Automated Descriptor Selection for Quantitative Structure–Activity Relationships Using Generalized Simulated Annealing

Jon M. Sutter, Steve L. Dixon, and Peter C. Jurs*

Department of Chemistry, 152 Davey Laboratory, Penn State University, University Park, Pennsylvania 16802

Received June 16, 1994[⊗]

The central steps in developing QSARs are generation and selection of molecular structure descriptors and development of the model. Recently, computational neural networks have been employed as nonlinear models for QSARs. Neural networks can be trained efficiently with a quasi-Newton method, but the results are dependent on the descriptors used and the initial parameters of the network. Thus, two potential opportunities for optimization arise. The first optimization problem is the selection of the descriptors for use by the neural network. In this study, generalized simulated annealing (GSA) is employed to select an optimal set of descriptors. The cost function used to evaluate the effectiveness of the descriptors is based on the performance of the neural network. The second optimization problem is selecting the starting weights and biases for the network. GSA is also used for this optimization. The result is an automated descriptor selection algorithm that is an optimization inside of an optimization. Application of the method to a QSAR problem shows that effective descriptor subsets are found, and they support models that are as good or better than those obtained using traditional linear regression methods.

INTRODUCTION

With growing environmental concern, a need for predicting the toxicity of compounds has emerged. Experimental assessment of toxicity can be costly, time consuming, and hazardous. Quantitative structure–activity relationships (QSARs) can be used to predict toxicity accurately without using experimental methods, provided the compounds are close structural analogs of compounds with experimentally known toxicities. The main purpose of a QSAR is to relate a given biological activity to the structural features of a particular molecule. Over the years many different methods have evolved for developing QSARs. Two of the most common are the Hansch method[1] and the Free–Wilson method.[2] Both approaches use mathematical functions to link biological activity and structure. They differ only in the descriptors used to encode structural characteristics. The Hansch method uses physicochemical constants, and the Free–Wilson method uses the frequency and position of chemical substituents (group contribution). A multiple linear regression (MLR) equation is then generated to estimate the biological activity of interest from the descriptor values.

In the past, QSAR investigators were interested in obtaining descriptors with ease and simplicity.[3] With increasing computer power the concern of descriptor generation has turned from simple acquisition and toward enhancing the ability to effectively describe the activity. Therefore it is beneficial to create a relatively large descriptor pool that maximizes the amount of information. After the descriptors are generated, a mathematical model that relates the numerical values of a subset of descriptors to the biological activity can be created. These mathematical models are typically generated using either MLR or computational neural networks (CNN),[4–6] the latter of which may be classified as nonlinear models. The MLR models are built using a training set and validated using an external prediction set. The CNN models are also built using a training set. However, the aid of a cross-validation set is needed to prevent overtraining. The CNN models are also validated using an external prediction set.

If the response is monotonic, and the departures from nonlinearity are not significant, then a linear model can be expected to adequately identify important parameters for a CNN. Consequently, descriptor subsets are frequently selected based on the computationally less expensive MLR model and then submitted to a CNN. However, since a linear function usually cannot accurately describe a nonlinear function without using extra nonlinear terms, it is possible that descriptor subsets that yield adequate CNN models will not yield adequate MLR models. It follows that if a descriptor subset is selected based on CNN results rather than MLR results, a superior model may result.[7] An optimization technique is therefore needed to generate a descriptor subset for CNN models since investigating all possible combinations is not feasible. In this study, generalized simulated annealing (GSA) is used in combination with CNNs to develop an automated descriptor selection routine.

THEORY

Multiple Linear Regression. A linear mathematical function that relates descriptor values to the toxicity may be created using multiple linear regression. For n observations and p indendent variables the general linear regression model is represented as

$$Y_i = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_p x_p \qquad (1)$$

where $\beta_0, \beta_1, ..., \beta_p$ are regression coefficients, $x_1, x_2, ..., x_p$ are the independent variables (numeric descriptors), and Y_i is the dependent variable of interest (biological activity). The regression equation can be represented in matrix notation as

$$\mathbf{Y} = \mathbf{X}\beta \qquad (2)$$

where **Y** is an $n \times 1$ vector of responses, **X** is an $n \times (p +$

[⊗] Abstract published in *Advance ACS Abstracts,* November 15, 1994.

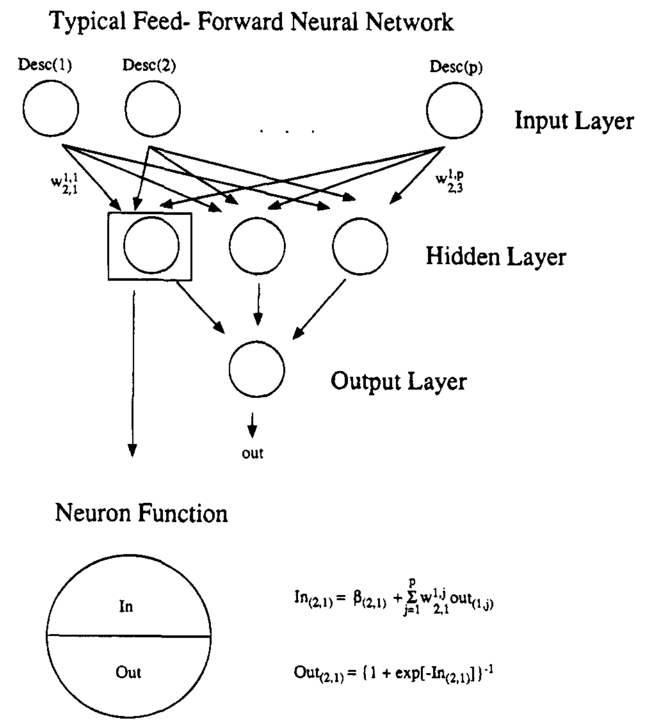
Typical Feed- Forward Neural Network
Desc(1)
Desc(2)
Desc(p)
Input Layer
$w^{1,1}_{2,1}$
$w^{1,p}_{2,3}$
Hidden Layer
Output Layer
out
Neuron Function
In
Out
$In_{(2,1)} = \beta_{(2,1)} + \sum_{j=1}^{p} w^{1,j}_{2,1} out_{(1,j)}$
$Out_{(2,1)} = \{1 + \exp[-In_{(2,1)}]\}^{-1}$

Figure 1. The neural network architecture used in this study.

1) matrix of independent variables, and β is a $(p + 1) \times 1$ vector of regression coefficients. The regression coefficients can be determined by the least-squares solution represented as

$$\beta = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y} \tag{3}$$

Once β is calculated, eq 2 can be used to estimate the dependent variable or activity for other compounds. Multiple linear regression techniques have proven to be very successful in QSAR studies. Recently, however, computational neural networks have been shown to improve QSAR models.[8-11]

Computational Neural Networks. Neural networks were originally designed as a model for the activity of the human brain. However, a CNN model can be thought of simply as a nonlinear regression model when applied to QSAR studies. The CNN is a mathematical function that can build models by a nonlinear least-squares method similar to MLR. However, since the model is nonlinear, the regression coefficients cannot be found in one step, and an iterative process must be used to determine the coefficients.

The neural network architecture that was used in this study is shown in Figure 1. The schematic depicts a fully-connected, feed-forward, three-layer neural network.[10] The CNN begins by performing a linear transformation on the input layer (descriptor values) from its original range to the interval [0,1]. The transformed values are then passed to the hidden layer. The input value of a hidden layer neuron is the summation of the products of the weights and the corresponding outputs of the previous input layer plus a bias term (β). An example of this calculation is shown in Figure 1. The first neuron in the hidden layer has an input of $In_{(2,1)}$. The output of the neuron, a sigmoidal transformation of the input, is also shown in Figure 1. The output layer will give a value between 0 and 1; therefore, an expansive linear transformation must usually be performed to obtain an estimate of the biological activity which can lie outside the interval [0,1].

The weights and biases are adjusted iteratively to minimize the sum-squared error for prediction of the target values (biological activity). The CNN method is much more computationally intensive than linear regression since the nonlinear regression coefficients (weights and biases) must be changed iteratively, which requires repeated evaluation of the network outputs. However, the greater mathematical flexibility found in CNNs often leads to models that are superior to MLR models, so the additional time required is often worth the effort.

Adjusting the weights and biases to fit target values is known as training the network. Because of the increased mathematical flexibility of CNNs and the large number of adjustable parameters, it is possible to obtain apparently good fits by chance or to overtrain the neural network. Recently, some guidelines for avoiding chance correlations and overtraining have been presented.[12] Chance correlations can be reduced by keeping the ratio of cases to connections larger than 3, and overtraining can be avoided by employing a cross-validation set.

If overtraining occurs, specific idiosyncrasies of the training set members will be assigned as significant contributors, and this can adversely affect the predictive ability of the network. To avoid this situation the data set is split into a training set and a cross-validation set. The weights and biases are adjusted based on the rms error of the training set members and the rms error of the cross-validation set is calculated periodically throughout the training. Overtraining is believed to occur when the rms error of the cross-validation set begins to rise. If training is stopped when the cross-validation error is at a minimum, then the network may be used with reasonable confidence for future predictions.

Training the network is nothing more than an optimization problem. When a large data set is used, or when the network has a large number of weights and biases, this task can become quite expensive computationally. It makes sense, then, to utilize an efficient optimization method for training. For these reasons, networks are trained in our work using the quasi-Newton[13] BFGS (Broyden–Fletcher–Goldfarb–Shanno)[14-18] algorithm, as opposed to the more widely reported steepest decent, back propagation (BP) algorithm.[19]

As with any quasi-Newton technique, the BFGS algorithm minimizes a real-valued function of n variables by carrying out a series of line minimizations in directions determined by information derived from the gradient of the function. The gradient is simply the vector of function derivatives taken with respect to each of the n parameters. In the case of neural networks, the function to minimize is the sum-squared error E_{net}, and the parameters are the network weights and biases, which are represented by the variables $x_1, x_2, ..., x_n$. The gradient vector **g** can be written as

$$\mathbf{g} = \begin{bmatrix} \partial E_{net}/\partial x_1 \\ \partial E_{net}/\partial x_2 \\ \vdots \\ \partial E_{net}/\partial x_n \end{bmatrix} \tag{4}$$

The entries in **g** are obtained by explicit differentiation of the error function with respect to the corresponding weights or biases.

Using the gradient vector, the BFGS optimization provides iterative estimates $\mathbf{G}_k$ of the inverse Hessian matrix for the error function. The Hessian is simply the matrix of second derivatives, the (i,j) entry of which is given by $\partial^2 E_{net}/\partial x_j \partial x_i$. This series of matrices and gradients is used to generate search directions $\mathbf{d}_k$ that are approximately mutually conjugate[18] with respect to the Hessian matrix. At cycle k, the BFGS steps are

1. Choose search direction $\mathbf{d}_k = -\mathbf{G}_k\mathbf{g}_k$
2. Determine the scalar α_k to minimize $E_{net}(\mathbf{x}_k + \alpha\mathbf{d}_k)$
3. Let $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k\mathbf{d}_k$
4. Compute the gradient $\mathbf{g}_{k+1}$ corresponding to the parameters $\mathbf{x}_{k+1}$
5. Update $\mathbf{G}$

$$\mathbf{G}_{k+1} = \mathbf{G}_k - \frac{\mathbf{G}_k\Delta\mathbf{g}_k\Delta\mathbf{x}_k^{T} + \Delta\mathbf{x}_k\Delta\mathbf{g}_k^{T}\mathbf{G}_k + \left[1 + \frac{\Delta\mathbf{g}_k^{T}\mathbf{G}_k\Delta\mathbf{g}_k}{\Delta\mathbf{x}_k^{T}\Delta\mathbf{g}_k}\right]\Delta\mathbf{x}_k\Delta\mathbf{x}_k^{T}}{\Delta\mathbf{x}_k^{T}\Delta\mathbf{g}_k}$$

where $\Delta\mathbf{x}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$ and $\Delta\mathbf{g}_k = \mathbf{g}_{k+1} - \mathbf{g}_k$

The line minimization parameter α_k can be estimated using a parabolic fit of the error along $\mathbf{d}_k$

$$E_{net}(\mathbf{x}_k + \alpha\mathbf{d}_k) = E_{net}(\mathbf{x}_k) + a\alpha + b\alpha^2 \qquad (5)$$

Two pieces of data are needed to find the constants a and b. Satisfactory results are obtained by using (1) the slope $\partial E_{net}/\partial\alpha$ at $\mathbf{x}_k$, which is given by $\mathbf{d}_k^{T}\mathbf{g}_k$, and (2) the error $E_{net}(\mathbf{x}_k + s\mathbf{d}_k)$, where s is an appropriately chosen step size. The minimum of this curve occurs when $\alpha = -a/(2b)$.

The initial estimate of $\mathbf{G}$ can be any positive-definite symmetric matrix, but an intelligent choice of the entries can significantly speed up the optimization. In the present application, a diagonal estimate of $\mathbf{G}$ is obtained from the gradient using two sets of weights and biases. After computing $\mathbf{g}$ for one set of network parameters, each weight or bias x_i is changed by a small amount Δx_i and the gradient is computed a second time. The corresponding change in $\mathbf{g}$ may be used to estimate the diagonal entries of $\mathbf{G}$

$$\mathbf{G}_{ii} \approx \Delta x_i/\Delta g_i \qquad (6)$$

The value is restricted to be positive and is bounded to [0.005, 0.1/GMIN] where GMIN is MAX(ABS-(GRAD(I),10^{-20}).

Use of a BFGS optimization technique significantly speeds up training of neural networks compared to the widely-used BP method. Training times are often reduced by an order of magnitude or more because the BFGS algorithm generates and utilizes second derivative information about the error function. Moreover, there is no possibility for oscillation of the error, which can happen with BP. Oscillations of the error can occur in BP if the learning rate and momentum are not chosen properly. There is a potential for oscillation anytime a full line minimization is not carried out. And, unlike BP, BFGS has no learning rate or momentum parameters that must be chosen by the user.

Generalized Simulated Annealing. Descriptor selection using CNN performance as the selection criterion is an enormous task. To investigate all possible descriptor combinations is impractical, and therefore an optimization technique is needed to reduce the time involved in performing this task. When choosing an optimization technique several factors must be considered. These include the simplicity of the algorithm, the likelihood of convergence to a global optimum, and the speed of the algorithm. Considering these factors, generalized simulated annealing (GSA) is an appropriate choice. GSA is a simple, semirandom, iterative improvement technique that does not converge to the first likely candidate.

Since the GSA method has been thoroughly discussed in the literature[20-22] only a brief description of how it pertains to this specific problem will be given. The GSA optimization routine is based on the physical process of annealing. The position of the atoms in an annealing solid represent the parameters being optimized, and the energy of the solid represents the cost function being optimized. As in annealing, the lowest cost function (energy configuration) is obtained by lowering the temperature slowly. In GSA, the initial temperature is established using a control parameter, β, and it is adjusted using an estimated minimum of the cost function, $C(o)$. Both are selected by the user.

If $C(s)$ is the current CNN cost function for set of parameters s, then a new set of parameters r is accepted if the new cost $C(r)$ is less than $C(s)$. If $C(r) > C(s)$, then the new parameters are accepted with a probability P which comes from a Boltzmann distribution

$$P = \exp\left(\frac{-\beta[C(r) - C(s)]}{C(s) - C(o)}\right) \qquad (7)$$

The empirical control parameter β is adjusted to control the number of detrimental steps accepted, and $C(o)$ is an estimate of the best cost function.

In the beginning of the optimization a large percentage of the detrimental steps are accepted, thus representing a high temperature. As the algorithm proceeds, the temperature is slowly reduced, decreasing the percentage of detrimental steps to be accepted. A detrimental step is accepted when P from eq 7 is greater than a random number generated uniformly on [0,1]. The acceptance of some detrimental steps allows GSA to move away from local minima. As $C(s)$ approaches the estimated minimum, $C(o)$, the P values decrease and detrimental steps have a smaller probability of being accepted. At this point the algorithm is theoretically converging to global conditions. Although no optimization technique is guaranteed to obtain global conditions, GSA is an excellent technique that has been shown to perform better than other optimization procedures in a variety of situations.[21,23]

Selecting an appropriate cost function that adequately describes the problem is perhaps the most important step in an optimization. Many different possibilities were investigated for the CNN descriptor selection problem. The cost function that yielded CNN models with the most predictive power is represented as

$$\text{Cost} = T(\text{err}) + |T(\text{err}) - \text{CV}(\text{err})| \qquad (8)$$

where T(err) is the root mean square (rms) error for the biological activity values for the members of the training set, and CV(err) is the rms error for the members of the cross-validation set. The sets of descriptors that support models with the lowest T(err) but that simultaneously have a similar

CV(err) seemed to have superior predictive power. As eq 8 reveals, a cross-validation set that is indicative of the entire set of compounds being investigated is essential for CNN descriptor selection by the approach.

Two modifications were made to the GSA algorithm for this study. The first modification converted GSA from a continuous to a discrete function optimization routine for descriptor selection. In a previous study, a discrete function was given continuity by changing only a fraction (50−75%) of the variables,[23] thereby passing some information from the previous subset to the next subset of variables. The technique worked reasonably well for that study and was applied to the descriptor selection routine.

The second modification was to bias the descriptors selected for the subset in order to speed the optimization procedure. The descriptor selection is achieved by calculating a quality value for each descriptor. The quality value for a particular descriptor is the cost function calculated by eq 8 when that descriptor is used as the sole input, with all other layers of the neural network the same. For example, if a final 5:3:1 architecture is desired, then each descriptor is subjected to a 1:3:1 neural network, and the resulting cost function is used as the quality value. This value is used to bias the descriptor selection during the optimization. The values are scaled so that the descriptor with the lowest (best) quality value has a 75% chance of being accepted, and the descriptor with the highest quality value has a 25% chance of being accepted into the new subset. These percentages were obtained empirically. The quality values are mapped onto the interval [0.288, 1.39], then the probability of acceptance is calculated by

$$P = \exp(-\mathrm{SQUAL}) \qquad (9)$$

where SQUAL is the scaled quality value. This number is compared to a random number on the interval [0,1]. The new descriptor is accepted if P is greater than the random number.

Each selected descriptor subset is used to train a neural network, and the resulting cost function is calculated. The BFGS training routine is a directional optimization routine, and therefore the cost function is dependent on the initial weights and biases. To improve the likelihood of locating the global minimum, the initial weights and biases were optimized using GSA. No modifications had to be made to GSA for optimizing the starting parameters of a CNN. However, to reduce the amount of time involved, a combination of GSA and BFGS was used to optimize the starting weights and biases. An iteration consists of a random selection of weights and biases using GSA followed by a short training of the BFGS network (10−20% of normal training) in which the cost function shown in eq 8 is calculated. The next step is to randomly select a new set of weights and biases in the neighborhood of the previous set followed by another short BFGS training session and the cost function calculation. After 200 iterations the optimal starting weights and biases are used to fully train the CNN and calculate the cost function for the given subset of descriptors.

EXPERIMENTAL SECTION

The procedure used in this study is given in the flow diagram shown in Figure 2. Most computations were performed using the ADAPT software system on a Sun 4/110 workstation.[24,25] However, the annealing programs and the neural network[10] computations were performed on a DEC 3000 AXP Model 500 workstation. All programs were written in Fortran 77.

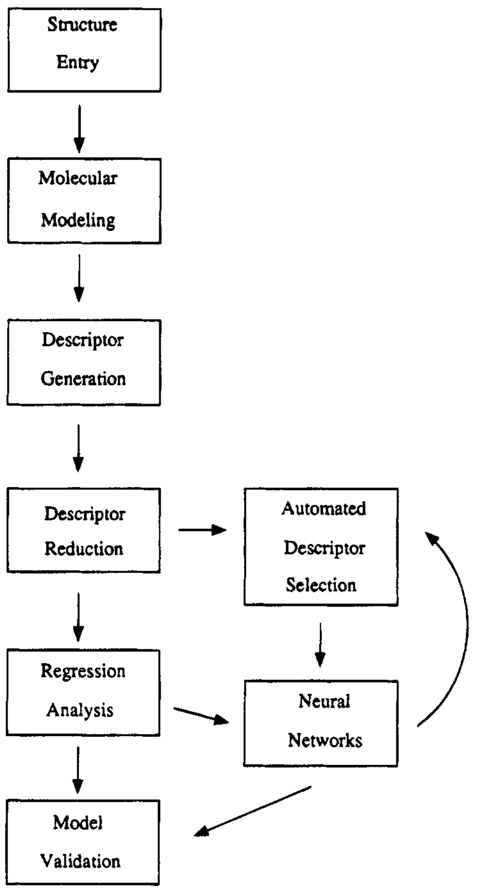

Figure 2. The flow diagram for the development of toxicity prediction models.

Two programs were developed for this study. The first, called ANNDES, uses GSA to optimize a subset of descriptors. For each subset of descriptors, a computational neural network training is performed to compute the cost function. The initial weights and biases are optimized using the program ANN.

The data set consisted of 140 functionally related compounds taken from ref 3. The data were split into a training set (130 compounds) and a prediction set (10 compounds) the same as ref 3 for comparison. The dependent variable was $-\log(\mathrm{LC}_{50})$, where LC_{50} is the concentration that causes 50% mortality in fathead minnows. The 96-h LC_{50} values of these compounds were taken from the literature.[3] Compounds and the corresponding $-\log(\mathrm{LC}_{50})$ values that were used in this study are presented in Table 1. For the multiple linear regression portion of the study, the training set consisted of compounds **1−130**, and the prediction set consisted of compounds **131−140**. For the CNN portion, the training set and the 13 member cross-validation set were taken from compounds **1−130**, and the prediction set consisted of compounds **131−140** represented by the superscript *b* in Table 1. The cross-validation set consisted of 13 randomly chosen compounds represented by the superscript *a* in Table 1 and was believed to be representative of the entire data set.

Table 1. Compounds and Corresponding $-\log(LC_{50})$ Values Used in This Study[c]

no.	compound	$-\log(LC_{50})$	no.	compound	$-\log(LC_{50})$
1	benzene	3.40	71	1-formyl-2,4-dichlorobenzene	4.99
2	bromobenzene	3.89	72	1-formyl-4-chlorobenzene	4.81
3	chlorobenzene	3.77	73	1-formyl-2-nitrobenzene	4.02
4	hydroxybenzene	3.51	74	1-formylbenzene	4.14
5	1,2-dichlorobenzene	4.40	75	1-formyl-2,4-dimethoxybenzene	3.92
6	1,3-dichlorobenzene[a]	4.30	76	1-formyl-2-hydroxy-5-bromobenzene[a]	5.19
7	1,4-dichlorobenzene	4.62	77	1-formyl-2-hydroxy-5-chlorobenzene	5.31
8	1-chloro-2-hydroxybenzene	4.02	78	1-formyl-2-hydroxybenzene	4.73
9	1-chloro-3-methylbenzene	3.84	79	1-formyl-2-methoxy-4-hydroxybenzene	4.02
10	1-chloro-4-methylbenzene	4.33	80	1-formyl-3-methoxy-4-hydroxybenzene	4.81
11	1,3-dihydroxybenzene	3.04	81	1-formyl-2-hydroxy-4,6-dimethoxybenzene	4.83
12	1-hydroxy-3-methoxybenzene	3.21	82	1-amino-2,3,4,5,6-pentafluorobenzene	3.69
13	1-hydroxy-2-methylbenzene	3.77	83	1-fluoro-4-nitrobenzene	3.70
14	1-hydroxy-3-methylbenzene	3.29	84	1-amino-4-fluorobenzene	3.82
15	1-hydroxy-4-methylbenzene	3.58	85	1-formyl-pentafluorobenzene[a]	5.25
16	1-hydroxy-4-nitrobenzene	3.36	86	1-formyl-2-chloro-6-fluorobenzene[a]	4.23
17	1,4-dimethoxybenzene	3.07	87	1-acyl-4-chloro-3-nitrobenzene	4.56
18	1,2-dimethylbenzene	3.48	88	1-acyl-2,4-dichlorobenzene	4.21
19	1,4-dimethylbenzene	4.21	89	1-acylbenzene	2.87
20	1-methyl-2-nitrobenzene	3.57	90	3-nitrobenzonitrile	3.39
21	1-methyl-3-nitrobenzene	3.63	91	4-nitrobenzonitrile	3.79
22	1-methyl-4-nitrobenzene	3.76	92	2-amino-4-nitrotoluene	3.34
23	1,3-dinitrobenzene	4.38	93	2-amino-6-nitrotoluene	3.48
24	1-amino-2-methyl-3-nitrobenzene[a]	3.48	94	3-amino-4-nitrotoluene	3.80
25	1-amino-2-methyl-4-nitrobenzene[a]	3.24	95	4-amino-2-nitrotoluene	3.77
26	1-amino-2-methyl-5-nitrobenzene	3.35	96	3-methyl-2-nitrophenyl	3.52
27	1-amino-2-methyl-6-nitrobenzene	3.80	97	5-methyl-2-nitrophenol	3.51
28	1-amino-3-methyl-6-nitrobenzene	3.80	98	1,5-dimethyl-2,4-dinitrobenzene	4.39
29	1-amino-2-nitro-4-methylbenzene	3.79	99	2-amino-4,6-dinitrotoluene	4.12
30	1-amino-3-nitro-4-methylbenzene	3.77	100	3-amino-2,4-dinitrotoluene	4.21
31	1,2,3-trichlorobenzene	4.89	101	3-amino-2,6-dinitrotoluene	4.26
32	1,2,4-trichlorobenzene[a]	5.00	102	4-amino-2,6-dinitrotoluene	4.46
33	1,3,5-trichlorobenzene	4.74	103	2,4-dinitro-5-methylphenol	4.92
34	1,3-dichloro-4-hydroxybenzene[a]	4.30	104	1,3,5-trinitrobenzene	5.29
35	1,2-dichloro-4-methylbenzene	4.74	105	1-chloro-2-propanol	2.58
36	1,3-dichloro-4-methylbenzene	4.54	106	2,2,2-trichloroethanol	2.70
37	1-hydroxy-2,4-dimethylbenzene	3.86	107	2,3-dibromopropanol	3.49
38	1-hydroxy-2,6-dimethylbenzene	3.75	108	cyclohexanol	2.15
39	1-hydroxy-3,4-dimethylbenzene	3.90	109	2-phenoxyethanol	2.60
40	1-hydroxy-2,4-dinitrobenzene	4.04	110	acetone	0.85
41	1,2,4-trimethylbenzene[a]	4.21	111	2-butanone	1.35
42	1-methyl-2,3-dinitrobenzene	5.01	112	2-pentanone	1.75
43	1-methyl-2,4-dinitrobenzene	3.75	113	3-methyl-2-butanone	2.00
44	1-methyl-2,6-dinitrobenzene	3.99	114	5-methyl-2-hexanone	2.86
45	1-methyl-3,4-dinitrobenzene	5.08	115	4-methyl-2-pentanone	2.29
46	1-methyl-3,5-dinitrobenzene	3.91	116	3,3-dimethyl-2-butanone	3.07
47	1-amino-2-methyl-3,5-dinitrobenzene[a]	4.12	117	acetophenone	2.87
48	1-amino-2-methyl-3,6-dinitrobenzene	5.34	118	benzophenone	4.09
49	1-amino-2,4-dinitro-3-methylbenzene	4.26	119	2,3,4-trichloroacetophenone	5.00
50	1-amino-2,6-dinitro-3-methylbenzene	4.21	120	2,4-dichloroacetophenone	4.16
51	1-amino-2,6-dinitro-4-methylbenzene	4.18	121	*tert*-butylmethyl ether	2.10
52	1-amino-3,5-dinitro-4-methylbenzene	4.46	122	diisopropyl ether	3.05
53	1,3,5-tribromo-2-hydroxybenzene	4.70	123	2,6-dimethoxytoluene	3.88
54	1,2,3,4-tetrachlorobenzene	5.43	124	diphenyl ether[a]	4.63
55	1,2,4,5-tetrachlorobenzene	5.85	125	*p*-nitrophenyl phenyl ether	4.91
56	1-methyl-2,4,6-trinitrobenzene	4.88	126	1,2-dichloroethane	2.92
57	1-hydroxy-2,3,4,5,6-pentachlorobenzene	6.06	127	1,1,2-trichloroethane	3.21
58	1-amino-4-bromobenzene	3.56	128	1,1,2,2-tetrachloroethane	3.92
59	1-amino-3,4-dichlorobenzene	4.33	129	pentachloroethane	4.44
60	1-amino-2,4-dinitrobenzene	4.07	130	hexachloroethane[a]	5.19
61	1-amino-2-chloro-4-methylbenzene	3.60	131	methylbenzene[a]	3.32
62	1-amino-2-chloro-4-nitrobenzene	3.93	132	1-amino-3-nitro-4-methylbenzene[b]	3.65
63	1-amino-2,3,4-trichlorobenzene[a]	4.74	133	1-chloro-2-methyl-4-hydroxybenzene[b]	4.27
64	1,3,5-trichloro-2,4-dinitrobenzene	6.09	134	1-aminobenzene[b]	2.84
65	1-amino-2,3,5,6-tetrachlorobenzene	5.93	135	1-formyl-2-hydroxy-3,5-dibromobenzene[b]	5.52
66	1-cyano-3,5-dibromo-4-hydroxybenzene	4.38	136	5-amino-2,4-dinitrotoluene[b]	4.91
67	1-cyano-2-amino-5-chlorobenzene	3.73	137	1-chloronaphthalene[b]	4.85
68	1-cyano-2-chloro-6-methylbenzene	4.00	138	pentachloronaphthalene[b]	6.01
69	1-cyano-2-methylbenzene	3.42	139	carbon tetrachloride[b]	3.75
70	1-formyl-2-chloro-5-nitrobenzene	4.72	140	1-formyl-2-fluorobenzene[b]	4.96

[a] Compound used in the cross-validation set for CNN. [b] Compound used in the prediction set for MLR and CNN. [c] The unit of LC_{50} is [mol/L]. The $-\log(LC_{50})$ values were taken from ref 3.

The structure entry was performed by sketching the compounds on a graphics terminal and then storing them as connection tables. The geometries were optimized using the semiempirical molecular orbital program (MOPAC)[26] with the PM3 Hamiltonian.[27]

A total of 165 descriptors were generated for each compound using the ADAPT software. The descriptor pool contained topological, electronic, and geometric descriptors. Topological descriptors include simple and valence-corrected connectivity indices. Some of the geometric descriptors were length-to-breadth ratios, radius of gyration, and molecular volume. Electronic descriptors included such quantities as the most negative atomic charge[28,29] in the molecule and dipole moment.

The number of descriptors was reduced to a set of 65 using standard procedures. Descriptors that contained little or no useful information and descriptors that were similar in value and content to other descriptors were rejected. Therefore descriptors that had the same value for nearly every compound were rejected, as were descriptors that were highly correlated ($R > 0.95$) with other descriptors. When pairwise correlation was encountered, the descriptor that had the more obvious feature representation or was easier to calculate was retained.

Models were generated using multiple linear regression. Leaps-and-bounds regression[30] was used to find a subset of descriptors that yielded a statistically sound model. The best descriptor subset selected based on multiple linear regression was used to build a CNN model, which improved the rms error and the predictive power of the model. Although the number of variables are the same in the regression model and the CNN model, there are more adjustable parameters in the CNN model, since each connection is considered an adjustable coefficient. The higher number of adjustable parameters and the nonlinear characteristics of the CNN model allow for more mathematical flexibility and an improved model is often created.

CNN models were also generated using an automated descriptor selection routine that used the CNN results as the selection criterion (ANNDES) instead of multiple linear regression. Unique descriptor subsets that yielded superior CNN models were found.

The multiple linear regression models were validated using standard statistical techniques. These techniques included inspection of the residuals, standard deviation, and multiple correlation coefficient. Both regression and CNN models were validated using external prediction. The prediction set was not used for descriptor selection, descriptor reduction, or model development, and it therefore represents a true unknown data set. The rms error was computed for the prediction set of the CNN models, and the standard deviation was computed for the regression models.

RESULTS AND DISCUSSION

Numerous multiple linear regression models were created, and the quality was determined by examining the multiple correlation coefficient (R), standard deviations of regression, and the number of variables in the model. The best model contained five descriptors, with $R = 0.914$ and $S = 0.370$. Models that contained a larger set of descriptors did not significantly improve the R and standard deviation and therefore were not used.

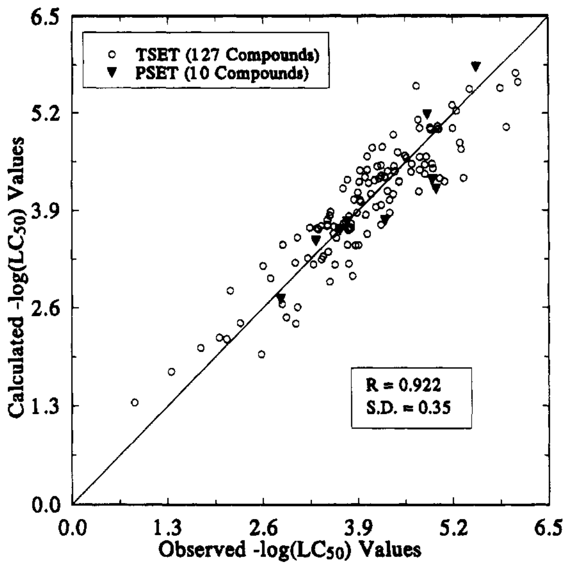

Figure 3. Calculated versus observed $-\log(LC_{50})$ values of a five-descriptor MLR model. TSET is the training set, PSET is the prediction set, R is the correlation coefficient of regression, and SD is the standard deviation of regression.

The final model contained one electronic, one geometric, and three topological descriptors. The electronic descriptor was the charge on the most negative atom (QNEG).[29] The geometric descriptor was the partial positive surface area (PPSA-1) developed by Stanton and Jurs.[31] The topological descriptors found were an aldehyde group contribution descriptor, a molecular connectivity descriptor ($^1\chi$), and a path count and length descriptor. $^1\chi$ is a path one valence-corrected molecular connectivity descriptor that was introduced by Randić[32] and applied to structure–activity analysis by Kier and Hall.[33] $^1\chi$ is a measure of the degree of branching in a molecule and thus correlates to many physical properties such as boiling points and retention indices. The path descriptor, originally reported by Randić,[34] is the total weighted number of paths between 0 and 45 in the structure divided by the number of atoms. The total weighted number of paths is given as $N(1) + N(2)\exp(-1) + N(3)\exp(-2)$ ... where $N(k)$ is number of paths of length k.

The model was improved by detecting and rejecting outliers. Outlier detection was performed using traditional regression diagnostics.[35] Standard statistical values such as residuals, standardized residuals, studentized residuals, and leverage were computed to detect possible outliers. Three compounds, 1-amino-4-bromobenzene (**58**), 1-amino-2,3,5,6-tetrachlorobenzene (**65**), and 1-amino-4-fluorobenzene (**84**), were found to be outliers, and when they were removed, the R value and the standard deviation of regression were improved to 0.922 and 0.351, respectively. Figure 3 shows the calculated versus the observed $-\log(LC_{50})$ values for this model.

The five descriptors found in this model were then fed to a CNN in an attempt to improve the predictive ability. Figure 4 is the calculated versus observed $-\log(LC_{50})$ values of the 5:3:1 architecture neural network. The program ANN was used to optimize the starting weights and biases. The quality of the model was assessed by calculating the residuals [actual − predicted] values of $-\log(LC_{50})$] of the prediction set compounds.

The regression model, the CNN model, and a group contribution model developed by Gao[3] were then compared.

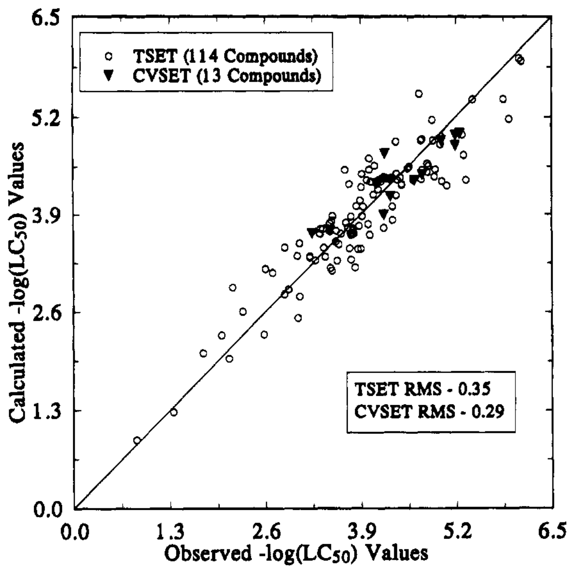

Figure 4. Calculated versus observed $-\log(LC_{50})$ values of a MLR selected five-descriptor CNN model. TSET is the training set, CVSET is the cross-validation set, and rms is the root mean square error.

Table 2. Residuals for the Prediction Set Compounds Using Three Different Models

	residuals[a]		
compound	Gao model[b]	MLR model[c]	CNN model
methylbenzene	0.06	−0.17	−0.28
1-amino-3-nitro-4-methylbenzene	0.29	−0.01	0.03
1-chloro-2-methyl-4-hydroxybenzene	0.07	0.50	0.54
1-aminobenzene	0.07	0.13	−0.05
1-formyl-2-hydroxy-3,5-dibromobenzene	0.01	−0.30	0.40
5-amino-2,4-dinitrotoluene	0.51	0.60	0.59
1-chloronaphthalene	0.48	−0.33	0.57
pentachloronaphthalene	−0.49	−1.14	−0.36
carbon tetrachloride	0.33	0.00	0.00
1-formyl-2-fluorobenzene	0.81	0.77	0.56
rms error	0.40	0.53	0.41

[a] Residual = observed − predicted of the $-\log(LC_{50})$ value. [b] Group contribution model that uses 16 substituents as descriptors.[3] [c] The MLR and CNN models use five descriptors.

Gao used 16 structural groups to develop a model. Table 2 provides a detailed comparison of the predictions of the three approaches. The 16-descriptor group contribution model gave the smallest overall rms error of 0.40. The CNN model using only five descriptors gave an rms error of 0.41.

ANNDES was then used to seek a better 5:3:1 CNN model. Since ANNDES is such a computationally intensive routine, it was believed that a good initial guess would improve the results. Therefore the quality value was calculated for each descriptor, and the descriptors with the five best values were used as the starting point. ANNDES was started several times using different starting weights and biases for the neural network routine. Table 3 shows the rms errors for the training, cross-validation, and prediction sets for three different ANNDES models. In general, the optimized models were either comparable or slightly better than the five descriptor CNN model found using regression. Figure 5 shows the calculated vs observed $-\log(LC_{50})$ values for the ANNDES 1 model.

Table 4 is a list of the eight descriptors used for the three models found using ANNDES. The only descriptor present

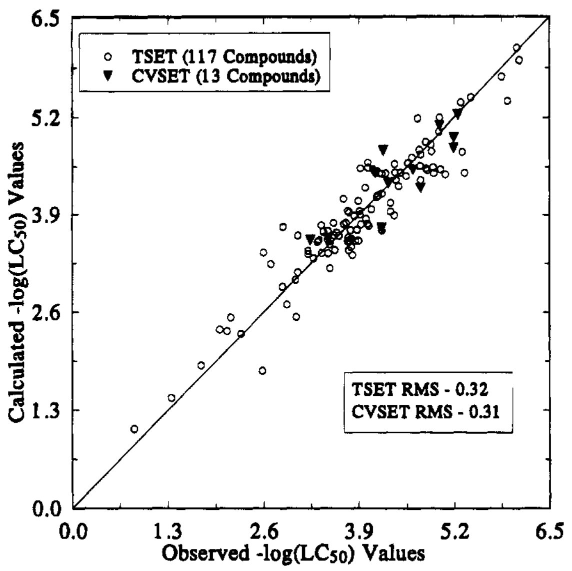

Figure 5. Calculated versus observed $-\log(LC_{50})$ values using a CNN model with the descriptor subset selected by generalized simulated annealing (ANNDES 1). TSET is the training set, CVSET is the cross-validation set, and rms is the root mean square error.

Table 3. The rms Errors of the Three Models Found Using Generalized Simulated Annealing

	rms errors		
model	training set	cross-validation set	prediction set
ANNDES 1	0.32	0.31	0.35
ANNDES 2	0.36	0.33	0.39
ANNDES 3	0.40	0.37	0.28

Table 4. Definitions of the descriptors Used in the Three Models Found Using Generalized Simulated Annealing

	ANNDES model			
descriptor	1	2	3	descriptor definition
CHO	x	x	x	aldehyde group contribution
PPSA-2	x	x	x	total charge weighted partial positive surface area[a]
V5C			x	five path valence molecular connectivity[b]
S5C	x	x		five path simple molecular connectivity[b]
N5C	x			the total count of five path lengths[b]
NN			x	the number of nitrogens
MREF	x	x	x	the whole-molecular molar refraction values[c]
SCSP2		x		the smallest sp^2 carbon atomic charge[d]

[a] See ref 31. [b] See ref 33. [c] See ref 36. [d] See ref 28.

that was also found in the MLR model is the aldehyde (CHO) group contribution descriptor. This descriptor was shown to be very important in all four models. It was also shown to have the highest toxicity contribution of all the chemical groups used in Gao's group contribution model.[3]

The geometric descriptor from the regression model (PPSA-1) was not found in any of the ANNDES models. However, the PPSA-2 descriptor, a weighted version of PPSA-1, was found for all three of the ANNDES models. The topological $^1\chi$ descriptor, a path one valence descriptor, was found for the regression model but not for the ANNDES models. However, all the ANNDES models contained a molecular connectivity descriptor of some kind. Also, the electronic descriptor found by regression, QNEG, was not found in any of the ANNDES models, but the atomic charge was shown to be important. The descriptors PPSA-2 and SCSP2 were found by ANNDES and further confirmed the importance of atomic charge in good toxicity models.

Although the models found using regression and ANNDES contain different descriptors, there are obvious similarities for each model. The most important features found in all the models used for predicting toxicity are the aldehyde groups, molecular connectivity, and atomic charge.

CONCLUSION

An automated descriptor selection routine for CNNs using GSA has been shown to be successful. CNN models found using this algorithm are similar or superior to CNN models found using regression techniques, which might be due to nonlinear features found in the QSAR. Nonlinear features cannot be successfully modeled using multiple linear regression without using extra nonlinear terms but seemingly do not pose a problem for the CNN based algorithm, ANNDES. Therefore there are many advantages to using generalized simulated annealing for feature selection. Most importantly, the models found by ANNDES seem to be superior to those found using regression.

However, there are disadvantages to using this technique. The main disadvantage is the large amount of time required to perform the optimization. Since time is a concern, the number of iterations must be limited. A small number of iterations may cause convergence to local conditions due to a quick decrease in the annealing temperature known as quenching. The fact that different descriptor subsets are found for each ANNDES run reveals the fact that there are many local minima and suggests local minima convergence by GSA. Future studies may investigate different optimization techniques for automated descriptor subset selection. An optimization technique that converges quickly to good conditions would probably be beneficial for this method.

The descriptor subset for each model found using regression and CNN is unique, but there are many similarities. Therefore, as with any QSAR model, features that are important to toxicity are often uncovered using this technique. The descriptors in this study reveal the importance of an aldehyde group for toxicity, which is confirmed in Gao's study. Also molecular connectivity and atomic charge were found to be important.

REFERENCES AND NOTES

(1) Hansch, C.; Leo, A. J. *Substituent Constants for Correlation Analysis in Chemistry and Biology*; John Wiley and Sons: New York, NY, 1979.

(2) Free, S. M.; Wilson, J. W. A mathematical contribution to structure-activity studies. *J. Med. Chem.* **1964**, *7*, 395.

(3) Gao, C.; Govind, R.; Tabak, H. H. Application of the group contribution method for predicting the toxicity of organic chemicals. *Environmental Toxicol. Chem.* **1992**, *11*, 631–636.

(4) Zupan, J.; Gasteiger, J. Neural networks: A new method for solving chemical problems or just a passing phase? *Anal. Chim. Acta* **1991**, *248*, 1–30.

(5) Borman, S. Neural network applications in chemistry begin to appear. *Chem. Eng. News* **1989**, *67*, 24–28.

(6) Lacey, M. E. Neural network technology and its application in chemical research. *Tet. Comp. Meth.* **1990**, *3*, 119–128.

(7) Wikel, J. H.; Dow, E. R. The use of neural networks for variable selection in QSAR. *Bioorg. Med. Chem. Lett.* **1993**, *3*, 645–51.

(8) Andrea, T. A.; Kalayeh, H. Applications of neural networks in quantitative structure-activity relationships of dihydrofolate reductase inhibitors. *J. Med. Chem.* **1991**, *34*, 2824–2836.

(9) Salt, D. W.; Salt, D. W.; Yildiz, N. The use of artificial neural networks in QSAR. *Pesticide Sci.* **1992**, *36*, 161.

(10) Xu, L.; Ball, J. W.; Dixon, S. L.; Jurs, P. C. Quantitative structure-activity relationships for toxicity of phenols using regression analysis and computational neural networks. *Environmental Toxicol. Chem.* **1994**, *13(5)*, 841–851.

(11) Aoyama, T.; Suzuki, Y.; Ichikawa, H. Neural networks applied to quantitative structure-activity relationships analysis. *J. Med. Chem.* **1990**, *33*, 2583.

(12) Livingstone, D. J.; Manallack, P. T. Statistics using neural networks: Chance effects. *J. Med. Chem.* **1993**, *36*, 1295–1297.

(13) Schlick, T. Optimization methods in computational chemistry. In *Reviews in Computational Chemistry*; Lipkowitz, B., Boyd, B., Eds.; VCH Publishers: New York, 1992; Vol. 3, Chapter 1.

(14) Broyden, C. G. The convergence of a class of double-rank minimization algorithms. *J. Inst. Maths. Appl.* **1970**, *6*, 76.

(15) Fletcher, R. A new approach to variable metric algorithms. *Comput. J.* **1970**, *13*, 317.

(16) Goldfarb, D. A family of variable-metric methods derived by variational means. *Math. Comput.* **1970**, *24*, 23.

(17) Shannon, D. F. Conditioning of quasi-Newton methods for function minimization. *Math. Comput.* **1970**, *24*, 647.

(18) Fletcher, R. *Practical Methods of Optimization, Vol. 1, Unconstrained Optimization*; Wiley: New York, 1980.

(19) Jansson, P. A. Neural networks: An overview. *Anal. Chem.* **1991**, *63*, 357A–362A.

(20) Bohachevsky, I. O.; Johnson, M. E.; Stein, M. L. Generalized simulated annealing for function optimization. *Technometrics* **1986**, *28*, 209–217.

(21) Kalivas, J. H.; Sutter, J. M.; Roberts, N. Global optimization by simulated annealing with wavelength selection for ultraviolet-visible spectrophotometry. *Anal. Chem.* **1989**, *61*, 2024–2030.

(22) Kalivas, J. H. Generalized simulated annealing for calibration sample selection from an existing set and orthoginization of undesigned experiments. *J. Chemometrics* **1991**, *5*, 37–48.

(23) Sutter, J. M.; Kalivas, J. H. Comparison of forward selection, backward elimination, and generalized simulated annealing for variable selection. *Microchem. J.* **1993**, *47*, 60–66.

(24) Stuper, A. J.; Brugger, W. E.; Jurs, P. C. *Computer-Assisted Studies of Chemical Structure and Biological Function*; Wiley-Interscience: New York, 1979.

(25) Jurs, P. C.; Chou, J. T.; Yuan, M. In *Computer-Assisted Drug Design*; Olson, E. C., Christoffersen, R. E., Eds.; The American Chemical Society: Washington, DC, 1979; pp 103–129.

(26) Stewart, J. P. P. MOPAC 6.0, *Quantum Chemistry Program Exchange*; Indiana University, Bloomington, IN, Program 455.

(27) Stewart, J. P. P. Mopac: A semiempirical molecular orbital program. *J. Comput.-Aided Mol. Des.* **1990**, *4*, 1.

(28) Dixon, S. L.; Jurs, P. C. Atomic charge calculations for quantitative structure-property relationships. *J. Comput. Chem.* **1992**, *13*, 492–504.

(29) Abraham, R. J.; Griffiths, L.; Loftus, P. Approaches to charge calculations in molecular mechanics. *J. Comput. Chem.* **1982**, *3*, 407.

(30) Furnival, G.; Wilson, R. Regression by leaps and bounds. *Technometrics* **1974**, *16*, 499–511.

(31) Stanton, D. T.; Jurs, P. C. Development and use of charged partial surface area structural descriptors in computer-assisted quantitative structure-property relationship studies. *Anal. Chem.* **1990**, *62*, 2323–2329.

(32) Randić, M. On characterization of molecular branching. *J. Am. Chem. Soc.* **1975**, *97*, 6609–6615.

(33) Kier, L. B.; Hall, L. H. *Molecular Connectivity in Structure-Activity Analysis*; Research Studies Press Ltd.; John Wiley & Sons: 1986; pp 18–20.

(34) Randić, M. Search for all self-avoiding paths for molecular graphs. *Comput. Chem.* **1979**, *3*, 5.

(35) Belsley, D. A.; Kuh, E.; Welsch, R. E. *Regression Diagnostics*; John Wiley & Sons: New York, 1980.

(36) Vogel, A. I. *Textbook of Organic Chemistry*; Chaucer, 1977; p 1034.

CI9400690