PHYSICAL REVIEW E VOLUME 55, NUMBER 6 JUNE 1997

Noise and randomlike behavior of perceptrons: Theory and application to protein structure prediction

M. Compiani
Dipartimento di Scienze Chimiche, Università di Camerino, 62032 Camerino MC, Italy

P. Fariselli and R. Casadio
Laboratorio di Biofisica, Dipartimento di Biologia, Università di Bologna, 40126 Bologna, Italy

(Received 30 August 1995; revised manuscript received 27 September 1996)

In the first part of this paper we study the performance of a single-layer perceptron that is expected to classify patterns into classes in the case where the mapping to be learned is corrupted by noise. Extending previous results concerning the statistical behavior of perceptrons, we distinguish two mutually exclusive kinds of noise (I noise and R noise) and study their effect on the statistical information that can be drawn from the output. In the presence of I noise, the learning stage results in the convergence of the output to the probabilities that the input occurs in each class. R noise, on the contrary, perturbs the learning of probabilities to the extent that the performance of the perceptron deteriorates and the network becomes equivalent to a random predictor. We derive an analytical expression for the efficiency of classification of inputs affected by strong R noise. We argue that, from the standpoint of the efficiency score, the network is equivalent to a device performing biased random flights in the space of the weights, which are ruled by the statistical information stored by the network during the learning stage. The second part of the paper is devoted to the application of our model to the prediction of protein secondary structures where one has to deal with the effects of R noise. Our results are shown to be consistent with data drawn from experiments and simulations of the folding process. In particular, the existence of coding and noncoding traits of the protein is properly rationalized in terms of R-noise intensity. In addition, our model provides a justification of the seeming existence of a relationship between the prediction efficiency and the amount of R noise in the sequence-to-structure mapping. Finally, we define an entropylike parameter that is useful as a measure of R noise. [S1063-651X(97)02004-7]

PACS number(s): 87.10.+e

I. INTRODUCTION

The present paper belongs to the mainstream of research that focuses on the statistical aspects of learning in neural networks (for a review see [1,2]); our main scope is to clarify their capability to detect statistical information in the learning set. Emphasis on these aspects is motivated by the consideration that in many cases of practical interest neural networks detect statistical features of the problem under study. We limit our investigation to the simplest feed-forward neural networks, viz., the single-layer perceptrons, on which analytical considerations can be carried out with relative ease. From now on, for simplicity, the single-layer perceptron will be referred to as the perceptron.

Our starting point is the notion that pattern overlap is simultaneously the strength and the weakness of perceptrons used as classifiers. In point of fact overlap promotes classification of never-seen-before inputs but, at the same time, generates noise [3] that poses limitations to the accuracy of classification. Generally, neural networks are used to build an artificial mapping linking the end states of processes that are too complex for being extensively simulated or theoretically investigated. In this context, beside using the network as a black box, it might be desirable to fully exploit the information captured by the network during the training stage. In view of the application of the network to an unknown mapping, these pieces of information may be helpful to get better insights into the problem at hand. To this aim we devote the first part of the paper to inquiry on the relationships between the statistical information extracted by the network and the characteristics of some test mappings. The very notion of a perceptron as a device sensitive to statistical features alludes to the capability of feed-forward nets to record information in the form of Bayesian probabilities [3–5]. However, storage of probabilistic information can be perturbed by a kind of noise (R noise) that was not taken into account so far. We suggest that for patterns substantially affected by R noise, the network can be likened to a random classifier as far as the efficiency of prediction is concerned. The performance can be evaluated by modeling the computation of each output neuron as a random walk in the space of the weights, where the probability of the individual steps is dictated by the statistical information stored in the weights converging onto the output neuron at hand. Under these conditions the perceptron is said to operate in randomlike mode. The main part of the paper is devoted to the full characterization of the randomlike behavior of a perceptron as it is faced with a noisy mapping.

A case in point is the primary-to-secondary structure mapping of proteins that is studied by means of perceptrons in the second part of the present work. As a matter of fact, a more specific motivation for the present investigation is the urgent need for a clarification of the limiting factors that affect the efficiency of prediction of protein secondary structures [3,6–8]. Some puzzling problems arise in the context of this application and demand proper explanation: on the one hand, the finding that perceptrons are as effective as predictors based on statistical methods and, on the other hand, the sensitivity of the network to the number of examples in each class (i.e., approximately the frequency of occurrence of inputs in each class) rather than to specific patterns.

The plan of the paper is the following. In Sec. II we describe the mapping the network is expected to learn. In Sec. III we classify the sorts of noise that potentially affect the unknown mapping. We then explain the appearance of a randomlike component that in Sec. IV is modeled as a set of concurrent random walks and arrive at an analytical expression for the network to predict each class. In Sec. V we specialize our considerations to the prediction of secondary structures of proteins [7,9–11]. Data from our previous experiments in this area are then used to make a semiquantitative check of the theoretical predictions of Sec. IV. The results of our simulations suggest the introduction of an entropylike measure of the single-pattern ambiguity, as well as a global measure of intensity of the noise affecting the sequence-to-structure mapping. Finally, in the conclusive section, we point out the bearings of our model on the folding code and the folding mechanism of globular proteins. Moreover, the limited performance of the network is traced back to the intensity of noise and to the noise-induced randomlike behavior of the perceptron. This allows us to draw conclusions as to the optimal learning strategy in the presence of noise.

II. GENERAL FEATURES OF THE CLASSIFICATION TASK

The general task we are considering consists of the reconstruction of a mapping which classifies symbolic patterns into the appropriate class. The classification discriminates among n_{cl} classes which will be labeled with greek letters α, β, γ, ... More formally, the mapping $\mathcal{M}_W$ consists of the set of associations $P_j \rightarrow c_k$ from a space of objects (also referred to as input patterns) $\mathcal{P} = \{P_j\}$ to an n_{cl}-dimensional space of classes $\mathcal{C}=\{\alpha,\beta,\gamma,\ldots\}$. The objects P_j are strings of W letters drawn from an alphabet $\mathcal{A}$ of N_S symbols; formally, $P_j=\{l_j^1,l_j^2,\ldots,l_j^W\}$. By a subpattern of P_j we mean any subset of P_j; patterns P_k and P_l sharing subpatterns of any length, i.e., such that for some $i, l_k^i = l_l^i$, are said to be overlapping.

The class c_k refers to the letter falling in the central position within the input window of size W. The training set $\mathcal{M}_W$ is given in the form of two corresponding sequences $S_P=\{l_1,l_2,l_3,\ldots\}$ $(l_j \in \mathcal{A})$ and $S_C=\{c_1,c_2,c_3,\ldots\}, c_i \in \mathcal{C}$. The input window is shifted letter by letter until the whole sequence S_P has been scanned. In any case the theory developed in the sequel is by no means restricted to this specific rule of pattern production. As far as the single-letter code is concerned, there exists a 1:1 correspondence between any letter l_i and the components of an N_S-dimensional binary vector, having all components set to zero but the one corresponding to the desired letter (orthonormal code). This code lends itself to creating an unbiased correspondence between the set of the possible letters and a set of labels each having non-nil overlap only with itself. On the whole there are $W \times N_S$ input neurons that take discrete values $\{0,1\}$ according to whether they are activated or not. Each class is represented by a single neuron in the output layer. Output neurons have continuous and real-valued activations ranging in]0,1[; collectively they form an n_{cl}-dimensional output vector $\vec{o}=(o_1,o_2,\ldots,o_{n_{cl}})$ (see the Appendix). We use the winner-take-all strategy (see the Appendix) to extract the final classification from the actual output of the network. The weights of the network are randomly initialized and iteratively corrected by the standard error backpropagation algorithm (see the Appendix). These specifics are sufficient to introduce the general arguments and the model of Sec. III and Sec. IV. For a more complete description of the architecture of the network within the context of the prediction of protein secondary structures, the reader is referred to the Appendix.

III. CLASSIFICATION OF NOISE

To make this paper self-contained, it is convenient to summarize the results of our previous work. Following [3] we distinguish two kinds of noise which we term intrinsic noise (I noise) and representational noise (R noise). I noise arises when the training set contains nonoverlapping ambiguous patterns, i.e., patterns that on distinct occurrences are classified in different classes. This kind of noise is due to the inherent ambiguity of the mapping $\mathcal{M}_W$ (the supervisor). I noise usually reflects the inadequate size of the input window such that typical markers of some patterns are missed.

R noise is a side effect of the single-letter orthonormal input code in that it arises as a consequence of the overlap among patterns associated with different classes. Therefore it happens that the same subpattern is alternatively classified in different classes as it occurs within different input patterns. Concisely, R noise is none other than I noise that affects one or more subpatterns rather than the whole pattern.

It follows that on processing an input pattern affected by R noise (ambiguous pattern), the perceptron has to weigh the simultaneous contributions of subpatterns that are no longer unanimous in indicating the same class. This kind of ambiguity originates eventually the randomlike behavior of the network that is described in Sec. III A. Clearly, a prerequisite for R noise to be present is the mixing of subpatterns; in the limit of increasing R noise intensity, correlations among the letters forming the input patterns are progressively weakened (see, for instance, Sec. V). Let us remark that there may be overlap without R noise, when all the subpatterns point to the same class. Finally, it is worth noting that occurrence of R noise depends on the input code: actually, one can always devise a new input code that nullifies overlap between the representations of any two patterns which, consequently, might be affected only by I noise. However, it would be erroneous to conclude that it is desirable to get rid of R noise; overlap, in fact, is the very basis of the generalization capability of perceptrons (as well as of other sorts of neural networks). In the following, for want of any specifications, we shall use the term noise to refer to the joint effect of I noise and R noise.

A. Noise intensity and perceptron's response

It is clear that were the mapping noiseless and were the weights initially set to zero, the unambiguous rule of association of any pattern P to class c_i would cause the perceptron to generate asymptotically a binary 0/1 output, the only

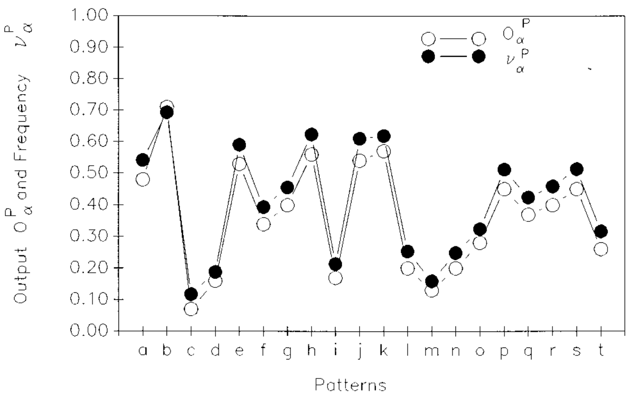

FIG. 1. Comparison of pattern frequencies with the outputs of a perceptron that has completed a 100 cycle training phase with orthonormal code and pattern updating. The plot shows that the activation levels o^P_α (○) of a generic output neuron α asymptotically approach the relative frequencies ν^P_α (●) of each input pattern P. The training set comprises 20 different patterns that are conventionally indicated with letters on the abscissa. Analogous behavior is exhibited by all of the output neurons, irrespective of their number.

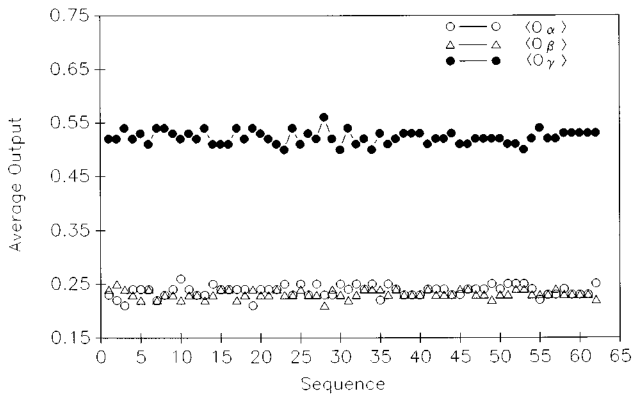

FIG. 2. Dependence of the average outputs on the composition in classes of the training set. The plot shows the output of a three-output perceptron upon presentation of the patterns of the training set, after completion of the learning phase on a random mapping (following the usual procedure described in the Appendix). To perform this test we broke up the original letter sequence S_P into 62 traits (indicated with numerals on the abscissa) and calculated the average activation $\langle o_i \rangle$, $i=\alpha,\beta,\gamma$, of the output neurons over each trait (marked, respectively, as ○,△,●). The traits correspond to the 62 proteins composing the training set L_{62} described in the Appendix. The plots show small fluctuations around mean values that closely reflect the probabilities p^A_i with which the random assignment to each class has been made in building up the training set. In this example, $p^A_\alpha=0.25, p^A_\beta=0.22, p^A_\gamma=0.53$. The average values and standard deviations (evaluated over the 62 traits of the training set) turn out to be $\langle o_\alpha \rangle=0.24\pm0.01, \langle o_\beta \rangle=0.23\pm0.01, \langle o_\gamma \rangle=0.52\pm0.01$.

output neuron with unit activation corresponding to class c_i. Distinguishing patterns as ambiguous and unambiguous is feasible if the mapping is known; yet this is usually not the case and, in addition, it may be desirable to replace this all-or-none distinction with a new continuous criterion. The idea is to rank patterns according to the distance of the corresponding outputs from the δ-like output of strictly unambiguous patterns. A reliability scale obtains which is useful to discriminate between reliable and unreliable patterns. Qualitatively, the output vector $\vec{o}=(o_1,o_2,\ldots)$ of a reliable pattern is strongly peaked on any class c_i, i.e., $o_k\approx\delta_{ik}$, whereas the typical output of unreliable patterns exhibits non-negligible spread of the activations on more output neurons. A measure of pattern reliability is introduced in Sec. V and generalized in Sec. V A.

Now we turn to characterizing the modes of operation of the perceptron as a function of the intensities of I noise and R noise. When the mapping is affected by I noise the meaning of the output $\vec{o}$ is susceptible to analytical investigation [4,5,12]. It turns out that under batch updating (see Appendix) the backpropagation algorithm ensures the condition $o^P_i \rightarrow \nu(c_i|P)\approx p(c_i|P)$ where ν and p are the relative frequency and, respectively, the probability of finding pattern P in class c_i. In the rest of the paper we use superscript P whenever it is necessary to emphasize the dependence of the parameter in question on the pattern P. When the corrections of the weights are accomplished according to the alternative pattern updating procedure (see Appendix), the convergence theorem does not apply; nonetheless we have experimentally verified that the learning algorithm still reproduces pattern probabilities (see Fig. 1). Thus, in the event the network is faced with a mapping affected by I noise, we are allowed to define a first mode of operation of the perceptron that will be referred to as Bayesian or pattern-sensitive mode, since the information stored by the network relates to the specific input pattern.

Also pure R noise is the cause of unreliable classifications, although the probabilistic meaning of the network's output cannot be any longer deciphered so easily since R noise interferes with the storage of Bayesian probabilities. As a matter of fact it has been noted that estimation of Bayesian probabilities is better when one output dominates over the others [5] (this is the case of reliable patterns). The perturbing effect of R noise can be seen most clearly in the limiting case of strong R noise. Expectedly, $p(c_i|P)\rightarrow p(c_i)$, where $p(c_i)$ is the probability that class c_i is predicted oblivious of the pattern P.

We can simulate this case by resorting to a random mapping that is defined by associating S_P (see Sec. II) with a random sequence S_C. Randomization has the ultimate effect of maximizing the intensity of R noise. A realization of this experiment on a version of the general problem described in Sec. II is illustrated in Fig. 2, with $\mathcal{C}=\{\alpha, \beta, \gamma\}$. S_P is the set of amino acid sequences introduced in Sec. V, where it is argued that it provides a well mixed set of symbolic subpatterns. S_C is generated by sampling the set $\mathcal{C}$ of the possible classes according to assigned *a priori* probabilities $p^A_\alpha, p^A_\beta, p^A_\gamma$. First, we order sequentially the input patterns by means of a label λ, and plot the outputs o^λ_i as a function of λ. If we now smooth out the resulting rugged curve by calculating local averages $\langle o_i \rangle$, we get the plot of Fig. 2. The interesting outcome is that although the network is no longer able to record any statistical regularities per individual pattern, $\langle o_i \rangle$ reflect the extant piece of information carried by $\mathcal{M}_W$, i.e., the relative abundance of examples per each class in the training set, $p^A_\alpha, p^A_\beta, p^A_\gamma$.

The limiting case just discussed illustrates a mode of classification (randomlike mode) which is antithetical to the Bayesian mode. The most striking departure from the Bayesian mode consists in the network exhibiting, in a sense, insensitivity to the input pattern and a critical sensitivity to the composition in classes of the training set. To define the randomlike mode it is convenient to introduce the notion of a random predictor, that is, a device which makes random classifications according to a probability density function (PDF) independent of the current input. The PDF specifies the probability p_i^H, $i \in [1,n_{cl}]$, with which the predictor associates the input pattern with class c_i.

A consistent definition of the efficiency of the network as a random predictor is the probability Π that the random prediction is correct:

$$\Pi = \sum_{i=1}^{n_{cl}} p_i^A p_i^H, \tag{1}$$

where the p_i^A, $i \in [1,n_{cl}]$ specify the actual distribution of structures in the test set. Clearly, Eq. (1) provides an alternative expression of the prediction score Q whose general definition (see the Appendix), at any rate, applies to any kind of predictor.

Following [3] we argue that the perceptron working on a random mapping is equivalent to a random predictor, in some average sense to be soon discussed. Strictly speaking, this equivalence is nonsense since the network is deterministic; instead, it makes sense with the proviso that it holds on average and from the point of view of the success score Q [expressed as in Eq. (1)].

To make this equivalence clear let us suppose that many different training sets are formed by changing the individual patterns, but keeping fixed their composition $p_\alpha^A, p_\beta^A, p_\gamma^A$ and the total number of patterns N. Now we train the perceptron on them separately and measure the efficiency Q on the same testing set. Changing the training set hardly affects the Q value of the peceptron, provided the learning set is sufficiently large [5,7]. Otherwise stated, Q will undergo minor fluctuations around an average value; however, and here lies the insensitivity to the input pattern, the unreliable patterns that are properly classified or misclassified vary upon changing the training set. The reason for this is that R noise has an unpredictable (i.e., training-set-dependent) influence on the classification of the unreliable patterns. In conclusion, only Q and the number and identity of the reliable patterns are invariant with respect to the particular choice of the training set. On the contrary, as far as the unreliable patterns are concerned, only the number of the correctly classified inputs is constant on average. This implies that the exact course of the unaveraged and rugged curve from which Fig. 2 has been derived is unpredictable, whereas the smoothed curve preserves the same global information irrespective of the training set. For a perceptron the probabilities p_i^H are not given *a priori* but are the outcome of the learning process. The experiment on the random mapping suggests that there should be some nonlinear function Φ_i relating p_i^H to the composition of the training set, i.e., $p_i^H = \Phi_i(p_\alpha^A, p_\beta^A, p_\gamma^A)$. In Sec. IV we make assumptions on the randomlike mode of the perceptron in order to estimate Φ_i.

The experiment illustrated in Fig. 2 makes it apparent that the perceptron trained on the random mapping exhibits a purely randomlike behavior and that its efficiency turns out to conform to Eq. (1). Under these conditions we have ascertained that not only $\langle o_\gamma \rangle > \langle o_\alpha \rangle, \langle o_\gamma \rangle > \langle o_\beta \rangle$ but also $o_\gamma^P > o_\alpha^P, o_\gamma^P > o_\beta^P, \forall P$. This implies that the perceptron classifies systematically the input in the most frequent class γ, that is to say $p_\gamma^H = 1$; from Eq. (1) it ensues that $\Pi = p_\gamma^A = 0.53$. On the other hand, the same value obtains if we calculate the value of Q as the fraction of correct guesses (see Appendix).

For mappings that are partially affected by R noise it is reasonable to think that the overall behavior of the network is a hybrid of the Bayesian and the randomlike component. The Bayesian mode prevails in the classification of the N_r reliable patterns, on which noise has only a minor effect, while the randomlike mode takes over when the perceptron deals with the N_u unreliable patterns. Clearly $N = N_r + N_u$, where N is the total number of patterns in the test set. To the aim of splitting the efficiency Q in the contributions corresponding to the two modes, we slightly modify the previous partition $N = N_r + N_u$ of the N patterns into a new partition $N = N^+ + N^-$. N^+ and N^- are defined as the patterns that are correctly predicted with unit probability and, respectively, with probability $\Pi < 1$. The two partitions are quite closely related to each other and would exactly coincide only in the limiting case of zero noise. This is suggested by the finding that patterns with increasing reliability are more and more surely assigned to the correct class, as is well exemplified below, in Table I. We take it that posing $N^+ \approx N_r$ is quite a good approximation since the discrepancies between the two partitions involve a tiny fraction of the whole set of patterns. By way of example, R noise may lead to the inclusion of incorrect reliable patterns among the N_r reliable ones; this is likely to occur when the said test patterns have significant overlap with patterns of the training set belonging to incorrect structures. In the new partition this small set of patterns is removed from N_r and categorized in N^-. Similarly, the set of unreliable and correct patterns is shifted from the set of the N_u to the set of the N^+ patterns. With the aid of the new partition we represent the behavior of the network as a statistical mixture of the two modes;

$$Q = \frac{N^+ + N^- \Pi}{N}, \tag{2}$$

where Π is the probability that the N^- patterns are correctly classified when the perceptron operates as a randomlike predictor. Equation (2) shows that both the randomlike component and the Bayesian component contribute to the efficiency of the perceptron in varying proportion according to the level of noise. With maximum R noise, as in the conditions of the random mapping, the Bayesian component is entirely supplanted by the random component [$Q \rightarrow \Pi$ as $N^+ \rightarrow 0$ in Eq. (2)], whereas at zero R noise the Bayesian component dominates ($Q \rightarrow 1$ as $N^+ \rightarrow N$). It is clear that for Eq. (2) to be strictly valid we must consider the partition N^+/N^-: anyhow, in the sequel, we find it more convenient to use the approximation $N^+ \approx N_r$ since dealing directly with reliable or unreliable patterns makes it easier reasoning on the mechanism of computation in the presence of noise.

IV. A STOCHASTIC MODEL FOR THE QUASIRANDOM MODE OF THE PERCEPTRON

The PDF $\{p_i^H\}$ characterizing the randomlike component of the perceptron establishes by successive approximations during the learning stage. The main goal of this section is the calculation of the p_i^H. This will be done by proposing a stochastic mechanism that simulates the randomlike mode of the perceptron as it classifies unreliable patterns.

We maintain that the N_u patterns are exclusively contributed by the strong mixing of subpatterns. The ground for this is that under the typical working conditions, where the generalization capabilities of perceptrons are stressed, the input window size W is much larger than the correlation length among subpatterns. Then we are allowed to think of the unreliable input patterns as being made up of noncorrelated subpatterns. Accordingly, upon presentation of an unreliable pattern, we consider the n weights w_{ij} $(n=W)$ contributing to the activation of the ith output neuron, as a random sample where the probability that w_{ij} is picked up is specified by a distribution function $f_i(w_{ij})$; the distributions f_i and f_k $(i\neq k)$ are assumed to be independent. Thus the input-dependent part of the local field of each output neuron (see Appendix) can be conceived as a random flight in the space of the weights terminating on the said neuron. The independent random walks provide therefore an effective mechanism for the synthesis of the N_u patterns and, to a good approximation, of the N^- patterns.

The probabilities $f_i(w_{ij})$ are approximated by the histograms of the weights w_{ij} linking the jth input neuron to the ith output neuron. This approximation is mitigated by the consideration that the precise analytical form of the functions $f_i(w_{ij})$ is immaterial to our model since, as we shall see below, only the first two moments enter explicitly the final expression of $\{p_i^H\}$.

The next step is the calculation of the sum F_i of the n weights for each class,

$$F_i=\sum_j w_{ij}, \tag{3}$$

that, in virtue of the binary input, represents the local field of the ith output neuron plus the threshold θ_i [see Eq. (A1) in the Appendix]. To this aim we think of F_i as a random walk of n steps, whose magnitudes are chosen according to the probability $f_i(w_{ij})$, with average $\langle w_{ij}\rangle$ and variance σ_i^2. Then the pertinent expression for the probability that the variable F_i takes on the value W_i is [13]

$$\mathrm{Prob}\{F_i\in[W_i,W_i+dW_i]\} =(2\pi n\sigma_i^2)^{-1/2}\exp\left\{\frac{-(W_i-n\langle w_{ij}\rangle)^2}{2n\sigma_i^2}\right\}. \tag{4}$$

Therefore, assuming mutual independence of the variables W_i, the joint probability that n weights sum to W_i $(i\in[1,n_{cl}])$ is

$$p_0(\vec{W})=\prod_{i=1}^{n_{cl}}(2\pi n\sigma_i^2)^{-1/2}\exp\left\{\frac{-(W_i-n\langle w_{ij}\rangle)^2}{2n\sigma_i^2}\right\}. \tag{5}$$

For a three-output perceptron with output labels $i=\alpha,\beta,\gamma$ the probability that the network selects, say the output α, according to the winner-take-all rule (Appendix) is

$$p_\alpha^H=\mathrm{Prob}\{W_\alpha-\theta_\alpha>W_\beta-\theta_\beta,W_\alpha-\theta_\alpha>W_\gamma-\theta_\gamma\} =\int_{-\infty}^{+\infty}dW_\alpha\int_{-\infty}^{W_\alpha-\theta_\alpha+\theta_\beta}dW_\beta\int_{-\infty}^{W_\alpha-\theta_\alpha+\theta_\gamma}p_0(\vec{W})dW_\gamma. \tag{6}$$

Using Eq. (5), Eq. (6) can be cast in the form

$$p_\alpha^H=\sqrt{\frac{q_\alpha q_\beta q_\gamma}{\pi^3}}\int_{-\infty}^{+\infty}dW_\alpha\exp[-q_\alpha(W_\alpha-A)^2] \times\int_{-\infty}^{W_\alpha-\theta_\alpha+\theta_\beta}dW_\beta\exp[-q_\beta(W_\beta-B)^2] \times\int_{-\infty}^{W_\alpha-\theta_\alpha+\theta_\gamma}dW_\gamma\exp[-q_\gamma(W_\gamma-C)^2], \tag{7}$$

where $A=n\langle w_\alpha\rangle$, $B=n\langle w_\beta\rangle$, $C=n\langle w_\gamma\rangle$, and $q_i=(2n\sigma_i^2)^{-1}$.

A more convenient form of Eq. (7) can be obtained by means of the transformation $W_\alpha-A=\lambda,(W_\beta-B)\sqrt{q_\beta}=\xi$, $(W_\gamma-C)\sqrt{q_\gamma}=\eta$,

$$p_\alpha^H=\sqrt{\frac{q_\alpha}{\pi^3}}\int_{-\infty}^{+\infty}d\lambda\ \exp[-\lambda^2 q_\alpha] \times\left[\frac{\sqrt{\pi}}{2}\mathrm{erf}[(\lambda+A-B-\theta_\alpha+\theta_\beta)\sqrt{q_\beta}] +\int_{-\infty}^{0}\exp(-\xi^2)d\xi\right] \times\left[\frac{\sqrt{\pi}}{2}\mathrm{erf}[\lambda+A-C-\theta_\alpha+\theta_\gamma)\sqrt{q_\gamma}] +\int_{-\infty}^{0}\exp(-\eta^2)d\eta\right]. \tag{8}$$

The calculations for p_β^H and p_γ^H run exactly in the same way with a trivial permutation of the indices.

V. THE CASE OF PROTEIN STRUCTURE PREDICTION

Now we pass to applying the general considerations made so far to the task of predicting the secondary structures of proteins. The sequences S_P and S_C defining the mapping to be learned are the sequence of protein primary structures (residue sequences) and, respectively, the sequence of secondary structures assigned to any residue. The input window can arrange W letters (residues) at a time, and is shifted along S_P as prescribed in Sec. II. Additional details concerning the protein structure task, other than those included in Sec. II, are relegated to the Appendix.

For the purpose of predicting protein secondary structures the size of the input window, usually including 10–20 residues, is normally insufficient to account for long-range interactions which play a significant role in determining the structure of unreliable subpatterns [14,15]. Moreover, we can assume that I noise is negligible because the probability that the same pattern occurs twice is negligible in the available data sets [16].

R noise is well documented in the literature on prediction methods of protein structures (see [17], and references therein) and is connected to the pseudorandom distribution of several properties along the chain [16,18–20]. Hints to the quasirandom characteristics of the sequence of proteins are to be found in the notion that real sequences have been described as ''slightly edited random sequences'' [20]. But quasirandomicity of sequences (caused by strong subpattern mixing) is only a prerequisite for R noise. The next ingredient is the substantial amount of unreliable patterns that, consistently with the definition given in Sec. III, are able to take different conformations depending on the particular environment. Characterization and counting of unreliable patterns, according to different criteria, are to be found, for example, in [14,16]. In [21] experimental evidence is presented that short equivocal sequences exhibit intrinsic preference for a given secondary structure that may be overridden by environmental influences. Such interactions may also be provided by the solvent or by distant residues that turn out to have proximity relationships with the trait at hand within the three-dimensional structure. This is the case of the strong modulation of the β-sheet propensity by the tertiary context [22]. Finally, R noise is signalled indirectly by the finding that the protein structure mapping is characterized by weak correlations. This is suggested by the phenomenological evidence that multilayer perceptrons are no better than single-layer perceptrons at classifying protein secondary structures [23,24].

Different degrees of unreliability of the patterns can be distinguished by means of the reliability index ρ introduced in [6] in the framework of a neural network approach to protein secondary structure prediction. For each pattern P it is a real number ranging in the interval [0, 1[and is defined as the absolute value of the difference between the two highest outputs o_1^P and o_2^P:

$$\rho=|o_1^P-o_2^P|. \qquad (9)$$

In this section our aim is to proceed to a semiquantitative test of the model of Sec. IV by using our predictions of α helices, β sheets, and random coils [7]. The weights of the perceptron after completion of the training stage (under the working conditions described in the Appendix) are used to construct three histograms, each one collecting the weights converging on each output neuron (Fig. 3).

From the histograms the parameters of the probability function $f_i(w_{ij})$ used in Eq. (4) are easily derived: $\langle w_\alpha\rangle=0.10$, $\sigma_\alpha^2=0.37$, $\langle w_\beta\rangle=0.01$, $\sigma_\beta^2=0.33$, $\langle w_c\rangle=-0.16$, $\sigma_c^2=0.35$. The thresholds θ_i have the following values: $\theta_\alpha=3.16$, $\theta_\beta=2.16$, and $\theta_c=-2.77$. Inserting these values in Eq. (8) and its analogs for the other structures, and fixing $n=W=17$ we get $p_\alpha^H=0.223, p_\beta^H=0.116, p_c^H=0.661$. By means of Eq. (1) and the composition in structures p_α^A, p_β^A, and p_c^A of the learning (L) and test (T) sets (see Appendix), we eventually estimate $\Pi_L=0.432$ and $\Pi_T=0.417$. Finally, by using the experimental value [7] of the performance, $Q=0.658$ for $N=11361$ residues of L_{62} (see the first row of Table I) we calculate $N_r=4520$ and $N_r/N=39.8\%$ from Eq. (2). The same procedure with $Q=0.628$ for $N=6634$ residues of T_{33} leads to $N_r=2400$ and $N_r/N=36.2\%$.

At this stage a semiquantitative check of our model of the random component can be made by comparing the above figures with the N_r value obtained by direct counting. In Table I we have sorted the patterns of the training set according to their reliability index ρ defined in Eq. (9) and computed the number of patterns N_k with $\rho\geqslant k$ $(k=0,0.1,0.2,\ldots,0.9)$. Then we have measured the efficiency Q_k by using the perceptron trained on the whole L_{62} to predict the N_k patterns.

Now it has been argued [25] that there exists an upper bound $Q\approx0.88\pm0.09$ beyond which there is no point in further improving on the efficiency of secondary structure prediction since identically folded conformations leave room for about 12% fluctuations in the secondary structures. A reasonable conjecture is that those patterns that are predicted by the network with accuracy $\geqslant0.88$ are sufficiently reliable for them to have strong intrinsic propensity toward the right structure. This property makes them eligible to correspond to the sought-for reliable patterns N_r. Posing $N_r\approx N(Q=0.88)$ in Table I, we get for L_{62} a value $N_r\approx2000$ and for T_{33} $N_r\approx700$, that are in the same range as the above theoretical estimate. Note that the agreement is better if we take into account that perceptrons tested on this same task domain tend to miss about 5% in performance so that the above threshold for Q is to be placed around 0.83 [23].

A. Noise and entropy of the protein structure mapping

In Sec. III A we have shown that the exact probabilistic meaning of the perceptron's output is corrupted by R noise although some remnant of probabilistic information can still be found even in the limiting case of the randomized mapping. Though one expects that the output deviates from $p(c_i|P)$, our experiments on real proteins show that the sum of the activities of the output neurons is very close to unity. The same kind of regularity has been reported in [5] for quite

TABLE I. The patterns of a typical learning set L_{62} and test set T_{33} are here grouped according to their reliability index (ρ) [see Eq. (9)]. N indicates the number of patterns with ρ not smaller than the value in the first column. Q is the efficiency of prediction of each subset of patterns when the perceptron is trained on L_{62}.

	L_{62}		T_{33}	
ρ	Q	N	Q	N
0	0.658	11361	0.628	6634
0.1	0.695	9608	0.660	5610
0.2	0.730	8043	0.690	4717
0.3	0.763	6600	0.719	3909
0.4	0.792	5311	0.739	3193
0.5	0.820	4140	0.769	2449
0.6	0.847	3010	0.788	1805
0.7	0.872	1985	0.815	1219
0.8	0.909	1126	0.869	688
0.9	0.940	361	0.942	224

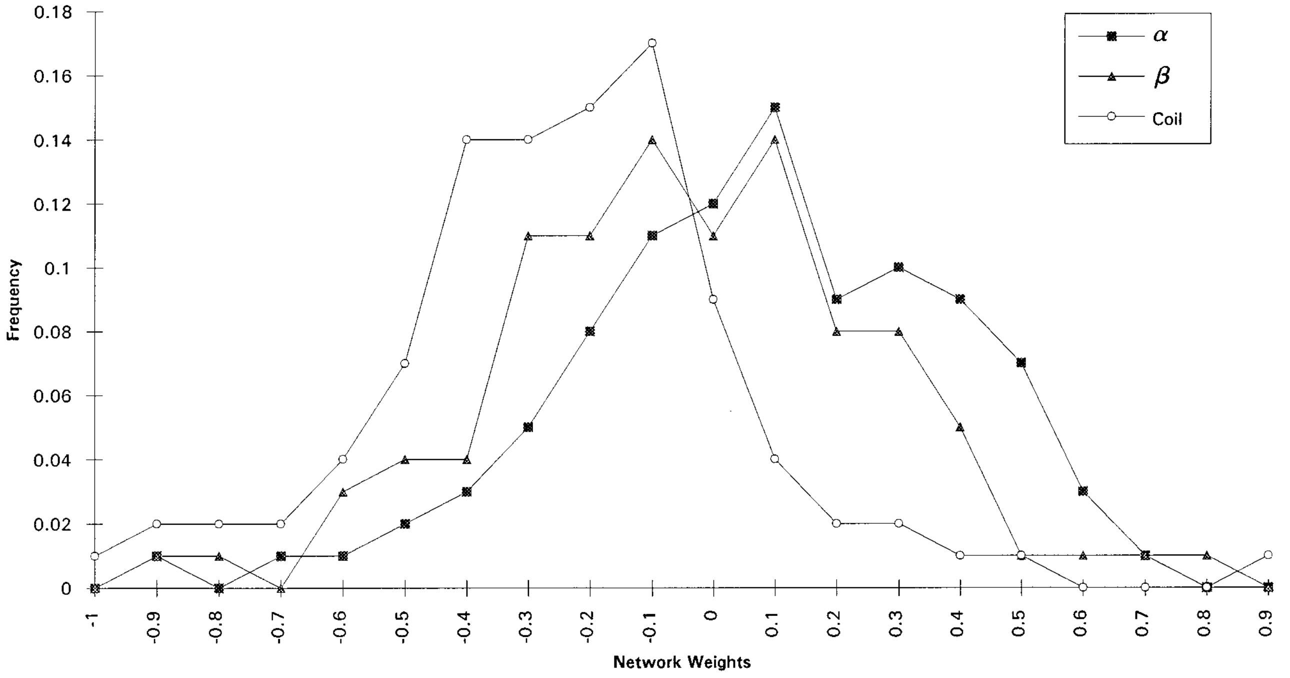

FIG. 3. Normalized histograms of the weights converging on each of the three output neurons of the perceptron used to predict α, β, and coil protein structures.

different task domains and network architectures. This is remarkable since there is no explicit constraint for this to occur. The averaged sums of the outputs for our standard training and test sets L_{62} and T_{33} (see the Appendix) are, respectively, 1.05 ∓ 0.11 and 1.09 ∓ 0.06 (see Fig. 4).

This property helps in devising an entropylike measure for each pattern P since the set of corresponding outputs can be viewed approximately as a scheme (in information theoretic parlance [26]). The usual expression

$$S^P = -\sum_{i=\alpha,\beta,\gamma} o_i^P \ln o_i^P \tag{10}$$

provides a measure of reliability per single pattern. Averaging S^P over the set of N patterns gives an average entropy of the mapping

$$S = \sum_P S^P/N. \tag{11}$$

S can be viewed as a measure of noise intensity that, in our special condition of nearly zero I noise (Sec. III), can be used as an indicator of R-noise intensity. As a matter of fact, comparing S values of the artificial random mapping and the real protein mapping [7] we verify that $S_{\text{random}}=0.96$ and $S_L=0.67$ and $S_T=0.69$ for L_{62} and T_{33}, respectively. Our entropylike measure of the mapping reconstructed by the perceptron is also a useful indicator of the information extracted by the network during the training stage. Actually it decreases from the initial value $S=0.96$ (of the perceptron with randomly initialized weights) to the final values 0.67–0.69 of S_L and S_T mentioned above. Consistently with its

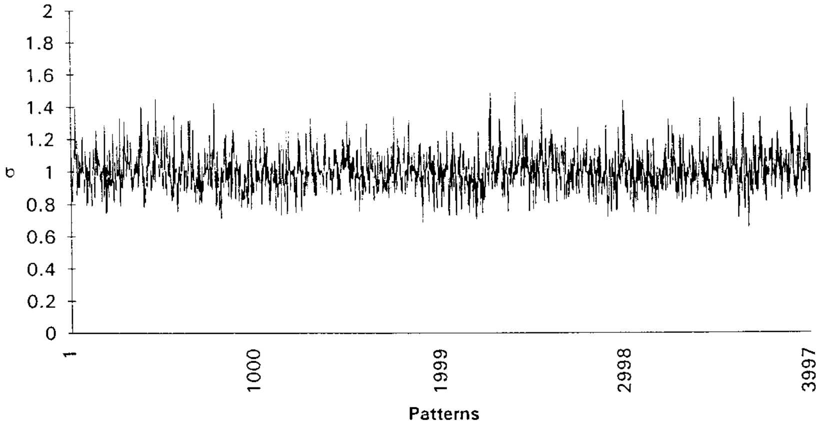

FIG. 4. The activations of the output units practically sum to one even in the presence of R noise. The plot shows the sum of the outputs (σ) for the first 4000 patterns of a three-output perceptron performing on the test set of protein sequences T_{33} (see the Appendix).

meaning, S takes on smaller values for subsets with increasing values of ρ. For instance, the small subset with $\rho = 0.9$ (see Table I) has $S = 0.30$.

VI. DISCUSSION

The present investigation focuses on the behavior of perceptrons used to study noisy mappings within the general framework specified in Sec. II. Our main scope is to make more quantitative the general notion that noise blurs the information carried by each pattern and eventually leads to a decline of the efficiency of classification. We have explored the effect on the performance of the kind of noise that arises when patterns, belonging to different classes, exhibit significant overlap with pronounced mixing of subpatterns. The major result of the present paper is the definition of a randomlike regime of the perceptron, which is useful to estimate the performance as the network is confronted with noisy (unreliable) patterns. As far as the prediction score is concerned, we have argued that in the presence of high noise the perceptron works in randomlike mode and the deterministic computation performed by each output neuron is equivalent to a one-dimensional random walk on the available connections. The ideal conditions for observing the stochastic regime of the network are reproduced in the experiment of Fig. 2 where a perceptron has been trained on a randomized mapping.

A suitable benchmark for the stochastic model proposed in Sec. IV is provided in Sec. V by our experiments of protein structure prediction. The effect of noise on the performance Q is most clearly seen by comparing the scores as the perceptron is tested on the same task but with different noise intensities. Actually, the performance in the task of predicting real protein structures (see Table I) is $Q_{\mathrm{real}} \approx 0.63 - 0.66$, whereas in the prediction of the random mapping of Fig. 2 it reduces to $Q_{\mathrm{random}} = 0.53$. The introduction of an average entropylike estimate of noise intensity in Sec. V A is a first step toward a quantitative correlation of noise and performance. The average entropy S is derived from the pattern entropy S^P that is reminiscent of similar structural entropies used in the literature (see, for example, [14]). The interesting novelty is that S^P is directly evaluated from the output of the perceptron. The reduction from Q_{real} to Q_{random} correlates with the increase of noise from $S_{\mathrm{real}} = 0.67 - 0.69$ to $S_{\mathrm{random}} = 0.96$. It is apparent that the unreliable patterns are mostly responsible for the amount of noise of the mapping; accordingly, they correspond to noncoding patterns with high pattern entropy S^P and cause the folding code to be fuzzy and nonlocal. It is a distinctive feature of noncoding patterns that local interactions are of minor importance and that their structure is essentially determined by the tertiary context.

In Eq. (2) the overall performance of the perceptron is split in two contributions and we have taken advantage of this representation to evaluate the number N_r of reliable patterns (by virtue of the approximation $N_r \approx N^+$). Before discussing our estimates let us posit that we feel confident of our results although it might seem questionable that we draw conclusions about the sequence-to-structure mapping on the ground of speculations made on the simplest possible feed-forward neural network. However, we maintain that the inferences made in the present paper are not seriously limited in generality since the perceptron provides the maximum possible information that can be extracted from the single-residue sequence. Actually, in the structure classification task, multilayer perceptrons either hand designed or starting from scratch and statistical approaches do not improve on the performance of the perceptron [8,23,24].

Some comments are in order as to the result of Sec. V, $N_r / N = 35 - 40\%$, as well as to the role of reliable patterns in protein folding. First we note that it is reasonable to identify the reliable patterns with the coding residues of a protein that have been searched for in different works on the folding code. In [14] it has been estimated that 60–70 % of the four residue patterns are coding. The apparent discrepancy with our range is mainly due to the larger probability that noise affects the pattern owing to our larger input window. A further comparison can be made with the finding of simulations of the folding process [27]. The estimated value of 25% for the fraction of residues that presumably stabilize the folding nucleus compares satisfactorily to our estimate; this might hint at the possible involvement of reliable patterns in the formation of the folding nucleus, to the extent that this accompanies fast formation of elements of secondary structure [28]. This is, however, an open problem which needs more work to be definitely answered [28], although the event that reliable patterns take part in the early stages of folding is consistent with their strong intrinsic propensities for the secondary native structure.

We are now in a position to answer the question whether the essential uncertainty that affects the prediction of protein secondary structures is due to the inadequacy of the neural network approach or to intrinsic deficiencies of the mapping under study. The above considerations about noise suggest that there is an unavoidable upper bound to the prediction of structures that is inherent to the mapping based on local information and cannot be overcome by merely enlarging the data set. Our estimates of the p_i^H indicate that the randomlike component of the perceptron is polarized towards the most abundant class (random coil). This reproduces the experimental observation that perceptrons working on the protein structure task overestimate the most abundant class (random coil) and underestimate the less frequent classes (α helix and β sheet) [7]. Thus it appears that the randomlike component of a perceptron trained on a real data set, similarly to the perceptron trained on a noisy mapping, is driven by generic pieces of information such as the composition in structures of the training set.

Finally, our analysis gives useful suggestions as to the best conditions for studying noisy mappings. In particular, we are able to rationalize the common strategy to use balanced training sets [7,25]. Actually, using training sets with quite similar fractions of the different structures amounts to reducing the effectiveness of the randomlike component and to favor the emergence of reliable patterns. The overall efficiency Q may happen to slightly decline while the number of correct guesses is much more uniformly distributed among the classes. Correspondingly, the overestimation of the more abundant class and underestimation of the less abundant structures is sensibly reduced [7]. This improves in particular the prediction of the underrepresented β patterns.

APPENDIX

In this appendix we give the full details concerning the perceptron architecture as well as the data base used to cope with the particular task of protein structure prediction illustrated in Sec. V. In this task domain the mapping $\mathcal{M}_W$ goes from the space of primary sequences (residue sequences) to the space of secondary structures described in terms of the three major classes α helix, β sheet, and random coil ($n_{cl}=3$). The input code envisages now one letter for each residue of the primary structure. For proteins the natural choice for the orthonormal binary code (see Sec. II) is $N_S=20$, each position within the 20-tuple corresponding to one of the 20 possible amino acid residues. The input layer is designed to read patterns of 17 residues at a time ($W=17$) and contains $17\times 20=340$ input neurons.

The relevant information on the 62 proteins of the training set L_{62} is drawn from the Brookhaven Protein Data Bank. For the sake of having results as homogeneous as possible to those reported in the literature, the set has substantial overlap with the sets used in most of the papers devoted to the prediction of protein structures. The labels of the individual proteins are listed in [7]. The pertinent information on each protein includes the residue sequence and the atomic coordinates of the crystallized protein. Assignment of the secondary structure of each protein is done by processing the amino acid sequence with the DSSP program described in [29].

The learning set L_{62} has 11 361 residues and the following composition in structures: $p_\alpha^A = 0.25$, $p_\beta^A = 0.22$, and $p_{\rm coil}^A = 0.53$. The testing set T_{33} comprises 6634 residues, with $p_\alpha^A = 0.28$, $p_\beta^A = 0.22$, and $p_{\rm coil}^A = 0.50$. The feed-forward network we have used is a perceptron with a single layer of adjustable weights. The output layer has three real-valued neurons with activation values in the range]0,1[, each coding for one of the three structures predicted. The nonlinear transfer function associated with each output neuron c is $(1+e^{-a_c^P})^{-1}$, where a_c^P is the local field defined by

$$a_c^P=\sum_{i=1}^{WN_S} w_{ci}I_i^P-\theta_c, \tag{A1}$$

I_i^P being the activation value (either 1 or 0) of the ith input neuron corresponding to pattern P and θ_c the adjustable threshold of the cth output neuron. The output turns out to be $o_c^P=f(a_c^P)$.

The output is interpreted according to the winner-take-all rule; this amounts to identifying the prediction of the network with the class corresponding to the output neuron with the highest activation. The learning algorithm is the standard backpropagation [30] which is operated with a learning rate $\delta=0.01$. Initial values of the weights are chosen randomly from the uniform distribution in the range $[-10^{-2},10^{-2}]$.

Two alternative learning strategies are usually envisaged. The first scheme (batch updating) prescribes that the weights undergo a one-shot change each time the network has scanned the whole training set. The function to minimize is the cumulative deviation from the desired output d_c^P,

$$E=\sum_p E(P)=\sum_P\sum_c (o_c^P-d_c^P)^2. \tag{A2}$$

The second scheme dictates that the weights are corrected after each pattern presentation (pattern updating), the error function being the function $E(P)$ of Eq. (A2). The latter procedure was used in the predictions of protein structures mentioned in Sec. V. The learning process is stopped when the fractional change of the error function per cycle is less than 5×10^{-4}. Throughout this paper we measure the performance of the network in terms of Q, defined as the ratio of the correct guesses to the total number of patterns N classified.

[1] T.L.H. Watkin, A. Rau, and M. Biehl, Rev. Mod. Phys. **65**, 500 (1993).

[2] B. Cheng and D.M. Titterington, Stat. Sci. **9**, 2 (1994).

[3] M. Compiani, P. Fariselli, and R. Casadio, Int. J. Neural Syst. Suppl. **6**, 195 (1995).

[4] D.W. Ruck, S.K. Rogers, M. Kabrisky, M.E. Oxley, and B.W. Suter, IEEE Trans. Neural Netw. **1**, 296 (1990).

[5] M.D. Richard and R.P. Lippmann, Neural Comput. **3**, 461 (1991).

[6] B. Rost and C. Sander, J. Mol. Biol. **232**, 584 (1993).

[7] P. Fariselli, M. Compiani, and R. Casadio, Eur. Biophys. J. **22**, 41 (1993).

[8] R.C. Yu and T. Head-Gordon, Phys. Rev. E **51**, 3619 (1995).

[9] M. Compiani, P. Fariselli, and R. Casadio, in *Proceedings of the Workshop on Parallel Architectures and Neural Networks*, edited by E.R. Caianiello (World Scientific, Singapore, 1991), pp. 227–237.

[10] J.D. Hirst and M.J.E. Sternberg, Biochemistry **31**, 7211 (1992).

[11] M. Compiani, P. Fariselli, and R. Casadio, in *Neural Networks in Biomedicine*, edited by F. Masulli, P.G. Morasso, and A. Schenone (World Scientific, Singapore, 1994), pp. 313–332.

[12] M. Compiani, P. Fariselli, and R. Casadio, in *Applications and Science of Artificial Neural Networks II*, edited by S.K. Rogers and D.W. Ruck, SPIE Proc. Vol. 2760 (SPIE, Bellingham, WA, 1996), pp. 597–607.

[13] S. Chandrasekhar, Rev. Mod. Phys. **15**, 1 (1943).

[14] S. Rackovsky, Proc. Natl. Acad. Sci. USA **90**, 644 (1993).

[15] N.D. Socci, W.S. Bialek, and J.N. Onuchic, Phys. Rev. E **49**, 3440 (1994).

[16] M.J. Rooman and S.J. Wodak, Nature (London) **335**, 45 (1988).

[17] M.J. Sippl, J. Mol. Biol. **213**, 859 (1990).

[18] P. Zielenkiewicz, D. Plochocka, and A. Rabczenko, Biophys. Chem. **31**, 139 (1988).

[19] S.H. White and R.E. Jacobs, Biophys. J. **57**, 911 (1990).

[20] O.B. Ptitsyn and M.V. Volkenstein, J. Biomol. Struct. Dynamics **4**, 137 (1986).

[21] L. Zhong and W.C. Johnson, Jr., Proc. Natl. Acad. Sci. USA **89**, 4462 (1992).

[22] D.L. Minor, Jr. and P.S. Kim, Nature (London) **371**, 264 (1994).
[23] N. Qian and T.J. Sejnowski, J. Mol. Biol. **202**, 865 (1988).
[24] F. Vivarelli, G. Giusti, N. Villani, R. Campanini, P. Fariselli, M. Compiani, and R. Casadio, Comput. Appl. Biosci. **11**, 253 (1995).
[25] B. Rost, C. Sander, and R. Schneider, J. Mol. Biol. **235**, 13 (1994).
[26] A.I. Khinchin, *Mathematical Foundations of Information Theory* (Dover, New York, 1957).
[27] V.I. Abkevich, A.M. Gutin, and E.I. Shakhnovich, Biochemistry **33**, 10 026 (1994).
[28] A.M. Gutin, V.I. Abkevich, and E.I. Shakhnovich, Biochemistry **34**, 3066 (1995).
[29] W. Kabsch and C. Sander, Biopolymers **22**, 2577 (1983).
[30] D.E. Rumelhart, G.E. Hinton, and R.J. Williams, Nature (London) **323**, 533 (1986).