CHAPTER 21

Computational Biology

Dimiter S. Dimitrov, Igor A. Sidorov
Laboratory of Experimental and Computational Biology, CCR, NCI-Frederick, National Institutes of Health, Frederick, Maryland, USA

Nikola Kasabov
Knowledge Engineering and Discovery Research Institute, School of Computer and Information Sciences, Auckland University of Technology, Auckland, New Zealand

CONTENTS

ISBN: 1-58883-042-X/$35.00

Handbook of Theoretical and Computational Nanotechnology
Edited by Michael Rieth and Wolfram Schommers
Volume X: Pages (1–45)

1. INTRODUCTION

A major goal of computational biology is to discover knowledge or to enhance knowledge discovery for biological systems through computation. Computational biology, a term coined from analogy to the role of computing in the physical sciences, is now coming into its own as a major element of contemporary biological and biomedical research [1]. Information and computational sciences provide essential tools for the next-generation biological science efforts, from focusing the direction of experimental studies to providing knowledge and insight that cannot otherwise be obtained. Going beyond the revolution in biology reflected in the successes of the genome project and driven by the power of molecular biology techniques, computational approaches will provide an underpinning for the integration of broad disciplines for development of a quantitative systems approach to understanding the mechanisms determining the life of the cell and organism. Another aspect of the integration of computation and biology is that biological systems can be viewed as special computing devices. This view emerges from considerations of how information is stored in and retrieved from the genes. Genes can only specify the properties of the proteins they code for, and any integrative properties of the system must be "computed" by their interactions. This provides a framework for analysis by simulation and sets practical bounds on what can be achieved by reductionist models [2].

Recent advances in many areas of biology, especially in genomics, are heavily rooted in engineering technology, from the capillary electrophoresis units used in large DNA sequencing projects to the photolithography and robotics technology used in chip manufacture, to the confocal imaging systems used to read those chips, to the beam and detector technology driving high-throughput mass spectroscopy. Further advances in materials science and nanotechnology promise to improve the sensitivity and cost of these technologies greatly in the near future [3]. Current research makes it possible to look at biological phenomena on a scale not previously possible: all genes in a genome, all transcripts in a cell, and all metabolic processes in a tissue.

These modern approaches produce massive quantities of data. GenBank, for example, now accommodates more than 10^{10} nucleotides of nucleic acid sequence data and continues to more than double in size every year. New technologies for assaying gene expression patterns, protein structure, protein–protein interactions, and so forth will provide even more data. How to handle these data, make sense of them, and render them accessible to biologists working on a wide variety of problems is the challenge facing computational biology and bioinformatics seeking to integrate computer science with applications derived from molecular biology.

One core aspect of research in computational biology focuses on database development: how to integrate and optimally query data from genomic DNA sequence, spatial and temporal patterns of mRNA expression, protein structure, immunological reactivity, clinical outcomes, publication records, and other sources. A second focus involves pattern-recognition algorithms for such areas as nucleic acid or protein sequence assembly, sequence alignment for similarity comparisons or phylogeny reconstruction, motif recognition in linear sequences or higher-order structure, and common patterns of gene expression. Both database integration and pattern recognition depend absolutely on accessing data from diverse sources and on being able to integrate, transform, and reproduce these data in new formats.

Computational biology is a fundamentally collaborative discipline, owing its very existence to the availability of rich and extensive data sets for analysis, integration, and manipulation.

Data accessibility and usability are therefore critical, raising concerns about data release policies—what constitutes primary data, who owns this resource, when and how data should be released, and what restrictions may be placed on further use.

2. COMPUTATIONAL GENOMICS AND PROTEOMICS

2.1. DNA Sequencing

DNA (deoxyribonucleic acid) is a nucleic acid polymer consisting of individual units termed nucleotides. Each nucleotide consists of one of four distinct nucleosides (deoxypentose sugar plus one of four bases (adenine [A], guanine [G], cytosine [C], and thymine [T])) and a phosphate group. Thymine is replaced by uracil (U) in RNA (ribonucleic acid). With respect to similarity in structure, nucleosides are divided in two classes: pyrimidines and purines. Nucleosides A, T, G, and C are capable of being linked together to form a long chain. The bases along the polymer can interact with complementary bases in the other strand: Adenine is capable of forming hydrogen bonds with thymine (A:T), and cytosine can pair with guanine (C:G). Thus, the DNA consists of two antiparallel strands and can be written, for instance, as

```
... A T C G C G T T A G C T A G C T ...
    | | | | | | | | | | | | | | | |
... T A G C G C A A T C G A T C G A ...
```

The main steps of DNA sequencing are the following: purified fragments of DNA are denatured to a single chain, and then one strand is hybridized to an oligonucleotide primer with small amounts of one of four chain-terminating nucleotides. After synthesis, the mixture, consisting of DNA fragments ending with one of the nucleotides, is electrophoresed to separate fragments by size. After this, one can calculate the probable order of the bands and predict the sequence. For the sequencing of larger molecules of DNA, the molecules are first randomly sheared, the fragments are sequenced then, and finally the sequence of the large molecule is assembled from the overlaps found.

After sequencing, the information about the sequence can be submitted to one of the data banks, including GenBank at the National Center of Biotechnology Information, National Library of Medicine, Washington, DC (http://www.ncbi.nlm.nih.gov/Entrez); the European Molecular Biology Laboratory (EMBL) Outstation at Hixton, England (http://www.ebi.ac.uk/embl/index.html); and the DNA DataBank of Japan (DDBJ) at Mishima, Japan (http://www.ddbj.nig.ac.jp/). A more extensive list of the data banks can be found in DBCAT (Public Catalog of Databases) located at http://www.infobiogen.fr/services/dbcat/. Different data banks may have different formats for storing the data, but most of them have the same features: sequence name and identification code, source organism, keywords to look up this entry, dates of entry and modification, and so forth.

2.2. DNA and RNA Structures

A nucleoside is one of the four DNA bases attached covalently to the sugar. The sugar in deoxynucleosides is 2′-deoxyribose, and ribose in ribonucleosides. The four different nucleosides of DNA are deoxyadenosine (dA), deoxyguanosine (dG), deoxycytosine (dC), and deoxythymidine (dT). A nucleotide is a nucleoside with one or more phosphate groups covalently attached to the 3′- or 5′-hydroxyl group or groups. The DNA backbone is a polymer with an alternating sugar–phosphate sequence. DNA is a normally double-stranded macromolecule with two polynucleotide chains (the double-helical nature of DNA was discovered in 1953 [4]). These chains are noncovalently held together by weak intermolecular forces and form a DNA molecule. Two DNA strands form a helical spiral, winding around a helix axis in a right-handed spiral with two polynucleotide chains running in opposite directions. The sugar–phosphate backbones wind around the helix axis. The bases of the individual nucleotides are on the inside of the helix. For DNA duplexes, the right-handed double helix

has 10 pairs per complete turn. Within the DNA double helix, the adenine:thymine base pair has two hydrogen bonds, compared to three in the guanine:cytosine pair. The two base pairs are required to be identical in dimensions by the Watson–Crick model. High-resolution X-ray crystallographic analysis of the ribodinucleoside monophosphate duplexes (G:C and A:U) showed that the distances between the glycosidic carbon atoms in the base pairs are close (10.67 and 10.48 Å, respectively).

RNA molecules are polynucleotides containing ribose sugars connected by phosphodiester linkages. Although RNA is generally single-stranded, double-stranded RNA molecules can be formed where uracil participates in a U:A pair. Single-stranded RNA have a tendency to fold back on themselves to form double-stranded structures like stacked double-helixes for the regions with paired bases and different loops (bulge, hairpin, internal, and multibranch) for unpaired bases. These elements form the RNA secondary structure.

Prediction of RNA secondary structure requires intensive computational resources. Usually, RNA resultant structure corresponds to the local minima of the free energy, and overall free energy of the molecular folded is the sum of the energies of the stacked base pairs and loops. However, the molecular environment and folding pathway, which can have a significant effect on this structure, should be accounted.

In Structurelab [5], dynamic programming algorithm and genetic algorithm were used for determination of the folding of the RNA molecules. This system allows researchers to pursue interactively and methodically a multiperspective analysis of RNA structure (multiple and individual). It uses various software modules and hardware complexes [6]. Secondary structure representation of RNA molecular structures is based on LISP's nested list notations, for instance [N(H)(H)(BH)(H)(H)(H)(BBBIH)], where the symbols are H, hairpin loop; B, bulge loop; I, internal loop; and M, multibranch loop.

Other packages that can be used for secondary structure prediction and presentation are *mfold*, http://www.bioinfo.rpi.edu/~zukerm/rna/, http://bioweb.pasteur.fr/seqanal/interfaces/mfold-simple.html, http://biotools.idtdna.com/mfold/, http://www.bioinfo.rpi.edu/applications/mfold/old/rna/form1.cgi; RNAfold, http://rna.tbi.univie.ac.at/cgi-bin/RNAfold.cgi; ESSA, http://www.inra.fr/bia/T/essa/Doc/essa_home.html; and CARD, http://rrna.uia.ac.be/card.html.

2.3. Protein Sequencing

Proteins are linear polymers of amino acids linked by a peptide bond. The primary structure of the protein (protein sequence) determines the ultimate three-dimensional structure of the protein. Amino acids are small molecules consisting of an amino group (NH_2), a carboxyl group (COOH), a hydrogen atom attached to the central carbon (α), and a side chain (or R group) attached to the central carbon. There are 20 standard amino acids, which can be grouped into classes based on the chemical properties conferred by their side chains. Amino acids can be charged (+/−), hydrophobic/hydrophilic, polar/nonpolar, and capable of H-bonding—allowing for weak interactions. Amino acids can form peptide bonds with each other through reaction of the carboxyl and amino groups.

2.4. Protein Structure

Protein structure is typically considered on several levels.

The primary structure is the sequence of amino acids that make up a protein. This sequence determines the ultimate three-dimensional structure of the protein. The secondary structure involved local folding of peptide, creating distinctive structures shared by many proteins including alpha (α) helices and beta (β) pleated sheets. These structures were predicted theoretically before the experimental determination of protein structure, and they are the only regular secondary structural elements present in proteins (there are also irregular structural elements: loop and coil). Helix is created by a curving of the polypeptide backbone, and sheet is formed by hydrogen bonds between adjacent polypeptide chains rather than within a single chain. There are two configurations for both elements: rightward/leftward for helix and parallel/antiparallel for sheet.

The tertiary structure is a global, three-dimensional structure of the polypeptide chain. At this level of structure, the side chains play a major role in creating the final structure. Protein folding is the process of forming a final three-dimensional tertiary structure. It is interesting to note that random polypeptide sequences almost never fold into an ordered structure, so protein sequences were selected by the evolution to achieve reproducible stable structure [7].

Finally, the quaternary structure is the way multiple subunits of a protein interact. Many proteins are formed from more than one polypeptide chain (i.e., they exist as a noncovalent association of two or more identical or different polypeptides folded independently). The quaternary structure describes the way in which the different subunits are packed together to form the overall structure of the protein. For example, the human hemoglobin molecule is made of four subunits. Other examples of the combination of nonidentical subunits are immunoglobulins and bovine hemoglobin.

One task that has been explored in the literature is predicting the secondary structure from the primary one. Segments of a protein can have different shapes in their secondary structure, which is defined by many factors—one of them being the amino-acid sequence itself.

Protein secondary structure prediction can be performed by different packages, among them

1. NNPREDICT (http://www.cmpharm.ucsf.edu/~nomi/nnpredict.html) predicts using a two-layer, feed-forward neural network and a separate program—a modification of the Parallel Distributed Programming suite (see for details; see Refs. [8, 9]).
2. PHDsec (http://cubic.bioc.columbia.edu/predictprotein) predicts secondary structure from multiple sequence alignments. Secondary structure is predicted by a system of neural networks rating for the three states, helix, strand, and loop, at an expected average accuracy greater than 72% [10–12].
3. PROFsec (http://cubic.bioc.columbia.edu/predictprotein) is an improved version of PHDsec, a profile-based neural network prediction of protein secondary structure.
4. JPRED (http://jura.ebi.ac.uk:8888/) is a consensus method for protein secondary structure prediction [13].

Qian and Sejnowski [14] investigated the use of multilayer perceptrons (MLPs) for the task of predicting the secondary structure based on available labeled data. In Ref. [15], an Evolving Fuzzy Neural Network (EFuNN) is trained on the data from Ref. [14] to predict the shape of an arbitrary new protein segment. A window of 13 aminoacids was used; there were 273 inputs and three outputs, and 18,000 examples for training were used. The block diagram of the EFuNN model is given in Fig. 1 (from Ref. [15]).

Prediction of the three-dimensional structure of proteins by homology modeling (described in detail in the next section) is based on the similarity of primary sequences of the protein being analyzed to those of a protein of experimentally determined structure (this method

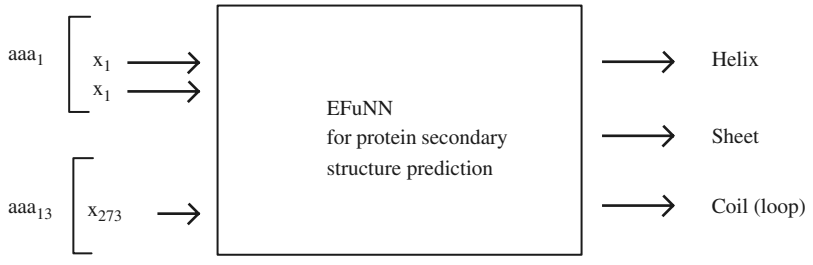

Figure 1. An artificial neural network model (in this case it is an evolving fuzzy neural network [15]) for the prediction of the protein secondary structure. Reprinted with permission from [15], N. Kasabov, "Evolving Connectionist Systems—Methods and Applications in Bioinformatics, Brain Study and Intelligent Machines." Springer, New York, 2002. © 2002, Springer.

assumes that significant identity between the two sequences exists). Algorithms for homology modeling can be found in the following servers:

1. SWISS-MODEL (http://www.expasy.ch/swissmod/SWISS-MODEL.html), an Automated Protein Modeling Server running at the GlaxoWellcome Experimental Research in Geneva, Switzerland (see, for details, http://www.expasy.ch/swissmod/SWISS-MODEL.html and [16, 17]).
2. CPHmodels (http://www.cbs.dtu.dk/services/CPHmodels/), Centre for Biological Sequence Analysis, The Technical University of Denmark, Denmark: Methods and databases developed to predict protein structures include Sowhat, a neural network–based method to predict contacts between C-alpha atoms from the amino acid sequence, and RedHom, a tool to find a subset with low sequence similarity in a database (see Ref. [18]).
3. 3D-JIGSAW Comparative Modelling Server (http://www.bmm.icnet.uk/~3djigsaw/); BioMolecular Modeling Group, Imperial Cancer Research Fund, London: an automated system to build three-dimensional models for proteins based on homologs of known structure ([19–21])
4. SDSC1-SDSC Structure Homology Modeling Server (http://cl.sdsc.edu/hm.html) and Databases and Tools for 3-D Protein Structure Comparison and Alignment (http://cl.sdsc.edu/ce.html); San Diego Supercomputing Center, San Diego, CA [22, 23].

2.5. Searching for Motifs in Sequences

To find a motif (or pattern or consensus) in the sequence, one first needs to define it. Let us consider a sequence as a vector of symbols

$$X = (x_1, x_2, \ldots, x_L)$$

where L is sequence length and all symbols x_i belong to a finite set of symbols (or alphabet)

$$x_i \in A = \{a_1, \ldots, a_K\} \quad i = 1, \ldots, L$$

For DNA sequences, the alphabet is simply set of four letters

$$A = \{A, T, G, C\}$$

Searching for a functional motif can be considered like comparing two sequences: one is the target sequence (S), and the other is the motif (M)

$$S = (s_1, s_2, \ldots, s_{L_S})$$
$$M = (m_1, m_2, \ldots, m_{L_M})$$

having length L_S and L_M, respectively.

Let now assume that function $w_{ij} = w(a_i, a_j)$ expresses the weight of a combination of two symbols in comparing sequences. The simplest equation for this function is

$$w(a_i, a_j) = \begin{cases} 1 & a_i = a_j \\ 0 & a_i \neq a_j \end{cases}$$

In this case, position j of the motif in target sequence can be found as a value giving maximum for the following score

$$Q_j = \sum_{i=1}^{L_M} w(m_i, s_{i+j-1}) \quad j = 1, \ldots, L_S - L_M + 1$$

Motif can be also represented as a probability (frequency or weight) $p_i(a_j)$ to find the jth symbol a_j from alphabet A in position i of the motif. Using the values of matrix $\|p_i(a_j)\|$ (position weight matrix, PWM), the score value for each position can be calculated as

$$Q_j = \sum_{i=1}^{L_M} p_i(s_{i+j-1}) \quad j = 1, \ldots, L_S - L_M + 1$$

If for the ith position in the motif $p_i(m_i) = 1$ and equals 0 for all other symbols, then this approach is equal to the simplest case of the weight function. Graphically, both methods can be represented as shown in the Fig. 2. The matrix shown in this picture is a dot plot (i.e., a visual representation of the similarity between two sequences). Each axis represents one of the two sequences to be compared, and each cell represents weight or probability function values. To calculate Q_j for position j in a target sequence, one should summarize the values on the diagonal, starting from the cell $(1, j)$. For a case in which weight function has a simplest representation (1 for the equal symbols and 0 for different ones) and the values of this function are represented as empty and filled cells for 0 and 1, respectively, then the whole diagonal will represent sequences sharing similarity. For sequences that share only patches of similarity, diagonal stretches will be shown.

The method of dot matrix analysis was first described in Ref. [24]. It can be also useful for finding inverted repeats and self-complimentary repeats. The use of an enhanced dot plot for nucleic and protein sequences was described in Ref. [25]. An additional description of the method can be found in Ref. [26].

The package for detection of patterns and structural motif in nucleotide sequences (PatSearch) is described in Ref. [27]. It allows scanning for specific combinations of oligonucleotide consensus sequences with defined order, orientation, and spacing, and it also allows mismatches and mispairing below a user-fixed threshold (available at http://bighost.area.ba.cnr.it/BIG/PatSearch). The possible pattern units for this package are as follows: string, palindrome, hairpin loop, position weight matrix, repeat. It also uses logical patterns such as "either/or" and length constraints for specific combination of pattern units.

The TRANSFAC database (available at http://www.gene-regulation.com) on eukaryotic transcriptional regulation comprises data on transcriptional factors, their target genes and regulatory binding sites [28], and tools for a matrix-based search of transcription-factors' binding sites (MATCH). The algorithm of MATCH uses two values to score hints: the matrix similarity score and the core similarity score, which is close to the MatInspector algorithm [29].

2.6. Sequence Alignment

There are two main type of sequence alignment: pairwise (comparing two sequences) and multiple sequence (comparing more than two sequences). Multiple-sequence alignment is the procedure of comparing sequences by searching for the similarity in the subsets that are

TARGET

		S_1	S_2	S_3	S_4	•••	S_{L_S}
	m_1	Q_{11}	Q_{12}	Q_{13}	Q_{14}	•••	Q_{1L_S}
M	m_2	Q_{21}	Q_{22}	Q_{23}	Q_{24}	•••	Q_{2L_S}
O	m_3	Q_{31}	Q_{32}	Q_{33}	Q_{34}	•••	Q_{3L_S}
T	•••	•••	•••	•••	•••	•••	•••
I	m_{L_M}	Q_{L_M1}	Q_{L_M2}	Q_{L_M3}	Q_{L_M4}	•••	$Q_{L_ML_S}$
F							

Figure 2. Searching for the motif in target sequence. Arrow shows the direction of calculating the score function $Q_j = \sum_{i=1}^{L_M} Q_{i,i+j-1}$, where Q_{ij} equals $w(m_i, s_j)$ or $p_i(s_j)$.

in the same order in the sequences. Each subset can consist of one or more characters of the sequence and the gap or gaps between them.

Comparison of two or more sequences has a lot of biological rationales. Among them is the idea that gene sequences may have derived from common ancestral sequences, and thus the changes in sequence (mutation, insertion, and deletion) can show us the evolutionary course of the particular molecule. Another reason for multiple-sequence alignment is indicating the regions of common origin that may in turn coincide with regions of similar structure or similar function. Results of alignment can be used as a starting point for solving various tasks (predicting *de novo* the secondary structure of proteins and other knowledge-based structure predictions, resolving phylogenetic issues, and interpreting data from the human genome).

Let us have n sequences $S_1, S_2, \ldots, S_n$, and have each sequence be represented as a vector

$$S_i = (s_1^i, s_2^i, \ldots, s_{L_i}^i) \quad i = 1, \ldots, n$$

where N_i is the length of the ith sequence, and

$$s_j^i \in A = \{a_1, \ldots, a_K\} \quad \text{for all } i \text{ and } j$$

Let us now assume that each sequence can be represented with gaps (insertions/deletion), so

$$S_i = (g_0^i, s_1^i, g_1^i, s_2^i, g_2^i, \ldots, g_{L_i-1}^i, s_{L_i}^i, g_{L_i}^i)$$

where g_j^i, $j = 0, \ldots, L_i$ are the gaps inserted in the ith sequence at jth place. Alignment of n sequences can be represented as a matrix $R = \|r_{ij}\|$, with the following properties: $r_{ij} \in A \cup \{\text{gap}\}$, so the gap is included in the alphabet; each row matrix represents the ith sequence with gaps $r_i = S_i$; and each column cannot consist only of gaps.

Score function for multiple alignments depends on the weight function [$w_{ij} = w(a_i, a_j)$, scoring matrix] and the so-called gap-penalty function. The latter describes the decrease in score for gaps of given length and consists of the constant term-describing penalty for opening the gap (a) and of the penalty for each element in gap (b). The usual formula for the penalty for the gap having length (so-called affine gap penalty) is

$$Q(g) = a + bg$$

and one of its extensions

$$Q(g) = \begin{cases} a + b(g - q) & g > q \\ a & g \leq q \end{cases}$$

where q means that gap penalty for each element will be added only when gap size is greater than q.

It is obvious that gap-penalty functions have to be appropriate to the weight function to obtain a reasonable alignment. If the gap-penalty function is high enough with respect to the scoring matrix values, final alignments will never have gaps. However, too-small values of the gap-penalty function will lead to the alignment having gaps everywhere.

Two alignments can be compared using the same score function. The typical matrix of alignment is represented in Fig. 3. It should be noted that several different alignments can provide approximately the same alignment score.

A key element in evaluating the quality of a sequence alignment is the score matrix (or substitution score matrix) $w_{ij} = w(a_i, a_j)$, which assigns a score for aligning any possible pair of sequence elements. The theory of amino acid substitution matrices is described in Ref. [30] and is applied to DNA sequence comparison in Ref. [31].

Percent accepted mutation matrices (PAM, or Dayhoff amino acid substitution matrices) list the probability of change from one amino acid to another in homologous proteins during evolution. In deriving the PAM matrices, each change in the current amino acid at a particular site is assumed to be independent of previous mutational events at that site [32], so amino acid substitutions can be viewed as a Markov model. To calculate the values of

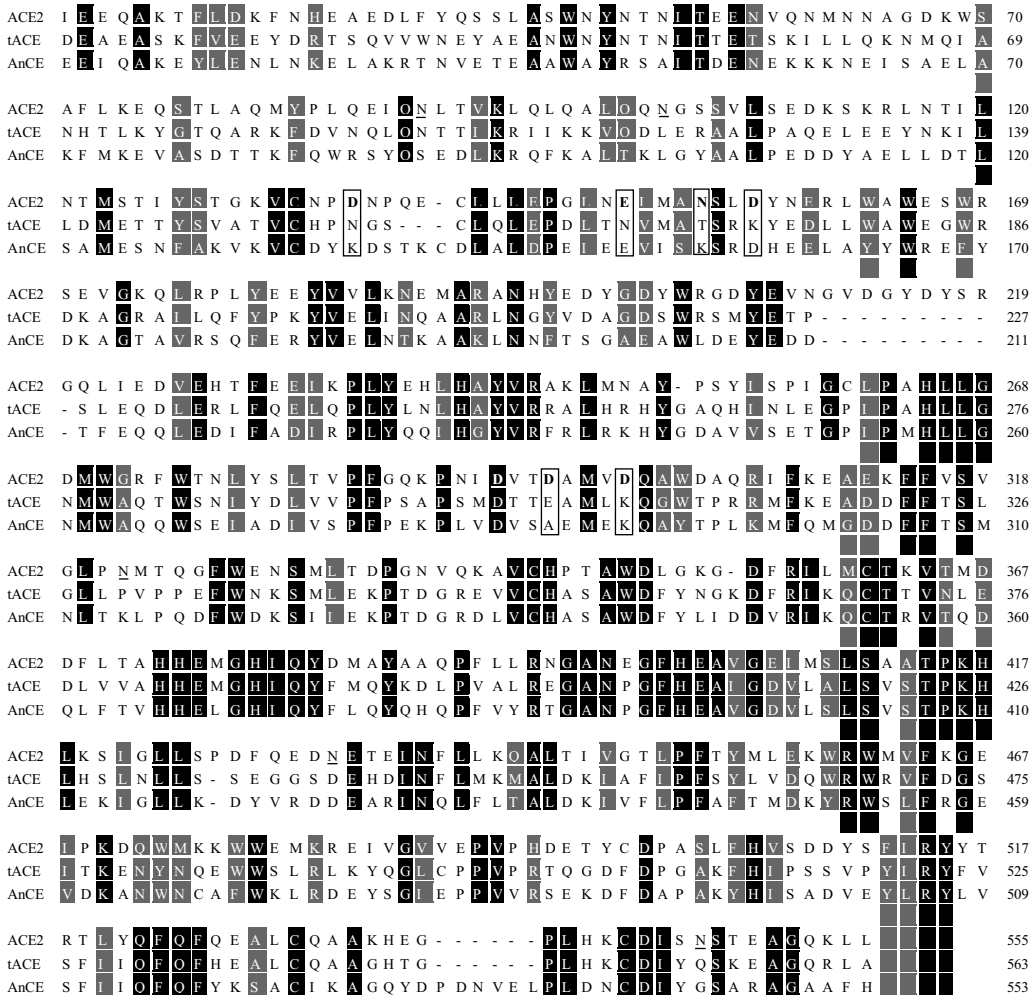

Figure 3. Multiple sequence alignment of angiotensin-converting enzyme 2 (ACE2), testis-specific ACE2 (tACE), and the *Drosophila* homolog of ACE2 (AnCE) using the program CLUSTALW. The sequence numbering is the same as in the crystal structures. The *N*-glycosylation sites are underlined; colored in blue, the putative binding residues in ACE2 are in boldface letters and boxed along with the corresponding aligned residues in tACE and AnCE. The identical and similar residues are shown in black and gray backgrounds, respectively. Conserved residues are in green, and the critical binding residues are in red.

Dayhoff matrices, amino acid substitutions that occur in groups of evolving proteins were estimated using 1572 changes in 71 groups of protein sequences with at least 85% similarity.

There are several approaches related to PAM. They can be based on exhaustive matching of the entire protein sequence database [33] or rapid generation of mutation data matrices from protein sequences [34], as well as accounting for the different patterns of mutation at low and high sequence divergence [35].

The BLOSSOM substitution matrix [36] is used for scoring protein sequence alignment and is based ondifferent type of sequence analysis and a much larger data set then the Dayhoff matrices. A 400×400 dipeptide substitution matrix that reports empirical probabilities for the interconversion of all pairs of dipeptides in proteins undergoing divergent evolution was presented in Ref. [37]. In Ref. [38], a mutation data matrix was calculated for membrane spanning segments.

Detailed descriptions of the methods and algorithms for alignment of pair of sequences and multiple sequence alignment can be found in Ref. [39].

In the late 1980s, fast algorithms for comparison DNA and protein sequences were developed that are capable of searching sequence databases, evaluating similarity scores, and identifying periodic structures based on local sequence similarity FASTP, FASTA, and

LFASTA [40, 41]. These algorithms achieve much of their speed and selectivity by using a look-up table to locate all identities between two sequences.

A new method, FastM, for the development of simple models of transcriptionally regulatory units was developed in Ref. [42]. It combines a search algorithm for individual transcriptional factor binding sites (MatInspector, see Ref. [29]) with a distance correlation function. The models are composed from various individual elements (hairpins, direct repeats, short multiple repeats, and terminal repeats; see Ref. [43] for details). FastM now is part of the GenomatixSuite, and the whole package (including ElDorado, a portal to explore various genomes; Gene2Promoter, for retrieval and analysis of promoters; BiblioSphere, for literature network mining; GEMS Launcher, Genomatix' genome exploring and modeling software package; MatInspector, the de facto standard for transcription-factor binding-site search; and PromoterInspector, highly specific prediction of mammalian promoter regions) is available at http://www.genomatix.de/cgi-bin/fastm2/fastm.pl.

A basic local alignment search tool (BLAST) for rapid sequence comparison was developed in 1990 [44]. It directly approximates alignments that optimize a measure of local similarity, the maximal segment pair (MSP) score. The basic algorithm is simple and robust, and it was applied in straightforward DNA and protein sequence database searches, motif searches, gene identification searches, and the analysis of multiple regions of similarity in long DNA sequences. A new generation of this algorithm for searching databases (gapped BLAST and PSI-BLAST) is described in Ref. [45].

2.7. Proteomics

Understanding the structures, interactions, and functions of all of a cell's or organism's proteins has been given a disciplinary title of its own: proteomics. The ultimate goal of proteomics is to characterize the information flow through protein networks.

The word proteome, coined in 1994 as a linguistic equivalent to the concept of genome, indicates proteins expressed by a genome. This term was coined by Marc Wilkins and colleagues and appeared for the first time in 1995 [46]. The term "proteome" is used to describe the complete set of proteins that is expressed, and modified following expression, by the entire genome in the lifetime of a cell. It can be used also as a description of proteins expressed by a cell at any stage. The generation of messenger RNA expression profiles is referred to as transcriptomics, as they are based around the process of transcription, and the mRNAs transcribed from a cell's genome is the transcriptome.

The field of proteomics has rapidly expanded and includes diverse technologies:

- Two-dimensional gel electrophoresis and mass spectrometry-based methods for protein profiling: These methods are currently the major experimental technologies for large-scale and high-throughput analysis of proteomes focusing on the proteins' identification and on their qualitative and quantitative comparison.
- Protein microarrays: Proteome profiling is a very powerful tool in clinical medicine for the identification of diagnostic markers. Clinical applications of proteomics can also provide information on drug targets, the mechanism of drug action, and drug-mediated toxicity.
- Yeast two-hybrid system [47]: Yeast two-hybrid assays are the main technology for large-scale interaction network construction. They can detect interactions between two known proteins or polypeptides. Two strategies (the matrix approach and the library-screening approach) have been tested to find the most efficient way to explore interactions within the proteome [48].
- Protein–protein interaction pathways and cell-signaling networks: This functional proteomics approach involves the generation of large-scale protein–protein interaction networks.
- High-throughput protein structural studies using mass spectrometry, nuclear magnetic resonance, and X-ray crystallography: The majority of drug target molecules are proteins, and for a large number of proteins, information on three-dimensional structures is required for drug discovery. Although high-throughput X-ray and nuclear magnetic

resonance methods hold much promise for protein three-dimensional structure determination in the future, at present the only efficient way to determine the structure is to implement large-scale homology modeling programs to accompany structural proteomics initiatives.
- High-throughput computational methods for protein three-dimensional structure and function determination.

The two key steps in classical proteomics are the separation of proteins derived from cells or tissues and their subsequent identification. One of the best methods of separation is two-dimensional gel electrophoresis. In this method, spots of a prepared mixture of proteins extracted from cells or tissues are applied to a polyacrylamide gel. The proteins, which can number in the tens of thousands, are separated along the gel in one direction according to two different properties: their molecular charge, by applying an electric field (isoelectric focusing), and their molecular mass, using SDS–PAGE (polyacrylamide gel) electrophoresis. A typical gel can reliably separate 2000 protein spots in this way, and the "best" ones can separate up to 11,000 protein spots. Proteins separated on the gel are stained using Coomassie blue dye, silver stains, or fluorescent dyes or by radiolabeling, and the proteins are then quantified using spectroscopic or radiographic techniques. Although two-dimensional gel electrophoresis gives the highest resolution of all available methods, there are difficulties in its using hydrophobic proteins, such as the cell-membrane-spanning receptor proteins (the most attractive drug targets), which usually do not dissolve in solvents used for isoelectric focusing (as well as proteins with very high relative molecular mass); low-abundance proteins, which cannot be recognized on the background of high-abundance "housekeeping" proteins (unfortunately, there is no amplification method for proteins, like there is a polymerase chain reaction method for genes); and proteins having very high molecular charges, or very low molecular mass, which will not separate on gels.

After separation, identification of the proteins begins with their digestion into fragments by specific proteases, and then the fragments are analyzed by mass spectroscopy in a process called peptide mass fingerprinting. In this approach, proteins are identified by comparing the mass of the peptide fragments with data predicted by genetic or protein sequence information.

Any experiment that involves a limited number of proteins can avoid the step with the two-dimensional gel separation by using other methods, such as high-performance liquid chromatography, gel filtration, or one-dimensional gel chromatography or microarray technology.

Protein microarray assays allow the identification and quantification of a large number of proteins from a small amount of a sample. They can be used for the analysis of interactions between proteins with other proteins, peptides, low–molecular weight compounds, oligosaccharides, or DNA.

The protein microarray is powerful tool for diagnostic and therapeutic purposes as well as for basic research. Array-based methods to study proteins allow high-throughput determination of protein functions in parallel [49]. It was demonstrated that a high-density antibody microarray could be applied for the global analysis of expression profiles of proteins, and different types of protein and peptide microarrays have been reported to be useful for immunoassays and for analyzes of enzymatic activity [50–52].

The major advantages of protein array technologies are based on the following features of the approach [53]: having a highly parallel and small solid-phase assay system, having a highly sensitive system, being useful for very high throughput approaches, having low consumptions of reagent samples, and having potentially attractive manufacturing costs.

The core technologies for protein microarrays currently practiced are surface chemistry for immobilization of proteins or capture agents, capture molecules that are immobilized onto a solid support and used for capturing target proteins or molecules, and systems to detect protein–protein interactions based on fluorescence, chemiluminescence, mass spectrometry (MS), and electrochemical or surface plasmon resonance (SPR). Capture on microarray can be specific (affibodies, antibodies, aptamers, and antibody sandwich formation) or unspecific based on electrostatic, van der Waals–hydrophobic, or metal–chelate interactions. Specific interaction microarrays have been described for receptor–ligand, protein–protein, protein–DNA, and enzyme–substrate interactions (see Ref. [54] for review).

Several studies illustrate the application of functional proteomics for the identification of regulated targets in specific pathways [55, 56].

Proteome profiling of microorganisms can also generates valuable knowledge that can be used for the development of metabolic and cellular engineering strategies. This approach includes the following steps [57]:

1. Obtain the proteome profiles of the microorganism under different conditions of interest.
2. Analyze the proteome profiling results based on biological, biochemical, and biotechnological information.
3. Develop a rational strategy for the engineering of the microorganism.
4. Compare and analyze the results (phenotypes) obtained by employing the microorganism before and after engineering.
5. Repeat the above steps until the results are satisfactory.

Protein microarrays have been used for the screening of molecular markers and pathway targets in patient-matched human tissue during disease progression. In contrast to protein arrays, in which immobilized capture molecules are directed against certain target proteins (e.g., an antibody), reverse-phase protein microarrays immobilize the whole repertoire of sample proteins that represent the state of individual tissue cell populations undergoing disease transitions.

Detailed review of the informatic tools for proteome profiling can be found in Refs. [58–60]. Development of software for two-dimensional gel image analysis began about 35 years ago [61–63] with further improvement that were made in the late 1980s [64–68]. There are many commercially available packages now, including DeCyder 2D Analysis, ImageMaster 2D Elite, http://www1.amershambiosciences.com/; Delta2D, http://www.decodon.com; GELLAB II+, www.scanalytics.com and http://www.lecb.ncifcrf.gov/gellab/index.html; GeneData Impressionist system, http://www.genedata.com; ImagepIQ, http://www.proteomesystems.com; Melanie 3, http://www.genebio.com; ProteinMine, http://www.scimagix. com; and TotalLab, http://www.nonlinear.com/products/totallab. Some of the two-dimensional gel image analysis packages can interact with automatic robotic systems.

Another approach used for identification of proteins is mass spectroscopy, and there are three different methods for identification based on mass-spectrometric data:

1. Peptide mass fingerprint or peptide mass map analysis: comparisons of peptide molecular weights determined by mass spectrometry with the theoretical masses of peptides produced *in silico* by digestion of sequences in a target database [69–73]. Database search tools include Mascot, http://www.matrixscience.com; Mowse, http://www.hgmp.mrc.ac.uk; PeptideSearch, http://www.narrador.embl-heidelberg.de; and Peptĭdent, http://www.expasy.ch.
2. Peptide sequence or peptide sequence tag query: peptide tandem mass spectrometry (MS/MS) data are combined with amino acid sequence or composition data to identify the protein from a protein or nucleotide sequence database [74, 75]. Database search tools include TagIdent, www.expasy.ch, and MS-Seq, http://prospector.ucsf.edu.
3. MS/MS ion search analysis: uninterpreted MS/MS data from a series of peptides in a complete LC-MS/MS run are matched with protein sequences in a protein- or nucleotide-sequence database to identify proteins without any manual sequence interpretation. The input data is usually a list of fragment ion mass and intensity values [73, 74]. Database search tools include PepFrag, http://prowl.rockefeller.edu, and Sequest, http://fields.scripps.edu.

The isotope-coded affinity tag (ICAT) approach and tandem MS were applied [76, 77] to quantitative protein profiling. Individual tandem MS spectra were searched against a human sequence database, and a variety of recently developed, publicly available software applications were used to sort, filter, analyze, and compare the results. In particular, robust statistical modeling algorithms were used to assign measures of confidence to both peptide sequences and the proteins from which they were likely derived, which were identified via the database searches. It was shown that these statistical tools allow the estimation of the accuracy of

peptide and protein identifications made. Data flow for automated database searching and statistical data analysis was represented in this work as follows: first, acquired MS/MS spectra are submitted to SEQUEST [74] for searching protein sequence databases to identify peptides and protein sequence matches for each recorded MS/MS spectrum; second, the search results are then submitted to PeptideProphet, a statistical data modeling algorithm [78], and this algorithm generates its own discriminant score for the peptide sequence assigned to each MS/MS spectrum, based on the weighting of a number of parameters for the peptide, including the various SEQUEST scores, the mass differential between the observed and calculated mass for the sequence in question, and so forth, third, combined SEQUEST/PeptideProphet output was displayed via the interface INTERACT, a Web-based application that allows the user to view the data as well as sort or filter it according to a range of user-definable parameters [79] (among other things, INTERACT can list all MS/MS scan-file locations with their assigned peptide sequences according to SEQUEST and their corresponding SEQUEST and PeptideProphet score values); finally, ProteinProphet [80] takes the INTERACT data file and derives a list of protein identifications and their corresponding scores from the observed peptide data.

It is not uncommon to find that mass spectrometric data cannot be correlated with any sequence in the database searched. This can happen for the following reasons [58]:

1. The sequence is absent in the database searched: the peptides may be derived from novel proteins or from variants (allelic or strain- or species-derived variants) of known proteins, or there may be errors in the reported sequence. *De novo* sequence analysis of peptides from the MS/MS spectra may be used to determine related proteins using homology-based database search methods (for instance, CIDentify [81], a homology-based sequence database search program).
2. The presence of unexpected co- or posttranslational modifications or chemical modifications (as artifacts of sample handling): FindMod (http://www.expasy.ch/tools/findmod) can be applied for high-throughput determination of protein posttranslational modification from peptide mass fingerprint data [82].
3. The peptide is produced by an unexpected or nonspecific cleavage.
4. The quality of the spectrum is poor or the spectrum originates from a nonpeptide contaminant. Purification and concentration of peptide samples before mass spectrometric analysis can be applied to improve the quality of the spectrum.

3. COMPUTATIONAL STRUCTURAL BIOLOGY

Technical advances have expanded the applicability of existing methods in structural biology and provide a basis for the discovery of general structural principles that underlie all cellular processes [83]. A major goal of computational structural biology is to help reach this goal and to predict structure and the structural basis of the function of biologically related molecules. Of all major classes of biomolecules including proteins, DNA, RNA, carbohydrates, and small molecules with biological activity, protein structures have been mostly studied computationally because of their importance and the variety of structures known. In the rest of this chapter, we will review the current state of three major areas in the structural biology of proteins: prediction of three-dimensional structures from sequence including homology modeling (for a recent review, see Ref. [84]) and fold recognition (recent review [85]); docking of proteins with known structures; and simulation of protein dynamics, with an emphasis on perhaps the most accurate methodology—homology modeling of protein three-dimensional structure.

Comparative (homology) modeling remains the only computational biology method at present that can provide models with a root-mean-square (rms) error lower than 2 Å. Computational methods for protein structure prediction based on related proteins of known structures were developed more than three decades ago [86]. Later, Greer outlined a basic protocol that is still followed today [87, 88]. Most homology modeling methods consist of four sequential steps [89]. The first step is to identify the proteins with known three-dimensional

structures that are related to the target sequence. The second step is to align them with the target sequence and to pick those known structures that will be used as templates. Any corrections in the alignment are made at this stage. The third step is to build the model for the target sequence given its alignment with the template structures. In the fourth step, the model is evaluated using a variety of criteria. If necessary, the alignment and model building are repeated until a satisfactory model is obtained. The main difference between the different comparative modeling methods is how the three-dimensional model is calculated from a given alignment (step 3 above). Because of the importance of step 3, sometimes it is divided into four stages [84]: backbone generation, loop modeling, side-chain modeling, and model optimization.

The level of sequence identity that is critical for the success of the homology modeling can be evaluated by sequence alignment programs such as BLAST, FASTA, and CLUSTALW. The latter program is used for multiple alignment, and the results contain significant amounts of additional information about the structural context that can be used to improve the alignment. It is especially useful, for example, to position deletions or insertions in places in which the sequences are widely divergent.

A widely used method is building a model by rigid-body assembly. The method constructs the model from a few core regions and from loops and side chains, which are obtained from dissected related structures. This assembly involves fitting the rigid bodies on the framework, which is defined as the average of the Cα atoms in the conserved regions of the fold. Another method is based on the approximate positions of conserved atoms from the templates to calculate the coordinates of other atoms. The third group of methods, modeling by satisfaction of spatial restraints, uses either distance geometry or optimization techniques to satisfy spatial restraints obtained from the alignment of the target sequence with homologous templates of known structure. In general, generation of the backbone is not difficult, especially when two or more template structures are available so complementary regions that are accurately determined are used.

The most significant problems in homology modeling remain the prediction of loop structures and side-chain modeling. The two main approaches for loop modeling are based on the use of already determined structures of loops and true *ab initio* prediction by using energy minimization or molecular dynamics techniques. These methods typically work relatively well for relatively short loops containing not more than six to eight residues. Most successful approaches for side-chain modeling are based on the use of libraries of common rotamers extracted from high-resolution X-ray structures. The prediction accuracy is typically much higher for residues in hydrophobic cores than for surface residues. This is mostly because of the flexibility of the side chains at the surface that can adopt multiple conformations.

After building the model, it can be optimized by several methods including iterative approaches. Such approaches are based on the idea of sequential prediction of loops and side-chains orientation, as well as energy minimization. The success of such an approach critically depends on the accuracy of the function describing the energy of the whole molecule. At present, the accuracy is not sufficient for accurate prediction, and typically only few hundred steps of energy minimization are used to avoid accumulation of small errors that can lead to structures completely different than the real one. Another straightforward approach is a molecular dynamics simulation of the model. Again, a major problem is the lack of sufficient accuracy of the force fields used for simulation; more accurate force fields are needed.

The best comparative techniques are able to produce models with good stereochemistry and overall structural accuracy when the modeling alignment is correct. The errors in comparative models depend on the level of sequence similarity between template and target sequences and on the errors in the template structure. They can be divided into five categories: side-chain errors, distortions and rigid-body changes in regions that are aligned correctly (e.g., loops, helices), distortions and rigid-body changes in insertions (e.g., loops), distortions in incorrectly aligned regions (loops and longer segments with low sequence identity to the templates), and incorrect folding resulting from an incorrect choice of a template. The consequence of these errors is that the comparative method can result in models with a main-chain rms error as low as 1 Å for 90% of the main-chain residues if a sequence

is at least 40% identical to one or more of the templates. In this range of sequence similarity, the alignment is mostly straightforward to construct, there are not many gaps, and structural differences between the proteins are usually limited to loops and side chains. When sequence identity is between 30% and 40%, the structural differences become larger and the gaps in the alignment are more frequent and longer. As a result, the main-chain rms error rises to ~1.5 Å for about 80% of residues. The rest of the residues are modeled with large errors because the methods generally cannot model structural distortions and rigid-body shifts and cannot recover from misalignments. Insertions longer than about eight residues usually cannot be modeled accurately at this times whereas shorter loops frequently can be modeled successfully.

An example of homology modeling with implications for elucidations of mechanisms of virus entry is described in more detail in the following. Enveloped viruses enter cells by binding their envelope glycoproteins to cell-surface receptors, followed by conformational changes leading to membrane fusion and delivery of the genome in the cytoplasm [90]. By using homology modeling, we have recently analyzed the three-dimensional structure of the angiotensin-converting enzyme 2 (ACE2) [91] that was recently identified as a functional receptor for the SARS virus [92], and its binding site on the SARS-CoVS glycoprotein was localized between amino acid residues 303 and 537 [93]. ACE2 is a homolog of the metalloprotease angiotensin-converting enzyme ACE [94, 95] and was found to be an essential regulator of heart function [96]. ACE exists in two isoforms: somatic ACE, which has two homologous domains each containing an active catalytic site, and testis-specific ACE (tACE), which corresponds to the C domain of somatic ACE and has only one active site. ACE2 has a high level of similarity (sequence identities 43% and 35% and similarities 61% and 55%, respectively) to tACE and the *Drosophila* homolog of ACE (AnCE). Recently, the crystal structures of tACE [97] and the *Drosophila* ACE homologue AnCE [98] have been determined at resolutions of 2.0 and 2.4 Å respectively.

These crystal structures were used as templates to build an accurate (rms deviation [rmsd] less than 0.5 Å) three-dimensional model of ACE2 by comparative (homology) modeling. Based on the ACE2 model, an analysis of the receptor-binding domain (RBD) of the SARS-CoV S glycoprotein, and similarity with other interactions of viral envelope glycoproteins (Env) with receptors [99], we proposed a possible mechanism of the ACE2 function as a receptor for the SARS virus. The analysis of the ACE2 model could also help in the design of experiments to further elucidate the structure and the dual function of ACE2.

The sequences of ACE2, tACE and AnCE, were aligned using the multiple sequence alignment program CLUSTALW [100]. The comparative modeling procedure COMPOSER [101, 102] implemented in SYBYL6.9 (Tripos Inc., St. Louis, MO) was used to build a three-dimensional model of the ACE2 structure. We used the tACE and AnCE structures to find out topologically equivalent residues based on structural alignment, and the structurally conserved regions (SCRs) were modeled. The structurally variable regions (loops) were modeled by using loops either from the corresponding location of the homologous protein or from the general protein database. The three-dimensional model of ACE2 was then subjected to energy minimization by using standard Tripos force fields and was finally validated with the PROCHECK program [103]. The coordinates were deposited to the protein data bank (PDB) (code: 1RIX).

The sequences of both ACE2 and the S RBD were scanned against the PROSITE [104] motifs to locate potential glycosylation sites. Six *N*-glycosylation sites with high probability of occurrences on ACE2 were predicted by PROSITE. Fully surface-exposed asparagine (*N*) residues were found at five of these sites, which were modeled by attaching *N*-acetylglucosamine moieties. Three *N*-glycosylation sites were found in the S RBD fragment and were modeled similarly. The areas of solvent accessibility (ASAs) were calculated with probe radius 1.4 Å by using the Lee and Richards's algorithm [105]. Electrostatistic potentials were calculated by using the program GRASP [106] with the following parameters: a protein dielectric constant of 2.0, a solvent dielectric constant of 80, an ion exclusion radius of 2.0 Å, a probe radius of 1.4 Å, and an ionic strength of 0.14 M. The calculated potentials were displayed at the solvent-accessible surface. The visualization of solvent accessibility, superpositioning of molecules, and calculation of surface hydrophobicity were performed by

using InsightII. The hydrophobicity of the surface residues was calculated according to the Kyte–Doolittle method [107] with a window size of 5 and hydrophobic and hydrophilic levels of 0.7 and −2.4, respectively.

To begin to understand the interactions between the SARS-CoV S glycoprotein and its recently identified receptor ACE2, we attempted to develop an accurate model of the ACE2 three-dimensional structure. We found two proteins, tACE and AnCE, with available high-resolution crystal structures and ACE2 sequence identities of 43% and 35% (sequence similarities are 61% and 55%), respectively; the sequence alignment of ACE2 with tACE and AnCE2 is shown in Fig. 3. Therefore, we have used homology modeling to build an accurate three-dimensional model of ACE2, as described in the section on materials and methods.

The architecture of the ACE2 model is very similar to the crystal structure of tACE (Fig. 4A). The superposition of the ACE2 model structure with the template structures of tACE and AnCE (Fig. 4B) shows very small deviation (rmsd less than 0.5 Å). A major feature of the ACE2 structure (and the template structures) is a deep channel on the top of the molecule that contains the catalytic site (Fig. 5A). A comprehensive analysis of the structure and function of the catalytic site was very recently reported after our model was completed [108]; here we will not discuss the enzymatic function of ACE2 but rather use the enzymatic site location for reference purposes. The channel is surrounded by ridges containing loops, helices and a portion of a β sheet. The long loop between N210 and Q221 that is missing in tACE and AnCE (Fig. 3) is on the ACE2 surface (Fig. 4B); note that the orientation of ACE2 in Fig. 4A is different than in Fig. 4A show this loop. Potential *N*-glycosylation sites were identified at six positions, 53, 90, 103, 322, 432, and 546, but only two of them (53 and 90) were aligned with the tACE structure (Fig. 3). They shared the pattern NXTX (except 103) and were modeled with a *N*-acetylglucosamine moiety (Fig. 5B). The direction of the main chain is illustrated in Fig. 5C.

Interactions of viral attachment proteins with protein receptor molecules are mostly determined by complementarity in surface charge distribution, hydrophobic interactions, and geometry; typically, carbohydrates are excluded from the binding sites [99]. In an attempt to provide a working hypothesis for possible regions involved in the interaction of the S glycoprotein with its receptor, we analyzed the ACE2 surface potential, solvent accessibility, hydrophobicity, and carbohydrate distribution. The surface of the deep channel at the top

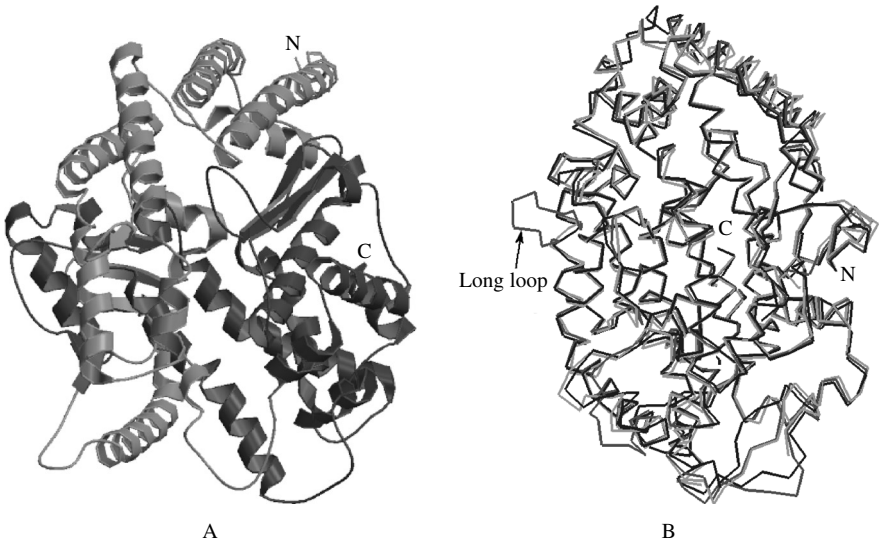

Figure 4. Model of the angiotensin-converting enzyme 2 (ACE2) structure. (A) A ribbon representation of the ACE2 model. The N and C termini are indicated. (B) Superposition of the ACE2 model structure with the crystal structures of testis-specific ACE (tACE) and the Drosophila homolog of ACE (AnCE) based on the Cα-atoms of ACE2, tACE, and AnCE (ACE2, dark gray; tACE, light gray; and AnCE, black). The long loop inserted between N210 and Q221 that is unique for ACE2 is indicated.

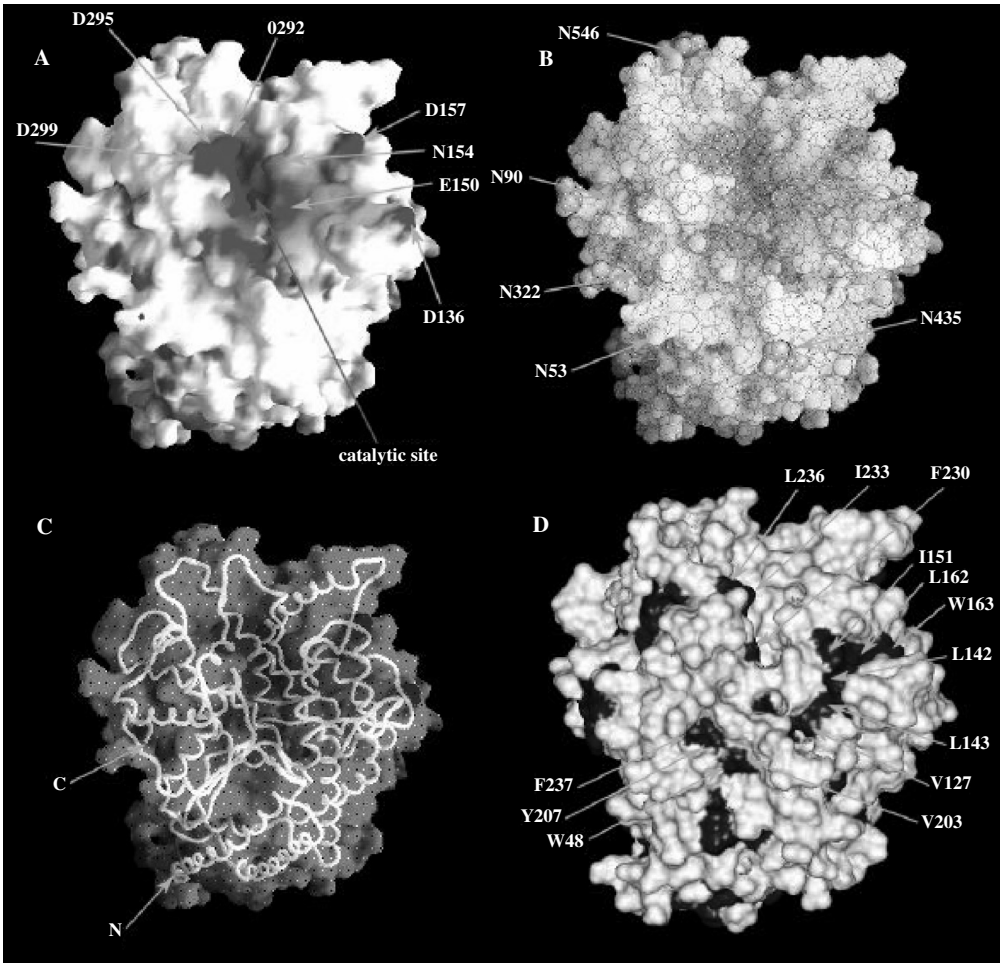

Figure 5. Analysis of the angiotensin-converting enzyme 2 model structure. (A) SARS S protein, receptor-binding domain model (fragment of amino acids 300–500 in S1 subunit). Ribbon diagram illustrating the receptor-binding domain of SARS S protein model (aa 300–499).

of ACE2 and the surrounding ridges is highly negatively charged (Fig. 5A). These ridges contain residues D136, E150, N154, D157, D292, D295, and D299, some of which have large ASA values (e.g., D136, N154, and D157 have values of 109, 108, and 80 Å^2, respectively; Fig. 6). Comparison of these residues with the corresponding residues from ACE that do not support fusion mediated by the S glycoprotein [92], and mouse ACE2 that binds to S but with somewhat lower affinity than human ACE2 (M. Farzan, personal communication) (Fig. 7), support the possibility that some of these residues contribute to specific binding. The hydrophobicity analysis revealed distinct hydrophobic patches in close proximity to the negatively charged ridges (Fig. 5D). There are at least three hydrophobic regions, comprising different residues including Phe, Trp, and Tyr, that could contribute to binding, in addition to the charged binding surface. All carbohydrate sites are topologically separate from the electronegative surface at the top of the molecule (Fig. 5B).

The sequence similarity of the S glycoprotein from the SARS virus with S glycoproteins from other coronoviruses or other proteins whose structures are available in the PDB is about 20% or lower. The sequence similarity of the attachment glycoprotein (S1) from the SARS-CoV to other coronanvirus S1 glycoproteins or other proteins with known three-dimensional structures is even lower. Such low sequence similarity does not allow accurate homology modeling. Because of the absence of significant sequence similarity, we built a model by threading (data not shown) a fragment (amino acid residues 300–537) containing the S RBD that we have recently identified [93]. The electrostatic analysis of the model

LYS 26 133.75	PRO 138 111.19	SER 257 57.50	SER 420 89.94
THR 27 106.68	GLN 139 124.05	PRO 258 65.89	GLY 422 45.02
ASP 30 90.32	GLU 140 96.63	ILE 259 167.18	PRO 426 120.83
LYS 31 146.81	PRO 146 62.39	SER 280 86.96	ASP 427 102.68
HIS 34 103.48	**GLU 150 82.31**	GLY 286 49.20	PHE 428 152.59
GLU 35 67.77	**ASN 154 107.65**	GLN 287 160.01	GLN 429 82.31
GLN 42 88.04	LEU 156 96.54	PRO 289 123.13	GLU 430 117.90
ASN 49 72.35	**ASP 157 79.60**	ASN 290 84.88	ASP 431 138.66
GLU 56 152.15	ASN 159 97.55	**ASP 292 83.20**	GLU 433 74.27
GLU 57 139.26	GLU 160 83.50	THR 294 52.85	THR 434 58.21
GLN 60 128.47	GLU 171 80.80	**ASP 295 54.90**	ILE 436 80.00
ASN 61 79.64	GLN 175 91.63	VAL 298 87.15	LEU 439 71.32
ASN 64 90.94	PRO 178 89.84	**ASP 299 95.50**	GLY 466 71.17
ASP 67 53.92	VAL 185 70.22	GLN 300 96.98	PRO 469 71.33
LYS 68 109.99	ARG 192 132.40	ALA 301 90.36	LYS 470 122.67
ALA 71 60.43	HIS 195 168.85	ASP 303 76.71	ASP 471 119.93
GLU 75 103.10	GLU 197 109.67	GLN 305 97.67	GLN 472 69.35
THR 78 67.46	ASP 206 71.43	ARG 306 96.24	LYS 475 132.50
LEU 79 73.79	GLU 208 146.30	LYS 309 81.39	GLU 479 81.98
GLN 81 110.23	VAL 209 76.31	LYS 313 119.53	ARG 482 104.76
MET 82 127.23	ASN 210 167.13	GLY 319 53.77	GLU 483 117.49
PRO 84 74.73	TYR 217 143.76	ASN 322 71.65	PRO 492 113.80
GLN 86 139.19	SER 218 82.84	THR 324 74.83	ASP 494 84.95
GLU 87 128.66	ARG 219 187.61	GLN 325 122.91	GLU 495 138.47
GLN 89 148.33	GLN 221 69.68	GLU 329 129.15	THR 496 85.99
ASN 90 69.13	GLU 224 73.82	ASN 330 68.41	TYR 510 93.92
LEU 91 131.77	ASP 225 75.54	THR 334 75.00	GLN 524 101.90
THR 92 69.98	HIS 228 88.24	ASP 335 76.53	GLU 527 95.03
LEU 95 83.29	GLU 231 81.03	GLY 337 84.09	ALA 528 47.83
GLN 98 93.66	GLU 232 121.46	ASN 338 115.17	GLN 531 138.22
SER 105 45.43	LYS 234 92.20	VAL 339 160.63	LYS 534 157.11
SER 109 72.60	PRO 235 74.61	LYS 353 109.79	GLU 536 137.88
GLU 110 129.44	LEU 236 88.19	ASP 367 76.79	GLY 537 61.61
ASP 111 86.02	GLU 238 84.35	ALA 386 39.72	PRO 538 67.51
LYS 114 116.29	HIS 239 127.38	ALA 387 103.41	LEU 539 75.65
ARG 115 111.85	ALA 242 46.26	GLN 388 77.36	LYS 541 93.81
THR 118 54.89	ALA 246 55.51	ASN 394 96.86	ASN 546 96.10
THR 129 63.36	ASN 250 117.89	ALA 396 81.30	THR 548 84.73
PRO 135 145.59	PRO 253 91.46	PRO 415 49.57	GLU 549 96.53
ASP 136 108.67	SER 254 80.22	LYS 416 129.10	GLN 552 114.05
ASN 137 58.91	TYR 255 141.34	LYS 419 93.66	LYS 553 118.29

Figure 6. Solvent-accessible surface areas (right column, in Angstroms squared) for angiotensin-converting enzyme 2 amino acid residues that are significantly exposed to solvent at the surface of the molecule. The cutoff for significant surface exposure here is assumed to be 45% ratio value, defined as the ratio of side-chain surface area to a "random coil" value per residue in the tripeptide Gly–X–Gly. The middle column represents the amino acid residue number.

revealed mostly positive charges on the surface and, in particular, an electronegative loop containing residues K439, R441, R444, H445, and K447. The hydrophobic analysis indicated several patches of hydrophobic residues around the positively charged loop region. One must note that although the size of the fragment is relatively small, the S RBD modeling has limitations in the absence of a template structure or structures with high sequence identity. Thus, the RBD model could significantly deviate or even be completely different from the real structure. In this aspect, modeling of the much larger S1 and S2 units is even less reliable. For example, a recent model [109] of S1 and S2 proposed putative-receptor (thought to be CD13) binding regions located in S2 instead of S1, where RBDs of coronaviruses should be. This is why we used our RBD model mostly as an illustration of possible complementary charged surfaces, hydrophobic patches, and β sheets and complemented it with an analysis of the secondary structure of the RBD fragment that also revealed the predominance of β

```
                155  167  171  174   300 303  307
Human ACE       ASN-ASN-THR-LSY-ASP-GLU-LYS
                  :    .    .    .    |   :    .
Human ACE2      ASP-GLU-ASN-ASP-ASP-ASP-ASP
                  .    |    .    |    |   |    :
Mouse ACE2      LYS-GLU-THR-ASP-ASP-ASP-ASN
                136  150  154  157  292 295  299
```

Figure 7. Conservation of amino acid residues in human angiotensin-converting enzyme (ACE), human ACE2, and mouse ACE2 that could contribute to interactions with the S glycoprotein. Identities are marked by a pipe (|), highly conservative replacements by a colon (:), and replacements with lower scores by a dot (·). The numbers denote the amino acid residue positions in the sequence. Note that the similarity of these ACE2 residues with the corresponding residues of mouse ACE2 is much higher than with the respective human ACE residues.

sheets (data not shown). In progress are our experiments for the SARS-CoV S glycoprotein RBD crystallization and determination of its three-dimensional structure.

Typically, virus receptors contain ridges that bind to cavities or to structures containing loops, cavities, and channels in the proteins mediating entry [99]. The model structure of ACE2 indicates that some or most of the ridges surrounding the cavity at the top of the molecule (Fig. 8) could serve as a likely binding region for the S glycoprotein for the following reasons. First, the top of the molecule is far away of the membrane and is likely to be easier to reach than membrane proximal regions. Second, protruding structures are likely to be used for binding by viral proteins; they will also ensure geometric complementarity with concave surfaces, as the S RBD domain could be based on our illustrative model (Fig. 8). Third, the negative charges of the ridges complement the positive charges of the RBD. Fourth, the hydrophobic patches around the charges could contribute to binding. Finally, the lack of carbohydrates at the top of the molecule could ensure high-affinity binding. Experiments currently in progress will determine the specific amino acid residues and their relative contribution to the interaction of ACE2 with the S glycoprotein. The ACE2 model developed here, and this proposition of binding regions, could help in the design and analysis of the experimental data and of the virus-binding function of ACE2.

4. COMPUTATIONAL CELL BIOLOGY

4.1. Introduction

As noted on the official Web site of the First International Symposium on Computational Cell Biology [http://caboy.uchc.edu/conference/], "The formulation of hypotheses based on complex experimental data is often impossible without the construction of computational models. Computational cell biology is an emerging discipline that responds to the need for computational methods to analyze and organize the abundance of experimental data on the structure and function of the cell."

Historically [110], mathematical biology has had limited success, turning in time into a somewhat abstract discipline. Several examples of early success in mathematical modeling in biology was demonstrated in following works: the Lotka–Volterra predator–prey model in ecology [111, 112], Hodgkin–Huxley's model of nerve conduction [113], Manfred Eigen's

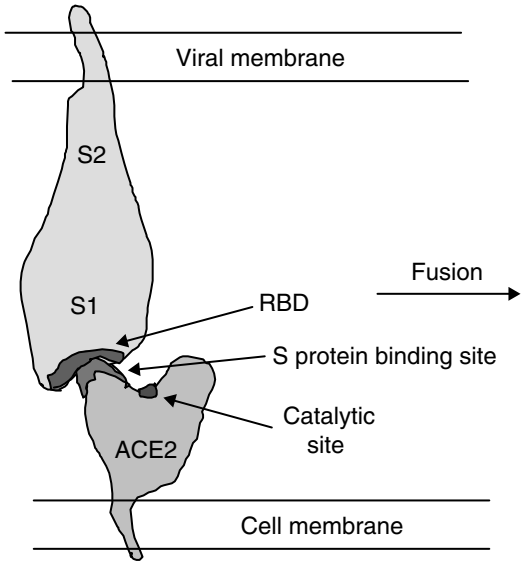

Figure 8. Schematic representation of the interaction between angiotensin-converting enzyme 2 (ACE2) and the SARS-CoV S glycoprotein receptor-binding domain (RBD) leading to binding and fusion. The RBD is depicted as a surface containing a cavity that binds a ridge close to the deep channel containing the catalytic site.

theory of molecular evolution [114], and the Gierer–Meinhardt theory of biological pattern formation [115]. Extensions of these models were considered later by many biologists as a mathematical refinement with limited practical utility (with some exceptions for the mathematical models in neurobiology and cardiology). Theoretical and experimental biology have grown largely apart, and the reason is that biology can rarely present simple experimental system for the evaluation of theoretical models. Biological systems are complex and open, and various factors can change the behavior of the system, so even simple computational modeling requires continuous interaction between model building and experimental verification. The interaction between theoretical models and experimental design can be represented schematically, as shown in Fig. 9 (modified from more detailed diagrams [110, 116]).

The complexity of biological objects can be defined as a "large number of functionally diverse, and frequently multifunctional, sets of elements which interact selectively and nonlinearly to produce coherent rather then complex behavior" [117]. Biological events occurring at various levels (such as organism, tissue, cell, and molecular), and the complexity of the system being modeled, lead to the need for integration of the different models. For instance, for modeling transduction of the activation signal into a cell, one may need to include gene regulatory network, models of proteins pathways, models of membrane, diffusion of molecules and ions into cell, and so forth. Some of these models may be available for investigators, but they are most likely different in format, programming language, and computing platforms, so one may need to develop the tools for unification of the models and of communication between them. At present, there are two ongoing projects for introducing standards in the model communication: System Biology Markup Language (http://www.cds.caltech.edu/erato/sbml/docs) and CellML (http://www.cellml.org).

The next challenge related to the complex nature of biological systems is the interpretation of the results of modeling, model prediction, and finding how the model should be changed with respect to new experimental data. For a model consisting of hundreds of equations and parameters (even for smaller numbers), the problems such as fitting to the set of experimental data, investigating and optimizing the model behavior, and correcting the model design with respect to the experimental data are not trivial. One of the possible ways to solve this problem is to identify the semiautonomous functional units in the model with known parameters and system behavior. So building a complex model or estimating model parameters can be accomplished in a stepwise or module-based manner. As an example, the modeling of activation of MAP kinase through receptor tyrosine kinases and GTP-binding protein RAS can include the modeling of three modules: RAS activation through signaling from the epidermal growth factor (EGF) receptor, recruitment of adaptor proteins Shc and Grb2, and activation of the exchange factor Sos; cascade of proteins kinases from Raf1 to MAP; and modulation of Raf1 by an inhibitory phosphorilation event by protein kinase A (PKA). All three modules have been modeled [118, 119] and can be combined in one system. Another example of using the approach is represented in [120], where a simple model of T-lymphocyte proliferation was combined with the model of viral hepatitis B, and then

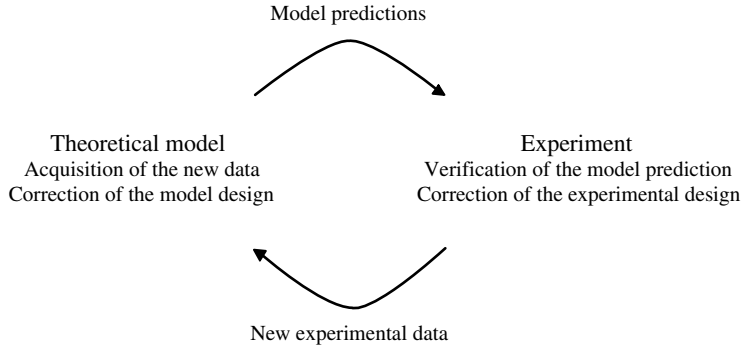

Figure 9. Interaction between theoretical model and experiment. Reprinted with permission from [15], N. Kasabov, "Evolving Connectionist Systems—Methods and Applications in Bioinformatics, Brain Study and Intelligent Machines." Springer, New York, 2002. © 2002, Springer.

the final, rather simple, model, consisting of only 17 equations and about 70 parameters, was fitted to the data describing the so-called "generalized picture" of hepatitis B [121]. In this work, three functional units corresponding to cell proliferation of helper and effector T-lymphocytes and production and consumption of soluble factors interleukin 1 and interleukin 2 were used, and the model was fitted to the data in three steps. By analogy to how biologists generally deal with the complexity of biological systems, this modeling approach is based on a hierarchical view of the biological object, that is composed of functional units with specific inputs and outputs. These units can communicate with other modules or units and can themselves be composed of functional units.

Additional problems that arise in computational biology are the optimization of the computational tools, improving the efficiency of existing algorithms, and creating new efficient packages. Complex computational problems require better algorithm development and powerful computers. It should be noted that it is not entirely a mathematical or computer problem, and in some cases, reconsideration of the underlying assumptions of the model and reformulation of these assumptions (that will lead to the changing in model equations) may provide significant improvement of the algorithm.

Another consistently raising (in the direction from experimental data toward theoretical model) issue, mentioned in Ref. [122] is the need for cell biologists to produce more quantitative results about biological processes. Some of the data are often difficult to obtain, and they have been insufficiently valued by biomedical researchers.

4.2. Computational Modeling for Cell Biology

There are five main phases of information processing and problem solving in most of the bioinformatics systems (as in a detailed description of the interaction between theoretical model and experimental design described above).

1. Data collection: Collecting biological samples and processing them, and primary processing of the data (quantification of the experimental results, normalization, statistical analysis of the data).
2. Analysis and extraction of the model features: Defining which features (parameters, variables, modules, etc.) are more relevant and which, therefore, should be used when creating a model for a particular problem (e.g., classification, prediction, decision making), making an assumptions model, and choosing the tools and algorithms for modeling.
3. Modeling the problem: Defining inputs, outputs, and type of model (e.g., probabilistic, rule-based, connectionist), training the model, and statistical verification.
4. Knowledge discovery *in silico*: Making calculation experiments, fitting model to the data, and gaining new knowledge through the analysis of the modeling results and the model itself.
5. Verifying the discovered knowledge *in vitro* and *in vivo*: Making biological experiments in both laboratory and in real life to confirm the discovered knowledge or predicted model behavior. Planning new experiments, changing experimental design, and collecting a new data set (go to the first phase above).

It is not uncommon in bioinformatics to find that models are characterized by small data sets (100 or fewer samples); static data sets, in which data do not change in time once the set was used to create a model; and no need for online adaptation and training on new data. For these tasks, the traditional statistical and artificial intelligence (AI) techniques are well suited. The traditional, offline modeling methods assume that data is static and that no new data are going to be added to the model. Before a model is created, data are analyzed and relevant features are selected again in an offline mode. The offline mode usually requires many iterations of data propagation for estimating the model parameters. Such methods for data analysis and feature extraction use principle component analysis (PCA), correlation analysis, offline clustering techniques (such as K-means, fuzzy C-means, etc.), self-organizing maps (SOMs), and many more techniques. Many modeling techniques are applicable for

these tasks; for example, statistical techniques (regression analysis, support vector machines), AI techniques (decision trees, hidden Markov models, finite automata), and neural-network techniques (MLP, LVQ, fuzzy neural networks).

Some of the modeling techniques allow for extracting knowledge (e.g., rules from the models) that can be used for explanation or knowledge discovery. Such models are the decision trees and the knowledge-based neural networks (KBNNs) [123].

Some of the tasks for data analysis and modeling in bioinformatics are characterized by large data sets, which are updated regularly, a need for online learning and adaptation and online new model creation from input data streams changing with time.

Knowledge adaptation based on a continuous stream of new data.

When creating models of complex processes in molecular biology, the following issues must be considered: how to model complex interactions between genes and proteins—between the human genome and the environment; modeling both stability and repetitiveness, because genes are relatively stable carriers of information; and dealing with uncertainty (when modeling gene expressions, there are many sources of uncertainty; among them are alternative splicing [generation of different mRNA isoforms from a single transcript] and mutation in genes caused by ionizing radiation, chemical contamination, replication errors, viruses that insert genes into host cells, etc). Mutated genes are expressed differently and may cause the production of different proteins. For large data sets and for continuously incoming data streams that require the model and the system to rapidly adapt to new data, it is more appropriate to use online, knowledge-based techniques and Evolving Connectionist System (ECOS) in particular, as it will be demonstrated below.

4.3. Microarray Gene Expression Data Analysis and Disease Profiling

One of the contemporary directions while searching for efficient drugs for many terminal illnesses, such as cancer or HIV, is the creation of gene profiles of these diseases and the subsequent finding of targets for treatment through gene-expression regulation. A gene profile is a pattern of expression of a number of genes that is typical for all, or for some, of the known samples of a particular disease. A disease profile would look like the following.

IF	(gene A is highly expressed)	AND
	...	
	(gene B is low expressed)	AND
	(gene C is very highly expressed)	
THEN	most probably this is cancer of type N	

Having such profiles for a particular disease makes it possible to set early diagnostic tests, so that a sample can be taken from a patient, the data related to the sample can be processed, and a profile obtained. This profile can be matched against existing gene profiles, and based on similarity, it can be predicted with certain probability whether the patient is in an early phase of a disease or whether he or she is at risk of developing the disease in the future with certain probability.

A methodology that consists of training an evolving system and extracting rules, which are presented as disease profiles, is illustrated schematically in Fig. 10. Each profile is a rule extracted from a trained ECOS, which on the figure is shown using colors: The higher the level of a gene expression, the brighter the color. Five profiles are visualized in Fig. 10. The first three represent a group of samples of class 1 (disease), the second two represent class 2 (normal cases). Each column in the condition part of the rules (profiles) represents the expression of one gene out of the 100 relevant genes used in this example.

4.3.1. Gene-Expression Data: A Biological Perspective

Microarray equipment is used widely at present to evaluate the level of gene expression [124]. Each point (pixel, cell) in a microarray represents the level of expression of a single gene. Five principal steps in the microarray technology are shown in Fig. 11. They are tissue collection, RNA extraction, microarray gene-expression calculation, scanning and image processing, and data analysis.

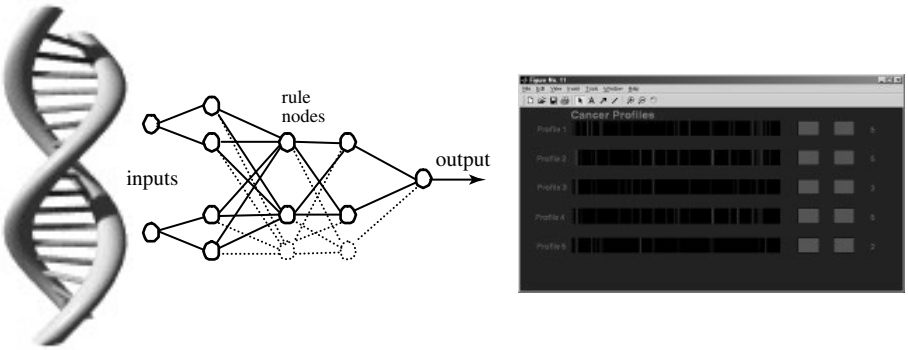

Figure 10. The pathway from DNA to disease profiling: DNA is used to obtain RNA microarray gene expression data; this data is used to train a model (a knowledge-based neural network in this case); profiles (rules) are extracted from the trained model—each rule, presented as one row, represents a gene expression pattern (the "IF" part of the rule) and the class (cancer or normal) associated with this pattern (the "THEN" part of the rule). Reprinted with permission from [15], N. Kasabov, "Evolving Connectionist Systems—Methods and Applications in Bioinformatics, Brain Study and Intelligent Machines." Springer, New York, 2002. © 2002, Springer.

The recent advent of cDNA microarray and genechip technologies means that it is now possible to simultaneously interrogate thousands of genes. The potential applications of this technology are numerous and include identifying markers for classification, diagnosis, disease outcome prediction, therapeutic responsiveness, and target identification. Microarray analysis might not identify unique markers (e.g., a single gene) of clinical utility for a disease because of the heterogeneity of the disease, but a prediction of the biological state of disease is likely to be more sensitive when identifying clusters of gene expression (profiles) [125].

For example, gene-expression clustering has been used to distinguish normal colon samples from tumors from within a 6500-gene set, although clustering according to clinical parameters was not undertaken [126]. Although distinction between normal and tumor tissue can be easily made using microscopy, this analysis represented one of the early attempts to classify biological samples through gene-expression clustering. The above data set is used in this section to extract profiles of colon cancer and normal tissue, using an EFuNN [127]. Another example of profiling is determining the distinction between two subtypes of leukemia; namely, acute myeloid leukemia (AML) and acute lymphoblastic leukemia (ALL) [128].

There are several traditional approaches to analyzing gene regulatory networks: logical or binary, chemical kinetic, stochastic kinetic, and so forth. A detailed discussion of these methods can be found in Refs. [118, 129–135].

Neural networks have already been used to create classification systems based on gene expression data. In Ref. [136], multilayer perceptron neural networks were used to achieve a classification of 93% of Ewings sarcomas, 96% of rhabdomyosarcomas, and 100% of neuroblastomas. From within a set of 6567 genes, 96 genes were used as variables in the classification system. Whether these results would be different using different classification methods needs further exploration.

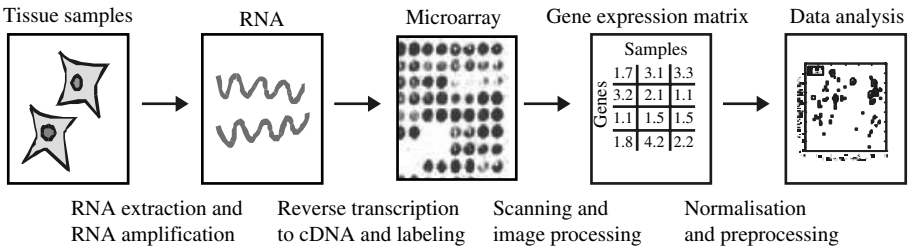

Figure 11. The pathway of the microarray technology: from DNA data to data analysis results.

A methodology for profiling of gene expression data from microarrays is described in Ref. [15]. It consists of the following phases:

1. Microarray data preprocessing: This phase aims at eliminating the low expressed genes, or genes that are not expressed sufficiently across the classes (e.g., controlled vs. tumor samples, or metastatic vs. nonmetastatic tumors, etc). Log transformation of the data can be applied to reduce the range of gene expression data. An example of how this transformation "squeezes" the gene expression values plotted in the two-dimensional principal components is given in Fig. 12. There are only two samples used (two cell lines) and only 150 genes, out of the 4000 on the microarray, that distinguish these samples.
2. Selecting a set of significant differentially expressed genes across the classes: Usually, the t-test is applied at this stage, with an appropriate threshold [137].
3. Finding subsets of both underexpressed and overexpressed genes from the selected ones in the previous step: Statistical analysis of these subsets is performed.
4. Clustering of the gene sets from step 3 that would reveal preliminary profiles of jointly overexpressed or underexpressed genes across the classes. An example of the hierarchical clustering of 12 microarray vectors (samples), each containing the expression of 50 genes after steps 1 to 3 were applied on the initial 4000-gene-expression data from the microarrays, is given in Fig. 13 [(a) samples in two-dimensional Sammon's projection space of the 50-dimensional gene expression space; (b) the similarity between the samples (columns), based on the 50 selected genes, and the similarity between the genes (rows) based on their expression in the 12 samples].
5. Building a classification model and extracting rules that define the profiles for each class: The rules would represent the fine grades of the common expression level of groups of genes. Through using thresholds, smaller or larger groups of genes can be selected from the profile.
6. Further training of the model on new data and updating the profiles: With the arrival of new labeled data (samples), the model needs to be updated (e.g., trained on additional data) and possibly have modified rules (profiles) extracted.

Two data sets are used here to illustrate the above methodology, which explores evolving systems for microarray data analysis.

4.3.2. Case Study: Gene Profiling of Two Classes of Leukemia

A data set of 72 classification examples for leukemia is used that consists of two classes and a large input space—the expression values of 6817 genes monitored by Affymetrix arrays [128]. The two types of leukemia are AML and ALL. The latter type can be subdivided further into

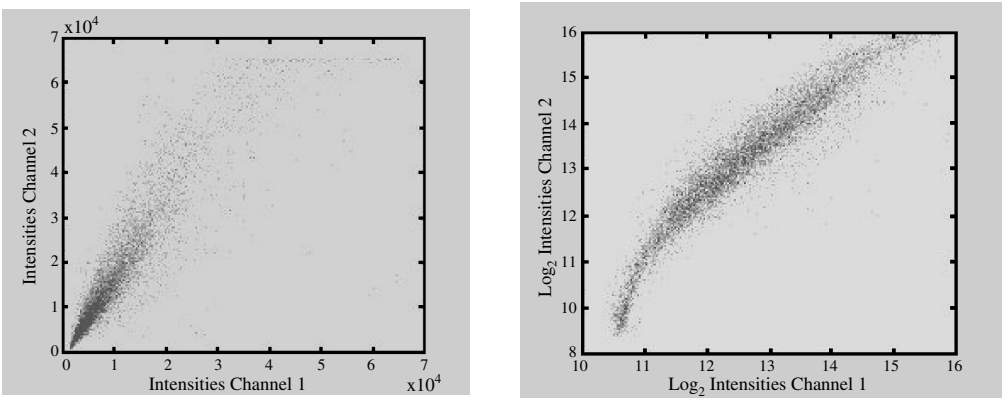

Figure 12. An illustration of the importance of the preprocessing phase in microarray gene expression data analysis on two channel expression measurements taken from the same tissue. Each spot is the expression of a gene. It can be seen from the figures on the left and on the right, respectively, that using log transformation of the values makes the values from the two channels more similar to each other (closer to the desirable diagonal perfect match line).

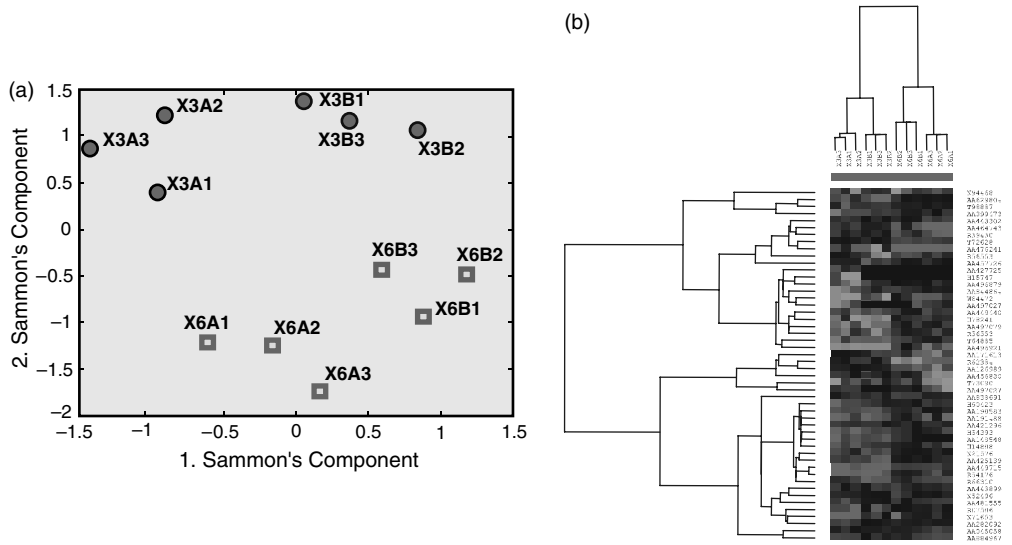

Figure 13. (a) Sammon's projection of 50 gene-expression variables of 12 gene-expression vectors (samples, taken from 12 tissue samples). (b) Hierarchical clustering of the data. The rows are labeled by the gene names, and the columns represent different samples. The lines link similar items (similarity is measured as correlation) in a hierarchical fashion.

T-cell and B-cell lineage classes. In Ref. [128], the data set was split into 38 cases (27 ALL and 11 AML) for training and 34 cases (20 ALL and 14 AML) for validation of a classifier system. The test set shows a higher heterogeneity with regard to tissue and age of patients, making any classification more difficult. So, the tasks are to, find a set of genes distinguishing ALL and AML, construct a classifier based on these data, and find a gene profile of each of the classes.

After having applied points 1 and 2, 100 genes were selected. A preliminary analysis on the separability of the two classes can be done through plotting the 72 samples in the two-dimensional PCA space. PCA consists of a linear transformation from the original set of variables (100 genes) to a new (smaller, two-dimensional) set of orthogonal variables (principal components) so that the variance of the data is maximal and ordered according to the principal components (see Fig. 14a).

The extracted rules for each class make up a profile of this class. One way of visually representing of these profiles is illustrated in Fig. 14b, where rules were extracted from a trained EFuNN with 100 genes.

To choose the model for gene-profiling and classification tasks, the gene-profiling task requires that the model meets the following requirements: it can be continuously trained on

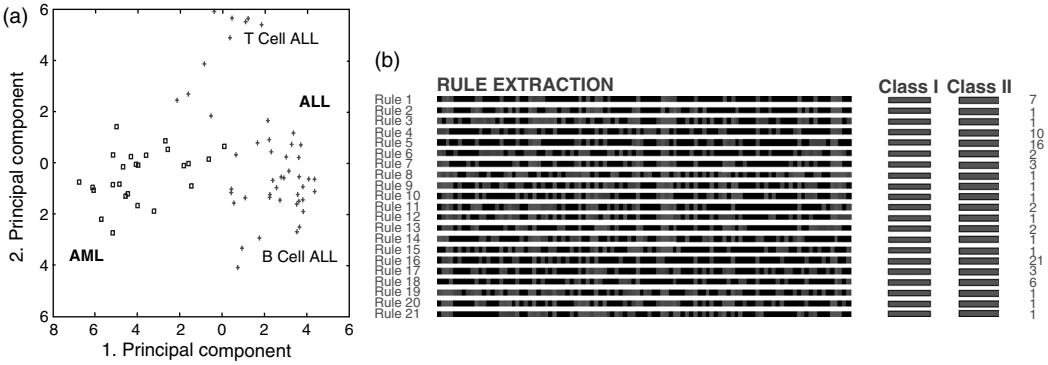

Figure 14. (a) The first two principal components of the leukemia 100 genes selected after a *t*-test is applied. (b) Some of the rules extracted from an Evolving Fuzzy Neural Network trained on the leukemia data and visualized as profile patterns (see Ref. [15]). AML, acute myeloid leukemia; ALL, acute lymphoblastic leukemia.

new data, the knowledge-based model is extracted (knowledge in the form of profiles), and the model gives an evaluation for the validity of the profiles.

4.4. Clustering the Time-Course Gene-Expression Data

The expression of genes in cell changes with time. Measuring the expression rate of each gene over time gives the gene a temporal profile of its expression level. Genes can be grouped together according to their similarity of temporal expression profiles.

This is illustrated here using case study data. To demonstrate the method, we used yeast gene-expression data that are available in public databases. We analyzed the gene expression during the mitotic cell cycle of different synchronized cultures as reported by [138] and by [139]. The data sets consisted of expression profiles for over 6100 open reading frames (ORFs).

In this study we did not reduce the original data set by applying a filter in form of a minimum variance. This leads to a higher number of clusters of weakly regulated genes, however, it diminished the possibility of missing co-regulated genes during the clustering.

To find upstream regulatory sequences we used Hughes' compiled set of upstream regions for the ORFs in yeast (Church lab, http://atlas.med.harvard.edu/).

One of the reasons for cluster analysis of time-course gene-expression data is to infer the function of novel genes by grouping them with genes of well-known functionality. This is based on the observation that genes that show similar activity patterns over time (coexpressed genes) are often functionally related and controlled by the same mechanisms of regulation (coregulated genes). The gene clusters generated by cluster analysis often relate to certain functions (e.g., DNA replication or protein synthesis). If a novel gene of unknown function falls into such a cluster, it is likely that this gene serves the same function as the other members of this cluster. This "guilt-by-association" method makes it possible to assign functions to a large number of novel genes by finding groups of coexpressed genes across a microarray experiment [140].

Different clustering algorithms have been applied to the analysis of time-course gene-expression data; k-means, SOM, and hierarchical clustering, to name just a few [140]. They all assign genes to clusters based on the similarity of their activity patterns. Genes with similar activity patterns should be grouped together, whereas genes with different activation patterns should be placed in distinct clusters. The cluster methods used so far have been restricted to a one-to-one mapping: one gene belongs to exactly one cluster. Although this principle seems reasonable in many fields of cluster analysis, it might be too limited for the study of microarray time-course gene-expression data. Genes can participate in different genetic networks and are frequently coordinated by a variety of regulatory mechanisms. For the analysis of microarray data, we may therefore expect that single genes can belong to several clusters. Several researchers have noted that genes were frequently highly correlated with multiple classes and that the definition of clear borders between gene-expression clusters often seemed arbitrary [141]. This is a motivation to use fuzzy clustering to assign single objects to several clusters.

A second reason for applying fuzzy clustering is the large noise component in microarray data resulting from biological and experimental factors. The activity of genes can show large variations under minor changes of the experimental conditions. Numerous steps in the experimental procedure contribute to additional noise and bias. A usual procedure to reduce the noise in microarray data is setting a threshold for a minimum variance of the abundance of a gene. Genes below this threshold are excluded from further analysis. However, the exact value of the threshold remains arbitrary because of the lack of an established error model and the use of filtering as preprocessing.

Because we usually have little information about the data structure in advance, a crucial step in the cluster analysis is selection of the number of clusters. Finding the "correct" number of clusters addresses the issue of cluster validity. This has turned out to be a rather difficult problem, as it depends on the definition of a cluster. Without prior information, a common method is the comparison of partitions resulting from different numbers of clusters. For assessing the validity of the partitions, several cluster validity functionals have been introduced [142]. These functionals should reach an optimum if the correct number of clusters

is chosen. When using evolving clustering techniques, the number of clusters does not need to be defined *a priori*.

In Ref. [15], an ESOM is evolved from the yeast gene temporal profiles used as input vectors. The number of clusters did not need to be specified in advance (Fig. 15).

It can be seen from Fig. 15 that clusters 72 and 70 are represented on the ESOM as neighboring nodes. The ESOM on the figure is plotted as a two-dimensional PCA projection. Cluster 72 has 43 members (genes that have similar temporal profiles), cluster 70 has 61 members, and cluster 5 has only three genes as cluster members. New cluster vectors can be created in an online mode if the distance between existing clusters and the new data vectors are above a chosen threshold.

5. COMPUTATIONAL SYSTEMS BIOLOGY

5.1. Introduction

The aim of computational systems biology is to understand complex biological objects in their entirety (i.e., at system level). It involves the integration of different approaches and tools: computer modeling, large-scale data analysis, and biological experimentation. One of the major challenges of the systems biology is the identification of the logic and dynamics of gene-regulatory and biochemical networks.

In Ref. [143], general systems theory was applied to biology, psychology, economics, and social science. In the view of this work, old-fashioned science "tried to explain observable phenomena by reducing them to an interplay of elementary units investigatable independently of each other." However, contemporary science empowers the importance of "wholeness," which can be defined as "problems of organization, phenomena not resolvable into local events, dynamic interactions manifest in the difference of behavior of parts when isolated or in higher configuration, etc.; in short, 'systems' of various orders not understandable by investigation of their respective parts in isolation." This remains an effective

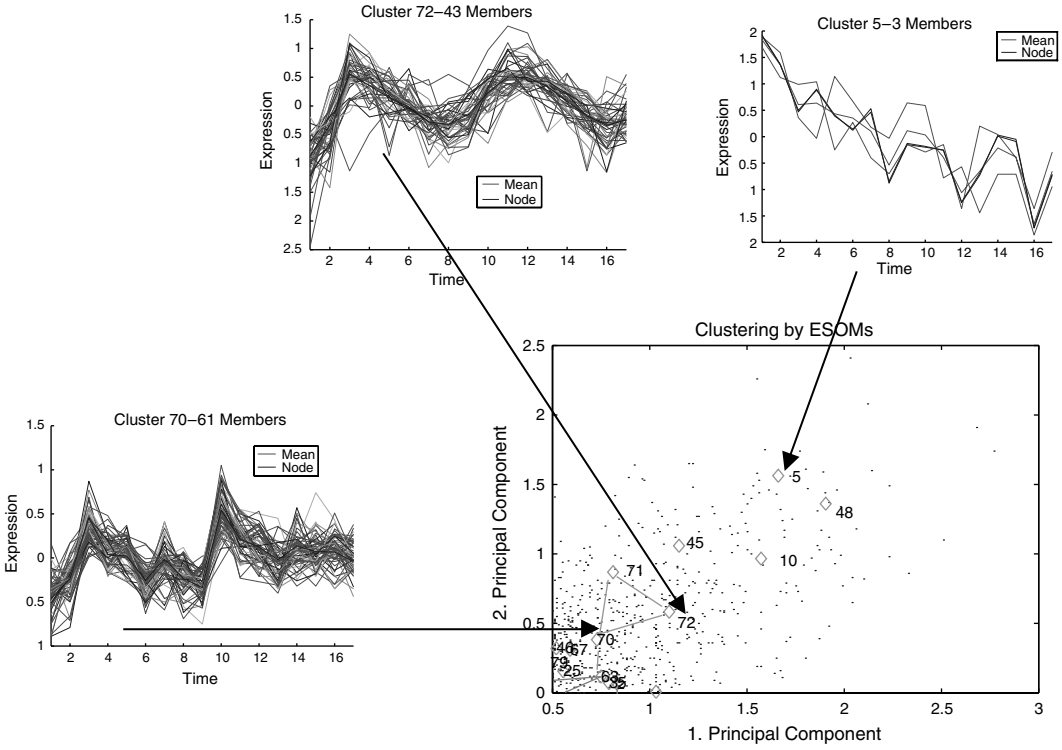

Figure 15. Using a clustering model, an evolving self-organized maps (ESOM) to cluster time-course yeast gene expression data (see Ref. [15]). Genes with similar time-course profiles are gathered in a same cluster.

definition of systems biology as practiced today, with the integration and application of mathematics, engineering, physics, and computer science to understanding a range of complex biological regulatory systems.

Computational cell biology is the most immediate beneficiary of the flood of the data that has emerged from the genomics and proteomics that are the main sources of enormously large data sets providing information about genes and proteins interaction. The main goal of systems biology is to integrate all these data in the whole, providing explanation and prediction of behavior of the system under study. The most feasible application of systems biology is to create a detailed model of cell regulation to provide system-level insights into mechanism-based drug discoveries [144–146]. The main subjects of computational systems biology are the following [147]: the structure of the systems, such as genes, metabolism, and signal transduction networks and physical structures; the dynamics of such systems; methods to control the systems; and methods to design and modify the systems for desired properties.

5.2. System-Level Understanding

System–level understanding is a recurrent theme in biology and has a long history [148–150]. The term "system–level understanding" was described in Ref. [147] as the shift of focus in understanding a system's structure and dynamics as a whole, rather than as the particular objects and their interactions. The property of a whole biological system, like a cell, cannot be understood by drawing a diagram of interconnection of the genes and proteins—this can give us only a static picture of the dynamics. System-level understanding of a biological system, however, can be derived from insight into four key properties [117]:

1. System structures: These include the gene regulatory network and biochemical pathways. They can also include the mechanisms of modulation for the physical properties of intracellular and multicellular structures by interaction.
2. System dynamics: System behavior over time under various conditions can be understood by identifying essential mechanisms underlying specific behaviors and through various approaches depending on the systems nature: metabolic analysis (finding a basis of elementary flux modes that describe the dominant reaction pathways within the network), sensitivity analysis (the study of how the variation in the output of a model can be apportioned, qualitatively or quantitatively, to different sources of variation), dynamic analysis methods such as phase portrait (geometry of the trajectories of the system in state space), and bifurcation analysis (bifurcation analysis traces time-varying changes in the state of the system in a multidimensional space, where each dimension represents a particular system parameter—concentration of the biochemical factor involved, rate of reactions/interactions, etc). As parameters vary, changes may occur in the qualitative structure of the solutions for certain parameter values. These changes are called bifurcations, and the parameter values are called bifurcation values.
3. The control method: Mechanisms that systematically control the state of the cell can be modulated to change system behavior and optimize potential therapeutic effect targets of the treatment.
4. The design method: Strategies to modify and construct biological systems having desired properties can be devised based on definite design principles and simulations, instead of blind trial-and-error.

As mentioned above, in reality, analysis of system dynamics and understanding the system structure are overlapping processes. In some cases, analysis of the system dynamics can give useful predictions in system structure (new interactions, additional members of the system). Different methods can be used to study the dynamical properties of the system: analysis of steady states allows us to find the systems states when there are no dynamical changes in system components; stability and sensitivity analyses provide insights into how systems behavior changes when stimuli and rate constants are modified to reflect dynamic behavior; bifurcation analysis, in which a dynamic simulator is coupled with analysis tools, can provide a detailed illustration of dynamic behavior [151, 152]; and flux balance analysis [153] can be used to predict the different metabolic patterns, as was done, for instance,

in [154] for predicting the switching in of the metabolic pathways in *Escherichia coli* under different nutritional conditions, based on knowledge of only the metabolic network structure. The choice of the analytical methods depends on the availability of the data that can be incorporated into the model, and on the nature of the model.

5.3. Properties of the Complex System

Robustness is a central issue in all complex systems, and it is essential for understanding the biological object functioning at the system level. Robust behavior in biochemical networks was reported a long time ago in Refs. [155, 156], as well as in more recent papers [157–160].

Robustness can be defined as the preservation of particular characteristics despite uncertainty in components or the environment [161]. Robust systems exhibit the following phenomenological properties [117]: adaptation, which denotes the ability to cope with environmental changes; parameter insensitivity, which indicates a system's relative insensitivity (to a certain extent) to specific kinetic parameters; and graceful degradation, which reflects the characteristic slow degradation of a system's functions after damage, rather than catastrophic failure.

These properties correspond to the following properties attained to robust systems in engineering: a form of system control such as negative feedback and feed-forward control; redundancy, whereby multiple components with equivalent functions are introduced for backup; and structural stability, where intrinsic mechanisms are built to promote stability and modularity, and where subsystems are physically or functionally insulated so that failure in one module does not spread to other parts and lead to system-wide catastrophe.

Conducting system-level analysis requires a comprehensive set of experimental data. Comprehensiveness in measurements requires consideration of three aspects [117]: first, factor comprehensiveness, which reflects the number of variables that can be measured at once; second, time-line comprehensiveness, which represents the time frame within which measurements are made; and third, item comprehensiveness, which refers to the simultaneous measurement of multiple items (instance, concentrations, localization, etc.)

Some systems may have a property of "spiraling complexity," which means that each module in the system that provides some advantage can lead also to the fragility of the system. To overcome a new threat, it is necessary to build a new module/subsystem that can also lead to the new failure/fragility. This evolution leads to an excess of complexity for the system, which in its turn leads to the robustness of the system.

The main feature of the concepts in evolutionary biology, the converged evolution, is that it leads to nearly optimal systems with similar gross characteristics, so simple arguments based on optimal design can explain functional relations between variables across many scales [162, 163]. Three other key elements (discovered with the use of computational modeling and experimentation) of the organizational principles used by cells are noted in Refs. [116, 132, 164].

1. Ultrasensivity: A response that is more sensitive to ligand concentration as compared to standard responses defined by the Michaelis–Menten equations [165–167];
2. Multistability: An existence of two or more stable states for the regulating network [168, 169];
3. Rhythmic behavior, functioning as a systemic oscillator: In Ref. [170], the gene regulatory network with this property was described; three transcriptional repressors were used to build an oscillating network in *E. coli*.

More general principles that seem to be necessary for the operation of a living system (and peculiar to the complex biological systems) were presented in Ref. [171]:

- Program: plan describing ingredients and interactions between them as living system persist through time.
- Improvisation: the ability to change the program with respect to changes in environment.
- Compartmentalization: division of the whole organisms into smaller compartments to centralize and specialize certain functions.

- Energy: living organism is an open system metabolizing energy.
- Regeneration: resynthesis of the constituents of the system.
- Adaptability: fast response that allows survival in quickly changing environments.
- Seclusion: the ability to allow thousands of reactions to occur with high efficiency in the tiny volume of living cells.

5.4. Representation of Gene-Regulatory and Biochemical Networks

Theoretically, each system can be described as a set of modules and protocols. Modules are components or subsystems of larger system that may have the following features [161]: possess identifiable interfaces to other modules, can be modified and evolved somewhat independently, facilitate simplified or abstract modeling, maintain some identity when isolated or rearranged, derive additional identity from the rest of the system.

The protocol is the set of rules describing and managing the relationship between modules and subsystems and processes in the system. It allow interfaces between modules and permit system functions. The set of protocols for a particular system can include such relationships between modules and components as activation, inhibition, feedback and feed-forward controls, and so forth.

Developing software for building networks and simulating experiments with the use of standardized technology and common infrastructure is important for systems biology. To solve the problem of software interoperability, two related packages were developed [172]: ERATO Systems Biology Workbench (SBW; a modular, broker-based message-passing framework for simplified interconnection between applications), and the Systems Biology Markup Language (SBML; open, extensible markup language (XML) based format for representing biochemical reaction networks). Initially, the SBWs focus was to provide interoperability for the following existing simulation tools: BioSpice [173], DBSolve [174, 175], e-Cell [176, 177], Gepasi [178, 179], Jarnac [180, 181], StochSim [182, 183], and Virtual Cell [184, 185].

SMBL is an XML-based language. XML [186] and originally it was designed to meet the challenges of large-scale electronic publishing (http://www.w3.org/XML/) and it is a dialect of the Standard Generalized Markup Language. The two draft versions of the SMBL were developed and released by the Caltech ERATO team in 2000, and the base-level definition of SBML was delivered in March 2001. Model definition in SMBL consists of the following components [187]:

- Unit definition: A name for a unit used in the expression of quantities in a model. Units may be supplied in a number of contexts in an SBML model, and it is convenient to have a facility both for setting default units and for allowing combinations of units to be given abbreviated names.
- Compartment: A container of finite volume for substances, in which reactions take place. Compartments do not necessarily have to correspond to actual structures inside or outside of a cell.
- Specie: A substance or entity that takes part in a reaction. Some example species are ions such as Ca^{2+} and molecules such as glucose or adenosine triphosphate. The primary qualities associated with specie are its initial amount and the compartment in which it is located.
- Parameter: A quantity that has a symbolic name; this name can be used in formulas in place of the value. Parameters can be global to a model or local to a single reaction.
- Reaction: A statement describing some transformation, transport or binding process that can change the amount of one or more species. For example, a reaction may describe how certain entities (reactants) are transformed into certain other entities (products). Reactions have associated rate laws describing how quickly they take place. Reactions are defined using lists of reactant species and products, their stoichiometric coefficients, and kinetic rate laws.
- Rule: A mathematical expression that is added to the differential equations constructed from the set of reactions and that can be used to set parameter values, establish constraints between quantities, and so forth.

Models of different complexity can be written in SMBL and then read by software packages and translated to an internal format. After that, one can model the dynamics, study the model behavior, and represent the results in plots.

CellML language is close to SMBL and is an open and XML-based standard [188]. CellML is being developed by the Bioengineering Institute, University of Auckland (http://www.cellml.org). It was designed for storing and exchanging biological simulation models. Two other projects are closely affiliated to CellML: AnatML, for exchanging information at the organ level—it can be used to store geometric information and documentation that was generated during a skeleton digitization project; and FieldML, to provide a description of spatially and temporally varying field information using finite elements—it is appropriate for storing geometry information inside AnatML, for spatial distribution of parameters inside compartments in CellML, or for spatial distribution of cellular model parameters across an entire organ.

Together, these XML-based technologies provide a complete vocabulary for describing "virtual" biological systems from the cellular to the organism level.

Various attempts were made to standardize the graphical representation of the biochemical and gene networks [189–193]. In Ref. [193], the following requirements for graphical notation system were formulated:

- Expressiveness: The ability to describe every possible relationship between objects.
- Semantically unambiguous: Different semantics should be assigned to different symbols.
- Visually unambiguous: Symbol should be clearly identified and not be mistaken for other symbols.
- Extension capability: The notation system should be easily extended.
- Mathematical translation: Availability to be directly applied for numerical analysis.
- Software support: Support of notations by software for drawing, viewing, editing, and translation into mathematical formalism.

To support the graphical notation system proposed in this work, a new process diagram editor (Cell Designer) for gene-regulatory and biochemical network was developed [194].

A research program aimed at creating a framework, experimental infrastructure, and computational environment for understanding, experimenting with, manipulating, and modifying a diverse set of fundamental biological processes at multiple scales and spatiotemporal modes is described in Ref. [195]. From a biological viewpoint, the basic issues of these projects are understanding common and shared structural motifs among biological processes, modeling biological noise resulting from interactions among a small number of key molecules or loss of synchrony, explaining the robustness of these systems in spite of such noise, and cataloging multistatic behavior and adaptation exhibited by many biological processes.

5.5. Artificial Life

5.5.1. E-Cell

Several other projects that aimed at computer modeling of the cell should be noted. The first one is the E-Cell Project—an international research project aiming to model and reconstruct biological phenomena *in silico* and developing necessary theoretical supports, technologies, and software platforms to allow precise whole-cell simulation (www.e-cell.org). This project started in 1996 and led first to the design and development of the first working version of the E-Cell simulation environment in 1996. Then, the self-sustaining cell model was constructed by abstracting the gene set of *Mycoplasma genitalium*—the smallest known genome whose complete 580 kb genome sequence was determined in 1995. Next, an attempt was made to model real cells and to develop a more sophisticated simulation environment for biological simulations, and new modeling projects for modeling a human erythrocyte, mitochondrion, *E. coli chemotaxis*, and gene expression/replication were run. A list of publications and a Windows version of the software (E-Cell, version 2) can be found in Web site listed earlier.

5.5.2. Virtual Cell

Virtual Cell (National Resource for Cell Analysis and Modeling, http://www.nrcam.uchc.edu/index.html) is another project that is aimed at providing a remote-user modeling and simulation environment using Java's Remote Method Invocation (RMI). The Virtual

Cell provides a formal framework for modeling biochemical, electrophysiological, and transport phenomena and considers localization in cell of the molecules that take part in these reactions [196]. This localization can take the form of a three-dimensional arbitrarily shaped cell; molecular species might be heterogeneously distributed in the cell. The geometry of the cell, including the locations and shapes of subcellular organelles, can be imported directly from microscope images. Such a model considers the diffusion of the molecules within the geometry. Users can create biological models of various types and run simulations on a remote server. A general-purpose solver is used to translate the initial biological description into a set of concise mathematical problems. The generated results can be reviewed in the software or exported in a variety of popular formats.

5.5.3. GEPASI

GEPASI (http://www.gepasi.org) is a software package intended for modeling biochemical systems [197, 198]. With the help of this package, one can simulate the kinetics of systems of biochemical reactions as well as fit models to data, optimize functions of the model, and perform metabolic control analysis and linear stability analysis. GEPASI simplifies the task of model building with its user-friendly interface and helps to translate the language of chemistry (reactions) to that of mathematics (matrices and differential equations). This package uses a set of sophisticated numerical algorithms that ensure that the results obtained are fast and accurate [199]. GEPASI is intended primarily for research purpose, but it also can be used for education.

5.5.4. In Silico Cell

In Silico Cell architecture supports the hierarchical modeling of biological system, and the creation of more complex models from simpler ones. In Silico Cell allows researchers to interface with the technology in a fashion that is most intuitive to their particular scientific background. This process is enabled by the use of CellML, an XML-based markup language for describing biological processes at the cellular and subcellular levels.

5.6. Computational System Biology: Modeling Issues

Tomita stated in his paper [176] that, "the cell is never conquered until its total behavior is understood, and the total behavior of the cell is never understood until it is modeled and simulated."

Modeling living cells *in silico* (in a computer) has many implications, one of which is testing new drugs through simulation rather than on patients. According to Ref. [200], human trials fail for 70–75% of the drugs that enter them.

Computer modeling of the processes in living cells is an extremely difficult task. There are several reasons for that, including that the processes in a cell are dynamic and depend on many variables, some of which are related to a changing environment; and that the processes of DNA transcription and protein translation are not fully understood.

A starting point for dynamic modeling of a cell would be dynamic modeling of a single gene-regulation process. In Ref. [201] the following methods for single-gene-regulation modeling are discussed, taking into account different aspects of the processes (chemical reactions, physical chemistry, kinetic changes of states, and thermodynamics): Boolean models, based on Boolean logic (true/false logic); differential equation models; stochastic models; hybrid Boolean/differential equation models; hybrid differential equations/stochastic models; neural network models; and hybrid connectionist-statistical models.

Some of these methods are briefly described below.

5.6.1. Boolean Models

Consider a set of N objects at time t_k $\{x_1^k, x_2^k, \ldots, x_N^k\}$, and each object can be in only two different states: on/off, 1/0, False/True, and so forth. For simplicity, let us assume

$$x_i^k \in \{0, 1\} \quad i = 1, \ldots, N$$

The state of the system at a given moment of time can be described as the states of all objects in this set. The state of a given object at the next time step t_{k+1} can be determined by a Boolean logic function (returning only two values, 0 or 1) whose input is the current state of the system.

$$x_i^{k+1} = B_i(x_1^k, x_2^k, \ldots, x_N^k)$$

Boolean function $B = \{B_1, B_2, \ldots, B_N\}$ can be represented as a truth table that consists of all possible system states (2^N) and corresponding states calculated using Boolean function. Because this function represents relations between all systems' states, it can be easily represented as diagram.

As an example, let us consider a system of two genes, A and B, which can be expressed (1) or not expressed (0), and a particular regulatory network that can be described as the following truth table.

All Possible System States	Next State
00	01
01	10
10	11
11	11

The graphical representation of the network is in Fig. 16.

One can see that state 11 in this diagram is stable (i.e., it leads to no change in system states). The Boolean function that corresponds to the truth table and diagram is

$$A^{k+1} = A^k | B^k$$
$$B^{k+1} = A^k | \neg B^k$$

where: | is logical OR and ¬ is logical NOT.

5.6.2. Kinetic Logic Models

This type of model is the extension of the Boolean one: each gene has finite number L of discrete values of states

$$x_i^k \in \{X_1, X_2, \ldots, X_L\} \quad i = 1, \ldots, N$$

So, for each gene, Boolean function should return one of the L possible values. In addition, genes may have different rates of changing their states. This type of relations is described by the more sophisticated function.

For the example shown above, let us assume that genes A and B can be not expressed, 0; expressed at low level, 1; or expressed, 2. Therefore, the total number of possible system states is nine, and one of the possible representations of the system is shown in Fig. 17.

For some models of this type, objects may have a different number of discrete values of states ($L = L_i$); moreover, there may be more than one possible next state for the

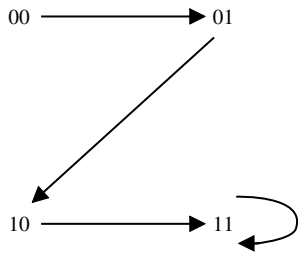

Figure 16. Diagram representation of the Boolean network for the set of two genes.

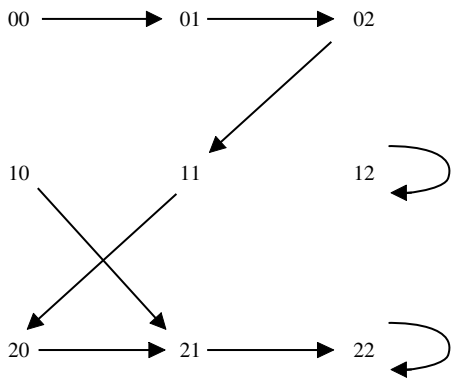

Figure 17. Diagram representation of the kinetic logic model for the set of two genes and three states for gene expression. System has two stable states ("12" and "22").

object in the system, and objects may change their states asynchronously. All these features make kinetic logic models more complex but also more tied to the biological system being modeled.

5.6.3. Ordinary Differential Equation Models

In their turn, differential equation models are the extension of the kinetic logic model. They are used usually to deal with continuous values characterizing the system state. Time is also continuous for these models, and changing of object states with time can be written as

$$\frac{dx_i}{dt} = F_i(t, x) \quad i = 1, \ldots, N$$

where $x = x(t) = [x_1(t), x_2(t), \ldots, x_N(t)]$. In general, the right-hand side of the ordinary differential equation of the Mth order can depend on derivatives of the different orders of all objects' states

$$F\left(t, x, \frac{dx}{dt}, \frac{d^2x}{dt^2}, \ldots, \frac{d^Mx}{dt^M}\right) = 0 \quad i = 1, \ldots, N$$

To calculate system dynamics, one should provide initial condition of the system (values for all object states at starting time point $t = t_0$)

$$x(t_0) = X^0$$

The above-mentioned general differential equation is said to be linear if F is a linear function of the variables x, $\frac{dx}{dt}$, $\frac{d^2x}{dt^2}$, $\ldots$, $\frac{d^Mx}{dt^M}$ and can be read as

$$a_M(t)x + a_{M-1}(t)\frac{dx}{dt} + \cdots + a_0(t)\frac{d^Mx}{dt^M} = f(t)$$

where: $a_i(t)$, $i = 1, \ldots, M$ and $f(t)$ some functions.

When the right-hand function is complex and does not allow an analytical solution, the system dynamics can be calculated numerically. There are many tools for solving this problem.

The equilibrium state of the system corresponds to the system state when there are no changes in all states with time. Therefore, the condition of equilibrium state can be found by solving the system of equations

$$F_i(t, x) = 0 \quad i = 1, \ldots, N$$

For instance, the simple model of exponential loss of the gene A expression with time can be expressed as

$$\frac{dA}{dt} = -kA \quad A(0) = A^0$$

where k is the rate of loss of gene expression with time and A^0 is the initial expression of gene A at time $t = 0$. For this simple equation, a solution can be found easily

$$A(t) = A_0 \exp(-kt)$$

and the equilibrium state for this system is trivial: $\overline{A} = 0$.

5.6.4. Partial Differential Equation Models

Partial differential equations can be used for the description of the system states if they change not only with time but with respect to other parameters (e.g., the object's size and location). As an example, let us describe the diffusion of expression of a particular gene in a cell. The equation can be read as follows

$$\frac{dA}{dt} = \alpha \frac{d^2A}{ds^2}$$

Here $A = A(t, s)$ is gene expression distributed in cells with respect to time t ($t \geq 0$) and coordinate s, $0 \leq s \leq S$. Having initial distribution of the expression of this gene in the cell

$$A(0, s) = g(s)$$

and boundary conditions

$$A(t, 0) = p_0(t) \quad A(t, S) = p_S(t)$$

where $g(s)$, $p_0(t)$, and $p_S(t)$ are given functions, one can calculate $A(t, s)$ at any time and coordinate point.

5.6.5. Stochastic Models

Stochastic models deal with the dynamic history of each object of the model. In other words, for each object, the next state must be calculated using the set of probabilistic rules. Each rule shows the probability of the object being changed in a particular interval of time, and the probability of it coming to each state. Therefore, the change of state in this type of model is probabilistic, not deterministic.

Let us assume that object x in the system has a finite state space with L states (as in the kinetic logic model): $\{X_1, X_2, \ldots, X_L\}$. For each time step t_{k+1} there is a transition probability $P(x_{k+1}|x_0, \ldots, x_k)$; and chain $x_0, \ldots, x_k$ represents the history of the system. Variables x_k form a Markov chain if and only if for any k

$$P(x_{k+1}|x_0, \ldots, x_k) = P(x_{k+1}|x_k)$$

In other words, the future state depends on the only present state. All probability values [$P(X_i|X_j)$, the probability of the system jumping from the ith to the jth state] form a transition matrix.

Suppose that the system can jump into state X_i at time t_k with transition rate λ_k^i. After calculating the probability of coming into the ith state at time t_k

$$p_k^i = \frac{\lambda_k^i}{\lambda} \quad \lambda = \sum_{i=1}^{L} \lambda_k^i$$

one can easily calculate the next state of the system. The formula for calculating the next time point depends on the distribution of the jumps $t_{k+1} - t_k$; for instance, for the case of the exponential process, the next time point is $t_{k+1} = t_k - \ln(r)/\lambda$, where r is a random value uniformly distributed in (0, 1).

Figure 18 shows the examples of dynamics calculated for the model of exponential decay for differential equation and stochastic models.

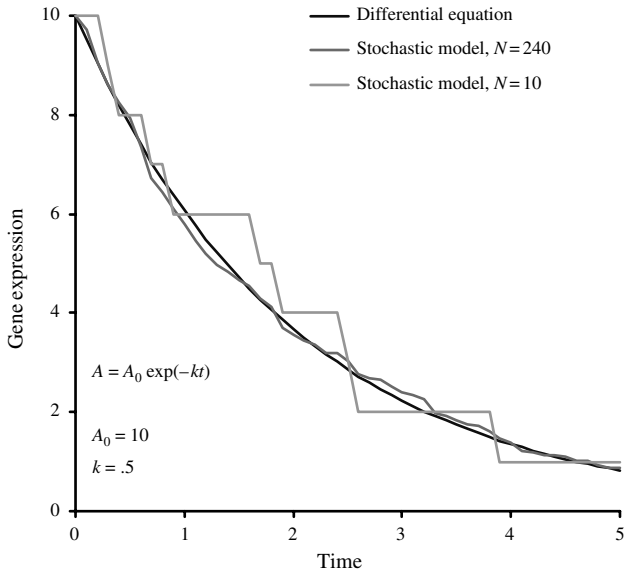

Figure 18. Examples of exponential decay calculated with the use of a differential equation and stochastic model. Two numbers of objects (N) were used for the stochastic model. One can see that with the increase of N, the solution of the stochastic model is the approximate better solution of the differential equation model.

5.6.6. Neural Network Models

Neural networks provide a model of computation that is different from traditional algorithms. Typically, they are not explicitly programmed to perform a given task; rather, they learn to do the task from examples of desired input/output behavior. The networks automatically generalize their processing knowledge into previously unseen situations, and they perform well for the noisy, incomplete, or inaccurate input data.

In general, the artificial neural network is a model consisting of interconnected units evolving in time. Connection between units i and j is usually characterized by the weight, denoted by w_{ij}. There are three important architectures of the neural network, based on the connectivity: recurrent (contains direct loops), feed-forward (contains no direct loops); and layered (units are organized into layers, and connections are between layers).

The behavior of each unit in time can be described by the time-dependent functions, stochastical process, Bayesian network, and so forth. Therefore, the ith unit receives total input x_i from the units connected to it and generates the response

$$f_i(x_i) = \sum_{j \in C} w_{ij} y_j$$

where C is the set of units having connection to the ith unit. When the response is represented as a threshold function

$$f(x) = \begin{cases} 1 & x > 0 \\ 0 & x \leq 0 \end{cases}$$

the unit is called a threshold gate and can generate only binary decisions.

The most usual application of the neural network is classification (to arrange input into number of classes). The most important feature of the neural network is learning for examples. It consists of the model fitting and parameter estimation, with the use of the training subset of input data, and validation of the model output, with the use of the validation subset.

5.6.7. Hybrid Models

Some models can combine features of different types of models. Among these models are

- Boolean/differential-equation models: Parameters of the differential equation can depend on discrete system states.

- Differential equations/stochastic models: Introduction in the right-hand side of the differential equations' statistical terms describing, for example, the noise in the system values. It should be noted that some differential models can show stochastic behavior for some values of the parameters. For instance, the Mackey–Glass differential equation

$$\frac{dx}{dt} = \frac{ax(t-\tau)}{1+x^{10}(t-\tau)} - bx$$

  behaves as a chaotic time series when the parameters have the following values: $x(0) = 1.2$, $\tau = 17$, $a = 0.2$, $b = 0.1$, and $x(t) = 0$, for $t < 0$.

The next step in dynamic cell modeling would be to try and model the regulation of more genes, and hopefully a large set of genes (see Ref. [202]). Patterns of collective regulation of genes, such as chaotic attractors, are observed in the above reference. Mutual information/entropy of clusters of genes can be evaluated.

5.7. Gene Network Modeling

In living systems, many dynamic, adaptive, evolving processes are observed at different levels, and at different stages of the development, that are involved in a complex interaction. At a molecular level and a cell level, the DNA, the RNA, and the protein molecules evolve and interact in a continuous way. The genes form dynamic gene networks (GNs) that define the complexity of the living organism [203]. It is not just the number of the genes in a genome, however, but the interaction between the genes that makes one organism more complex than another.

Many functions are associated with a neuronal cell and with neural networks in the brain [204]. An ensemble of cells (neurons) operates in concert, defining the function of the neural network (e.g., perception of a sound, or a brain disease such as epilepsy [205]). At the level of the whole brain, a complex dynamic interaction is observed, and certain cognitive functions are performed (e.g., speech and language learning, visual pattern recognition).

The genes, encoded in the DNA, which are transcribed into RNA and then translated into proteins in each cell, contain important information related to the brain activities. A specific gene from the genome relates to the activity of a neuronal cell in terms of a specific function, but the functioning of the brain is much more complex than that. The interaction between the genes is what defines the functioning of a neuron. Even in the presence of a mutated gene in the genome that is known to cause a brain disease, the neurons can still function normally provided a certain pattern of interaction between the genes is maintained—a certain state of the GRN [206]. However, if there is no mutated gene in the genome, certain abnormalities in brain functioning can be observed, as defined by a certain state of the interaction between the genes [205]. The above-cited and many other observations point to the significance of modeling a neuron and a neuronal ensemble at the gene level to predict the state of the ensemble. The process of modeling the gene interaction for the purpose of brain understanding is a significant challenge to biologists, mathematicians, information and computer scientists, brain scientists, and researchers from many other areas.

Models of GRN, derived from gene-expression RNA data, have been developed using different mathematical and computational methods, such as statistical correlation techniques [207, 208], evolutionary computation [209, 210], neural networks [211, 212], differential equations [213], and others [133, 214]. In Ref. [215], a simple GN model of five genes and gene clusters is derived from the time-course gene-expression data of a leukemia cell line U937 treated with retinoic acid with two phenotype states—cancer and normal. The model uses adaptive artificial neural networks—evolving connectionist systems, trained on data in an adaptive mode [15].

A simple GRN of four genes is given in Fig. 19.

6. IMPLICATIONS FOR MEDICINE

Profiling gene and protein expression using DNA and protein arrays has a tremendous effect in molecular-based classifications of diseases. There are two important tasks, among others, in this area: finding the correlation between subsets of genes/proteins and disease features

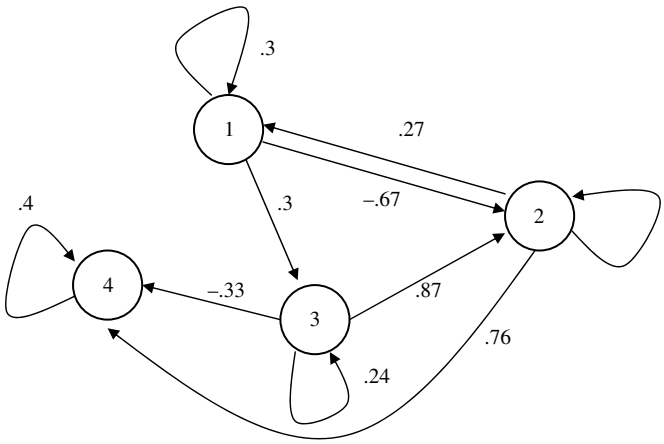

Figure 19. A simple gene regulatory network representing only four genes (the nodes) and their relative interaction strength (the arcs). Four functions are used to calculate the activity of each gene, depending on the activity of other genes in the network; these functions are not shown.

(progression, localization, etc.), and identifying the smallest informative set of genes/proteins associated with specific disease features.

Microarray technology offers an opportunity to screen thousands of genes simultaneously, to be monitored in parallel. New disease subtypes or molecularly distinct forms of the disease can be identified with the use of this technology: B-cell lymphoma [216], two molecularly distinct forms of diffuse large B-cell lymphoma with gene-expression patterns indicative of different stages of B-cell differentiation were identified; breast tumors [217, 218], gene-expression patterns provided a distinctive molecular portrait of each tumor in a set of 65 surgical specimens of human breast tumors from 42 different individuals; human acute leukemia [128], automatic discovery of the distinction between AML and ALL. In Ref. [219], cDNA microarray technology was used to explore variation in gene expression in 60 cell lines of human cancer, and a consistent relationship between the gene expression patterns and the tissue of origin was found. Specific features of these gene expression patterns appeared to be related to physiological properties of the cell lines (doubling time in culture, drug metabolism, and interferon response).

For some cases, DNA microarray technology is an inadequate method, as was noted in Ref. [220] for autoimmune diseases: the disease may manifest not in itself at the RNA level but, rather, at the protein one; protein function can be regulated by posttranslational modifications—phosphorylation, glycosylation, and sulfation, as well as many other modifications, are extremely important for protein function, as they can determine activity, stability, localization, and turnover; and there may be nonpredictive correlations between RNA expression and protein expression and function.

The correlation between levels of mRNA measured in oligonucleotide microarrays and protein is an important issue in DNA microarray technology. The lack of this correlation means that the predictive property of the gene expression is independent of gene function. In Ref. [221], two-dimensional PAGE, mass spectrometry, and Affymetrix oligonucleotide microarrays were used to identify proteins showing increased expression in lung adenocarcinoma and to examine whether the changes in protein expression may be attributable to transcriptional or other mechanisms of regulation. A comparison of the mRNA expression values and the protein expression values within the same tumor samples shows that expression of only two of 14 genes correlated significantly with levels of proteins. The lack of correlation between mRNA and protein level was also noted in Refs. [222, 223]. Also in some cases, using chips from different companies may give results that are not significantly correlated [224].

There is no strict linear relationship between genes and the "proteome" of a cell. Proteomics is complementary to genomics because it focuses on the gene products, and for

this reason proteomics directly contributes to drug development, as almost all drugs are directed against proteins.

Proteomics is a promising tool for the identification of new diagnostic tools (identification of disease markers or proteins that appear or disappear during the course of a disease), development of drugs, improvement of the efficiency of clinical trials (availability of biologically relevant markers for drug efficacy and safety), and clinical diagnostic testing.

These approaches include [225] the analysis of protein expression in normal and disease tissue, analysis of secreted proteins in cell lines and primary cultures, and direct serum protein profiling. Aberrantly expressed proteins might represent new markers. MS allows yielding comprehensive profiles of peptides and proteins without the need of first separating them, and it is highly suited for marker identification.

The changes in protein expression that enable tumors to initiate and progress in the local tissue microenvironment were analyzed in Ref. [226] with the use of an antibody microarray. It was demonstrated that quantitative, and potentially qualitative, differences in expression patterns of multiple proteins within epithelial cells reproducibly correlate with tumor progression.

A reverse-phase protein array approach with immobilization of the tissue's proteins was reported in Ref. [227]. These arrays were used for the screening of molecular markers and pathway targets in patient-matched human tissue during disease progression. In contrast to previous protein arrays that immobilize the probe, reverse-phase protein arrays immobilize the whole repertoire of patient proteins that represent the state of individual tissue cell populations undergoing disease transitions. A high degree of sensitivity, precision, and linearity was achieved, making it possible to quantify the phosphorylated status of signal proteins in human tissue cell subpopulations.

APPENDIX: GLOSSARY

Artificial neural networks are biologically inspired computational models that consist of processing elements (called neurons) and the connections between them, with coefficients (weights) bound to the connections, which constitute the neuronal structure. To the structure are also attached training and recall algorithms. One of the most popular training algorithms is the backpropagation algorithm for adjusting the connection weights in a neural network, where the gradient descent rule is used for finding the optimal connection weights w_{ij} that minimize a global error E. A change of weight Δw_{ij} at a cycle $(t+1)$ is in the direction of the negative gradient of the error E.

Bayesian probability The following formula, which represents the conditional probability between two events C and A, is known as the Bayes Formula (Tamas Bayes, eighteenth century)

$$p(A|C) = \frac{p(A|C)p(A)}{p(C)}$$

Using the Bayes formula involves difficulties, mainly concerning the evaluation of the prior probabilities $p(A)$, $p(C)$, $p(C|A)$. In practice (e.g., in statistical pattern recognition), the latter is assumed to be of a Gaussian type. The Bayes theorem assumes that if the condition C consists of condition elements $C_1, C_2, \ldots, C_k$ they are independent (which may not be the case in some applications).

Clustering Based on a measured distance between instances (objects, points, vectors) from the problem space, subareas in the problem space of closely grouped instances can be defined. These areas are called clusters. They are defined by their cluster centers and by the membership of the data points to them. A center c_i of a cluster C_i is defined as an instance the mean of the distances to which from each instance in the cluster, is minimum. Let us have a set X of p data items represented in an n-dimensional space. A clustering procedure results in defining k disjoint subsets (clusters), such that every data item (n-dimensional vector) belongs to one only cluster. A cluster membership function M_i is defined for each

of the clusters $C_1, C_2, \ldots, C_k$:

$$M_i : X \to \{0, 1\}$$
$$M_i(x) = \begin{cases} 1 & x \in C_i \\ 0 & x \notin C_i \end{cases}$$

where x is a data instance (vector) from X. In fuzzy clustering, one data vector may belong to several clusters to certain degree of membership, with all of the degrees summing up to 1.

Data are the numbers, the characters, and the quantities operated on by a computer.

Data normalization is a transformation of data from its original scale into another, predefined scale (e.g., [0, 1]). Normalization is linear when uses the following formula (for the case of a targeted scale of [0, 1])

$$v_{\text{norm}} = \frac{v - x_{\min}}{x_{\max} - x_{\min}}$$

where v is a current value of the variable x; $x_{\min}$ is the minimum value for this variable, and $x_{\max}$ is the maximum value for that variable x in the data set.

Distance between data vectors A way of measuring difference between data vectors. The distance between two data points in an n-dimensional geometrical space can be measured in several ways, for example

Hamming:

$$D_{ab} = \sum |a_i - b_i|$$

Euclidean distance:

$$E_{ab} = \sqrt{\frac{1}{n} \sum (a_i - b_i)^2}$$

Fuzzy clustering is a procedure of clustering data into possibly overlapping clusters, such that each of the data examples may belong to each of the clusters to a certain degree. The procedure aims at finding the cluster centers V_i $(i = 1, 2, \ldots, c)$ and the cluster membership functions μi, which define to what degree each of the n examples belong to the ith cluster. The number of clusters c is either defined *a priori* (supervised type of clustering) or chosen by the clustering procedure (unsupervised type of clustering). The result of a clustering procedure can be represented as a fuzzy relation $\mu_{i,k}$ such that

(i) $\sum \mu_{i,k} = 1$, for each $k = 1, 2, \ldots, n$; (the total membership of an instance to all the clusters equals 1)
(ii) $\sum_{\mu_{i,k}}$ for each $i = 1, 2, \ldots, c$ (there are no empty clusters)

Information is the ordered, structured, interpreted data—the news.

Knowledge is the theoretical or practical understanding of a subject: gained experience, true and justified belief, the way we do things.

Knowledge-based neural networks (KBNNs) These are prestructured neural networks allow for data and knowledge manipulation, including learning from data, rule insertion, rule extraction, adaptation, and reasoning. KBNNs have been developed either as a combination of symbolic AI systems and NNs, as a combination of fuzzy logic systems and NNs, or as other hybrid systems. Rule insertion and rule extraction operations are typical operations for a KBNN to accommodate existing knowledge along with data and to produce an explanation of what the system has learned.

Kohonen Self-Organizing Map (SOM) A self-organized map neural network for unsupervised learning invented by Professor Teuvo Kohonen and developed by him and other researchers [228, 229].

Multilayer perceptron network (MLP) is a neural network that consists of an input layer, at least one intermediate or "hidden" layer, and one output layer, with the neurons from each layer being fully connected (or, in some particular applications, partially connected) to the neurons from the next layer.

Multiple sequence alignment is the procedure of comparing sequences by searching for the similarity in the subsets that are in the same order in the sequences. Each subset can consist of one or more characters of the sequence and gaps between them.

Principle component analysis (PCA) Finding a smaller number of m components $Y = (y_1, y_2, \ldots, y_m)$ (aggregated variables) that can represent the goal function $F(x_1, x_2, \ldots, x_n)$ of n variables, $n > m$ to a desired degree of accuracy Θ (i.e., $F = MY + \Theta$, where M is a matrix that has to be found through the PCA).

Probability theory is based on the following three axioms:

Axiom 1. $0 \leq p(E) \leq 1$ The axiom defines the probability $p(E)$ of an event E as a real number in the closed interval [0, 1]. A probability $p(E) = 1$ indicates a certain event, and $p(E) = 0$ indicates an impossible event.

Axiom 2. $\sum p(E_i)$ $E_1 \cup E_2 \cup \cdots \cup E_k = U$, where U is a problem space (universum);

Axiom 3. $p(E_1 \vee E_2) = p(E_1) + p(E_2)$, where E_1 and E_2 are mutually exclusive events. This axiom indicates that if the events E_1 and E_2 cannot occur simultaneously, the probability of one or the other happening is the sum of their probabilities.

REFERENCES

1. J. C. Wooley and M. N. Varma, *Basic Life Sci.* 63, 1 (1994).
2. S. Brenner, *Novartis Found. Symp.* 213, 106 (1998).
3. D. S. Roos, *Science* 291, 1260 (2001).
4. J. D. Watson and F. H. C. Crick, *Nature* 171, 737 (1953).
5. B. A. Shapiro and W. Kasprzak, *J. Mol. Graphics* 14, 194 (1996).
6. B. A. Shapiro, W. Kasprzak, J. C. Wu, and K. Currey, in "Pattern Discovery in Biomolecular Data" (J. T. L. Wang, B. A. Shapiro, and D. Shasha, Eds.), p. 183. Oxford University Press, New York, 1999.
7. J. S. Richardson, *Biophysics* 63, 1186 (1992).
8. J. L. McClelland and D. E. Rummelhart, in "Explorations in Parallel Distributed Processing," Vol. 3, p. 318. MIT Press, Cambridge, MA, 1988.
9. D. G. Kneller, F. E. Cohen, and R. Langridge, *J. Mol. Biol.* 214, 171 (1990).
10. B. Rost and C. Sander, *Proc. Nat. Acad. Sci.* 90, 7558 (1993).
11. B. Rost and C. Sander, *J. Mol. Biol.* 232, 584 (1993).
12. B. Rost and C. Sander, *Protein* 19, 55 (1994).
13. J. A. Cuff and G. J. Barton, *Proteins* 34, 508 (1999).
14. N. Qian and T. Sejnowski, *J. Theoret. Biol.* 202, 865 (1988).
15. N. Kasabov, "Evolving Connectionist Systems—Methods and Applications in Bioinformatics, Brain Study and Intelligent Machines." Springer, New York, 2002.
16. N. Guex and M. C. Peitsch, *Electrophoresis* 18, 2714 (1997).
17. M. C. Peitsch, *Biochem. Soc. Trans.* 24, 274 (1996).
18. O. Lund, K. Frimand, J. Gorodkin, H. Bohr, J. Bohr, J. Hansen, and S. Brunak, *Protein Eng.* 10, 1241 (1997).
19. P. A. Bates, L. A. Kelley, R. M. MaxCallum, and M. J. E. Sternberg, *Prot. Struct. Funct. Genet.* 5, 39 (2001).
20. P. A. Bates and M. J. E. Sternberg, *Prot. Struct. Funct. Genet.* 3, 47 (1999).
21. B. Contrereas-Moreira and P. A. Bates, *Bioinformatics* 18, 1141 (2002).
22. I. N. Shindyalov and P. E. Bourne, "Forth meeting on the critical assessment of techniques for protein structure prediction," p. A-92, 2000.
23. I. N. Shindyalov and P. E. Bourne, *Nucl. Acids Res.* 29, 228 (2001).
24. A. J. Gibbs and G. A. McIntyre, *Eur. J Biochem.* 16, 1 (1970).
25. J. V. J. Maizel and R. P. Lenk, *Proc. Natl. Acad. Sci.* 78, 7665 (1981).
26. D. J. States and M. S. Boguski, in "Sequence Analysis Primer" (M. Gribskov and J. Devereux, Eds.), p. 92. Stockton Press, NY, 1991.
27. G. Grillo, F. Licciulli, S. Liuni, E. Sbisa, and G. Pesole, *Nucl. Acids Res.* 31, 3608 (2003).
28. V. Matys, E. Fricke, R. Geffers, E. Gossling, M. Haubrock, R. Hehl, K. Hornischer, D. Karas, A. E. Kel, O. V. Kel-Margoulis, D. U. Kloos, S. Land, B. Lewicki-Potapov, H. Michael, R. Munch, I. Reuter, S. Rotert, H. Saxel, M. Scheer, S. Thiele, and E. Wingender, *Nucl. Acids Res.* 31, 374 (2003).
29. K. Quandt, K. Frech, H. Karas, E. Wingender, and T. Werner, *Nucl. Acids Res.* 23, 4878 (1995).
30. S. F. Altshul, *J. Mol. Biol.* 219, 555 (1991).
31. D. J. States, W. Gish, and S. F. Altshul, *Methods* 3, 66 (1991).
32. M. O. Dayhoff, in "Atlas of Protein Sequence and Structure," Vol. 5, suppl. 3. National Biomedical Research Foundation, Georgetown University, Washigton, DC, 1978.
33. G. H. Gonnet, M. A. Cohen, and S. A. Benner, *Science* 256, 1443 (1992).
34. D. T. Jones, W. R. Taylor, and J. M. Thornton, *Comput. Appl. Biosci.* 8, 275 (1992).
35. S. A. Benner, M. A. Cohen, and G. H. Gonnet, *Protein Eng.* 7, 1323 (1994).
36. S. Henikoff and J. G. Henikoff, *Proc. Natl. Acad. Sci.* 89, 10915 (1991).
37. G. H. Gonnet, M. A. Cohen, and S. A. Benner, *Biochem. Biophys. Res. Commun.* 199, 489 (1994).

38. D. T. Jones, W. R. Taylor, and J. M. Thornton, *FEBS Lett.* 339, 269 (1994).
39. D. W. Mount, "Bioinformatics." Cold Spring Harbor Laboratory Press, Cold Spring Harbor, NY, 2001.
40. D. J. Lipman and W. R. Pearson, *Science* 2277, 1435 (1985).
41. W. R. Pearson and D. J. Lipman, *Proc. Natl. Acad. Sci.* 85, 2444 (1988).
42. A. Klingenhoff, K. Frech, K. Quandt, and T. Werner, *Bioinformatics* 15, 180 (1999).
43. K. Frech, J. Danescu-Mayer, and T. Werner, *J. Mol. Biol.* 270, 674 (1997).
44. S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman, *J. Mol. Biol.* 215, 403 (1990).
45. S. F. Altschul, T. L. Madden, A. A. Schaffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman, *Nucl. Acids Res.* 25, 3389 (1997).
46. V. C. Wasinger, S. J. Cordwell, A. Cerpa-Poljak, J. X. Yan, A. A. Gooley, M. R. Wilkins, M. W. Duncan, R. Harris, K. L. Williams, and I. Humphery-Smith, *Electrophoresis* 16, 1090 (1995).
47. S. Fields and O. Song, *Nature* 340, 245 (1989).
48. V. Schachter, *Comp. Proteomics Suppl.* 32, S16 (2002).
49. J. S. Albala, *Expert Rev. Mol. Diagn.* 1, 145 (2001).
50. H. Zhu, J. F. Klemic, S. Chang, P. Bertone, A. Casamayor, K. G. Klemic, D. Smith, M. Gerstein, M. A. Reed, and M. Snyder, *Nat. Genet.* 26, 283 (2000).
51. G. MacBeath and S. L. Schreiber, *Science* 289, 1760 (2000).
52. P. Arenkov, A. Kukhtin, A. Gemmell, S. Voloshchuk, V. Chupeeva, and A. Mirzabekov, *Anal. Biochem.* 278, 123 (2000).
53. Y. Lee, E. K. Lee, Y. W. Cho, T. Matsui, I.-C. Kang, T.-S. Kim, and M. H. Han, *Proteomics* 3, 2289 (2003).
54. M. F. Templin, D. Stoll, J. M. Schwenk, O. Potz, S. Kramer, and T. O. Joos, *Proteomics* 3, 2155 (2003).
55. T. S. Lewis, J. B. Hunt, L. D. Aveline, K. R. Jonscher, D. F. Louie, J. M. Yeh, T. S. Nahreini, K. A. Resing, and N. G. Ahn, *Mol. Cell* 6, 1343 (2000).
56. J. H. McKerrow, V. Bhargava, E. Hansell, S. Huling, T. Kuwahara, M. Matley, L. Coussens, and R. Warren, *Mol. Med* 6, 460 (2000).
57. M. J. Han and S. Y Lee, *Proteomics* 3, 2317 (2003).
58. D. N. Chakravarti, B. Chakravarti, and I. Moutsatsos, *Comp. Proteomics Suppl.* 32, S4 (2002).
59. R. C. Beavis and D. Fenyo, in "Proteomics: A Trends Guide" (W. Blackstock and M. Mann, Eds.), p. 22. Elsevier, Amsterdam, 2000.
60. D. Fenyo, *Curr. Opin. Biotechnol.* 11, 391 (2000).
61. N. L. Anderson, J. Taylor, A. E. Scandora, B. P. Coulter, and N. G. Anderson, *Clin. Chem.* 27, 1807 (1981).
62. J. I. Garrels, *J. Biol. Chem.* 254, 7961 (1979).
63. P. F. Lemkin and L. E. Lipkin, *Comput. Biomed. Res.* 14, 272 (1981).
64. R. Appel, D. Hochstrasser, C. Roch, M. Funk, A. F. Muller, and C. Pellegrini, *Electrophoresis* 9, 136 (1988).
65. R. D. Appel, D. F. Hochstrasser, M. Funk, J. R. Vargas, C. Pelegrini, A. F. Muller, and J. R. Scherrer, *Electrophoresis* 12, 722 (1991).
66. T. Pun, D. F. Hochstrasser, R. D. Appel, M. Funk, V. Villars-Augsburger, and C. Pelegrini, *Appl. Theor. Electrophor.* 1, 3 (1988).
67. D. G. Rowlands, A. Flook, P. I. Payne, A. van Hoff, T. Niblett, and S. McKee, *Electrophoresis* 9, 820 (1988).
68. J. I. Garrels, *J. Biol. Chem.* 264, 5269 (1989).
69. W. J. Henzel, T. M. Bulleci, J. T. Stults, S. C. Wong, C. Grimley, and C. Watanabe, *Proc. Natl. Acad. Sci.* 90, 5011 (1993).
70. P. James, M. Quadroni, E. Carafoli, and G. Gonnet, *Biochem. Biophys. Res. Commun.* 195, 58 (1993).
71. M. Mann, P. Hojrup, and P. Roepstorff, *Biol. Mass Spectrom.* 22, 338 (1993).
72. D. J. C. Pappin, P. Hojrup, and A. J. Bleasby, *Curr. Biol.* 3, 327 (1993).
73. D. N. Perkins, D. J. Pappin, D. M. Creasy, and J. S. Cottrell, *Electrophoresis* 20, 3551 (1999).
74. J. K. Eng, A. L. McCormack, and J. R. I. Yates, *J. Am. Soc. Mass Spetrom.* 5, 976 (1994).
75. M. Mann and M. Wilm, *Anal. Chem.* 66, 4390 (1994).
76. P. D. Von Haller, E. Yi, S. Donohoe, K. Vaughn, A. Keller, A. I. Nesvizhskii, J. Eng, X. J. Li, D. R. Goodlett, R. Aebersold, and J. D. Watts, *Mol. Cell Proteomics.* 2, 428 (2003).
77. P. D. Von Haller, E. Yi, S. Donohoe, K. Vaughn, A. Keller, A. I. Nesvizhskii, J. Eng, X. J. Li, D. R. Goodlett, R. Aebersold, and J. D. Watts, *Mol. Cell Proteomics* 2, 426 (2003).
78. A. Keller, A. I. Nesvizhskii, E. Kolker, and R. Aebersold, *Anal. Chem.* 74, 5383 (2002).
79. D. K. Han, J. Eng, H. Zhou, and R. Aebersold, *Nat. Biotechnol.* 19, 946 (2001).
80. A. I. Nesvizhskii, A. Keller, E. Kolker, and R. Aebersold, *Anal. Chem.* 75, 4646 (2003).
81. J. A. Taylor and R. S. Johnson, *Anal. Chem.* 73, 2594 (2001).
82. M. R. Wilkins, E. Gasteiger, A. A. Gooley, B. R. Herbert, M. P. Molloy, P. A. Binz, K. Ou, J. C. Sanchez, A. Bairoch, K. L. Williams, and D. F. Hochstrasser, *J. Mol. Biol.* 289, 645 (1999).
83. A. Sali, R. Glaeser, T. Earnest, and W. Baumeister, *Nature* 422, 216 (2003).
84. E. Krieger, S. B. Nabuurs, and G. Vriend, *Methods Biochem. Anal.* 44, 509 (2003).
85. A. Godzik, *Methods Biochem. Anal.* 44, 525 (2003).
86. W. J. Browne, A. C. North, D. C. Phillips, K. Brew, T. C. Vanaman, and R. L. Hill, *J. Mol. Biol.* 42, 65 (1969).
87. J. Greer, *Proc. Natl. Acad. Sci.* 77, 3393 (1980).
88. J. Greer, *Proteins* 7, 317 (1990).
89. R. Sanchez, U. Pieper, F. Melo, N. Eswar, M. A. Marti-Renom, M. S. Madhusudhan, N. Mirkovic, and A. Sali, *Nat. Struct. Biol.* 7, 986 (2000).
90. D. S. Dimitrov, *Cell* 101, 697 (2000).

91. P. Prabakaran, X. Xiao, and D. S. Dimitrov, *Biochem. Biophys. Res. Commun.* 314, 235 (2004).
92. W. Li, M. J. Moore, N. Vasilieva, J. Sui, S. K. Wong, M. A. Berne, M. Somasundaran, J. L. Sullivan, K. Luzuriaga, T. C. Greenough, H. Choe, and M. Farzan, *Nature* 426, 450 (2003).
93. X. Xiao, S. Chakraborti, A. S. Dimitrov, K. Gramatikoff, and D. S. Dimitrov, *Biochem. Biophys. Res. Commun.* 312, 1159 (2003).
94. M. Donoghue, F. Hsieh, E. Baronas, K. Godbout, M. Gosselin, N. Stagliano, M. Donovan, B. Woolf, K. Robison, R. Jeyaseelan, R. E. Breitbart, and S. Acton, *Circ. Res.* 87, E1 (2000).
95. S. R. Tipnis, N. M. Hooper, R. Hyde, E. Karran, G. Christie, and A. J. Turner, *J. Biol. Chem.* 275, 33238 (2000).
96. M. A. Crackower, R. Sarao, G. Y. Oudit, C. Yagil, I. Kozieradzki, S. E. Scanga, A. J. Oliveira-dos-Santos, J. da Costa, L. Zhang, Y. Pei, J. Scholey, C. M. Ferrario, A. S. Manoukian, M. C. Chappell, P. H. Backx, Y. Yagil, and J. M. Penninger, *Nature* 417, 822 (2002).
97. R. Natesh, S. L. Schwager, E. D. Sturrock, and K. R. Acharya, *Nature* 421, 551 (2003).
98. H. M. Kim, D. R. Shin, O. J. Yoo, H. Lee, and J. O. Lee, *FEBS Lett.* 538, 65 (2003).
99. D. S. Dimitrov, *Nat. Rev. Microbiol.* 2, 109 (2004).
100. J. D. Thompson, D. G. Higgins, and T. J. Gibson, *Nucl. Acids Res.* 22, 4673 (1994).
101. M. J. Sutcliffe, F. R. Hayes, and T. L. Blundell, *Protein Eng.* 1, 385 (1987).
102. M. J. Sutcliffe, I. Haneef, D. Carney, and T. L. Blundell, *Protein Eng.* 1, 377 (1987).
103. R. A. Laskowski, M. W. MacArthur, D. S. Moss, and J. M. Thornton, *J. Appl. Cryst.* 26, 283 (1993).
104. C. J. Sigrist, L. Cerutti, N. Hulo, A. Gattiker, L. Falquet, M. Pagni, A. Bairoch, and P. Bucher, *Brief. Bioinform.* 3, 265 (2002).
105. B. Lee and F. M. Richards, *J. Mol. Biol.* 55, 379 (1971).
106. A. Nicholls, K. A. Sharp, and B. Honig, *Proteins* 11, 281 (1991).
107. J. Kyte and R. F. Doolittle, *J. Mol. Biol.* 157, 105 (1982).
108. J. L. Guy, R. M. Jackson, K. R. Acharya, E. D. Sturrock, N. M. Hooper, and A. J. Turner, *Biochemistry* 42, 13185 (2003).
109. O. Spiga, A. Bernini, A. Ciutti, S. Chiellini, N. Menciassi, F. Finetti, V. Causarono, F. Anselmi, F. Prischi, and N. Niccolai, *Biochem. Biophys. Res. Commun.* 310, 78 (2003).
110. A. Levchenko, *Mol. Biol. Rep.* 28, 83 (2001).
111. A. J. Lotka, *J. Amer. Chem. Soc.* 42, 1595 (1920).
112. V. Volterra, *Mem. Acad. Lincei* 2, 31 (1926).
113. A. L. Hodgkin and A. F. Huxley, *J. Physiol.* (*Lond.*) 117, 500 (1952).
114. M. Eigen, *Naturwissenschaften* 58, 465 (1971).
115. A. Gierer and H. Meinhardt, *Kybernetik* 12, 30 (1972).
116. S. R. Neves and R. Iyengar, *BioEssays* 24, 1110 (2002).
117. H. Kitano, *Nature* 420, 206 (2002).
118. U. Bhalla and R. Iyengar, *Science* 283, 381 (1999).
119. B. N. Kholodenko, O. V. Demin, G. Moehren, and J. B. Hoek, *J. Biol. Chem.* 274, 30169 (1999).
120. I. A. Sidorov and A. A. Romanyukha, *Math. Biosci.* 115, 187 (1993).
121. G. I. Marchuk, R. V. Petrov, A. A. Romahyukha, and G. A. Bocharov, *J. Theoret. Biol.* 151, 1 (1991).
122. L. W. Loew, A. Cowan, and I. Moraru, "Cranwell Resort." Lenox, MA, 2001.
123. I. Cloete and J. Zurada, "Knowledge-Based Neurocomputing." MIT Press, Cambridge, MA, 2000.
124. "Microarray Biochip Technology." Eaton Publishing, Natick, MA, 2001.
125. M. Futschik and N. Kasabov, in "RECOMB'2001 Proceedings—Currents in Computational Molecular Biology 2001" (T. Lengauer and D. Sankoff, Eds.), p. 175. Montreal, 2001.
126. U. Alon, N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine, *Proc. Nat. Acad. Sci.* 96, 6745 (1999).
127. N. Kasabov, Adaptive Learning Method and System, patent 503882, New Zealand (2000).
128. T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E. S. Lander, *Science* 286, 531 (1999).
129. J. Hasty, D. McMillen, I. Farren, and J. J. Collins, *Nat. Rev. Genet.* 2, 268 (2001).
130. G. Weng, U. S. Bhalla, and R. Iyengar, *Science* 284, 92 (1999).
131. P. Smolen, D. A. Baxter, and J. H. Byrne, *Neuron* 26, 567 (2000).
132. H. Jeff, I. Farren, D. Milos, M. David, and J. J. Collins, *Chaos* 11, 207 (2001).
133. H. de Jong, *J. Comput. Biol.* 9, 67 (2002).
134. S. Huang, *Pharmacogenomics* 2, 203 (2001).
135. H. Bolouri and E. H. Davidson, *BioEssays* 24, 1118 (2002).
136. J. Khan, J. Wei, M. Ringner, L. H. Saal, M. Ladanyi, F. Westerman, F. Berthold, M. Schwab, C. R. A. Nionescu, C. Peterson, and P. S. M. Meltzer, *Nat. Med.* 7, 673 (2001).
137. A. Metcalfe, "Statistics in Engineering—A Practical Approach." Chapman & Hall, London, 1994.
138. R. J. Cho, M. J. Campbell, E. A. Winzeler, L. Steinmetz, A. Conway, L. Wodicka, T. G. Wolfsberg, A. E. Gabrielian, D. Landsman, D. J. Lockhart, and R. W. Davis, *Mol. Cell* 2, 73 (1998).
139. P. T. Spellman, G. Sherlock, M. Q. Zhang, V. R. Iyer, K. Anders, M. B. Eisen, P. O. Brown, D. Botstein, and B. Futcher, *Mol. Biol. Cell* 9, 3273 (1998).
140. J. L. deRisi, V. R. Iyer, and P. O. Brown, *Science* 275, 680 (1997).
141. S. Chu, J. DeRisi, M. Eisen, J. Mulholland, D. Botstein, P. O. Brown, and I. Herskowitz, *Science* 282, 699 (1998).

142. N. Pal and J. C. Bezdek, *IEEE Trans. Fuzzy Syst.* 3, 370 (1995).
143. L. von Bertalanffy, "General System Theory, Foundations, Development, Applications." George Braziller, New York, 1969.
144. J. Gibbs, *Science* 287, 1969 (2000).
145. C. Sander, *Science* 287, 1977 (2000).
146. D. Noble, *Science* 295, 1678 (2002).
147. H. Kitano, in "Foundations of Systems Biology." MIT Press, Cambridge, MA, 2001.
148. N. Weiner, "Cybernetics or Control and Communication in the Animal and the Machine." MIT Press, Cambridge, MA, 1948.
149. L. von Bertalnffy, "Modern Theories of Development: An Introduction to Theoretical Biology." Oxford University Press, New York, 1933.
150. M. A. Savageau, "Biochemical Systems Theory." Addison-Wesley, Reading, MA, 1976.
151. K. C. Chen, A. Csikasz-Nagy, G. B. ., J. Val, B. Novak, and J. J. Tyson, *Mol. Biol. Cell* 11, 369 (2000).
152. M. T. Borisuk and J. J. Tyson, *J. Theoret. Biol.* 195, 69 (1998).
153. K. J. Kauffman, P. Prakash, and J. S. Edwards, *Curr. Opin. Biotechnol.* 14, 491 (2003).
154. J. S. Edwards, R. U. Ibarra, and P. B. O., *Nat. Biotechnol.* 19, 125 (2001).
155. M. A. Savageau, *Curr. Topics Cell. Regulation* 6, 63 (1972).
156. H. Kacser and J. A. Burns, *Symp. Soc. Exp. Biol.* 27, 65 (1973).
157. U. Alon, M. G. Surette, N. Barkai, and S. Leibler, *Nature* 397, 168 (1999).
158. G. von Dassow, E. Mier, M. Munro, and M. Odell, *Nature* 406, 188 (2000).
159. T.-M. Yi, Y. Huang, M. I. Simon, and J. Doyle, *Proc. Natl. Acad. Sci.* 97, 4649 (2000).
160. H. Kurata and K. Taira, in "Proceedings of the Fourth Annual International Conference on Computational Molecular Biology." Tokyo, Japan, 2000, Vol. 36.
161. M. E. Csete and J. C. Doyle, *Science* 295, 1664 (2002).
162. "Scaling in Biology." Oxford University Press, New York, 2000.
163. J. Whitfield, *Nature* 413, 342 (2001).
164. P. Smolen, D. A. Baxter, and J. H. Byrne, *AJP—Cell Physiol.* 274, C531 (1998).
165. D. E. Koshland, Jr., A. Goldbeter, and J. B. Stock, *Science* 217, 220 (1982).
166. A. Goldbeter and D. E. Koshland, Jr., *Proc. Natl. Acad. Sci.* 78, 6840 (1981).
167. J. E. Ferrell, Jr. and E. M. Machleder, *Science* 280, 895 (1998).
168. J. E. Ferrell and X. Wen, *Chaos* 11, 227 (2001).
169. U. S. Bhalla and I. Ravi, *Chaos* 11, 221 (2001).
170. M. B. Elowitz and S. Leibler, *Nature* 403, 335 (2000).
171. J. Koshland, *Science* 295, 2215 (2002).
172. M. Hucka, A. Finney, H. Sauro, H. Bolouri, J. Doyle, and H. Kitano, in "Foundations of Systems Biology." MIT Press, Cambridge, MA, 2001.
173. A. P. Arkin, http://gobi.lbl.gov/∼aparkin/Stuff/Software.html (2001).
174. I. Goryanin, T. C. Hodgman, and E. Selkov, *Bioinformatics* 15, 749 (1999).
175. I. Goryanin, http://websites.ntl.com/∼igor.goryanin/ (2001).
176. M. Tomita, Y. Nakayama, Y. Naito, T. Shimizu, K. Hashimoto, K. Takahashi, Y. Matsuzaki, K. Yugi, F. Miyoshi, Y. Saito, A. Kuroki, T. Ishida, T. Iwata, M. Yoneda, M. Kita, Y. Yamada, E. Wang, S. Seno, M. Okayama, A. Kinoshita, Y. Fujita, R. Matsuo, T. Yanagihara, D. Watari, S. Ishinabe, and S. Miyamoto, http://www.e-cell.org (2001).
177. M. Tomita, K. Hashimoto, K. Takahashi, T. S. Shimizu, Y. Matsuzaki, F. Miyoshi, T. Saito, S. Tanida, K. Yugi, J. C. Venter, and C. Hutchison, *Bioinformatics* 15, 72 (1999).
178. P. Mendes, Trends *Biochem. Sci.* 22, 361 (1997).
179. P. Mendes, http://www.gepasi.org/ (2001).
180. H. M. Sauro, *Math. Comput. Modelling* 15, 15 (1991).
181. H. Sauro and D. A. Fell, in "Animating the Cellular Map 9th International BioThermoKinetics Meeting" (J.-H. S. R. J. M. a. S. J. L. Hofmeyr, Ed.). Stellenbosch University Press, 2000.
182. D. Bray, C. Firth, N. Le Novère, and T. Shimizu, http://www.anat.cam.ac.uk/comp-cell (2001).
183. C. J. Morton-Firth and D. Bray, *J. Theoret. Biol.* 192, 117 (1998).
184. J. Schaff, B. Slepchenko, and L. M. Loew, in "Methods in Enzymology," Vol. 321, p. 1. Academic Press, 2000.
185. J. C. Schaff, B. Slepchenko, F. Morgan, J. Wagner, D. Resasco, D. Shin, Y. S. Choi, L. Loew, J. Carson, A. Cowan, I. Moraru, J. Watras, M. Teraski, and C. Fink, http://www.nrcam.uchc.edu/ (2001).
186. T. Bray, J. Paoli, and C. M. Sperberg-McQueen, www.w3.org/TR/1998/REC-xml-19980210 (1998).
187. M. Hucka, A. Finney, H. Sauro, H. Bolouri, J. C. Doyle, H. Kitano, A. P. Arkin, B. J. Bornstein, D. Bray, A. Cornish-Bowden, A. A. Cuellar, S. Dronov, E. D. Gilles, M. Ginkel, V. Gor, I. I. Goryanin, W. J. Hedley, T. C. Hodgman, J.-H. Hofmeyr, P. J. Hunter, N. S. Juty, J. L. Kasberger, A. Kremling, U. Kummer, N. Le Nov'ere, L. M. Loew, D. Lucio, P. Mendes, E. Minch, E. D. Mjolsness, Y. Nakayama, M. R. Nelson, P. F. Nielsen, T. Sakurada, J. C. Schaff, B. E. Shapiro, T. S. Shimizu, H. D. Spence, J. Stelling, K. Takahashi, M. Tomita, J. Wagner, and J. Wang, *Bioinformatics* 19, 524 (2003).
188. W. J. Hedley, M. R. Nelson, D. P. Bullivant, and P. F. Nielson, *Philos. Trans. R. Soc. Lond. A* 359, 1073 (2001).
189. K. W. Kohn, *Mol. Biol. Cell.* 10, 2703 (1999).
190. K. Kohn, *Chaos* 11, 84 (2001).
191. I. Pirson, N. Fortemaison, C. Jacobs, S. Dremier, J. E. Dumont, and C. Maenhaut, *Trends Cell Biol.* 10, 404 (2000).

192. D. L. Cook, J. F. Farley, and S. J. Tapscott, *Genome* 2, research0012.1 (2001).
193. H. Kitano, *BIOSILICO* 1, 169 (2003).
194. A. Funahashi, M. Morohashi, and H. Kitano, *BIOSILICO* 1, 159 (2003).
195. B. Mishra, R. S. Daruwala, Y. Zhou, N. Ugel, A. Policriti, M. Antoniotti, S. Paxia, M. Rejali, A. Rudra, V. Cherepinsky, N. Silver, W. Casey, C. Piazza, M. Simeoni, P. Barbano, M. Spivak, J. Feng, O. Gill, M. Venkatesh, F. Cheng, B. Sun, I. Ioniata, T. Anantharaman, E. J. Hubbard, A. Pnueli, D. Harel, V. Chandru, R. Hariharan, M. Wigler, F. Park, S. C. Lin, Y. Lazebnik, F. Winkler, C. R. Cantor, A. Carbone, and M. Gromov, *OMICS* 7, 253 (2003).
196. B. M. Slepchenko, J. C. Schaff, I. Macara, and L. M. Loew, *Trends Cell Biol.* 13, 570 (2003).
197. P. Mendes, *Comput. Applic. Biosci.* 9, 563 (1993).
198. P. Mendes, *Trends Biochem. Sci.* 22, 361 (1997).
199. P. Mendes and D. B. Kell, *Bioinformatics* 14, 869 (1998).
200. R. Zacks, *MIT Technol. Rev.* 37 (2001).
201. M. A. Gibson and E. Mjolsness, in "Computational Modeling of Genetic and Biochemical Networks" (J. M. Bower and H. Bolouri, Eds.), p. 1. MIT Press, Cambridge, MA, 2001.
202. R. Somogyi, S. Fuhrman, and X. Wen, in "Computational Modelling of Genetic and Biochemical Network" (J. M. Bower and H. Bolouri, Eds.), p. 120. MIT Press, Cambridge, MA, 2001.
203. P. Baldi and S. Brunak, "Bioinformatics—A Machine Learning Approach." 2001.
204. M. Arbib, "The Handbook of Brain Theory and Neural Networks." MIT Press, 2003.
205. V. Crunelli and N. Lereshe, *Nat. Rev. Neurosci.* 3, 371 (2002).
206. R. Morita, E. Miyazaki, C. G. Fong, and et al., *Biochem. Biophys. Res. Commun.* 248, 307 (1998).
207. P. D'Haeseleer, S. Liang, and R. Somogyi, *Bioinformatics* 16, 707 (2000).
208. S. Liang, S. Fuhrman, and R. Somogyi, 3 (1998).
209. S. Ando, E. Sakamoto, and H. Iba, (2002), p. 1249.
210. G. Fogel and D. Corne, "Evolutionary Computation for Bioinformatics." Morgan Kaufmann, 2003.
211. J. Vohradsky, *J. Biol. Chem.* 276, 36168 (2001).
212. J. Vohradsky, *FASEB J.* 15, 846 (2001).
213. K. W. Kohn and D. S. Dimitrov, "Computer Modeling and Simulation of Complex Biological Systems." 1999.
214. J. Bower and H. E. Bolouri, "Computational Modelling of Genetic and Biochemical Networks." MIT Press, 2001.
215. N. Kasabov and D. Dimitrov, "ICONIP'2002—International Conference on Neuro-Information Processing," Singapore, 2002.
216. A. A. Alizadeh, M. B. Eisen, R. E. Davis, C. Ma, I. S. Lossos, A. Rosenwald, J. C. Boldrick, H. Sabet, T. Tran, X. Yu, J. I. Powell, L. Yang, G. E. Marti, T. Moore, J. J. Hudson, L. Lu, D. B. Lewis, R. Tibshirani, G. Sherlock, W. C. Chan, T. C. Greiner, D. D. Weisenburger, J. O. Armitage, R. Warnke, and L. M. Staudt, *Nature* 403, 503 (2000).
217. C. M. Perou, T. Sorlie, M. B. Eisen, M. van de Rijn, S. Jeffrey, C. A. Rees, J. R. Pollack, D. T. Ross, H. Jonsen, L. A. Akslen, O. Fluge, A. Pergamenshikov, C. Williams, S. X. Zhu, P. E. Lonning, A.-L. Borresen-Dale, P. O. Brown, and D. Botstein, *Nature* 406, 747 (2000).
218. T. Sorlie, C. M. Perou, R. Tibshirani, T. Aas, S. Geisler, H. Johnsen, T. Hastie, M. B. Eisen, M. van de Rijn, S. S. Jeffrey, T. Thorsen, H. Quist, J. C. Matese, P. O. Brown, D. Botstein, P. E. Lonning, and A. L. Borresen-Dale, *Proc. Natl. Acad. Sci.* 98, 10869 (2001).
219. D. T. Ross, U. Scherf, M. B. Eisen, C. M. Perou, C. Rees, P. Spellman, V. Iyer, S. S. Jeffrey, R. M. Van de, M. Waltham, A. Pergamenschikov, J. C. Lee, D. Lashkari, D. Shalon, T. G. Myers, J. N. Weinstein, D. Botstein, and P. O. Brown, *Nat. Genet.* 24, 227 (2000).
220. W. H. Robinson, L. Steinman, and P. J. Utz, *Arthritis Rheumatism* 46, 885 (2002).
221. G. Chen, T. G. Gharib, C. C. Huang, D. G. Thomas, K. A. Shedden, J. M. G. Taylor, S. L. R. Kardia, D. E. Misek, T. J. Giordano, M. D. Iannettoni, M. B. Orringer, S. M. Hanash, and D. G. Beer, *Clin. Cancer Res.* 8, 2298 (2002).
222. S. Gygi, Y. Rochon, B. R. Franza, and R. Aebersold, *Mol. Cell. Biol.* 19, 1720 (1999).
223. B. Futcher, G. I. Latter, P. Monardo, C. S. McLaughlin, and J. I. Garrels, *Mol. Cell. Biol.* 19, 7357 (1999).
224. P. K. Tan, T. J. Downey, E. L. Spitznagel, Jr., P. Xu, D. Fu, D. S. Dimitrov, R. A. Lempicki, B. M. Raaka, and M. C. Cam, *Nucl. Acids Res.* 31, 5676 (2003).
225. S. Hanash, *Nature* 422, 226 (2003).
226. V. Knezevic, C. Leethanakul, V. E. Bichsel, J. M. Worth, V. V. Prabhu, J. S. Gutkind, L. A. Liotta, P. J. Munson, E. F. Petricoin, and D. B. Krizman, *Proteomics* 1, 1271 (2001).
227. C. P. Paweletz, L. Charboneau, V. E. Bichsel, N. L. Simone, T. Chen, J. W. Gillespie, M. R. Emmert-Buck, M. J. Roth, E. F. Petricoin, and L. A. Liotta, *Oncogene* 20, 1981 (2001).
228. T. Kohonen, *IEEE* 78, 1464 (1990).
229. T. Kohonen, "Self-Organizing Maps." Springer, 1997.