Evolutionary Chromatographic Law Identification by Recurrent Neural Nets [1]

Alessandro Fadda and Marc Schoenauer
CMAP - Ecole Polytechnique - 91128 Palaiseau Cedex - FRANCE
{Alessandro.Fadda,Marc.Schoenauer}@polytechnique.fr

Abstract

Analytic chromatography is a physical process whose aim is the separation of the components of a chemical mixture, based on their different affinities for some porous medium through which they are percolated. This paper presents an application of evolutionary recurrent neural nets optimization to the identification of the internal law of chromatography. New mutation operators involving the parameters of a single neuron are introduced. Furthermore, the strategy for using of the different kind of mutation takes into account the past history of the neural net at hand. The first results for one- and two-component mixtures demonstrate the basic feasibility of the recurrent neural net approach. A strategy to improve the robustness of the results is presented.

1 INTRODUCTION

In many fields of engineering science, a physical phenomenon is modeled by a law which is not precisely known. The difficulty of the identification of such a law can be estimated from the available information about the problem:

- Known properties of the target function can turn the identification problem into some parametric optimization problem (e.g. the coefficients of linear regression models), or can provide a natural basis of the search space (e.g. $sin(kx), k \in I\!N$ for periodic functions). Otherwise, a general model must be chosen to specify a search space, large enough to contain a good solution, but ensuring a tractable search.

- The target function sometimes is experimentally accessible, i.e. the values taken by the function for some inputs are known. Identification is then termed *numerical regression*. In other cases, only the effects of the target function (through some simulation of a physical phenomenon) are accessible. The identification problem is then an *inverse problem*. And most inverse

[1] Presented at the 4^{th} *Annual Conference on Evolutionary Algorithms*, San Diego, March 1995.

problems cannot be set and solved explicitly in terms of the target function.

The identification problem then turns into an optimization problem, and the choice of best algorithm depends on both the model for the function and the available data:

- quadratic minimization for numerical regression, from simple linear models to multiple nonlinear methods (see Bevington 1969);
- complex gradient-based optimization algorithms when derivatives exist and are computable (see Polak 1971, for general parametric optimization, or Le Cun 1987, for back-propagation in feed-forward neural nets);
- reinforcement learning when a reinforcement signal is available, that qualifies the value given by the function for some given inputs, as in Barto et al. (1983) or Watkins and Dayan (1992);
- stochastic optimization, for very large search spaces, with little regularity of optimization problem, like neural nets models (see Yao 1993, for a general survey of this approach), or genetic programming, described in Koza (1994).

This paper addresses the problem of identification of the internal law of *chromatography*, a physical process whose aim is the separation of the components of a chemical mixture, based on their different affinities for some porous medium through which they percolate. Neither a model for that law, nor examples of values it takes, are available from chemical engineers. This typically is an inverse problem.

This problem is handled by applying evolutionary computation to a recurrent neural net model for the function to identify. Evolutionary computation has been used to evolve neural nets in many applications (see again Yao 1993). One benefit of using evolutionary methods for inverse problems is that the fitness function can take into account all available data, as is highlighted in this paper.

The chromatographic process is introduced in the section 2, together with the system of partial differential equations governing it and its numerical approximation. The characteristics of the inverse problem naturally led to propose the recurrent neural nets model, optimized by an evolutionary algorithm. This algorithm is presented in section 3, and its differences with the GNARL algorithm (Angeline 1993) are highlighted, including the introduction of new mutation operators and fine control of the strategy of application for these operators. The results of this algorithm on the inverse problem of chromatography are discussed in section 4. The sensitivity of the identification process with respect to the experimental conditions are outlined, and a method is proposed to overcome this difficulty through the design of the fitness functions.

2 BACKGROUND

Analytic chromatography

Chromatography is a technique for separating the different *components* of a chemical mixture[1] by selective absorption in a solid porous medium[2].

The moving fluid mixture is percolated through the motionless medium in a column, considered 1-dimensional. The various components of the mixture propagate in the column at different speeds, because of their different affinities with the solid medium. In the ideal case, every component has its own fixed propagation speed, and, if the column is sufficiently long, different components come out at the end of the column at different times: they are separated. But the speed of a component depends on every other component present in the mixture, which makes the fundamental equations of a chromatographic process highly nonlinear.

The art of chromatographic separation requires predicting the different proportions of every component of the mixture at the end of the column (called the *chromatogram*) along the experiment.

The direct problem

Given these equilibrium equations for all M components of the mixture, the mass conservation equations in the column can be written as the $M \times M$ system of partial differential equations (1)(Valentin and Guiochon 1976).

$$\begin{cases} \frac{\partial w(z,t)}{\partial z} + \frac{\partial f(w(z,t))}{\partial t} = 0 \text{ for } (z,t) \in [0,L] \times [0,T] \\ w(0,t) = w^{inj}(t) \text{ for } t \in [0,T] \\ w(z,0) = w^{init}(z) \text{ for } z \in [0,L] \end{cases} \quad (1)$$

where $w(z,t) = (w_1(z,t), w_2(z,t), ..., w_M(z,t)) \in I\!R^M$ is the vector of concentrations of the components of the mixture, w^{inj} and w^{init} are the *experimental conditions*, respectively, the initial state of the concentrations along the column and the injection at the beginning of the column, and L and T respectively denote the length of the column and the duration of the experiment.

The mathematical flow f in (1) is a function from $I\!R^M$ in $I\!R^M$. It is termed the *isotherm* function. When the isotherm function is given, it is known (James 1990) that system (1) is a well-posed nonlinear hyperbolic system, with positive eigenvalues. This enables a numerical resolution of (1), using a standard nonlinear hyperbolic numerical scheme such as Godunov scheme, for example (Godunov 1959, Rouchon et al. 1989).

The Godunov scheme

Given a space step Δz and a time step Δt such that $E\Delta z = L$ and $S\Delta t = T$ for some integers E and S, the Godunov scheme for the system (1) is

$$\begin{cases} w_j^{n+1} = w_j^n - \lambda(f(w_j^n) - f(w_{j-1}^n)) \\ w_j^0 = w_{init}(j\Delta z) \\ w_0^n = w_{inj}(n\Delta t) \end{cases} \tag{2}$$

where w_j^n is an approximation of the solution of (1) at $(n\Delta z, j\Delta t)$. It is known that the discrete solution of (2) converges to w, solution of (1) as $\Delta z, \Delta t$ goes to 0, with $\lambda = \Delta z/\Delta t$ fixed.

For prescribed $f, w^{inj}(t)$ and $w^{init}(t)$, system (2) can be numerically solved and an approximation of the chromatogram $w_i(L,t)_{0\leq t\leq T}$ can be derived: The direct problem is numerically solved.

$$(f, w^{init}(z)_{z\in[0,L]}, w^{inj}(t)_{t\in[0,T]}) \longrightarrow w(L,t)_{t\in[0,T]} \tag{3}$$

The inverse problem

But the problem chemical engineers have to deal with is that in most cases, the isotherm function is not precisely known.

Moreover, values taken by such functions are almost impossible to obtain experimentally. So the only information available to identify the isotherm are experimental chromatograms with known experimental conditions. This inverse problem can be stated as:

$$(w(L,t)_{t\in[0,T]}, w^{init}(z)_{z\in[0,L]}, w^{inj}(t)_{t\in[0,T]}) \longrightarrow f \tag{4}$$

As pointed out in the introduction, this problem reduces to parametric optimization once a model is chosen for the isotherm. But the widely used analytical models of Langmuir (1916) or James and Valentin (1991) are quite limited: First, they do not apply to gas chromatography; second, the solution of the identification problem by standard gradient-like methods for these models involves posing and solving (both theoretically and numerically) the adjoint problem, imposing further restrictions on the domain of validity of the model, as done in Sepulveda (1994) and James and Sepulveda (1994). Last, the latter method does not scale up, as the number of parameters to identify is of order M^d, where d is the degree of the polynomial model.

The limitations of parametric models are addressed by considering more general models: The neural network models are investigated in the next section.

The recurrent neural net model

No analytic model is known as universal in the chromatographic community; no natural property of the isotherm function allows to restrict the search space to a subspace of the space of continuous functions. Moreover, neural nets are theoretically known to be *universal approximators* (White 1990) in the space of continuous functions, and have practically demonstrated good generalization properties. So neural nets are the model chosen for the isotherm function.

First investigations considered feed-forward neural nets, i.e. nets with no cycle in the graph of connections. This type of neural net

has demonstrated its utility and its robustness using both the back-propagation learning algorithm (Rumelhart et al. 1986, LeCun 1987) or evolutionary computation techniques (see Yao 1993 and the references herein, or Ronald and Schoenauer 1994).

But the first disappointing results of Harouni (1992) led to consideration of the recurrent neural net model, in which the graph of connections can contain cycles: As a superset of feed-forward nets, recurrent neural nets hopefully exhibit a larger class of behaviors. It is true that back-propagation does not apply to recurrent neural nets. But this is not a limitation for the problem at hand, as no direct observations of the isotherm values are available.

Recurrent neural nets can be used in two different ways: Either dynamically, to approximate time dependent phenomenon, in which case their dynamical behavior is the important feature to study (see Beer (1994) for one of the first theoretical study on the subject); or statically, their output being some equilibrium value reached after stabilization.

The isotherm function of the chromatographic process describes the amount of each component of the mixture which is absorbed by the solid porous medium of the column, given the concentrations of all other components: this value does not depend on the *history* of the process, and the static approach is clearly the one to choose here.

Finally, to handle the scale-up problem, both the weights *and the connectivity* of the neural nets should be identified. So only stochastic methods can possibly address the resulting optimization problem. Among stochastic methods one may apply the reinforcement learning approach, as in Barto et al. (1983), but this technique is based on some reinforcement signal for each value computed by the net, which is not available here. So an evolutionary algorithm, derived from the GNARL algorithm of Angeline (1993) is chosen and adapted to the problem of chromatography in next section.

3 METHOD

After briefly reviewing the GNARL algorithm, this section presents some original mutation operators, together with their strategy of use over generations.

The original GNARL algorithm

The GNARL algorithm is thoroughly described in Angeline (1993) and Angeline et al. (1994). Its main characteristics are briefly recalled for the sake of completeness. The aim of the algorithm is to determine both the structure and the weights of a recurrent neural net in order to optimize a given fitness function. It uses the evolutionary programming paradigm (Fogel et al. 1966, Fogel 1992), a top-down approach to evolutionary optimization. A population of neural nets evolves during a succession of generations. At each generation, the best half of the population (in term of fitness optimization) undergoes

a mutation operator whose severity is dynamically adjusted, depending of the relative fitness of the net at hand.

The position of GNARL in the evolutionary computation landscape will not be discussed here (see Angeline 1993).

GNARL algorithm involves two kinds of mutation operator. The parametric mutation modifies all the weights of the network, while the structural mutations modify the topology (connections) of the network.

Two specializations of GNARL weight mutation are implemented, and a new parametric mutation operator is introduced: the activation mutation. This latter operator was motivated by past experiments with feed-forward neural nets (Schoenauer and Ronald 1994, Ronald and Schoenauer 1994).

Parametric mutations

One-weight mutation

The actual GNARL weight mutation perturbs all weights by addition of a gaussian noise :

If w is a weight of some connection, its new value $\tilde{w}$ is defined by:

$$\begin{cases} \tilde{w} = w + N(0,T) \\ \text{with } T = U(0,1)(1 - \frac{f}{f_{max}}) \end{cases} \tag{5}$$

where $N(\mu, \sigma^2)$ is a gaussian variable with standard deviation σ,
$U(0,1)$ is a uniform random variable over $[0,1]$,
f is the fitness of the network at hand,
and f_{max} is the maximum fitness of the population.

In addition to this one-weight mutation, denoted W, two more local weight mutations are defined:
• the mutation of all input weights of a (randomly chosen) neuron, denoted W_{in};
• the mutation of all output weights of a (randomly chosen) neuron, denoted W_{out}.

These operators allow a network to adjust the neurons one by one. The same motivation led to the following activation mutation.

Mutation of the activation function

In each neuron, the activation function is the standard sigmoid function: $f_\beta(x) = \frac{1}{1+\exp(-2\beta x)}$; but the activation factor β depends on the neuron.

The activation mutation operator $\boldsymbol{\beta}$ parallels the one-weight mutation. The new value $\tilde{\beta}$ is defined by:

$$\begin{cases} \tilde{\beta} = \beta + N(0,T) \\ \text{with } T = U(0,1)(1 - \frac{f}{f_{max}}) \end{cases} \tag{6}$$

This operator results in multiplying all weights in input of a neuron by the same constant.

Structural mutations

The same structural mutations as in Angeline (1993) are used:
• Mutation of the connections (the mutation 'add one connection' is noted $C+$ and the mutation 'remove one connection' is noted $C-$);
• Mutation on the neurons (the mutation 'add one neuron' is noted $U+$ and the mutation 'remove one neuron' is noted $U-$).

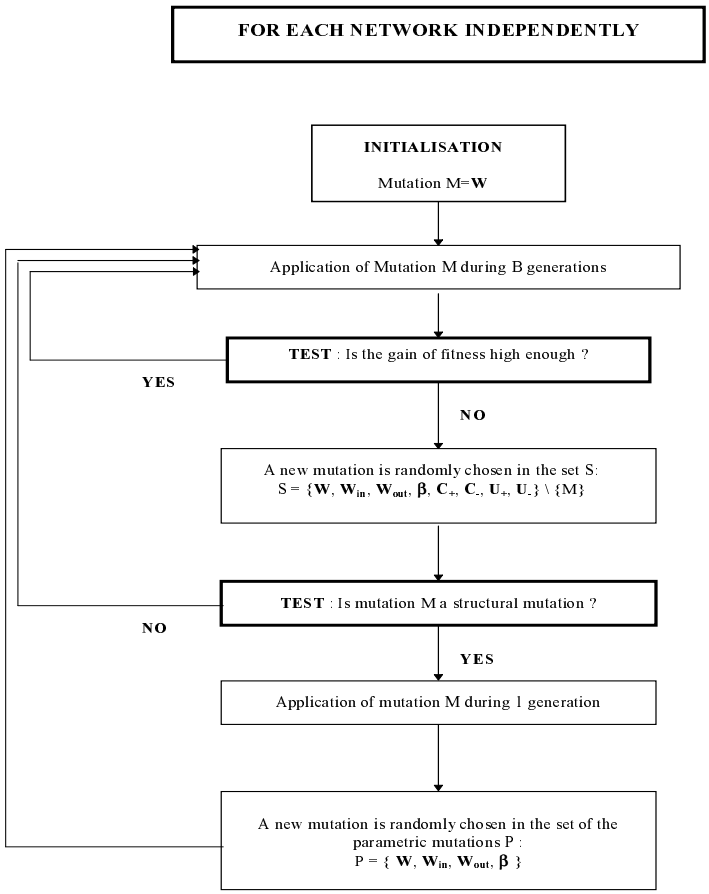


Figure 1: Flow chart of the mutation strategy.

The mutation strategy

Parametric mutations address weight optimization — depending on the structure of the neural net at hand. So it seems natural to allow several parametric mutations for one structural mutation. The standard strategy consists in setting a parametric mutation rate higher than the structural one. An alternative is to impose several parametric mutations after each structural one, thus giving a chance for new structures to adjust their weights before being changed again.

An outlook of the resulting strategy is given in Figure 1. Note that this strategy is applied for each network independently.

1. In the beginning, for each network a parametric mutation (generally W) is chosen.
2. The current parametric mutation is applied during a prescribed number B of generations, after which the increase of fitness is computed. If this increase is high enough, i.e., if :
$$Fitness(t) - Fitness(t-B) > \frac{Fitness(t)}{100}$$
then the current mutation remains the same, and the algorithm goes back to 2.
3. Otherwise, another mutation is randomly chosen according to some given probability.
4. If the new mutation is a structural mutation, it is applied only once before a parametric mutation is randomly chosen.
5. After the choice of a new parametric mutation, the algorithm returns to 2.

The next section presents the first results of this algorithm applied to the problem of the identification of the isotherm in chromatography described in section 2. The values of the different parameters of the algorithm are:
Population size = 25, $B = 10$,
$P_W = P_{\boldsymbol{\beta}} = P_{C+} + P_{C-} = 0.15$
and $P_{W_{in}} = P_{W_{out}} = P_{U+} = P_{U-} = 0.1$.

4 RESULTS

Given some observed chromatogram(s) $w^{exp}(t)_{t\in[0,T]}$, and the corresponding experimental conditions (w^{inj} and w^{init})[3] the goal is to identify the isotherm function f such that the solution of system (1) produces the prescribed chromatogram(s).

The neural networks considered here are restricted to recurrent neural nets with M input neurons and M output neurons. For each neural network, and each triple (w^{exp}, w^{init}, w^{inj}), the computation of w^{NET} is achieved using the Godunov scheme presented in (2) on the system of equations

$$\begin{cases} \frac{\partial w^{NET}}{\partial z} + \frac{\partial f(w^{NET})}{\partial t} = 0 \text{ for } (z,t) \in [0,L] \times [0,T] \\ w^{NET}(0,t) = w^{inj}(t) \text{ for } t \in [0,T] \\ w^{NET}(z,0) = w^{init}(z) \text{ for } z \in [0,L] \end{cases} \tag{7}$$

An important remark is that such a computation involves $M \times E \times S$ computations of the outputs of the network for some given inputs. One such computation is the stabilization of the network from a prescribed starting point (activation 0 for every neuron)[4] Therefore, an

important issue, investigated further on, is which experiments (w^{inj}, w^{init}, $w^{exp}(t)_{t\in[0,T]}$) should be considered in order to obtain satisfying results at a tractable computational cost.

For practical reasons, results of numerical solutions of system (1) with Langmuir isotherm Langmuir (1916) are used instead of actual experimental results. Thus two different types of plots are presented to illustrate each result. A first plot directly compares the estimated and the original isotherm. A second plot shows both the experimental chromatogram and that resulting from the identified network. In both cases, the experimental data are labeled "Langmuir" and the result of the identification "Network."

One component, one experiment

First results deal with a one-component mixture[5]. The experimental data consists in one experiment, namely w^{init} and w^{inj} together with the experimental chromatogram, $w^{exp}(t)_{t\in[0,T]}$.

The fitness of a network is given by:

$$Fitness = \sum_{c=1}^{M} \left\| w_c^{NET} - w_c^{exp} \right\|_2 \tag{8}$$

$$= \sum_{c=1}^{M} \left\{ \int_0^T \left| w_c^{NET}(L,t) - w_c^{exp}(t) \right|^2 dt \right\}^{1/2} \tag{9}$$

where w_c^{NET} is the cth component of the solution of system (7).

The experimental conditions are $w^{init} = 0.0047$ and $w^{inj} = 0.0012$, and the results are shown in figure 2 and 3. The optimum was reached in an average of 150 generations.

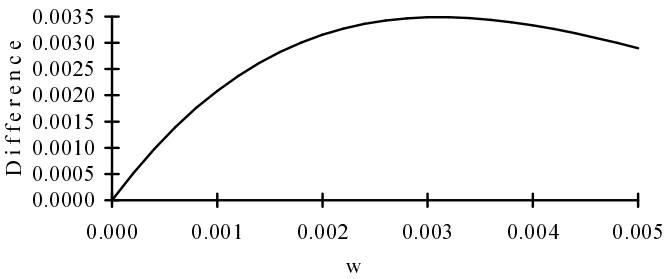


Figure 2. The absolute difference between the *Langmuir* isotherm, that was used to generate the *experimental* data, and the *Network* isotherm, the result of the evolutionary identification.

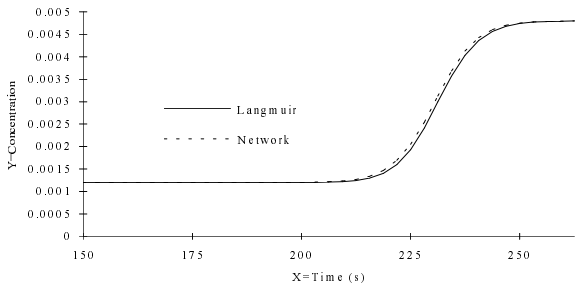


Figure 3. The two chromatograms. The *Langmuir* chromatogram is the *experimental data*. The *Network* chromatogram is the output computed using the identified network.

Of course, the results of the whole identification process should be independent from the training data, be it experimental (w^{init} and w^{inj}) or numerical (time and space step). No dependency on the numerical parameters is observed, but the results are sensitive to the experimental conditions:

The isotherm function determined above is used with other experimental conditions. The corresponding error is plotted in Figure 4 as a function of w^{init} and w^{inj} in the domain $[0, 0.005] \times (0, 0.005]$.

The net whose plot is shown in figure 3 was found using the values $w^{init} = 0.0047$ and $w^{inj} = 0.0012$. The error on the chromatogram was 2.7410^{-3}. Applying the same net to compute the chromatogram under different experimental conditions, namely in $[0, 0.005] \times (0, 0.005]$, and comparing this chromatogram to that of Langmuir isotherm leads to errors whose plot is given is Figure 4.

The error is 0 along the diagonal, due to the conservative form of system (1) and its numerical approximation (3). It takes the value 2.7410^{-3} for the experimental training conditions $(0.0047, 0.0012)$. But the sensitivity to the experimental conditions is witnessed by the amount of error far from the diagonal.

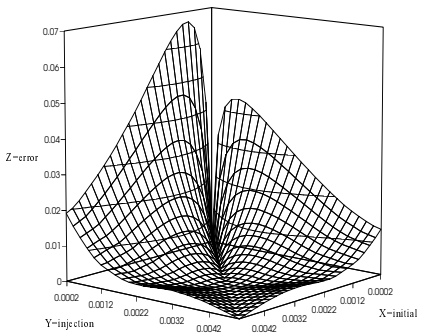


Figure 4. Error versus experimental conditions in domain $[0, 0.005] \times (0, 0.005]$. The network which is tested here is the solution of the evolutionary identification process with initial conditions $(0.0047, 0.0012)$. The plot represents the L_2 norm of the difference between the chromatogram computed using the *Langmuir* isotherm and the identified isotherm for other values of (w^{init}, w^{inj}).

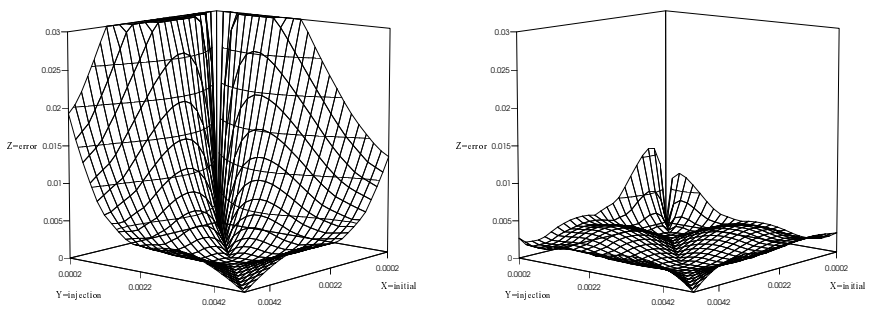
One component, multiple experiments

In order to avoid such a sensitivity, N different experimental conditions $(w_i^{init}, w_i^{inj})_{i\in[1,N]}$ together with the corresponding chromatograms $(w_i^{exp}(t)_{t\in[0,T]})_{i\in[1,N]}$ are used to compute the fitness. The initial pints are chosen on a fixed regular grid on $[0, 0.005] \times (0, 0.005]$.

$$Fitness = \frac{1}{N} \sum_{i=1}^{i=N} \sum_{c=1}^{M} \left\| w_{i,c}^{NET} - w_{i,c}^{exp} \right\|_2 \tag{10}$$

where $w_{i,c}^{NET}$ is the cth component of the solution of system (7) with experimental conditions (w_i^{init}, w_i^{inj}).

Figure 5 shows the plot of the error on the whole domain $[0, 0.005] \times (0, 0.005]$ for (w^{init}, w^{inj}) for different values of N. Note that $N = 1$ corresponds to Figure 4, though the scale has been adjusted to match with the other plots. Here again, the optimum was reached in an average of 150 generations, but the computational cost of one generation is proportional to N.



$N = 1$, $err_{max} = 0.06689$ $N = 4$, $err_{max} = 0.01106$

$N = 9$, $err_{max} = 0.00876$ $N = 25$, $err_{max} = 0.00929$

Figure 5. Error in the optimization domain for $N = 1, 4, 9, 25$. A net is evolved using N *experimental chromatograms* generated by N direct resolutions of the system (2) with (w^{init}, w^{inj}) on grid points of the $[0, 0.005] \times (0, 0.005]$ domain. The plots represent in each case the L_2 norm of the difference between the chromatogram computed using the *Langmuir* isotherm and the identified isotherm for other values of (w^{init}, w^{inj}).

These results show that increasing the number of experimental conditions taken into account improves the accuracy over the whole domain at first. But the error does not necessarily decrease when N increases. So one has to find the minimal N ensuring the optimal accuracy ($N = 9$ here). Figure 6 and 7 show the results (isotherm and chromatogram) of a network optimized with $N = 9$.

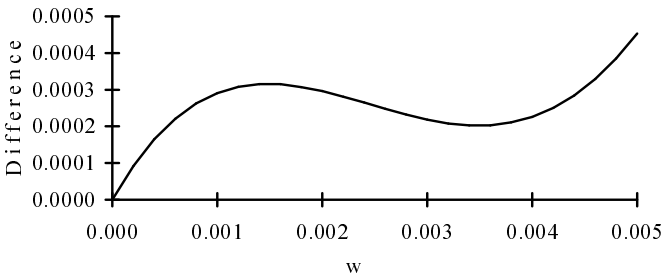


Figure 6. The absolute difference between the *Langmuir* isotherm, that was used to generate the *experimental* data, and the *Network* isotherm, the result of the evolutionary identification using 9 experimental chromatograms in the fitness function.

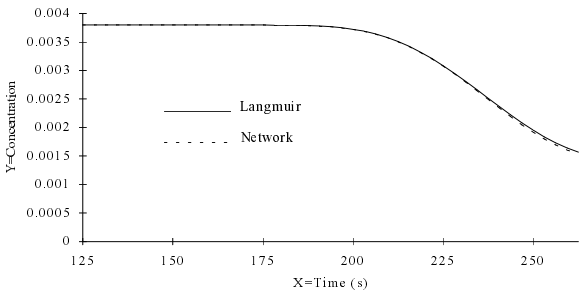


Figure 7. One chromatogram for $N = 9$. One chromatogram computed using the best identified isotherm (obtained using 9 *experimental* chromatograms), is shown together with the corresponding *experimental* chromatogram of the *Langmuir* isotherm.

Notice that the results presented in Figure 6 use the same experimental conditions ($w^{init} = 0.0047$ and $w^{inj} = 0.0012$) than the results of Figure 2, and show that the error on the isotherm has been lowered by about an order of magnitude. Increasing the number of experimental points does not only decrease the worse error in the optimization domain, it also increases the accuracy of the results on the learning points.

Last, the plots of Figure 5 show that the largest errors are due to very low values of both the injection and the initial state. This suggests that using the renormalization technique introduced by Ronald and Schoenauer (1994) might be beneficial.

Two component mixture

First results in the two component case have been obtained with a fitness function involving a single experiment, as defined by equations (8) and (9). Figure 8 shows the chromatograms for both components corresponding to the original and the estimated isotherm: both are nicely fitted, in around 300 generations of evolution.

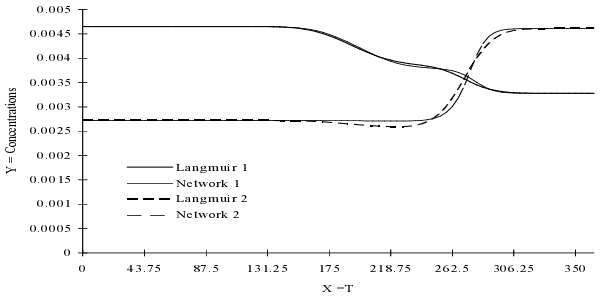


Figure 8. Chromatograms for two components. An example of the two chromatograms — one for each component of the mixture — obtained with the result of the evolutionary identification together with the corresponding *experimental* chromatograms of the *Langmuir* isotherm.

Nevertheless, the robustness issue for the two component case must be carefully studied. First, the numerical solution of system (1) seems to depend on the numerical parameters (i.e. coarseness of the discretization of the domain $[0, L] \times [0, T]$). Second, the choice of grid points to use the multiple experiments fitness defined by (10) becomes intractable as the number of components increases.

Discussion

These first satisfactory results demonstrate the feasibility of the proposed approach.

This evolutionary identification can be compared to "classical" identification based on gradient-like methods, as used in Sepulveda (1994) and James and Sepulveda (1994), since the training experimental conditions are computed from the analytical Langmuir model. In the one-component case, classical identification reduces to the optimization of two parameters. In this simple case, classical optimization strongly depends on the initialization point, and it fails to converge to the global optimum in many cases. But when it converges, it is faster than the evolutionary optimization by about two orders of magnitude.

Two additional points have to be raised: First, analytical models have very limited domains of validity, and the gradient-like method does not scale up; second, there is no standard way to take into account multiple experiments in the cited deterministic method. This greatly limits the robustness of its results.

In opposition, the method presented here is quite promising with respect to the scaling up. Regarding the one experiment case, the two-component problem is comparable in terms of computational cost to the one-component problem.

The robustness issue, however, must be thoroughly taken into account. If the experiments used for training follow a regular grid of the target validity domain, then it is exponential with respect to the number of components. As suggested by the error plots of Figure 4 and 5 in section 4, a more clever distribution of the experiments might do the job at a tractable cost.

5 SUMMARY AND FUTURE WORK

In this paper, the identification of the isotherm function is addressed by evolutionary optimization. Evolutionary optimization is used to discover both the topology and the weights of a recurrent neural network modeling the isotherm function. The isotherm identification is an inverse problem. The direct problem, solved by standard numerical techniques, gives means to estimate the quality of a candidate isotherm function. The fitness function used by the evolutionary algorithm derives from this estimation.

The evolutionary algorithm used in this paper is inspired by the GNARL algorithm of Angeline (1993). New mutation operators are implemented, and an evolutionary strategy of application of the different mutation operators, based on the recent evolution of the fitness of the network is implemented.

The first results demonstrate the feasibility of the approach. But when an analytical model reduces the identification problem to some parametric optimization, standard deterministic methods must be preferred.

Further research will investigate how to overcome the sensitivity of the results on the experimental conditions. Two directions will be considered: As shown by both theoretical and experimental work on the back-propagation algorithm (see Matsuoka 1992, and Grandvalet and Canu 1995), incorporating noise during the learning phase of a neural net increases the robustness of the resulting net; Second, evolution could simultaneously consider two populations, the neural nets modeling the isotherm, and the experiments used to train the nets. In such a co-evolutionary framework, inspired by Hillis (1990), the fitness of an experimental conditions would be the corresponding error committed by the current population of neural nets.

Another challenging issue deals with the addition of knowledge to speed-up evolution. The new strategy for applying the mutation operators presented in this paper addresses the evolutionary part of that issue. Two further steps in that direction are currently investigated: As in evolution strategies and evolutionary programming, the standard deviation for the mutation of a neuron can be evolved within the neuron itself (see Baeck et al. 1992; Fogel et al. 1992); or, as demonstrated efficient for genetic algorithms in Sebag and Schoenauer (1994), some knowledge about what mutation to perform, or what neuron to mutate, can be provided by inductive learning and observations about current evolution. The new mutation operators introduced herein should preferably apply on neurons known to perform poorly, if such knowledge is available. Being able to learn and

use such knowledge may indeed lead to a break-through in evolutionary optimization of neural nets.

Notes

1 A component can be made of various elements, which are chemically different but have similar behaviors in a chromatographic column.
2 The motionless medium can also be liquid, the underlying phenomenon is then termed *adsorption*.
3 Though w^{inj} and w^{init} are functions of time and space respectively, they were set to constants in each experiment
4 If a network does not stabilize after a prescribed number of iterations, its fitness is set to 0 and it disappears.
5 The model presented in section 2 is a simplified model: even in the one-component case, some interaction between the only component and the fluid used to force percolation produces nonlinear effects.

Acknowledgments

The authors wish to thank Michèle Sebag, of LMS at Ecole Polytechnique, for helpful discussions and careful proof-reading of the paper.

References

Angeline P. J. (1993). Evolutionary Algorithms and Emergent Intelligence. PhD thesis, Ohio State University, Colombus.

Angeline P. J., G. M. Saunders and J. B. Pollack (1994). An Evolutionary Algorithm that Constructs Recurrent Neural Networks. *IEEE Transactions on Neural Networks*. XXX

Baeck T., G. Rudolph and H.-P. Schwefel (1992). Evolutionary Programming and Evolution Strategies: Similarities and Differences. *Proceedings of the Second Annual Conference on Evolutionary Programming*, ed. D. B. Fogel and W. Atmar, La Jolla, CA: Evolutionary Programming Society, 11-22.

Barto A. G., R. S. Sutton and C. W. Anderson (1983). Neuron like elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man, and Cybernetics* **13**: 835-846.

Beer R. (1994). On the Dynamics of a Continuous Hopfield Neuron with Self-Connection. Tech. Report CES-94-1, Case Western Reserve University, Cleveland.

Bevington P. R. (1969). *Data reduction and Error Analysis for the Physical Sciences*, New-York, NY: McGraw-Hill.

Grandvalet Y. and S. Canu (1995). Comments on "Noise injection into inputs in back propagation learning" by Matsuoka 1992. To appear in *IEEE transactions on Systems, Man and Cybernetics*.

Fogel L. J., A. J Owens and M. J. Walsh (1966). *Artificial Intelligence through Simulated Evolution*, New York, NY: John Wiley.

Fogel D. B., An Analysis of Evolutionary Programming, *Proceedings of the First Annual Conference on Evolutionary Programming*, ed. D. B. Fogel and W. Atmar, La Jolla, CA: Evolutionary Programming Society, 43-51.

Fogel D. B., L. J. Fogel, W. Atmar and G. B. Fogel (1992). Hierarchic Methods of Evolutionary Programming. *Proceedings of the First Annual Conference on Evolutionary Programming*, ed. L. J. Fogel and W. Atmar, La Jolla, CA: Evolutionary Programming Society, 175-182.

Godunov S. K. (1959). A Finite Difference Method for Numerical Computation of Discontinuous Solutions of the Equations of Flow Dynamics. *Math USSR Sb.*, **47**: 271-290.

Harouni P. (1992). Approximation de fonctions réelles par réseaux neuro-génétiques. Msc thesis, Université Pierre et Marie Curie, Paris.

Hillis D. (1990). Co-evolving parasites improve simulated evolution as an optimization procedure. *Physica D* **42**: 228-234.

James F. (1990). Sur la modélisation mathématiques des équilibres diphasiques et des colonnes de chromatographie. PhD thesis, École Polytechnique.

James F. and M. Sepulveda (1994). Parameter identification for a hyperbolic system arising in chemical engineering. *Inverse Problems* **10** (6): 1299-1314.

James F. and P. Valentin (1991). An analytical model of multicomponent isothermal adsorption equilibrium, Adsorption Processes for Gas Separation, CNRS-NSF Gif 91, ed. F. Meunier and M.D. Levan, *Récents progrès en Génie des Procédés*, **5** (17): 49-56.

Koza J. (1994). *Genetic Programming, II*, Cambridge, MA: MIT Press.

Langmuir I. (1916). *Jour. Am. Chem. Soc.*, **38**: 2221.

Le Cun Y. (1987). *Modèles connexionnistes de l'apprentissage*. PhD thesis, Université Pierre et Marie Curie, Paris.

Matsuoka K. (1992). Noise injection into inputs in back propagation learning. *IEEE transactions on Systems, Man and Cybernetics*, **22** (3): 436-440.

Polak E. (1971). *Computational Methods in Optimization*, New York, NY: Academic Press.

Ronald E. and M. Schoenauer (1994). Genetic lunar lander : an experiment in accurate neuro-control. *Proceedings of the* 3^{rd} *Conference on Parallel Problems Solving from Nature*, ed. H.-P. Schwefel, Y. Davidor and R. Männer, Heidelberg: Springer Verlag, 452-461.

Rouchon P., M. Schoenauer, P. Valentin and G. Guiochon (1989). Numerical Simulation of Band Propagation in Nonlinear Chromatography. *Preparative-Scale Chromatography*, ed. E. Grushka, New York, NY: Marcel Dekker Inc., 1-42.

Rumelhart D. E. and J. L. McCelland (1986). *Parallel Distributed Processing : Exploration in the Microstructure of Cognition*, I & II, Cambridge, MA: MIT Press.

Schoenauer M. and E. Ronald (1994). Neuro-genetic truck backer-upper controller. *Proceedings of WCCI 94, IEEE Conference on Evolutionary Computation*, Orlando, FL.

Sebag M. and M. Schoenauer (1994). Controlling Crossover through Inductive Learning, *Proceedings of the* 3^{rd} *Conference on Parallel Problems Solving from Nature*, ed. H.-P. Schwefel, Y. Davidor and R. Männer, Heidelberg: Springer Verlag, 209-218.

Sepulveda M. (1993). Identification de paramètres pour un système hyperbolique. Application a l'estimation des isothermes en chromatographie, PhD thesis, École Polytechnique, Palaiseau.

Valentin P. and G. Guiochon (1976). Propagation of Finite Concentration in Gas Chromatography, *Separation Science*, **10**: 245-305.

Watkins C.J. and P. Dayan (1992). Q-learning, *Machine Learning*, **8**: 279-292.

White H. (1990). Connectionist Nonparametric Regression: Multilayer Feedforward Networks Can Learn Arbitrary Mappings. *Neural Networks*, **3**: 535-549.

Yao X. (1993). A Review of Evolutionary Artificial Neural Networks. *Int. Jour. of Intelligent Systems*, **8**: 539-567.