e.g., protein or peptide. Deconvolution is explained in the next section.

Two types of ionizers are used frequently in proteomics (Lesk 2004; Liebler et al. 2002), the matrix-assisted laser desorption ionization (MALDI) and electrospray ionization (ESI). The MALDI ionizer generates predominantly singly charged ions (which may be positive or negative but generally only positive ions are analyzed by MS) and used most often in peptide mass fingerprinting because of its high accuracy in measuring peptide mass. An extension of the MALDI ionizer, termed surface-enhanced laser desorption/ionization or SELDI (Forde and McCutchen-Maloney 2002; Tang et al. 2004; Wright 2002; Yip and Lomas 2002), has recently been developed for identifying proteins of different sizes on protein arrays.

The ESI ionizer generates ions carrying different charges because peptides or proteins ionized by ESI ionizers are in aqueous solutions. Recall that proteins will carry a net charge when the pH of the solution differs from the protein isoelectric

point (pI, review Chap. 10 if you have forgotten the computation of pI). They become more and more positively charged in acidic solutions and more and more negatively charged in basic solutions. For MS analysis, the peptides are typically ionized to carry positive charge in acidic solutions. Because a protein or a peptide may have multiple lysine, arginine, and histidine residues, the resulting peptide or protein ions may consequently carry multiple charges. For example, a peptide with molecular mass of  $m$  may have  $n$  different types of charged ions from the ESI ionizer, designated as  $\text{ion}_1, \text{ion}_2, \dots, \text{ion}_n$ , carrying  $z_0, (z_0 + 1), (z_0 + 2), \dots, (z_0 + n - 1)$  positive charges (protons), respectively. Note that we have designated  $z_0$  as the charge of the least-charged ion.

Each proton adds one atomic mass to the ion, so the actual molecular masses of  $\text{ion}_1, \text{ion}_2, \dots, \text{ion}_n$  carrying  $z_0, (z_0 + 1), (z_0 + 2), \dots, (z_0 + n - 1)$  protons are  $(m + z_0), (m + z_0 + 1), (m + z_0 + 2), \dots, (m + z_0 + n - 1)$ , respectively. However, MS does not measure the molecular mass of the ions directly. Instead it outputs the  $m_{\text{ion}}/z_{\text{ion}}$  ratios where  $m_{\text{ion}}$  is the mass of the ion (i.e.,  $m$  plus the total mass of extra protons) and  $z_{\text{ion}}$  is the positive charge of the ion (the number of the protons). Given the  $n$  types of differently charged ions, an ESI-MS will output  $(m + z_0)/z_0, (m + z_0 + 1)/(z_0 + 1), (m + z_0 + 2)/(z_0 + 1), \dots, (m + z_0 + n - 1)/(z_0 + n - 1)$ . Charge deconvolution in the next section takes this output to estimate  $m$  and  $z_0$ .

Aside from an ionizer, a MS will always have a mass analyzer. Frequently used mass analyzers are time of flight (TOF), quadrupole, or ion trap analyzers. Different MS instrumentations are often specified by the combination of the ionizer and the mass analyzer. For example, MALDI-TOF MS is a frequent combination for peptide mass fingerprinting after digesting proteins into small peptides, and SELDI-TOF MS is used typically for large-scale protein identification in protein arrays involving proteins over a wide mass range.

A mass analyzer has fixed measurement range of  $m/z$  values. If the maximum  $m/z$  range for a given mass analyzer is 2000, then the analyzer can measure the mass of a peptide up to the molecular mass of 2000 when ions are singly charged, as is the case with MALDI ionizer. If ions carry multiple charges, as in the case with ESI ionizers, then the same mass analyzer can be used to measure the molecular mass of much larger peptides or even entire proteins. For example, if the ion mass is 10,000 but it carries 10 positive charges, then its  $m/z$  ratio is only about 1000, well within the measurement range of the mass analyzer. For this reason, MS with an ESI ionizer is able to measure the molecular mass of much larger molecules than that with a MALDI ionizer.

There are excellent descriptions of MS hardware in the proteomic framework (e.g., Liebler et al. 2002) for readers who are interested in MS hardware. I will focus only on what is important but missing in other books on proteomics.

### 3 Charge Deconvolution

Deconvolution in MS literature refers to the protocol of computing the molecular mass from the distribution of multiply charged ions of the molecule of interest. Such multiply charged ions are typical in MS with an ESI ionizer. MS data obtained with a MALDI ionizer do not need charge deconvolution because ions are predominantly singly charged and the peptide mass can be derived directly as the  $m/z$  ratio minus the proton mass (which is 1).

Let us start with a simple example taken from Liebler (2002, p. 67). An ESI-MS analysis of a peptide revealed two ions, ion<sub>1</sub> with a  $m_{\text{ion}}/z_{\text{ion}} = 784.7$  and ion<sub>2</sub> with a  $m_{\text{ion}}/z_{\text{ion}} = 1567.9$ . How to estimate the molecular mass ( $m$ ) of the peptide?

There are two categories of deconvolution methods, one being probabilistic and the other deterministic. Here we employ only a representative of the deterministic method because it is simpler and in most cases sufficient.

Recall that we have designated  $z_0$  as the charge of the least-charged ion, which is ion<sub>2</sub> in this example (note that, with the same  $m$ , the more charge, the smaller the  $m/z$  ratio). Ion<sub>1</sub> then carries  $(z_0 + 1)$  protons. Each proton adds one atomic mass to the peptide, so the expected  $m/z$  values for ion<sub>1</sub>, and ion<sub>2</sub>, designated as  $mz_1$ ,  $mz_2$ , respectively, can be expressed as:

$$\begin{aligned} mz_1 &= (m + z_0 + 1)/(z_0 + 1) \\ mz_2 &= (m + z_0)/z_0 \end{aligned} \quad (19.1)$$

The least-square estimation of the two parameters (i.e.,  $m$  and  $z_0$ ) is to minimize the sum of squared deviation (SS) between the observed  $mz_i$  values, i.e., 784.7 and 1567.9, and their, respectively, expected  $mz_1$  and  $mz_2$  in Eq. (19.1):

$$\begin{aligned} SS &= (mz_1 - 784.7)^2 + (mz_2 - 1567.9)^2 \\ &= \left(\frac{m+z_0+1}{z_0+1} - 784.7\right)^2 + \left(\frac{m+z_0}{z_0} - 1567.9\right)^2 \end{aligned} \quad (19.2)$$

To minimize SS, we take partial derivative of SS with respect to  $m$  and  $z_0$ , set the two partial derivatives to 0 and solve for  $m$  and  $z_0$ . The two partial derivatives are:

$$D_m = \frac{\partial SS}{\partial m} = \frac{2\left(\frac{m+z_0+1}{z_0+1} - 784.7\right)}{z_0 + 1} + \frac{2\left(\frac{m+z_0}{z_0} - 1567.9\right)}{z_0} \quad (19.3)$$

$$\begin{aligned} D_{z_0} &= \frac{\partial SS}{\partial z_0} = 2\left(\frac{m+z_0+1}{z_0+1} - 784.7\right)\left(\frac{1}{z_0+1} - \frac{m+z_0+1}{(z_0+1)^2}\right) \\ &\quad + 2\left(\frac{m+z_0}{z_0} - 1567.9\right)\left(\frac{1}{z_0} - \frac{m+z_0}{z_0^2}\right) \end{aligned} \quad (19.4)$$

Setting  $D_m$  and  $D_{z_0}$  to 0 and solving the simultaneous equations with the constraint of  $z_0 > 0$  result in  $m = 1567.9032$  and  $z_0 = 1.0006$  (which is taken to

mean 1). This means that ion<sub>1</sub> carries two ( $=z_0 + 1$ ) protons, and ion<sub>2</sub> carries only one proton.

One may ask why we can't just assume  $z_0 = 1$ , so that the  $m_{\text{ion}}/z_{\text{ion}}$  for ion<sub>2</sub> (= 1567.9) can be taken as a direct measure of  $m$ . The reason is that  $z_0$  is often not 1. For long peptides or proteins, the chance of getting singly charged ion in an ESI-MS instrumentation is typically quite small. It might help to introduce a numerical illustration.

Amino acid residues that may carry charges are mainly the three basic amino acids (arginine, lysine, histidine) and two acidic amino acids (glutamic acid and aspartic acid), plus the *N*-terminal amino acid with an amino group and *C*-terminal amino acid with a carboxyl group. In acidic solutions (say pH = 3), the probability of the amino group of the three basic amino acids being protonated is nearly 1 according to the following equation (see Chap. 10 for its derivation):

$$P_{\text{NH}_3^+} = \frac{1}{10^{pH-pK_a} + 1} \quad (19.5)$$

You may substitute pH = 3 and the  $pK_a$  value (12.50, 10.79, and 6.50 for arginine, lysine, and histidine, respectively, Table 17.1 in Chap. 17) into the equation to verify that the proportion is nearly 1. The probability of the amino group in the *N*-terminal amino acid, with its  $pK_a = 8.56$ , being protonated is also nearly 1.

The probability of the two acidic amino acids being protonated can be calculated according to the following equation:

$$P_{\text{RCOO}^-} = \frac{10^{pH-pK_a}}{1 + 10^{pH-pK_a}} \quad (19.6)$$

Now suppose a protein contains 10 lysine residues and no arginine, histidine, aspartic acid, and glutamic acid residues. In a solution with pH = 3, the 10 lysine residues are protonated, so is the amino group of the *N*-terminal amino acid. The *C*-terminal carboxyl has a probability of 0.21595 being protonated (you may verify this by substituting  $pK_a = 3.56$  for the *C*-terminal carboxyl into the equation above). With N copies of such a protein in the solution, there will be only two ions, one with its *C*-terminal carboxyl protonated and the other not, with their, respectively, proportions being 0.21595 and 1–0.21595. This implies that the net charges of the two ions will be 11 and 10, respectively. There will essentially be no ion carrying fewer than 10 charges. In this case  $z_0 = 10$ .

One may argue that our example is too artificial, with a protein carrying no negatively charged residues such as aspartic acid or glutamic acid residues. We will now consider the case when the protein not only contains 10 lysine residues but also 10 aspartic acid residues (but no arginine, histidine, and glutamic acid residues as before). The proportion of aspartic acid residues (whose  $pK_a = 3.91$ ) being protonated is 0.10955 according to Eq. 19.6. The frequency distribution of the proteins with 0, 1, ..., 10 aspartic acid residues protonated is then specified by the binomial

**Table 19.1** Output from ESI-MS with estimated  $z$  in the last column

Ion	$m/z$	Number	$z$
1	687.72	1000	18
2	727.91	4600	17
3	773.39	9000	16
4	824.93	13,400	15
5	883.74	13,400	14
6	951.67	10,000	13
7	1030.88	8000	12
8	1124.46	7500	11
9	1236.91	6500	10
10	1374.16	6000	9
11	1545.66	5600	8
12	1766.29	1000	7

distribution of  $(p + q)^{10}$ , where  $p = 0.10955$  and  $q = 1 - p$ . Thus, given  $N$  copies of such a protein in the solution, the proportion of proteins with 0, 1, ..., 10 aspartic acid residues protonated is 0.31340, 0.38557, 0.21346, 0.07003, 0.01508, 0.00223, 0.00023, 0.00002, 0.00000, 0.00000, and 0.00000, respectively. The proportion is also the proportion of protein ions carrying 10, 9, ..., 0 extra protons if we ignore the *N*-terminal and *C*-terminal amino acids (you may take into consideration the *N*-terminal and *C*-terminal amino acids as an exercise). Obviously, the chance of having a protein ion carrying five extra protons is already very small ( $= 0.00223$ ), and the chance of having an ion carrying fewer than three extra protons is essentially zero. Thus, the chance of having  $z_0 = 1$  is often negligibly small, and we consequently cannot assume  $z_0 = 1$ . In short, it is necessary to estimate both  $z_0$  and  $m$ .

Now that you are convinced that we cannot assume  $z_0 = 1$ , we will introduce a more complicated example involving a long peptide, with the output from ESI-MS shown in Table 19.1. There are 12 types of differentially charged ions with their respective  $m/z$  ratios. The estimated molecular mass of the molecule is about 12,358, and the estimated  $z$  is listed in the last column of Table 19.1. How did we get such estimates?

We first will ignore the column headed by “Number” and use only information in the column headed by “ $m/z$ .” Again recall that  $z_0$  is the number of charges (i.e., extra protons) carried by the least-charged ion, which is ion<sub>12</sub> with its  $m/z = 1766.29$ . Ion<sub>11</sub>, ion<sub>10</sub>, ..., and ion<sub>1</sub> carry  $(z_0 + 1)$ ,  $(z_0 + 2)$ , ... and  $(z_0 + 11)$  protons, respectively. Designate  $m$  as the mass of the peptide. Because the molecular mass of each proton is 1, the actual mass of ion<sub>12</sub>, ion<sub>11</sub>, ..., and ion<sub>1</sub> is  $(m + z_0)$ ,  $(m + z_0 + 1)$ , ...,  $(m + z_0 + 11)$ . Thus defined, the expected  $m/z$  values for ion<sub>1</sub>, ion<sub>2</sub>, ..., ion<sub>12</sub>, designated as  $mz_1$ ,  $mz_2$ , ...,  $mz_{12}$ , respectively, can be expressed as:

$$\begin{aligned} mz_1 &= (m + z_0 + 11)/(z_0 + 11) \\ mz_2 &= (m + z_0 + 10)/(z_0 + 10) \\ &\dots \\ mz_{12} &= (m + z_0 + 0)/(z_0 + 0) \end{aligned} \tag{19.7}$$

The least-square estimation of the two parameters (i.e.,  $m$  and  $z_0$ ) is to minimize the sum of squared deviation between the observed  $m/z$  values (i.e., 687.72, 727.91, etc., in Table 19.1) and the expected  $mz_1$ ,  $mz_2$ , etc., in Eq. (19.7):

$$SS = (mz_1 - 687.72)^2 + (mz_2 - 727.91)^2 + \cdots + (mz_{12} - 1766.29)^2 \quad (19.8)$$

To minimize SS, one naturally would take partial derivatives of SS with respect to  $m$  and  $z_0$  to obtain  $D_m$  and  $D_{z_0}$ , set them to 0, and solve the simultaneous equations to obtain  $m$  and  $z_0$ , just as we have done before with only two ions. However, with 12 ions,  $D_m$  and  $D_{z_0}$  may become too complicated for analytically solutions of  $m$  and  $z_0$  to be obtained. So it is time to learn how to obtain numerical solutions by numerical iteration.

We can substitute different values of  $m$  and  $z_0$  to obtain  $D_m$  and  $D_{z_0}$  values. The  $m$  and  $z_0$  values that make  $D_m$  and  $D_{z_0}$  values closest to 0 are the best estimates. Table 19.2 shows such an iteration process.

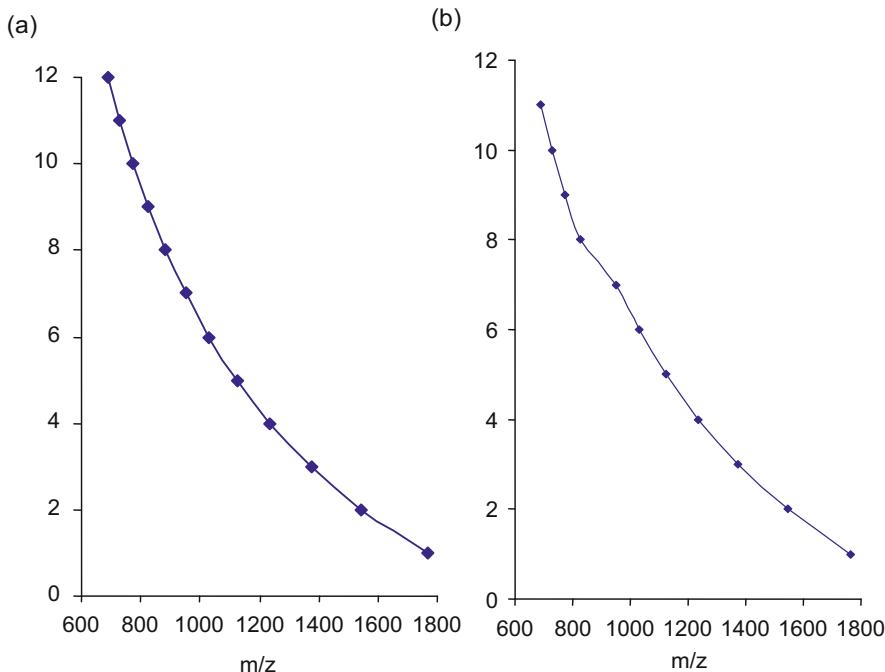
We first tried  $m$  values from 12,350 to 12,370 and  $z_0 = 7$ , and we found the  $m$  value equal to 12,358 to yield the smallest  $D_m$  and  $D_{z_0}$  values (Table 19.2). Then we fix  $m = 12,358$  but vary  $z_0$  values from 6 to 8 and found both  $z_0 = 6$  and  $z_0 = 8$  to be very poor estimates, resulting in very large  $D_m$  and  $D_{z_0}$  values (Table 19.2). Thus, we conclude that our  $z_0 = 7$ , and the  $z$  values in the last column of Table 19.1 are easily obtained because  $z_{12} = z_0$ ,  $z_{11} = z_0 + 1$ ,  $z_{10} = z_0 + 2$ , ...,  $z_1 = z_0 + 11$ . Of course one can continue the iteration process to obtain more accurate estimate of  $m$ .

More advanced algorithms would incorporate the column headed by “Number” in Table 19.1 into a weighted estimation. However, this is often not necessary given the outstanding performance of modern MS.

One may ask what we should do when some of the ions are missing, e.g., ion<sub>6</sub> may not get ionized and consequently will not have its  $m/z$  value reported. One can formulate more advanced probabilistic methods to evaluate the probability of

**Table 19.2** Estimation of  $m$  and  $z_0$  by computer iteration

$m$	$z_0$	$D_m$	$D_{z_0}$
12,350	7	-1.61	2012.45
12,360	7	0.38	-533.41
12,370	7	2.37	-3083.39
12,365	7	1.38	-1807.89
12,361	7	0.58	-788.22
12,359	7	0.18	-278.64
12,358	7	-0.02	-23.91
12,357	7	-0.22	230.78
12,358	6	323.38	-512885.17
12,358	8	-206.63	251562.87



**Fig. 19.1** Missing-ion plot, with all ions present (a) and with one ion missing (b)

missing ions. However, a simpler method is just to plot the observed  $m/z$  ratio versus the series of  $n, n-1, n-2, \dots, 1$  where  $n$  is the number of types of differently charged ions (e.g., 12 in our example in Table 19.1). Such a plot, which I call missing-ion plot, for data in Table 19.1 is shown in Fig. 19.1a.

The curve is smooth when no ion is missing (Fig. 19.1). In contrast, when one ion is missing (e.g., when ion<sub>6</sub> in Table 19.1 with  $m/z = 951.67$  is missing), the curve is no longer smooth (Fig. 19.1b). The curve would become even more twisted if two consecutive ions are missing (by two consecutive ions I mean two ions carrying  $i$  and  $i + 1$  charges, respectively). Modern MS data are so accurate that a missing ion can generally be identified by such a plot.

How many positively charged ions we should expect to have, given a peptide “DAFLGSFLYEYSR”? The last R (arginine residue) and the *N*-terminal amino group can both be protonated. So we should have only two different positively charged ions, one with  $z = 1$  and the other with  $z = 2$ . This is in fact the peptide that provides us with the first example in this section where we have ion<sub>1</sub> with a  $m_{\text{ion}}/z_{\text{ion}} = 784.7$  and ion<sub>2</sub> with a  $m_{\text{ion}}/z_{\text{ion}} = 1567.9$ .

## 4 Peptide Mass Fingerprinting

Peptide mass fingerprinting (PMF) is for protein identification. For example, one may perform 2D-SDS-PAGE of liver proteins between a normal person and a liver cancer patient. Comparing the two gels, one may find a dot that is different between the two. One naturally wishes to know what protein the dot represents. Establishing a link from a protein dot on the gel to a protein-coding gene on the genome is where PMF shines. Below I detail the four essential steps in PMF.

### 4.1 Peptide Digestion

The first step in PMF is to cut out the protein dot on the gel and digest it into peptides by using one of proteases (Table 19.3). For example, trypsin cuts after Arg and Lys residues when the residue is not immediately followed by a Pro (Table 19.3). The purpose of including amino acid frequencies in Table 19.3 is to correct a

**Table 19.3** Amino acid frequencies from 34,179 annotated human CDSs from GenBank and protease cleavage site

AA	Percent	Trypsin	Chymotrypsin	Asp N	Glu C	Lys C
Ala	7.21					
Arg	5.96	X\Pro				
Asn	3.48					
Asp	4.65			X		
Cys	2.29					
Gln	4.75					
Glu	7.02				X\Pro	
Gly	6.84					
His	2.60					
Ile	4.19					
Leu	9.77					
Lys	5.62	X\Pro				X\Pro
Met	2.10					
Phe	3.51		X\Pro			
Pro	6.65					
Ser	8.34					
Thr	5.33					
Trp	1.25		X\Pro			
Tyr	2.54		X\Pro			
Val	5.89					
Sum	100	11.58	7.30	4.65	7.02	5.62

X – cut after the specific amino acid; \Pro – cleavage inhibited if the cleavage site is followed by proline

misconception. Some authors (e.g., Liebler et al. 2002, p. 52) have remarked that chymotrypsin may cleave too frequently because it cleaves at three amino acid residues (Phe, Trp, and Try) to yield too many peptides that are too small to be informative in MS analysis. However, for human proteins, trypsin is expected to cut much more frequently than chymotrypsin because Arg and Lys account for a total of 11.58% of all amino acid residues, whereas Phe, Trp, and Tyr jointly account for only 7.30% of all amino acid residues (Table 19.3). This implies that trypsin will cleave the proteins into much smaller peptides than chymotrypsin.

Trypsin is widely used in protein digestion in MS analysis. However, it is not suitable for all proteins. For example, the human *DEXI* gene codes for 95 amino acids which include only one Arg and no Lys residue. The protein consequently cannot produce suitable peptides for MS analysis with trypsin digestion. However, 13 of its residues are Phe, Trp, and Tyr, and it can consequently be cut by chymotrypsin into peptides suitable for MS analysis. It also contains nine Glu residues and can be cut by Glu C into peptides suitable for MS analysis. On the other hand, the human *PRB3* gene codes 351 amino acid residues but contains only one Glu residue and no Phe, Trp, or Tyr residue, i.e., Glu C will cut it only once and chymotrypsin will not cut it at all. However, it contains 17 Arg and 17 Lys residues and can be cut by trypsin into peptides with lengths well suited for MS analysis. A large-scale peptide mass fingerprinting will almost always involve digestion with more than one protease.

It is almost always a good practice to have a rough estimate of the average peptide length as well as the distribution of the peptide lengths after digestion with a certain protease. For example, suppose we digest a human protein with trypsin which cleaves the protein after the Lys and Arg residues when they are not followed by a Pro. From Table 19.3, we know that Lys and Arg residues (hereafter referred to as KR residues) jointly account for 11.58% of the amino acid residues of human proteins and Pro accounts for 6.65%, which is also the probability that a KR residue is followed by a Pro. Thus, the probability that a KR residue that is not followed by a Pro (i.e., the probability of cleavage by trypsin) is:

$$p = 0.1158(1 - 0.0665) = 0.1081 \quad (19.9)$$

The distribution of the peptide length ( $l$ ) follows the geometric distribution:

$$P(L = l|p) = (1 - p)^{l-1}p, \quad (19.10)$$

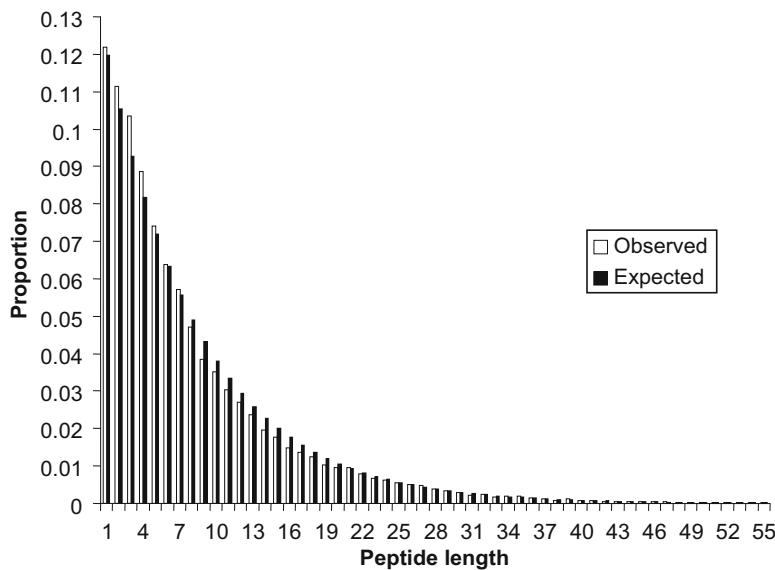
The expected mean and variance of the peptide length are then, respectively:

$$\begin{aligned} E(L) &= \frac{1}{p} = 9.25 \\ \text{Var}(L) &= \frac{(1-p)}{p^2} = 76.32 \end{aligned} \quad (19.11)$$

In contrast, digesting the same protein with chymotrypsin would generate peptides with an expected mean length of 14.67 and an expected variance of the peptide lengths equal to 200.67.

In the genome of the mammalian gastric pathogen, *Helicobacter pylori* (strain 26,695), the proportions of Arg, Lys, and Pro are 0.0345, 0.0894, and 0.0328, respectively, based on its 1576 annotated proteins. If we have a representative sample of *H. pylori* proteins and use trypsin to completely digest all these proteins, what is the expected mean length of the resulting peptides?

From the three proportions for Arg, Lys, and Pro, we can estimate  $p = (0.0345 + 0.0894)*(1 - 0.0328) = 0.119836$ . Thus, the expected mean and variance of the peptide lengths, after complete trypsin digestion, are 8.3447 and 61.2898, respectively. The observed distribution, based on an *in silico* trypsin digestion of a random sample of *H. pylori*, is close to the expected distribution (Fig. 19.2). Generally peptides of length between 5 and 21 residues can be measured accurately by MALDI-TOF MS. The distribution in Fig. 19.2 has a proportion of 0.504300196 of the peptides with lengths between 5 and 21. This means that about half of the digested peptides from *H. pylori* proteins can be measured by MALDI-TOF MS with high accuracy.



**Fig. 19.2** Observed and expected distribution of peptide length, from trypsin digestion of a random sample of proteins in *H. pylori* Str 26695

## 4.2 MS Determination of Peptide Mass

Now that the protein dot has been reduced to a number of peptides (referred to hereafter as query peptides), we proceed to the second step in PMF by subjecting these query peptides to MS to obtain peptide masses. Suppose we now have determined the peptide mass of  $n$  peptides resulting from digesting the protein dot from the 2-D gel. We typically will create a file, say ProteinDot1.txt, to store these  $n$  peptide masses,  $m_1, m_2, \dots, m_n$ . For example, one protein dot from a sample of *H. pylori* proteins could have eight peptides with their molecular masses determined by MS:

```
832.94
974.05
1105.16
1526.68
1537.92
1653.86
1680.87
2231.42
```

The list of eight peptide masses, hereafter referred to as query peptide masses, will be matched against the peptide mass of all possible peptides from an in silico digestion of all *H. pylori* proteins to identify the protein. This brings us to the third step of creating a database of peptide masses.

## 4.3 Protein Database and In Silico Digestion

The third step of PMF is to obtain a relevant database of proteins and perform an in silico digestion, using the same protease that has been used to digest the protein dot. If one is working on human, then the relevant database of proteins would obviously be human proteins. For example, one may retrieve all 34,169 annotated human CDSs, translate them into proteins, and perform an in silico digestion to obtain all possible peptides with their respective masses. We have already learned how to compute the molecular mass of a peptide in Chap. 11.

If one is working on *H. pylori*, then one may retrieve the annotated protein-coding genes in sequenced *H. pylori* genomes and perform in silico digestion to obtain all possible peptides with their, respectively, masses. The in silico trypsin digestion of all 1576 annotated proteins in *H. pylori* strain 26,695 generates 61,160 peptides, which are partially shown in Table 19.4.

The list of peptide masses from the database will be referred to as database peptide masses (designated by  $M_j$ ) to distinguish them from the query peptide masses (designated by  $m_i$ ) which pertain to peptides from an isolated protein (e.g., a protein dot in a 2D-SDS-PAGE).

**Table 19.4** Partial list of peptide masses from trypsin digestion of the 1576 annotated proteins in *H. pylori* strain 26,695, ascending sorted by mass. GeneInd specifies the gene from which the peptide is from. Note that the peptide at the C-terminal of the protein may not end with K (Lys) or R (Arg)

Mass	GeneInd	Peptide	Mass	GeneInd	Peptide
832.934	38	QFTYFK	1526.688	1386	SLGNNLYYNTYVR
832.941	48	NNFPTLK	...		
832.941	319	FPTINNK	1537.808	71	GVAFSLLSFLLEGGLK
832.941	476	GFNAPSLK	1537.919	319	MLVGASLLTHALJAK
832.962	106	LGEAMANK	1538.613	747	FFDLGEYFEEDK
...		...			
974.023	853	GVEAEVQDK	1653.84	852	IHFQAQNQQLFSSAK
974.052	319	YYTIDALK	1653.856	319	YFAFLDWQGYGMR
974.052	507	VYDLSYYK	1653.861	1379	MIGGSENIESAISFAK
974.066	57	ITNEQIEK	1653.87	510	EIVAYLDEYIIGQK
974.066	171	TALVENEAK	...		
974.066	1235	YSIDLALHR	1680.817	517	VPNQATFYDDLQAAK
...		...	1680.868	242	IGLNQNQEIDAQNPK
1105.114	5	DDDNLALSSR	1680.868	319	ALFVDEHEFEIGFK
1105.155	319	EESAAPSWK	1680.883	1360	MDEVDLIFEEEAIK
1105.184	493	VDYNNYYLR	...		
...		...	2230.723	1277	FYATLALSCVFLTTTNILVK
1526.644	87	GLDEAIEFLEYYV	2231.42	319	EVTNYQTGYTNHTSVNSVK
1526.684	319	VFAFYVGNYHF	2231.42	1503	DFYEEELLYLGLLEEQNDK
			2231.499	208	EVDVLGGMGTTDHSGLQYR

#### 4.4 Protein Identification

The fourth and final step in PMF is to search each of the query peptide masses (e.g., the eight query peptide masses from a *H. pylori* protein dot in Sect. 4.2) against *H. pylori* database peptide masses (Table 19.4). For example, the first query peptide mass, 832.94, matches five database peptide masses (Table 19.4); the second query peptide mass, 974.05, matches six database peptide masses; and so on. The number of matches depends on the accuracy of MS. If peptide mass  $m_i$  is accurate to 0.5 Dalton, then any database peptide within the peptide mass range of  $(m_i - 0.5, m_i + 0.5)$  would be considered as a match.

We note that all eight query peptide masses match peptides from the protein with gene index (GeneInd) of 319 (Table 19.4). We can therefore conclude that the query peptides come from the gene with GeneInd = 319, which has a locus\_tag of HP0324 and is described as “outer membrane protein (omp10)” in the GenBank file (NC\_000915). There is no other gene that comes close to this gene in term of the number of matches.

The intuitive argument above linking a protein dot and a protein-coding gene on the genome has problems, especially in eukaryotes where proteins differ dramatically in lengths. For example, the mammalian dystrophin protein coded by the *dmd* gene contains about 4000 amino acid residues coded by about 75 exons. It consequently would have a much large probability (100 times to be exact) of getting a random match than a small protein of 40 amino acid residues. We therefore need to develop a statistical framework to help us decide whether our identification is correct or not.

We have two hypotheses, i.e., the protein dot is HP0324 (designated as  $\theta_{\text{Yes}}$ ) and it is not (designated as  $\theta_{\text{No}}$ ). Designate  $Y_{ij}$  as the event of query peptide mass  $m_i$  matching database peptide mass  $M_j$  of HP0324 (19 peptides are generated by *in silico* digestion of the annotated HP0324 protein, so  $j = 1, 2, \dots, 19$ ). The likelihood of  $Y_{ij}$ , i.e., the probability of  $Y_{ij}$  happening given  $\theta_{\text{Yes}}$  is true, is generally close to 1 (although molecular mechanisms such as posttranslational modification may reduce this value slightly), i.e.:

$$p(Y_{ij}|\theta_{\text{Yes}}) \approx 1 \quad (19.12)$$

If the protein dot is not HP0324, then what is the probability of event  $Y_{ij}$  happening? This can be estimated empirically by searching the 17  $M_j$  values from HP0324 against the rest of the  $M$  values from other *H. pylori* proteins. Designate the number of matches for  $M_1, M_2, \dots, M_{17}$  as  $N_1, N_2, \dots, N_{17}$ , respectively, we can estimate:

$$p(Y_{ij}|\theta_{\text{No}}) = \frac{\sum_{j=1}^{17} N_j}{17(N_T - 17)} \quad (19.13)$$

where  $N_T$  is the total number of database peptides results from in silico digestion. For simplicity, let's assume that  $p(Y_{ij}|\theta_{\text{No}}) = 0.0003$ .

Now designating the protein length of HP0324 as  $L_{\text{HP0324}}$ , and the total length of all 1576 *H. pylori* proteins as  $L_T$ , we have the prior probabilities for  $\theta_{\text{Yes}}$  and  $\theta_{\text{No}}$  as:

$$\begin{aligned} p(\theta_{\text{Yes}}) &= \frac{L_{\text{HP0324}}}{L_{\text{Total}}} = \frac{255}{503015} = 0.0005 \\ p(\theta_{\text{No}}) &= 1 - p(\theta_{\text{Yes}}) = 0.9995 \end{aligned} \quad (19.14)$$

According to Bayes' theorem, the probability that  $\theta_{\text{No}}$  is true is:

$$\begin{aligned} P(\theta_{\text{No}}|Y_{ij}) &= \frac{P(Y_{ij}|\theta_{\text{No}})P(\theta_{\text{No}})}{P(Y_{ij}|\theta_{\text{Yes}})P(\theta_{\text{Yes}}) + P(Y_{ij}|\theta_{\text{No}})P(\theta_{\text{No}})} \\ &= \frac{0.0003 \times 0.9995}{1 \times 0.0005 + 0.0003 \times 0.9995} \approx 0.37488 \end{aligned} \quad (19.15)$$

Thus, with only one  $Y_{ij}$  event, we cannot reject  $\theta_{\text{No}}$ . However, we have eight  $m_i$  values that all match  $M_j$  values from HP0324. So the final probability that  $\theta_{\text{No}}$  is true is  $0.37488^8 = 0.0004$ . So  $\theta_{\text{No}}$  can be conclusively rejected, and we conclude that the protein dot is indeed HP0324. The formulation above can be further refined by taking into consideration the fact that peptides of different lengths have different matching probabilities against database peptide mass.

Peptide mass fingerprinting, together with quantification of protein abundance, ultimately leads to two types of data for further bioinformatic analysis. The first type is between the control and the experiment and the second type is what is known as time-course data, obtained by sampling the proteome of a cell lineage over different time points. For example, one may synchronize the development cycle of yeast cells and sampling the proteome at regular time intervals during the yeast cell cycle, or trigger the developmental cascade of a stem cell lineage and sample the proteome at regular time intervals. These two types of data parallel those from transcriptomic experiments and can be analyzed similarly to identify genes that are upregulated or downregulated at the protein level.

## Postscript

The thought of a reader reaching the end of a book always sends a thrilling feeling to the author. I have heard that many Chinese children today no longer dream about becoming authors of science. Instead, they want to become American president, and

their parents would typically beam with pride when the youngsters expressed such ambitions. This has been terrifying to me. The world cannot afford to have many American presidents – just one seems to be damaging enough. It would seem more desirable for our younger generations to have modest dreams of writing books that are somewhat readable from the beginning to the end.

Dreaming of writing a book is nice and noble. Dreaming of becoming American president could be brutal and bloody. May the young minds not be corrupted by the evil of power!

Extreme power disrupts harmony, and harmony is the essence of life. We biologists know only too well that harmony manifests at all levels of organization of living beings, and disrupting harmony destroys not only the beauty of life but also life itself. Yet harmony among different cellular components interacting within a cell and harmony among different cells interacting within an organism can only be achieved by these cells and cellular components following certain rules. If some cells break the rule, if they get out of the checking system, disasters such as cancer emerge, often with the consequence that either the misbehaving cell has to die or the organism has to die. Overtime multicellular organisms have evolved very complicated checking systems to safeguard against misbehaving cells. Life is charming with these checking systems but turns ugly when such checking systems are broken.

Human communities and societies are not much different. People, including national leaders such as presidents and prime ministers, will want to gather more power to break the checking system. According to major religions, every individual has a sinful nature and has to be checked. Perhaps no one had better understanding of the sinful nature of human beings than the founding fathers of the United States of America who, based on this fundamental understanding, established a great political checking system that has dramatically alleviated the detrimental effect of misbehaving leaders. They recognized every human to have a good half and an evil half. The good half can be employed to accomplish public services, but the evil half deserves constant monitoring and checking.

How far has the present American administration deviated from the path chosen by the founding fathers! Instead of drawing a line within ourselves to recognize the good half and the evil half, they draw a line among nations to recognize good nations and rogue nations. Everything good is “American” and everything else is “un-American.” How similar this is to former communist regimes where everything good was communist and everything bad was anti-communist!

Nature has created us, probably equal, but not perfect. We commit errors, seek forgiveness, and try to improve ourselves. The books we write are not perfect either. We solicit criticisms from our colleagues and seek forgiveness for serious errors and egregious omissions. Just as we revise books to improve them, we cleanse our souls to become better citizens. With the recognition of imperfection in us, we behave better, our books read better, and the world moves closer to perfection and harmony.

# Appendix

## The Delta Method for Deriving Parameter Variance

Delta method has been used extensively in deriving variances of parameters (Kimura and Ohta 1972; Waddell and Steel 1997a; Xia 2007b, pp. 256–262). When a variable  $Y$  is a function of a variable  $X$ , i.e.,  $Y = F(X)$ , the delta method allows us to obtain approximate formulation of the variance of  $Y$  if (1)  $Y$  is differentiable with respect to  $X$  and (2) the variance of  $X$  is known. The same can be extended to more variables. Take, for example, the simplest case of  $Y = F(X)$ . Regardless of the functional relationship between  $Y$  and  $X$ , we always have

$$\Delta Y \approx \left( \frac{dY}{dX} \right) \Delta X \quad (\text{A.1})$$

$$(\Delta Y)^2 \approx \left( \frac{dY}{dX} \right)^2 (\Delta X)^2. \quad (\text{A.2})$$

where  $\Delta Y$  and  $\Delta X$  are small changes in  $Y$  and  $X$ , respectively.

Note that the variance of  $Y$  is the expectation of the squared deviations of  $Y$ , i.e.,

$$\begin{aligned} V(Y) &= E(\Delta Y)^2 \\ V(X) &= E(\Delta X)^2. \end{aligned} \quad (\text{A.3})$$

Replacing  $(\Delta Y)^2$  and  $(\Delta X)^2$  in Eq. (A.2) with  $V(Y)$  and  $V(X)$ , we have

$$V(Y) \approx \left( \frac{dY}{dX} \right)^2 V(X). \quad (\text{A.4})$$

This relationship allows us to obtain an approximate formulation of the variance of either  $Y$  or  $X$  if we know either  $V(X)$  or  $V(Y)$ .

## Variance of Allele Frequency for a Recessive Allele

As students are always eager to have more illustrative examples, and because I myself belong to the lesser folks who cannot see the beauty of equations without rendering them to numbers, I will present another example, taken from a book on population genetics (Li 1976), in which we know the variance of  $Y$  and want to estimate the variance of  $X$ .

Given a locus with one dominant allele (A) and one recessive allele (a), we have only two distinguishable phenotypes, dominants (AA, Aa) and recessives (aa). You might be interested to know that the severe human disease cystic fibrosis is determined by one locus with a dominant allele and a recessive allele. The disease is caused by homozygosity for the recessive allele. How to estimate the frequency of the recessive allele and its variance?

Let  $D$  and  $R$  be the observed numbers of dominants and recessives in a sample of  $N$  random individuals ( $N = D + R$ ). Our estimate of the frequency of allele a, designated  $q$ , is

$$\begin{aligned} q^2 &= R/N \\ q &= \sqrt{R/N} \end{aligned} \tag{A.5}$$

In the case of cystic fibrosis, suppose we found 4 patients out of 10,000 surveyed, so  $q = 1/50$ . Now we proceed to find the variance of  $q$ . From the binomial distribution, we know the variance of  $q^2$  to be

$$V(q^2) = \frac{q^2(1-q^2)}{N} \tag{A.6}$$

In the framework of the delta method with  $Y = F(X)$ , we have  $Y = q^2$ ,  $X = q$ , and  $dY/dX = 2q$ . We already know the variance of  $Y$  (i.e.,  $q^2$ ) in Eq. (A.6), and the variance of  $X$  can be obtained as follows:

$$\begin{aligned} V(Y) &= \frac{q^2(1-q^2)}{N} = \left(\frac{dY}{dq}\right)^2 V(q) = (2q)^2 V(q) \\ V(q) &= \frac{V(Y)}{(2q)^2} = \frac{\frac{q^2(1-q^2)}{N}}{4q^2} = \frac{1-q^2}{4N} \end{aligned} \tag{A.7}$$

You might have noticed that  $q^2 = R/N$  and  $(1-q^2) = D/N$ . So we have

$$V(q) = \frac{1-q^2}{4N} = \frac{D/N}{4N} = \frac{D}{4N^2} \tag{A.8}$$

In the case of cystic fibrosis with  $N = 10,000$  and  $R = 4$ ,  $q = 1/50$ ,  $V(q) = 0.000002499$ .

### **Variance of JC69 Distance**

The delta method is not needed for deriving the variance of JC69 distance ( $D_{JC69}$ ) because  $D_{JC69}$  does have a likelihood estimator of its variance, but we will apply the delta method just for illustrating the method. We note that  $D_{JC69}$  is a function of  $P_{\text{diff}}$  (the proportion of sites differing between two aligned sequences), and the variance of  $P_{\text{diff}}$  is known from the binomial distribution:

$$D_{JC69} = -\frac{3}{4} \ln \left( 1 - \frac{4P_{\text{diff}}}{3} \right) \quad (\text{A.9})$$

$$V(P_{\text{diff}}) = \frac{P_{\text{diff}}(1 - P_{\text{diff}})}{L} \quad (\text{A.10})$$

where  $L$  is the length of the two aligned sequences. From the expression of  $D_{JC69}$  in Eq. (A.9), we have

$$\begin{aligned} \frac{\partial D_{JC69}}{\partial P_{\text{diff}}} &= \frac{1}{1 - \frac{4P_{\text{diff}}}{3}} \\ V(D_{JC69}) &= \left( \frac{\partial D_{JC69}}{\partial P_{\text{diff}}} \right)^2 V(P_{\text{diff}}) = \frac{P_{\text{diff}}(1 - P_{\text{diff}})}{L \left( 1 - \frac{4P_{\text{diff}}}{3} \right)^2}. \end{aligned} \quad (\text{A.11})$$

which is the same as the likelihood estimator.

### **The Variance of K80 Distance**

The distance for the K80 model is

$$\begin{aligned} D_{K80} &= 2\alpha t + 4\beta t = \frac{1}{2} \ln(a) + \frac{1}{4} \ln(b), \quad \text{where} \\ a &= \frac{1}{1 - 2P - Q} \quad \text{and} \quad b = \frac{1}{1 - 2Q}. \end{aligned} \quad (\text{A.12})$$

The variance of  $D_{K80}$  can be derived by the delta method as before:

$$dD_{K80} = \left( \frac{\partial D_{K80}}{\partial P} \right) dP + \left( \frac{\partial D_{K80}}{\partial Q} \right) dQ = d_1 \bullet dP + d_2 \bullet dQ \quad (\text{A.13})$$

$$\begin{aligned}
V(D_{K80}) &= (dD_{K80})^2 = [d_1 \bullet dP + d_2 \bullet dQ]^2 \\
&= d_1^2 dP^2 + 2d_1 d_2 dPdQ + d_2^2 dQ^2 \\
&= d_1^2 V(P) + 2d_1 d_2 \text{Cov}(P, Q) + d_2^2 V(Q) \\
&= [d_1 \quad d_2] \begin{bmatrix} V(P) & \text{Cov}(P, Q) \\ \text{Cov}(P, Q) & V(Q) \end{bmatrix} \begin{bmatrix} d_1 \\ d_2 \end{bmatrix}
\end{aligned} \tag{A.14}$$

Recall that  $P$  stands for the proportion of sites that differ by a transitional change and  $Q$  stands for the proportion of sites that differ by a transversional change. Designate  $R$  as the proportion of identical sites ( $R = 1 - P - Q$ ). From the trinomial distribution of  $(R + P + Q)^L$ , we have

$$\begin{aligned}
V(P) &= \frac{P(1-P)}{L} \\
V(Q) &= \frac{Q(1-Q)}{L} \\
\text{Cov}(P, Q) &= -\frac{PQ}{L}.
\end{aligned} \tag{A.15}$$

Substituting these into Eq. (A.14), we have the variance of  $D_{K80}$ :

$$V(D_{K80}) = (dD_{K80})^2 = \frac{a^2 P + c^2 Q - (aP + cQ)^2}{L} \tag{A.16}$$

where  $c = (a + b)/2$ , with  $a$  and  $b$  defined in Eq. (A.12).

Note that Eq. (A.14) is a general equation for computing the variance by the delta method. For any function  $Y = F(X_1, X_2, \dots, X_n)$ , the variance of  $Y$  is obtained by the variance-covariance matrix of  $X_i$  multiplied left and right by the vector of partial derivatives of  $Y$  with respect to  $X_i$ .

## Illustration of Expectation-Maximization (EM) Algorithm

EM algorithm is used often in bioinformatics. Here is a numerical illustration (the simplest possible, I believe) of the EM algorithm to estimate  $q$  (the frequency of the recessive allele  $a$ ). Note that in this particular case we do not need to use the EM algorithm to estimate  $q$  because  $q$  is already given in Eq. (A.5). However, gaining some familiarity with the EM algorithm may be useful in other situations.

There are three genotypes involving the cystic fibrosis locus, AA, Aa, and aa, with AA and Aa indistinguishable phenotypically. Designate  $p = 1 - q$  and let  $N_{AA}$ ,  $N_{Aa}$ , and  $N_{aa}$  be the number of AA, Aa, and aa genotypes, respectively. Using the notations above, we have  $D = (N_{AA} + N_{Aa})$  and  $R = N_{aa}$ . To facilitate exposition, let's suppose that  $D = 9996$  and  $R = 4$ .

Since we cannot observe  $N_{AA}$  and  $N_{Aa}$  directly (they are indistinguishable phenotypically), the two numbers represent incomplete data from a three-category trinomial distribution. The complete data specification is as follows:

$$f(N_{AA}, N_{Aa}, N_{aa}|q) = \frac{N!}{N_{AA}!N_{Aa}!N_{aa}!} [p^2]^{N_{AA}} [2pq]^{N_{Aa}} [q^2]^{N_{aa}} \quad (\text{A.17})$$

The EM algorithm consists of two steps, the estimation step (or E-step) and the maximization step (or M-step). Let us start by setting  $q = 0.1$ . For the E-step, we estimate  $N_{AA}$  and  $N_{Aa}$  as follows (with the subscript in  $N_{AA.1}$  and  $N_{Aa.1}$  indicating the first E-step):

$$\begin{aligned} N_{AA.1} &= D \frac{p^2}{p^2 + 2pq} = 8178.545455 \\ N_{Aa.1} &= D \frac{2pq}{p^2 + 2pq} = 1817.454545 \end{aligned} \quad (\text{A.18})$$

Substituting these into Eq. (A.17), we can obtain  $q$  by the maximum likelihood method, i.e., taking the derivative of  $f(N_{AA}, N_{Aa}, N_{aa}|q)$  with respect to  $q$ , setting the derivative to 0, and solving the resulting equation for  $q$ . This gives

$$q_1 = \frac{N_{Aa} + 2N_{aa}}{2N} = 0.091272727 \quad (\text{A.19})$$

where the subscript 1 in  $q_1$  indicates the estimated  $q$  in the first M-step. We now repeat the E-step according to the following equations equivalent to Eq. (A.18):

$$\begin{aligned} N_{AA.i} &= D \frac{p^2}{p^2 + 2pq_{i-1}} \\ N_{Aa.i} &= D \frac{2pq_{i-1}}{p^2 + 2pq_{i-1}} \\ p &= 1 - q_{i-1} \end{aligned} \quad (\text{A.20})$$

and the M-step with  $q_i$  obtained from the following equation equivalent to Eq. (A.19):

$$q_i = \frac{N_{Aa.i} + 2N_{aa}}{2N} \quad (\text{A.21})$$

Repeating the E-step and M-step will result in  $q_i$  asymptotically approaching 0.02 and  $N_{Aa}$  and  $N_{AA}$  approaching 392 and 9604, respectively.

# References

- Abdel-Hameed EA, Ji H, Shata MT (2016) HIV-induced epigenetic alterations in host cells. *Adv Exp Med Biol* 879:27–38
- Abolbaghaei A, Silke JR, Xia X (2017) How changes in anti-SD sequences would affect SD sequences in *Escherichia coli* and *Bacillus subtilis*. *G3 (Bethesda, Md)* 7(5):1607–1615
- Abraham EP, Chain E (1940) An enzyme from bacteria able to destroy penicillin. *Rev Infect Dis* 10 (4):677–678
- Abraham EP, Chain E, Fletcher CM, Florey HW, Gardner AD, Heatley NG, Jennings MA (1941) Further observations on penicillin. *Lancet* 238(6155):177–189
- Abraham JM, Feagin JE, Stuart K (1988) Characterization of cytochrome c oxidase III transcripts that are edited only in the 3' region. *Cell* 55(2):267–272
- Adamski FM, McCaughey KK, Jorgensen F, Kurland CG, Tate WP (1994) The concentration of polypeptide chain release factors 1 and 2 at different growth rates of *Escherichia coli*. *J Mol Biol* 238(3):302–308
- Aerts S, Van Loo P, Thijs G, Mayer H, de Martin R, Moreau Y, De Moor B (2005) TOUCAN 2: the all-inclusive open source workbench for regulatory sequence analysis. *Nucleic Acids Res* 33 (Web Server):W393–W396
- Aerts S, van Helden J, Sand O, Hassan BA (2007) Fine-tuning enhancer models to predict transcriptional targets across multiple genomes. *PLoS One* 2(11):e1115
- Ahn BY, Jones EV, Moss B (1990) Identification of the vaccinia virus gene encoding an 18-kilodalton subunit of RNA polymerase and demonstration of a 5' poly(A) leader on its early transcript. *J Virol* 64(6):3019–3024
- Aird WC, Parvin JD, Sharp PA, Rosenberg RD (1994) The interaction of GATA-binding proteins and basal transcription factors with GATA box-containing core promoters. A model of tissue-specific gene expression. *J Biol Chem* 269(2):883–889
- Akaike H (1973) Information theory and an extension of maximum likelihood principle. In: Petrov BN, Csaki F (eds) Second international symposium on information theory. Akademiai Kiado, Budapest, pp 267–281
- Akaike H (1974) A new look at the statistical model identification. *IEEE Trans Autom Control* 19:716–723
- Akashi H (1994) Synonymous codon usage in *Drosophila melanogaster*: natural selection and translational accuracy. *Genetics* 136(3):927–935
- Akashi H, Gojobori T (2002) Metabolic efficiency and amino acid composition in the proteomes of *Escherichia coli* and *Bacillus subtilis*. *Proc Natl Acad Sci USA* 99(6):3695–3700

- Alatortsev VS, Cruz-Reyes J, Zhelonkina AG, Sollner-Webb B (2008) Trypanosoma brucei RNA editing: coupled cycles of U deletion reveal processive activity of the editing complex. *Mol Cell Biol* 28(7):2437–2445
- Alderwick LJ, Seidel M, Sahm H, Besra GS, Eggeling L (2006) Identification of a novel arabinofuranosyltransferase (AftA) involved in cell wall arabinan biosynthesis in *Mycobacterium tuberculosis*. *J Biol Chem* 281(23):15653–15661
- Allen A, Flemstrom G, Garner A, Kivilakso E (1993) Gastroduodenal mucosal protection. *Physiol Rev* 73(4):823–857
- Alm RA, Trust TJ (1999) Analysis of the genetic diversity of *Helicobacter pylori*: the tale of two genomes. *J Mol Med* 77(12):834–846
- Alm RA, Ling LS, Moir DT, King BL, Brown ED, Doig PC, Smith DR, Noonan B, Guild BC, deJonge BL et al (1999) Genomic-sequence comparison of two unrelated isolates of the human gastric pathogen *Helicobacter pylori*. *Nature* 397(6715):176–180
- Alm RA, Bina J, Andrews BM, Doig P, Hancock RE, Trust TJ (2000) Comparative genomics of *Helicobacter pylori*: analysis of the outer membrane protein families. *Infect Immun* 68(7):4155–4168
- Althaus E, Caprara A, Lenhof HP, Reinert K (2002) Multiple sequence alignment with arbitrary gap costs: computing an optimal solution using polyhedral combinatorics. *Bioinformatics* 18(Suppl 2):S4–S16
- Altschul SF (1996) Local alignment statistics. *Meth Enzymol* 274:460–480
- Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ (1990) Basic local alignment search tool. *J Mol Biol* 215(3):403–410
- Altschul SF, Madden TL, Schaffer AA, Zhang J, Zhang Z, Miller W, Lipman DJ (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res* 25:3389–3402
- Anderson KP, Crable SC, Lingrel JB (1998) Multiple proteins binding to a GATA-E box-GATA motif regulate the erythroid Kruppel-like factor (EKLF) gene. *J Biol Chem* 273(23):14347–14354
- Andersson DI, Kurland CG (1983) Ram ribosomes are defective proofreaders. *Mol Gen Genet* 191 (3):378–381
- Arava Y, Wang Y, Storey JD, Liu CL, Brown PO, Herschlag D (2003) Genome-wide analysis of mRNA translation profiles in *Saccharomyces cerevisiae*. *Proc Natl Acad Sci USA* 100(7):3889–3894
- Arbibe L, Sansonetti PJ (2007) Epigenetic regulation of host response to LPS: causing tolerance while avoiding toll errancy. *Cell Host Microbe* 1(4):244–246
- Arnqvist G (2006) Sensory exploitation and sexual conflict. *Philos Trans R Soc Lond Ser B Biol Sci* 361(1466):375–386
- Arvaniti E, Moulou P, Vakrakou A, Chatziantoniou C, Chadjichristos C, Kavvadas P, Charonis A, Politis PK (2016) Whole-transcriptome analysis of UUO mouse model of renal fibrosis reveals new molecular players in kidney diseases. *Sci Rep* 6:26235
- Ast G (2004) How did alternative splicing evolve? *Nat Rev Genet* 5(10):773–782
- Auch AF, Henz SR, Holland BR, Goker M (2006) Genome BLAST distance phylogenies inferred from whole plastid and whole mitochondrion genome sequences. *BMC Bioinform* 7:350
- Awan AR, Manfredo A, Pleiss JA (2013) Lariat sequencing in a unicellular yeast identifies regulated alternative splicing of exons that are evolutionarily conserved with humans. *Proc Natl Acad Sci USA* 110(31):12762–12767
- Axon AT (1999) Are all helicobacters equal? Mechanisms of gastroduodenal pathology and their clinical implications. *Gut* 45(Suppl 1):I1–I4
- Bablanian R, Banerjee AK (1986) Poly(riboadenyllic acid) preferentially inhibits in vitro translation of cellular mRNAs compared with vaccinia virus mRNAs: possible role in vaccinia virus cytopathology. *Proc Natl Acad Sci USA* 83(5):1290–1294
- Bablanian R, Coppola G, Masters PS, Banerjee AK (1986) Characterization of vaccinia virus transcripts involved in selective inhibition of host protein synthesis. *Virology* 148(2):375–380

- Bablanian R, Goswami SK, Esteban M, Banerjee AK (1987) Selective inhibition of protein synthesis by synthetic and vaccinia virus-core synthesized poly(riboadenylic acids). *Virology* 161(2):366–373
- Bablanian R, Scribani S, Esteban M (1993) Amplification of polyadenylated nontranslated small RNA sequences (POLADS) during superinfection correlates with the inhibition of viral and cellular protein synthesis. *Cell Mol Biol Res* 39(3):243–255
- Bag J (2001) Feedback inhibition of poly(A)-binding protein mRNA translation. A possible mechanism of translation arrest by stalled 40 S ribosomal subunits. *J Biol Chem* 276 (50):47352–47360
- Bag J, Bhattacharjee RB (2010) Multiple levels of post-transcriptional control of expression of the poly (A)-binding protein. *RNA Biol* 7(1):5–12
- Baik SC, Kim KM, Song SM, Kim DS, Jun JS, Lee SG, Song JY, Park JU, Kang HL, Lee WK et al (2004) Proteomic analysis of the sarcosine-insoluble outer membrane fraction of *Helicobacter pylori* strain 26695. *J Bacteriol* 186(4):949–955
- Bailey TL, Williams N, Misleh C, Li WW (2006) MEME: discovering and analyzing DNA and protein sequence motifs. *Nucleic Acids Res* 34(Web Server issue):W369–W373
- Baird SD, Turcotte M, Korneluk RG, Holcik M (2006) Searching for IRES. *RNA* 12(10):1755–1785
- Baird SD, Lewis SM, Turcotte M, Holcik M (2007) A search for structurally similar cellular internal ribosome entry sites. *Nucleic Acids Res* 35(14):4664–4677
- Baldi P, Brunak S (2001) Bioinformatics: the machine learning approach. The MIT Press, Cambridge, MA
- Bamford DH, Caldentey J, Bamford JK (1995) Bacteriophage PRD1: a broad host range DSDNA tectivirus with an internal membrane. *Adv Virus Res* 45:281–319
- Bao J, Bedford MT (2016) Epigenetic regulation of the histone-to-protamine transition during spermiogenesis. *Reproduction* 151(5):R55–R70
- Baron D, Cocquet J, Xia X, Fellous M, Guiguen Y, Veitia RA (2004) An evolutionary and functional analysis of FoxL2 in rainbow trout gonad differentiation. *J Mol Endocrinol* 33:705–715
- Bastianelli G, Bouillon A, Nguyen C, Crublet E, Petres S, Gorgette O, Le-Nguyen D, Barale JC, Nilges M (2011) Computational reverse-engineering of a spider-venom derived peptide active against Plasmodium falciparum SUB1. *PLoS One* 6(7):e21812
- Bauerfeind P, Garner R, Dunn BE, Mobley HL (1997) Synthesis and activity of *Helicobacter pylori* urease and catalase at low pH. *Gut* 40(1):25–30
- Baumgartner HK, Montrose MH (2004) Regulated alkali secretion acts in tandem with unstirred layers to regulate mouse gastric surface pH. *Gastroenterology* 126(3):774–783
- Beier H, Grimm M (2001) Misreading of termination codons in eukaryotes by natural nonsense suppressor tRNAs. *Nucleic Acids Res* 29(23):4767–4782
- Bell D, Bell AH, Bondaruk J, Hanna EY, Weber RS (2016) In-depth characterization of the salivary adenoid cystic carcinoma transcriptome with emphasis on dominant cell type. *Cancer* 122 (10):1513–1522
- Ben-Gal I, Shani A, Gohr A, Grau J, Arivis S, Shmilovici A, Posch S, Grosse I (2005) Identification of transcription factor binding sites with variable-order Bayesian networks. *Bioinformatics* 21 (11):2657–2666
- Benjamini Y, Hochberg Y (1995) Controlling the false discovery rate: a practical and powerful approach to multiple testing. *J R Stat Soc B* 57(1):289–300
- Benjamini Y, Yekutieli D (2001) The control of the false discovery rate in multiple hypothesis testing under dependency. *Ann Stat* 29:1165–1188
- Bennetzen JL, Hall BD (1982) Codon selection in yeast. *J Biol Chem* 257(6):3026–3031
- Benoit G, Lemaitre C, Lavenier D, Drezen E, Dayris T, Uricaru R, Rizk G (2015) Reference-free compression of high throughput sequencing data with a probabilistic de Bruijn graph. *BMC Bioinform* 16:288

- Benzer S, Champe SP (1962) A change from nonsense to sense in the genetic code. Proc Natl Acad Sci USA 48:1114–1121
- Berg JM, Tymoczko JL, Stryer L (2002) Biochemistry. W. H. Freeman and Co, New York
- Berger MF, Levin JZ, Vijayendran K, Sivachenko A, Adiconis X, Maguire J, Johnson LA, Robinson J, Verhaak RG, Sougnez C et al (2010) Integrative analysis of the melanoma transcriptome. Genome Res 20(4):413–427
- Bergsten E, Uutela M, Li X, Pietras K, Ostman A, Heldin CH, Alitalo K, Eriksson U (2001) PDGF-D is a specific, protease-activated ligand for the PDGF beta-receptor. Nat Cell Biol 3(5):512–516
- Bertholet C, Van Meir E, ten Heggeler-Bordier B, Wittek R (1987) Vaccinia virus produces late mRNAs by discontinuous synthesis. Cell 50(2):153–162
- Besemer J, Borodovsky M (2005) GeneMark: web software for gene finding in prokaryotes, eukaryotes and viruses. Nucleic Acids Res 33(Web Server issue):W451–W454
- Bestor TH, Coxon A (1993) The pros and cons of DNA methylation. Curr Biol 6:384–386
- Betney R, de Silva E, Krishnan J, Stansfield I (2010) Autoregulatory systems controlling translation factor expression: thermostat-like control of translational accuracy. RNA 16(4):655–663
- Beznoskova P, Gunisova S, Valasek LS (2016) Rules of UGA-N decoding by near-cognate tRNAs and analysis of readthrough on short uORFs in yeast. RNA 22(3):456–466
- Bhagwat M, Aravind L (2007) PSI-BLAST tutorial. Methods Mol Biol 395:177–186
- Bhatia B, Ponia SS, Solanki AK, Dixit A, Garg LC (2014) Identification of glutamate ABC-transporter component in Clostridium perfringens as a putative drug target. Bioinformation 10 (7):401–405
- Bibikova M, Barnes B, Tsan C, Ho V, Klotzle B, Le JM, Delano D, Zhang L, Schroth GP, Gunderson KL et al (2011) High density DNA methylation array with single CpG site resolution. Genomics 98(4):288–295
- Bickel DR (2003) Robust cluster analysis of microarray gene expression data with the number of clusters determined biologically. Bioinformatics 19(7):818–824
- Bierme H, Hamon M, Cossart P (2012) Epigenetics and bacterial infections. Cold Spring Harb Perspect Med 2(12):a010272
- Bigaud E, Corrales FJ (2016) Methylthioadenosine (MTA) regulates liver cells proteome and methylproteome: implications in liver biology and disease. Mol Cell Proteomics 15(5):1498–1510
- Birney E, Stamatoyannopoulos JA, Dutta A, Guigo R, Gingeras TR, Margulies EH, Weng Z, Snyder M, Dermitzakis ET, Thurman RE et al (2007) Identification and analysis of functional elements in 1% of the human genome by the ENCODE pilot project. Nature 447(7146):799–816
- Bjorkholm B, Lundin A, Sillen A, Guillemin K, Salama N, Rubio C, Gordon JI, Falk P, Engstrand L (2001) Comparison of genetic divergence and fitness between two subclones of *Helicobacter pylori*. Infect Immun 69(12):7832–7838
- Bjornsson A, Isaksson LA (1996) Accumulation of a mRNA decay intermediate by ribosomal pausing at a stop codon. Nucleic Acids Res 24(9):1753–1757
- Blackburne BP, Whelan S (2013) Class of multiple sequence alignment algorithm affects genomic analysis. Mol Biol Evol 30(3):642–653
- Blakqori G, van Knippenberg I, Elliott RM (2009) Bunyamwera orthobunyavirus S-segment untranslated regions mediate poly(A) tail-independent translation. J Virol 83(8):3637–3646
- Blanchet S, Cornu D, Argentini M, Namy O (2014) New insights into the incorporation of natural suppressor tRNAs at stop codons in *Saccharomyces cerevisiae*. Nucleic Acids Res 42 (15):10061–10072
- Blanchette M, Tompa M (2002) Discovery of regulatory elements by a computational method for phylogenetic footprinting. Genome Res 12(5):739–748
- Blanchette M, Bataille AR, Chen X, Poitras C, Laganiere J, Lefebvre C, Deblois G, Giguere V, Ferretti V, Bergeron D et al (2006) Genome-wide computational prediction of transcriptional regulatory modules reveals new insights into human gene expression. Genome Res 6(5):656–668

- Boehringer D, Thermann R, Ostareck-Lederer A, Lewis JD, Stark H (2005) Structure of the hepatitis C virus IRES bound to the human 80S ribosome: remodeling of the HCV IRES. *Structure* 13(11):1695
- Bogenhaugen DF, Clayton DA (2003) The mitochondrial DNA replication bubble has not burst. *Trends Biochem Sci* 28(7):357–360
- Bolden JE, Peart MJ, Johnstone RW (2006) Anticancer activities of histone deacetylase inhibitors. *Nat Rev Drug Discov* 5(9):769–784
- Borodovsky M, McIninch J (1993) GENMARK: parallel gene recognition for both DNA strands. *Comput Chem* 17:123–133
- Bossi L (1983) Context effects: translation of UAG codon by suppressor tRNA is affected by the sequence following UAG in the message. *J Mol Biol* 164(1):73–87
- Bossi L, Ruth JR (1980) The influence of codon context on genetic code translation. *Nature* 286 (5769):123–127
- Brauch H, Weirich G, Brieger J, Glavac D, Rodl H, Eichinger M, Feurer M, Weidt E, Puranakanittha C, Neuhaus C et al (2000) VHL alterations in human clear cell renal cell carcinoma: association with advanced tumor stage and a novel hot spot mutation. *Cancer Res* 60 (7):1942–1948
- Brazma A, Hingamp P, Quackenbush J, Sherlock G, Spellman P, Stoeckert C, Aach J, Ansorge W, Ball CA, Causton HC et al (2001) Minimum information about a microarray experiment (MIAME)-toward standards for microarray data. *Nat Genet* 29(4):365–371
- Britten RJ (1986) Rates of DNA sequence evolution differ between taxonomic groups. *Science* 231:1393–1398
- Brooks DR, McLennan DA (1991) Phylogeny, ecology and behavior: a research program in comparative biology. University of Chicago Press, Chicago
- Brown CM, Stockwell PA, Trotman CN, Tate WP (1990) Sequence analysis suggests that tetranucleotides signal the termination of protein synthesis in eukaryotes. *Nucleic Acids Res* 18 (21):6339–6345
- Brown M, Hughey R, Krogh A, Mian IS, Sjolander K, Haussler D (1993) Using Dirichlet mixture priors to derive hidden Markov models for protein families. *Proc Int Conf Intell Syst Mol Biol* 1:47–55
- Brown TA, Cecconi C, Tkachuk AN, Bustamante C, Clayton DA (2005) Replication of mitochondrial DNA occurs by strand displacement with alternative light-strand origins, not via a strand-coupled mechanism. *Genes Dev* 19(20):2466–2476
- Brumme ZL, Dong WW, Yip B, Wynhoven B, Hoffman NG, Swanstrom R, Jensen MA, Mullins JI, Hogg RS, Montaner JS et al (2004) Clinical and immunological impact of HIV envelope V3 sequence variation after starting initial triple antiretroviral therapy. *AIDS* 18(4):F1–F9
- Bucklew JA (1990) Large deviation techniques in decision, simulation, and estimation. Wiley, New York
- Bulmer M (1990) The effect of context on synonymous codon usage in genes with low codon usage bias. *Nucleic Acids Res* 18(10):2869–2873
- Bulmer M (1991) The selection-mutation-drift theory of synonymous codon usage. *Genetics* 129:897–907
- Bumann D, Aksu S, Wendland M, Janek K, Zimny-Arndt U, Sabarth N, Meyer TF, Jungblut PR (2002) Proteome analysis of secreted proteins of the gastric pathogen Helicobacter pylori. *Infect Immun* 70(7):3396–3403
- Burge C, Karlin S (1997) Prediction of complete gene structures in human genomic DNA. *J Mol Biol* 268:78–94
- Burge CB, Karlin S (1998) Finding the genes in genomic DNA. *Curr Opin Struct Biol* 8(3):346–354
- Burnham KP, Anderson DR (2002) Model selection and multimodel inference: a practical information-theoretic approach. Springer, New York

- Bury-Mone S, Skouloubris S, Labigne A, De Reuse H (2001) The *Helicobacter pylori* UreI protein: role in adaptation to acidity and identification of residues essential for its activity and for acid activation. Mol Microbiol 42(4):1021–1034
- Calderone TL, Stevens RD, Oas TG (1996) High-level misincorporation of lysine for arginine at AGA codons in a fusion protein expressed in *Escherichia coli*. J Mol Biol 262(4):407–412
- Cao Y, Janke A, Waddell PJ, Westerman M, Takenaka O, Murata S, Okada N, Paabo S, Hasegawa M (1998) Conflict among individual mitochondrial proteins in resolving the phylogeny of eutherian orders. J Mol Evol 47(3):307–322
- Capecchi MR (1967) Polypeptide chain termination in vitro: isolation of a release factor. Proc Natl Acad Sci USA 58(3):1144–1151
- Capuano F, Mulleder M, Kok R, Blom HJ, Ralser M (2014) Cytosine DNA methylation is found in *Drosophila melanogaster* but absent in *Saccharomyces cerevisiae*, *Schizosaccharomyces pombe*, and other yeast species. Anal Chem 86(8):3697–3702
- Cardon LR, Burge C, Clayton DA, Karlin S (1994) Pervasive CpG suppression in animal mitochondrial genomes. Proc Natl Acad Sci USA 91:3799–3803
- Carlini DB (2005) Context-dependent codon bias and messenger RNA longevity in the yeast transcriptome. Mol Biol Evol 22(6):1403–1411
- Carroll J, Fearnley IM, Shannon RJ, Hirst J, Walker JE (2003) Analysis of the subunit composition of complex I from bovine heart mitochondria. Mol Cell Proteomics 2(2):117–126
- Carullo M, Xia X (2008) An extensive study of mutation and selection on the wobble nucleotide in tRNA anticodons in fungal mitochondrial genomes. J Mol Evol 66(5):484–493
- Censini S, Lange C, Xiang Z, Crabtree JE, Ghiaia P, Borodovsky M, Rappuoli R, Covacci A (1996) Cag, a pathogenicity island of *Helicobacter pylori*, encodes type I-specific and disease-associated virulence factors. Proc Natl Acad Sci USA 93(25):14648–14653
- Cesar Sanchez J, Padron G, Santana H, Herrera L (1998) Elimination of an HulFN alpha 2b readthrough species, produced in *Escherichia coli*, by replacing its natural translational stop signal. J Biotechnol 63(3):179–186
- Chakrabarti S, Lanczycki CJ (2007) Analysis and prediction of functionally important sites in proteins. Protein Sci 16(1):4–13
- Chakraborty R (1977) Estimation of time of divergence from phylogenetic studies. Can J Genet Cytol 19:217–223
- Chambaud I, Heilig R, Ferris S, Barbe V, Samson D, Galisson F, Moszer I, Dybvig K, Wroblewski H, Viari A et al (2001) The complete genome sequence of the murine respiratory pathogen *Mycoplasma pulmonis*. Nucleic Acids Res 29(10):2145–2153
- Chan S-W, Egan P (2009) Effects of hepatitis C virus envelope glycoprotein unfolded protein response activation on translation and transcription. Arch Virol 154(10):1631–1640
- Chan PP, Lowe TM (2009) GtRNAdb: a database of transfer RNA genes detected in genomic sequence. Nucleic Acids Res 37(Database issue):D93–D97
- Chang SY, McGary EC, Chang S (1989) Methionine aminopeptidase gene of *Escherichia coli* is essential for cell growth. J Bacteriol 171(7):4071–4072
- Charig CR, Webb DR, Payne SR, Wickham JE (1986) Comparison of treatment of renal calculi by open surgery, percutaneous nephrolithotomy, and extracorporeal shockwave lithotripsy. Br Med J (Clin Res Ed) 292(6524):879–882
- Chen JJ, Peck K, Hong TM, Yang SC, Sher YP, Shih JY, Wu R, Cheng JL, Roffler SR, Wu CW et al (2001) Global analysis of gene expression in invasion by a lung cancer model. Cancer Res 61 (13):5223–5230
- Chen Q, Yan M, Cao Z, Li X, Zhang Y, Shi J, Feng GH, Peng H, Zhang X, Qian J et al (2016) Sperm tsRNAs contribute to intergenerational inheritance of an acquired metabolic disorder. Science 351(6271):397–400
- Chilingaryan A, Gevorgyan N, Vardanyan A, Jones D, Szabo A (2002) Multivariate approach for selecting sets of differentially expressed genes. Math Biosci 176(1):59–69
- Chithambaram S, Prabhakaran R, Xia X (2014a) Differential codon adaptation between dsDNA and ssDNA phages in *escherichia coli*. Mol Biol Evol 31(6):1606–1617

- Chithambaram S, Prabhakaran R, Xia X (2014b) The effect of mutation and selection on codon adaptation in *Escherichia coli* bacteriophage. *Genetics* 197(1):301–315
- Cho RJ, Campbell MJ, Winzeler EA, Steinmetz L, Conway A, Wodicka L, Wolfsberg TG, Gabrielian AE, Landsman D, Lockhart DJ et al (1998) A genome-wide transcriptional analysis of the mitotic cell cycle. *Mol Cell* 2(1):65–73
- Chou PY, Fasman GD (1978a) Empirical predictions of protein conformation. *Annu Rev Biochem* 47:251–276
- Chou PY, Fasman GD (1978b) Prediction of the secondary structure of proteins from their amino acid sequence. *Adv Enzymol Relat Areas Mol Biol* 47:45–148
- Chu C, Qu K, Zhong FL, Artandi SE, Chang HY (2011) Genomic maps of long noncoding RNA occupancy reveal principles of RNA-chromatin interactions. *Mol Cell* 44(4):667–678
- Chu C, Quinn J, Chang HY (2012) Chromatin isolation by RNA purification (ChIRP). *J Vis Exp* 61: e3912
- Chuang SE, Daniels DL, Blattner FR (1993) Global regulation of gene expression in *Escherichia coli*. *J Bacteriol* 175(7):2026–2036
- Clark AT (2015) DNA methylation remodeling in vitro and in vivo. *Curr Opin Genet Dev* 34:82–87
- Claverie JM (1994) Some useful statistical properties of position-weight matrices. *Comput Chem* 18(3):287–294
- Claverie JM, Audic S (1996) The statistical significance of nucleotide position-weight matrix matches. *Comput Appl Biosci* 12(5):431–439
- Clayton DA (1982) Replication of animal mitochondrial DNA. *Cell* 28(4):693–705
- Clayton DA (2000) Transcription and replication of mitochondrial DNA. *Hum Reprod* 15(Suppl 2):11–17
- Cocquet J, De Baere E, Gareil M, Pannetier M, Xia X, Fellous M, Veitia RA (2003) Structure, evolution and expression of the FOXL2 transcription unit. *Cytogenet Genome Res* 101:206–211
- Coessens B, Thijs G, Aerts S, Marchal K, De Smet F, Engelen K, Glenisson P, Moreau Y, Mathys J, De Moor B (2003) INCLUSive: a web portal and service registry for microarray and regulatory sequence analysis. *Nucleic Acids Res* 31(13):3468–3470
- Coghlan A, Wolfe KH (2000) Relationship of codon bias to mRNA concentration and protein length in *Saccharomyces cerevisiae*. *Yeast* 16(12):1131–1145
- Comeron JM, Aguade M (1998) An evaluation of measures of synonymous codon usage bias. *J Mol Evol* 47(3):268–274
- Correa P (1997) *Helicobacter pylori* as a pathogen and carcinogen. *J Physiol Pharmacol* 48(Suppl 4):19–24
- Cottrell JS (1994) Protein identification by peptide mass fingerprinting. *Pept Res* 7(3):115–124
- Cottrell JS, Sutton CW (1996) The identification of electrophoretically separated proteins by peptide mass fingerprinting. *Methods Mol Biol* 61:67–82
- Covacci A, Falkow S, Berg DE, Rappuoli R (1997) Did the inheritance of a pathogenicity island modify the virulence of *Helicobacter pylori*? *Trends Microbiol* 5(5):205–208
- Covell DG, Wallqvist A, Rabow AA, Thanki N (2003) Molecular classification of cancer: unsupervised self-organizing map analysis of gene expression microarray data. *Mol Cancer Ther* 2(3):317–332
- Cox SS, van der Giezen M, Tarr SJ, Crompton MR, Tovar J (2006) Evidence from bioinformatics, expression and inhibition studies of phosphoinositide-3 kinase signalling in *Giardia intestinalis*. *BMC Microbiol* 6:45
- Craigie WJ, Caskey CT (1986) Expression of peptide chain release factor 2 requires high-efficiency frameshift. *Nature* 322(6076):273–275
- Craigie WJ, Caskey CT (1987) The function, structure and regulation of *E. coli* peptide chain release factors. *Biochimie* 69(10):1031–1041
- Craigie WJ, Cook RG, Tate WP, Caskey CT (1985) Bacterial peptide chain release factors: conserved primary structure and possible frameshift regulation of release factor 2. *Proc Natl Acad Sci USA* 82(11):3616–3620

- Craigie WJ, Lee CC, Caskey CT (1990) Recent advances in peptide chain termination. *Mol Microbiol* 4(6):861–865
- Crick FH (1966) Codon—anticodon pairing: the wobble hypothesis. *J Mol Biol* 19(2):548–555
- Curran JF, Yarus M (1988) Use of tRNA suppressors to probe regulation of *Escherichia coli* release factor 2. *J Mol Biol* 203(1):75–83
- Czerwoniec A, Dunin-Horkawicz S, Purta E, Kaminska KH, Kasprzak JM, Bujnicki JM, Grosjean H, Rother K (2009) MODOMICS: a database of RNA modification pathways. 2008 update. *Nucleic Acids Res* 37(Database issue):D118–D121
- Danchin A (2002) The Delphic boat : what genomes tell us. Harvard University Press, Cambridge, MA
- David E, Tramontin T, Zemmel R (2009) Pharmaceutical R&D: the road to positive returns. *Nat Rev Drug Discov* 8(8):609–610
- Davies J, Jones DS, Khorana HG (1966) A further study of misreading of codons induced by streptomycin and neomycin using ribopolynucleotides containing two nucleotides in alternating sequence as templates. *J Mol Biol* 18(1):48–57
- Dayhoff MO, Schwartz RM, Orcutt BC (1978) A model of evolutionary change in proteins. In: Dayhoff MO (ed) *Atlas of protein sequence and structure*. National Biomedical Research Foundation, Washington, DC, pp 345–352
- Delorenzi M, Speed T (2002) An HMM model for coiled-coil domains and a comparison with PSSM-based predictions. *Bioinformatics* 18(4):617–625
- Deng R, Huang M, Wang J, Huang Y, Yang J, Feng J, Wang X (2006) PTreeRec: phylogenetic tree reconstruction based on genome BLAST distance. *Comput Biol Chem* 30(4):300–302
- Deng W, Lee J, Wang H, Miller J, Reik A, Gregory PD, Dean A, Blobel GA (2012) Controlling long-range genomic interactions at a native locus by targeted tethering of a looping factor. *Cell* 149(6):1233–1244
- Deng Q, Ramskold D, Reinius B, Sandberg R (2014a) Single-cell RNA-seq reveals dynamic, random monoallelic gene expression in mammalian cells. *Science* 343(6167):193–196
- Deng W, Rupon JW, Krivega I, Breda L, Motta I, Jahn KS, Reik A, Gregory PD, Rivella S, Dean A et al (2014b) Reactivation of developmentally silenced globin genes by forced chromatin looping. *Cell* 158(4):849–860
- Desper R, Gascuel O (2002) Fast and accurate phylogeny reconstruction algorithms based on the minimum-evolution principle. *J Comput Biol* 9(5):687–705
- Dewey CN, Rogozin IB, Koonin EV (2006) Compensatory relationship between splice sites and exonic splicing signals depending on the length of vertebrate introns. *BMC Genomics* 7:311
- Diehn M, Eisen MB, Botstein D, Brown PO (2000) Large-scale identification of secreted and membrane-associated gene products using DNA microarrays. *Nat Genet* 25(1):58–62
- Dobin A, Davis CA, Schlesinger F, Drenkow J, Zaleski C, Jha S, Batut P, Chaisson M, Gingeras TR (2013) STAR: ultrafast universal RNA-seq aligner. *Bioinformatics* 29(1):15–21
- Dobzhansky T (1973) Nothing in biology makes sense except in the light of evolution. *Am Biol Teach* 35:125–129
- Donly BC, Edgar CD, Adamski FM, Tate WP (1990) Frameshift autoregulation in the gene for *Escherichia coli* release factor 2: partly functional mutants result in frameshift enhancement. *Nucleic Acids Res* 18(22):6517–6522
- Doolittle RF, Hunkapiller MW, Hood LE, Devare SG, Robbins KC, Aaronson SA, Antoniades HN (1983) Simian sarcoma virus onc gene, v-sis, is derived from the gene (or genes) encoding a platelet-derived growth factor. *Science* 221(4607):275–277
- Dorokhov YL, Skulachev MV, Ivanov PA, Zvereva SD, Tjulkina LG, Merits A, Gleba YY, Hohn T, Atabekov JG (2002) Polypurine (A)-rich sequences promote cross-kingdom conservation of internal ribosome entry. *Proc Natl Acad Sci USA* 99(8):5301–5306
- dos Reis M, Savva R, Wernisch L (2004) Solving the riddle of codon usage preferences: a test for translational selection. *Nucleic Acids Res* 32(17):5036–5044 Print 2004

- Doudna JA, Sarnow P (2007) Translation initiation by viral internal ribosome entry sites. In: Mathews MB, Sonenberg N, Hershey J (eds) *Translational control in biology and medicine*. Cold Spring Harbor Laboratory Press, Cold Spring Harbor, pp 129–154
- Drews J, Ryser S (1997) The role of innovation in drug development. *Nat Biotechnol* 15(13):1318–1319
- Drouin G, Daoud H, Xia J (2008) Relative rates of synonymous substitutions in the mitochondrial, chloroplast and nuclear genomes of seed plants. *Mol Phylogenet Evol* 49(3):827–831
- Drummond A, Rambaut A (2007) BEAST: Bayesian evolutionary analysis by sampling trees. *BMC Evol Biol* 7(1):214
- Drummond A, Rodrigo AG (2000) Reconstructing genealogies of serial samples under the assumption of a molecular clock using serial-sample UPGMA. *Mol Biol Evol* 17(12):1807–1815
- Drummond A, Forsberg R, Rodrigo AG (2001) The inference of stepwise changes in substitution rates using serial sequence samples. *Mol Biol Evol* 18(7):1365–1371
- Drummond AJ, Pybus OG, Rambaut A, Forsberg R, Rodrigo AG (2003a) Measurably evolving populations. *Trends Ecol Evol* 18(9):481–488
- Drummond A, Pybus OG, Rambaut A (2003b) Inference of viral evolutionary rates from molecular sequences. *Adv Parasitol* 54:331–358
- Durbin R (1998) *Biological sequence analysis : probabilistic models of proteins and nucleic acids*. Cambridge University Press, Cambridge
- Duret L, Mouchiroud D (1999) Expression pattern and, surprisingly, gene length shape codon usage in *Caenorhabditis*, *Drosophila*, and *Arabidopsis*. *Proc Natl Acad Sci USA* 96(8):4482–4487
- DuRose JB, Scheuner D, Kaufman RJ, Rothblum LI, Niwa M (2009) Phosphorylation of eukaryotic translation initiation factor 2alpha coordinates rRNA transcription and translation inhibition during endoplasmic reticulum stress. *Mol Cell Biol* 29(15):4295–4307
- Duval M, Korepanov A, Fuchsbaier O, Fechter P, Haller A, Fabbretti A, Choulier L, Micura R, Klaholz BP, Romby P et al (2013) Escherichia coli Ribosomal protein S1 unfolds structured mRNAs onto the ribosome for active translation initiation. *PLoS Biol* 11(12):e1001731
- Eckhardt F, Lewin J, Cortese R, Rakyan VK, Attwood J, Burger M, Burton J, Cox TV, Davies R, Down TA et al (2006) DNA methylation profiling of human chromosomes 6, 20 and 22. *Nat Genet* 38(12):1378–1385
- Eddy SR (1996) Hidden Markov models. *Curr Opin Struct Biol* 6(3):361–365
- Eddy SR (1998) Profile hidden Markov models. *Bioinformatics* 14(9):755–763
- Edgar RC (2004) MUSCLE: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Res* 32(5):1792–1797
- Edgar RC, Batzoglou S (2006) Multiple sequence alignment. *Curr Opin Struct Biol* 16(3):368–373
- Efron B (1982) The jackknife, the bootstrap and other resampling plans. Society for Industrial and Applied Mathematics, Philadelphia
- Ehnman M, Missiaglia E, Folestad E, Selfe J, Strell C, Thway K, Brodin B, Pietras K, Shipley J, Ostman A et al (2013) Distinct effects of ligand-induced PDGFRalpha and PDGFRbeta signaling in the human rhabdomyosarcoma tumor cell and stroma cell compartments. *Cancer Res* 73(7):2139–2149
- Ehrenberg M, Tenson T (2002) A new beginning of the end of translation. *Nat Struct Biol* 9(2):85–87
- Einstein A, Russell B, Dewey J, Millikan RA, Dreiser T, Wells HG, Nansen F, Jeans SJ, Babbitt I, Keith SA et al (1931) *Living philosophies*. Simon and Schuster, New York
- Eisen MB, Spellman PT, Brown PO, Botstein D (1998) Cluster analysis and display of genome-wide expression patterns. *Proc Natl Acad Sci U S A* 95(25):14863–14868
- Elf J, Nilsson D, Tenson T, Ehrenberg M (2003) Selective charging of tRNA isoacceptors explains patterns of codon usage. *Science* 300(5626):1718–1722
- Elroy-Stein O, Merrick W (2007) Translation initiation via cellular internal ribosome entry sites. In: Mathews MB, Sonenberg N, Hershey J (eds) *Translational control in biology and medicine*. Cold Spring Harbor Laboratory Press, Cold Spring Harbor, pp 155–172

- Engel E, Peskoff A, Kauffman GL Jr, Grossman MI (1984) Analysis of hydrogen ion concentration in the gastric gel mucus layer. *Am J Phys* 247(4 Pt 1):G321–G338
- Engelberg-Kulka H (1981) UGA suppression by normal tRNA Trp in Escherichia coli: codon context effects. *Nucleic Acids Res* 9(4):983–991
- Epstein CB, Butow RA (2000) Microarray technology – enhanced versatility, persistent challenge. *Curr Opin Biotechnol* 11(1):36–41
- Eswarappa SM, Potdar AA, Koch WJ, Fan Y, Vasu K, Lindner D, Willard B, Graham LM, DiCorleto PE, Fox PL (2014) Programmed translational readthrough generates antiangiogenic VEGF-Ax. *Cell* 157(7):1605–1618
- Evans T, Felsenfeld G, Reitman M (1990) Control of globin gene transcription. *Annu Rev Cell Biol* 6:95–124
- Eyre-Walker A (1996) The close proximity of Escherichia coli genes: consequences for stop codon and synonymous codon use. *J Mol Evol* 42(2):73–78
- Eyre-Walker A, Bulmer M (1993) Reduced synonymous substitution rate at the start of enterobacterial genes. *Nucleic Acids Res* 21:4599–4603
- Ezzell C (2002) Proteins rule. *Sci Am* 286(4):40–47
- Farazi TA, Waksman G, Gordon JI (2001) The biology and enzymology of protein N-myristoylation. *J Biol Chem* 276(43):39501–39504
- Farnham PJ, Platt T (1981) Rho-independent termination: dyad symmetry in DNA causes RNA polymerase to pause during transcription in vitro. *Nucleic Acids Res* 9(3):563–577
- Fasman GD, Chou PY (1974) Prediction of protein conformation: consequences and aspirations. In: Blout ER, Bovey FA, Goodman M, Latan N (eds) *Peptides, polypeptides and proteins*. Wiley, New York, pp 114–125
- Fatemi M, Hermann A, Pradhan S, Jeltsch A (2001) The activity of the murine DNA methyltransferase Dnmt1 is controlled by interaction of the catalytic domain with the N-terminal part of the enzyme leading to an allosteric activation of the enzyme after binding to methylated DNA. *J Mol Biol* 309(5):1189–1199
- Felsenstein J (1973) Maximum-likelihood and minimum-steps methods for estimating evolutionary trees from data on discrete characters. *Syst Zool* 22:240–249
- Felsenstein J (1978a) Cases in which parsimony and compatibility methods will be positively misleading. *Syst Zool* 27:401–410
- Felsenstein J (1978b) The number of evolutionary trees. *Syst Zool* 27:27–33
- Felsenstein J (1981) Evolutionary trees from DNA sequences: a maximum likelihood approach. *J Mol Evol* 17:368–376
- Felsenstein J (1985) Confidence limits on phylogenies: an approach using the bootstrap. *Evolution* 39:783–791
- Felsenstein J (2004) Inferring phylogenies. Sinauer, Sunderland
- Felsenstein J, Churchill GA (1996) A Hidden Markov Model approach to variation among sites in rate of evolution. *Mol Biol Evol* 13(1):93–104
- Feng DF, Doolittle RF (1987) Progressive sequence alignment as a prerequisite to correct phylogenetic trees. *J Mol Evol* 25(4):351–360
- Feng DF, Doolittle RF (1990) Progressive alignment and phylogenetic tree construction of protein sequences. *Methods Enzymol* 183:375–387
- Fernandez-Pinar R, Lo Sciuto A, Rossi A, Ranucci S, Bragonzi A, Imperi F (2015) In vitro and in vivo screening for novel essential cell-envelope proteins in *Pseudomonas aeruginosa*. *Sci Rep* 5:17593
- Fickett JW (1996) Quantitative discrimination of MEF2 sites. *Mol Cell Biol* 16(1):437–441
- Figeys D (2002) Adapting arrays and lab-on-a-chip technology for proteomics. *Proteomics* 2 (4):373–382
- Figeys D (2003a) Novel approaches to map protein interactions. *Curr Opin Biotechnol* 14(1):119–125
- Figeys D (2003b) Proteomics in 2002: a year of technical development and wide-ranging applications. *Anal Chem* 75(12):2891–2905

- Fisher RA (1926) The arrangement of field experiments. *J Minist Agric* 33:503–513
- Fisher RA (1936) The use of multiple measurements in taxonomic problems. *Ann Eugenics* 7:179–188
- Fitch WM (1971) Toward defining the course of evolution: minimum change for a specific tree topology. *Syst Zool* 20:406–416
- Fitch WM, Margoliash E (1967) Construction of phylogenetic trees. *Science* 155:279–284
- Fleischmann RD, Adams MD, White O, Clayton RA, Kirkness EF, Kerlavage AR, Bult CJ, Tomb JF, Dougherty BA, Merrick JM et al (1995) Whole-genome random sequencing and assembly of *Haemophilus influenzae* Rd. *Science* 269(5223):496–512
- Fong TC, Emerson BM (1992) The erythroid-specific protein cGATA-1 mediates distal enhancer activity through a specialized beta-globin TATA box. *Genes Dev* 6(4):521–532
- Forde CE, McCutchen-Malone SL (2002) Characterization of transcription factors by mass spectrometry and the role of SELDI-MS. *Mass Spectrom Rev* 21(6):419–439
- Forrester WC, Epner E, Driscoll MC, Enver T, Brice M, Papayannopoulou T, Groudine M (1990) A deletion of the human beta-globin locus activation region causes a major alteration in chromatin structure and replication across the entire beta-globin locus. *Genes Dev* 4(10):1637–1649
- Frank C, Makkonen H, Dunlop TW, Matilainen M, Vaisanen S, Carlberg C (2005) Identification of pregnane X receptor binding sites in the regulatory regions of genes involved in bile acid homeostasis. *J Mol Biol* 346(2):505–519
- Fraser CM, Gocayne JD, White O, Adams MD, Clayton RA, Fleischmann RD, Bult CJ, Kerlavage AR, Sutton G, Kelley JM et al (1995) The minimal gene complement of *Mycoplasma genitalium*. *Science* 270(5235):397–403
- Frederico LA, Kunkel TA, Shaw BR (1990) A sensitive genetic assay for the detection of cytosine deamination: determination of rate constants and the activation energy. *Biochemistry (Mosc)* 29 (10):2532–2537
- Frishman D, Mironov A, Mewes HW, Gelfand M (1998) Combining diverse evidence for gene recognition in completely sequenced bacterial genomes. *Nucleic Acids Res* 26(12):2941–2947
- Frolova LY, Tsivkovskii RY, Sivolobova GF, Oparina NY, Serpinsky OI, Blinov VM, Tatkov SI, Kisseelev LL (1999) Mutations in the highly conserved GGQ motif of class 1 polypeptide release factors abolish ability of human eRF1 to trigger peptidyl-tRNA hydrolysis. *RNA* 5(8):1014–1020
- Frottin F, Martinez A, Peynot P, Mitra S, Holz RC, Giglione C, Meinnel T (2006) The proteomics of N-terminal methionine cleavage. *Mol Cell Proteomics* 5(12):2336–2349
- Furukawa R, Hachiya T, Ohmomo H, Shiwa Y, Ono K, Suzuki S, Satoh M, Hitomi J, Sobue K, Shimizu A (2016) Intraindividual dynamics of transcriptome and genome-wide stability of DNA methylation. *Sci Rep* 6:26424
- Putcher B, Latter GI, Monardo P, McLaughlin CS, Garrels JI (1999) A sampling of the yeast proteome. *Mol Cell Biol* 19(11):7357–7368
- Gaasterland T, Bekiranov S (2000) Making the most of microarray data [news]. *Nat Genet* 24 (3):204–206
- Gallie DR, Tanguay R (1994) Poly(A) binds to initiation factors and increases cap-dependent translation in vitro. *J Biol Chem* 269(25):17166–17173
- Gal-Mor O, Finlay BB (2006) Pathogenicity islands: a molecular toolbox for bacterial virulence. *Cell Microbiol* 8(11):1707–1719
- Galtier N, Lobry JR (1997) Relationships between genomic G+C content, RNA secondary structures, and optimal growth temperature in prokaryotes. *J Mol Evol* 44(6):632–636
- Gao L, Qi J (2007) Whole genome molecular phylogeny of large dsDNA viruses using composition vector method. *BMC Evol Biol* 7:41
- Gapp K, Jawaid A, Sarkies P, Bohacek J, Pelczar P, Prados J, Farinelli L, Miska E, Mansuy IM (2014) Implication of sperm RNAs in transgenerational inheritance of the effects of early trauma in mice. *Nat Neurosci* 17(5):667–669
- Gascuel O, Steel M (2006) Neighbor-joining revealed. *Mol Biol Evol* 23(11):1997–2000

- Ge Y, Sealfon SC, Speed TP (2008) Some step-down procedures controlling the false discovery rate under dependence. *Stat Sin* 18(3):881–904
- Geller AI, Rich A (1980) A UGA termination suppression tRNATrp active in rabbit reticulocytes. *Nature* 283(5742):41–46
- Geman S, Geman D (1984) Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Trans Pattern Anal Mach Intell* 6:721–741
- Ghaemmaghami S, Huh WK, Bower K, Howson RW, Belle A, Dephoure N, O’Shea EK, Weissman JS (2003) Global analysis of protein expression in yeast. *Nature* 425(6959):737–741
- Gibbs JB (2000) Mechanism-based target identification and drug discovery in cancer research. *Science* 287(5460):1969–1973
- Giglione C, Vallon O, Meinnel T (2003) Control of protein life-span by N-terminal methionine excision. *EMBO J* 22(1):13–23
- Giglione C, Boulard A, Meinnel T (2004) Protein N-terminal methionine excision. *Cell Mol Life Sci* 61(12):1455–1474
- Gilbert WV (2010) Alternative ways to think about cellular internal ribosome entry. *J Biol Chem* 285(38):29033–29038
- Gilbert WV, Zhou K, Butler TK, Doudna JA (2007) Cap-independent translation is required for starvation-induced differentiation in yeast. *Science* 317(5842):1224–1227
- Gillespie JH (1991) The causes of molecular evolution. Oxford University Press, Oxford
- Gojobori T, Li WH, Graur D (1982) Patterns of nucleotide substitution in pseudogenes and functional genes. *J Mol Evol* 18(5):360–369
- Gonzalez B, Ceciliani F, Galizzi A (2003) Growth at low temperature suppresses readthrough of the UGA stop codon during the expression of *Bacillus subtilis* flgM gene in *Escherichia coli*. *J Biotechnol* 101(2):173–180
- Gorodkin J, Heyer LJ, Brunak S, Stormo GD (1997) Displaying the information contents of structural RNA alignments: the structure logos. *Comput Appl Biosci* 13(6):583–586
- Goto M, Washio T, Tomita M (2000) Causal analysis of CpG suppression in the *Mycoplasma* genome. *Microb Comp Genomics* 5(1):51–58
- Gotoh O (1982) An improved algorithm for matching biological sequences. *J Mol Biol* 162(3):705–708
- Gould SJ, Vrba ES (1982) Exaptation – a missing term in the science of form. *Paleobiology* 8:4–15
- Gouy M (1987) Codon contexts in enterobacterial and coliphage genes. *Mol Biol Evol* 4(4):426–444
- Gouy M, Gautier C (1982) Codon usage in bacteria: correlation with gene expressivity. *Nucleic Acids Res* 10:7055–7064
- Gowri-Shankar V, Rattray M (2007) A reversible jump method for Bayesian phylogenetic inference with a nonhomogeneous substitution model. *Mol Biol Evol* 24(6):1286–1299
- Grahn AM, Butcher SJ, Bamford JKH, Bamford DH (2006) PRD1: dissecting the genome, structure and entry. In: Calendar R (ed) *The bacteriophages*. Oxford University Press, Oxford, pp 176–185
- Gramm J, Niedermeier R (2002) Breakpoint medians and breakpoint phylogenies: a fixed-parameter approach. *Bioinformatics* 18(Suppl 2):S128–S139
- Grantham R (1974) Amino acid difference formula to help explain protein evolution. *Science* 185:862–864
- Graveley BR (2005) Mutually exclusive splicing of the insect Dscam pre-mRNA directed by competing intronic RNA secondary structures. *Cell* 123(1):65–73
- Grech B, Maetschke S, Mathews S, Timms P (2007) Genome-wide analysis of chlamydiae for promoters that phylogenetically footprint. *Res Microbiol* 158(8–9):685–693
- Grigg GW (1996) Sequencing 5-methylcytosine residues by the bisulphite method. *DNA Seq* 6 (4):189–198
- Grigg G, Clark S (1994) Sequencing 5-methylcytosine residues in genomic DNA. *BioEssays* 16 (6):431–436

- Grosjean H, Marck C, de Crecy-Lagard V (2007) The various strategies of codon decoding in organisms of the three domains of life: evolutionary implications. *Nucleic Acids Symp Ser (Oxf)* 51:15–16
- Grosjean H, de Crecy-Lagard V, Marck C (2010) Deciphering synonymous codons in the three domains of life: co-evolution with specific tRNA modification enzymes. *FEBS Lett* 584(2):252–264
- Grossi de Sa MF, Standart N, Martins de Sa C, Akhayat O, Huesca M, Scherrer K (1988) The poly (A)-binding protein facilitates in vitro translation of poly(A)-rich mRNA. *Eur J Biochem* 176 (3):521–526
- Guindon S, Dufayard JF, Lefort V, Anisimova M, Hordijk W, Gascuel O (2010) New algorithms and methods to estimate maximum-likelihood phylogenies: assessing the performance of PhyML 3.0. *Syst Biol* 59(3):307–321
- Gumbel EJ (1958) Statistics of extremes. Columbia University Press, New York
- Gupta SK, Kececioglu JD, Schaffer AA (1995) Improving the practical space and time efficiency of the shortest-paths approach to sum-of-pairs multiple sequence alignment. *J Comput Biol* 2 (3):459–472
- Gusfield D (1997) Algorithms on strings, trees, and sequences : computer science and computational biology. Cambridge University Press, Cambridge
- Gygi SP, Rochon Y, Franzia BR, Aebersold R (1999) Correlation between protein and mRNA abundance in yeast. *Mol Cell Biol* 19(3):1720–1730
- Haas J, Park E-C, Seed B (1996) Codon usage limitation in the expression of HIV-1 envelope glycoprotein. *Curr Biol* 6(3):315–324
- Hacker J, Kaper JB (2000) Pathogenicity islands and the evolution of microbes. *Annu Rev Microbiol* 54:641–679
- Hacker J, Blum-Oehler G, Muhldorfer I, Tschape H (1997) Pathogenicity islands of virulent bacteria: structure, function and impact on microbial evolution. *Mol Microbiol* 23(6):1089–1097
- Hamajima N, Goto Y, Nishio K, Tanaka D, Kawai S, Sakakibara H, Kondo T (2004) *Helicobacter pylori* eradication as a preventive tool against gastric cancer. *Asian Pac J Cancer Prev* 5(3):246–252
- Hanada K, Suzuki Y, Gojobori T (2004) A large variation in the rates of synonymous substitution for RNA viruses and its relationship to a diversity of viral infection and transmission modes. *Mol Biol Evol* 21(6):1074–1080
- Hartigan JA (1975) Clustering algorithms. Wiley, New York
- Hasegawa M, Kishino H (1989) Heterogeneity of tempo and mode of mitochondrial DNA evolution among mammalian orders. *Jpn J Genet* 64(4):243–258
- Hasegawa M, Kishino H, Yano T (1985) Dating of the human-ape splitting by a molecular clock of mitochondrial DNA. *J Mol Evol* 22(2):160–174
- Haustead DJ, Stevenson A, Saxena V, Marriage F, Firth M, Silla R, Martin L, Adcroft KF, Rea S, Day PJ et al (2016) Transcriptome analysis of human ageing in male skin shows mid-life period of variability and central role of NF-kappaB. *Sci Rep* 6:26846
- Hayes WS, Borodovsky M (1998) How to interpret an anonymous bacterial genome: machine learning approach to gene identification. *Genome Res* 8(11):1154–1171
- Heath JR, Ribas A, Mischel PS (2016) Single-cell analysis tools for drug discovery and development. *Nat Rev Drug Discov* 15(3):204–216
- Hein J (1990) A unified approach to phylogenies and alignments. *Methods Enzymol* 183:625–644
- Hein J (1994) TreeAlign. *Methods Mol Biol* 25:349–364
- Hendy MD, Penny D (1982) Branch and bound algorithms to determine minimal evolutionary trees. *Math Biosci* 60:133–142
- Hendy MD, Penny D (1989) A framework for the quantitative study of evolutionary trees. *Syst Zool* 38:297–309
- Henikoff S, Henikoff JG (1992) Amino acid substitution matrices from protein blocks. *Proc Natl Acad Sci U S A* 89:10915–10919

- Henz SR, Huson DH, Auch AF, Nieselt-Struwe K, Schuster SC (2005) Whole-genome prokaryotic phylogeny. *Bioinformatics* 21(10):2329–2335
- Herman JL, Challis CJ, Novak A, Hein J, Schmidler SC (2014) Simultaneous Bayesian estimation of alignment and phylogeny under a joint model of protein sequence and structure. *Mol Biol Evol* 31(9):2251–2266
- Hernández G (2008) Was the initiation of translation in early eukaryotes IRES-driven? *Trends Biochem Sci* 33(2):58
- Hernandez G, Vazquez-Pianzola P, Sierra JM, Rivera-Pomar R (2004) Internal ribosome entry site drives cap-independent translation of reaper and heat shock protein 70 mRNAs in *Drosophila* embryos. *RNA* 10(11):1783–1797
- Herniou EA, Luque T, Chen X, Vlak JM, Winstanley D, Cory JS, O'Reilly DR (2001) Use of whole genome sequence data to infer baculovirus phylogeny. *J Virol* 75(17):8117–8126
- Hertz GZ, Stormo GD (1999) Identifying DNA and protein patterns with statistically significant alignments of multiple sequences. *Bioinformatics* 15(7–8):563–577
- Hertz GZ, Hartzell GW 3rd, Stormo GD (1990) Identification of consensus patterns in unaligned DNA sequences known to be functionally related. *Comput Appl Biosci* 6(2):81–92
- Hertzberg L, Izraeli S, Domany E (2007) STOP: searching for transcription factor motifs using gene expression. *Bioinformatics* 23(14):1737–1743
- Hiard S, Maree R, Colson S, Hoskisson PA, Titgemeyer F, van Wezel GP, Joris B, Wehenkel L, Rigali S (2007) PREDetector: a new tool to identify regulatory elements in bacterial genomes. *Biochem Biophys Res Commun* 357(4):861–864
- Hickson RE, Simon C, Perrey SW (2000) The performance of several multiple-sequence alignment programs in relation to secondary-structure features for an rRNA sequence. *Mol Biol Evol* 17 (4):530–539
- Higashi K, Kashiwagi K, Taniguchi S, Terui Y, Yamamoto K, Ishihama A, Igarashi K (2006) Enhancement of +1 frameshift by polyamines during translation of polypeptide release factor 2 in *Escherichia coli*. *J Biol Chem* 281(14):9527–9537
- Higgins DG (1994) CLUSTAL V: multiple alignment of DNA and protein sequences. *Methods Mol Biol* 25:307–318
- Higgs PG, Attwood TK (2005) Bioinformatics and molecular evolution. Blackwell, Malden
- Higgs PG, Ran W (2008) Coevolution of codon usage and tRNA genes leads to alternative stable states of biased codon usage. *Mol Biol Evol* 25(11):2279–2291
- Hiller K, Grote A, Scheer M, Munch R, Jahn D (2004) PrediSi: prediction of signal peptides and their cleavage positions. *Nucleic Acids Res* 32(Web Server issue):W375–W379
- Hirao I, Kimoto M (2010) Expansion of the genetic alphabet in nucleic acids by creating new base pairs. In: Mayer G (ed) The chemical biology of nucleic acids. Wiley, Chichester, pp 39–62
- Hirsh D, Gold L (1971) Translation of the UGA triplet in vitro by tryptophan transfer RNA's. *J Mol Biol* 58(2):459–468
- Hirst JD, Sternberg MJ (1991) Prediction of ATP/GTP-binding motif: a comparison of a perceptron type neural network and a consensus sequence method [corrected]. *Protein Eng* 4(6):615–623
- Hoagland MB, Stephenson ML, Scott JF, Hecht LI, Zamecnik PC (1958) A soluble ribonucleic acid intermediate in protein synthesis. *J Biol Chem* 231(1):241–257
- Hobolth A, Christensen OF, Mailund T, Schierup MH (2007) Genomic relationships and speciation times of human, chimpanzee, and gorilla inferred from a coalescent hidden Markov model. *PLoS Genet* 3(2):e7
- Hofacker IL (2003) Vienna RNA secondary structure server. *Nucleic Acids Res* 31(13):3429–3431
- Hofacker IL, Fekete M, Stadler PF (2002) Secondary structure prediction for aligned RNA sequences. *J Mol Biol* 319(5):1059–1066
- Hofer A, Steverding D, Chabes A, Brun R, Thelander L (2001) *Trypanosoma brucei* CTP synthetase: a target for the treatment of African sleeping sickness. *Proc Natl Acad Sci U S A* 98(11):6412–6416

- Hogeweg P, Hesper aB (1984) The alignment of sets of sequences and the construction of phylogenetic trees: an integrated method. *J Mol Evol* 20:175–186
- Holmes I, Bruno WJ (2001) Evolutionary HMMs: a Bayesian approach to multiple alignment. *Bioinformatics* 17(9):803–820
- Holstege FC, Jennings EG, Wyrick JJ, Lee TI, Hengartner CJ, Green MR, Golub TR, Lander ES, Young RA (1998) Dissecting the regulatory circuitry of a eukaryotic genome. *Cell* 95(5):717–728 Transcriptomic data at [http://web.wi.mit.edu/young/pub/data/orf\\_transcriptome.txt](http://web.wi.mit.edu/young/pub/data/orf_transcriptome.txt)
- Hou C, Zhao H, Tanimoto K, Dean A (2008) CTCF-dependent enhancer-blocking by alternative chromatin loop formation. *Proc Natl Acad Sci U S A* 105(51):20398–20403
- Hua S, Sun Z (2001) Support vector machine approach for protein subcellular localization prediction. *Bioinformatics* 17(8):721–728
- Hudson RR (1992) Gene trees, species trees and the segregation of ancestral alleles. *Genetics* 131 (2):509–513
- Huelsenbeck JP, Larget B, Alfaro ME (2004) Bayesian phylogenetic model selection using reversible jump Markov chain Monte Carlo. *Mol Biol Evol* 21(6):1123–1133
- Hughes D (1987) Mutant forms of tufA and tufB independently suppress nonsense mutations. *J Mol Biol* 197(4):611–615
- Hui A, de Boer HA (1987) Specialized ribosome system: preferential translation of a single mRNA species by a subpopulation of mutated ribosomes in *Escherichia coli*. *Proc Natl Acad Sci U S A* 84(14):4762–4766
- Hunt RH (2004) Will eradication of *Helicobacter pylori* infection influence the risk of gastric cancer? *Am J Med* 117(Suppl 5A):86S–91S
- Hurst LD, Merchant AR (2001) High guanine-cytosine content is not an adaptation to high temperature: a comparative analysis amongst prokaryotes. *Proc R Soc Lond B* 268:493–497
- Huynen M, Dandekar T, Bork P (1998) Differential genome analysis applied to the species-specific features of *Helicobacter pylori*. *FEBS Lett* 426(1):1–5
- Hwang S, Gou Z, Kuznetsov IB (2007) DP-Bind: a web server for sequence-based prediction of DNA-binding residues in DNA-binding proteins. *Bioinformatics* 23(5):634–636
- Hyatt D, Chen GL, Locascio PF, Land ML, Larimer FW, Hauser LJ (2010) Prodigal: prokaryotic gene recognition and translation initiation site identification. *BMC Bioinform* 11:119
- Igarashi K, Kashiwagi K (2006) Polyamine Modulon in *Escherichia coli*: genes involved in the stimulation of cell growth by polyamines. *J Biochem* 139(1):11–16
- Ikemura T (1981a) Correlation between the abundance of *Escherichia coli* transfer RNAs and the occurrence of the respective codons in its protein genes. *J Mol Biol* 146:1–21
- Ikemura T (1981b) Correlation between the abundance of *Escherichia coli* transfer RNAs and the occurrence of the respective codons in its protein genes: a proposal for a synonymous codon choice that is optimal for the *E coli* translational system. *J Mol Biol* 151:389–409
- Ikemura T (1982) Correlation between the abundance of yeast transfer RNAs and the occurrence of the respective codons in protein genes. Differences in synonymous codon choice patterns of yeast and *Escherichia coli* with reference to the abundance of isoaccepting transfer RNAs. *J Mol Biol* 158(4):573–597
- Ikemura T (1985) Codon usage and tRNA content in unicellular and multicellular organisms. *Mol Biol Evol* 2:13–34
- Ikemura T (1992) Correlation between codon usage and tRNA content in microorganisms. In: Hatfield DL, Lee BJ, Pirtle RM (eds) Transfer RNA in protein synthesis. CRC Press, Boca Raton, pp 87–111
- Ilkow CS, Mancinelli V, Beatch MD, Hobman TC (2008) Rubella virus capsid protein interacts with poly(a)-binding protein and inhibits translation. *J Virol* 82(9):4284–4294
- Ingolia NT (2010) Genome-wide translational profiling by ribosome footprinting. *Methods Enzymol* 470:119–142
- Ingolia NT (2014) Ribosome profiling: new views of translation, from single codons to genome scale. *Nat Rev Genet* 15(3):205–213

- Ingolia NT (2016) Ribosome footprint profiling of translation throughout the Genome. *Cell* 165(1):22–33
- Ingolia NT, Ghaemmaghami S, Newman JRS, Weissman JS (2009) Genome-wide analysis in vivo of translation with nucleotide resolution using ribosome profiling. *Science* 324(5924):218–223
- Ingolia NT, Lareau LF, Weissman JS (2011) Ribosome profiling of mouse embryonic stem cells reveals the complexity and dynamics of mammalian proteomes. *Cell* 147(4):789–802
- Ingolia NT, Brar GA, Stern-Ginossar N, Harris MS, Talhouarne GJ, Jackson SE, Wills MR, Weissman JS (2014) Ribosome profiling reveals pervasive translation outside of annotated protein-coding genes. *Cell Rep* 8(5):1365–1379
- Ingram VM (1956) A specific chemical difference between the globins of normal human and sickle-cell anaemia haemoglobin. *Nature* 178(4537):792–794
- Ingram VM (1957) Gene mutations in human haemoglobin: the chemical difference between normal and sickle cell haemoglobin. *Nature* 180(4581):326–328
- Ingrosso D, Perna AF (2009) Epigenetics in hyperhomocysteinemic states. A special focus on uremia. *Biochim Biophys Acta* 1790(9):892–899
- Ingrosso D, Cimmino A, Perna AF, Masella L, De Santo NG, De Bonis ML, Vacca M, D'Esposito M, D'Urso M, Galletti P et al (2003) Folate treatment and unbalanced methylation and changes of allelic expression induced by hyperhomocysteinaemia in patients with uraemia. *Lancet* 361(9370):1693–1699
- Ink BS, Pickup DJ (1990) Vaccinia virus directs the synthesis of early mRNAs containing 5' poly(A) sequences. *Proc Natl Acad Sci U S A* 87(4):1536–1540
- Insinga A, Minucci S, Pelicci PG (2005a) Mechanisms of selective anticancer action of histone deacetylase inhibitors. *Cell Cycle* 4(6):741–743
- Insinga A, Monestiroli S, Ronzoni S, Gelmetti V, Marchesi F, Viale A, Altucci L, Nervi C, Minucci S, Pelicci PG (2005b) Inhibitors of histone deacetylases induce tumor-selective apoptosis through activation of the death receptor pathway. *Nat Med* 11(1):71–76
- Ito T, Bulger M, Pazin MJ, Kobayashi R, Kadonaga JT (1997) ACF, an ISWI-containing and ATP-utilizing chromatin assembly and remodeling factor. *Cell* 90(1):145–155
- Ito K, Uno M, Nakamura Y (2000) A tripeptide ‘anticodon’ deciphers stop codons in messenger RNA. *Nature* 403(6770):680–684
- Jackson RJ, Hellen CU, Pestova TV (2010) The mechanism of eukaryotic translation initiation and principles of its regulation. *Nat Rev Mol Cell Biol* 11(2):113–127
- Jacob F (1982) The possible and the actual. University of Washington Press, Seattle, p 70
- Jacob F (1988) The statue within: an autobiography. Basic Books, Inc., New York
- Jacob F, Monod J (1961) Genetic regulatory mechanisms in the synthesis of proteins. *J Mol Biol* 3:318–356
- Jacobson A, Favreau M (1983) Possible involvement of poly(A) in protein synthesis. *Nucleic Acids Res* 11(18):6353–6368
- James P, Quadrini M, Carafoli E, Gonnet G (1994) Protein identification in DNA databases by peptide mass fingerprinting. *Protein Sci* 3(8):1347–1350
- Jan E, Sarnow P (2002) Factorless ribosome assembly on the internal ribosome entry site of cricket paralysis virus. *J Mol Biol* 324(5):889–902
- Jan E, Thompson SR, Wilson JE, Pestova TV, Hellen CU, Sarnow P (2001) Initiator Met-tRNA-independent translation mediated by an internal ribosome entry site element in cricket paralysis virus-like insect viruses. *Cold Spring Harb Symp Quant Biol* 66:285–292
- Janin L, Schulz-Trieglaff O, Cox AJ (2014) BEETL-fastq: a searchable compressed archive for DNA reads. *Bioinformatics* 30(19):2796–2801
- Jank P, Shindo-Okada N, Nishimura S, Gross HJ (1977) Rabbit liver tRNA<sup>Val</sup>I. Primary structure and unusual codon recognition. *Nucleic Acids Res* 4(6):1999–2008
- Jayaswal V, Jermiin LS, Robinson J (2005) Estimation of phylogeny using a general markov model. *Evol Bioinform Online* 1:62–80

- Jenkins GM, Holmes EC (2003) The extent of codon usage bias in human RNA viruses and its evolutionary origin. *Virus Res* 92(1):1–7
- Jensen JL, Hein J (2005) Gibbs sampler for statistical multiple alignment. *Stat Sin* 15:889–907
- Jia W, Higgs PG (2008) Codon usage in mitochondrial genomes: distinguishing context-dependent mutation from translational selection. *Mol Biol Evol* 25(2):339–351
- Jin P, Alisch RS, Warren ST (2004a) RNA and microRNAs in fragile X mental retardation. *Nat Cell Biol* 6(11):1048–1053
- Jin VX, Leu YW, Liyanarachchi S, Sun H, Fan M, Nephew KP, Huang TH, Davuluri RV (2004b) Identifying estrogen receptor alpha target genes using integrated computational genomics and chromatin immunoprecipitation microarray. *Nucleic Acids Res* 32(22):6627–6635
- Jin VX, O'Geen H, Iyengar S, Green R, Farnham PJ (2007) Identification of an OCT4 and SRY regulatory module using integrated computational and experimental genomics approaches. *Genome Res* 17(6):807–817
- Johnston TC, Parker J (1985) Streptomycin-induced, third-position misreading of the genetic code. *J Mol Biol* 181(2):313–315
- Johnston TC, Borgia PT, Parker J (1984) Codon specificity of starvation induced misreading. *Mol Genet MGG* 195(3):459–465
- Jones DT, Taylor WR, Thornton JM (1992) The rapid generation of mutation data matrices from protein sequences. *Comput Appl Biosci* 8:275–282
- Jorgensen F, Adamski FM, Tate WP, Kurland CG (1993) Release factor-dependent false stops are infrequent in *Escherichia coli*. *J Mol Biol* 230(1):41–50
- Josse J, Kaiser AD, Kornberg A (1961) Enzymatic synthesis of deoxyribonucleic acid VII. Frequencies of nearest neighbor base-sequences in deoxyribonucleic acid. *J Biol Chem* 236:864–875
- Jukes TH, Cantor CR (1969) Evolution of protein molecules. In: Munro HN (ed) *Mammalian protein metabolism*. Academic, New York, pp 21–123
- Kaishima M, Ishii J, Matsuno T, Fukuda N, Kondo A (2016) Expression of varied GFPs in *Saccharomyces cerevisiae*: codon optimization yields stronger than expected expression and fluorescence intensity. *Sci Rep* 6:35932
- Kamalakaran S, Radhakrishnan SK, Beck WT (2005) Identification of estrogen-responsive genes using a genome-wide analysis of promoter elements for transcription factor binding sites. *J Biol Chem* 280(22):21491–21497
- Kanehisa M (2013) Molecular network analysis of diseases and drugs in KEGG. *Methods Mol Biol* 939:263–275
- Kanehisa M, Sato Y, Kawashima M, Furumichi M, Tanabe M (2016) KEGG as a reference resource for gene and protein annotation. *Nucleic Acids Res* 44(D1):D457–D462
- Kaneko T, Tanaka A, Sato S, Kotani H, Sazuka T, Miyajima N, Sugiura M, Tabata S (1995) Sequence analysis of the genome of the unicellular cyanobacterium *Synechocystis* sp. strain PCC6803. I. Sequence features in the 1 Mb region from map positions 64% to 92% of the genome. *DNA Res* 2(4):153–166 191–8
- Kaneko T, Sato S, Kotani H, Tanaka A, Asamizu E, Nakamura Y, Miyajima N, Hiroswa M, Sugiura M, Sasamoto S et al (1996) Sequence analysis of the genome of the unicellular cyanobacterium *Synechocystis* sp. strain PCC6803. II. Sequence determination of the entire genome and assignment of potential protein-coding regions. *DNA Res* 3(3):109–136
- Karlin S, Burge C (1995) Dinucleotide relative abundance extremes: a genomic signature. *TIG* 11 (7):283–290
- Katsafanas GC, Moss B (2007) Colocalization of transcription and translation within cytoplasmic poxvirus factories coordinates viral expression and subjugates host functions. *Cell Host Microbe* 2(4):221
- Karlin S, Mrazek J (1996) What drives codon choices in human genes. *J Mol Biol* 262:459–472
- Kass RE, Raftery AE (1995) Bayes factors. *J Am Stat Assoc* 90(430):773–795
- Katoh K, Toh H (2008) Recent developments in the MAFFT multiple sequence alignment program. *Brief Bioinform* 9(4):286–298

- Katoh K, Toh H (2010) Parallelization of the MAFFT multiple sequence alignment program. *Bioinformatics* 26(15):1899–1900
- Katoh K, Kuma K, Toh H, Miyata T (2005) MAFFT version 5: improvement in accuracy of multiple sequence alignment. *Nucleic Acids Res* 33(2):511–518
- Katoh K, Asimenos G, Toh H (2009) Multiple alignment of DNA sequences with MAFFT. *Methods Mol Biol* 537:39–64
- Kawashima T, Douglass S, Gabunilas J, Pellegrini M, Chanfreau GF (2014) Widespread use of non-productive alternative splice sites in *Saccharomyces cerevisiae*. *PLoS Genet* 10(4): e1004249
- Kazan K (2003) Alternative splicing and proteome diversity in plants: the tip of the iceberg has just emerged. *Trends Plant Sci* 8(10):468–471
- Keeling PJ, Doolittle WF (1996) A non-canonical genetic code in an early diverging eukaryotic lineage. *EMBO J* 15(9):2285–2290
- Kersulyte D, Chalkauskas H, Berg DE (1999) Emergence of recombinant strains of *Helicobacter pylori* during human infection. *Mol Microbiol* 31(1):31–43
- Kim H, Park H (2004) Prediction of protein relative solvent accessibility with support vector machines and long-range interaction 3D local descriptor. *Proteins* 54(3):557–562
- Kim DW, Lee KH, Lee D (2005) Detecting clusters of different geometrical shapes in microarray gene expression data. *Bioinformatics* 21(9):1927–1934
- Kimura M (1968) Evolutionary rate at the molecular level. *Nature* 217:624–626
- Kimura M (1977) Preponderance of synonymous changes as evidence for the neutral theory of molecular evolution. *Nature* 267:275–276
- Kimura M (1980) A simple method for estimating evolutionary rates of base substitutions through comparative studies of nucleotide sequences. *J Mol Evol* 16:111–120
- Kimura M (1983) The neutral theory of molecular evolution. Cambridge University Press, Cambridge
- Kimura M, Ohta T (1972) On the stochastic model for estimation of mutational distance between homologous proteins. *J Mol Evol* 2:87–90
- King MC, Jukes TH (1969) Non-Darwinian evolution. *Science* 164:788–798
- Kingsford C, Patro R (2015) Reference-based compression of short-read sequences using path encoding. *Bioinformatics* 31(12):1920–1928
- Kioussis D, Vanin E, deLange T, Flavell RA, Grosveld FG (1983) Beta-globin gene inactivation by DNA translocation in gamma beta-thalassaemia. *Nature* 306(5944):662–666
- Kishino H, Hasegawa M (1989) Evaluation of the maximum likelihood estimate of the evolutionary tree topologies from DNA sequence data, and the branching order in Hominoidea. *J Mol Evol* 29:170–179
- Kishino H, Hasegawa M (1990) Converting distance to time: application to human evolution. *Methods Enzymol* 183:550–570
- Kjer KM (1995) Use of ribosomal-RNA secondary structure in phylogenetic studies to identify homologous positions – an example of alignment and data presentation from the frogs. *Mol Phylogenet Evol* 4(3):314–330
- Kliman RM, Bernal CA (2005) Unusual usage of AGG and TTG codons in humans and their viruses. *Gene* 352:92
- Kobayashi H, Akitomi J, Fujii N, Kobayashi K, Altaf-Ul-Amin M, Kurokawa K, Ogasawara N, Kanaya S (2007) The entire organization of transcription units on the *Bacillus subtilis* genome. *BMC Genomics* 8:197
- Kodama Y, Shumway M, Leinonen R (2012) The sequence read archive: explosive growth of sequencing data. *Nucleic Acids Res* 40(Database issue):D54–D56
- Kohonen T (2001) Self-organizing maps. Springer, Berlin
- Komar AA, Hatzoglou M (2005) Internal ribosome entry sites in cellular mRNAs: mystery of their existence. *J Biol Chem* 280(25):23425–23428

- Korenke GC, Fuchs S, Krasemann E, Doerr HG, Wilichowski E, Hunneman DH, Hanefeld F (1996) Cerebral adrenoleukodystrophy (ALD) in only one of monozygotic twins with an identical ALD genotype. *Ann Neurol* 40(2):254–257
- Korkmaz G, Holm M, Wiens T, Sanyal S (2014) Comprehensive analysis of stop codon usage in bacteria and its correlation with release factor abundance. *J Biol Chem* 289(44):30334–30342
- Kornblith AR (2005) Promoter usage and alternative splicing. *Curr Opin Cell Biol* 17(3):262–268
- Kozak M (1978) How do eucaryotic ribosomes select initiation regions in messenger RNA? *Cell* 15(4):1109–1123
- Kozak M (1980a) Evaluation of the “scanning model” for initiation of protein synthesis in eucaryotes. *Cell* 22(1 Pt 1):7–8
- Kozak M (1980b) Influence of mRNA secondary structure on binding and migration of 40S ribosomal subunits. *Cell* 19(1):79–90
- Kozak M (1981) Possible role of flanking nucleotides in recognition of the AUG initiator codon by eukaryotic ribosomes. *Nucleic Acids Res* 9(20):5233–5252
- Kozak M (1986) Point mutations define a sequence flanking the AUG initiator codon that modulates translation by eukaryotic ribosomes. *Cell* 44(2):283–292
- Kozak M (1991) Effects of long 5' leader sequences on initiation by eukaryotic ribosomes in vitro. *Gene Expr* 1(2):117–125
- Kozak M (1997) Recognition of AUG and alternative initiator codons is augmented by G in position +4 but is not generally affected by the nucleotides in positions +5 and +6. *EMBO J* 16(9):2482–2492
- Kozak M (1999) Initiation of translation in prokaryotes and eukaryotes. *Gene* 234(2):187–208
- Kozak M (2005) A second look at cellular mRNA sequences said to function as internal ribosome entry sites. *Nucleic Acids Res* 33(20):6593–6602
- Kozak M (2007) Some thoughts about translational regulation: forward and backward glances. *J Cell Biochem* 102(2):280–290
- Krasemann EW, Meier V, Korenke GC, Hunneman DH, Hanefeld F (1996) Identification of mutations in the ALD-gene of 20 families with adrenoleukodystrophy/adrenomyeloneuropathy. *Hum Genet* 97(2):194–197
- Kreutzer DA, Essigmann JM (1998) Oxidized, deaminated cytosines are a source of C → T transitions in vivo. *Proc Natl Acad Sci U S A* 95(7):3578–3582
- Krogh A, Mian IS, Haussler D (1994) A hidden Markov model that finds genes in *E. coli* DNA. *Nucleic Acids Res* 22(22):4768–4778
- Kudla G, Murray AW, Tollervey D, Plotkin JB (2009) Coding-sequence determinants of gene expression in *Escherichia coli*. *Science* 324(5924):255–258
- Kullback S (1959) Information theory and statistics. Wiley, New York
- Kullback S (1987) The Kullback-Leibler distance. *Am Stat* 41:340–341
- Kullback S, Leibler RA (1951) On information and sufficiency. *Ann Math Stat* 22:79–86
- Kumar S, Filipski A (2007) Multiple sequence alignment: in pursuit of homologous DNA positions. *Genome Res* 17(2):127–135
- Kumar KK, Shelokar PS (2008) An SVM method using evolutionary information for the identification of allergenic proteins. *Bioinformation* 2(6):253–256
- Kumar S, Stecher G, Tamura K (2016) MEGA7: molecular evolutionary genetics analysis version 7.0 for bigger datasets. *Mol Biol Evol* 33(7):1870–1874
- Kungulovski G, Jeltsch A (2016) Epigenome editing: state of the art, concepts, and perspectives. *Trends Genet* 32(2):101–113
- Kurland CG (1987) Strategies for efficiency and accuracy in gene expression. *Trends Biochem Sci* 12:126
- Kutlar A (2007) Sickle cell disease: a multigenic perspective of a single gene disorder. *Hemoglobin* 31(2):209–224
- Kuznetsov IB, Gou Z, Li R, Hwang S (2006) Using evolutionary and structural information to predict DNA-binding sites on DNA-binding proteins. *Proteins* 64(1):19–27
- Kypr J, Mrazek JAN (1987) Unusual codon usage of HIV. *Nature* 327(6117):20

- Kyte J, Doolittle RF (1982) A simple method for displaying the hydropathic character of a protein. *J Mol Biol* 157:105–132
- Lacerda R, Menezes J, Romao L (2016) More than just scanning: the importance of cap-independent mRNA translation initiation for cellular stress response and cancer. *Cell Mol Life Sci* 74 (9):1659–1680
- Laemmli UK (1970) Cleavage of structural proteins during the assembly of the head of bacteriophage T4. *Nat Biotechnol* 227:680–685
- Lake JA (1994) Reconstructing evolutionary trees from DNA and protein sequences: paralinear distances. *Proc Natl Acad Sci U S A* 91:1455–1459
- Lamendola DE, Duan Z, Yusuf RZ, Seiden MV (2003) Molecular description of evolving paclitaxel resistance in the SKOV-3 human ovarian carcinoma cell line. *Cancer Res* 63(9):2200–2205
- Lamond AI (1988) RNA editing and the mysterious undercover genes of trypanosomatid mitochondria. *Trends Biochem Sci* 13(8):283–284
- Lanave C, Preparata G, Saccone C, Serio G (1984) A new method for calculating evolutionary substitution rates. *J Mol Evol* 20(1):86–93
- Lander ES, Linton LM, Birren B, Nusbaum C, Zody MC, Baldwin J, Devon K, Dewar K, Doyle M, FitzHugh W et al (2001) Initial sequencing and analysis of the human genome. *Nature* 409 (6822):860–921
- Lang BF, Burger G, O'Kelly CJ, Cedergren R, Golding GB, Lemieux C, Sankoff D, Turmel M, Gray MW (1997) An ancestral mitochondrial DNA resembling a eubacterial genome in miniature. *Nature* 387(6632):493–497
- Langmead B, Salzberg SL (2012) Fast gapped-read alignment with Bowtie 2. *Nat Methods* 9 (4):357–359
- Langmead B, Schatz MC, Lin J, Pop M, Salzberg SL (2009a) Searching for SNPs with cloud computing. *Genome Biol* 10(11):R134
- Langmead B, Trapnell C, Pop M, Salzberg SL (2009b) Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. *Genome Biol* 10(3):R25
- Langmead B, Hansen KD, Leek JT (2010) Cloud-scale RNA-sequencing differential expression analysis with Myrna. *Genome Biol* 11(8):R83
- Lawrence CE, Altschul SF, Boguski MS, Liu JS, Neuwald AF, Wootton JC (1993) Detecting subtle sequence signals: a Gibbs sampling strategy for multiple alignment. *Science* 262(5131):208–214
- Lee C, Wang Q (2005) Bioinformatics analysis of alternative splicing. *Brief Bioinform* 6(1):23–33
- Leinonen R, Sugawara H, Shumway M (2011) The sequence read archive. *Nucleic Acids Res* 39 (Database):D19–D21
- Lemay DG, Hwang DH (2006) Genome-wide identification of peroxisome proliferator response elements using integrated computational genomics. *J Lipid Res* 47(7):1583–1587
- Lesk AM (2004) Introduction to protein science: architecture, function and genomics. Oxford University Press, New York
- Li CC (1976) First course in population genetics. The Boxwood Press, Pacific Grove
- Li W-H (1983) Evolution of duplicate genes and pseudogenes. Sinauer, Sunderland
- Li W-H (1997) Molecular evolution. Sinauer, Sunderland
- Li X, Chang YH (1995) Amino-terminal protein processing in *Saccharomyces cerevisiae* is an essential function that requires two distinct methionine aminopeptidases. *Proc Natl Acad Sci U S A* 92(26):12357–12361
- Li GL, Leong TY (2005) Feature selection for the prediction of translation initiation sites. *Genomics Proteomics Bioinformatics* 3(2):73–83
- Li W-H, Tanimura M (1987) The molecular clock runs more slowly in man than in apes and monkeys. *Nature* 326:93–96
- Li WH, Wu CI (1987) Rates of nucleotide substitution are evidently higher in rodents than in man. *Mol Biol Evol* 4(1):74–82
- Li WH, Gojobori T, Nei M (1981) Pseudogenes as a paradigm of neutral evolution. *Nature* 292 (5820):237–239

- Li W-H, Wolfe KH, Sourdis J, Sharp PM (1987) Reconstruction of phylogenetic trees and estimation of divergence times under nonconstant rates of evolution. *Cold Spring Harb Symp Quant Biol* 52:847–856
- Li F, Ge P, Hui WH, Atanasov I, Rogers K, Guo Q, Osato D, Falick AM, Zhou ZH, Simpson L (2009) Structure of the core editing complex (L-complex) involved in uridine insertion/deletion RNA editing in trypanosomatid mitochondria. *Proc Natl Acad Sci U S A* 106(30):12306–12310
- Liang KC, Wang X, Anastassiou D (2008) A profile-based deterministic sequential Monte Carlo algorithm for motif discovery. *Bioinformatics* 24(1):46–55
- Lieberman N, Gandin V, Svitkin YV, David M, Virgili G, Jaramillo M, Holcik M, Nagar B, Kimchi A, Sonenberg N (2015) DAP5 associates with eIF2beta and eIF4AI to promote Internal Ribosome Entry Site driven translation. *Nucleic Acids Res* 43(7):3764–3775
- Lieberman-Aiden E, van Berkum NL, Williams L, Imakaev M, Ragoczy T, Telling A, Amit I, Lajoie BR, Sabo PJ, Dorschner MO et al (2009) Comprehensive mapping of long-range interactions reveals folding principles of the human genome. *Science* 326(5950):289–293
- Liebler DC, TBDC L III., fb JRY, Publisher : c (2002) Introduction to proteomics: tools for the new biology. Humana Press, Totowa
- Liljenstrom H, von Heijne G (1987) Translation rate modification by preferential codon usage: intragenic position effects. *J Theor Biol* 124(1):43–55
- Lim VI (1994) Analysis of action of wobble nucleoside modifications on codon-anticodon pairing within the ribosome. *J Mol Biol* 240(1):8–19
- Lin JP, Aker M, Sitney KC, Mortimer RK (1986) First position wobble in codon-anticodon pairing: amber suppression by a yeast glutamine tRNA. *Gene* 49(3):383–388
- Lin HC, Tsai K, Chang BL, Liu J, Young M, Hsu W, Louie S, Nicholas HB Jr, Rosenquist GL (2003) Prediction of tyrosine sulfation sites in animal viruses. *Biochem Biophys Res Commun* 312(4):1154–1158
- Lin GN, Cai Z, Lin G, Chakraborty S, Xu D (2009) ComPhy: prokaryotic composite distance phylogenies inferred from whole-genome gene sets. *BMC Bioinform* 10(Suppl 1):S5
- Lindahl T (1993) Instability and decay of the primary structure of DNA. *Nature* 362:709–715
- Lipman DJ, Pearson WR (1985) Rapid and sensitive protein similarity searches. *Science* 227 (4693):1435–1441
- Lipman DJ, Altschul SF, Kececioglu JD (1989) A tool for multiple sequence alignment. *Proc Natl Acad Sci U S A* 86(12):4412–4415
- Lipscombe D (2005) Neuronal proteins custom designed by alternative splicing. *Curr Opin Neurobiol* 15(3):358–363
- Lithwick G, Margalit H (2005) Relative predicted protein levels of functionally associated proteins are conserved across organisms. *Nucleic Acids Res* 33(3):1051–1057
- Liu J, Louie S, Hsu W, Yu KM, Nicholas HB Jr, Rosenquist GL (2008) Tyrosine sulfation is prevalent in human chemokine receptors important in lung disease. *Am J Respir Cell Mol Biol* 38(6):738–743
- Liu X, Jiang H, Gu Z, Roberts JW (2013) High-resolution view of bacteriophage lambda gene expression by ribosome profiling. *Proc Natl Acad Sci U S A* 110(29):11928–11933
- Livesey R (2002) Have microarrays failed to deliver for developmental biology? *Genome Biol* 3(9): comment2009
- Lobry JR (1996) Asymmetric substitution patterns in the two DNA strands of bacteria. *Mol Biol Evol* 13(5):660–665
- Lockhart PJ, Steel MA, Hendy MD, Penny D (1994) Recovering evolutionary trees under a more realistic model of sequence evolution. *Mol Biol Evol* 11:605–612
- Lodish HF, Nathan DG (1972) Regulation of hemoglobin synthesis. Preferential inhibition of and globin synthesis. *J Biol Chem* 247(23):7822–7829
- Lopez P, Philippe H, Myllykallio H, Forterre P (1999) Identification of putative chromosomal origins of replication in Archaea. *Mol Microbiol* 32(4):883–886

- Lowry JA, Atchley WR (2000) Molecular evolution of the GATA family of transcription factors: conservation within the DNA-binding domain. *J Mol Evol* 50(2):103–115
- Lu C, Bablanian R (1996) Characterization of small nontranslated polyadenylated RNAs in vaccinia virus-infected cells. *Proc Natl Acad Sci U S A* 93(5):2037–2042
- Lunter G, Rocco A, Mimouni N, Heger A, Caldeira A, Hein J (2008) Uncertainty in homology inferences: assessing and improving genomic sequence alignment. *Genome Res* 18(2):298–309
- Lustig F, Boren T, Guind YS, Elias P, Samuelsson T, Gehrk CW, Kuo KC, Lagerkvist U (1989) Codon discrimination and anticodon structural context. *Proc Natl Acad Sci U S A* 86(18):6873–6877
- Ma B, Nussinov R (2004) Release factors eRF1 and RF2: a universal mechanism controls the large conformational changes. *J Biol Chem* 279(51):53875–53885
- Ma P, Xia X (2011) Factors affecting splicing strength of yeast genes. *Comp Funct Genomics*: Article ID 212146, 13 pages
- Ma S, Musa T, Bag J (2006) Reduced stability of mitogen-activated protein kinase kinase-2 mRNA and phosphorylation of poly(A)-binding protein (PABP) in cells overexpressing PABP. *J Biol Chem* 281(6):3145–3156
- MacKay VL, Li X, Flory MR, Turcott E, Law GL, Serikawa KA, Xu XL, Lee H, Goodlett DR, Aebersold R et al (2004) Gene expression analyzed by high-resolution state array analysis and quantitative proteomics: response of yeast to mating pheromone. *Mol Cell Proteomics* 3 (5):478–489
- Madden SL, Galella EA, Zhu J, Bertelsen AH, Beaudry GA (1997) SAGE transcript profiles for p53-dependent growth regulation. *Oncogene* 15(9):1079–1085
- Maher CA, Kumar-Sinha C, Cao X, Kalyana-Sundaram S, Han B, Jing X, Sam L, Barrette T, Palanisamy N, Chinnaiyan AM (2009) Transcriptome sequencing to detect gene fusions in cancer. *Nature* 458(7234):97–101
- Mannella CA, Neuwald AF, Lawrence CE (1996) Detection of likely transmembrane beta strand regions in sequences of mitochondrial pore proteins using the Gibbs sampler. *J Bioenerg Biomembr* 28(2):163–169
- Marck C, Grosjean H (2002) tRNomics: analysis of tRNA genes from 50 genomes of Eukarya, Archaea, and Bacteria reveals anticodon-sparing strategies and domain-specific features. *RNA* 8 (10):1189–1232
- Marin A, Xia X (2008) GC skew in protein-coding genes between the leading and lagging strands in bacterial genomes: new substitution models incorporating strand bias. *J Theor Biol* 253(3):508–513
- Martinez MA, Vartanian J-P, Simon W-H (1994) Hypermutation of RNA using human immunodeficiency virus type 1 reverse transcriptase and biased dNTP concentrations. *Proc Natl Acad Sci U S A* 91(25):11787–11791
- Matin A, Zychlinsky E, Keyhan M, Sachs G (1996) Capacity of *Helicobacter pylori* to generate ionic gradients at low pH is similar to that of bacteria which grow under strongly acidic conditions. *Infect Immun* 64(4):1434–1436
- McNulty DE, Claffee BA, Huddleston MJ, Porter ML, Cavnar KM, Kane JF (2003) Mistranslational errors associated with the rare arginine codon CGG in *Escherichia coli*. *Protein Expr Purif* 27(2):365–374
- McPherson DT (1988) Codon preference reflects mistranslational constraints: a proposal. *Nucleic Acids Res* 16(9):4111–4120
- Medawar PB, Medawar JS (1983) Aristotle to zoos: a philosophical dictionary of biology. Harvard University Press, Cambridge, MA
- Meinnel T, Mechulam Y, Blanquet S (1993) Methionine as translation start signal: a review of the enzymes of the pathway in *Escherichia coli*. *Biochimie* 75(12):1061–1075

- Melo EO, de Melo Neto OP, Martins de Sa C (2003a) Adenosine-rich elements present in the 5'-untranslated region of PABP mRNA can selectively reduce the abundance and translation of CAT mRNAs in vivo. *FEBS Lett* 546(2–3):329–334
- Melo EO, Dhalia R, Martins de Sa C, Standart N, de Melo Neto OP (2003b) Identification of a C-terminal poly(A)-binding protein (PABP)-PABP interaction domain: role in cooperative binding to poly (A) and efficient cap distal translational repression. *J Biol Chem* 278(47):46357–46368
- Menaker RJ, Sharaf AA, Jones NL (2004) *Helicobacter pylori* infection and gastric cancer: host, bug, environment, or all three? *Curr Gastroenterol Rep* 6(6):429–435
- Mendz GL, Hazell SL (1996) The urea cycle of *Helicobacter pylori*. *Microbiology* 142(Pt 10):2959–2967
- Meng SY, Hui JO, Haniu M, Tsai LB (1995) Analysis of translational termination of recombinant human methionyl-neurotrophin 3 in *Escherichia coli*. *Biochem Biophys Res Commun* 211(1):40–48
- Metropolis N (1987) The beginnning of the Monte Carlo method. *Los Alamos Sci* 15(Special issue):125–130
- Meyer IM, Durbin R (2004) Gene structure conservation aids similarity based gene prediction. *Nucleic Acids Res* 32(2):776–783
- Miller JH, Albertini AM (1983) Effects of surrounding sequence on the suppression of nonsense codons. *J Mol Biol* 164(1):59–71
- Miller CG, Kukral AM, Miller JL, Movva NR (1989) pepM is an essential gene in *Salmonella typhimurium*. *J Bacteriol* 171(9):5215–5217
- Milman G, Goldstein J, Scolnick E, Caskey T (1969) Peptide chain termination. 3. Stimulation of in vitro termination. *Proc Natl Acad Sci U S A* 63(1):183–190
- Min Jou W, Haegeman G, Ysebaert M, Fiers W (1972) Nucleotide sequence of the gene coding for the bacteriophage MS2 coat protein. *Nature* 237(5350):82–88
- Minakshi R, Padhan K, Rani M, Khan N, Ahmad F, Jameel S (2009) The SARS coronavirus 3a protein causes endoplasmic reticulum stress and induces ligand-independent downregulation of the type 1 interferon receptor. *PLoS One* 4(12):e8342
- Mine T, Muraoka H, Saika T, Kobayashi I (2005) Characteristics of a clinical isolate of urease-negative *Helicobacter pylori* and its ability to induce gastric ulcers in Mongolian gerbils. *Helicobacter* 10(2):125–131
- Mitra SK, Lustig F, Akesson B, Lagerkvist U (1977) Codon-anticodon recognition in the valine codon family. *J Biol Chem* 252(2):471–478
- Miura F, Kawaguchi N, Sese J, Toyoda A, Hattori M, Morishita S, Ito T (2006) A large-scale full-length cDNA analysis to explore the budding yeast transcriptome. *Proc Natl Acad Sci* 103(47):17846–17851
- Miyata T, Yasunaga T (1980) Molecular evolution of mRNA: a method for estimating evolutionary rates of synonymous and amino acid substitutions from homologous nucleotide sequences and its application. *J Mol Evol* 16(1):23–36
- Miyata T, Miyazawa S, Yasunaga T (1979) Two types of amino acid substitutions in protein evolution. *J Mol Evol* 12(3):219–236
- Mlera L, Lam J, Offerdahl DK, Martens C, Sturdevant D, Turner CV, Porcella SF, Bloom ME (2016) Transcriptome analysis reveals a signature profile for tick-borne Flavivirus persistence in HEK 293T cells. *MBio* 7(3):e00314–e00316
- Mobley HL, Hu LT, Foxal PA (1991) *Helicobacter pylori* urease: properties and role in pathogenesis. *Scand J Gastroenterol* 187(Supplement):39–46
- Moerschell RP, Hosokawa Y, Tsunasawa S, Sherman F (1990) The specificities of yeast methionine aminopeptidase and acetylation of amino-terminal methionine in vivo. Processing of altered iso-1-cytochromes c created by oligonucleotide transformation. *J Biol Chem* 265(32):19638–19643
- Moffat JG, Rudolph J, Bailey D (2014) Phenotypic screening in cancer drug discovery – past, present and future. *Nat Rev Drug Discov* 13(8):588–602

- Moi P, Loudianos G, Lavinha J, Murru S, Cossu P, Casu R, Oggiano L, Longinotti M, Cao A, Pirastu M (1992) Delta-thalassemia due to a mutation in an erythroid-specific binding protein sequence 3' to the delta-globin gene. *Blood* 79(2):512–516
- Monteiro PT, Mendes ND, Teixeira MC, d'Orey S, Tenreiro S, Mira NP, Pais H, Francisco AP, Carvalho AM, Lourenco AB et al (2008) YEASTRACT-DISCOVERER: new tools to improve the analysis of transcriptional regulatory associations in *Saccharomyces cerevisiae*. *Nucleic Acids Res* 36(Database issue):D132–D136
- Mora L, Heurgue-Hamard V, Champ S, Ehrenberg M, Kisselev LL, Buckingham RH (2003) The essential role of the invariant GGQ motif in the function and stability in vivo of bacterial release factors RF1 and RF2. *Mol Microbiol* 47(1):267–275
- Mora L, Heurgue-Hamard V, de Zamaroczy M, Kervestin S, Buckingham RH (2007) Methylation of bacterial release factors RF1 and RF2 is required for normal translation termination in vivo. *J Biol Chem* 282(49):35638–35645
- Morin R, Bainbridge M, Fejes A, Hirst M, Krzywinski M, Pugh T, McDonald H, Varhol R, Jones S, Marra M (2008a) Profiling the HeLa S3 transcriptome using randomly primed cDNA and massively parallel short-read sequencing. *BioTechniques* 45(1):81–94
- Morin RD, O'Connor MD, Griffith M, Kuchenbauer F, Delaney A, Prabhu AL, Zhao Y, McDonald H, Zeng T, Hirst M et al (2008b) Application of massively parallel sequencing to microRNA profiling and discovery in human embryonic stem cells. *Genome Res* 18(4):610–621
- Morita M, Shimozawa N, Kashiwayama Y, Suzuki Y, Imanaka T (2011) ABC subfamily D proteins and very long chain fatty acid metabolism as novel targets in adrenoleukodystrophy. *Curr Drug Targets* 12(5):694–706
- Moriyama EN, Powell JR (1997) Codon usage bias and tRNA abundance in *Drosophila*. *J Mol Evol* 45(5):514–523
- Mortazavi A, Williams BA, McCue K, Schaeffer L, Wold B (2008) Mapping and quantifying mammalian transcriptomes by RNA-Seq. *Nat Methods* 5(7):621–628
- Mottagui-Tabar S, Isaksson LA (1997) Only the last amino acids in the nascent peptide influence translation termination in *Escherichia coli* genes. *FEBS Lett* 414(1):165–170
- Moult J, Hubbard T, Fidelis K, Pedersen JT (1999) Critical assessment of methods of protein structure prediction (CASP): round III. *Proteins* 37(Suppl 3):2–6
- Muller HJ, Altenburg E (1930) The frequency of translocations produced by X-rays in *Drosophila*. *Genetics* 15(4):283–311
- Murphy J, Mahony J, Ainsworth S, Nauta A, van Sinderen D (2013) Bacteriophage orphan DNA methyltransferases: insights from their bacterial origin, function, and occurrence. *Appl Environ Microbiol* 79(24):7547–7555
- Murtagh F (1984) Complexities of hierachic clustering algorithms: state of the art. *Comput Stat Q* 1:101–113
- Muto A, Osawa S (1987) The guanine and cytosine content of genomic DNA and bacterial evolution. *Proc Natl Acad Sci U S A* 84:166–169
- Nachman MW, Crowell SL (2000) Estimate of the mutation rate per nucleotide in humans. *Genetics* 156(1):297–304
- Nakai K, Horton P (1999) PSORT: a program for detecting sorting signals in proteins and predicting their subcellular localization. *Trends Biochem Sci* 24(1):34–36
- Nakamoto T (2006) A unified view of the initiation of protein synthesis. *Biochem Biophys Res Commun* 341(3):675–678
- Nakamura Y, Ito K, Matsumura K, Kawazu Y, Ebihara K (1995) Regulation of translation termination: conserved structural motifs in bacterial and eukaryotic polypeptide release factors. *Biochem Cell Biol* 73(11–12):1113–1122
- Nakamura Y, Ito K, Isaksson LA (1996) Emerging understanding of translation termination. *Cell* 87 (2):147–150
- Nakamura Y, Gojobori T, Ikemura T (2000) Codon usage tabulated from international DNA sequence databases: status for the year 2000. *Nucleic Acids Res* 28(1):292

- Nakashima H, Fukuchi S, Nishikawa K (2003) Compositional changes in RNA, DNA and proteins for bacterial adaptation to higher and lower temperatures. *J Biochem (Tokyo)* 133(4):507–513
- Nasvall SJ, Chen P, Bjork GR (2007) The wobble hypothesis revisited: uridine-5-oxyacetic acid is critical for reading of G-ending codons. *RNA* 13(12):2151–2164
- Needleman SB, Wunsch CD (1970) A general method applicable to the search of similarities in the amino acid sequence of two proteins. *J Mol Biol* 48:443–453
- Nei M (1996) Phylogenetic analysis in molecular evolutionary genetics. *Annu Rev Genet* 30:371–403
- Nei M, Kumar S (2000) Molecular evolution and phylogenetics. Oxford University Press, New York
- Neuwald AF, Liu JS, Lawrence CE (1995) Gibbs motif sampling: detection of bacterial outer membrane protein repeats. *Protein Sci* 4(8):1618–1632
- Ngumbela KC, Ryan KP, Sivamurthy R, Brockman MA, Gandhi RT, Bhardwaj N, Kavanagh DG (2008) Quantitative effect of suboptimal codon usage on translational efficiency of mRNA encoding HIV-1 *gag* in intact T cells. *PLoS One* 3(6):e2356
- Nicholas HB Jr, Chan SS, Rosenquist GL (1999) Reevaluation of the determinants of tyrosine sulfation. *Endocrine* 11(3):285–292
- Nichols T, Hayasaka S (2003) Controlling the familywise error rate in functional neuroimaging: a comparative review. *Stat Meth Med Res* 12(5):419–446
- Nicolae M, Pathak S, Rajasekaran S (2015) LFQC: a lossless compression algorithm for FASTQ files. *Bioinformatics* 31(20):3276–3281
- Nishimura S, Takahashi S, Kuroha T, Suwabe N, Nagasawa T, Trainor C, Yamamoto M (2000) A GATA box in the GATA-1 gene hematopoietic enhancer is a critical element in the network of GATA factors and sites that regulate this gene. *Mol Cell Biol* 20(2):713–723
- Nissen P, Kjeldgaard M, Thirup S, Polekhina G, Reshetnikova L, Clark BF, Nyborg J (1995) Crystal structure of the ternary complex of Phe-tRNAPhe, EF-Tu, and a GTP analog. *Science* 270(5241):1464–1472
- Noedl H, Se Y, Schaecher K, Smith BL, Socheat D, Fukuda MM (2008) Evidence of artemisinin-resistant malaria in western Cambodia. *N Engl J Med* 359(24):2619–2620
- Noedl H, Socheat D, Satimai W (2009) Artemisinin-resistant malaria in Asia. *N Engl J Med* 361 (5):540–541
- Noedl H, Se Y, Sriwichai S, Schaecher K, Teja-Isavadharm P, Smith B, Rutvisuttinunt W, Bethell D, Surasri S, Fukuda MM et al (2010) Artemisinin resistance in Cambodia: a clinical trial designed to address an emerging problem in Southeast Asia. *Clin Infect Dis* 51(11):e82–e89
- Nomenclature Committee of the International Union of Biochemistry (1985) Nomenclature for incompletely specified bases in nucleic acid sequences. Recommendations 1984. *Eur J Biochem* 150:1–5
- Notredame C, O'Brien EA, Higgins DG (1997) RAGA: RNA sequence alignment by genetic algorithm. *Nucleic Acids Res* 25(22):4570–4580
- Numanagic I, Bonfield JK, Hach F, Voges J, Ostermann J, Alberti C, Mattavelli M, Sahinalp SC (2016) Comparison of high-throughput sequencing data compression tools. *Nat Methods* 13 (12):1005–1008
- Nur I, Szfy M, Razin A, Glaser G, Rottem S, Razin S (1985) Prokaryotic and eucaryotic traits of DNA methylation in spiroplasmas (mycoplasmas). *J Bacteriol* 164(1):19–24
- Nussinov R (1984) Doublet frequencies in evolutionary distinct groups. *Nucleic Acids Res* 12 (3):1749–1763
- O'Brien JD, She ZS, Suchard MA (2008) Dating the time of viral subtype divergence. *BMC Evol Biol* 8:172
- Obenauer JC, Cantley LC, Yaffe MB (2003) Scansite 2.0: proteome-wide prediction of cell signaling interactions using short sequence motifs. *Nucleic Acids Res* 31(13):3635–3641
- Ohta T, Gray TA, Rogan PK, Buiting K, Gabriel JM, Saitoh S, Muralidhar B, Bilienska B, Krajewska-Walasek M, Driscoll DJ et al (1999) Imprinting-mutation mechanisms in Prader-Willi syndrome. *Am J Hum Genet* 64(2):397–413

- Ordway JM, Fenster SD, Ruan H, Curran T (2005) A transcriptome map of cellular transformation by the fos oncogene. *Mol Cancer* 4(1):19
- Orkin SH (1990) Globin gene regulation and switching: circa 1990. *Cell* 63(4):665–672
- Orkin SH (1992) GATA-binding transcription factors in hematopoietic cells. *Blood* 80(3):575–581
- Osawa S, Jukes TH, Muto A, Yamao F, Ohama T, Andachi Y (1987) Role of directional mutation pressure in the evolution of the eubacterial genetic code. *Cold Spring Harb Symp Quant Biol* 52:777–789
- Osterman IA, Evfratov SA, Sergiev PV, Dontsova OA (2013) Comparison of mRNA features affecting translation initiation and reinitiation. *Nucleic Acids Res* 41(1):474–486
- Ostrin EJ, Li Y, Hoffman K, Liu J, Wang K, Zhang L, Mardon G, Chen R (2006) Genome-wide identification of direct targets of the Drosophila retinal determination protein Eyeless. *Genome Res* 16(4):466–476
- Ota S, Li WH (2000) NJML: a hybrid algorithm for the neighbor-joining and maximum-likelihood methods. *Mol Biol Evol* 17(9):1401–1409
- Ota S, Li WH (2001) NJML+: an extension of the NJML method to handle protein sequence data and computer software implementation. *Mol Biol Evol* 18(11):1983–1992
- Otu HH, Sayood K (2003) A new sequence distance measure for phylogenetic tree construction. *Bioinformatics* 19(16):2122–2130
- Palidwor GA, Perkins TJ, Xia X (2010) A general model of codon bias due to GC mutational bias. *PLoS One* 5(10):e13431
- Palstra RJ, Tolhuis B, Splinter E, Nijmeijer R, Grosveld F, de Laat W (2003) The beta-globin nuclear compartment in development and erythroid differentiation. *Nat Genet* 35(2):190–194
- Pandey RR, Mondal T, Mohammad F, Enroth S, Redrup L, Komorowski J, Nagano T, Mancini-Dinardo D, Kanduri C (2008) Kcnq1ot1 antisense noncoding RNA mediates lineage-specific transcriptional silencing through chromatin-level regulation. *Mol Cell* 32(2):232–246
- Pappin DJ, Hojrup P, Bleasby AJ (1993) Rapid identification of proteins by peptide-mass fingerprinting. *Curr Biol* 3(6):327–332
- Park SY, Cromie MJ, Lee EJ, Groisman EA (2010) A bacterial mRNA leader that employs different mechanisms to sense disparate intracellular signals. *Cell* 142(5):737–748
- Parker J (1989) Errors and alternatives in reading the universal genetic code. *Microbiol Rev* 53 (3):273–298
- Patel GP, Bag J (2006) IMP1 interacts with poly(A)-binding protein (PABP) and the autoregulatory translational control element of PABP-mRNA through the KH III-IV domain. *FEBS J* 273 (24):5678–5690
- Patel GP, Ma S, Bag J (2005) The autoregulatory translational control element of poly(A)-binding protein mRNA forms a heteromeric ribonucleoprotein complex. *Nucleic Acids Res* 33 (22):7074–7089
- Pauling L, Itano HA, Singer SJ, Wells IC (1949) Sickle cell anemia a molecular disease. *Science* 110(2865):543–548
- Pazin MJ, Kamakaka RT, Kadonaga JT (1994) ATP-dependent nucleosome reconfiguration and transcriptional activation from preassembled chromatin templates. *Science* 266(5193):2007–2011
- Pazin MJ, Sheridan PL, Cannon K, Cao Z, Keck JG, Kadonaga JT, Jones KA (1996) NF-kappa B-mediated chromatin reconfiguration and transcriptional activation of the HIV-1 enhancer in vitro. *Genes Dev* 10(1):37–49
- Pazin MJ, Hermann JW, Kadonaga JT (1998) Promoter structure and transcriptional activation with chromatin templates assembled in vitro. A single Gal4-VP16 dimer binds to chromatin or to DNA with comparable affinity. *J Biol Chem* 273(51):34653–34660
- Peabody MA, Laird MR, Vlasschaert C, Lo R, Brinkman FS (2016) PSORTdb: expanding the bacteria and archaea protein subcellular localization database to better reflect diversity in cell envelope structures. *Nucleic Acids Res* 44(D1):D663–D668

- Pearson WR (1990) Rapid and sensitive sequence comparison with FASTP and FASTA. *Methods Enzymol* 183:63–98
- Pearson WR (1994) Using the FASTA program to search protein and DNA sequence databases. *Methods Mol Biol* 24:307–331
- Pearson WR (1998) Empirical statistical estimates for sequence similarity searches. *J Mol Biol* 276 (1):71–84
- Pearson WR, Lipman DJ (1988) Improved tools for biological sequence comparison. *Proc Natl Acad Sci U S A* 85:2444–2448
- Pei J, Kim BH, Grishin NV (2008) PROMALS3D: a tool for multiple protein sequence and structure alignments. *Nucleic Acids Res* 36(7):2295–2300
- Percudani R, Pavesi A, Ottonello S (1997) Transfer RNA gene redundancy and translational selection in *Saccharomyces cerevisiae*. *J Mol Biol* 268(2):322–330
- Pereira SL, Baker AJ (2006) A mitogenomic timescale for birds detects variable phylogenetic rates of molecular evolution and refutes the standard molecular clock. *Mol Biol Evol* 23(9):1731–1740
- Pestova TV, Shatsky IN, Fletcher SP, Jackson RJ, Hellen CU (1998) A prokaryotic-like mode of cytoplasmic eukaryotic ribosome binding to the initiation codon during internal translation initiation of hepatitis C and classical swine fever virus RNAs. *Genes Dev* 12(1):67–83
- Pestova TV, Lomakin IB, Hellen CU (2004) Position of the CrPV IRES on the 40S subunit and factor dependence of IRES/80S ribosome assembly. *EMBO Rep* 5(9):906–913
- Petronis A (2004) The origin of schizophrenia: genetic thesis, epigenetic antithesis, and resolving synthesis. *Biol Psychiatry* 55(10):965–970
- Petronis A (2006) Epigenetics and twins: three variations on the theme. *Trends Genet* 22(7):347–350
- Petronis A, Gottesman II, Kan P, Kennedy JL, Basile VS, Paterson AD, Popendikyte V (2003) Monozygotic twins exhibit numerous epigenetic differences: clues to twin discordance? *Schizophr Bull* 29(1):169–178
- Petrullo LA, Gallagher PJ, Elseviers D (1983) The role of 2-methylthio-N6-isopentenyladenosine in readthrough and suppression of nonsense codons in *Escherichia coli*. *Mol Gen Genet* 190 (2):289–294
- Petry S, Brodersen DE, FVt M, Dunham CM, Selmer M, Tarry MJ, Kelley AC, Ramakrishnan V (2005) Crystal structures of the ribosome in complex with release factors RF1 and RF2 bound to a cognate stop codon. *Cell* 123(7):1255–1266
- Pevzner PA (2000) Computational molecular biology: an algorithmic approach. The MIT Press, Cambridge, MA
- Pielou EC (1984) The interpretation of ecological data: a primer on classification and ordination. Wiley, New York
- Pietras K, Sjöblom T, Rubin K, Heldin CH, Ostman A (2003) PDGF receptors as cancer drug targets. *Cancer Cell* 3(5):439–443
- Pinheiro JC, Bates DM (2000) Mixed-effects models in S and S-PLUS. Springer, Berlin/Heidelberg
- Pleiss JA, Whitworth GB, Bergkessel M, Guthrie C (2007) Rapid, transcript-specific changes in splicing in response to environmental stress. *Mol Cell* 27(6):928–937
- Pobre V, Arraiano CM (2015) Next generation sequencing analysis reveals that the ribonucleases RNase II, RNase R and PNPase affect bacterial motility and biofilm formation in *E. coli*. *BMC Genomics* 16:72
- Poole ES, Brown CM, Tate WP (1995) The identity of the base following the stop codon determines the efficiency of in vivo translational termination in *Escherichia coli*. *EMBO J* 14(1):151–158
- Poole ES, Major LL, Mannering SA, Tate WP (1998) Translational termination in *Escherichia coli*: three bases following the stop codon crosslink to release factor 2 and affect the decoding efficiency of UGA-containing signals. *Nucleic Acids Res* 26(4):954–960
- Popa A, Lebrigand K, Barbry P, Waldmann R (2016) Pateamine A-sensitive ribosome profiling reveals the scope of translation in mouse embryonic stem cells. *BMC Genomics* 17:52

- Poulos MG, Batra R, Charizanis K, Swanson MS (2011) Developments in RNA splicing and disease. *Cold Spring Harb Perspect Biol* 3(1):a000778
- Povolotskaya IS, Kondrashov FA, Ledda A, Vlasov PK (2012) Stop codons in bacteria are not selectively equivalent. *Biol Direct* 7:30
- Prabhakaran R, Chithambaram S, Xia X (2015) Escherichia coli and Staphylococcus phages: effect of translation initiation efficiency on differential codon adaptation mediated by virulent and temperate lifestyles. *J Gen Virol* 96(Pt 5):1169–1179
- Prensner JR, Iyer MK, Balbin OA, Dhanasekaran SM, Cao Q, Brenner JC, Laxman B, Asangani IA, Grasso CS, Kominsky HD et al (2011) Transcriptome sequencing across a prostate cancer cohort identifies PCAT-1, an unannotated lincRNA implicated in disease progression. *Nat Biotechnol* 29(8):742–749
- Press WH, Teukolsky SA, Titterling WT, Flannery BP (1992) Numerical recipes in C: the art of scientific computing. Cambridge University Press, Cambridge
- Prival MJ (1996) Isolation of glutamate-inserting ochre suppressor mutants of *Salmonella typhimurium* and *Escherichia coli*. *J Bacteriol* 178(10):2989–2990
- Ptashne M (1986) A genetic switch: gene control and phage lambda. Cell Press and Blackwell Scientific, Cambridge, MA
- Pure GA, Robinson GW, Naumovski L, Friedberg EC (1985) Partial suppression of an ochre mutation in *Saccharomyces cerevisiae* by multicopy plasmids containing a normal yeast tRNAGln gene. *J Mol Biol* 183(1):31–42
- Pyronnet S, Pradayrol L, Sonenberg N (2000) A cell cycle-dependent internal ribosome entry site. *Mol Cell* 5(4):607–616
- Qin ZS, McCue LA, Thompson W, Mayerhofer L, Lawrence CE, Liu JS (2003) Identification of co-regulated genes through Bayesian clustering of predicted regulatory binding sites. *Nat Biotechnol* 21(4):435–439
- Qu K, McCue LA, Lawrence CE (1998) Bayesian protein family classifier. *Proc Int Conf Intell Syst Mol Biol* 6:131–139
- Raaum RL, Sterner KN, Noviello CM, Stewart C-B, Disotell TR (2005) Catarrhine primate divergence dates estimated from complete mitochondrial genomes: concordance with fossil and nuclear DNA evidence. *J Hum Evol* 48(3):237
- Rabiner LR (1989) A tutorial on hidden Markov models and selected applications in speech recognition. *Proc IEEE* 77(2):257–286
- Rahi SJ, Pecani K, Ondracka A, Oikonomou C, Cross FR (2016) The CDK-APC/C oscillator predominantly entrains periodic cell-cycle transcription. *Cell* 165(2):475–487
- Rambaut A, Bromham L (1998) Estimating divergence dates from molecular sequences. *Mol Biol Evol* 15(4):442–448
- Ran W, Higgs PG (2012) Contributions of speed and accuracy to translational selection in bacteria. *PLoS One* 7(12):e51652
- Rannala B, Yang Z (2007) Inferring speciation times under an episodic molecular clock. *Syst Biol* 56(3):453–466
- Rashid M, Saha S, Raghava GP (2007) Support Vector Machine-based method for predicting subcellular localization of mycobacterial proteins using evolutionary information and motifs. *BMC Bioinformatics* 8:337
- Razin A, Razin S (1980) Methylated bases in mycoplasmal DNA. *Nucleic Acids Res* 8(6):1383–1390
- Regier JC, Shultz JW, Zwick A, Hussey A, Ball B, Wetzer R, Martin JW, Cunningham CW (2010) Arthropod relationships revealed by phylogenomic analysis of nuclear protein-coding sequences. *Nature* 463(7284):1079–1083
- Reinert K, Stoye J, Will T (2000) An iterative method for faster sum-of-pairs multiple sequence alignment. *Bioinformatics* 16(9):808–814
- Rektorschek M, Buhmann A, Weeks D, Schwan D, Bensch KW, Eskandari S, Scott D, Sachs G, Melchers K (2000) Acid resistance of *Helicobacter pylori* depends on the UreI membrane protein and an inner membrane proton barrier. *Mol Microbiol* 36(1):141–152

- Rice P, Longden I, Bleasby A (2000) EMBOSS: the European molecular biology open software suite. *Trends Genet* 16(6):276–277
- Rideout WMI, Coetzee GA, Olumi AF, Jones PA (1990) 5-Methylcytosine as an endogenous mutagen in the human LDL receptor and p53 genes. *Science* 249:1288–1290
- Rimsky L, Hauber J, Dukovich M, Malim MH, Langlois A, Cullen BR, Greene WC (1988) Functional replacement of the HIV-1 rev protein by the HTLV-1 rex protein. *Nature* 335 (6192):738–740
- Rinn JL, Kertesz M, Wang JK, Squazzo SL, Xu X, Brugmann SA, Goodnough LH, Helms JA, Farnham PJ, Segal E et al (2007) Functional demarcation of active and silent chromatin domains in human HOX loci by noncoding RNAs. *Cell* 129(7):1311–1323
- Ritland K, Clegg M (1990) Optimal DNA sequence divergence for testing phylogenetic hypotheses. In: Molecular evolution. Alan R. Liss, New York, pp 289–296
- Roberts A, Pachter L (2013) Streaming fragment assignment for real-time analysis of sequencing experiments. *Nat Methods* 10(1):71–73
- Roberts A, Trapnell C, Donaghey J, Rinn JL, Pachter L (2011) Improving RNA-Seq expression estimates by correcting for fragment bias. *Genome Biol* 12(3):R22
- Roberts A, Feng H, Pachter L (2013a) Fragment assignment in the cloud with eXpress-D. *BMC Bioinform* 14:358
- Roberts A, Schaeffer L, Pachter L (2013b) Updating RNA-Seq analyses after re-annotation. *Bioinformatics* 29(13):1631–1637
- Robertson G, Hirst M, Bainbridge M, Bilenky M, Zhao Y, Zeng T, Euskirchen G, Bernier B, Varhol R, Delaney A et al (2007) Genome-wide profiles of STAT1 DNA association using chromatin immunoprecipitation and massively parallel sequencing. *Nat Methods* 4(8):651–657
- Robinson M, Lilley R, Little S, Emtage JS, Yarranton G, Stephens P, Millican A, Eaton M, Humphreys G (1984) Codon usage can affect efficiency of translation of genes in *Escherichia coli*. *Nucleic Acids Res* 12(17):6663–6671
- Rodgers AB, Morgan CP, Leu NA, Bale TL (2015) Transgenerational epigenetic programming via sperm microRNA recapitulates effects of paternal stress. *Proc Natl Acad Sci U S A* 112 (44):13699–13704
- Rogozin IB, Managadze D, Shabalina SA, Koonin EV (2014) Gene family level comparative analysis of gene expression in mammals validates the ortholog conjecture. *Genome Biol Evol* 6 (4):754–762
- Rosenberg MS, Kumar S (2003) Heterogeneity of nucleotide frequencies among evolutionary lineages and phylogenetic inference. *Mol Biol Evol* 20(4):610–621
- Rosenblatt F (1958) The perceptron: a probabilistic model for information storage and organization in the brain. *Psychol Rev* 65(6):386–408
- Ross S, Giglione C, Pierre M, Espagne C, Meinnel T (2005) Functional and developmental impact of cytosolic protein N-terminal methionine excision in *Arabidopsis*. *Plant Physiol* 137(2):623–637
- Roth JR (1970) UGA nonsense mutations in *Salmonella typhimurium*. *J Bacteriol* 102(2):467–475
- Rouchka EC (1997) A brief overview of Gibbs Sampling. IBC Statistics Study Group, Washington University, Institute for Biomedical Computing
- Ruiz LM, Armengol G, Habeych E, Orduz S (2006) A theoretical analysis of codon adaptation index of the *Boophilus microplus* bm86 gene directed to the optimization of a DNA vaccine. *J Theor Biol* 239(4):445–449
- Ryan MJ, Fox JH, Wilczynski W, Rand AS (1990) Sexual selection for sensory exploitation in the frog *Physalaemus pustulosus*. *Nature* 343:66–67
- Ryden SM, Isaksson LA (1984) A temperature-sensitive mutant of *Escherichia coli* that shows enhanced misreading of UAG/A and increased efficiency for some tRNA nonsense suppressors. *Mol Gen Genet* 193(1):38–45

- Rzhetsky A, Nei M (1994) Unbiased estimates of the number of nucleotide substitutions when substitution rate varies among different sites. *J Mol Evol* 38(3):295–299
- Rzhetsky A, Nei M (1995) Tests of applicability of several substitution models for DNA sequence data. *Mol Biol Evol* 12(1):131–151
- Saadatpour A, Lai S, Guo G, Yuan GC (2015) Single-cell analysis in cancer genomics. *Trends Genet* 31(10):576–586
- Sachs AB, Davis RW, Kornberg RD (1987) A single domain of yeast poly(A)-binding protein is necessary and sufficient for RNA binding and cell viability. *Mol Cell Biol* 7(9):3268–3276
- Sachs G, Meyer-Rosberg K, Scott DR, Melchers K (1996) Acid, protons and *Helicobacter pylori*. *Yale J Biol Med* 69(3):301–316
- Sachs G, Weeks DL, Melchers K, Scott DR (2003) The gastric biology of *Helicobacter pylori*. *Annu Rev Physiol* 65(1):349–369
- Saha S, Sparks AB, Rago C, Akmaev V, Wang CJ, Vogelstein B, Kinzler KW, Velculescu VE (2002) Using the transcriptome to annotate the genome. *Nat Biotechnol* 20(5):508–512
- Saitou N, Nei M (1987) The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Mol Biol Evol* 4:406–425
- Sakaluk SK (2000) Sensory exploitation as an evolutionary origin to nuptial food gifts in insects. *Proc Biol Sci* 267(1441):339–343
- Salzberg SL, Delcher AL, Kasif S, White O (1998) Microbial gene identification using interpolated Markov models. *Nucleic Acids Res* 26(2):544–548
- Sambrook JF, Fan DP, Brenner S (1967) A strong suppressor specific for UGA. *Nature* 214 (5087):452–453
- Samso M, Palumbo MJ, Radermacher M, Liu JS, Lawrence CE (2002) A Bayesian method for classification of images from electron micrographs. *J Struct Biol* 138(3):157–170
- Sancar A, Sancar GB (1988) DNA repair enzymes. *Annu Rev Biochem* 57:29–67
- Sanderson MJ (1997) A nonparametric approach to estimating divergence times in the absence of rate constancy. *Mol Biol Evol* 14:1218–1232
- Sankoff D (1975) Minimal mutation trees of sequences. *J SIAM Appl Math* 28:35–42
- Sankoff D, Morel C, Cedergren RJ (1973) Evolution of 5S RNA and the non-randomness of base replacement. *Nat New Biol* 245(147):232–234
- Sankoff D, Cedergren RJ, Lapalme G (1976) Frequency of insertion-deletion, transversion, and transition in the evolution of 5S ribosomal RNA. *J Mol Evol* 7(2):133–149
- Sawa T, Ohno-Machado L (2003) A neural network-based similarity index for clustering DNA microarray data. *Comput Biol Med* 33(1):1–15
- Schena M (1996) Genome analysis with gene expression microarrays. *BioEssays* 18(5):427–431
- Schena M (2003) Microarray analysis. Wiley-Liss, New York
- Schena M, Shalon D, Davis RW, Brown PO (1995) Quantitative monitoring of gene expression patterns with a complementary DNA microarray. *Science* 270(5235):467–470
- Schena M, Heller RA, Theriault TP, Konrad K, Lachenmeier E, Davis RW (1998) Microarrays: biotechnology's discovery platform for functional genomics [see comments]. *Trends Biotechnol* 16(7):301–306
- Schmucker D, Clemens JC, Shu H, Worby CA, Xiao J, Muda M, Dixon JE, Zipursky SL (2000) *Drosophila Dscam* is an axon guidance receptor exhibiting extraordinary molecular diversity. *Cell* 101(6):671–684
- Schneider TD, Stephens RM (1990) Sequence logos: a new way to display consensus sequences. *Nucleic Acids Res* 18(20):6097–6100
- Schuler M, Connell SR, Lescoute A, Giesebecke J, Dabrowski M, Schroeer B, Mielke T, Penczek PA, Westhof E, Spahn CM (2006) Structure of the ribosome-bound cricket paralysis virus IRES RNA. *Nat Struct Mol Biol* 13(12):1092–1096

- Schwartz S, Silva J, Burstein D, Pupko T, Eyras E, Ast G (2008) Large-scale comparative analysis of splicing signals and their corresponding splicing factors in eukaryotes. *Genome Res* 18(1):88–103
- Schwarz G (1978) Estimating the dimension of a model. *Ann Stat* 6(2):461–464
- Schwer B, Stunnenberg HG (1988) Vaccinia virus late transcripts generated in vitro have a poly(A) head. *EMBO J* 7(4):1183–1190
- Schwer B, Visca P, Vos JC, Stunnenberg HG (1987) Discontinuous transcription or RNA processing of vaccinia virus late messengers results in a 5' poly(A) leader. *Cell* 50(2):163–169
- Scolnick EM, Caskey CT (1969) Peptide chain termination. V. The role of release factors in mRNA terminator codon recognition. *Proc Natl Acad Sci U S A* 64(4):1235–1241
- Scolnick E, Tompkins R, Caskey T, Nirenberg M (1968) Release factors differing in specificity for terminator codons. *Proc Natl Acad Sci U S A* 61(2):768–774
- Scott D, Weeks D, Melchers K, Sachs G (1998) The life and death of *Helicobacter pylori*. *Gut* 43(Suppl 1):S56–S60
- Scott DR, Marcus EA, Weeks DL, Sachs G (2002) Mechanisms of acid resistance due to the urease system of *Helicobacter pylori*. *Gastroenterology* 123(1):187–195
- Seetharam R, Heeren RA, Wong EY, Braford SR, Klein BK, Aykent S, Kotts CE, Mathis KJ, Bishop BF, Jennings MJ et al (1988) Mistranslation in IGF-1 during over-expression of the protein in Escherichia coli using a synthetic gene containing low frequency codons. *Biochem Biophys Res Commun* 155(1):518–523
- Segurel L, Bon C (2017) On the evolution of lactase persistence in humans. *Annu Rev Genomics Hum Genet* 18:297–319
- Sendler E, Johnson GD, Mao S, Goodrich RJ, Diamond MP, Hauser R, Krawetz SA (2013) Stability, delivery and functions of human sperm RNAs at fertilization. *Nucleic Acids Res* 41(7):4104–4117
- Seo EY, Namkung JH, Lee KM, Lee WH, Im M, Kee SH, Tae Park G, Yang JM, Seo YJ, Park JK et al (2005) Analysis of calcium-inducible genes in keratinocytes using suppression subtractive hybridization and cDNA microarray. *Genomics* 86(5):528–538
- Serero A, Giglione C, Sardini A, Martinez-Sanz J, Meinnel T (2003) An unusual peptide deformylase features in the human mitochondrial N-terminal methionine excision pathway. *J Biol Chem* 278(52):52953–52963
- Shadel GS, Clayton DA (1997) Mitochondrial DNA maintenance in vertebrates. *Annu Rev Biochem* 66:409–435
- Sharma U, Conine CC, Shea JM, Boskovic A, Derr AG, Bing XY, Belleannee C, Kucukural A, Serra RW, Sun F et al (2016) Biogenesis and function of tRNA fragments during sperm maturation and fertilization in mammals. *Science* 351(6271):391–396
- Sharp PM (1986) What can AIDS virus codon usage tell us? *Nature* 324(6093):114
- Sharp PM, Bulmer M (1988) Selective differences among translation termination codons. *Gene* 63(1):141–145
- Sharp PM, Li WH (1987) The codon adaptation index – a measure of directional synonymous codon usage bias, and its potential applications. *Nucleic Acids Res* 15(3):1281–1295
- Sharp PM, Tuohy TM, Mosurski KR (1986) Codon usage in yeast: cluster analysis clearly differentiates highly and lowly expressed genes. *Nucleic Acids Res* 14(13):5125–5143
- Sheppard K, Yuan J, Hohn MJ, Jester B, Devine KM, Soll D (2008) From one amino acid to another: tRNA-dependent amino acid biosynthesis. *Nucleic Acids Res* 36(6):1813–1825
- Sheridan PL, Sheline CT, Cannon K, Voz ML, Pazin MJ, Kadonaga JT, Jones KA (1995) Activation of the HIV-1 enhancer by the LEF-1 HMG protein on nucleosome-assembled DNA in vitro. *Genes Dev* 9(17):2090–2104
- Sheth N, Roca X, Hastings ML, Roeder T, Krainer AR, Sachidanandam R (2006) Comprehensive splice-site analysis using comparative genomics. *Nucl Acids Res* 34(14):3955–3967
- Shimodaira H, Hasegawa M (1999) Multiple comparisons of log-likelihoods with applications to phylogenetic inference. *Mol Biol Evol* 16(8):1114–1116

- Shine J, Dalgarno L (1974a) The 3'-terminal sequence of *Escherichia coli* 16S ribosomal RNA: complementarity to nonsense triplets and ribosome binding sites. *Proc Natl Acad Sci U S A* 71(4):1342–1346
- Shine J, Dalgarno L (1974b) Identical 3'-terminal octanucleotide sequence in 18S ribosomal ribonucleic acid from different eukaryotes. A proposed role for this sequence in the recognition of terminator codons. *Biochem J* 141(3):609–615
- Shine J, Dalgarno L (1975) Determinant of cistron specificity in bacterial ribosomes. *Nature* 254(5495):34–38
- Shirokikh NE, Spirin AS (2008) Poly(A) leader of eukaryotic mRNA bypasses the dependence of translation on initiation factors. *Proc Natl Acad Sci U S A* 105(31):10738–10743
- Shoemaker RH (2006) The NCI60 human tumour cell line anticancer drug screen. *Nat Rev Cancer* 6(10):813–823
- Shoemaker DD, Schadt EE, Armour CD, He YD, Garrett-Engele P, McDonagh PD, Loerch PM, Leonardson A, Lum PY, Cavet G et al (2001) Experimental annotation of the human genome using microarray technology. *Nature* 409(6822):922–927
- Shoemaker R, Deng J, Wang W, Zhang K (2010) Allele-specific methylation is prevalent and is contributed by CpG-SNPs in the human genome. *Genome Res* 20(7):883–889
- Shpaer EG (1986) Constraints on codon context in *Escherichia coli* genes. Their possible role in modulating the efficiency of translation. *J Mol Biol* 188(4):555–564
- Siavoshi F, Malekzadeh R, Daneshmand M, Smoot DT, Ashktorab H (2004) Association between *Helicobacter pylori* infection in gastric cancer, ulcers and gastritis in Iranian patients. *Helicobacter* 9(5):470
- Siepel A, Haussler D (2004a) Combining phylogenetic and hidden Markov models in biosequence analysis. *J Comput Biol* 11(2–3):413–428
- Siepel A, Haussler D (2004b) Phylogenetic estimation of context-dependent substitution rates by maximum likelihood. *Mol Biol Evol* 21(3):468–488
- Siepel A, Haussler D (2005) Phylogenetic hidden Markov models. In: Nielsen R (ed) Statistical methods in molecular evolution. Springer, New York, pp 325–351
- Sim J, Kim SY, Lee J (2005) PPRODO: prediction of protein domain boundaries using neural networks. *Proteins* 59(3):627–632
- Simpson RM, Bruno AE, Bard JE, Buck MJ, Read LK (2016) High-throughput sequencing of partially edited trypanosome mRNAs reveals barriers to editing progression and evidence for alternative editing. *RNA* 22(5):677–695
- Sloane AJ, Duff JL, Wilson NL, Gandhi PS, Hill CJ, Hopwood FG, Smith PE, Thomas ML, Cole RA, Packer NH et al (2002) High throughput peptide mass fingerprinting and protein macroarray analysis using chemical printing strategies. *Mol Cell Proteomics* 1(7):490–499
- Smircich P, Eastman G, Bispo S, Duhagon MA, Guerra-Slompo EP, Garat B, Goldenberg S, Munroe DJ, Dallagiovanna B, Holetz F et al (2015) Ribosome profiling reveals translation control as a key mechanism generating differential gene expression in *Trypanosoma cruzi*. *BMC Genomics* 16:443
- Smit AF (1999) Interspersed repeats and other mementos of transposable elements in mammalian genomes. *Curr Opin Genet Dev* 9(6):657–663
- Smith TF, Waterman MS (1981) Identification of common molecular subsequences. *J Mol Biol* 147(1):195–197
- Smith AB, Pisani D, Mackenzie-Dodds JA, Stockley B, Webster BL, Littlewood DT (2006) Testing the molecular clock: molecular and paleontological estimates of divergence times in the Echinoidea (Echinodermata). *Mol Biol Evol* 23(10):1832–1851
- Smyth RP, Davenport MP, Mak J (2012) The origin of genetic diversity in HIV-1. *Virus Res* 169(2):415–429
- Smyth RP, Schlub TE, Grimm AJ, Waugh C, Ellenberg P, Chopra A, Mallal S, Cromer D, Mak J, Davenport MP (2014) Identifying recombination hot spots in the HIV-1 genome. *J Virol* 88(5):2891–2902

- Sneath PHA (1962) The construction of taxonomic groups. In: Ainsworth GC, Sneath PHA (eds) Microbial classification. Cambridge University Press, Cambridge, pp 289–332
- Sokal RR, Michener CD (1958) A statistical method for evaluating systematic relationships. Univ Kans Sci Bull 28:1409–1438
- Solnick JV, Hansen LM, Salama NR, Boonjakuakul JK, Syvanen M (2004) Modification of *Helicobacter pylori* outer membrane protein expression during experimental infection of rhesus macaques. Proc Natl Acad Sci U S A 101(7):2106–2111
- Sommerer N, Centeno D, Rossignol M (2006) Peptide mass fingerprinting: identification of proteins by maldi-tof. Methods Mol Biol 355:219–234
- Sonenberg N, Meerovitch K (1990) Translation of poliovirus mRNA. Enzyme 44(1–4):278–291
- Sorensen MA, Kurland CG, Pedersen S (1989) Codon usage determines translation rate in *Escherichia coli*. J Mol Biol 207:365–377
- Staden R (1984) Computer methods to locate signals in nucleic acid sequences. Nucleic Acids Res 12(1 Pt 2):505–519
- Stamm S, Ben-Ari S, Rafalska I, Tang Y, Zhang Z, Toiber D, Thanaraj TA, Soreq H (2005) Function of alternative splicing. Gene 344:1–20
- Steinberg MH, Rodgers GP (2001) Pathophysiology of sickle cell disease: role of cellular and genetic modifiers. Semin Hematol 38(4):299–306
- Steitz JA, Jakes K (1975) How ribosomes select initiator regions in mRNA: base pair formation between the 3' terminus of 16S rRNA and the mRNA during initiation of protein synthesis in *Escherichia coli*. Proc Natl Acad Sci U S A 72(12):4734–4738
- Stepankiw N, Raghavan M, Fogarty EA, Grimson A, Pleiss JA (2015) Widespread alternative and aberrant splicing revealed by lariat sequencing. Nucleic Acids Res 43(17):8488–8501
- Stingl K, Uhlemann Em EM, Deckers-Hebestreit G, Schmid R, Bakker EP, Altendorf K (2001) Prolonged survival and cytoplasmic pH homeostasis of *Helicobacter pylori* at pH 1. Infect Immun 69(2):1178–1180
- Stingl K, Altendorf K, Bakker EP (2002a) Acid survival of *Helicobacter pylori*: how does urease activity trigger cytoplasmic pH homeostasis? Trends Microbiol 10(2):70–74
- Stingl K, Uhlemann E-M, Schmid R, Altendorf K, Bakker EP (2002b) Energetics of *Helicobacter pylori* and its implications for the mechanism of urease-dependent acid tolerance at pH 1. J Bacteriol 184(11):3053–3060
- Stormo GD, Schneider TD, Gold L, Ehrenfeucht A (1982a) Use of the ‘Perceptron’ algorithm to distinguish translational initiation sites in *E. coli*. Nucleic Acids Res 10(9):2997–3011
- Stormo GD, Schneider TD, Gold LM (1982b) Characterization of translational initiation sites in *E. coli*. Nucleic Acids Res 10(9):2971–2996
- Stormo GD, Schneider TD, Gold L (1986) Quantitative analysis of the relationship between nucleotide sequence and functional activity. Nucleic Acids Res 14(16):6661–6679
- Stoye J, Moulton V, Dress AW (1997) DCA: an efficient implementation of the divide-and-conquer approach to simultaneous multiple sequence alignment. Comput Appl Biosci 13(6):625–626
- Strelbel K (2005) APOBEC3G & HTLV-1: inhibition without deamination. Retrovirology 2(1):37
- Strigini P, Brickman E (1973) Analysis of specific misreading in *Escherichia coli*. J Mol Biol 75 (4):659–672
- Su HL, Liao CL, Lin YL (2002) Japanese encephalitis virus infection initiates endoplasmic reticulum stress and an unfolded protein response. J Virol 76(9):4162–4171
- Sueoka N (1964) On the evolution of informational macromolecules. Academic, New York
- Suerbaum S, Smith JM, Bapumia K, Morelli G, Smith NH, Kunstmann E, Dyrek I, Achtman M (1998) Free recombination within *Helicobacter pylori*. Proc Natl Acad Sci U S A 95 (21):12619–12624
- Suerbaum S, Josenhans C, Sterzenbach T, Drescher B, Brandt P, Bell M, Droege M, Fartmann B, Fischer HP, Ge Z et al (2003) The complete genome sequence of the carcinogenic bacterium *Helicobacter hepaticus*. Proc Natl Acad Sci U S A 100(13):7901–7906
- Sun XY, Yang Q, Xia X (2013) An improved implementation of effective Number of Codons ( $N_c$ ). Mol Biol Evol 30:191–196

- Sund J, Ander M, Aqvist J (2010) Principles of stop-codon reading on the ribosome. *Nature* 465(7300):947–950
- Supek F, Smuc T (2010) On relevance of codon usage to expression of synthetic and natural genes in *Escherichia coli*. *Genetics* 185(3):1129–1134
- Sutton CW, Pemberton KS, Cottrell JS, Corbett JM, Wheeler CH, Dunn MJ, Pappin DJ (1995) Identification of myocardial proteins from two-dimensional gels by peptide mass fingerprinting. *Electrophoresis* 16(3):308–316
- Sved J, Bird A (1990) The expected equilibrium of the CpG dinucleotide in vertebrate genomes under a mutation model. *Proc Natl Acad Sci U S A* 87:4692–4696
- Svitkin YV, Imataka H, Khaleghpour K, Kahvejian A, Liebig HD, Sonenberg N (2001) Poly(A)-binding protein interaction with eIF4G stimulates picornavirus IRES-dependent translation. *RNA* 7(12):1743–1752
- Swofford D (1993) Phylogenetic analysis using parsimony. Illinois Natural History Survey, Champaign
- Tajima F (1993) Unbiased estimation of evolutionary distance between nucleotide sequences. *Mol Biol Evol* 10(3):677–688
- Tajima F, Nei M (1984) Estimation of evolutionary distance between nucleotide sequences. *Mol Biol Evol* 1(3):269–285
- Takezaki N, Nei M (1994) Inconsistency of the maximum parsimony method when the rate of nucleotide substitution is constant. *J Mol Evol* 39(2):210–218
- Takezaki N, Rzhetsky A, Nei M (1995) Phylogenetic test of the molecular clock and linearized trees. *Mol Biol Evol* 12(5):823–833
- Tamai I, Sai Y, Kobayashi H, Kamata M, Wakamiya T, Tsuji A (1997) Structure-internalization relationship for adsorptive-mediated endocytosis of basic peptides at the blood-brain barrier. *J Pharmacol Exp Ther* 280(1):410–415
- Tamura K, Kumar S (2002) Evolutionary distance estimation under heterogeneous substitution pattern among lineages. *Mol Biol Evol* 19(10):1727–1736
- Tamura K, Nei M (1993) Estimation of the number of nucleotide substitutions in the control region of mitochondrial DNA in humans and chimpanzees. *Mol Biol Evol* 10:512–526
- Tamura K, Nei M, Kumar S (2004) Prospects for inferring very large phylogenies by using the neighbor-joining method. *Proc Natl Acad Sci U S A* 101(30):11030–11035
- Tamura K, Dudley J, Nei M, Kumar S (2007) MEGA4: molecular evolutionary genetics analysis (MEGA) software version 4.0. *Mol Biol Evol* 24(8):1596–1599
- Tanabe M, Kanehisa M (2012) Using the KEGG database resource. *Curr Protoc Bioinformatics Chapter 1:Unit1 12*
- Tanaka M, Ozawa T (1994) Strand asymmetry in human mitochondrial DNA mutations. *Genomics* 22(2):327–335
- Tang N, Tornatore P, Weinberger SR (2004) Current developments in SELDI affinity technology. *Mass Spectrom Rev* 23(1):34–44
- Tang Y, Gao XD, Wang Y, Yuan BF, Feng YQ (2012) Widespread existence of cytosine methylation in yeast DNA measured by gas chromatography/mass spectrometry. *Anal Chem* 84(16):7249–7255
- Taniguchi T, Weissmann C (1978) Inhibition of Qbeta RNA 70S ribosome initiation complex formation by an oligonucleotide complementary to the 3' terminal region of *E. coli* 16S ribosomal RNA. *Nature* 275(5682):770–772
- Tao H, Bausch C, Richmond C, Blattner FR, Conway T (1999) Functional genomics: expression analysis of *Escherichia coli* growing on minimal and rich media. *J Bacteriol* 181(20):6425–6440
- Taramelli R, Kioussis D, Vanin E, Bartram K, Groffen J, Hurst J, Grosfeld FG (1986) Gamma delta beta-thalassaemias 1 and 2 are the result of a 100 kbp deletion in the human beta-globin cluster. *Nucleic Acids Res* 14(17):7017–7029

- Tate WP, Brown CM (1992) Translational termination: “stop” for protein synthesis or “pause” for regulation of gene expression. *Biochemistry (Mosc)* 31(9):2443–2450
- Tate WP, Mannerling SA (1996) Three, four or more: the translational stop signal at length. *Mol Microbiol* 21(2):213–219
- Tate WP, Mansell JB, Mannerling SA, Irvine JH, Major LL, Wilson DN (1999) UGA: a dual signal for ‘stop’ and for recoding in protein synthesis. *Biochemistry (Mosc)* 64(12):1342–1353
- Tavaré S (1986) Some probabilistic and statistical problems in the analysis of DNA sequences. In: Miura RM (ed) *Some mathematical questions in biology – DNA sequence analysis*. American Mathematical Society, Providence, pp 57–86
- Team GE (2011) Closure of the NCBI SRA and implications for the long-term future of genomics data storage. *Genome Biol* 12(3):402
- Tech M, Merkl R (2003) YACOP: enhanced gene prediction obtained by a combination of existing methods. *In Silico Biol* 3(4):441–451
- Terasaki T, Deguchi Y, Sato H, K-i H, Tsuji A (1991) In vivo transport of a Dynorphin-like analgesic peptide, E-2078, through the blood-brain barrier: an application of brain microdialysis. *Pharm Res* 8(7):815
- Terenin IM, Dmitriev SE, Andreev DE, Royall E, Belsham GJ, Roberts LO, Shatsky IN (2005) A cross-kingdom internal ribosome entry site reveals a simplified mode of internal ribosome entry. *Mol Cell Biol* 25(17):7879–7888
- Thijs G, Lescot M, Marchal K, Rombauts S, De Moor B, Rouze P, Moreau Y (2001) A higher-order background model improves the detection of promoter regulatory elements by Gibbs sampling. *Bioinformatics* 17(12):1113–1122
- Thijs G, Marchal K, Lescot M, Rombauts S, De Moor B, Rouze P, Moreau Y (2002a) A Gibbs sampling method to detect overrepresented motifs in the upstream regions of coexpressed genes. *J Comput Biol* 9(2):447–464
- Thijs G, Moreau Y, De Smet F, Mathys J, Lescot M, Rombauts S, Rouze P, De Moor B, Marchal K (2002b) INCLUSiVe: integrated clustering, upstream sequence retrieval and motif sampling. *Bioinformatics* 18(2):331–332
- Thompson JD, Higgins DG, Gibson TJ (1994) CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, positions-specific gap penalties and weight matrix choice. *Nucleic Acids Res* 22:4673–4680
- Thompson W, Rouchka EC, Lawrence CE (2003) Gibbs recursive sampler: finding transcription factor binding sites. *Nucleic Acids Res* 31(13):3580–3585
- Thompson W, Palumbo MJ, Wasserman WW, Liu JS, Lawrence CE (2004) Decoding human regulatory circuits. *Genome Res* 14(10A):1967–1974
- Thorne JL, Kishino H (1992) Freeing phylogenies from artifacts of alignment. *Mol Biol Evol* 9 (6):1148–1162
- Thorne JL, Kishino H (2005) Estimation of divergence times from molecular sequence data. In: Nielsen R (ed) *Statistical methods in molecular evolution*. Springer, New York, pp 233–256
- Tinn O, Oakley TH (2008) Erratic rates of molecular evolution and incongruence of fossil and molecular divergence time estimates in Ostracoda (Crustacea). *Mol Phylogenet Evol* 48(1):157–167
- Tjaden B (2015) De novo assembly of bacterial transcriptomes from RNA-seq data. *Genome Biol* 16:1
- Tolhuis B, Palstra RJ, Splinter E, Grosveld F, de Laat W (2002) Looping and interaction between hypersensitive sites in the active beta-globin locus. *Mol Cell* 10(6):1453–1465
- Tomatsu S, Orii KO, Bi Y, Gutierrez MA, Nishioka T, Yamaguchi S, Kondo N, Orii T, Noguchi A, Sly WS (2004) General implications for CpG hot spot mutations: methylation patterns of the human iduronate-2-sulfatase gene locus. *Hum Mutat* 23(6):590–598
- Tomb JF, White O, Kerlavage AR, Clayton RA, Sutton GG, Fleischmann RD, Ketchum KA, Klenk HP, Gill S, Dougherty BA et al (1997) The complete genome sequence of the gastric pathogen *Helicobacter pylori*. *Nature* 388(6642):539–547

- Toronen P, Kolehmainen M, Wong G, Castren E (1999) Analysis of gene expression data using self-organizing maps. *FEBS Lett* 451(2):142–146
- Trapnell C (2015) Defining cell types and states with single-cell genomics. *Genome Res* 25(10):1491–1498
- Trapnell C, Pachter L, Salzberg SL (2009) TopHat: discovering splice junctions with RNA-Seq. *Bioinformatics* 25(9):1105–1111
- Trapnell C, Williams BA, Pertea G, Mortazavi A, Kwan G, van Baren MJ, Salzberg SL, Wold BJ, Pachter L (2010) Transcript assembly and quantification by RNA-Seq reveals unannotated transcripts and isoform switching during cell differentiation. *Nat Biotechnol* 28(5):511–515
- Trapnell C, Roberts A, Goff L, Pertea G, Kim D, Kelley DR, Pimentel H, Salzberg SL, Rinn JL, Pachter L (2012) Differential gene and transcript expression analysis of RNA-seq experiments with TopHat and Cufflinks. *Nat Protoc* 7(3):562–578
- Trapnell C, Hendrickson DG, Sauvageau M, Goff L, Rinn JL, Pachter L (2013) Differential analysis of gene regulation at transcript resolution with RNA-seq. *Nat Biotechnol* 31(1):46–53
- Trudel MV, Vincent AT, Attere SA, Labbe M, Derome N, Culley AI, Charette SJ (2016) Diversity of antibiotic-resistance genes in Canadian isolates of *Aeromonas salmonicida* subsp. *salmonicida*: dominance of pSN254b and discovery of pAsa8. *Sci Rep* 6:35617
- Trutschl M, Dinkova TD, Rhoads RE (2005) Application of machine learning and visualization of heterogeneous datasets to uncover relationships between translation and developmental stage expression of *C. elegans* mRNAs. *Physiol Genomics* 21(2):264–273
- Tuller T, Waldman YY, Kupiec M, Ruppin E (2010) Translation efficiency is determined by both codon bias and folding energy. *Proc Natl Acad Sci U S A* 107(8):3645–3650
- Valenzuela M, Cerdá O, Toledo H (2003) Overview on chemotaxis and acid resistance in *Helicobacter pylori*. *Biol Res* 36(3–4):429–436
- Van de Peer Y, Neefs JM, De Rijk P, De Wachter R (1993) Reconstructing evolution from eukaryotic small-ribosomal-subunit RNA sequences: calibration of the molecular clock. *J Mol Evol* 37(2):221–232
- Van Dooren S, Pybus OG, Salemi M, Liu HF, Goubaud P, Remondéguí C, Talarmin A, Gotuzzo E, Alcantara LC, Galvao-Castro B et al (2004) The low evolutionary rate of human T-cell lymphotropic virus type-1 confirmed by analysis of vertical transmission chains. *Mol Biol Evol* 21(3):603–611
- Van Esch H, Devriendt K (2001) Transcription factor GATA3 and the human HDR syndrome. *Cell Mol Life Sci* 58(9):1296–1300
- van Hemert FJ, Berkhouit B (1995) The tendency of lentiviral open reading frames to become A-rich: constraints imposed by viral genome organization and cellular tRNA availability. *J Mol Evol* 41(2):132–140
- van Weringh A, Ragonnet-Cronin M, Pranckeviciene E, Pavon-Eternod M, Kleiman L, Xia X (2011) HIV-1 modulates the tRNA pool to improve translation efficiency. *Mol Biol Evol* 28(6):1827–1834
- Vartanian J-P, Henry M, Wain-Hobson S (2002) Sustained G->A hypermutation during reverse transcription of an entire human immunodeficiency virus type 1 strain Vau group O genome. *J Gen Virol* 83(4):801–805
- Vasilescu J, Figeys D (2006) Mapping protein-protein interactions by mass spectrometry. *Curr Opin Biotechnol* 17(4):394–399
- Vazquez-Pianzola P, Hernandez G, Suter B, Rivera-Pomar R (2007) Different modes of translation for hid, grim and sickle mRNAs in *Drosophila*. *Cell Death Differ* 14(2):286–295
- Velculescu VE, Zhang L, Vogelstein B, Kinzler KW (1995) Serial analysis of gene expression. *Science* 270(5235):484–487
- Velculescu VE, Zhang L, Zhou W, Vogelstein J, Basrai MA, Bassett DE Jr, Hieter P, Vogelstein B, Kinzler KW (1997) Characterization of the yeast transcriptome. *Cell* 88(2):243–251
- Velculescu VE, Madden SL, Zhang L, Lash AE, Yu J, Rago C, Lal A, Wang CJ, Beaudry GA, Ciriello KM et al (1999) Analysis of human transcriptomes. *Nat Genet* 23(4):387–388

- Velculescu VE, Vogelstein B, Kinzler KW (2000) Analysing uncharted transcriptomes with SAGE. *Trends Genet* 16(10):423–425
- Vellanoweth RL, Rabinowitz JC (1992) The influence of ribosome-binding-site elements on translational efficiency in *Bacillus subtilis* and *Escherichia coli* in vivo. *Mol Microbiol* 6(9):1105–1114
- Venter JC, Adams MD, Myers EW, Li PW, Mural RJ, Sutton GG, Smith HO, Yandell M, Evans CA, Holt RA et al (2001) The sequence of the human genome. *Science* 291(5507):1304–1351
- Vert JP (2002) Support vector machine prediction of signal peptide cleavage site using a new class of kernels for strings. *Pac Symp Biocomput* 7:649–660
- Vestergaard B, Van LB, Andersen GR, Nyborg J, Buckingham RH, Kjeldgaard M (2001) Bacterial polypeptide release factor RF2 is structurally distinct from eukaryotic eRF1. *Mol Cell* 8(6):1375–1382
- Vestergaard B, Sanyal S, Roessle M, Mora L, Buckingham RH, Kastrup JS, Gajhede M, Svergun DI, Ehrenberg M (2005) The SAXS solution structure of RF1 differs from its crystal structure and is similar to its ribosome bound cryo-EM structure. *Mol Cell* 20(6):929–938
- Viterbi AJ (1967) Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Trans Inf Theory* 13(2):260–269
- Vlasschaert C, Xia X, Coulombe J, Gray DA (2015) Evolution of the highly networked deubiquitinating enzymes USP4, USP15, and USP11. *BMC Evol Biol* 15:230
- Vlasschaert C, Xia X, Gray DA (2016) Selection preserves Ubiquitin Specific Protease 4 alternative exon skipping in therian mammals. *Sci Rep* 6:20039
- Vlasschaert C, Cook D, Xia X, Gray DA (2017) The evolution and functional diversification of the deubiquitinating enzyme superfamily. *Genome Biol Evol* 9(3):558–573
- Voelter-Mahlknecht S (2016) Epigenetic associations in relation to cardiovascular prevention and therapeutics. *Clin Epigenetics* 8:4
- Waddell PJ, Steel MA (1997a) General time-reversible distances with unequal rates across sites: mixing gamma and inverse Gaussian distributions with invariant sites. *Mol Phylogenet Evol* 8(3):398–414
- Waddell PJ, Steel MA (1997b) General time-reversible distances with unequal rates across sites: mixing lambda and inverse Gaussian distributions with invariant sites. *Mol Phylogenet Evol* 8(3):398–414
- Wade PA, Wolffe AP (2001) ReCoGnizing methylated DNA. *Nat Struct Biol* 8(7):575–577
- Walsh D, Arias C, Perez C, Halladin D, Escandon M, Ueda T, Watanabe-Fukunaga R, Fukunaga R, Mohr I (2008) Eukaryotic translation initiation factor 4F architectural alterations accompany translation initiation factor redistribution in poxvirus-infected cells. *Mol Cell Biol* 28(8):2648–2658
- Wang HC, Hickey DA (2002) Evidence for strong selective constraint acting on the nucleotide composition of 16S ribosomal RNA genes. *Nucleic Acids Res* 30(11):2501–2507
- Wang G, Humayun MZ, Taylor DE (1999) Mutation as an origin of genetic variability in *Helicobacter pylori*. *Trends Microbiol* 7(12):488–493
- Wang J, Delabie J, Aasheim H, Smeland E, Myklebost O (2002) Clustering of the SOM easily reveals distinct gene expression patterns: results of a reanalysis of lymphoma study. *BMC Bioinform* 3:36
- Wang HC, Xia X, Hickey DA (2006) Thermal adaptation of ribosomal RNA genes: a comparative study. *J Mol Evol* 63(1):120–126
- Wang Z, Gerstein M, Snyder M (2009) RNA-Seq: a revolutionary tool for transcriptomics. *Nat Rev Genet* 10(1):57–63
- Wang X, Kim Y, Ma Q, Hong SH, Pokusaeva K, Sturino JM, Wood TK (2010) Cryptic prophages help bacteria cope with adverse environments. *Nat Commun* 1:147
- Wang M, Weiss M, Simonovic M, Haertinger G, Schrimpf SP, Hengartner MO, von Mering C (2012) PaxDb, a database of protein abundance averages across all three domains of life. *Mol Cell Proteomics* 11(8):492–500

- Washburn MP, Wolters D, Yates JR 3rd (2001) Large-scale analysis of the yeast proteome by multidimensional protein identification technology. *Nat Biotechnol* 19(3):242–247
- Waterfield MD, Scrace GT, Whittle N, Stroobant P, Johnsson A, Wasteson A, Westermark B, Heldin CH, Huang JS, Deuel TF (1983) Platelet-derived growth factor is structurally related to the putative transforming protein p28<sup>sis</sup> of simian sarcoma virus. *Nature* 304(5921):35–39
- Waterman MS, Vingron M (1994) Rapid and accurate estimates of statistical significance for sequence data base searches. *Proc Natl Acad Sci U S A* 91(11):4625–4628
- Webster J, Oxley D (2005) Peptide mass fingerprinting: protein identification using MALDI-TOF mass spectrometry. *Methods Mol Biol* 310:227–240
- Weeks DL, Eskandari S, Scott DR, Sachs G (2000) A H+-gated urea channel: the link between *Helicobacter pylori* urease and gastric colonization. *Science* 287(5452):482–485
- Wei Y, Xia X (2017) The role of +4U as an extended translation termination signal in bacteria. *Genetics* 205(2):539–549
- Wei Y, Wang J, Xia X (2016) Coevolution between stop codon usage and release factors in bacterial species. *Mol Biol Evol* 33(9):2357–2367
- Wei Y, Silke JR, Xia X (2017) Elucidating the 16S rRNA 3' boundaries and defining optimal SD/aSD pairing in *Escherichia coli* and *Bacillus subtilis* using RNA-Seq data. *Sci Rep.* <https://doi.org/10.1038/s41598-017-17918-6>
- Weigert MG, Garen A (1965) Base composition of nonsense codons in *E. coli*. evidence from amino-acid substitutions at a tryptophan site in alkaline phosphatase. *Nature* 206(988):992–994
- Weiner AM, Weber K (1973) A single UGA codon functions as a natural termination signal in the coliphage q beta coat protein cistron. *J Mol Biol* 80(4):837–855
- Weir BS (1990) Genetic data analysis. Sinauer Associates, Sunderland
- Weiss RB, Dunn DM, Dahlberg AE, Atkins JF, Gesteland RF (1988) Reading frame switch caused by base-pair formation between the 3' end of 16S rRNA and the mRNA during elongation of protein synthesis in *Escherichia coli*. *EMBO J* 7(5):1503–1507
- Wen Y, Marcus EA, Matrubutham U, Gleeson MA, Scott DR, Sachs G (2003) Acid-adaptive genes of *Helicobacter pylori*. *Infect Immun* 71(10):5921–5939
- Wentzel AM, Stancek M, Isaksson LA (1998) Growth phase dependent stop codon readthrough and shift of translation reading frame in *Escherichia coli*. *FEBS Lett* 421(3):237–242
- Wilks SS (1938) The large-sample distribution of the likelihood ratio for testing composite hypotheses. *Annals Math Stat* 9:60–62
- Williams CL, Preston T, Hossack M, Slater C, McColl KE (1996) *Helicobacter pylori* utilises urea for amino acid synthesis. *FEMS Immunol Med Microbiol* 13(1):87–94
- Williams KP, Sobral BW, Dickerman AW (2007) A robust species tree for the alphaproteobacteria. *J Bacteriol* 189(13):4578–4586
- Wilson DS, Nock S (2002) Functional protein microarrays. *Curr Opin Chem Biol* 6(1):81–85
- Wilson KS, von Hippel PH (1995) Transcription termination at intrinsic terminators: the role of the RNA hairpin. *Proc Natl Acad Sci U S A* 92(19):8793–8797
- Winston F, Botstein D, Miller JH (1979) Characterization of amber and ochre suppressors in *Salmonella typhimurium*. *J Bacteriol* 137(1):433–439
- Wolfe KH, Li WH, Sharp PM (1987) Rates of nucleotide substitution vary greatly among plant mitochondrial, chloroplast and nuclear DNAs. *Proc Natl Acad Sci U S A* 84:9054–9058
- Wong KM, Suchard MA, Huelsenbeck JP (2008) Alignment uncertainty and genomic analysis. *Science* 319(5862):473–476
- Wright F (1990) The ‘effective number of codons’ used in a gene. *Gene* 87(1):23–29
- Wright GL Jr (2002) SELDI proteinchip MS: a platform for biomarker discovery and cancer diagnosis. *Expert Rev Mol Diagn* 2(6):549–563
- Wu J, Bag J (1998) Negative control of the poly(A)-binding protein mRNA translation is mediated by the adenine-rich region of its 5'-untranslated region. *J Biol Chem* 273(51):34535–34542
- Wu CI, Li WH (1985) Evidence for higher rates of nucleotide substitution in rodents than in man. *Proc Natl Acad Sci U S A* 82(6):1741–1745

- Wu J, Tzanakakis ES (2013) Deconstructing stem cell population heterogeneity: single-cell analysis and modeling approaches. *Biotechnol Adv* 31(7):1047–1062
- Xia X (1996) Maximizing transcription efficiency causes codon usage bias. *Genetics* 144:1309–1320
- Xia X (1998a) How optimized is the translational machinery in *Escherichia coli*, *Salmonella typhimurium* and *Saccharomyces cerevisiae*? *Genetics* 149(1):37–44
- Xia X (1998b) The rate heterogeneity of nonsynonymous substitutions in mammalian mitochondrial genes. *Mol Biol Evol* 15:336–344
- Xia X (2000) Phylogenetic relationship among horseshoe crab species: the effect of substitution models on phylogenetic analyses. *Syst Biol* 49:87–100
- Xia X (2001) Data analysis in molecular biology and evolution. Kluwer Academic Publishers, Boston
- Xia X (2003) DNA methylation and mycoplasma genomes. *J Mol Evol* 57:S21–S28
- Xia X (2005) Mutation and selection on the anticodon of tRNA genes in vertebrate mitochondrial genomes. *Gene* 345(1):13–20
- Xia X (2006) Topological bias in distance-based phylogenetic methods: problems with over- and underestimated genetic distances. *Evol Bioinforma* 2:375–387
- Xia X (2007a) The +4G site in Kozak consensus is not related to the efficiency of translation initiation. *PLoS One* 2:e188
- Xia X (2007b) Bioinformatics and the cell: modern computational approaches in genomics, proteomics and transcriptomics. Springer US, New York
- Xia X (2007c) An improved implementation of codon adaptation index. *Evol Bioinforma* 3:53–58
- Xia X (2008) The cost of wobble translation in fungal mitochondrial genomes: integration of two traditional hypotheses. *BMC Evol Biol* 8:211
- Xia X (2009) Information-theoretic indices and an approximate significance test for testing the molecular clock hypothesis with genetic distances. *Mol Phylogenet Evol* 52:665–676
- Xia X (2012a) DNA replication and strand asymmetry in prokaryotic and mitochondrial genomes. *Curr Genomics* 13(1):16–27
- Xia X (2012b). Position Weight Matrix, Gibbs Sampler, and the associated significance tests in motif characterization and prediction. *Scientifica* 2012: Article ID 917540, 15 pp
- Xia X (2012c) Rapid evolution of animal mitochondria. In: Singh RS, Xu J, Kulathinal RJ (eds) Evolution in the fast lane: rapidly evolving genes and genetic systems. Oxford University Press, Oxford, pp 73–82
- Xia X (2013) DAMBE5: a comprehensive software package for data analysis in molecular biology and evolution. *Mol Biol Evol* 30:1720–1728
- Xia X (2014) Phylogenetic bias in the likelihood method caused by missing data coupled with among-site rate variation: an analytical approach. In: Basu M, Pan Y, Wang J (eds) Bioinformatics research and applications. Springer, New York, pp 12–23
- Xia X (2015) A major controversy in codon-anticodon adaptation resolved by a new codon usage index. *Genetics* 199:573–579
- Xia X (2016) PhyPA: phylogenetic method with pairwise sequence alignment outperforms likelihood methods in phylogenetics involving highly diverged sequences. *Mol Phylogenet Evol* 102:331–343
- Xia X (2017a) ARSDA: a new approach for storing, transmitting and analyzing transcriptomic data. G3: Genes|Genomes|Genetics. <https://doi.org/10.1101/114470>
- Xia X (2017b) Bioinformatics and drug discovery. *Curr Top Med Chem* 17(15):1709–1726
- Xia X (2017c) DAMBE6: new tools for microbial genomics, phylogenetics and molecular evolution. *J Hered* 108(4):431–437. <https://doi.org/10.1093/jhered/esx033>
- Xia X (2017d) Self-organizing map for characterizing heterogeneous nucleotide and amino acid sequence motifs. *Computation* 5(4):43
- Xia X, Holcik M (2009) Strong eukaryotic IRESs have weak secondary structure. *PLoS One* 4(1): e4136

- Xia X, Kumar S (2006) Codon-based detection of positive selection can be biased by heterogeneous distribution of polar amino acids along protein sequences. In: Markstein P, Xu Y (eds) Computational systems bioinformatics: proceedings of the conference CSB 2006. Imperial College Press, London, pp 335–340
- Xia X, Lemey P (2009) Assessing substitution saturation with DAMBE. In: Lemey P, Salemi M, Vandamme AM (eds) The phylogenetic handbook, 2nd edn. Cambridge University Press, Cambridge, pp 615–630
- Xia X, Li WH (1998) What amino acid properties affect protein evolution? *J Mol Evol* 47(5):557–564
- Xia X, Palidwor G (2005) Genomic adaptation to acidic environment: evidence from *Helicobacter pylori*. *Am Nat* 166(6):776–784
- Xia X, Xie Z (2001a) AMADA: analysis of microarray data. *Bioinformatics* 17:569–570
- Xia X, Xie Z (2001b) DAMBE: software package for data analysis in molecular biology and evolution. *J Hered* 92(4):371–373
- Xia X, Xie Z (2002) Protein structure, neighbor effect, and a new index of amino acid dissimilarities. *Mol Biol Evol* 19(1):58–67
- Xia X, Yang Q (2011) A distance-based least-square method for dating speciation events. *Mol Phylogenet Evol* 59(2):342–353
- Xia X, Yuen KY (2005) Differential selection and mutation between dsDNA and ssDNA phages shape the evolution of their genomic AT percentage. *BMC Genet* 6(1):20
- Xia X, Hafner MS, Sudman PD (1996) On transition bias in mitochondrial genes of pocket gophers. *J Mol Evol* 43:32–40
- Xia XH, Wei T, Xie Z, Danchin A (2002) Genomic changes in nucleotide and dinucleotide frequencies in *Pasteurella multocida* cultured under high temperature. *Genetics* 161(4):1385–1394
- Xia X, Xie Z, Kjer KM (2003a) 18S ribosomal RNA and tetrapod phylogeny. *Syst Biol* 52(3):283–295
- Xia X, Xie Z, Salemi M, Chen L, Wang Y (2003b) An index of substitution saturation and its application. *Mol Phylogenet Evol* 26(1):1–7
- Xia X, Wang H, Xie Z, Carullo M, Huang H, Hickey D (2006) Cytosine usage modulates the correlation between CDS length and CG content in prokaryotic genomes. *Mol Biol Evol* 23 (7):1450–1454
- Xia X, Huang H, Carullo M, Betran E, Moriyama EN (2007) Conflict between translation initiation and elongation in vertebrate mitochondrial genomes. *PLoS One* 2:e227
- Xia X, MacKay V, Yao X, Wu J, Miura F, Ito T, Morris DR (2011) Translation initiation: a regulatory role for poly(A) tracts in front of the AUG codon in *saccharomyces cerevisiae*. *Genetics* 189(2):469–478
- Xiao L, Wang K, Teng Y, Zhang J (2003) Component plane presentation integrated self-organizing map for microarray data analysis. *FEBS Lett* 538(1–3):117–124
- Xu Z, Hao B (2009) CVTree update: a newly designed phylogenetic study platform using composition vectors and whole genomes. *Nucleic Acids Res* 37(Web Server):W174–W178
- Yamaoka Y, Kita M, Kodama T, Imamura S, Ohno T, Sawai N, Ishimaru A, Imanishi J, Graham DY (2002) *Helicobacter pylori* infection in mice: role of outer membrane proteins in colonization and inflammation. *Gastroenterology* 123(6):1992–2004
- Yang Z (1995) A space-time process model for the evolution of DNA sequences. *Genetics* 139:993–1005
- Yang Z (2006) Computational molecular evolution. Oxford University Press, Oxford
- Yang Z, Yoder AD (2003) Comparison of likelihood and Bayesian methods for estimating divergence times using multiple gene Loci and calibration points, with application to a radiation of cute-looking mouse lemur species. *Syst Biol* 52(5):705–716
- Yang Z, O'Brien JD, Zheng X, Zhu HQ, She ZS (2007) Tree and rate estimation by local evaluation of heterochronous nucleotide data. *Bioinformatics* 23(2):169–176

- Yang Z, Bruno DP, Martens CA, Porcella SF, Moss B (2010) Simultaneous high-resolution analysis of vaccinia virus and host cell transcriptomes by deep RNA sequencing. *Proc Natl Acad Sci U S A* 107(25):11513–11518
- Yates JR (2004a) Mass spectral analysis in proteomics. *Annu Rev Biophys Biomol Struct* 33:297–316
- Yates JR (2004b) Mass spectrometry as an emerging tool for systems biology. *BioTechniques* 36 (6):917–919
- Yip TT, Lomas L (2002) SELDI ProteinChip array in oncoproteomic research. *Technol Cancer Res Treat* 1(4):273–280
- Yoder AD, Yang Z (2000) Estimation of primate speciation dates using local molecular clocks. *Mol Biol Evol* 17(7):1081–1090
- Yoon JH, De S, Srikanth S, Abdelmohsen K, Grammatikakis I, Kim J, Kim KM, Noh JH, White EJ, Martindale JL et al (2014) PAR-CLIP analysis uncovers AUFI impact on target RNA fate and genome integrity. *Nat Commun* 5:5248
- Yoshinaka Y, Katoh I, Copeland TD, Oroszlan S (1985) Murine leukemia virus protease is encoded by the gag-pol gene and is synthesized through suppression of an amber termination codon. *Proc Natl Acad Sci U S A* 82(6):1618–1622
- You J, Cohen RE, Pickart CM (1999) Construct for high-level expression and low misincorporation of lysine for arginine during expression of pET-encoded eukaryotic proteins in Escherichia coli. *BioTechniques* 27(5):950–954
- Young JA, Johnson JR, Benner C, Yan SF, Chen K, Le Roch KG, Zhou Y, Winzeler EA (2008) In silico discovery of transcription regulatory elements in Plasmodium falciparum. *BMC Genomics* 9:70
- Yu KM, Liu J, Moy R, Lin HC, Nicholas HB Jr, Rosenquist GL (2002) Prediction of tyrosine sulfation in seven-transmembrane peptide receptors. *Endocrine* 19(3):333–338
- Yu Q, Chen D, König R, Mariani R, Unutmaz D, Landau NR (2004) APOBEC3B and APOBEC3C are potent inhibitors of simian immunodeficiency virus replication. *J Biol Chem* 279 (51):53379–53386
- Yu Y, Sweeney TR, Kafasla P, Jackson RJ, Pestova TV, Hellen CU (2011) The mechanism of translation initiation on Aichivirus RNA mediated by a novel type of picornavirus IRES. *EMBO J* 30(21):4423–4436
- Yuan ZC, Zaheer R, Morton R, Finan TM (2006) Genome prediction of PhoB regulated promoters in Sinorhizobium meliloti and twelve proteobacteria. *Nucleic Acids Res* 34(9):2686–2697
- Yuan J, Sheppard K, Soll D (2008) Amino acid modifications on tRNA. *Acta Biochim Biophys Sin Shanghai* 40(7):539–553
- Zhang S, Ryden-Aulin M, Isaksson LA (1996) Functional interaction between release factor one and P-site peptidyl-tRNA on the ribosome. *J Mol Biol* 261(2):98–107
- Zhang L, Zhou W, Verculescu VE, Kern SE, Hruban RH, Hamilton SR, Vogelstein B, Kinzler KW (1997) Gene expression profiles in normal and cancer cells. *Science* 276(5316):1268–1272
- Zhang HM, Ye X, Su Y, Yuan J, Liu Z, Stein DA, Yang D (2010) Coxsackievirus B3 infection activates the unfolded protein response and induces apoptosis through downregulation of p58IPK and activation of CHOP and SREBP1. *J Virol* 84(17):8446–8459
- Zharkikh A (1994) Estimation of evolutionary distances between nucleotide sequences. *J Mol Evol* 39:315–329
- Zheng CL, Fu XD, Gribskov M (2005) Characteristics and regulatory elements defining constitutive splicing and different modes of alternative splicing in human and mouse. *RNA* 11 (12):1777–1787
- Zhou J, Korostelev A, Lancaster L, Noller HF (2012) Crystal structures of 70S ribosomes bound to release factors RF1, RF2 and RF3. *Curr Opin Struct Biol* 22(6):733–742
- Zhu C, Byrd RH, Lu P, Nocedal J (1997) Algorithm 778: L-BFGS-B: fortran subroutines for large-scale bound-constrained optimization. *ACM Trans Math Softw* 23(4):550–560
- Zhu J, Liu JS, Lawrence CE (1998) Bayesian adaptive sequence alignment algorithms. *Bioinformatics* 14(1):25–39

- Zhu Z, Li L, Zhang Y, Yang Y, Yang X (2015a) CompMap: a reference-based compression program to speed up read mapping to related reference sequences. *Bioinformatics* 31(3):426–428
- Zhu Z, Zhang Y, Ji Z, He S, Yang X (2015b) High-throughput DNA sequence data compression. *Brief Bioinform* 16(1):1–15
- Zid BM, Rogers AN, Katewa SD, Vargas MA, Kolipinski MC, Lu TA, Benzer S, Kapahi P (2009) 4E-BP extends lifespan upon dietary restriction by enhancing mitochondrial activity in *Drosophila*. *Cell* 139(1):149–160
- Zien A, Ratsch G, Mika S, Scholkopf B, Lengauer T, Muller KR (2000) Engineering support vector machine kernels that recognize translation initiation sites. *Bioinformatics* 16(9):799–807
- Zon LI, Gurish MF, Stevens RL, Mather C, Reynolds DS, Austen KF, Orkin SH (1991) GATA-binding transcription factors in mast cells regulate the promoter of the mast cell carboxypeptidase A gene. *J Biol Chem* 266(34):22948–22953
- Zuckerkandl E, Pauling L (1965) Evolutionary divergence and convergence in proteins. In: Bryson V, Vogel HJ (eds) *Evolving genes and proteins*. Academic, New York, pp 97–166

# Index

## A

Acetylation, 418–420  
Acid-resistance  
  evolution, 407–408  
  *H. pylori*, 406  
  hypotheses and tests, 405–406  
  positively charged membrane proteins, 406  
  urease gene cluster, 405–406  
Acid-resistance in *H. pylori*, 406  
Affine function gap penalty, 33, 34, 40, 55, 60  
Align nucleotide sequences against amino acid sequences, 62–75  
Alignment  
  affine function gap penalty, 55–62  
  constant gap penalty, 36–39  
Amino acids  
  asparagine, 419  
  dissimilarity, 316  
  hydropathy index, 48–55  
  lysine, 418–420  
  molecular weight, 48–55  
  proline, 419  
  properties, 48–55  
  serine, 419  
  substitution, 48–55  
  threonine, 419  
  usage, 204, 234–237  
ARSDA, 115–117, 120–123, 125

## B

*Bacillus subtilis*, 179, 183, 184, 226, 248, 250, 257  
Backtrack matrix, 36–38, 61, 64, 65, 158, 162

Binomial distribution, 4, 5, 382, 383, 425, 438, 439  
Bit-score, 7, 22  
BLAST  
  bit-score, 7, 22  
  e-value, 123, 408  
  Karlin parameter, 10  
BLOSUM matrix, 46–48, 51, 53, 55  
  derivation, 46–48  
Bootstrapping, 337–338, 369–370  
Branch-and-bound search, 334–335  
CCephalochordates, 207  
Charge deconvolution, 423–428  
Codon  
  family, 199, 202, 206, 207, 210, 212–216, 221–223, 226–229, 233, 253, 315, 316  
  usage bias, 84, 197, 211, 214–216, 221, 224, 225, 229, 233, 237, 416  
Codon adaptation index (CAI), 84, 181, 197, 215, 221–230, 233, 416  
  problems, 227  
Codon-anticodon adaptation, 204–208, 210, 211, 213, 221, 223, 229  
Codon degeneracy  
  2-fold, 203  
Codon usage, 208  
  CAI, 84, 181, 197, 215, 221–230, 233, 416  
  ITE, 178, 181, 186, 197, 206, 215, 223, 226–233, 416  
  mutation bias, 208–210  
  RSCU, 197, 215–221, 224  
  tAI, 215, 226, 229  
Codon usage bias, 84, 197, 211, 214–216, 221, 224, 225, 229, 233, 237, 416

- Constant gap penalty, 33, 34, 36, 40, 55, 58, 60, 64, 65  
 Content sensors, 255–268  
 CpG, 258, 262, 263  
     deficiency, 206, 258, 261–263, 268, 348  
     dinucleotide, 256, 258, 262, 266, 348, 349, 420
- D**  
 DAMBE, 69, 72, 77, 82, 83, 90, 91, 98, 100, 101, 108–110, 123, 159, 167, 174, 178, 182, 192, 206, 223, 227, 229, 257, 260, 288, 300, 309, 316, 337, 338, 344, 345, 350, 356, 367, 369, 371, 399, 403, 413, 414, 417  
 Dating  
     application, 370–378  
     internal-node calibration, 356–367  
     local clock, 361–366  
     multiple distance matrices, 374–375  
     multiple genes, 360–361  
     tip-dating, 367–369  
 Dating speciation and gene duplication events, 356–378  
 Deamination, 258  
 Delta method, 437  
 De novo motif discovery, 100–101, 142  
 Dinucleotide frequencies, 259  
 Distance-based phylogenetic methods, 350–355  
     branch length estimation by LS, 350–352  
     dating, 356–378  
     minimum evolution (ME), 350–352  
     neighbor-joining (NJ), 352–355  
     UPGMA, 133–137  
 Distribution  
     binomial, 5  
     Poisson, 5, 8, 9, 273, 277, 283, 286, 338  
 DNA methylation, 258–261  
     controversy, 261–263  
     CpG deficiency, 206, 258, 261–263, 268, 348  
     indices, 258–260  
     key questions, 265–267  
     phylogeny-based study, 261–263  
 Dynamic programming  
     Fitch algorithm, 66, 68, 164–169, 327–330, 333, 338, 339  
     forward algorithm, 164–169  
     profile alignment, 35–62  
     pruning algorithm, 34, 381, 382, 385, 387–389, 391–394  
 Sankoff algorithm, 34, 327–334, 338  
 sequence alignment, 63–66  
 Viterbi algorithm, 34, 145, 152, 157–164, 167
- E**  
 EM algorithm, 440, 441  
 Emission probabilities, 156  
 Empirical substitution matrix, 35, 43, 44, 48, 51, 301, 317  
*Escherichia coli*, 29, 30, 49, 93, 97, 98, 115, 116, 121–123, 170, 175, 176, 178–183, 194, 204, 217, 218, 224, 225, 227, 228, 230, 233, 235–237, 240, 241, 243–245, 247, 249, 250, 253, 257, 310, 403, 406, 409, 416  
 Eukaryotic translation initiation, 186–190  
     poly(A), 184–188, 190, 219  
 e-value, 123, 408  
 Evolutionary distance  
     F84, 281–288  
     GTR, 272, 279, 294, 297, 300–309, 312, 345  
     HKY85, 281–288  
     independent estimation, 300, 344–350  
     JC69, 270, 273–276, 291–292, 295–298, 307, 310–313, 333, 386, 387, 389, 391, 392, 394  
     K80, 270, 276–282, 284, 292–293, 297–301, 306, 307, 311, 313, 333, 345, 439  
     simultaneous estimation, 300, 344–350  
     simultaneous estimation (LS), 346–347  
     simultaneous estimation (ML), 345–346  
     TN93, 43, 272, 280, 288–294, 314, 345  
 Extreme value distribution, 9, 89
- F**  
 False discovery rate (FDR), 86  
     Benjamini-Hochberg, 86  
     Benjamini-Yekutieli, 86  
 FASTA algorithm, 6, 15–20, 23  
 Fetal hemoglobin (HbF), 27  
 Fitch algorithm, 66, 68, 327–330, 333, 338, 339  
 F84 model, 281–288  
 2-Fold degenerate site, 203  
 Forward algorithm, 164–169
- G**  
 Gap extension, 7, 8, 33, 40, 57, 61, 62, 70  
 Gap open, 7, 8, 33, 40, 57, 61, 62, 70  
 GC skew, 256–258

Gene expression, 2, 3, 24, 25, 111, 113–115, 118, 121–124, 129–132, 136–138, 140–143, 204, 206, 215, 230, 233, 247, 264, 416, 419  
Gene prediction, 3, 169–171  
Genetic code, 199–204  
    evolution, 199–204  
    relationships, 199–204  
Genomic pI profiling, 403  
Gibbs sampler, 39, 66, 78, 84, 98–111, 114, 129, 142, 382  
    algorithmic details, 102–111  
    de novo motif discovery, 100–101, 142  
    motif sampler, 101, 110, 111  
Global alignment, 39  
Globin, 24, 27, 111, 265  
Glycosylation, 419  
GTR model, 272, 279, 294, 297, 300–309, 312, 345  
Guide tree, 34, 35, 58, 60, 63, 68, 69, 344  
Gumbel distribution, 9, 89

## H

*Helicobacter pylori*, 397–412, 416, 431  
Henderson-Hasselbalch equation, 400  
Hi-C, 23, 24, 265  
Hidden Markov Model (HMM), 34, 145, 146, 151, 152, 154–158, 164, 167, 169–172  
    emission probabilities, 156  
    forward algorithm, 164–169  
    gene prediction, 3, 170, 171  
    training, 153–157  
    Viterbi algorithm, 34, 145, 152, 157–164, 167  
Hierarchical clustering algorithm, 130  
Highly expressed genes (HEG), 182, 203, 205, 206, 211, 215, 219, 221, 226–229, 246, 247, 252, 253  
HKY85 model, 281–288  
Homology, 263  
Indels, 261  
Index of translation elongation (ITE), 178, 181, 186, 197, 206, 211, 215, 223, 226–233, 416  
In silico 2-D gel, 414–418  
Internal ribosome entry, 190–193  
    secondary structure, 190–193  
Introns, 255, 260  
Isoacceptor tRNA, 198, 215, 233  
Isoelectric point, 413, 417, 418

## J

Jackknifing, 337–338, 369–370  
JC69 model, 270, 273–276, 291–292, 295–298, 307, 310–313, 333, 386, 387, 389, 391, 392, 394

## K

Karlin parameter, 10  
Karlin-Altschul parameter, 10, 12, 14  
K80 model, 270, 276–282, 284, 292–293, 297–301, 306, 307, 311, 313, 333, 345, 439  
Kozak consensus, 1, 97, 170, 184, 193–194, 202  
Kullback–Leibler divergence, 106  
Kullback-Leibler information, 101, 106, 382

## L

Lactase, 27, 264  
Lancelets, 207  
Likelihood ratio test, 149, 150, 269, 275, 279, 306, 311, 312, 334, 340, 375, 381, 385, 389  
    substitution models, 312–313  
Local alignment, 16, 34, 39  
LogDet distance, 347–349  
Long-branch attraction, 328, 332, 333, 335–336  
Lowly expressed gene (LEG), 182, 203, 215, 226, 227, 246, 247, 253

## M

Mahalanobis distance, 131  
Markov, 34, 98, 145–151, 255  
Markov model  
    the transition probability matrix, 145, 148, 150–152, 155, 157–159, 166, 170, 210, 274, 278, 291, 294, 317, 323  
Match-mismatch matrix, 40–55, 57  
Match-mismatch matrix for nucleotide sequences, 41–48  
Match-mismatch matrix for proteins, 48–55  
Methylation, 256, 258–263  
Methyltransferase, 262, 263  
Microarray, 418  
Migration distance, 414–416, 419  
ML phylogenetics  
    bias, 392–394  
    brute-force approach, 385–387  
    Fitch algorithm, 68, 327–330, 333, 338, 339

- ML phylogenetics (cont.)**
- imposing a clock, 389–392
  - long-branch attraction, 328, 332, 333, 335–336
  - missing data, 392–394
  - pruning algorithm, 34, 381, 382, 385, 387–389, 391–394
  - Sankoff algorithm, 34, 327–334, 338
  - statistical tests, 338–341
- Model selection**
- information-theoretic indices, 312–313
  - likelihood ratio test, 314
- Molecular mass**, 413–419
- Motif**, 419
- MS determination of peptide mass, 432
- Multiple alignment** with a guide tree, 68–70
- Mutation**, 263
- synonymous, 171, 259
- Mutation bias** and codon usage, 208–210
- Mutation rate**, 256
- Mycobacterium tuberculosis**, 28, 243
- Mycoplasma genitalium*, 261, 262
- Mycoplasma pneumoniae*, 262
- Mycoplasma pulmonis*, 262, 263
- pI**
- computation, 399–403
- Plasmodium falciparum*, 29
- Poisson distribution**, 5, 8, 9, 273, 277, 283, 286, 338
- Polymerases**, 419
- Position weight matrix (PWM)**, 77–98, 101–107, 111
- background frequencies, 82–83
  - pseudocounts, 83–84
  - refine multiple alignment, 90–93
  - statistical tests, 84–90
  - XOR problem, 97–98
- Posttranslational modification**, 417, 418
- Profile alignment**, 34, 63, 64, 66, 69
- Progressive alignment**, 35, 344
- Protein identification**, 422, 434–435
- Protein isoelectric point**
- computation, 399–403
- Protein mass spectrometry**, 422–423
- Protein secondary structure**, 153–157
- Protein substitution model**, 324
- Pruning algorithm**, 34, 381, 382, 385, 387–389, 391–394
- N**
- Near-cognate tRNA, 198, 249, 250
- Needleman-Wunsch algorithm, 35
- Non-hierarchical cluster algorithm, 137–143
- Nonsynonymous codons, 53
- Nonsynonymous mutation, 53, 203
- Nonsynonymous substitution, 203, 211, 233, 259, 315
- N-terminal methionine excision, 193, 194
- O**
- Origin of replication, 255–257
- P**
- PAM matrix, 44–46, 51
- derivation, 44–46
- PAM30 matrix, 12
- Paralinear distance, 347–349
- Peptide digestion, 429–431
- Peptide mass fingerprinting, 422, 423, 429–435
- peptide digestion, 429–431
  - peptide mass, 432
  - protein identification, 422, 434–435
- Perceptron, 419
- Phylogeny, 262, 263
- R**
- Read per kilobases of transcript per million mapped reads (RPKM), 118, 122, 123
- Relative synonymous codon usage (RSCU), 197, 215–221, 224
- Release factors
- discovery, 240–241
  - GGQ motif, 243–244
  - structure, 241–243
- Ribosome occupancy, 189
- Ribosome profiling, 24, 26, 185, 186
- S**
- Saccharomyces cerevisiae*, 12, 26, 49, 79, 117, 136, 137, 185–190, 205, 206, 213, 214, 235–237, 249, 250, 413, 416, 417
- SAGE, 418
- Sankoff algorithm, 34, 327–334, 338
- Scoring scheme, 3, 7, 33–36, 38–41, 48, 55, 63, 64, 70, 71, 73, 344, 382
- Self-organizing map (SOM), 1, 130, 137–143
- similarity and distance indices, 131–133
- Sequence alignment with secondary structure, 71–72
- Sequence annotation, 260
- Simpson paradox, 237

- Smith-Waterman algorithm, 35, 39  
SOM algorithm, 130, 137–143  
Spontaneous deamination, 206, 256, 258–261  
**S**  
Stop codon  
  differential usage, 251–253  
  effect of flanking nucleotides, 244–245  
  extended stop signal, 244–245  
Stop codon usage  
  mutation bias, 246–247  
  release factor abundance, 247–248  
  tRNA effect, 249–251  
Strand asymmetry, 256  
Strand-displacement model (mtDNA), 257  
Substitution model  
  amino acid models, 317–333  
  F84, 270, 272, 280–289, 333, 344, 345, 371, 375, 376  
  GTR, 270, 272, 279, 294, 297, 300–309, 311–314, 344, 345  
  HKY85, 270, 272, 285–289, 314, 344, 373, 377  
  JC69, 270, 273–276, 281, 291–292, 294–299, 304, 307, 310–314, 333, 344, 386, 387, 389, 391, 392, 394, 439  
  K80, 270, 276–282, 284, 289, 291–294, 297–301, 306, 307, 311–314, 333, 344–346, 439–440  
  P matrix, 164, 165, 169, 271, 296–300, 302–304, 312, 317, 319  
  Q matrix, 272, 293, 299, 303–305, 317, 319  
  rate matrix, 269, 272, 273, 276, 281, 285, 288, 292–309, 317, 322, 324  
  selection, 311–314  
  TN93, 43, 270, 272, 275, 280, 288–294, 304, 311–314, 333, 344, 345  
Substitution saturation, 309–311  
Synonymous codons, 198, 210, 215, 226, 227, 229, 234, 235, 253  
Synonymous mutation, 203  
Synonymous substitution, 259, 315
- T**
- tAI  
  problems, 226–227  
Thalassemia, 24, 27, 33, 265  
Tip-dating, 367–369  
TN93 model, 43, 272, 280, 288–294, 314, 345  
Traceback matrix, 37  
Transcription, 419, 420  
Transcriptomic analysis  
allocating reads to paralogous genes, 115–119  
contrast ARSDA with Cufflinks, 124–125  
file size reduction, 119–121  
refine gene annotation, 121–124  
Transcriptomic data analysis, 114, 117, 129, 350  
The Transition probability matrix, 145, 148, 150–152, 155, 157–159, 166, 170, 210, 274, 278, 291, 294, 317, 323  
Translation elongation  
  a major controversy, 229–232  
Translation elongation accuracy, 232–234  
Translation elongation efficiency  
  two hypotheses, 210–211  
Translation initiation, 1, 77, 81, 97, 101, 117, 173–195, 202, 204, 205, 210, 211, 229–232, 236, 245  
aSD, 127, 170, 176, 202, 239  
bacteria, 175–184  
DtoStart, 177–181, 183  
eukaryotes, 184–194  
phage host specificity, 183–184  
SD/aSD model, 176–179  
secondary structure, 181–182  
Translation termination efficiency, 245, 251, 253–254  
tRNA  
  isoacceptor, 198, 215, 233  
  near-cognate, 198, 249, 250  
tRNA adaptation index (tAI), 215, 226, 229  
tRNA notation, 198–199  
tRNA pool, 208  
Trypanosoma brucei, 27  
Tunicates, 207
- U**
- UPGMA algorithm, 133–137  
Urochordates, 207  
USP15, 27  
USP4, 27, 81, 117, 126
- V**
- Vaccinia virus, 188, 189  
Viterbi algorithm, 34, 145, 152, 157–164, 167
- W**
- Wobble hypothesis, 211, 213, 214