# Prediction of Enantiomeric Excess in a Combinatorial Library of Catalytic Enantioselective Reactions

João Aires-de-Sousa*,† and Johann Gasteiger‡

*Departamento de Química, CQFB and REQUIMTE, Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa, 2829-516 Monte de Caparica, Portugal, and Computer-Chemie-Centrum, Institute of Organic Chemistry, University of Erlangen-Nürnberg, Nägelsbachstrasse 25, D-91052 Erlangen, Germany*

*Received February 7, 2004*

A quantitative structure−enantioselectivity relationship was established for a combinatorial library of enantioselective reactions performed by addition of diethyl zinc to benzaldehyde. Chiral catalysts and additives were encoded by their chirality codes and presented as input to neural networks. The networks were trained to predict the enantiomeric excess. With independent test sets, predictions of enantiomeric excess could be made with an average error as low as 6% ee. Multilinear regression, perceptrons, and support vector machines were also evaluated as modeling tools. The method is of interest for the computer-aided design of combinatorial libraries involving chiral compounds or enantioselective reactions. This is the first example of a quantitative structure−property relationship based on chirality codes.

## Introduction

High-throughput screening (HTS) of chiral catalysts for enantioselective reactions is a powerful promising approach to the discovery of new catalysts.[1,2] Combinatorial libraries are particularly useful for the optimization of catalytic systems. The number of experiments that can be screened using reasonable resources is, however, usually limited. Therefore, knowledge extraction from previous experiments and its use for the design of additional experiments is crucial. We propose to train artificial neural networks (ANN) with data from combinatorial libraries to make predictions of enantiomeric excess given the molecular structures. The knowledge acquired by the networks can then be applied to new situations.[3]

Artificial neural networks[4] are computational tools that learn from training examples and have the capability to apply these models to new situations to make predictions. We have developed chirality codes to represent the chirality of molecular structures by real numbers.[5−7] They are fixed-length codes which can be used, for example, as input to neural networks. These codes were shown to qualitatively correlate with enantiomeric selectivity in enantioselective reactions and in chiral chromatography. Instead of measuring chirality by a single value, the chirality code is a molecular transform that *represents* the chirality of a molecule using a spectrum-like, fixed-length code and includes information about the geometry of chiral centers, properties of the atoms in their neighborhoods, and bond lengths. The code distinguishes between enantiomers and yields descriptors with symmetrical values for opposite enantiomers.

In this paper, chirality codes represent catalysts and additives and are used to train feed-forward neural networks to predict the enantiomeric excess (ee). Our study was based on the experimental results obtained by Long and Ding[8] for the enantioselective addition of diethyl zinc to benzaldehyde in the presence of a racemic catalyst (RC) and an enantiopure chiral additive (CA) (Scheme 1). The results obtained by neural networks were compared to those from perceptrons, MLR (multilinear regression), and support vector machines.

* Corresponding author. Fax: +351 21 2948550. E-mail: jas@fct.unl.pt.
† Universidade Nova de Lisboa.
‡ University of Erlangen-Nürnberg.

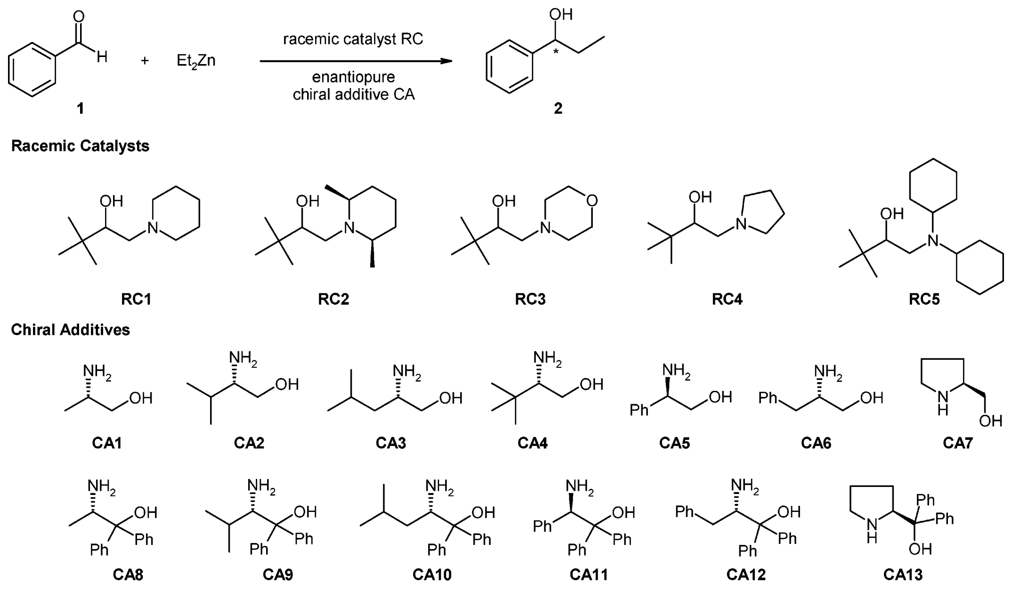
## Methodology

**Representation of Molecular Structures by Chirality Codes.**[5] The chirality code is calculated by considering combinations of four atoms (i, j, k, and l), each atom belonging to a different ligand of the chiral center. A combination is characterized by a chirality signal, $s_{ijkl}$, and by a real value, $e_{ijkl}$, incorporating information related to atomic physicochemical properties and interatomic distances. The parameter $e_{ijkl}$ is defined by eq 1, which considers the four atoms of the combination, i, j, k, and l, each of them belonging to a different ligand (A, B, C, and D, respectively) of a chiral center.

$$e_{ijkl} = \frac{a_i a_j}{r_{ij}} + \frac{a_i a_k}{r_{ik}} + \frac{a_i a_l}{r_{il}} + \frac{a_j a_k}{r_{jk}} + \frac{a_j a_l}{r_{jl}} + \frac{a_k a_l}{r_{kl}} \qquad (1)$$

$a_i$ is a property of atom i, such as atomic charge, and $r_{ij}$ is the distance between atoms i and j. To consider the 3D structure but make the chirality code independent of a specific conformer, $r_{ij}$ was taken as the sum of the bond lengths between atoms i and j on the path with minimum number of bond counts. The chirality signal, $s_{ijkl}$, can attain values of +1 or −1, depending on the ranking of atomic properties $a_i$, $a_j$, $a_k$, $a_l$ and the configurational arrangement of the four ligands (the chirality signals of two corresponding combinations in opposite enantiomers have symmetrical values). All these combinations are incorporated by a distribution function (eq 2), where $u$ is the running variable, $n_X$ is the number of atoms in ligand X, and $b$ is a smoothing

**Scheme 1.** Generation of a Library of Enantioselective Reactions, by Combination of Five Racemic Catalysts (RC1−RC5) and 13 Chiral Additives (CA1−CA13)



factor. In practice, *b* controls the width of the peaks obtained by a graphical representation of $f_{\text{CICC}}(u)$ vs *u*.

$$f_{\text{CICC}}(u) = \sum_{i}^{n_A} \sum_{j}^{n_B} \sum_{k}^{n_C} \sum_{l}^{n_D} s_{\text{ijkl}} \exp[-b(u - e_{\text{ijkl}})^2] \qquad (2)$$

The chirality code is obtained by sampling the distribution function at predefined points. Figure 1 shows the representation of the chirality code for two chiral molecules. The actual range of *u* used in an application is chosen according to the range of atomic properties related to the range of observed interatomic distances for the given molecules. The number of discrete points of $f_{\text{CICC}}(u)$ determines the resolution of the chirality code. For the experiments described here, the chirality codes were calculated by sampling $f_{\text{CICC}}(u)$ at 101 evenly spaced values of *u*, with *u* ranging between −0.02 e$^2$ Å$^{-1}$ and +0.02 e$^2$ Å$^{-1}$ (for the racemic catalysts) or −0.04 e$^2$ Å$^{-1}$ and +0.04 e$^2$ Å$^{-1}$ (for the chiral additives). The ranges of *u* were chosen to cover all the nonzero values of $f_{\text{CICC}}(u)$ in the specific sets of molecules. Atomic charge was used as the atomic property, and hydrogen atoms not bonded to the chiral center were neglected. The charge values were calculated using the PEOE procedure[9] implemented in PETRA, and the molecular geometries were constructed by CORINA.[10] Both software packages can be tested on the website http://www2.chemie.uni-erlangen.de and are available from Molecular Networks, GmbH, http://www.mol-net.de.

**Data Sets.** The experimental results obtained by Long and Ding[8] for the enantioselective addition of diethyl zinc to benzaldehyde in the presence of a racemic catalyst (RC) and an enantiopure chiral additive (CA) were used for this investigation (Scheme 1). All the combinations of 5 racemic catalysts RC and 13 enantiopure additives CA were tested, giving rise to a combinatorial library of 65 reactions. Each of the 65 entries of the library was represented by chirality

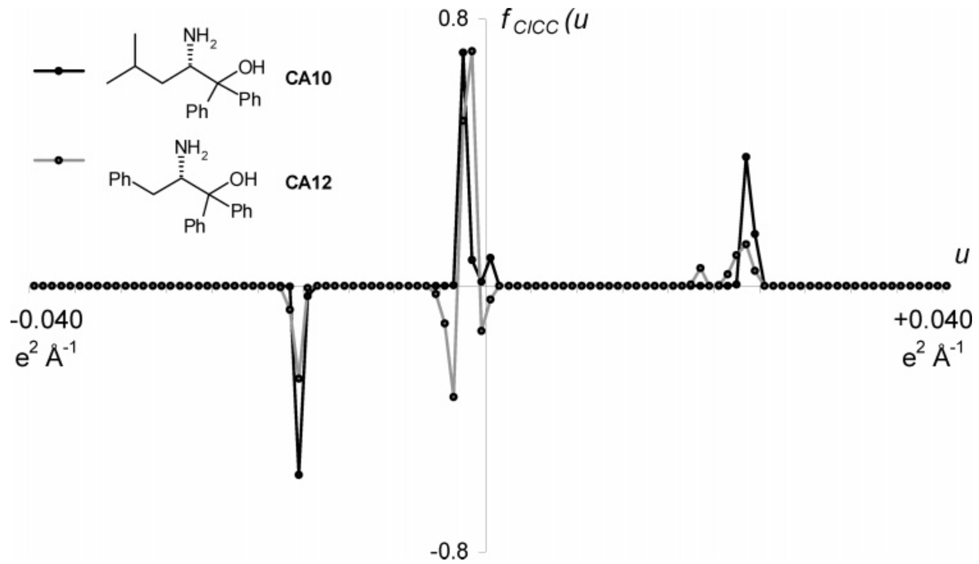


**Figure 1.** Representation of $f_{\text{CICC}}(u)$ vs *u* for CA10 and CA12 sampled at 101 evenly separated values between −0.040 e$^2$ Å$^{-1}$ and +0.040 e$^2$ Å$^{-1}$. Partial atomic charge was used as the atomic property, and hydrogen atoms not bonded to the chiral center were neglected.

code-based descriptors of the corresponding racemic catalyst and chiral additive. While chiral additives CA were represented by their chirality codes (101 values), racemic catalysts RC were represented by *absolute values* of their chirality codes (101 values), because they were used as *racemic* mixtures.

Components of chirality codes that always exhibited negligible values over the entire data set were excluded. In addition, some components were excluded by removal of correlations: the descriptors were sorted in decreasing order of the Pearson correlation coefficients when used in a global search for two-descriptor correlations, and removed stepwise to avoid correlations above 0.9. Twenty-eight components remained. The 65 objects were randomly split into three data sets of the same size (sets A, B, and C), each data set covering the whole range of observed ee values (−71% to +16%).

A *conformation-independent* chirality code (CICC)[5] was employed here due to the high conformational flexibility of the involved chiral molecules, which results in an ensemble of conformations.

**Neural Networks.**[4] Feed-forward ANNs were trained with the back-propagation of errors algorithm.[11] The network architecture consisted of an input layer with a number of input neurons corresponding to the number of selected components of the chirality codes and a bias (with value 1), a hidden layer with one or two hidden neurons and a bias (with value 1), and one output neuron (for the ee). A network was trained with data sets A and B to give the ee values as the output. Eight random reactions from the combined A + B set were left out and were used as a cross-validation set. It was assured that the cross-validation set covers the entire range of output. The training set was iteratively presented to the network until a minimum error was obtained for the cross-validation set (typically 1000−2000 epochs). Predictions were then obtained for set C and compared with experimental values. ANNs were trained and applied with in-house developed software based on JATOON Java applets.[12]

**Perceptrons.** A perceptron is essentially a feed-forward ANN with no hidden layer. The input neurons are directly connected to the output neurons. Perceptrons were implemented with in-house developed software, as for the ANNs, and optimization of the weights was also performed by the delta rule.[4]

**Multilinear Regression (MLR).** The multilinear regression analysis was performed with the glm function of the R program version 1.9.0.[13]

**Support Vector Machines.** The SVM models were generated with the LIBSVM[14] implementation of the $\nu$-SVM algorithm. Radial basis kernels were used because these are expected to yield superior performance in most practical applications. A grid search was performed with the supplied gridregression.py script to find optimum values for parameters C, gamma, and nu. A model was developed with data sets A + B and then tested with data set C, another model was developed with sets A + C to get predictions for set B, and another model was developed with sets B + C to obtain predictions for set A.

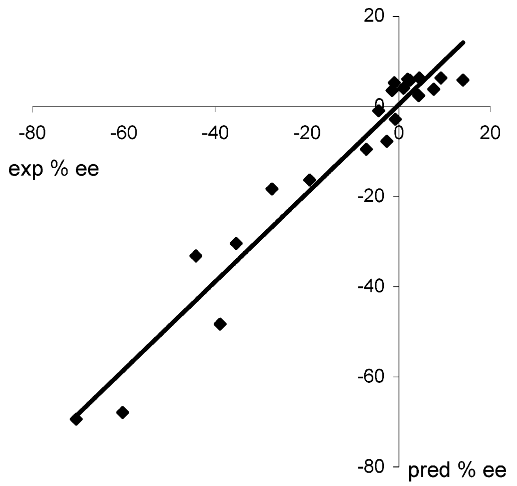


**Figure 2.** Experimental ee vs neural network prediction for the reactions of the test set C ($R^2 = 0.946$). The network consisted of 11 input neurons, 1 hidden neuron, and 1 output neuron.

## Results and Discussion

A neural network was trained with data sets A and B, using 28 input neurons. After training, the model was tested with data set C that was not used for training, and a root-mean-square of errors (RMSE) of 5.5% ee was obtained. The architecture of this particular network, with only one hidden neuron, allows for easy analysis of the relative impact of the descriptors on the output. The difference between the maximum and the minimum values of a descriptor in the training set was multiplied by the corresponding weight of the trained network. These products varied between 2.97 and 0.03 for the 28 descriptors and were used to assess the relevance of the descriptors in the model. By excluding the descriptors whose product was <1, 11 descriptors remained and were used to train a new neural network (with 11 input neurons and 1 hidden neuron). This neural network has $12 \times 1 + 2 \times 1 = 14$ weights. After training with sets A and B (44 reactions), the test for set C yielded again RMSE = 5.5% ee, which shows how the initial network had implicitly performed a selection of variables by assigning very small values to some weights. In Figure 2, the individual predictions are plotted against the experimental values.

This feed-forward neural network has strong similarities to a perceptron and even to a multilinear regression; however, the FFNN and the perceptron are not linear models. In these models, even if the net input of the hidden neuron is a linear combination of the inputs, the net input is transformed by a *nonlinear* (sigmoid) transfer function. Experiments were performed with a perceptron and with MLR, and compared with neural networks (Table 1). Additionally, the experiments were repeated with data set B as the test set (data sets A and C were used for training) and with data set A as the test set (data sets B and C were used for training). The results for the 3-fold cross-validation procedure are also included in Table 1. The quality of the predictions slowly improved from a purely linear model (MLR) to a perceptron to a FFNN with one hidden neuron. Inclusion of a second hidden neuron increased the number of weights to 27, which is still reasonable, further improving the predictions for the test sets and showing that the ramified way of data processing in FFNNs is an advantage in this application.

**Table 1.** Comparison of the Prediction Power of FFNNs, Perceptrons, MLR, and SVM

| model | | A | B | C | combined (3-fold cv) |
|---|---|---|---|---|---|
| MLR (11 variables) | $R^2$ | 0.739 | 0.804 | 0.818 | 0.776 |
| | RMSE | 13 | 11.7 | 10.2 | 11.7 |
| perceptron (11 × 1) | $R^2$ | 0.768 | 0.868 | 0.918 | 0.845 |
| | RMSE | 12.5 | 9.1 | 7.2 | 9.9 |
| FFNN (11 × 1 × 1) | $R^2$ | 0.873 | 0.91 | 0.946 | 0.906 |
| | RMSE | 9.1 | 7.6 | 5.5 | 7.6 |
| FFNN (11 × 2 × 1) | $R^2$ | 0.891 | 0.93 | 0.961 | 0.923 |
| | RMSE | 8.7 | 7.0 | 4.8 | 6.9 |
| SVM (11 variables) | $R^2$ | 0.713 | 0.762 | 0.838 | 0.748 |
| | RMSE | 18.0 | 10.9 | 11.3 | 14.2 |

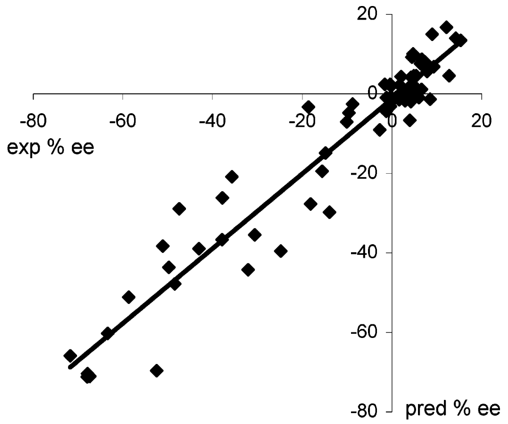


**Figure 3.** Experimental ee vs neural networks predictions for all the reactions in the library ($R^2$ = 0.923). Network architecture consisted of 11 input neurons, two hidden neurons, and one output neuron. Predictions for reactions belonging to a given subset were obtained by a network trained with other subsets.

Finally, experiments were carried out with the support vector machines (SVM) methodology for regression using the same 11 variables as the input and the same three data sets, A, B, and C. Support vector machines are learning systems that use a hypothesis space of linear functions in a high dimensional feature space, trained with a learning algorithm from optimization theory that implements a learning bias derived from statistical learning theory.[15] Unlike ANN, there is only one minimum in the optimization problem, resulting in the rapid generation of a unique solution. With this particular data set, SVM have not performed so well as FFNNs.

Neural networks performed consistently better than the other methods. The predictions for all the reactions (all test sets) obtained by the FFNNs with two hidden neurons are combined in Figure 3. Excellent results were obtained (average absolute error of prediction: 5.2% ee, RMSE 6.9% ee). The predictions were analyzed to evaluate the ability of the networks to detect "promising reactions", and the following results were found. In the entire library, there are seven reactions with an enantioselectivity higher than 50% ee. The networks correctly predicted an ee ≥50% for eight reactions: seven reactions with experimental ee higher than 50% and another reaction with experimental ee 38.3%. This means that if the networks were used to virtually screen the 65 reactions and to make suggestions of promising reactions, they would not miss a single reaction and would suggest one additional reaction, which exhibits, in fact, relatively high ee.

Beyond quantitatively predicting the enantioselectivity, the networks also correctly predicted which of the two enantiomers was preferred for 53 out of 65 reactions. The 12 wrong predictions of enantiomeric preference corresponded to reactions with experimental ee lower than 7%.

## Conclusion

A quantitative structure−enantioselectivity relationship was established between catalysts (and additives) and enantiomeric excess in enantioselective reactions. The model was based on chirality codes. Different techniques were used for modeling—feed-forward neural networks, perceptrons, multilinear regressions, and support vector machines—with FFNNs yielding the best predictions (standard error of 6.9% ee). The results here described demonstrate that chirality codes and artificial neural networks can be used together in applications involving chirality to learn from combinatorial data. The method has obvious applications in computer-assisted design of combinatorial libraries.

**Acknowledgment.** J.A.S thanks Fundação para a Ciência e a Tecnologia (MCES, Lisboa, Portugal) for financial support.

## References and Notes

(1) For reviews, see: (a) Reetz, M. T. *Angew. Chem.* **2001**, *113*, 292−320; *Angew. Chem., Int. Ed.* **2001**, *40*, 284−310. (b) Gennari, C.; Piarulli, U. *Chem. Rev.* **2003**, *103*, 3071−3100.
(2) For recent developments, see: (a) Reetz, M. T. *Angew. Chem.* **2002**, *114*, 1391−1394; *Angew. Chem., Int. Ed.* **2002**, *41*, 1335−1338. (b) Taran, F.; Gauchet, C.; Mohar, B.; Meunier, S.; Valleix, A.; Renard, P. Y.; Créminon, C.; Grassi, J.; Wagner, A.; Mioskowski, C. *Angew. Chem.* **2002**, *114*, 132−135; *Angew. Chem., Int. Ed.* **2002**, *41*, 124−127.
(3) For an example of structure-based predictions of biological activity by a neural network using data from a combinatorial library, see: Schneider, G.; Nettekoven, M. *J. Comb. Chem.* **2003**, *5*, 233−237.
(4) Zupan, J.; Gasteiger, J. *Neural Networks in Chemistry and Drug Design*, 2nd ed.; Wiley-VCH: Weinheim, 1999.
(5) Aires-de-Sousa, J.; Gasteiger, J. *J. Chem. Inf. Comput. Sci.* **2001**, *41*, 369−375.
(6) Aires-de-Sousa, J.; Gasteiger, J. *J. Mol. Graphics Model.* **2002**, *20* (5), 373−388.
(7) Aires-de-Sousa, J.; Gasteiger, J.; Gutman, I.; Vidović, D. *J. Chem. Inf. Comput. Sci.* **2004**, *44*, 831−836.
(8) Long, J.; Ding, K. *Angew. Chem.* **2001**, *113*, 562−565; *Angew. Chem., Int. Ed.* **2001**, *40*, 544−547.
(9) Gasteiger, J.; Marsili, M. *Tetrahedron* **1980**, *36*, 3219−3228.
(10) Sadowski, J.; Gasteiger, J. *Chem. Rev.* **1993**, *93*, 2567−2581.
(11) Rumelhart, D.; Durbin, R.; Golden, R.; Chauvin, Y. Back-propagation: The Basic Theory. In *Mathematical Perspectives on Neural Networks*; Smolensky, P., Mozer, M. C., Rumelhart, D. E., Eds.; Lawrence Earlbaum Assoc.: Hillsdale, NJ, 1996; pp 533−566.
(12) Aires-de-Sousa, J. *Chemom. Intell. Lab. Syst.* **2002**, *61*, 167−173.
(13) R program is freely available from http://www.r-project.org.
(14) Chang, C.-C.; Lin, C.-J. LIBSVM: A library for support vector machines. 2003, Software available at http://www.csie.ntu.edu.tw/∼cjlin/libsvm.
(15) Cristianini, N.; Shawe-Taylor, J. *An Introduction to Support Vector Machines*; Cambridge University Press: Cambridge, U.K., 2000.

CC049961Q