Journal of Computer-Aided Molecular Design, **12:** 53–61, 1998.
KLUWER/ESCOM

Predicting nucleic acid torsion angle values using artificial neural networks

M.L.M. Beckers, W.J. Melssen & L.M.C. Buydens*
Laboratory for Analytical Chemistry, Faculty of Science, University of Nijmegen, Toernooiveld 1, 6525 ED Nijmegen, The Netherlands

Received 4 June 1997; Accepted 4 November 1997

Key words: artificial neural networks, nucleic acids, prediction, torsion angles

Summary

By means of an error back-propagation artificial neural network, a new method to predict the torsion angles χ, ζ and α from torsion angles δ, ϵ, β and γ for nucleic acid dinucleotides is introduced. To build a model, training sets and test sets of 163 and 81 dinucleotides, respectively, with known crystal structures, were assembled. With 7 hidden units in a three-layered network a model with good predictive ability is constructed. About 70 to 80% of the residuals for predicted torsion angles are smaller than 10 degrees. This means that such a model can be used to construct trial structures for conformational analysis that can be refined further. Moreover, when reasonable estimates for δ, ϵ, β and γ are extracted from COSY experiments, this procedure can easily be extended to predict torsion angles for structures in solution.

Introduction

Several parameters, which define the three-dimensional (3D) spatial structure of nucleic acids, are correlated. Attempts have been made to calculate correlation coefficients between different helical parameters [1, 2] and between pairs of torsion angles [3, 4]. However, multiple correlations between helical parameters and between torsion angles are expected. Therefore, recently multivariate analysis techniques were introduced to specify the relations between torsion angles in nucleic acids [5, 6]. These studies suggest that it is possible to predict certain torsion angles from another set of 'predictor' torsion angles. Which torsion angles are amenable for use as a predictor set?

The most widely used technique to provide 3D structural information about nucleic acids in solution is Nuclear Overhauser Effect (NOE) spectroscopy. Several refinement methods make use of spatial restraints, obtained from NOE experiments, to derive at a spatial model for a molecule. However, NOE spectra sometimes provide too little information to characterize the (fine) structure of certain elements in nucleic acids, e.g., the conformation of furanose rings. J-coupling data may contain additional information in this case. DQF-correlation spectroscopy (DQF-COSY) is nowadays the standard COSY experiment for correlating J-coupled spins [7–9]. By means of the Karplus equations, J-couplings can be translated to torsion angles [10–12]. The backbone torsion angles ϵ, β and γ, as well as the furanose ring, hence torsion angle δ, are amenable to J-coupling. Torsion angles ζ, α and χ cannot be characterized in this manner because the necessary coupling spins are missing. Therefore, it is interesting to investigate whether the former set of torsion angles can be used as predictor variables in a model to predict the latter set of torsion angles.

For this purpose several multivariate calibration techniques, such as Principal Component Regression (PCR) and Partial Least Squares (PLS), are available. A model is sought that predicts the proper output variables for a certain set of input variables. The complexity in the present data set can be modeled with a maximum of only 4 (latent) variables, i.e., there are 4 predictor variables, when using PCR or PLS. If one

*To whom correspondence should be addressed.

uses an artificial neural network for this modeling task, the number of input variables is also 4. However, the proper model to deal with the complexity of the data set can now be fine-tuned by using an extra so-called hidden layer in addition to the input layer and output layer. In particular a so-called error back-propagation network is an appropriate choice. In these kind of networks the error between the predicted variables, that result from the network model, and the actual variables, is back-propagated during a training session. This error determines how weight connections between the network connections, or neurons, should be adapted to produce a proper prediction model.

In this paper it is demonstrated that a simple three-layered error back-propagation network can be trained to find a model that predicts ζ, α and χ from δ, ϵ, β and γ. This procedure is a new method in nowadays conformational analysis. The model can be used to construct an initial structure that can be further refined using traditional methods, such as (restrained) energy minimization. Training sets and test sets were constructed from a dinucleotide data matrix described in a previous paper [6]. This data matrix contains known torsion angles from crystal structures. However, because it is so well defined it provides an excellent means to test the capabilities of an artificial neural network in predicting torsion angles. The error back-propagation network was tested for robustness by initializing the weights several times differently and assembling different training sets and test sets. In all cases acceptable predictive ability was achieved. Hence, the method looks promising in case enough real COSY (in solution) data can be used.

Materials and methods

PCR and PLS

To make a calibration model in which a set of predictor variables, **X**, predicts another set of variables, **Y**, one can assume the model:

$$\mathbf{Y} = \mathbf{Xb} + \mathbf{e}. \tag{1}$$

With a proper calibration method the model parameters, **b**, are chosen such that the residuals (noise), **e**, are minimized and the predictive ability of the model is maximized:

$$\hat{\mathbf{y}} = \mathbf{Xb}, \tag{2}$$

$$\min \sum (\mathbf{y} - \hat{\mathbf{y}}), \tag{3}$$

The estimate of the regression vector, **b**, is calculated using the so-called pseudo-inverse of **X**, which is denoted as $\mathbf{X}^+$, hence:

$$\hat{\mathbf{b}} = \mathbf{X}^+\mathbf{y}. \tag{4}$$

The first step in the process is a singular value decomposition of **X**, i.e., $\mathbf{X} = \mathbf{USV}^\mathbf{T}$. Next a low-dimensional approximation of the data is made so that relevant information is retained while the noise is filtered out. Finally, the pseudo-inverse is calculated as:

$$\mathbf{X}^+ = \overline{\mathbf{VS}}^{-\mathbf{1}}\overline{\mathbf{V}}^\mathbf{T}. \tag{5}$$

The bars over the matrix presentations indicate that the decomposition is truncated to the optimal number of (latent) variables that are used to build the model.

PCR is a method based on Principal Component Analysis. Hence, a linear combination of the original variables is formed, i.e., the latent variables. The regression in PCR is performed on these new (latent) variables. In PLS, however, there is a simultaneous decomposition of the **X** matrix and **Y** matrix (or vector). Then, the coherence in **Y** is used to decompose **X** in PLS (principal) components [13–16].

Error back-propagation network

Artificial neural networks (ANNs) may be subdivided into three basic types, i.e., networks for modeling, self-organization, and optimization, respectively. Obviously, in this paper we search for a model for the relation between a given set of 'input' torsion angles and a set of associated 'output' torsion angles. Therefore, we use a modeling ANN, i.e., a network that is capable of transforming an input pattern to an associated output pattern. Such networks are trained in a supervised manner. This means that they are provided with input-output patterns of which the model that needs to be built is extracted. With such an extracted model the network may be used (after a proper validation procedure) to predict output patterns for given input patterns that were previous unknown for the network. The most simple ANN type for this purpose is the multi-layer feed-forward or error back-propagation ANN [17, 18].

These networks consist of three or more layers of neurons or units. There is one input layer, an output layer and one or more intermediate (hidden) layers. Hence, the network topology is defined by the number of layers and the number of neurons in each layer.

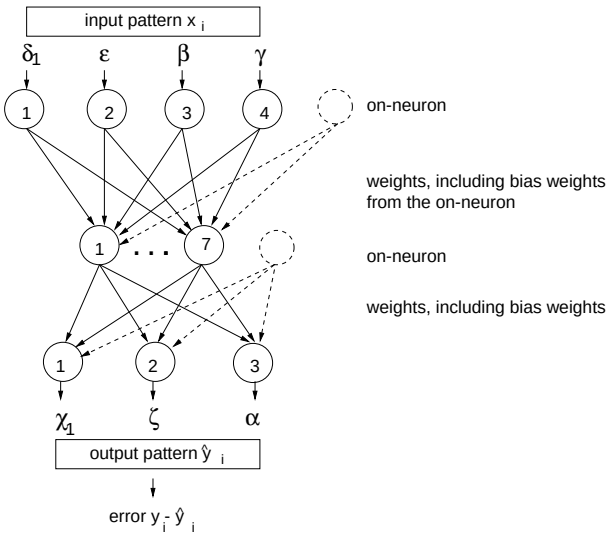

Figure 1. The topology of the error back-propagation artificial neural network that is used in this study.

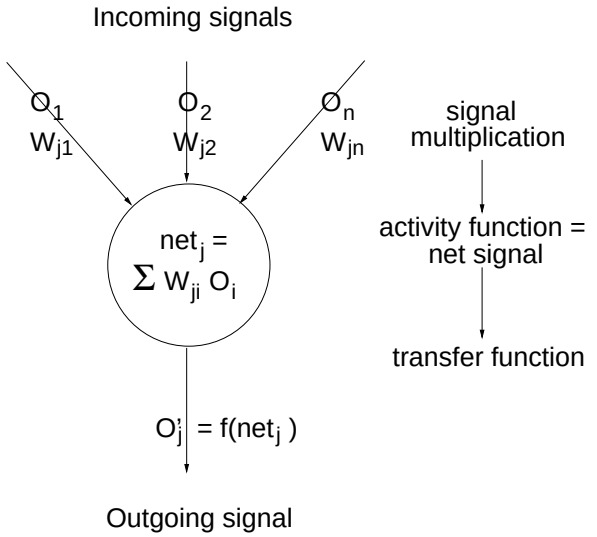

Figure 2. Processing of signals for a single neuron from the hidden layer.

Neurons in successive layers are all interconnected. The propagation of signals from one layer to another is regulated by weights. For this purpose the incoming signals and outcoming signals for a neuron are multiplied with a weight associated with the connection. By means of weights a link between input patterns and associated output patterns is established as depicted in Figure 1. Hence, weights can be thought of to contain knowledge about the input-output relation.

The received weighted signals together determine the net input of the neuron as can be seen in Figure 2. For the neurons in a layer the net input of neuron j is given by

$$\mathrm{net_j} = \sum\nolimits_{\mathrm{i}} \mathrm{w_{ji}} \mathrm{o_i}. \tag{6}$$

Here, the index i refers to the neurons in the previous layer, $\mathrm{w_{ji}}$ is the weight from neuron i to j, and $\mathrm{o_i}$ indicates the output of neuron i. The activity of a neuron is determined by the net input via an activity function. In most networks the activity of neuron j is given by $\mathrm{act_j}$=$\mathrm{net_j}$. In this paper $\mathrm{net_j}$ will be used to denote the activity of neuron j. In its turn the activity of a neuron determines the transmitted signal (output) of the neuron via a transfer function. Examples of transfer functions include a linear function, a threshold function and a sigmoid function. In case of a sigmoid function one obtains

$$\mathrm{o_j} = \mathrm{f}(\mathrm{net_j}) = \frac{1}{1 + \mathrm{e}^{-\mathrm{net_j} + \theta_\mathrm{j}}}. \tag{7}$$

Here, the bias θ_j influences the horizontal offset of the sigmoid. It may be treated as the weight from an additional input neuron (with a fixed output value of 1) to a neuron j.

Training the network

Signal propagation through the network is determined by the weights of the connections between neurons. Initially, weight settings are unknown and hence weights are given random values. Training (or learning) is the process of updating the weights to correct values, i.e., the error between desired output patterns and actual output patterns for given input patterns is minimal. For this purpose we used the back-propagation learning rule [19]. Here, the error, which is the difference between the desired output pattern and the actual output pattern, is a function of the weights. By means of a gradient descence approach, the back-propagation rule tries to locate the minimum error in this weight space. The error (E) is given by

$$\mathrm{E} = \sum \mathrm{E_p},\ \mathrm{E_p} = \frac{1}{2} \sum\nolimits_{\mathrm{j}} (\mathrm{d_{pj}} - \mathrm{o_{pj}})^2, \tag{8}$$

where $\mathrm{d_{pj}}$ and $\mathrm{o_{pj}}$ are the desired output and actual output at neuron j, for pattern p, respectively[1]. The back-propagation rule implies that weights are computed in a backward fashion. Hence, the weights of the output layer are adapted first. Then the weights between neurons in two consecutive intermediate layers are computed and finally the weights in the input layer

[1] $\frac{1}{2}$ is included for mathematical reasons.

are computed. This is described by (t stands for the iteration number)

$$\Delta_p w_{ji}(t+1) = \eta \delta_{pj} o_{pi} + \alpha \Delta_p w_{ji}(t). \quad (9)$$

In fact $\Delta_p w_{ji} = \eta \delta_{pj} o_{pi}$ denotes the adaptation of the weight from neuron i to neuron j in the next layer for pattern p, where o_{pi} is the output of neuron i and η is the learning rate (we will explain the error correction term δ_{pj} later). The learning rate has a large influence on the training procedure. If it is chosen too small, the convergence of the weight set to an optimum will be accurate but very slow. The network might get stuck in a local minimum in this case. On the other hand, if the learning rate is too high, the system might oscillate.

To circumvent this the momentum term α is introduced. For the output layer the error correction term δ_{pj} is given by

$$\delta_{pj} = (d_{pj} - o_{pj}) f_j'(net_{pj}), \quad (10)$$

while for the hidden layer it is given by

$$\delta_{pj} = f_j'(net_{pj}) \sum_k \delta_{pk} w_{ki}. \quad (11)$$

During the training session one might monitor the *recognition* of input-output relations by the network via a Root Mean Squared Error (RMSE) for recognition per variable (for n patterns in the training set)

$$\text{RMSE} = \sqrt{\sum (d_{pj} - o_{pj})^2 / n}. \quad (12)$$

Testing the network

To test the performance of the network, the weights of the training set during a training session can be used to *predict* output patterns for test set input patterns (which are also taken from the original data set). In other words, test set input patterns are fed to the network and the output patterns given by the network are compared with the desired output patterns. Monitoring often occurs with an RMSE for *prediction* (see Equation (12)) for n' patterns in the test set.

Data set

Table 1 summarizes the data that was used in this study. It is based on the study of conformational mobility of dinucleotide steps by El Hassan and Calladine [2]. Corresponding helix types are indicated. Each single strand sequence was subdivided in dinucleotide steps. We describe the dinucleotide steps by the nine

Table 1. Structures from which dinucleotide steps were taken

Sequence	Helix type
Dodecamers	
d(CCGTACGTACGG)	A
d(GCGTACGTACGC)	A
d(CGC**I**AATT**A**GCG)	B
d(CGCGAATTCGCG)	B
d(CGCAAATTTGCG)	B
d(CGCGAATTCGCG)	B
d(CGC**A**AATT**C**GCG)	B
d(CGC**G**AATT**T**GCG)	B
d(CGC**A**AATT**I**GCG)	B
d(CGC**G**AATT**G**GCG)	B
Decamers	
d(CGATCGATCG)	B
d(CGGTATACGC)	A
d(GCGTATACGC)	A
d(GCGTATACGC)	A
d(CGATCGATCG)	B
d(CCAGGCCTGG)	B
d(CCAGGCCTGG)	B
d(CCAA**CI**TTGG)	B
d(CCAA**CI**TTGG)	B
d(CCAA**GA**TTGG)	B
d(CCAACGTTGG)	B
d(CGATTAATCG)	B
d(CGATATATCG)	B
Octamers	
d(GG**I**GC**T**CC)	A
d(GGGTACCC)	A
d(CCCCGGGG)	A
d(GCCCGGGC)	A
d(GCCCGGGC)	A
d(GG**G**GC**T**CC)	A
d(CTCTAGAG)	A
d(GG**UAUA**CC)	A
d(GGG**GT**CCC)	A
d(GGGATCCC)	A
d(GGGGCCCC)	A
d(GTACGTAC)	A
Tetramer	
d(CCGG)	A

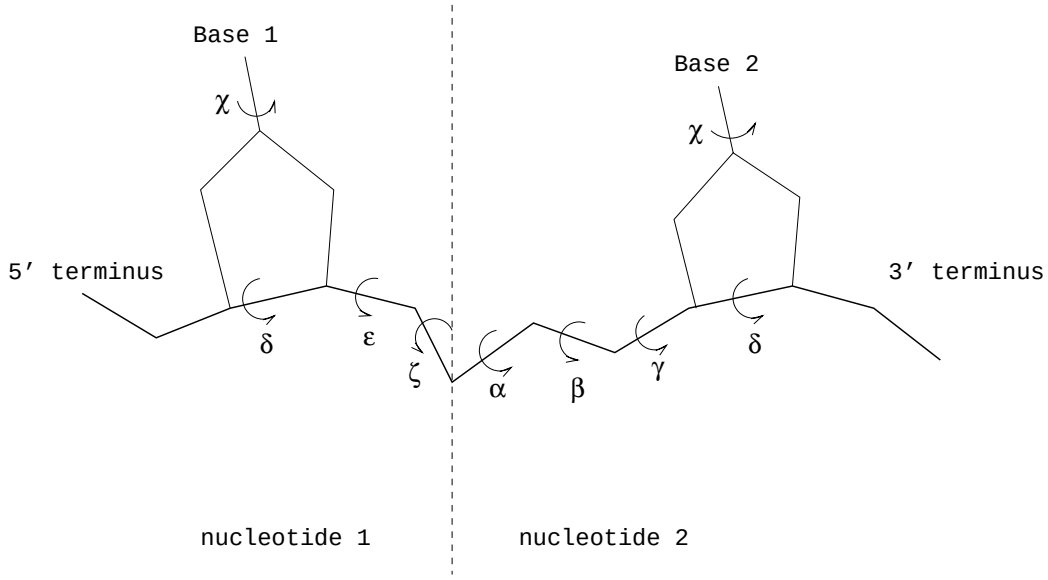

Figure 3. Torsion angle representation of a dinucleotide that is used in this study. The backbone conformation is represented by torsion angles α, β, γ, ϵ and ζ. The conformation of the furanose ring is defined here by δ. The orientation of the base with respect to the sugar ring is given by torsion angle χ.

torsion angles depicted in Figure 3. Hence, the data matrix contains nine columns or variables. The individual dinucleotides, represented by nine variables, are the rows, or patterns of the data matrix. Dinucleotides containing the bases inosine and uracil as well as dinucleotides with bases in mismatched base pairs were not added to the data matrix. They are indicated in boldface in Table 1. Also dinucleotides that had one or more torsion angle combination(s) in forbidden areas, see Reference [12], were not added to the data matrix. This resulted in a total of 244 dinucleotide entries. The first 144 patterns are B-DNA entries. The B-DNA patterns are subdivided in a B_I-family of rotamers (ϵ[*tr*], ζ[*g*–]) and a B_{II}-family of rotamers (ϵ[*g*–], ζ[*tr*]). The remaining 100 entries represent A-DNA conformations. It appeared that the data matrix also contained 5 so-called crankshaft patterns (α[*tr*], γ[*tr*])[2].

The patterns in this matrix were first randomized and then split into a training set and a test set. In a random manner 163 patterns were assigned to the training set and the remaining 81 patterns to the test set. Next, torsion angles δ_1, ϵ, β and γ were selected as predictor variables, while χ_1, ζ and α represent the variables to be predicted, see Figure 4.

Finally, because the ANN can only deal with data within a certain range, both the training set and the test set were scaled. We tried several scaling techniques but auto-scaling to the means and standard deviations of the training set worked best.

Configuration

PCR and PLS

Both in PCR and PLS a low-dimensional approximation of the data is made so that relevant information is retained while the noise is filtered out. The input matrix, **X**, contains 4 variables. Hence, in the PCR and PLS procedure a maximum of 4 latent variables can be achieved. Once the regression vector, **b**, for these procedures is obtained for the training set it is possible to predict the output patterns, $\hat{\mathbf{y}}$, for the test set (see Equation (2)).

To determine the optimal number of latent variables for the models a cross-validation procedure is performed. For this purpose, during the training session, a new training set of fewer patterns, from the original training set, is created. The remaining patterns form a test set. Recognition is performed on the new training set and prediction is tested with the resulting test sets. This procedure is performed 10 times. From the resulting errors the optimal number of variables is determined.

[2] *g*+: 0–120°; *tr*: 120–240°; *g*−: 240–360°.

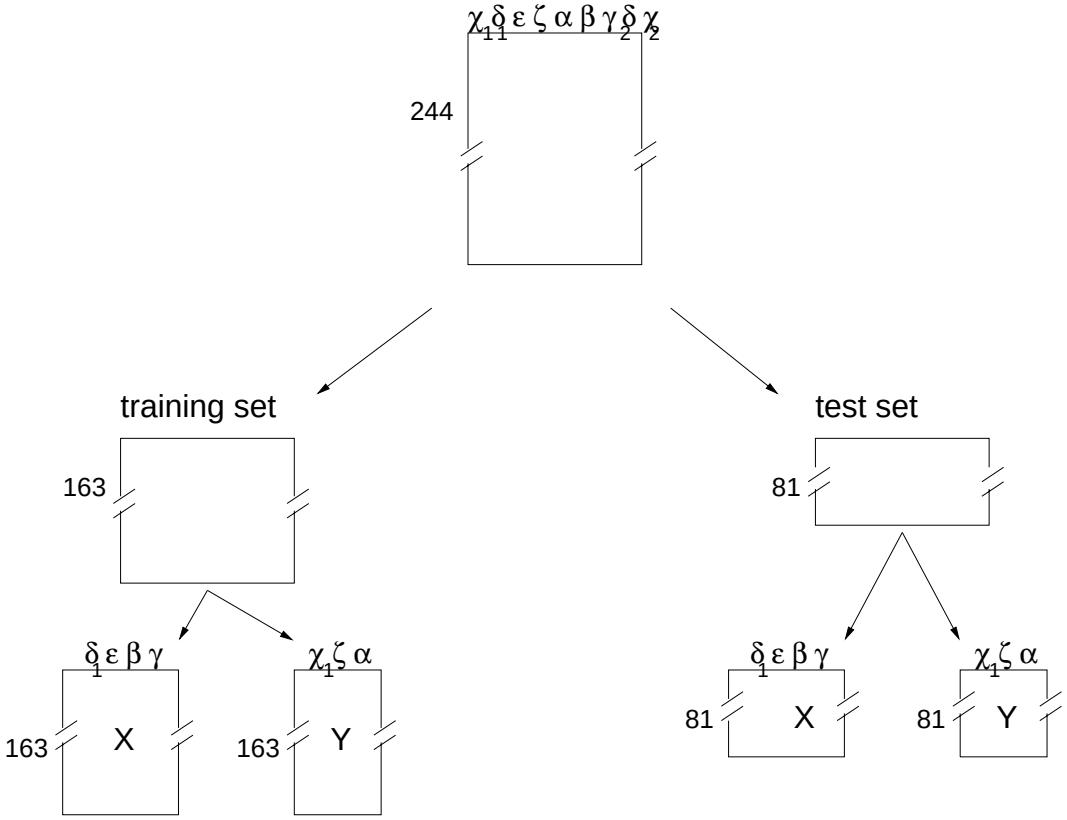

Figure 4. Creation of training sets and test sets from the original data matrix.

Error back-propagation network

Table 2 depicts the configurational settings of the error back-propagation network that is used in this study. Obviously, the number of input neurons corresponds to the number of predictor variables, which in this case is 4, and the number of output neurons corresponds to the number of variables to be predicted, which in this case is 3. The number of hidden units is an adjustable parameter.

The activity function corresponds to the net signal. A sigmoid transfer function is used. Once the network topology is defined the next step is to initialize the weights. In the present configuration weights are initialized between –0.05 and 0.05. The distribution of weights within this range depends on a random seed value.

Hence, for a particular run, the parameters that need to be set are the number of hidden units, the learning rate and the momentum.

For initial experiments 4, or 5, hidden units were chosen. This is based on the fact that the number of independent variables in the data matrix, i.e., the true dimensionality, was estimated to be 4 [6]. The choice of initial η and α values resulted from earlier studies. In trial and error experiments optimal values for these two parameters were found.

The next experiment entailed the choice for the optimal number of hidden units. A network with too small a number of hidden units will not be capable to deal with the complexity of the problem and result in

Table 2. Configurational settings for the error back-propagation network

#input neurons	4
#hidden neurons	4,5
#output neurons	3
#epochs (max.)	30 000
α	0.0002
η	0.0025
activity function	net_j
transfer function	sigmoid
AttF	20

a large error of prediction. Increasing the number of hidden units initially leads to a decrease in the error of prediction. However, if the number becomes too high the error may start to increase again. Moreover, the number of degrees of freedom of the network dramatically increases which is not desired generally. We ran networks with the number of hidden units increasing from 1 to 20.

The robustness of the network was tested in two manners. First we selected the optimal number of hidden units from the aforementioned experiment. For this configuration we ran the network 10 times. Each run was started with different initialized weights, i.e., each time a different random seed was used to create weight matrices. Then, we constructed 10 different training sets and test sets from the original data matrix. This procedure is depicted in Figure 4. The network with the optimal number of hidden units was trained with all of these sets.

Hardware and software

PCR and PLS procedures provided in the pls-toolbox of Matlab for Unix Workstations, version 4.2c (The MathWorks, Inc) were used. The multi-layered error back-propagation ANN was programmed in the Matlab environment within the Laboratory for Analytical Chemistry. The algorithm is based on the research described in Reference [20]. Both PCR, PLS and the ANN were run on a Sparc workstation.

Results

Table 2 depicts the initial configurational settings for the ANN. The table gives values for η and α that were found after a small optimization session. In successive experiments the number of hidden units was varied to determine the optimal number of hidden units. Figure 5 depicts that with increasing number of hidden units the predictive ability of the network initially improves and gradually stabilizes. The minimum error of prediction was found for 7 hidden units. In Figure 5 the RMSE for prediction (for the test set) that was normalized (to result in an overall error of prediction) is depicted. The normalized error is called the Normalized Standard Error (NSE) here.

The robustness of the network was tested by replicating the runs for 7 hidden units with different random seeds (not depicted). In another experiment the runs were replicated for different training and test sets. The construction of these sets was outlined in Figure 4.

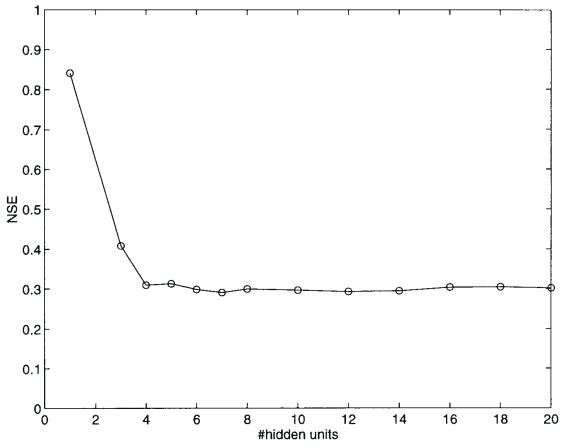

Figure 5. Normalized standard error (NSE) of prediction for the error back-propagation network with different numbers of neurons in the hidden layer.

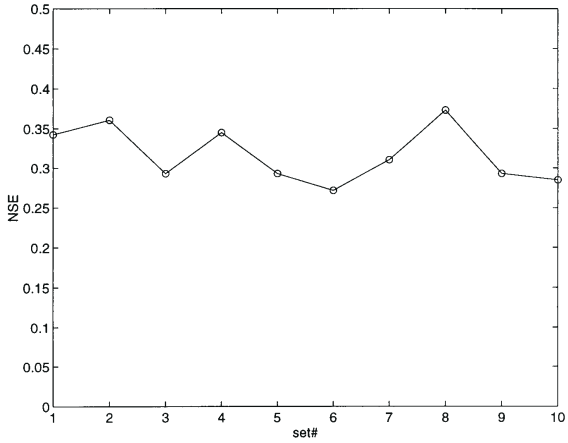

Figure 6. Normalized standard error (NSE) of prediction for the error back-propagation network with 7 neurons in the hidden layer for several training sets and test sets.

The results of the replicate experiments are depicted in Figure 6.

RMSE of prediction values were calculated for predicted torsion angles for the test set that produced the lowest NSE in Figure 6. Table 3 depicts the RMSE of prediction for predicted χ_1, ζ and α torsion angles of the test set by means of the back-propagation network. These predicted torsion angles of the test set are plotted versus their actual values in Figure 7.

Another manner to express the predictive ability of the model is to monitor the residuals between predicted and actual torsion angle values. In Table 4 the percentage of objects from the test set of which the residual value was less than a certain threshold is de-

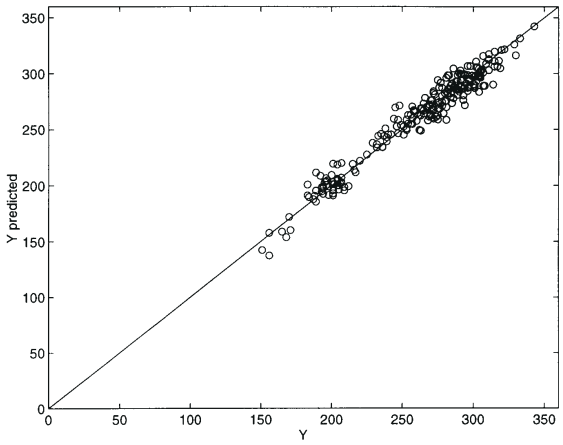

Figure 7. Predicted χ, ζ and α torsion angles of the test set plotted versus their actual values for the error back-propagation network model.

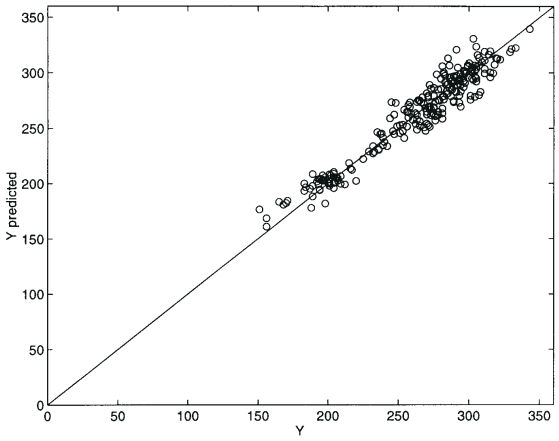

Figure 8. Predicted χ, ζ and α torsion angles of the test set plotted versus their actual values for the PCR model.

Table 3. RMSE of prediction for predicted torsion angles of the test set for the error back-propagation network (ANN), the PCR and PLS method

	ANN	PCR	PLS
χ_1	8.4	9.0	9.0
ζ	9.8	12.6	12.6
α	7.8	9.2	9.2

picted. E.g., with the back-propagation network, 79 of the 81 objects, corresponding to 98%, had a residual value smaller than 20° for χ_1. For the ANN some 70 to 80% of the objects have residuals within a range of 10°. This is reasonably good for conformational analysis purposes, hence, the model predicts well.

Table 4. Percentage of objects with residuals below thresholds of 30, 20, 10 and 5 deg, respectively, for predicted torsion angles of the test sets. Obviously, PCR and PLS results were comparable. Therefore, only PCR results are depicted. Largest residuals of the back-propagation model for χ_1, ζ and α were 23, 24 and 17, respectively. Largest residuals of the PCR and PLS model for χ_1, ζ and α were 29, 30 and 24, respectively

	χ_1		ζ		α	
	ANN	PCR	ANN	PCR	ANN	PCR
<30°	100	100	100	100	100	100
<20°	98	98	94	89	100	96
<10°	79	70	69	58	78	74
<5°	47	43	42	33	42	41

For both PCR and PLS the training set was split 10 times during a cross-validation procedure (as described earlier). From this procedure the optimal number of latent variables, which was 4 in all cases, resulted.

Then, to obtain the best model, regression was performed and the resulting regression coefficients were used to predict the χ, ζ and α torsion angles of the test set. The results are depicted in Table 3 and Figure 8. Since all 4 input variables were used in both PCR and PLS, these methods clearly represent a regular Multiple Linear Regression. When the training set was split more than 10 times in the PCR/PLS cross-validation procedure the results did not improve.

Discussion

A variety in NSEs is seen when different training sets and test sets, created form the data matrix, were used. This is depicted in Figure 6. This may be the result of classes, present in the data matrix, that are only represented by a small number of objects, e.g., there are only five crankshaft objects. In this case the training set may not be completely representative for the test set. To solve this problem a larger number of objects from such a class should be added to the data matrix. However, most sets resulted in acceptable predictive ability, expressed in terms of low RMSE of prediction values for predicted torsion angles and percentages of residuals within a certain range for predicted torsion angles (see hereafter).

The network is robust towards different initializations. This was observed for all sets when the weights

were initialized with different random seed values. The resulting variation in NSE for prediction values was very small. Hence, an extensive cross-validation procedure to test the robustness of the network was not performed.

It can be seen that the prediction of the ANN method is slightly better than the PCR and PLS method. Hence, the choice of the number of neurons in the hidden layer enables the user to create a model that is more capable of handling complexity in the data matrix.

Conclusions

We built models to predict the torsion angles χ, ζ and α from torsion angles δ, ϵ, β and γ for nucleic acid dinucleotides. For this purpose an error back-propagation ANN, PCR and PLS were used which is a new development in this area.

The data matrix that was used contains crystal structures. However, the predictive ability of the methods looks promising. Moreover, once a model is built, the prediction of torsion angles from an unknown object, or sets of objects, is very fast. Therefore, the procedure offers perspectives in predicting torsion angles for structures in solution. Especially multidimensional correlation spectroscopy (or the combination of multidimensional correlation and NOE spectroscopy) may lead to reasonable estimates of δ, ϵ, β and γ torsion angle values. With properly built and validated models good values for other torsion angles are predicted from these estimates. These predicted torsion angles might form the basis for trial structures in conformational analysis. These structures can be refined further with other methods.

Acknowledgements

The anonymous referees are acknowledged for suggestions concerning the manuscript. Dr. E.P.P.A. Derks is acknowledged for the development of the error back-propagation network and fruitful discussions.

References

1. Gorin, A.A., Zhurkin, V.B. and Olson, W.K., J. Mol. Biol., 247 (1995) 33.
2. El Hassan, M.A. and Calladine, C.R., J. Mol. Biol., 259 (1996) 95.
3. Fratini, A.V., Kopka, M.L., Drew, H.R. and Dickerson, R.E., J. Biol. Chem., 247 (1982) 14686.
4. Conner, B.N., Yoon, C., Dickerson, J.L. and Dickerson, R.E., J. Mol. Biol., 174 (1984) 663.
5. Pearlman, D.A. and Kim, S.-H., J. Biomol. Struct. Dyn., 4 (1986) 49.
6. Beckers, M.L.M. and Buydens, L.M.C., J. Comput. Chem., (1998) in press.
7. Piantini, U., Sørensen, O.W. and Ernst, R.R., J. Am. Chem. Soc., 104 (1982) 6800.
8. Shaka, A.J. and Freeman, R., J. Magn. Reson., 51 (1983) 169.
9. Widmer, H. and Wüthrich, K., J. Magn. Reson., 74 (1987) 316.
10. Haasnoot, C.A.G., de Leeuw, F.A.A.M. and Altona, C., Tetrahedron, 36 (1980) 2783.
11. Lankhorst, P.P., Haasnoot, C.A.G., Erkelens, C. and Altona, C., J. Biomol. Struct. Dyn., 1 (1984) 1387.
12. Mooren, M.M.W., On nucleic acid structure analysis by NMR, Ph. D. Thesis, University of Nijmegen, Nijmegen, the Netherlands, 1993.
13. Wold, H., Multivariate Analysis, Academic Press, New York, NY, U.S.A., 1966.
14. Wold, S., In Kowalski, B. (Ed.), Chemometrics: Mathematics and Statistics in Chemistry, Reidel, Dordrecht, the Netherlands, 1984.
15. Jolliffe, I.T., Principal Component Analysis, Springer-Verlag, New York, NY, U.S.A., 1986.
16. Geladi, P. and Kowalski, B., Anal. Chim. Acta, 185 (1986) 1.
17. Smits, J.R.M., Melssen, W.J., Buydens, L.M.C. and Kateman, G., Chemom. Intell. Lab. Systems, 22 (1994) 165.
18. Zupan, J. and Gasteiger, J., Neural Networks for Chemists: An Introduction, VCH Verlagsgesellschaft, Weinheim, Germany, 1993.
19. Rumelhart, D.E., McCleland, J.L. and the PDP research group, Parallel distributed processing. Explorations in the microstructure of cognition. Volume 1: Foundations; Volume 2: Psycological and Biological Methods. MIT Press, London, UK, 1986.
20. Anguita, D., Parodi, G. and Zunino, R., In Proceedings of the World Congress on Neural Networking, Portland, Oregon, Volume 1, Lawrence Erlbaum/INNS Press, 1993, p. 165.