AIP Conference Proceedings

Representation of neural networks as Lotka-Volterra systems

Cite as: AIP Conference Proceedings **465**, 155 (1999); https://doi.org/10.1063/1.58279
Published Online: 27 March 2008

Yves Moreau, Stéphane Louiès, Joos Vandewalle, et al.

View Online Export Citation

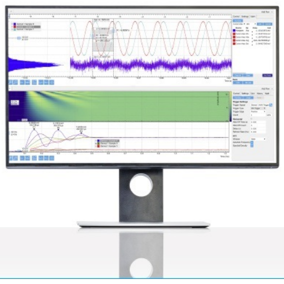
ARTICLES YOU MAY BE INTERESTED IN

Solving ordinary differential equations using neural networks
AIP Conference Proceedings **1974**, 020070 (2018); https://doi.org/10.1063/1.5041601

Implementation of the dynamical system of the deposit and loan growth based on the Lotka-Volterra model and the improved model
AIP Conference Proceedings **1723**, 030007 (2016); https://doi.org/10.1063/1.4945065

Numerical simulation of stochastic model for deposit and loan volume based on the Lotka - Volterra model
AIP Conference Proceedings **2023**, 020216 (2018); https://doi.org/10.1063/1.5064213

Challenge us.
What are your needs for periodic signal detection?
Watch
Zurich Instruments



© 1999 American Institute of Physics.

Representation of neural networks as Lotka–Volterra systems

Yves Moreau[1], Stéphane Louiès[2], Joos Vandewalle[1], Léon Brenig[2]

[1]*Katholieke Universiteit Leuven, Department of Electrical Engineering*
Kardinaal Mercierlaan 94, B-3001 Leuven, Belgium
{yves.moreau,joos.vandewalle}@esat.kuleuven.ac.be
[2]*Université Libre de Bruxelles,*
Boulevard du Triomphe, B-1050 Bruxelles, Belgium
{slouies,lbrenig}@ulb.ac.be

Abstract. We study changes of coordinates that allow the representation of the ordinary differential equations describing continuous-time recurrent neural networks into differential equations describing predator-prey models —also called Lotka-Volterra systems. We transform the equations for the neural network first into quasi-monomial form [1], where we express the vector field of the dynamical system as a linear combination of products of powers of the variables. In practice, this transformation is possible only if the activation function is the hyperbolic tangent or the logistic sigmoïd. From this quasi-monomial form, we can directly transform the system further into Lotka-Volterra equations. The resulting Lotka-Volterra system is of higher dimension than the original system, but the behavior of its first variables is equivalent to the behavior of the original neural network.

Keywords—Continuous-time neural networks, Equivalence of dynamical systems, Lotka-Volterra systems, Predator-prey models, Quasi-monomial forms.

LOTKA–VOLTERRA SYSTEMS

We show here how to represent different types of neural networks (having the hyperbolic tangent or the logistic sigmoid as activation function) into Lotka–Volterra systems (also called predator-prey models) of the form:

$$\dot{z}_i = \lambda_i z_i + z_i \sum_{j=1}^{m} M_{ij} z_j, \qquad \text{with } j = 1, \dots, m.$$

The corresponding Lotka–Volterra system will be of higher dimension than the original system but its n first variables have a behavior equivalent or identical to that of the original system (if n is the number of variables of the original system).

Such a Lotka–Volterra representation of neural networks is of interest for several reasons. Lotka–Volterra systems are an important tool in mathematical biology for the modeling of competition between species and a classical subject of dynamical system theory [2,3]. They have therefore been the object of intense scrutiny [4]. These systems have simple quadratic nonlinearities (not all quadratic terms are possible in each equation), which might be easier to analyze than the sigmoidal saturations of the neural networks. We thus expect that this representation will allow us to apply methods and results from the study of Lotka–Volterra systems to neural networks. We think, in particular, of the application of conditions for the global asymptotic stability of an equilibrium developed for quasi-monomial systems and Lotka–Volterra systems [5,6] to neural networks. Also, early work by Grossberg exploited one of the changes of variables that we present here to study convergence and oscillations in generalized competitive networks and generalized Lotka–Volterra systems [7]. Our result also serves as a simple proof that Lotka–Volterra systems enjoy the same properties of approximation of dynamical systems as dynamical neural networks; and we might want to investigate them further from that point of view.

We have structured this work as follows. We consider first the simple case of a dynamical perceptron. We transform it into what we call a quasi-monomial form. This form allows us to transform the system further into a Lotka–Volterra system. Second, we consider the case of the dynamical perceptron with self-connections—that is, where each neuron has a linear connection to itself. We perform the same transformations first into a quasi-monomial form and then into a Lotka–Volterra system; we also discuss how to treat possible singularities of these transformations. Third, we show that the same transformations apply to much more general types of continuous-time recurrent neural networks with hyperbolic tangent or logistic activation functions. In the case of the logistic activation function, we see that the transformation does not present any singularity. Finally, we clarify what the exact constraints on the activation function are and we show that both the hyperbolic tangent and the logistic sigmoid satisfy these constraints.

DYNAMICAL PERCEPTRONS

We consider first n-dimensional dynamical perceptrons:

$$\dot{x} = \sigma(Ax + a_0), \tag{1}$$

where $x \in \mathbb{R}^n$, $A \in \mathbb{R}^{n\times n}$, $a_0 \in \mathbb{R}^n$, and σ is a continuous sigmoidal function. We consider this type of system because its simplicity makes our calculations easier to follow, yet it permits to illustrate the essential points of our method. We do all developments for the special case of $\sigma(x) = \tanh(x)$. Another possibility is the use of the logistic sigmoid $\sigma(x) = 1/(1 + e^{-x})$. We discuss in a later section what constrains the choice of the activation function. For the dynamical perceptron (1), choosing the logistic sigmoid would be meaningless since the vector field would then have positive components everywhere and the topology of the flow would be trivial (whereas the hyperbolic tangent guarantees the existence of an equilibrium). For the types of networks we describe in the following sections, this restriction to the hyperbolic tangent does not apply. But since the developments depend on the choice of the activation function, we limit ourselves to the hyperbolic tangent; all developments can be done similarly for the logistic sigmoid.

Sigmoidal Change of Coordinates

We first apply the change of coordinates $y = Ax + a_0$ (often called Sigma–Pi transformation [8]) and get the following equation:

$$\dot{y} = A\sigma(y).$$

As long as A is invertible (A is $n \times n$), we can directly revert the change of coordinates. Moreover, if A is singular, we in fact have an s-dimensional dynamics, where s is the rank of the matrix A. Indeed, if we look at two different trajectories $\bar{x}(t)$ and $\tilde{x}(t)$ of the system $\dot{x} = \sigma(Ax + a_0)$ having the property that their initial conditions $\bar{x}_0$ and $\tilde{x}_0$ are such that $A\bar{x}_0 = A\tilde{x}_0$, the initial conditions $\bar{y}_0 = A\bar{x}_0 + a_0$ and $\tilde{y}_0 = A\tilde{x}_0 + a_0$ for the system $\dot{y} = A\sigma(y)$ are equal. This means that $\bar{y}(t) = \tilde{y}(t)$ for all times. But the vector field of the initial system is $\dot{x} = \sigma(Ax + a_0) = \sigma(y)$, thus $\dot{\bar{x}}(t) = \dot{\tilde{x}}(t)$ for all times. This means that the trajectories $\bar{x}(t)$ and $\tilde{x}(t)$ through $\bar{x}_0$ and $\tilde{x}_0$ always remain parallel. Since the kernel of A is of dimension $n - s$, this means that the dynamics is in fact s-dimensional. If we define $\gamma(x_0) = x_0 - A^\dagger A x_0$ where $A^\dagger$ is the Moore–Penrose pseudoinverse of A, then $x(t) = A^\dagger(y(t) - a_0) + \gamma(x_0)$ with the initial condition $y_0 = Ax_0 + a_0$.

From here on, it is necessary to look at the equations coordinate by coordinate, so we have

$$\dot{y}_i = \sum_{j=1}^{n} A_{ij}\sigma(y_j), \quad \text{with } i = 1, \dots, n. \tag{2}$$

Now, we decide to apply the change of coordinates $z_i = \sigma(y_i)$; we get

$$\dot{z}_i = \sigma'(y_i)\dot{y}_i. \tag{3}$$

It is here that we make use of our special choice of activation function, $\sigma(y) = \tanh(y)$. The essential property of the activation function we need is that we can express its derivative as a polynomial of the activation function:

$$\sigma'(y) = \tanh'(y) = 1/\cosh^2(y) = 1 - \tanh^2(y) = 1 - \sigma^2(y). \tag{4}$$

Substituting (4) and (2) in Equation 3, we have

$$\dot{z}_i = (1 - z_i^2) \sum_{j=1}^{n} A_{ij} z_j. \tag{5}$$

We see that this form begins to resemble the Lotka–Volterra model except for the factor $(1 - z_i^2)$ instead of z_i. This means that the system is polynomial of degree three. However, the vector field of the system has now become a linear combination of products of powers of the variables, which is what we call a quasi-monomial form.

Example

We follow our developments at the hand of an example to clarify the procedure. We look at the case $n = 3$. The equations of the recurrent neural network are

$$\begin{cases} \dot{x}_1 &= \tanh(A_{11}x_1 + A_{12}x_2 + A_{13}x_3 + a_{01}), \\ \dot{x}_2 &= \tanh(A_{21}x_1 + A_{22}x_2 + A_{23}x_3 + a_{02}), \\ \dot{x}_3 &= \tanh(A_{31}x_1 + A_{32}x_2 + A_{33}x_3 + a_{03}). \end{cases}$$

We can check that the equations become, after the affine and sigmoidal changes of coordinates,

$$\begin{cases} \dot{z}_1 &= (1 - z_1^2)(A_{11}z_1 + A_{12}z_2 + A_{13}z_3), \\ \dot{z}_2 &= (1 - z_2^2)(A_{21}z_1 + A_{22}z_2 + A_{23}z_3), \\ \dot{z}_3 &= (1 - z_3^2)(A_{31}z_1 + A_{32}z_2 + A_{33}z_3). \end{cases} \tag{6}$$

Transformation into Quasi-Monomial Forms

By quasi-monomial form, we mean a dynamical system for which we can express the component of the vector field for each variable as the product of this variable by a linear combination of products of powers of all variables.

Definition 1. *A* quasi-monomial form *is a dynamical system of the form*

$$\dot{z}_i = z_i\Big(\sum_{j=1}^{r} \bar{A}_{ij} \prod_{k=1}^{n} z_k^{B_{jk}}\Big), \quad \textit{with } i = 1, \ldots, n. \tag{7}$$

The number of different products of powers of the variables is r and we define $\bar{A} = [\bar{A}_{ij}] \in \mathbb{R}^{n \times r}$ and $B = [B_{jk}] \in \mathbb{R}^{r \times n}$. It is important to note that the exponents B_{jk} can be real values (they are not restricted to natural numbers), hence the name of quasi-monomial form. Now, to reach this form, we write Equation 5 as

$$\dot{z}_i = z_i \left[\sum_{j=1}^{n} A_{ij}(z_i^{-1} z_j - z_i z_j) \right]. \tag{8}$$

We then need to collect all possible monomials and use their exponents to construct the matrix B and collect all the coefficients of the monomials correspondingly to form the matrix $\bar{A}$. We illustrate it first on the following example and then do it systematically in the next section. Once we have the quasi-monomial form, Brenig has shown how to transform such a quasi-monomial form into a Lotka–Volterra system [1]. Other authors [9,5] have discovered similar transformations independently.

Example (cont'd)

To illustrate this somewhat obtuse notation (7), we take again the case $n = 3$. Expanding the expression (6), we get

$$\begin{cases} \dot{z}_1 &= z_1(A_{11} - A_{11}z_1^2 + A_{12}z_1^{-1}z_2 - A_{12}z_1z_2 + A_{13}z_1^{-1}z_3 + A_{13}z_1z_3) \\ \dot{z}_2 &= z_2(A_{21}z_1z_2^{-1} - A_{21}z_1z_2 + A_{22} - A_{22}z_2^2 + A_{23}z_2^{-1}z_3 - A_{23}z_2z_3) \\ \dot{z}_3 &= z_3(A_{31}z_1z_3^{-1} - A_{31}z_1z_3 + A_{32}z_2z_3^{-1} - A_{32}z_2z_3 + A_{33} - A_{33}z_3^2) \end{cases}$$

We see that there are $2n^2 = 18$ terms, but only $r = (3n^2 - n + 2)/2 = 13$ different monomials. The matrix B is then

$$B = \begin{pmatrix} 2 & 1 & 1 & 0 & 0 & 0 & -1 & -1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 2 & 1 & 0 & 1 & 0 & -1 & -1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 2 & 0 & 1 & 0 & 1 & -1 & -1 & 0 \end{pmatrix}^T ;$$

while the matrix $\bar{A}$ is

$$\bar{A} = \begin{pmatrix} -A_{11} & -A_{12} & -A_{13} & 0 & 0 & 0 & A_{12} & A_{13} & 0 & 0 & 0 & 0 & A_{11} \\ 0 & -A_{21} & 0 & -A_{22} & -A_{23} & 0 & 0 & 0 & A_{21} & A_{23} & 0 & 0 & A_{22} \\ 0 & 0 & -A_{31} & 0 & -A_{32} & -A_{33} & 0 & 0 & 0 & 0 & A_{31} & A_{32} & A_{33} \end{pmatrix} .$$

How should we interpret these matrices? If we look at the first row of B, we have a row vector $(2, 0, 0)$ (note that the notation above uses a transposition); and if we look at the first column of $\bar{A}$, we have a column vector $(-A_{11}, 0, 0)^T$. These vectors mean that the expression for the system contains a monomial of the form $z_1^2z_2^0z_3^0 = z_1^2$; and that it appears in the first equation with a coefficient of $-A_{11}$ and is absent from the second and third equations. Checking this mechanism for the whole matrices, we indeed have $\dot{z}_i = z_i\left(\sum_{j=1}^{13} \bar{A}_{ij} \prod_{k=1}^{3} z_k^{B_{jk}}\right)$ with $i = 1, 2, 3$.

Computation of the Matrices of the Quasi-Monomial Forms

We derive here the structure of the matrices $\bar{A}$ and B of the quasi-monomial form for the dynamical perceptron in a systematic fashion. Again, we write the equations as

$$\dot{z}_i = z_i \left[\sum_{j=1}^{n} A_{ij}(z_i^{-1}z_j - z_iz_j) \right] . \tag{9}$$

We see that the expression between brackets contains $2n$ monomials; for the n differential equations this gives a total of $2n^2$ terms. There are two types of monomials: z_iz_j and $z_i^{-1}z_j$. The first type of monomial can take two different forms: z_i^2 (when $i = j$) or z_iz_j with $j > i$ (when $i \neq j$). There are n monomials of the first form and $n(n-1)/2$ monomials of the second form. The second type of monomials can also take two different forms: 1 (when $i = j$) or $z_i^{-1}z_j$ (when $i \neq j$). There is one monomial of the first form and $n(n-1)$ monomials of the second form. These different contributions give a total of $(3n^2 - n + 2)/2$ different monomials to share among $2n^2$ terms. Carefully collecting all these terms into row vectors of exponents to form the matrix B and all their corresponding coefficients into column vectors to form the matrix $\bar{A}$, we can write these matrices as

$$B = \left[C^{(1)}; C^{(2)}; \ldots ; C^{(n)}; D^{(1)}; D^{(2)}; \ldots D^{(n)}; O^{1 \times n}\right], \tag{10}$$

$$\bar{A} = \left[P^{(1)}P^{(2)} \ldots P^{(n)}R^{(1)}R^{(2)} \ldots R^{(n)}V\right], \tag{11}$$

(recall that we use the semicolon to stack up submatrices) where we define the submatrices $C^{(i)}, D^{(i)}, O^{1 \times n}$, and $P^{(i)}, R^{(i)}, V$ as follows. The matrix $O^{n \times m}$ is a matrix of zeros of dimension $n \times m$; later, we will also use $I^{n \times m}$ to denote a diagonal matrix of ones of dimension $n \times m$. The matrix $C^{(i)}$ represents all the monomials of the form z_i^2 and z_iz_j (with $j > i$) and thus the dimension of the matrix depends on the index i with the matrix having $n - i + 1$ rows and n columns. We can write it as

$$C^{(i)} = \left(\begin{array}{ccc|c|ccc} 0 & \cdots & 0 & 2 & 0 & \cdots & 0 \\ \hline & & & 1 & & & \\ & O^{(n-i)\times(i-1)} & & 1 & & I^{(n-i)\times(n-i)} & \\ & & & \vdots & & & \\ & & & 1 & & & \end{array}\right).$$

Each of its rows is associated with the corresponding column of the matrix $P^{(i)}$, which collects all the coefficients of the monomials z_i^2 and $z_i z_j$ (with $j > i$). Such a matrix has again variable dimension with n rows and $n-i+1$ columns. We can write it as

$$P^{(i)} = \left(\begin{array}{ccccc} \multicolumn{5}{c}{O^{(i-1)\times(n-i+1)}} \\ \hline -A_{ii} & -A_{i,(i+1)} & -A_{i,(i+2)} & \cdots & -A_{in} \\ 0 & -A_{(i+1),i} & 0 & \cdots & 0 \\ \vdots & 0 & -A_{(i+2),i} & & \vdots \\ \vdots & \vdots & & \ddots & \\ 0 & 0 & 0 & \cdots & -A_{ni} \end{array}\right).$$

Turning back to the matrix of exponents B, we collect all the exponents of the monomials of the form $z_i^{-1} z_j$ (with $j \neq i$) in the matrix $D^{(i)}$ which has always $n-1$ rows and n columns:

$$D^{(i)} = \left(\begin{array}{c|c|c} I^{(i-1)\times(i-1)} & \begin{matrix} -1 \\ \vdots \end{matrix} & O^{(i-1)\times(n-i)} \\ \hline O^{(n-i)\times(i-1)} & \begin{matrix} \vdots \\ -1 \end{matrix} & I^{(n-i)\times(n-i)} \end{array}\right).$$

We collect the coefficients of these monomials in the corresponding columns of $R^{(i)}$, which also has $n-1$ rows and n columns:

$$R^{(i)} = \left(\begin{array}{cccccc} \multicolumn{6}{c}{O^{(i-1)\times(n-1)}} \\ \hline A_{i1} & \cdots & A_{i,(i-1)} & A_{i,(i+1)} & \cdots & A_{in} \\ \hline \multicolumn{6}{c}{O^{(n-i)\times(n-1)}} \end{array}\right).$$

We are now only left with the constant monomial (which we represent by the row vector $O^{1\times n}$) and its coefficients, which we collect in the column vector

$$V = (A_{11}, \ldots, A_{nn})^T.$$

Taking all this notation into consideration, we can represent the matrices A and B according to Equations 10 and 11.

Example

We now present the different matrices for the case of a four-dimensional system. This example clarifies how we have collected the different terms in the matrices $\bar{A}$ and B. The submatrices are $C^{(1)}, C^{(2)}, C^{(3)}, C^{(4)}$ and $D^{(1)}, D^{(2)}, D^{(3)}, D^{(4)}$ for B:

$$C^{(1)} = \left(\begin{array}{c|ccc} 2 & 0 & 0 & 0 \\ \hline 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{array}\right), \quad C^{(2)} = \left(\begin{array}{c|c|cc} 0 & 2 & 0 & 0 \\ \hline 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{array}\right),$$

$$C^{(3)} = \left(\begin{array}{cc|c|c} 0 & 0 & 2 & 0 \\ \hline 0 & 0 & 1 & 1 \end{array}\right), \quad C^{(4)} = \left(\begin{array}{ccc|c} 0 & 0 & 0 & 2 \end{array}\right);$$

$$D^{(1)} = \left(\begin{array}{c|ccc} -1 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ -1 & 0 & 0 & 1 \end{array}\right), \quad D^{(2)} = \left(\begin{array}{c|c|cc} 1 & -1 & 0 & 0 \\ \hline 0 & -1 & 1 & 0 \\ 0 & -1 & 0 & 1 \end{array}\right),$$

$$D^{(3)} = \left(\begin{array}{cc|c|c} 1 & 0 & -1 & 0 \\ 0 & 1 & -1 & 0 \\ \hline 0 & 0 & -1 & 1 \end{array}\right), \quad D^{(4)} = \left(\begin{array}{ccc|c} 1 & 0 & 0 & -1 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & -1 \end{array}\right).$$

We have decomposed the matrix A into the submatrices $P^{(1)}, P^{(2)}, P^{(3)}, P^{(4)}$, and $R^{(1)}, R^{(2)}, R^{(3)}, R^{(4)}$ and the vector V:

$$P^{(1)} = \left(\begin{array}{c|ccc} -A_{11} & -A_{12} & -A_{13} & -A_{14} \\ \hline 0 & -A_{21} & 0 & 0 \\ 0 & 0 & -A_{31} & 0 \\ 0 & 0 & 0 & -A_{41} \end{array}\right), \quad P^{(3)} = \left(\begin{array}{c|c} 0 & 0 \\ 0 & 0 \\ -A_{33} & -A_{34} \\ \hline 0 & -A_{43} \end{array}\right),$$

$$P^{(2)} = \left(\begin{array}{c|cc} 0 & 0 & 0 \\ -A_{22} & -A_{23} & -A_{24} \\ \hline 0 & -A_{32} & 0 \\ 0 & 0 & -A_{42} \end{array}\right), \quad P^{(4)} = \left(\begin{array}{c} 0 \\ 0 \\ 0 \\ -A_{44} \end{array}\right);$$

$$R^{(1)} = \left(\begin{array}{ccc} A_{12} & A_{13} & A_{14} \\ \hline 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{array}\right), \quad R^{(2)} = \left(\begin{array}{ccc} 0 & 0 & 0 \\ \hline A_{21} & A_{23} & A_{24} \\ \hline 0 & 0 & 0 \\ 0 & 0 & 0 \end{array}\right),$$

$$R^{(3)} = \left(\begin{array}{ccc} 0 & 0 & 0 \\ 0 & 0 & 0 \\ \hline A_{31} & A_{32} & A_{34} \\ \hline 0 & 0 & 0 \end{array}\right), \quad R^{(4)} = \left(\begin{array}{ccc} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ \hline A_{41} & A_{42} & A_{43} \end{array}\right);$$

$$V = (A_{11}, A_{22}, A_{33}, A_{44})^T.$$

TRANSFORMATION OF THE QUASI-MONOMIAL FORMS INTO LOTKA-VOLTERRA SYSTEMS

From this quasi-monomial form, we can get a Lotka-Volterra system by a single change of coordinates. The whole transformation relies now on a trick: We look at each product of powers of the variables that appear in the quasi-monomial form as a new variable of our dynamical system. As mentioned in the introduction, we define Lotka-Volterra systems as follows (note that we abandon the notation for the linear term; we can always replace this term by making sure that one of the variables remains constant, so that one of the terms produces the necessary linear contribution).

Definition 2. *A* Lotka-Volterra system *is a dynamical system of the form*

$$\dot{z}_i = z_i \sum_{j=1}^{m} M_{ij} z_j, \qquad \text{with } j = 1, \dots, m. \tag{12}$$

We have the following theorem [1].

Theorem 1 (Brenig). *For any quasi-monomial form (7) of dimension n and with r terms, there exists a Lotka-Volterra system (12) of dimension $n + r$ for which the dynamics of the first n variables is identical to the dynamics of the quasi-monomial form.*

Proof. With r the number of different monomials in the quasi-monomial form, we define the supplementary variables $z_{n+1}, \dots, z_{n+r}$ as

$$z_{n+k} = \prod_{l=1}^{n} z_l^{B_{kl}}, \qquad \text{with } k = 1, \dots, r.$$

If we now look at the system of $n + r$ variables $\dot{z}_i$, taking into account these new definitions, we have for the first n variables

$$\dot{z}_i = z_i \sum_{j=1}^{r} \bar{A}_{ij} z_{n+j}, \qquad \text{with } i = 1, \dots, n.$$

Because of our special definition of the new variables, many simplifications occur in the expression of their time derivative, as seen in Equation 13. We first use the formula for the derivative of a product (14). Then we try to reconstruct the original variables since they are almost preserved by the derivation (15); the necessary correction leads to the appearance of a logarithmic derivative (16), which in turn transforms the exponent of the monomials into multiplicative coefficients (17). Lastly, the remaining correction produces the new expression of the vector fields of the first n variables, except for the disappearance of the multiplicative variable at the front (18). Writing these computations down, we have, for the r supplementary variables,

$$\dot{z}_{n+j} = (\prod_{k=1}^{n} z_k^{B_{jk}})' \qquad \text{with } j = 1, \dots, r \tag{13}$$

$$= \sum_{l=1}^{n} (\prod_{\substack{k=1 \\ k \neq l}}^{n} z_k^{B_{jk}})(z_l^{B_{jl}})' \tag{14}$$

$$= \sum_{l=1}^{n} (z_{n+j}/z_l^{B_{jl}})(z_l^{B_{jl}})' \tag{15}$$

$$= z_{n+j} \sum_{l=1}^{n} (\ln(z_l^{B_{jl}}))' \tag{16}$$

$$= z_{n+j} \sum_{l=1}^{n} B_{jl} \dot{z}_l / z_l \tag{17}$$

$$= z_{n+j} \sum_{l=1}^{n} B_{jl} \sum_{m=1}^{r} \bar{A}_{lm} z_{n+m} \tag{18}$$

$$= z_{n+j} \sum_{m=1}^{r} ((\sum_{l=1}^{n} B_{jl} \bar{A}_{lm}) z_{n+m}). \tag{19}$$

Thus, we can regroup all these terms by defining the $(n + r) \times (n + r)$ matrix M as

$$M = \begin{pmatrix} O^{n \times n} & \bar{A} \\ O^{r \times n} & B\bar{A} \end{pmatrix},$$

since $\sum_{l=1}^{n} B_{jl} \bar{A}_{lm} = (B\bar{A})_{jm}$; and the total system becomes then of the form

$$\dot{z}_i = z_i \sum_{j=1}^{n+r} M_{ij} z_j, \qquad \text{with } i = 1, \dots, n + r,$$

which is precisely an $n + r$-dimensional Lotka–Volterra system! □

In the case of the dynamical perceptron, we have now the following corollary.

Corollary 1. *For any dynamical perceptron (1) of dimension n, there exists a Lotka–Volterra system (12) of dimension* $(3n^2+n+2)/2$ *for which the dynamics of the first n variables is equivalent to the dynamics of the dynamical perceptron.*

Proof. This corollary is a direct consequence of our transformation of the dynamical perceptron into a quasi-monomial form and of Theorem 1. After the transformation the dynamical perceptron into a quasi-monomial form with matrices (10) and (11) (whose respective dimensions are $r \times n$ and $n \times r$, with $r = (3n^2 - n + 2)/2$), Theorem 1 associates a Lotka–Volterra system of dimension $n + r$ to the quasi-monomial form. Supposing we had a solution $z(t) = (z_1(t), z_2(t), \dots, z_{n+r}(t))^T$ of the equivalent Lotka–Volterra system, we can then retrieve the behavior of the original system by transforming back the n first variables (if the matrix A is invertible):

$$x(t) = A^{-1}(\sigma^{-1}((z_1(t), \dots, z_n(t))^T) - a_0).$$

If the matrix is not invertible, we can apply the argument from the section on dynamical perceptrons to retrieve the solution. □

DYNAMICAL PERCEPTRONS WITH SELF-CONNECTIONS

One criticism that we can formulate about the dynamical perceptron is that its global behavior is often similar to the behavior around its fixed point. Although we can obtain complex dynamics by a careful choice of parameters, this is a delicate operation. The dynamics of such systems appear to be essentially limited. One way to obtain complex oscillatory behaviors more easily is to add a proportional self-connection to each variable:

$$\dot{x}_i = -\lambda_i x_i + \sigma(\sum_{j=1}^{n} A_{ij}x_j + a_{0i}). \tag{20}$$

The reason why we can obtain oscillatory behaviors more easily is that we can choose A to make one of the fixed points unstable and then choose the λ_i sufficiently large so that the behavior of the system always remain bounded. A further remark is that this kind of system can have multiple isolated equilibria, while the previous system either had to have a single equilibrium or a connected set of equilibria. The dynamics of the systems with inhibitory connections is thus richer. Also, the activation function for this system is not limited to the hyperbolic tangent anymore; in a later section, we discuss other possible choices and conclude that we could use the logistic sigmoid as well.

Setup of the Quasi-Monomial Forms

We represent this system by a Lotka–Volterra system. The first step is to define the following auxiliary variables:

$$w_i = \sigma(\sum_{j=1}^{n} A_{ij}x_j + a_{0i}).$$

Looking at the derivative of w, we then have the following dynamics for the system of x and w variables:

$$\begin{array}{rcl} \dot{x}_i &=& w_i - \lambda_i x_i, \\ \dot{w}_i &=& (1-w_i^2)\sum_{j=1}^{n} A_{ij}(w_j - \lambda_j x_j). \end{array} \tag{21}$$

This dynamics is essentially in quasi-monomial form and we can rewrite it as

$$\dot{x}_i = x_i(-\lambda_i + x_i^{-1}w_i),$$
$$\dot{w}_i = w_i \sum_{j=1}^{n}(-\lambda_j A_{ij}x_j w_i^{-1} - \lambda_j A_{ij}x_j w_i + A_{ij}w_i^{-1}w_j + A_{ij}w_i w_j).$$

In the next subsection, we show how to write the matrices $\bar{A}$ and B of the quasi-monomial form of this system. Again, we just have to carefully collect all the exponents of the monomials in the matrix B and all their coefficients in the matrix $\bar{A}$. After this transformation into quasi-monomial form, the transformation from quasi-monomial form to Lotka–Volterra directly applies in the same fashion except that the variable x is directly part of the Lotka–Volterra system and does not need to be retrieved by a change of coordinates (we pay this advantage by having a larger Lotka–Volterra system than for the dynamical perceptron).

Matrices of the quasi-monomial forms for dynamical perceptrons with self-connections

We recall that, for the system (20), the quasi-monomial form is

$$\begin{aligned} \dot{x}_i &= x_i(-\lambda_i + x_i^{-1} w_i), \\ \dot{w}_i &= w_i(\textstyle\sum_{j=1}^{n}(-\lambda_j A_{ij} x_j w_i^{-1} - \lambda_j A_{ij} x_j w_i + A_{ij} w_i^{-1} w_j + A_{ij} w_i w_j)). \end{aligned} \tag{22}$$

To use the quasi-monomial formalism in the same way as in the previous section, we need to unify our formalism and use a single set of variables. We therefore define the following variable $u = (x; w)$:

$$\begin{aligned} u_i = x_i, &\quad \text{with } i = 1, \dots, n; \\ u_{n+i} = w_i, &\quad \text{with } i = 1, \dots, n. \end{aligned}$$

If the matrices $\bar{A}^{(\lambda)}$ and $B^{(\lambda)}$ contain respectively all the coefficients and all the exponents of the different monomials, we can write the dynamics of u in quasi-monomial form:

$$\dot{u}_i = u_i(\sum_{j=1}^{m} \bar{A}_{ij}^{(\lambda)} \prod_{k=1}^{2n} u_k^{B_{jk}^{(\lambda)}}).$$

To build the matrices $\bar{A}^{(\lambda)}$ and $B^{(\lambda)}$, we have to look at all the monomials in (22). If we first focus on the terms $w_i^{-1} w_j$ and $w_i w_j$, which already appeared in in the section on the quasi-monomial form for the dynamical perceptron, we see that we have exactly the same contribution in the matrix $B^{(\lambda)}$ as in the case of the dynamical perceptron. We have the same cancellation of the factors of the terms in $w_i^{-1} w_j$ or $w_i w_j$ as in the section on the quasi-monomial form for the dynamical perceptron. They contribute to the matrix $B^{(\lambda)}$ with a first submatrix equal to B. Note that the left part of the matrix $B^{(\lambda)}$ is for powers of x while the right part is for powers of w. The next blocks take respectively into account the monomials of the form $x_i^{-1} w_i, x_i w_j^{-1}$, and $x_i w_j$. None of these terms ever add up or cancel out, which explains the plain diagonal and column structure of these submatrices. We have

$$B^{(\lambda)} = \left(\begin{array}{c|c} O^{r\times n} & B \\ \hline -I^{n\times n} & I^{n\times n} \\ \hline I^{n\times n} & -K^{(1)} \\ \vdots & \vdots \\ I^{n\times n} & -K^{(n)} \\ \hline I^{n\times n} & K^{(1)} \\ \vdots & \vdots \\ I^{n\times n} & K^{(n)} \end{array}\right),$$

where we define $K^{(i)}$ as

$$K^{(i)} = \left(\begin{array}{c|c|c} & 1 & \\ O^{n\times(i-1)} & \vdots & O^{n\times(n-i)} \\ & 1 & \end{array}\right).$$

Similarly, we collect all the coefficients of the monomials in $\bar{A}^{(\lambda)}$. We place the contributions relating to the equations for the x variables in the top part of the matrix while we place the contributions for w in the bottom part of the matrix. The contributions for the monomials $w_i^{-1}w_j$ and w_iw_j are exactly the same for the variables w as before; the first bottom submatrix is therefore equal to A. These monomials have no influence on the variables x and the first top submatrix is therefore equal to zero; except in the case where $w_i^{-1}w_j = 1$ corresponding to the $-\lambda_i$ terms in the equations for x, which generate a column of $-\lambda_i$ in the otherwise empty matrix Q. The other submatrices take correspondingly into account the coefficients of the monomials of the form $x_i^{-1}w_i, x_iw_j^{-1}$, and x_iw_j. We thus have

$$\bar{A}^{(\lambda)} = \left(\begin{array}{c|c|ccc|ccc} Q^{n\times r} & I^{n\times n} & O^{n\times n} & \cdots & O^{n\times n} & O^{n\times n} & \cdots & O^{n\times n} \\ \hline A & O^{n\times n} & -L^{(1)} & \cdots & -L^{(n)} & L^{(1)} & \cdots & L^{(n)} \end{array}\right),$$

where we define Q and $L^{(i)}$ as

$$Q = \left(\begin{array}{c|c} & -\lambda_1 \\ O^{n\times(r-1)} & \vdots \\ & -\lambda_n \end{array}\right), \qquad L^{(i)} = \left(\begin{array}{c} O^{(i-1)\times n} \\ \hline A_{i1}\lambda_1 \quad \cdots \quad A_{in}\lambda_n \\ \hline O^{(n-i)\times n} \end{array}\right).$$

The following corollary is an immediate consequence of the previous developments and of our main theorem.

Corollary 2. *For any dynamical perceptron with self-connections (20) of dimension n, there exists a Lotka–Volterra system (12) of dimension $(7n^2+n+2)/2$ for which the dynamics of the first n variables is identical to the dynamics of the perceptron.*

The structure of the matrices $\bar{A}^{(\lambda)}$ and $B^{(\lambda)}$ shows that the dimension of the quasi-monomial form is $2n$ and the number of monomials is $r^{(\lambda)} = (3n^2 - n + 2)/2 + n + 2n^2$ (where n is the dimension of the dynamical perceptron with self-connections), which added together give the dimension of the corresponding Lotka–Volterra system. Note that, while in Corollary 1 the dynamics of the first n variables was equivalent to the dynamics of the dynamical perceptron by a smooth transformation, the dynamics the first n variables of the Lotka–Volterra system is here identical to the dynamics of the dynamical perceptron with self-connections. But, again, the Lotka–Volterra system is here about twice as large as the one in Corollary 1.

DYNAMICAL MULTILAYER PERCEPTRONS

The transformations we have showed for the dynamical perceptron and the dynamical perceptron with self-connections apply also to the dynamical multilayer perceptron. To avoid an unnecessary clutter of notation, we take the specific example of a two–hidden-layer continuous-time multilayer perceptron:

$$\dot{x} = \rho(C\rho(B\rho(Ax + a_0) + b_0) + c_0),$$

where we have n states, p hidden units in the first layer, q hidden units in the second layer, and the activation function ρ is the hyperbolic tangent. This means $x(t) \in \mathbb{R}^n, A \in \mathbb{R}^{p\times n}, B \in \mathbb{R}^{q\times p}, C \in \mathbb{R}^{n\times q}, a_0 \in \mathbb{R}^p, b_0 \in \mathbb{R}^q, c_0 \in \mathbb{R}^n$. The argument applies to many kinds of multilayer perceptron by doing the same type of computations again.

In the previous subsection, we showed that, if we use the hyperbolic tangent as our activation function, the transformation of the quasi-monomial form into a Lotka–Volterra system is discontinuous. To avoid this problem, we transform the network to use the logistic sigmoid as our activation function. If we denote by $I^{n\times 1}$ a column vector of ones, we have

$$\begin{aligned} \dot{x} &= \rho(C\rho(B2\sigma(2Ax + 2a_0) - BI^{p\times 1} + b_0) + c_0) \\ &= \rho(C2\sigma(4B\sigma(2Ax + 2a_0) + 2(b_0 - BI^{p\times 1})) - CI^{q\times 1} + c_0) \\ &= 2\sigma(4C\sigma(4B\sigma(2Ax + 2a_0) + 2(b_0 - BI^{p\times 1})) + 2(c_0 - CI^{q\times 1})) - I^{n\times 1}. \end{aligned}$$

By a trivial change of variables, we see that it is enough to consider systems of the form

$$\dot{x} = \sigma(C\sigma(B\sigma(Ax + a_0) + b_0) + c_0) + d_0$$

to cover the case of the two–hidden-layer continuous-time multilayer perceptron with the hyperbolic tangent as its activation.

If we write the equations for this new dynamical multilayer perceptron coordinate by coordinate, we get

$$\dot{x}_i = \sigma(\sum_{l=1}^{q} C_{il}\sigma(\sum_{k=1}^{p} B_{lk}\sigma(\sum_{j=1}^{n} A_{kj}x_j + a_{0k}) + b_{0l}) + c_{0i}) + d_{0i}, (i = 1, \dots, n).$$

We define the following vector of p new variables: $y = \sigma(Ax + a_0)$. Again, we want to discuss the rank of the matrix A. If the matrix A is of rank n, then we can uniquely retrieve x from $y = \sigma(Ax + a_0)$. If the rank s of the matrix A is inferior to n, we have the same argument as in the section on the dynamical perceptron: Two initial conditions $\bar{x}_0$ and $\tilde{x}_0$ such that $A\bar{x}_0 = A\tilde{x}_0$ have the same dynamics in the new variables $\bar{y}(t) = \tilde{y}(t)$ for all times, thus $\bar{x}(t) = \sigma(C\sigma(B\bar{y}(t) + b_0) + c_0) = \tilde{x}(t)$ for all times. This relation means that the trajectories $\bar{x}(t)$ and $\tilde{x}(t)$ through $\bar{x}_0$ and $\tilde{x}_0$ always remain parallel. Since the kernel of A is of dimension $n - s$, this parallelism means that the dynamics is in fact s-dimensional. The dynamics of the variables $y_i = \sigma(\sum_{j=1}^{p} A_{ij}x_j + a_{0i})$ is thus

$$\begin{aligned}
\dot{y}_i &= \sigma'(\sum_{j=1}^{n} A_{ij}x_j + a_{0i})(\sum_{j=1}^{n} A_{ij}\dot{x}_j) \\
&= \sigma(\sum_{j=1}^{n} A_{ij}x_j + a_{0i})(1 - \sigma(\sum_{j=1}^{n} A_{ij}x_j + a_{0i}))(\sum_{j=1}^{n} A_{ij}\dot{x}_j) \\
&= y_i(1 - y_i)(\sum_{j=1}^{n} A_{ij}(\sigma(\sum_{l=1}^{q} C_{jl}\sigma(\sum_{k=1}^{p} B_{lk}y_k + b_{0l}) + c_{0j}) + d_{0j})).
\end{aligned}$$

We can remark that the form of the dynamics of y closely resembles that of the dynamics of x; but that the first hidden layer of the multilayer perceptron has disappeared. This encourages us to define, in the same fashion as before, a new vector of q variables: $z = \sigma(By + b_0)$. The dynamics of the variables $z_i = \sigma(\sum_{k=1}^{p} B_{ik}y_k + b_{0i})$ is thus

$$\begin{aligned}
\dot{z}_i &= \sigma'(\sum_{k=1}^{p} B_{ik}y_k + b_{0i})(\sum_{k=1}^{p} B_{ik}\dot{y}_k) = z_i(1 - z_i)(\sum_{k=1}^{p} B_{ik}\dot{y}_k) \\
&= z_i(1 - z_i)(\sum_{k=1}^{p} B_{ik}y_k(1 - y_k)(\sum_{j=1}^{n} A_{kj}(\sigma(\sum_{l=1}^{q} C_{jl}z_l + c_{0j}) + d_{0j}))).
\end{aligned}$$

Enthusiastically, we pursue further by defining a new vector of n variables: $w = \sigma(Cz + c_0)$. The dynamics of the variables $w_i = \sigma(\sum_{l=1}^{q} C_{il}z_l + c_{0i})$ is thus

$$\begin{aligned}
\dot{w}_i &= \sigma'(\sum_{l=1}^{q} C_{il}z_l + c_{0i})(\sum_{l=1}^{q} C_{il}\dot{z}_l) = w_i(1 - w_i)(\sum_{l=1}^{q} C_{il}\dot{z}_l), \\
\dot{w}_i &= w_i(1 - w_i)(\sum_{l=1}^{q} C_{il}z_l(1 - z_l)(\sum_{k=1}^{p} B_{lk}y_k(1 - y_k)(\sum_{j=1}^{n} A_{kj}(w_j + d_{0j})))).
\end{aligned}$$

Now, substituting the definitions for y, z, w in the previous equations and noting that, because of the same definitions, $\dot{x} = w + d_0$, we get the following system:

$$\begin{aligned}
\dot{x}_i &= w_i + d_{0i}, \\
\dot{y}_i &= y_i(1 - y_i)(\sum_{j=1}^{n} A_{ij}(w_j + d_{0j})), \\
\dot{z}_i &= z_i(1 - z_i)(\sum_{k=1}^{p} B_{ik}y_k(1 - y_k)(\sum_{j=1}^{n} A_{kj}(w_j + d_{0j}))), \\
\dot{w}_i &= w_i(1 - w_i)(\sum_{l=1}^{q} C_{il}z_l(1 - z_l)(\sum_{k=1}^{p} B_{lk}y_k(1 - y_k)(\sum_{j=1}^{n} A_{kj}(w_j + d_{0j})))).
\end{aligned}$$

The dynamics of the first n variables of this $(n+p+q+n)$-dimensional system (with appropriate initial conditions: $x_0 = x_0, y_0 = \sigma(Ax_0 + a_0), z_0 = \sigma(By_0 + b_0), w_0 = \sigma(Cz_0 + c_0)$!) is identical to the dynamics of the original system. Further, we see that we have written the ODE for each variable as a product of the variable itself by a linear combination of products of the variables (and a constant), which is precisely the definition of the quasi-monomial form. An exception is that the first equation is not in quasi-monomial form. But we know that $y = \sigma(Ax + a_0)$ so that we can retrieve x from the knowledge of y. We can thus restrict our attention to the system in y, z, and w, which is a Lotka–Volterra system. Furthermore we know that a fixed point of the original system ($\dot{x} = 0$) must be a fixed point of the Lotka–Volterra system because y, z, and w are static functions of x. Moreover, since $x = w + d_0$, $\dot{x} = 0$ is equivalent to $w = -d_0$. To find fixed points of the original systems, we have to find the fixed points of the Lotka–Volterra system for which $w = -d_0$. We therefore see that there is a strong relationship between the fixed points of the neural network and those of the Lotka–Volterra system, and that we have been capable to avoid the singularities associated with the use of the hyperbolic tangent.

The transformation of the equations for y, z, and w into a Lotka–Volterra system proceeds exactly in the same way as in the sections on the dynamical perceptron and on the dynamical perceptron with self-connections. However, writing down the matrices $\tilde{A}$ and B is a lengthy and tedious procedure and does not bring any further insight. We therefore do not perform it here. This computation will be helpful only when we use a computer algebra system to analyze the resulting Lotka–Volterra system.

CHOICE OF THE ACTIVATION FUNCTION

We chose the activation function σ as $\sigma(\xi) = \tanh(\xi)$ for simplicity. But we are not directly limited to this activation function. The essential property that we used was the fact that $\sigma'(\xi) = 1 - \sigma^2(\xi)$. For the logistic sigmoid $\sigma(\xi) = 1/(1 + e^{-\xi})$, we have $\sigma'(\xi) = \sigma(\xi)(1 - \sigma(\xi))$ and we can do the calculations again without difficulty. The condition that the activation function has to satisfy is that its derivative be a sum of quasi-monomials of itself. If we do not require saturation at plus and minus infinity, we can use $\sigma(\xi) = e^{\xi}$, but the system is not globally Lipschitz and the solution might blow-up in finite time, which is an annoyance. A square function $\sigma(\xi) = \xi^2$ suffers from the same problem. The hyperbola $\sigma(\xi) = 1/\xi$ gives $\sigma'(\xi) = -1/\sigma^2(\xi)$, but then the vector field is not even bounded, which is even worse. In general, we could consider the condition $\sigma'(\xi) = L(\sigma(\xi))$, where $L(.)$ is a finite Laurent series $L(y) = (1/y^m)\sum_{k=0}^{n} a_k y^k$, or even more generally, a Puiseux series. For the Laurent series, the solution of the differential equation is given by separation of variables $\frac{\sigma^m(\xi)}{L(\sigma(\xi))} d\sigma(\xi) = d\xi$. We can then expand the term $\sigma^m(\xi)/L(\sigma(\xi))$ as a polynomial plus a sum of simple fractions, and integrate them. This integration produces a rational function plus a sum of logarithmic terms and arctangent terms and $\sigma(\xi)$ is found by inverting this relation. This inversion does not lead to an explicit solution in general. If we further require that the vector field be globally Lipschitz to guarantee existence of the trajectories for all times, the possible solutions become scarce. From the most frequently used activation functions, we have been able to identify only the hyperbolic tangent and the logistic sigmoid as satisfying these conditions.

CONCLUSION

An important corollary of this representation of continuous-time recurrent multilayer perceptrons into higher-dimensional Lotka–Volterra systems is that we can use Lotka–Volterra systems to approximate arbitrarily closely the dynamics of any finite-dimensional dynamical system for any finite time. The argument for this universal approximation property is exactly the same as in [10] and says that, for any given finite time, the trajectories of two sufficiently close dynamical systems remain close (this is true only for a finite time). This statement is based on the classical use of the Gronwall inequality for dynamical systems. It could therefore be interesting to use Lotka–Volterra systems as approximators for dynamical systems.

The restriction on the choice of the activation function (which is in practice limited to the hyperbolic tangent or the logistic sigmoid) is theoretically strong: There is no guarantee that systems with different choices of activation function have qualitatively the same behavior. Indeed, we can give the example of systems with inputs for which the controllability properties can differ for different types of activation function [11], even if they are monotone increasing sigmoids. However, the practical limitation is fairly weak in the sense that the possible activation functions are the ones most commonly used and that we have universal approximation properties for dynamical systems.

We have presented techniques to represent different types of neural networks (dynamical perceptrons, dynamical perceptrons with self-connections, and multilayer perceptrons) into predator-prey (Lotka–Volterra) systems. These techniques relied on looking at the dynamics of the output of every neuron in the network and using the special properties of the derivative of the hyperbolic tangent or logistic sigmoid activation function to write the system as what we called a quasi-monomial form—that is a dynamical system where the vector field is a linear combination of products of powers of the state variables. Once we had this quasi-monomial form, we looked at the dynamics of these products of powers of the state variables to obtain directly a Lotka–Volterra system [1]. Our expectation is that these transformations will be useful to migrate techniques for the analysis of Lotka–Volterra systems into the field of dynamical neural networks—for example, techniques to study convergence or stability properties of dynamical systems, or discover first integrals of motion.

Acknowledgments

Yves Moreau is a Research Assistant with the F.W.O.-Vlaanderen. Joos Vandewalle is Full Professor at the K.U.Leuven. Their work is supported by several institutions: the Flemish Government, GOA-MIPS (Model-based Information Processing Systems); the FWO Research Community, ICCoS (Identification and Control of Complex Systems); the Belgian State, DWTC - Interuniversity Poles of Attraction Programme (IUAP P4-02 (1997-2001): Modeling, Identification, Simulation and Control of Complex Systems.
Léon Brenig and Stéphane Louies are grateful to the CATHODE ESPRIT Working Group (WG #24490) for scientific and financial support. They would like also to thank A. Figueiredo and T. M. da Rocha from the Physics Departement of the University of Brasilia for fruitful scientific discussions.

REFERENCES

1. L. Brenig, *Physics Letters A* **133**, 378 (1988).
2. R. H. MacArthur, *Proceedings of the National Academy of Sciences, USA* **54**, 1369 (1969).
3. R. M. May and W. J. Leonard, *SIAM Journal of Applied Mathematics* **29**, 243 (1975).
4. Y. Takeuchi, *Global dynamical properties of Lotka-Volterra systems*, World Scientific Publishing Company, Singapore, 1996.
5. J.-L. Gouzé, Transformation of polynomial differential systems in the positive orthant, Technical Report INRIA-1308, Institut National de Recherche en Informatique et en Automatique: Unité de Recherche INRIA-Sophia Antipolis, France, 1990.
6. R. Redheffer and W. Walter, *Journal of Differential Equations* **52**, 245 (1984).
7. S. Grossberg, *Journal of Mathematical Analysis and Applications* **66**, 470 (1978).
8. S. Grossberg, *Neural Networks* **1**, 17 (1988).
9. M. Peschel and W. Mende, *The predator-prey model*, Springer-Verlag, New York, 1986.
10. K.-I. Funahashi and Y. Nakamura, *Neural Networks* **6**, 801 (1993).
11. E. D. Sontag and H. J. Sussman, *Systems and Control Letters* **30**, 177 (1997).