An. Ştiinţ. Univ. Al. I. Cuza Iaşi. Mat. (N.S.)
Tomul LXVI, 2020, f. 2

An artificial neural network approach for the dynamics of enzyme-catalysed reactions

Ana-Maria Moşneagu · Iulian Stoleriu

Abstract Biochemical reactions are chemical reactions that take place inside the cells of living beings. Catalysts are substances that enable a chemical reaction to proceed at faster rates by lowering down the activation energy needed for a reaction. Enzymes are biological catalysts that play a crucial role in accelerating the biochemical reactions. One important feature of enzymatic reactions is the enzyme saturation, which occurs when all the enzyme active sites are already occupied by substrate. At saturation, the reaction rate appears to attain an apparent maximum. Therefore, it is assumed that, after a short transient state, the dynamics of the reaction reaches a quasi-steady state. The dynamics of the reactions that take place in a biochemical process can be captured by a set of coupled ordinary differential equations. In this paper, we employ an artificial neural network (ANN) approach to approximate the solutions of the obtained dynamical system. Our method is based on a mixture of feed-forward artificial neural networks and optimization techniques. Several numerical examples are performed to show the efficiency of the method, and the numerical results are compared with the ones obtained via the classical Runge-Kutta method.

Keywords artificial neural network (ANN) · feed-forward neural networks · ordinary differential equations · enzyme-kinetics · quasi-steady state · open system

Mathematics Subject Classification (2010) 92B20 · 92C45 · 68T07 · 34D05 · 34C25

1 Introduction

All living beings need energy to support life. The vital energy comes from the reactions that take place within living things, which are called biochemical reactions. Two of the most important biochemical reactions are the photosynthesis and the cellular respiration. Therefore, the study of how biochemical reactions take place in living organisms is of great importance for the understanding of life. Besides this, the interest of scientist in enzyme-catalysed processes and reactions for organic synthesis is constantly growing in the pharmaceutical or cosmetics industry.

A biochemical reaction is the transformation of one molecule into a different molecule inside the cell. These reactions are mediated by some catalysts, which are substances that can significantly speed up the rate of chemical reactions. Typically, catalysts are large molecules that can process millions of molecules every second. Catalysts are said to exert a catalytic reaction, and a reaction in which a catalyst is involved is called a catalysed reaction. An enzyme is a protein that acts as a catalyst inside a living cell and thus help to support life. Enzymes are found in all tissues and fluids of the body. Without enzymes, most chemical reactions in living things would occur too slow to keep organisms alive. They act on substrates (a substrate is reactant, other than catalyst, participating in a catalysed reaction), and they facilitate the reactions in which they participate, but they are not themselves consumed in the process. A chemical reaction occurs only if the molecules possess a minimum amount of energy (called the activation energy), and the magnitude of this energy will determine how fast a reaction will proceed. The lower the activation energy, the faster will be the reaction. In chemical reactions, the activation energy is usually obtained by increasing the temperature or the concentration of the reactants. As biological systems cannot raise heat or concentrations at will, enzymes facilitate the speeding up process by lowering the activation energy without changing the Gibbs free-energy required for this reaction to occur.

Enzyme kinetics deals with time-dependent enzyme reactions that serve to describe mechanisms of enzyme catalysis and regulation. Of major importance is the study of the rates at which enzymatic reactions proceed. These rates may be influenced by factors such as: enzyme or substrate concentrations, the presence of activators or inhibitors, temperature, pH etc. Enzymes contain some unique sites (called active sites) where other molecules can bind, forming a mixture complex. The most important and thus most extensively studied ligand is the substrate which is then converted into product by the enzyme. The substrate temporarily binds to the enzyme and, doing so, the activation energy needed for a conversion into product is lowered. After the binding, substrates undergo an enzyme-induced chemical change, and then they are converted into products.

Complex biochemical processes are usually described as a succession of simple and reversible binding steps and largely irreversible catalytic steps. Each step constitutes an elementary reaction. The simplest case of an enzyme-catalysed reaction is the conversion of a single substrate into product, as for isomerisations and cleavage reactions (for more details related to this subject one can consult, for example, [1]). If we denote by E, S, C and P the concentrations of, respectively, the free enzyme, the free substrate, the enzyme-substrate complex and the product, then the schematic representation of this biochemical process is represented in Figure 1, where k_1, k_{-1} and k_2 are constant parameters associated with the rates of the reaction. Note that the concentrations of the reactants are changing in time. In the first step, enzyme and substrate bind to each other to form a complex C. As the enzyme facilitates the crossing over the activation barrier, the substrate is transformed rapidly into product. Enzymes and products then dissociate to form E and P. The first reaction that takes place is reversible,

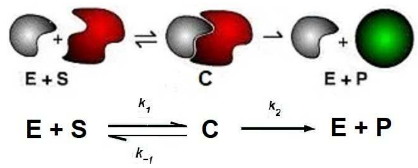

Fig. 1 A canonical enzyme-catalysed reaction

that is, under certain conditions, the complex may break-down at a rate k_{-1}. For simplicity, it is generally assumed that the second reaction is irreversible. Typically, it is assumed that the rate k_2 is much greater than the rate of the reverse reaction (practically, one assumes that product is continually removed), which prevents the reverse reaction of the second step from occurring. In reality, this is not always the case.

Let us also note that the set of chemical reactions, as they are presented here, acts as being separated from the outside world, and thus can only be considered as a schematic model for an *in vitro* biochemical process (that is, a process that takes place in an isolated environment). However, in a living system, large numbers of enzymatic reactions are networked in a complex manner. A particular enzymatic reaction, as the one in Figure 1, is embedded in such a pathway, taking product molecules from the previous reaction step and supplying substrate to the next step. Moreover, a living system may take up substrates from external sources and releases products to them. Therefore, all enzymatic reactions in a living system are *in vivo*, and so they are subject to substrate input and product removal. In [15] and [16], the authors added an input of substrate (constant or periodic) to the scheme presented in Figure 1, and then studied how the dynamics of the newly created models are changing as a consequence of this fact.

In Chemistry and Chemical engineering, the law of mass action is employed to explain and predict behaviours of solutions that are in dynamic equilibrium. This law states that the rate of the chemical reaction is directly proportional to the product of the concentrations of the reactants. The natural language for describing chemical kinetics in a mass action approximation are networks of coupled ordinary differential equations. Using the law of mass action, one can write the governing equations for the biochemical reaction in Figure 1 as in (2.1) - (2.4) in the next section.

In order to solve this system of differential equations, the time-dependent concentration changes must be known, fact that is hard to realize in practice. There is no known method that can provide an analytical solution to the system. One can, however, calculate numerical solutions for any specific values of the initial conditions and kinetic parameters by evaluating the equations on a computer. For example, one can easily discretize the above system of equations and then simulate its solutions on a computer using some conventional numerical methods, such as the Runge-Kutta methods, the predictor-corrector methods or the finite-element methods, by which the solution can be approximated to the desired level of accuracy.

In this paper we follow a different approach, and model the dynamics of the above system by employing an artificial neural network setting. Similar approaches have been used in [6] and [8] but for different mathematical problems. Machine-learning algorithms have become increasingly popular in solving a wide variety of problems that arise in sciences. Motivated by the known fact that a multilayer perceptron with one hidden layer can approximate any function to arbitrary accuracy, it is reasonable to consider this type of network architecture as a candidate model to approximate the solution of the above system of differential equations.

An artificial neural network is a piece of a computing system that is trained to simulate the way the human brain analyses and processes information. We shall use here a feed-forward ANN as the basic approximation element of the solution of the system, whose hyper-parameters (weights and biases) are adjusted at each iterative step to minimize an appropriate cost function. To train the network we employ optimization techniques, which require the computation of the gradient of the cost function with respect to the network parameters. The model function is expressed as the sum of two terms: the first term satisfies the initial conditions and it contains no adjustable hyper-parameters; the second term is a feed-forward neural network with one hidden layer. Starting from a set of randomly chosen parameters, the ANN will be trained to satisfy the set of differential equations, and then to predict our solution on the desired time interval. The trial solution of the model is generated by training the algorithm. The computational complexity of the method does not increase considerably with the number of sampling points. The method we present here can be easily adapted to solve any linear or nonlinear ordinary and partial differential equations.

The idea of approximating the solutions of differential equations with neural networks was firstly implemented in [4], where training was accomplished by minimizing a loss function based on the networks satisfaction of the boundary conditions and the differential equations.

The approximate solution by ANN has many benefits compared with traditional numerical methods. We mention here some of them (for more details, see [6]).

- The ability of implicitly detecting complex nonlinear relationships between dependent and independent variables.
- Any solution obtained via ANNs is differentiable and written in a closed analytic form, that can be easily employed in any subsequent calculation. As a comparison with the Runge-Kutta method, the latter one offers only a discrete approximation of the solution.
- The number of model parameters that are needed to perform a solution approximation based on ANN's is far less than any other solution technique. Therefore, compact solution models are obtained, with low demand on memory space.

- The ANN method can be applied to a wide range of mathematical problems, such as: ordinary differential equations (ODEs), partial differential equations (PDEs), systems of ODEs or PDEs, regression analysis etc.
- The method can be implemented in hardware, by the use of neuroprocessors, and hence offer the opportunity to tackle in real time difficult differential equation problems arising in many engineering applications.
- The method can also be efficiently implemented on parallel architectures.

However, we also mention here some disadvantages of the ANN setting:

- The main disadvantage of the ANN methods is that an ANN model is a black-box model, and thus very little is understood from what happens inside the model. With other words, there is no simple link between the weights and the function that is approximated. Despite the fact that a proper built ANN model can approximate any function, studying its structure won't give you any insights on the structure of the function being approximated.
- The network architecture includes a set of hyper-parameters that has to be tuned in the model, fact that can become very laborious and/or computationally expensive for large networks.
- The tuning of these parameters comes with the minimization of some cost function. In some cases you don't get the nice guarantees that come with convex optimization, such as global minima.

If the benefits and the drawbacks of using an ANN method are weighted, the decision is evidently in the favour of using it.

Another important limitation, this time of any numerical method implemented on a computer, is that they are scale-aware, that is, it cannot deal with large intervals for the range of the input layer, as the errors accumulated over a larger range build up. In this paper, we have overcome this inconvenient by decomposing the time interval used in simulations into smaller intervals, and then applied the ANN algorithm for each subsequent interval. The solution at the end of each interval becomes the initial solution for the new time interval. At the end, we have glued together these solutions and obtained the solution on the desired time interval. However, when it is possible, the input data can be scaled to a smaller size interval, such as [0, 1] or [0, 2], where the approximation errors do not accumulate very fast. The problem with this approach is that, when scaling down t, the derivative of the solution with respect to t increases, making the solution oscillating more drastically and thus harder to fit.

The remaining of the paper is organized as follows. The next section contains a review of the most important mathematical models for the *in vitro* enzyme-catalysed reaction depicted in Figure 1. We shall discuss the mathematical models derived via the law of mass action, via the Michaelis-Menten formalism and via the Briggs-Haldane mechanism. In Section 3, we discuss how the previous mathematical models can be adapted to analyse the biochemical reaction depicted in Figure 3, and compare the results with the ones obtained in the previous section. Section 4 introduces the setting for the numerical approximation of the enzyme-catalysed reaction using an ANN approach. Section 5 contains some numerical tests to illustrate the effectiveness of the proposed numerical method. Finally, some comments and ideas of future work are presented in the last section of the paper.

2 Mathematical models for enzyme-catalysed reactions *in vitro*

By employing the law of mass action, one can transform the reactions in Figure 1 into the following system of differential equations:

$$\frac{dE}{dt} = -k_1 ES + k_{-1}C + k_2 C, \tag{2.1}$$
$$\frac{dS}{dt} = -k_1 ES + k_{-1}C, \tag{2.2}$$
$$\frac{dC}{dt} = k_1 ES - k_{-1}C - k_2 C, \tag{2.3}$$
$$\frac{dP}{dt} = k_2 C. \tag{2.4}$$

This system is usually accompanied by a set of initial conditions of the form

$$E(0) = E_0, \;\; S(0) = S_0, \;\; C(0) = 0, \;\; P(0) = 0.$$

One can easily observe that, by adding the first and the third equation, one arrives at $E(t) + C(t) = E_0 = const.$, which is the conservation of enzyme law. However, the system obeys a second conservation law, that is

$$S(t) + C(t) + P(t) = S_0, \; t \geq 0.$$

Using the law for the conservation of enzyme, the system can be reduced to only three equations:

$$\frac{dS}{dt} = -k_1(E_0 - C)S + k_{-1}C, \tag{2.5}$$
$$\frac{dC}{dt} = k_1(E_0 - C)S - (k_{-1} + k_2)C, \tag{2.6}$$
$$\frac{dP}{dt} = k_2 C. \tag{2.7}$$

The last equation is decoupled from the first two equations, and represents the turnover rate v in product concentration. Therefore, the minimal set of equations that describes the evolution of the biochemical reaction is formed only by two equations, namely (2.5) and (2.6).

Under some specific assumptions, Michaelis and Menten proposed in [9] a mathematical model that further simplifies this system of equations. This model is widely known as the Michaelis-Menten model and is now one of the best-known models of enzyme kinetics. The assumptions in their model are:

- the substrate and enzyme react very rapidly to form the enzyme-substrate complex C;
- the reaction rate k_2 is relatively small compared to k_{-1};
- the substrate concentration greatly exceeds the enzyme concentration, so that all enzyme activation sites become saturated with substrate. If this is the case, one may assume that the complex creation remains fairly constant in time and an equilibrium is established between E, S and C.

With other words, Michaelis and Menten assumed that the enzyme reactions do not follow the law of mass action directly. At equilibrium, we shall have that

$$k_1\, E\, S = k_{-1}\, C.$$

Using the conservation of energy, one can easily obtain that

$$C = \frac{E_0\, S}{K_d + S},$$

where $K_d = \dfrac{k_{-1}}{k_1}$ is the dissociation constant for the enzyme-substrate complex. Hence, the production rate is

$v = \dfrac{dP}{dt} = \dfrac{v_{max}\, S}{K_d + S}$, where $v_{max} = k_2\, E_0$ is the maximum reaction velocity.

We can easily see that for $S \ll K_d$, we have $S + K_d \approx K_d$, therefore $v \propto S$. This means that at low substrate concentrations the enzyme will not be saturated with substrate and the complex concentration C (and therefore the production rate v) will be proportional to the concentration of S. On the other hand, for $S \gg K_d$, we can write $S + K_d \approx S$ and then $v = v_{max}$. This means that for large amounts of substrate, the reaction takes place at constant maximum velocity.

One major critique of the Michaelis-Menten approach is that the assumption $k_2 \ll k_{-1}$ is not always valid, and thus the equilibrium is only an approximation. An alternative analysis of the system was undertaken by Briggs and Haldane in 1925 in [3]. The main idea of their approach is that the complex concentration remains fairly constant on the time-scale of product formation, assumption that is known as *the quasi-steady-state assumption*. Based of the Briggs-Haldane approach, a complex formation rapidly occurs over a short period of time. When the substrate concentration is much larger than the enzyme concentration, the enzyme will be saturated with substrate and the reaction cannot proceed any faster by adding more substrate. As a consequence, the reaction production rate remains fairly constant in time. From equation (2.6) we see that the complex concentration C is also constant in time, and thus $\frac{dC}{dt} \approx 0$. By setting $\frac{dC}{dt} = 0$ in (2.6),

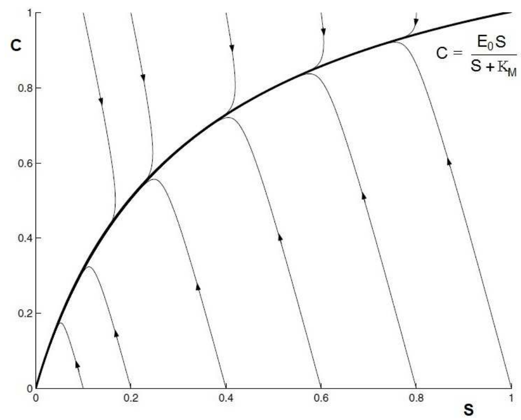

Fig. 2 The solutions of (2.5) and (2.6) for different initial data (thin lines), approaching the invariant manifold (2.9) (bold line).

the system formed by the differential equations (2.5) and (2.6) reduces to a differential-algebraic system:

$$\frac{dS}{dt} = -k_1(E_0 - C)S + k_{-1}C, \tag{2.8}$$

$$C = \frac{E_0\, S}{S + K_M}, \tag{2.9}$$

where $K_M = \frac{k_{-1}+k_2}{k_1}$ is the so called Michaelis-Menten constant. The production rate becomes

$v = \dfrac{dP}{dt} = \dfrac{v_{max}\, S}{S + K_M}$, where $v_{max} = k_2\, E_0$ is the maximum reaction velocity.

We also obtain that, during the quasi-steady state, the substrate concentration satisfies the equation

$$\frac{dS}{dt} = -\frac{v_{max}\, S}{S + K_M}. \tag{2.10}$$

In [11], Schnell and Mendoza found a closed form solution for the problem (2.10) with $S(0) = S_0$, which is

$$S(t) = K_M\, W\left(\frac{S_0}{K_M}\, exp\left(\frac{S_0 - v_{max}t}{K_M}\right)\right),$$

where $W(\cdot)$ is the Lambert W function.

As we have seen, the derivation of this formalism is based on certain assumptions, principally, the quasi-steady state assumption (QSSA). The validity of these assumptions has been the subject of much attention. It

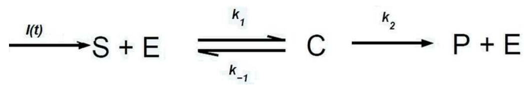

Fig. 3 A canonical enzyme-catalysed reaction with added substrate input

was initially proposed that a necessary condition for the standard or classical (sQSSA) assumption is that the initial substrate concentration greatly exceeds the concentration of the enzyme. This situation often arises in laboratory experiments, but is less common for reactions *in vivo*. Consequently, new weaker necessary conditions on the initial concentrations have been derived. Such validity conditions are, for example:

$$E_0 \ll S_0+K_M \quad \text{(in [12], [13])} \quad \text{or} \quad k_2 E_0 \ll k_1(E_0+S_0+K_M)^2 \quad \text{(in [2], [13])}.$$

Moreover, when the initial enzyme concentration greatly exceeds that of the substrate concentration, a reverse quasi-steady state assumption (rQSSA) can be made ([13] and below). In the case of a very large enzyme concentration when compared with the substrate concentration ($E_0 \gg S_0$ and $E_0 \gg \frac{k_2}{k_1}$), the substrate remains fairly constant in time and one may assume that $\frac{dS}{dt} \approx 0$. By using this assumption in the system (2.5) and (2.6), we arrive at

$$S = \frac{K_d C}{E_0 - C}, \qquad \left(K_d = \frac{k_{-1}}{k_1}\right)$$
$$\frac{dC}{dt} = -k_2 C.$$

In both approaches, the dynamics of the system (2.1) - (2.4) is composed of two phases: the fast transient, in which the substrate rapidly binds to the enzymes and the quasi-steady-state phase, in which enzyme becomes saturated with substrate and the reaction rate becomes maximum and constant.

3 Mathematical models for enzyme-catalysed reactions *in vivo*

Although much effort has been made to elucidate the conditions for applying the sQSSA or the rQSSA to derive Michaelis-Menten formalism, almost all previous work has concentrated on the reactions with no substrate input. As an enzymatic reaction is only a small part of a larger pathway of reactions, we must also consider the set of equations (2.1) - (2.4) in a *in vivo* environment. Typically, enzymatic reactions take product molecules from the previous reaction step in a pathway and supply substrate to the next step. In [14], [15] and [16], the authors added a substrate input to the biochemical reaction of the general form

$$I(t) = I_0(1 + \epsilon \sin(\omega t)), \quad t \geq 0, \tag{3.1}$$

and investigated the effects of this substrate input on the derivation and validity of classical quasi-steady state assumptions. The form of this input is motivated by the fact that, in the case of a periodic input, the amplitude of input to an enzymatic reaction is generally controlled by the maximum enzyme activities of preceding enzyme reactions. Therefore, the amplitude of such inputs is usually limited to a certain range. The questions raised in these papers were:

– What are the conditions for the validity of the sQSSA (resp., the rQSSA)?
– Given that sQSSA (or rQSSA) holds in the case of an isolated biochemical reaction, that is $I_0 = 0$, what is the effect of input on the QSSA?
– Given arbitrary kinetic rates and initial conditions, can an input I_0 be found that moves the system into the QSS?

The time evolution of the concentrations of the reactants in Figure 3 can be described by the following system of ordinary differential equations:

$$\frac{dE}{dt} = -k_1 ES + k_{-1}C + k_2 C, \tag{3.2a}$$

$$\frac{dS}{dt} = I(t) - k_1 ES + k_{-1}C, \tag{3.2b}$$

$$\frac{dC}{dt} = k_1 ES - k_{-1}C - k_2 C, \tag{3.2c}$$

together with the decoupled equation,

$$\frac{dP}{dt} = k_2 C. \tag{3.2d}$$

The typical initial data are:

$$E(0) = E_0, \;\; S(0) = S_0, \;\; C(0) = 0 \text{ and } P(0) = 0. \tag{3.3}$$

The case of constant substrate input (that is, when $\epsilon = 0$, thus $I(t) \equiv I_0$), was intensively studied in [14] and [15]. By using the global invariant manifold theory, it was found that, if the quasi-steady state assumption holds for a system without input (that is, with $I_0 = 0$), then it continues to hold for all positive inputs. Moreover, it was found that the QSSA gets even better, in the sense that the necessary condition for validity is relaxed by increasing I_0, and thus the QSSA is valid for a wider range of parameters and initial data. Furthermore, the errors in product production rate decrease with increasing I_0. These results are contained in the following theorem:

Theorem 3.1 1. *Suppose that the initial data and kinetic parameters are chosen such that the following conditions hold for $I_0 = 0$.*

- (H_1) : *$S(t)$ is approximately constant during the transient period,*
- (H_2) : *the transient time scale t_1 is much less than the time scale t_2 for the quasi-steady-state period.*

Then they hold all $I_0 \geq 0$.
2. *For any initial data and any choice of values of the kinetic parameters, there exists an $I_0 \geq 0$ (dependent on the initial data and kinetic parameters), such that (H_1) and (H_2) hold for all $I \geq I_0$.*

It was also noted that, even in the case where the introduction of (a possible sufficiently large) input maintains or widens the applicability of a QSSA, the time scales associated with the transient and QSS periods, the form of the quasi-steady state solution itself, and the associated reaction rate equation are all dependent on the value of the input. Therefore the formalism established in the absence of input does not, in general, provide accurate information regarding systems with input.

In [16], the authors investigated the effects of small perturbations on the dynamics of the system (3.5), on the timescales for each phase in the dynamics, and on the validity conditions for the quasi-steady state. In order to be able to perform comparison with the previous cases, the initial data for the system (3.2) was chosen as

$$E(0) = E_0, \;\; S(0) = S_0 + \hat{S}, \;\; C(0) = \hat{C} \;\text{ and }\; P(0) = 0, \tag{3.4}$$

with $\hat{S} = \frac{K_M I_0}{k_2 E_0}$ and $\hat{C} = \frac{I_0}{k_2}$, where $K_M = \frac{k_2+k_{-1}}{k_1}$. Note that, in this case, the enzyme conservation law is

$$E(t) + C(t) = const. = E_0 + \hat{C}.$$

Using this, and following similar arguments to those outlined above, it is straightforward to show that if we consider the initial data given in (3.4), then the system of differential equations (3.2) can be rewritten as

$$\frac{dS}{dt} = I(t) - k_1(E_0 + \hat{C} - C)S + k_{-1}C, \tag{3.5a}$$
$$\frac{dC}{dt} = k_1(E_0 + \hat{C} - C)S - k_1 K_M C, \tag{3.5b}$$

together with the uncoupled equation for $P(t)$, that is equation (3.2d).
For ϵ sufficiently small, the dynamics of the system can be divided into three phases. Firstly, a rapid transient takes place, which occurs on the timescale

$$t_1 \approx \frac{1}{k_1(S_0 + \hat{S} + K_M)}. \tag{3.6}$$

During the transient phase, the substrate concentration S remains approximately unaltered but one can observe a rapid increase in complex concentration C. Notice that this time scale is unaltered by the introduction of the perturbation.

In the second stage, the enzyme becomes saturated with input and a quasi-steady-state occurs. Making the sQSSA with respect to system (3.5), the following differential-algebraic system was derived:

$$\frac{dS}{dt} = I(t) - \frac{k_2 E_0 S}{S + K_M}, \quad C = \frac{(E_0 + \hat{C})S}{S + K_M}. \tag{3.7}$$

In the second stage, the solution of the full system remains very close to the quasi-steady-state manifold, that is very close to the solution of the approximate system (3.7). The quasi-steady-state occurs on the timescale

$$t_2 \approx \frac{S_0 + \hat{S} + K_M}{k_2 E_0} + \mathcal{O}(\epsilon), \tag{3.8}$$

By setting $\epsilon = 0$ in t_2, one can rediscover the quasi-steady-state timescale for the system with constant input I_0.

Finally, in the third stage, the solution to the full system settles to a seemingly periodic behaviour with period $\frac{2\pi}{\omega}$, i.e. the same period as the input oscillation. By using the Brouwer Fixed Point Theorem, it was proven that

Theorem 3.2 *For any value of ϵ, $0 \le \epsilon \le 1$, system (3.5) has at least one positive $\frac{2\pi}{\omega}$−periodic solution. Moreover, for ϵ sufficiently small, $\frac{2\pi}{\omega}$−periodic solution is unique and varies continuously with ϵ.*

We see that the period of the periodic solution equals the period of the input oscillations. Numerical approximation of the Floquet multipliers shows that the periodic solution is linearly stable. As the frequency ω is increased, the limit cycle reduces in size.

It was also concluded in [16] that that the standard quasi-steady-state approximation remains valid to the full dynamics for a wide range of input amplitudes and frequencies. The most accurate QSSA is found when the input performs high frequency, small amplitude oscillations around a positive mean value. This accuracy remains high even in the case of large perturbations, provided the frequency of oscillation is sufficiently high. The periodic solution becomes a stronger attractor with increasing ϵ, which perhaps is to be expected, as the forcing term becomes more dominant in the dynamics. This fact will be also shown graphically in the numerical experiments using ANN's in the next section.

4 The artificial neural network setting

In this section, our purpose is to approximate the solution of system (3.5) on a finite time interval $[0, T]$ via a numerical method that combines artificial neural networks (ANN) and optimization techniques. Similar approaches have been used in [6] and [8] but for different mathematical problems. The universal approximation theorem (see [5]) states that a multilayer perceptron network with one hidden layer can approximate any continuous function to arbitrary accuracy. For this reason, we shall build in the following a network architecture that can approximate both concentrations S and C. For more details on machine learning and the architecture of neural networks, one can consult, for example, [10].

If we consider a discretization of time interval $[0,\, T]$ into a set of arbitrary points $\{t_i\},\ i = 1, \ldots, n$, the system (3.5) will be approximated by

$$\begin{aligned} \frac{dS}{dt}(t_i) &= I_0(1 + \epsilon \sin(\omega t_i)) - k_1(E_0 + \hat{C} - C(t_i))S(t_i) + k_{-1}C(t_i), \\ \frac{dC}{dt}(t_i) &= k_1(E_0 + \hat{C} - C(t_i))S(t_i) - k_1 K_M C(t_i),\ i = 1, \ldots, n. \end{aligned} \tag{4.1}$$

subject to the constraints imposed by the initial conditions $S(0) = S_0 + \hat{S}$ and $C(0) = \hat{C}$.

As specified in the beginning of this section, we intend to use multilayer perceptron networks with one hidden layer to approximate the solution of (4.1). In Figure 4, we have represented a neural network with three layers: one input layer, one hidden layer, and one linear output layer. We have

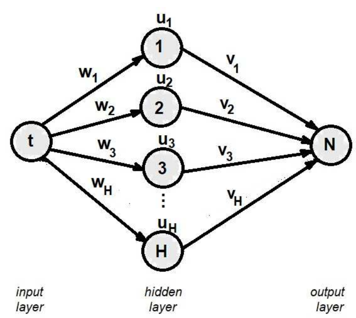

Fig. 4 The architecture of the three-layer ANN

taken $t \in [0, T]$ as the input layer. We consider that the hidden layer has H sigmoid units, and the neural network adjustable parameters are the input and output weights w_j and v_j, and the biases u_j of each hidden unit j, $j = 1, \ldots, H$. Here, w_j denotes the weight from the input unit to the hidden unit j, and v_j denotes the weight from the hidden unit j to the output. Usually, the numbers of nodes H in the hidden layer are taken by trial and error method. The output of the network is

$$N(t, w, u, v) = \sum_{j=1}^{H} v_j \sigma(w_j t + u_j), \tag{4.2}$$

where $w = (w_1, w_2, \ldots, w_H)$, $u = (u_1, u_2, \ldots, u_H)$, $v = (v_1, v_2, \ldots, v_H)$, and $\sigma(x) = \frac{1}{1+e^{-x}}$ is the sigmoid activation function.

We are now building the trial solution for the method. We consider that each component of a trial solution (S^*, C^*) of the system (4.1) is represented by a sum of two terms: the first one includes the corresponding initial condition, and the other one includes a feed-forward neural network with the architecture already specified. The second term implies the neural network adjustable parameters that will be determined using some optimization techniques. Unlike the second term, the first one does not contain any adjustable training parameters. Thus, we consider a neural network for each component of a trial solution (S^*, C^*) of (4.1) :

$$\begin{aligned} S^*(t, w^1 u^1, v^1) &= S_0 + \hat{S} + tN_1(t, w^1, u^1, v^1), \\ C^*(t, w^2, u^2, v^2) &= \hat{C} + tN_2(t, w^2, u^2, v^2), \end{aligned} \tag{4.3}$$

where, as in (4.2),

$$N_1(t, w^1, u^1, v^1) = \sum_{j=1}^{H} v_j^1 \sigma(w_j^1 t + u_j^1), \tag{4.4}$$

$$N_2(t, w^2, u^2, v^2) = \sum_{j=1}^{H} v_j^2 \sigma(w_j^2 t + u_j^2),$$

with $w^1 = (w_1^1, w_2^1, \ldots, w_H^1)$, $u^1 = (u_1^1, u_2^1, \ldots, u_H^1)$, $v^1 = (v_1^1, v_2^1, \ldots, v_H^1)$, the adjustable parameters of the neural network corresponding to S, and $w^2 = (w_1^2, w_2^2, \ldots, w_H^2)$, $u^2 = (u_1^2, u_2^2, \ldots, u_H^2)$, $v^2 = (v_1^2, v_2^2, \ldots, v_H^2)$ the adjustable parameters of the neural network corresponding to C. Let us notice that the trial solution satisfies the initial conditions for the system (4.1).

For the easiness of writing, we denote by f_{S^*} the right hand side of the first equation of the approximated system (4.1), and by f_{C^*} the right hand side of the second equation with respect to the trial solution (S^*, C^*). The problem of finding the solution for (4.1), namely the approximated solution of the system (3.5) over the discretization set of points $\{t_i\}$, $i = 1, \ldots, n$, is equivalent to determine the solution (S^*, C^*) that satisfies the following unconstrained optimization problem

$$\underset{\theta_1, \theta_2}{Minimize}\ E(\theta_1, \theta_2), \tag{4.5}$$

where $\theta_1 = (w^1, u^1, v^1)$, $\theta_2 = (w^2, u^2, v^2)$, and

$$\begin{aligned} E(\theta_1, \theta_2) = &\sum_{i=1}^{n} \left(\frac{dS^*}{dt}(t_i, \theta_1) - f_{S^*}(t_i, \theta_1, \theta_2) \right)^2 + \\ &\sum_{i=1}^{n} \left(\frac{dC^*}{dt}(t_i, \theta_2) - f_{C^*}(t_i, \theta_1, \theta_2) \right)^2. \end{aligned} \tag{4.6}$$

The minimization of the cost function (4.6) represents in fact a way to training the neural network. The reason is as follows: the approximation error corresponding to each input t_i has to become zero. The cost function (4.6) is also called the error function, and the training algorithm is called back-propagation. Firstly, the set of hyper-parameters (θ_1, θ_2) of the neural networks are initialized with uniformly generated random numbers in the interval [0, 1]. Then, we calculate the corresponding cost functional and its gradient. Our aim is to efficiently minimize the error function, and obtain optimal values for θ_1 and θ_2. With the updated parameters, we resume the learning process until the specified number of epochs is reached. The final optimal values for θ_1 and θ_2 are used to predict the solution of (4.1) for any arbitrary set of test points from $[0, T]$.

In order to evaluate the total error, we need to compute the following derivatives, taking into account relations (4.3) and (4.4). We shall have:

$$\frac{dS^*}{dt}(t,\theta_1)=N_1(t,\theta_1)+t\frac{dN_1}{dt}(t,\theta_1)=\sum_{j=1}^{H}v_j^1\sigma(w_j^1t+u_j^1)+t\sum_{j=1}^{H}v_j^1w_j^1\sigma'(w_j^1t+u_j^1),$$
$$\frac{dC^*}{dt}(t,\theta_2)=N_2(t,\theta_2)+t\frac{dN_2}{dt}(t,\theta_2)=\sum_{j=1}^{H}v_j^2\sigma(w_j^2t+u_j^2)+t\sum_{j=1}^{H}v_j^2w_j^2\sigma'(w_j^2t+u_j^2),$$

for an arbitrary $t \in [0,T]$. Thus, the cost functional (4.6) becomes

$$\begin{aligned}E(\theta_1,\theta_2)=&\sum_{i=1}^{n}\left(\sum_{j=1}^{H}v_j^1\sigma(w_j^1t_i+u_j^1)+t_i\sum_{j=1}^{H}v_j^1w_j^1\sigma'(w_j^1t_i+u_j^1)-f_{S^*}(t_i,\theta_1,\theta_2)\right)^2+\\&\sum_{i=1}^{n}\left(\sum_{j=1}^{H}v_j^2\sigma(w_j^2t_i+u_j^2)+t_i\sum_{j=1}^{H}v_j^2w_j^2\sigma'(w_j^2t_i+u_j^2)-f_{C^*}(t_i,\theta_1,\theta_2)\right)^2.\end{aligned}$$

The error computation not only involves the outputs, but also the derivatives of the network output with respect to its inputs. Therefore, in order to apply an appropriate minimization algorithm to the problem (4.5), we need to evaluate the gradient of the error. Firstly, lest us denote by

$$A_i(t_i,\theta_1,\theta_2)=\frac{dS^*}{dt}(t_i,\theta_1)-f_{S^*}(t_i,\theta_1,\theta_2),$$
$$B_i(t_i,\theta_1,\theta_2)=\frac{dC^*}{dt}(t_i,\theta_2)-f_{C^*}(t_i,\theta_1,\theta_2).$$

In the following, we evaluate the derivatives of the error with respect to all of its inputs.

$$\frac{\partial E}{\partial w_j^1}(\theta_1,\theta_2) = 2\sum_{i=1}^{n} A_i(t_i,\theta_1,\theta_2)\left[t_i v_j^1\left(2\sigma'(w_j^1 t_i + u_j^1) + w_j^1 t_i \sigma''(w_j^1 t_i + u_j^1)\right)\right.$$
$$\left. -\frac{\partial f_{S^*}}{\partial w_j^1}(t_i,\theta_1,\theta_2)\right] - 2\sum_{i=1}^{n} B_i(t_i,\theta_1,\theta_2)\frac{\partial f_{C^*}}{\partial w_j^1}(t_i,\theta_1,\theta_2)$$

$$\frac{\partial E}{\partial w_j^2}(\theta_1,\theta_2) = 2\sum_{i=1}^{n} B_i(t_i,\theta_1,\theta_2)\left[t_i v_j^2\left(2\sigma'(w_j^2 t_i + u_j^2) + w_j^2 t_i \sigma''(w_j^2 t_i + u_j^2)\right)\right.$$
$$\left. -\frac{\partial f_{C^*}}{\partial w_j^2}(t_i,\theta_1,\theta_2)\right] - 2\sum_{i=1}^{n} A_i(t_i,\theta_1,\theta_2)\frac{\partial f_{S^*}}{\partial w_j^2}(t_i,\theta_1,\theta_2)$$

$$\frac{\partial E}{\partial u_j^1}(\theta_1,\theta_2) = 2\sum_{i=1}^{n} A_i(t_i,\theta_1,\theta_2)\left[v_j^1\left(\sigma'(w_j^1 t_i + u_j^1) + w_j^1 t_i \sigma''(w_j^1 t_i + u_j^1)\right)\right.$$
$$\left. -\frac{\partial f_{S^*}}{\partial u_j^1}(t_i,\theta_1,\theta_2)\right] - 2\sum_{i=1}^{n} B_i(t_i,\theta_1,\theta_2)\frac{\partial f_{C^*}}{\partial u_j^1}(t_i,\theta_1,\theta_2)$$

$$\frac{\partial E}{\partial u_j^2}(\theta_1,\theta_2) = 2\sum_{i=1}^{n} B_i(t_i,\theta_1,\theta_2)\left[v_j^2\left(\sigma'(w_j^2 t_i + u_j^2) + w_j^2 t_i \sigma''(w_j^2 t_i + u_j^2)\right)\right.$$
$$\left. -\frac{\partial f_{C^*}}{\partial u_j^2}(t_i,\theta_1,\theta_2)\right] - 2\sum_{i=1}^{n} A_i(t_i,\theta_1,\theta_2)\frac{\partial f_{S^*}}{\partial u_j^2}(t_i,\theta_1,\theta_2)$$

$$\frac{\partial E}{\partial v_j^1}(\theta_1,\theta_2) = 2\sum_{i=1}^{n} A_i(t_i,\theta_1,\theta_2)(c + w_j^1 t_i \sigma'(w_j^1 t_i + u_j^1)$$
$$-\frac{\partial f_{S^*}}{\partial v_j^1}(t_i,\theta_1,\theta_2)) - 2\sum_{i=1}^{n} B_i(t_i,\theta_1,\theta_2)\frac{\partial f_{C^*}}{\partial v_j^1}(t_i,\theta_1,\theta_2)$$

$$\frac{\partial E}{\partial v_j^2}(\theta_1,\theta_2) = 2\sum_{i=1}^{n} B_i(t_i,\theta_1,\theta_2)(\sigma(w_j^2 t_i + u_j^2) + w_j^2 t_i \sigma'(w_j^2 t_i + u_j^2)$$
$$-\frac{\partial f_{C^*}}{\partial v_j^2}(t_i,\theta_1,\theta_2)) - 2\sum_{i=1}^{n} A_i(t_i,\theta_1,\theta_2)\frac{\partial f_{S^*}}{\partial v_j^2}(t_i,\theta_1,\theta_2)$$

In the above formulae, we have used the following derivatives:

$$\frac{\partial f_{S^*}}{\partial w_j^1}(t_i,\theta_1,\theta_2) = -k_1(E_0 + \hat{C} - C^*(t_i,\theta_2))t_i^2 v_j^1 \sigma'(w_j^1 t_i + u_j^1)$$
$$\frac{\partial f_{C^*}}{\partial w_j^1}(t_i,\theta_1,\theta_2) = k_1(E_0 + \hat{C} - C^*(t_i,\theta_2))t_i^2 v_j^1 \sigma'(w_j^1 t_i + u_j^1)$$

$$\frac{\partial f_{S^*}}{\partial u_j^1}(t_i, \theta_1, \theta_2) = -k_1(E_0 + \hat{C} - C^*(t_i, \theta_2))t_i v_j^1 \sigma'(w_j^1 t_i + u_j^1)$$
$$\frac{\partial f_{C^*}}{\partial u_j^1}(t_i, \theta_1, \theta_2) = k_1(E_0 + \hat{C} - C^*(t_i, \theta_2))t_i v_j^1 \sigma'(w_j^1 t_i + u_j^1)$$

$$\frac{\partial f_{S^*}}{\partial v_j^1}(t_i, \theta_1, \theta_2) = -k_1(E_0 + \hat{C} - C^*(t_i, \theta_2))t_i \sigma(w_j^1 t_i + u_j^1)$$
$$\frac{\partial f_{C^*}}{\partial v_j^1}(t_i, \theta_1, \theta_2) = k_1(E_0 + \hat{C} - C^*(t_i, \theta_2))t_i \sigma(w_j^1 t_i + u_j^1)$$

$$\frac{\partial f_{S^*}}{\partial w_j^2}(t_i, \theta_1, \theta_2) = (k_1 S^*(t_i, \theta_1) + k_{-1})t_i^2 v_j^2 \sigma'(w_j^2 t_i + u_j^2)$$
$$\frac{\partial f_{C^*}}{\partial w_j^2}(t_i, \theta_1, \theta_2) = -(k_1 S^*(t_i, \theta_1) + k_{-1})t_i^2 v_j^2 \sigma'(w_j^2 t_i + u_j^2)$$

$$\frac{\partial f_{S^*}}{\partial u_j^2}(t_i, \theta_1, \theta_2) = (k_1 S^*(t_i, \theta_1) + k_{-1})t_i v_j^2 \sigma'(w_j^2 t_i + u_j^2)$$
$$\frac{\partial f_{C^*}}{\partial u_j^2}(t_i, \theta_1, \theta_2) = -(k_1 S^*(t_i, \theta_1) + k_{-1})t_i v_j^2 \sigma'(w_j^2 t_i + u_j^2)$$

$$\frac{\partial f_{S^*}}{\partial v_j^2}(t_i, \theta_1, \theta_2) = (k_1 S^*(t_i, \theta_1) + k_{-1})t_i \sigma(w_j^2 t_i + u_j^2)$$
$$\frac{\partial f_{C^*}}{\partial u_j^2}(t_i, \theta_1, \theta_2) = -(k_1 S^*(t_i, \theta_1) + k_{-1})t_i \sigma(w_j^2 t_i + u_j^2)$$

In our numerical simulation, we have used the Levenberg-Marquardt algorithm (LMA) to minimize the error function, by taking into account the computed gradient. This algorithm combines two important optimization techniques: the Gauss-Newton algorithm and the gradient descent method. The conceptual algorithm is as follows:

STEP 0:

- set the initial conditions, the model parameters, and the number of epochs;
- generate a set of random training points $t_i \in [0,\, T],\ i = 1,..,n$, and two sets of adjustable parameters, θ_1 and θ_2, uniform randomly chosen, for both neural networks N_1 and N_2;

STEP 1:

- for each epoch:
  - using the current values of hyper-parameters θ_1 and θ_2, compute the outputs N_1 and N_2, the trial solution

    $$(S^*, C^*) = (S^*(t_i, \theta_1), C^*(t_i, \theta_2))$$

    of system (4.1), the error function by (4.6) and the corresponding gradient;
  - apply LMA and find a local minimum for (4.6);
  - update θ_1 and θ_2;

Step 2:

- for any set of test points and using the optimal values of the parameters θ_1 and θ_2 obtained from STEP 2, approximate the solution of system (4.1) by (4.3).

When the time interval $[0, T]$ is too large, e.g. $T \gg 1$, the accumulation of errors leads to a weaker accuracy of the approximated solution (see [17]). In order to improve the accuracy on the whole time domain, we propose a domain decomposition technique as follows: the time interval $[0, T]$ is decomposed into N sub-intervals, such that $[0, T] = \bigcup_{k=1}^{N} [t_k, t_{k+1}]$, with $t_k = \frac{(k-1)T}{N}$ and $t_{k+1} = \frac{kT}{N},\ k = 1, \ldots, N$.
Then, the problem is solved for each time subinterval $[t_k,\ t_{k+1}],\ k = 1, \ldots, N$, using the already described method. For each sub-interval, the initial condition is set to be the approximate solution at the final point in the previous sub-domain. This domain decomposition technique leads to a consistent improved accuracy.

5 Numerical experiments

In this section, we perform numerical experiments using the proposed numerical scheme. In all experiments we have taken $n = 10$ randomly chosen training points inside each subinterval $[t_k,\ t_{k+1}],\ k = 1, \ldots, N$, and $H = 4$ neurons on the hidden layer. All simulations are performed using Matlab software ([7]). The number of split subintervals N varies from experiment to experiment, as needed.

5.1 Experiment 1

In this experiment we wanted to compare the proposed ANN scheme with the classic Runge-Kutta scheme. After the split of the whole time interval into 9 equal subintervals, we have computed for each subinterval the approximate solutions obtained via both methods. At the end of the algorithm, we glued the corresponding solutions at the extremities of these subintervals, such that the solution on each subinterval evaluated at the end-point becomes the initial value for the solution in the next interval. At the end, the two solutions (ANN and R-K) are plotted together in the same picture (see Figure 5). Here we have considered: $E_0 = 2$, $S_0 = 2$, $k_1 = 3$, $k_2 = 1$, $k_{-1} = 1$, $I_0 = 1$, $\omega = \frac{\pi}{5}$, $N = T = 9$, $\varepsilon = 0.1$.

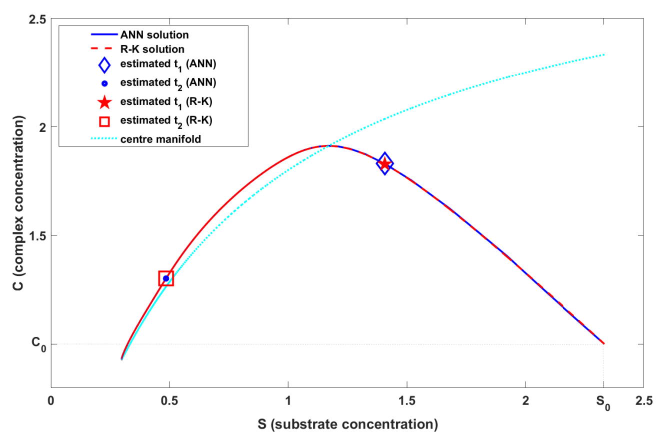

Fig. 5 Dynamics of the system approximated by the ANN method (blue curve) and the Runge-Kutta method (red curve). Here we have used the following parameters: $E_0 = 2$, $S_0 = 2$, $k_1 = 3$, $k_2 = 1$, $k_{-1} = 1$, $I_0 = 1$, $\omega = \frac{\pi}{5}$, $T = 9$, $\varepsilon = 0.1$

As one can observe from the figure, the two solutions are very close to each other. Basically, they overlap on each other on the entire length of the interval $[0, T]$. As also stated before, the ANN solution is analytical, written in a closed form, and can be used to predict values for the solution functions even outside the interval used in simulation, but not too far.

For this choice of initial data and parameters for the system (3.5), the steady state assumption is not valid. As the end-point of the simulation interval (T) is not too large, the solution has not reached the third phase of the dynamics, that is the periodic solution phase. However, we have computed numerically the estimated timescales for the first two stages, namely t_1 and t_2, via both numerical methods. One can see that we obtain the same values for the estimated timescales, regardless the numerical method used.

5.2 Experiment 2

In the second experiment we have simulated the dynamics of the system (3.5) over the interval [0, 75] using the ANN method. The parameters used in this simulation are: $E_0 = 2$, $S_0 = 10$, $k_1 = 3$, $k_2 = 1$, $k_{-1} = 1$, $I_0 = 1$, $\omega = 0.1\,\pi$, $\varepsilon = 1$ and $N = T = 75$. As we have expected and also represented in Figure 6, all three stages of the dynamics are reached by the solution. Here, we have also estimated the timescales for the transient and the quasi-steady-state. The limit cycle has a period of $20\,\pi$, the same as the initial input oscillations.

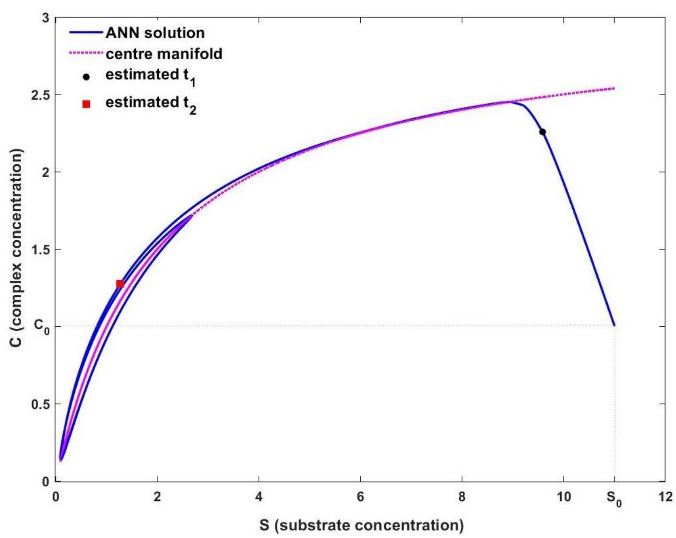

Fig. 6 Dynamics of the system approximated by the ANN method. Here we have used the following parameters: $E_0 = 2$, $S_0 = 10$, $k_1 = 3$, $k_2 = 1$, $k_{-1} = 1$, $I_0 = 1$, $\omega = 0.1\,\pi$, $T = 75$, $\varepsilon = 1$

5.3 Experiment 3

In this experiment we have approximated four different shapes of limit cycles that are obtained by simulating the solutions of the system (3.5) via the ANN method, and then plotted them in Figure 7. As one can observe, these limit cycles come in various shapes and sizes, depending on the set of parameters and the initial data. Inside each limit cycle we have represented the corresponding steady state $(\hat{S}, \hat{C})$.

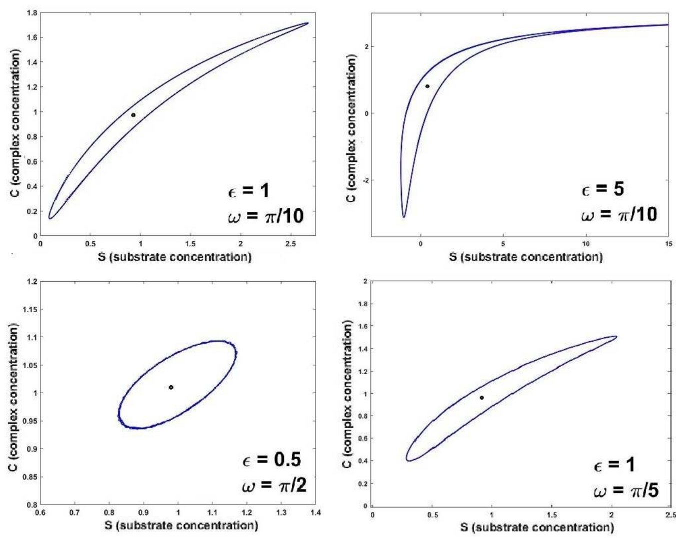

Fig. 7 The limit cycles and the steady state of the system approximated by the ANN method for various values of ε and ω. In all pictures, $I_0 = 1,\ k_1 = k_2 = k_{-1} = 1$.

5.4 Experiment 4

In this experiment we use the graphical representation of the hyper-parameters $\theta_1 = (w^1,\ u^1,\ v^1)$ and $\theta_2 = (w^2,\ u^2,\ v^2)$ for both ANN's, to show their convergence to the corresponding optimal values. As mentioned in the introduction, there is no analytical method to prove theoretically the convergence of the hyper-parameters, as the entire optimization process takes place inside the black box of the neural network.

We have considered here $N = T = 1$, a number of 10 randomly chosen training points inside the interval [0, 1], and 4 neurons on the hidden layer. Figure 8 shows the convergence of the computed values for the weight parameters $w_h^1,\ u_h^1$ and $v_h^1\quad (h = 1, \ldots, 4)$ in just a few epochs. A similar figure is obtained for the weight parameters $w_h^2,\ u_h^2$ and $v_h^2\quad (h = 1, \ldots, 4)$, which has not been included here. The dynamics of the approximated solutions (obtained with both, the ANN method (blue curve) and the Runge-Kutta method (red curve)) are drawn in Figure 9. As one can see, these two solutions a very close to each other. For this choice of T, the solutions have just reached the quasi-steady-state.

6 Conclusion

In this paper we have reviewed the most important models for the classic biochemical reaction presented in Figures 1 and 3. In Figure 1 the bio-

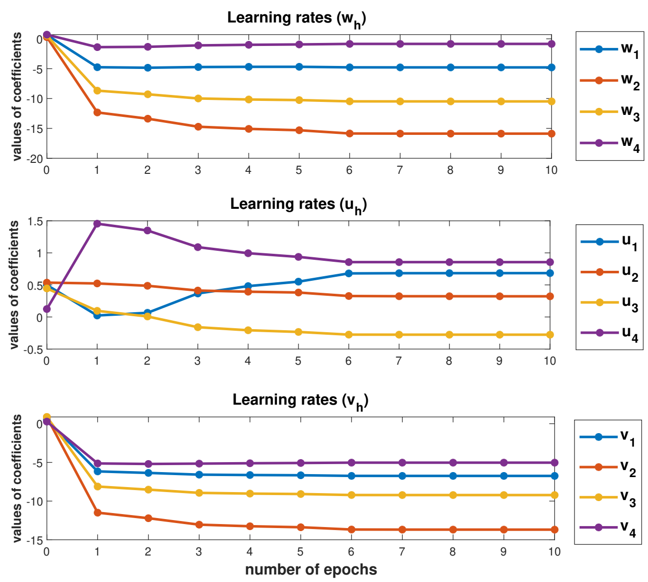

Fig. 8 Convergence of computed values for the hyper-parameters as the number of epochs increases. Here, we have used the following parameters: $E_0 = 2$, $S_0 = 10$, $k_1 = 1$, $k_2 = 1$, $k_{-1} = 1$, $I_0 = 1$, $\omega = \frac{\pi}{10}$, $T = 1$, $\varepsilon = 1$.

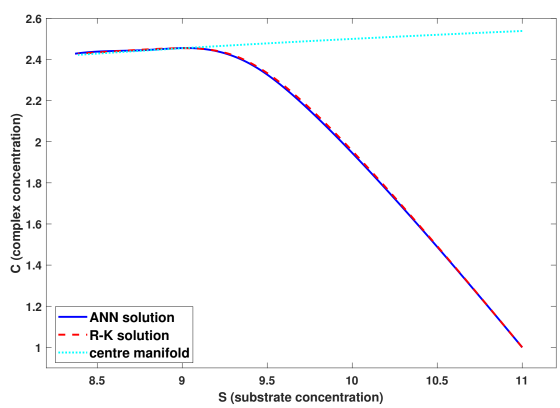

Fig. 9 Dynamics of the system approximated by the ANN method (blue curve) and the Runge-Kutta method (red curve). Here we have used the following parameters: $E_0 = 2$, $S_0 = 10$, $k_1 = 1$, $k_2 = 1$, $k_{-1} = 1$, $I_0 = 1$, $\omega = \frac{\pi}{10}$, $T = 1$, $\varepsilon = 1$.

chemical reaction is considered *in vitro*, while in Figure 3 the biochemical reaction is considered to be *in vivo*, that is as a part of a large biochemical pathway. The mathematical models presented here are systems of differential equations derived via either the mass action law, the Michaelis-Menten formalism, or the Briggs-Haldane approach. We have commented here on the validity of the assumptions corresponding to each of the models. As these equations are nonlinear and difficult to solve analytically, numerical methods are necessary to describe the dynamics of the solutions. There are various numerical methods that one can chose, such as the classic Runge-Kutta methods, predictor-corrector methods, finite element methods and many others. The accuracy of the solution in all these conventional methods depends on the dimension of the grid. In all these methods, one must take a coarser grid in order to speed up the computer simulation. Also, the numerical solutions obtained by those schemes are typically stored in a discretized form, that is, in an array of floating point numbers.

We proposed here a method based on the artificial neural networks, which is independent of a grid choice for the time domain. This method could be very useful especially when the domain dimension is larger than 1. The method combines artificial neural networks (ANN) and optimization techniques. The numerical solution is obtained fast and with less computational effort and limitations. Using the ANN method, the solution is stored as a set of hyper-parameters, which requires much less memory than storing the solution as a discretized array.

Our numerical simulations show that proposed model solution acts as a good interpolation technique as well as a good extrapolation method for test points that are situated close enough to the time interval used in training. This is due to the fact that the analytically differentiable form of the approximate solution could be used in predicting the behaviour of the system in any set of test points.

We are currently working to extend the ANN approach to model enzyme-catalysed reactions with diffusion, as the biochemical reactions take place in a three dimensional space. We also intend to incorporate some random factors into the model, for a better grasp of the reality.

References

1. Bisswanger, H. – *Enzyme Kinetics Principles and Methods*, Wiley-VCH Verlag GmbH, Weinheim (2002).
2. Borghans, J.A.M.; de Boer, R.J.; Segel, L.A. – *Extending the quasi-steady state approximation by changing variables*, Bull. Math. Biol. **58** (1996), no. 1, 43-63.
3. Briggs, G.E.; Haldane, J.B.S. – *A note on the kinetics of enzyme action*, Biochem. J. **19** (1925), no. 2, 338-339.
4. Dissanayake, M.W.M.G.; Phan-Thien, N. – *Neural-network-based approximations for solving partial differential equations*, Comm. Numer. Meth. Eng. **10** (1994), no. 3, 195-201.
5. Hornick, K.; Stinchcombe, M.; White, H. – *Multilayer feedforward networks are universal approximators*, Neural Netw. **2** (1989), no. 5, 359-366.

6. Lagaris, I.E.; Likas, A.; Fotiadis, D.I. – *Artificial neural networks for solving ordinary and partial differential equations*, IEEE Trans. Neural. Netw. **9** (1998), no. 5, 987-1000.
7. The Math Works – *Inc. MATLAB*. Version 2017b, The Math Works, Inc., Computer Software. www.mathworks.com (2017).
8. Malek, A.; Shekari Beidokhti, R. – *Numerical solution for high order differential equations using a hybrid neural networkoptimization method*, Appl. Math. Comput. **183** (2006), no. 1, 260-271.
9. Michaelis, L.; Menten, M.L. – *Die Kinetik der Invertinwirkung*, Biochem. Z. **49** (1913), 333-369.
10. Mitchell, T.M. – *Machine Learning*, McGraw-Hill, Inc., USA (1997).
11. Schnell, S.; Mendoza, C. – *Closed form solution for time-dependent enzyme kinetics*, J. Theor. Biol. **187** (1997), no. 2, 207-212.
12. Segel, L.A. – *On the validity of the steady state assumption of enzyme kinetics*, Bull. Math. Biol. **50** (1988), no. 6, 579-593.
13. Segel, L.A.; Slemrod, M. – *The quasi-steady-state assumption: a case study in perturbation*, SIAM Rev. **31** (1989), no. 3, 446-477.
14. Stoleriu, I.; Davidson, F.A.; Liu, J.L. – *Asymptotic approximations for Michaelis-Menten type kinetics for enzyme-catalysed reactions with constant input*, Tech. Rep., University of Dundee (2002), 1-17.
15. Stoleriu, I.; Davidson, F.A.; Liu, J.L. – *Quasi-steady state assumptions for non-isolated enzyme-catalysed reactions*, J. Math. Biol. **48** (2004), no. 1, 82-104.
16. Stoleriu I.; Davidson F.A.; Liu J.L. – *Effects of periodic input on the quasi-steady state assumptions for enzyme-catalysed reactions*, J. Math. Biol. **50** (2005), no. 2, 115-132.
17. Toselli, A.; Widlund, O.B. – *Domain decomposition methods - algorithms and theory*, Springer Series in Computational Mathematics, Springer-Verlag, Berlin **34** (2005).

Received: 2.XI.2020 / Accepted: 19.XI.2020

Authors

Ana-Maria Moşneagu,
Faculty of Mathematics,
"Alexandru Ioan Cuza" University of Iaşi,
Bvd. Carol I, No. 11,
700506, Iaşi, România,
E-mail: anamaria.mosneagu@uaic.ro

Iulian Stoleriu,
Faculty of Mathematics,
"Alexandru Ioan Cuza" University of Iaşi,
Bvd. Carol I, No. 11,
700506, Iaşi, România,
E-mail: iulian.stoleriu@uaic.ro
Exact Sciences Department,
"Ion Ionescu de la Brad" University of
Agricultural Sciences and Veterinary Medicine,
Faculty of Horticulture,
str. Aleea M. Sadoveanu, No. 3,
700490, Iaşi, România,
E-mail: iulian.stoleriu@uaiasi.ro