Nonlinear Analysis: Modelling and Control, Vol. 29, No. 1, 53–70
https://doi.org/10.15388/namc.2024.29.33604

Vilnius University Press

# Applying artificial neural networks to solve the inverse problem of evaluating concentrations in multianalyte mixtures from biosensor signals

**Ignas Dapšys**[1], **Raimondas Čiegis**, **Vadimas Starikovičius**

Department of Mathematical Modelling,
Faculty of Fundamental Sciences,
Vilnius Gediminas Technical University,
Saulėtekio ave. 11, LT-10223 Vilnius, Lithuania
ignas.dapsys@vilniustech.lt; raimondas.ciegis@vilniustech.lt;
vadimas.starikovicius@vilniustech.lt

**Received:** April 24, 2023 / **Revised:** October 24, 2023 / **Published online:** November 11, 2023

**Abstract.** We investigate the ill-posedness of the inverse biosensor problem when the biosensor signals are corrupted by noise. To solve the problem, we employ feed-forward and convolutional neural networks. Computational experiments were performed with different levels of additive and multiplicative noises for the batch and flow injection analysis modes of the biosensor. Obtained results show that the largest errors of recovered concentrations are located on the edges of the training domain. We have found that the inverse problem is less ill-posed in the flow injection analysis mode and concentrations can be reliably recovered for higher levels of noise compared to the batch mode. This finding is confirmed by the application of the DIRECT global optimization method to the considered inverse biosensor problem.

**Keywords:** biosensor, artificial neural network, mathematical modelling, inverse problem, ill-posed problem, noise.

## 1 Introduction

Biosensors are devices for the detection and analysis of chemical compounds based on biochemical processes [3]. One of the most widespread biosensor types are enzyme-based amperometric biosensors [23], where the analytes in a mixture are enzymatically converted into products, which produce an electric current that is measured. In order to use this signal to determine the concentrations of the sample analytes, we need to establish a dependence between concentrations of analytes and electric current using known samples. Unknown samples can then be analyzed by measuring their current signal

[1]Corresponding author.

and, by using the inverse of the dependence, determining their composition, i.e., solving an inverse biosensor problem. For a single substrate, this can be achieved very easily, but for multiple substrates, its solution is complicated by the ill-posedness of the problem [33]: if the signals in question are similar enough to be practically identical or have small perturbations such as when the biosensor response is under the effect of noise, their concentrations may be vastly different. This can negatively impact the precision of the device [5].

One promising method to alleviate the ill-posedness of such problems is to use neural networks, which have successfully been used to solve various inverse problems such as determining the initial condition of the heat equation from the temperature profile [27], finding the boundary condition [9, 19], or the heat source [11]. An example of an inverse problem – the determination of parameters for the reaction-diffusion equation – is given in [22]. A review of neural networks for solving linear inverse problems is given in [16].

We have found a large amount of relevant biosensor literature [6, 7, 32]. Of particular interest are papers that employ neural networks to analyze various chemical mixtures. Applications of simple feedforward neural networks can be found in [17, 18, 29]. Another popular, more sophisticated architecture is the convolutional neural network (CNN), which is widely used for computer vision applications. Such a network has convolutional layers, comprised of filters of various sizes, which capture localized patterns in data. Convolutional networks have been applied for analysis of 1-dimensional spectroscopic data [24]. Such networks are also used for biosensors, examples are given in [1,10,15]. An advantage of convolutional neural networks is that such an architecture can remove noise from input data [25]. Therefore, we have performed experiments with this architecture for noisy signals in this paper.

Note that not all of these biosensors are based on enzymes. However, our goal is the development of data analysis methods, and they can be applied to other biosensor types. We chose amperometric biosensors since they have well established mathematical models [3].

Almost all of featured biosensor literature involves using physical prototypes. This approach has the disadvantage that such devices are expensive and time consuming to produce. Therefore, we think it is feasible to conduct our experiments on a virtual biosensor model, which allows for a larger output of data without the use of physical devices. We intend to apply our methods to real data in the future.

Some biosensor papers deal with mathematical modelling. Žilinskas et al. [33] studied the biosensor problem in detail, formulated as an optimization problem. They found that it is ill-posed, as there are biosensor signals that are almost identical for different concentration values, and suggest using global optimization. This approach works best for a small amount of samples as a large amount would take a long time to compute. The inverse biosensor problem has a lot of parameters for global optimization, which also greatly increases the computational cost. However, if we use a large amount of parameters, we can afford to optimize them to be good enough to achieve our goals, they do not have to be globally optimal, and this can be achieved much faster by neural networks. In our paper, we compared the performance of the parallel DIRECT method for global optimization to that of neural networks. Baronas et al. [5] tested biosensor

performance for noisy signals, when the response was analyzed using local optimization methods, for two analytes. The biosensor had good precision if the noise level is not too large. But for three analytes, a small noise level can lead to large errors, and local optimization is still time consuming, when we have a lot of responses to analyze. Litvinas and Baronas [20] used artificial neural networks to analyze biosensor responses for four analytes. The authors only used a simple feedforward network architecture, which had a small relative prediction error, but effects of noise and alternative network architectures were untested.

Various other biosensor models can be considered such as microfluidic biosensors [17] and micro-disk biosensors [18]. A biosensor model, which uses fractional power elliptic operators to model diffusion, is given in [8].

This paper is organized as follows: in Section 2, we describe mathematical models of biosensors and their electric current signals, which are also given in [3] as well as the biosensor analysis modes: batch and flow injection analysis. We restate the inverse biosensor problem in terms of this model. We also discuss the biosensor control modes, which are used to determine which process is the bottleneck in the biosensor. Section 3 describes the methods used for the solution of the inverse biosensor problem. We use feedforward and convolutional neural networks. Next, we define two models of noise using a Gaussian random variable. The setup of computational experiments is presented in Section 4. Here we discuss the generation of biosensor datasets for neural network training and testing, the parameters used in the biosensor model, the parameters of employed neural networks. In Section 5, we present and analyze the results of computational experiments. And in Section 6 the final conclusions are given.

## 2 Problem statement

### 2.1 Biosensor model

The chemical process we are modelling is called enzymatic substrate conversion, where an enzyme converts $m$ substrates into products described by the chemical equation

$$\mathrm{S}_i \xrightarrow{\mathrm{E}} \mathrm{P}_i.$$

Substrate concentrations are denoted by $S_i$, and product concentrations – by $P_i$, $i = \overline{1,m}$. Relevant substrate parameters are $V_i$ – the maximal reaction rate for every substrate and $K_i$ – the Michaelis–Menten constant.

The biosensor is comprised of three elements: the enzyme layer, the outer diffusion layer, and the electrode. The outer diffusion layer brings the substrate from the bulk solution into the enzyme layer, where the enzymatic reaction takes place. Reaction products generate a current signal, which is picked up by the electrode. By making a few assumptions such as having a symmetric electrode and a uniformly distributed enzyme with a uniform layer thickness $d$, we may ignore diffusion in other dimensions and use a 1-dimensional model.

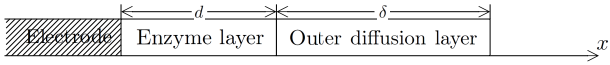


**Figure 1.** Schematic layout of a biosensor.

The model equations are based on the law of chemical reaction kinetics and the Fick's law of diffusion. Enzymatic layer equations are as follows:

$$\frac{\partial S_{i,e}}{\partial t} = D_{S_{i,e}} \frac{\partial^2 S_{i,e}}{\partial x^2} - \frac{(V_i/K_i) S_{i,e}}{1 + \sum_{j=1}^{m} S_{j,e}/K_j},$$

$$\frac{\partial P_{i,e}}{\partial t} = D_{P_{i,e}} \frac{\partial^2 P_{i,e}}{\partial x^2} + \frac{(V_i/K_i) S_{i,e}}{1 + \sum_{j=1}^{m} S_{j,e}/K_j}, \quad i = \overline{1,m},\ x \in (0,d),\ t > 0,$$

where $S_{i,e}$ and $P_{i,e}$ are the substrate and product concentrations in the enzyme layer, respectively. $D_{S_{i,e}}$ and $D_{P_{i,e}}$ – their respective coefficients of diffusion, $d$ is the enzyme layer thickness.

To make computations easier, we scale the substrate and product concentrations by their respective Michaelis–Menten constants: $\widetilde{S}_i = S_i/K_i$, $\widetilde{P}_i = P_i/K_i$, $i = \overline{1,m}$. This is done for all layers. Enzymatic layer equations now look as follows:

$$\frac{\partial \widetilde{S}_{i,e}}{\partial t} = D_{S_{i,e}} \frac{\partial^2 \widetilde{S}_{i,e}}{\partial x^2} - \frac{(V_i/K_i) \widetilde{S}_{i,e}}{1 + \sum_{j=1}^{m} \widetilde{S}_{j,e}}, \tag{1}$$

$$\frac{\partial \widetilde{P}_{i,e}}{\partial t} = D_{P_{i,e}} \frac{\partial^2 \widetilde{P}_{i,e}}{\partial x^2} + \frac{(V_i/K_i) \widetilde{S}_{i,e}}{1 + \sum_{j=1}^{m} \widetilde{S}_{j,e}}, \quad i = \overline{1,m},\ x \in (0,d),\ t > 0. \tag{2}$$

Diffusion layer equations are as follows:

$$\frac{\partial \widetilde{S}_{i,b}}{\partial t} = D_{S_{i,b}} \frac{\partial^2 \widetilde{S}_{i,b}}{\partial x^2}, \tag{3}$$

$$\frac{\partial \widetilde{P}_{i,b}}{\partial t} = D_{P_{i,b}} \frac{\partial^2 \widetilde{P}_{i,b}}{\partial x^2}, \quad i = \overline{1,m},\ x \in (d,\ d+\delta),\ t > 0, \tag{4}$$

where $\widetilde{S}_{i,b}$, $\widetilde{P}_{i,b}$, $D_{S_{i,b}}$, and $D_{P_{i,b}}$ are the substrate and product concentrations and coefficients of diffusion in the diffusion layer, $\delta$ is its thickness. The Nernst outer diffusion layer model is used, where $\delta$ is constant. These equations are well known – their description can be found in [13,26], as well as in a book by Kulys et al. [3], where the authors provide examples of investigation of biosensor properties using such models.

Next, we have the initial model conditions. First of all, there are no substrates or products in any of the layers:

$$\widetilde{S}_{i,e}(x,0) = 0, \quad \widetilde{P}_{i,e}(x,0) = 0, \quad x \in [0,d], \tag{5}$$

$$\widetilde{S}_{i,b}(x,0) = 0, \quad \widetilde{P}_{i,b}(x,0) = 0, \quad x \in [d,\ d+\delta),\ i = \overline{1,m}. \tag{6}$$

Boundary conditions are given as follows. There are no products on the surface of the electrode ($x = 0$) since they react with it and disappear from the biosensor. Also, substrates cannot flow through the electrode:

$$\widetilde{P}_{i,e}(0,t) = 0, \qquad D_{S_{i,e}} \frac{\partial \widetilde{S}_{i,e}}{\partial x}\bigg|_{x=0} = 0. \tag{7}$$

There are no products in the bulk solution:

$$\widetilde{P}_{i,b}(d+\delta,t) = 0, \quad i = \overline{1,m}. \tag{8}$$

The substrate and product concentrations must match on the inner boundary between layers as well as their fluxes:

$$D_{S_{i,e}} \frac{\partial \widetilde{S}_{i,e}}{\partial x}\bigg|_{x=d} = D_{S_{i,b}} \frac{\partial \widetilde{S}_{i,b}}{\partial x}\bigg|_{x=d}, \quad \widetilde{S}_{i,e}(d,t) = \widetilde{S}_{i,b}(d,t), \tag{9}$$

$$D_{P_{i,e}} \frac{\partial \widetilde{P}_{i,e}}{\partial x}\bigg|_{x=d} = D_{P_{i,b}} \frac{\partial \widetilde{P}_{i,b}}{\partial x}\bigg|_{x=d}, \quad \widetilde{P}_{i,e}(d,t) = \widetilde{P}_{i,b}(d,t), \quad i = \overline{1,m}. \tag{10}$$

Substrate concentrations outside of the biosensor depend on the analysis mode. In this paper, we compare the performance of two analysis modes: batch mode and flow injection analysis mode. For the batch mode, substrate concentrations remain constant for the duration of the experiment:

$$\widetilde{S}_{i,b}(d+\delta,t) = \widetilde{S}_{i,0}. \tag{11}$$

For the FIA mode, the substrates are in contact with the biosensor for a limited amount of time $t_f$, called the injection time, after which concentrations are reduced to 0:

$$\widetilde{S}_{i,b}(d+\delta,t) = \begin{cases} \widetilde{S}_{i,0}, & t \leqslant t_f, \\ 0, & t > t_f. \end{cases} \tag{12}$$

The biosensor response is an electric current signal, which depends on the composition of the sample solution. More precisely, the signal depends on the fluxes of all products on the surface of the electrode:

$$I(t) = \sum_{i=1}^{m} K_i n_i F D_{P_{i,e}} \frac{\partial \widetilde{P}_{i,e}}{\partial x}\bigg|_{x=0}, \tag{13}$$

where $n_i$ is the number of electrons carrying an electric charge on the electrode for each product, $F = 96485.33$ C/mol is the Faraday constant. We assume that we can only observe and measure the net current of all products – we have no information on separate currents for each product. See in Fig. 2 the examples of biosensor response – electric current signals $I(t)$ obtained by (13) for the mixture of three analytes considered in this work.

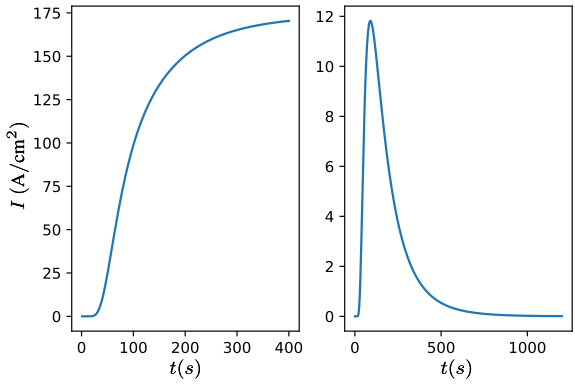


**Figure 2.** Examples of biosensor response – electrical signals $I(t)$ (13). Left – batch mode, right – flow injection analysis mode.

We can now formulate the inverse biosensor problem in terms of the model: given the biosensor signal $I(t)$, determine the initial substrate concentrations $\mathbf{c}_0 = (\widetilde{S}_{1,0}, \ldots, \widetilde{S}_{m,0})$. We solve the problem in the usual way by creating a model of the inverse dependence from known samples and their biosensor signals. We do this by changing model parameters to minimize the error between the known concentrations and the model predictions. This inverse problem is ill-posed, which, in terms of the model, means that if two signals $I_1$, $I_2$ satisfy $\|I_1 - I_2\|_\infty < \varepsilon$, which can occur if the signals are very similar or they have small perturbations $I_2 = I_1 + \varepsilon$, then $\|\mathbf{c}_0^{(1)} - \mathbf{c}_0^{(2)}\|_2$ may be large.

The process to reduce the ill-posedness of inverse problems is called regularization [2]. A well-known method for this is called Tikhonov regularization [28], where we add a penalty term, which performs solution smoothing and decreases the sensitivity of the problem to errors. Obtained solutions have similar properties to those of the unregularized problem. This method is widely used for neural networks as well – it is included in many neural network software packages. Here regularization is used to reduce the effect of irrelevant data features to network predictions. The penalty term may be the $L_1$ or the $L_2$ norm.

## 2.2 Biosensor control modes

Biosensor operation can be described in terms of the diffusion modulus, also known as the Damköhler number, which is the ratio between the reaction rate $V_i/K_i$ and the diffusion rate $D_{\widetilde{S}_{i,e}}/d^2$ in the enzyme layer:

$$\Phi_i^2 = \frac{V_i d^2}{K_i D_{S_{i,e}}}, \quad i = \overline{1,m}.$$

This parameter is used in [4, 5, 21] to determine the control mode of the biosensor. If the modulus is larger than 1 for all substrates, the biosensor is under diffusion control. If the modulus is less than 1, the biosensor is under reaction control. If the control mode is diffusion control for some substrates and reaction mode for the others, we have mixed

control. In other words, these modes can tell us which process is the bottleneck in the biosensor operation.

## 3 Methods used

In this paper, we investigate artificial neural networks [31] – a machine learning method, which models the functions of the human brain. A neural network is comprised of a series of layers, with each layer gradually processing the data by multiplying it by the layer weights and applying an activation function. Such a method requires training, which uses a backpropagation method to change the weights of the model, so that the error between the known and predicted values of the training data is minimized. Neural network training is influenced by the amount of data: larger datasets require more complex networks with more weights, which can lead to a smaller training error, but they take more time to train, and if the network is too complex, its performance on unseen testing data would be poor – this is called overfitting. To avoid these pitfalls, we need to design the network to be as simple as possible to maintain sufficient precision or reduce the amount of training data by, for example, dimensionality reduction.

We used two neural network architectures – feedforward and convolutional networks. These architectures are simple to implement using off-the-shelf software packages and consume less computing resources than more sophisticated neural networks. In feedforward neural networks, each layer is fully connected with the preceding and the following layer. These networks are the simplest, but they require preprocessing of the data to reduce dimensionality. Convolutional neural networks (CNN) learn a suitable nonlinear lower dimensional representation of data by themselves and reuse the same weights for different parts of data, thus improving generalization, but are more complex than feedforward networks. The main components of a CNN are convolutional and pooling layers. Convolutional layers, as their name implies, perform the convolution operation, which detects certain patterns in data to get a local representation by applying a sliding filter on a sample, which is learned during training. Pooling layers downsample their inputs and combine data from multiple convolution filters to get a more global representation by applying a sliding window, where the network computes an average or a maximum value of the elements in that window. Multiple pairs of these layers are arranged in sequence before a flattening layer followed by a fully connected layer. The inputs to the convolutional network are normalized full biosensor signals. Both architectures use the concentration values the signal represents as outputs. The loss function for training is the mean squared error. To train both types of networks, we used the Adam optimizer with batch learning. Neural networks were implemented in the Python programming language using the TensorFlow and Keras [12] libraries on the VILNIUS TECH Vanagas cluster with an Nvidia Tesla P100 GPU.

The dimensionality reduction method we use for feedforward neural networks is the principal component analysis (PCA), which transforms the raw data into principal components (PCs) – they contain the most informative features of the dataset. This method has been used in previous papers (see [4, 20]). In our research, we found that performing data normalization before PCA yielded better results.Let $\mathbf{X}$ be an $n \times p$ response matrix,

where $n$ is the amount of biosensor responses, and $p$ is the amount of time samples per response. Normalization is performed by

$$\mathbf{X}_{\text{norm}} = \frac{\mathbf{X} - J_{n,1}\mu}{J_{n,1}\mathbf{s}},$$

where $\mathbf{X}_{\text{norm}}$ is the normalized response matrix. $\mu$ and $\mathbf{s}$ are column-wise mean and standard deviation of $\mathbf{X}$, respectively (size $1 \times p$). $J_{n,1}$ is an $n \times 1$ matrix of ones. Division is performed element-wise.

To simulate the influence of noise, we use two different types: additive and multiplicative. The first type corresponds to electrical noise for small substrate concentrations compared to the Michaelis constant $K_i$, where there is no dependence on signal [14]:

$$I_{an}(t_i) = I(t_i) + \sigma X. \tag{14}$$

The second type is for larger concentrations when such dependence exists [14]:

$$I_{mn}(t_i) = I(t_i)(1 + \sigma X).$$

Here $X$ is a normally distributed random variable with mean of 0 and standard deviation of 1. $\sigma$ is the noise level. Noise was applied for both training and testing data and only one type for all biosensor responses at a time, regardless of concentration magnitude. This was done to simplify the testing process since, in order to find out at which concentration level the noise types change, we would need physical biosensor data. Still, the approach used in this paper is informative of the effects of noise on neural network precision.

### 3.1 Error measurement

The performance of the neural network was measured in terms of relative absolute percentage error given by

$$\frac{|a_i(I) - \widetilde{S}_{i,0}|}{|\widetilde{S}_{i,0}|} \cdot 100\%, \quad i = \overline{1,m}, \tag{15}$$

where $a_i(I)$ is the $i$th value of the neural network output when its inputs are either the principal components of response $I$ or the response itself, and $\widetilde{S}_{i,0}$ are the true boundary concentration values. We calculate mean and maximum errors over all concentrations ($i = \overline{1,m}$), so that each signal $I$ has mean and maximum error values. Then we compute the mean and maximum error values over all signals to yield mean and maximum error values for the entire testing dataset.

## 4 Experimental setup

### 4.1 Data generation

In this work, we model the mixture of 3 analytes. The scaled value $\widetilde{S}_{i,0} = S_{i,0}/K_i$ of initial concentration ranges from 3.2 to 12.8 for all analytes: $i = 1, 2, 3$. Such concentration

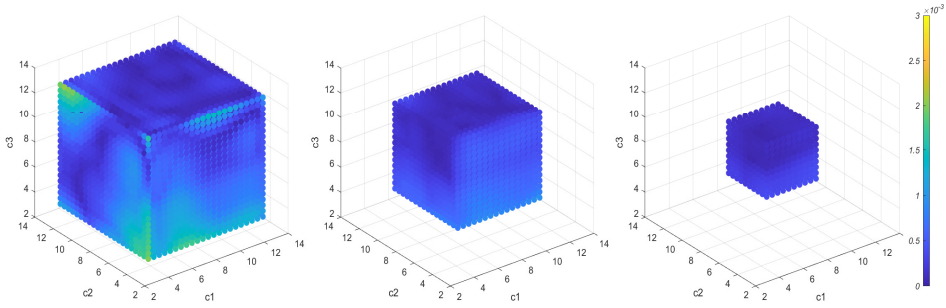


**Figure 3.** Initial concentration error distribution. Left – original mesh $D_{tr}$ (16), middle – 3 outer layers removed, right – 6 layers removed.

values are used in [5,21]. Training set concentrations are defined in a cube $[3.2, 12.8]^3$ by a 3-dimensional uniform discrete mesh $D_{tr}$:

$$D_{tr} = D \times D \times D, \quad D = \left\{ 3.2\left(1 - \frac{k}{N-1}\right) + 12.8\frac{k}{N-1},\ k = \overline{0, N-1} \right\}. \quad (16)$$

We use $N = 22$, which gives 10648 points in $D_{tr}$.

Each point $\mathbf{c}_0 = (\widetilde{S}_{1,0}, \widetilde{S}_{2,0}, \widetilde{S}_{3,0})$ in $D_{tr}$ is used in initial condition (11) for batch mode or (12) for FIA mode to numerically solve the mathematical biosensor model (1)–(12) and generate biosensor response $I(t)$ (13). We solve system (1)–(12) using the finite volume method. Nonlinear differential equations are linearized using the predictor-corrector method, and in each time layer, they are solved by factorizing the system of linear equations by tridiagonal matrix algorithm, also known as the Thomas algorithm.

To create a testing set, we generate 3000 uniformly distributed random points inside the cube $[3.2, 12.8]^3$:

$$D_{ts} = \left\{ \mathbf{c}_0^{(j)} = \left(\widetilde{S}_{1,0}^{(j)}, \widetilde{S}_{2,0}^{(j)}, \widetilde{S}_{3,0}^{(j)}\right):\ \widetilde{S}_{i,0}^{(j)} = 3.2(1 - \xi_i) + 12.8\xi_i,\ j = \overline{1, 3000} \right\},$$

where $\xi_i \sim \mathcal{U}(0, 1)$ is a uniformly distributed random variable. Then for each point $\mathbf{c}_0^{(j)}$ in $D_{ts}$, we solve problem (1)–(12) numerically as before.

After the first tests of neural networks, we have noticed that the largest errors of recovered initial concentrations are located close to the boundaries of the training domain $[3.2, 12.8]^3$. In Fig. 3, we show the distribution of errors on the mesh $D_{tr}$ (16) for the training dataset. Here every point is colored according to the largest of all errors (15) for the 3 concentrations. If we remove 3 outer mesh layers from all sides, we can see that the maximal errors are decreasing and the largest errors were on the edges of the original cube.

Following these results, we have decided to create another testing dataset for our study. We generate 3000 uniformly distributed random points inside the reduced cube $[5.6, 10.4]^3$:

$$D_{rc} = \left\{ \mathbf{c}_0^{(j)} = \left(\widetilde{S}_{1,0}^{(j)}, \widetilde{S}_{2,0}^{(j)}, \widetilde{S}_{3,0}^{(j)}\right):\ \widetilde{S}_{i,0}^{(j)} = 5.6(1 - \xi_i) + 10.4\xi_i,\ j = \overline{1, 3000} \right\},$$

**Table 1.** Constant model parameters.

| Parameter | Value | Parameter | Value | Parameter | Value |
|---|---|---|---|---|---|
| $D_{S_{i,e}}$ | $3 \cdot 10^{-6}$ | $D_{P_{i,b}}$ | $6 \cdot 10^{-6}$ | $n_i$ | 2 |
| $D_{P_{i,e}}$ | $3 \cdot 10^{-6}$ | $K_i$ | $10^{-4}$ | | |
| $D_{S_{i,b}}$ | $6 \cdot 10^{-6}$ | $\delta$ | 0.04 | | |

where $\xi_i \sim \mathcal{U}(0,1)$ is a uniformly distributed random variable. For each point $\mathbf{c}_0^{(j)}$ in $D_{rc}$, we solve problem (1)–(12) numerically as before.

In our numerical experiments, we used these parameter values of model (1)–(4): the enzyme layer thickness $d = 0.02$, and the maximal enzymatic rates $V_i = 5 \cdot 10^{i-9}$, $i = 1, 2, 3$. Other parameters are given in Table 1. Simulation time in batch mode is $T = 400$, timestep $\tau = 0.25$. Response current values $I(t_i)$ are sampled every second. The space interval contains $N = 1000$ points, 333 of which belong to the enzyme layer. In flow injection analysis mode, injection time is $t_f = 3$, and simulation time was increased to $T = 1200$.

Data generation code was written using the C++ language. The training set has 10468 responses, whereas both testing sets have 3000 responses each, for each of the biosensor analysis modes. In order to reduce the generation time, each dataset was generated in parallel using the master-slave paradigm, where each response was computed on the separate CPU core. On the VILNIUS TECH Vanagas cluster, the training dataset was generated in 2.5 min. with 20 cores.

### 4.2 Experiment description

In the first experiment, we investigated the effect of additive noise (14) on the biosensor precision for the increasing levels of noise – $\sigma$ using the feed-forward network. At first, we used 4 principal components (PCs) for as input on input layer and gradually increased the amount to 80. The neural network architecture and hyperparameters were chosen by the coordinate search method until the maximal testing error stopped decreasing.

We chose between 1 and 2 hidden layers and the amount of neurons in each one out of five different values: 18, 28, 32, 36, and 46. During our tests, we found out that a neural network with a single hidden layer of 32 neurons gave the best results. The hidden layer uses the tanh activation function, while the output layer with 3 neurons uses a linear activation function. We trained the network using the Adam optimizer with a batch size of 64 and a learning rate of $1.25 \cdot 10^{-4}$. The amount of training epochs varied depending on the noise level.

Further tests were conducted in the second experiment by employing 1D convolutional neural networks (CNN). In this work, we used an architecture consisting of 4 pairs of convolution – max pooling layers. The architecture and hyperparameters were picked by using the coordinate search process. We used five different CNN configurations given in Table 2. Between the fully connected layer with 36 neurons and the output layer of 3 neurons, we used the dropout layer to prevent overfitting – the dropout rate was 0.1. The activation function used in convolution and fully connected layers was the rectified linear unit (ReLU). The network was trained using the Adam optimizer with a batch size

**Table 2.** Convolutional neural network configurations. The layers columns indicate which convolution-pooling layer pairs had the specified parameters. The FC column is the amount of neurons in the fully connected layer.

| Kernels | Layers | Kernel size | Layers | Pool size | Layers | FC |
|---|---|---|---|---|---|---|
| | | | Network 1 | | | |
| 10 | 1, 2 | 4 | 4 | 2 | 1, 2, 3, 4 | 36 |
| 12 | 3, 4 | 6 | 3 | | | |
| | | 8 | 2 | | | |
| | | 10 | 1 | | | |
| | | | Network 2 | | | |
| 10 | 1 | 10 | 1, 2, 3, 4 | 2 | 1,2,3,4 | 18 |
| 12 | 2 | | | | | |
| 14 | 3 | | | | | |
| 16 | 4 | | | | | |
| | | | Network 3 | | | |
| 10 | 1, 2, 3, 4 | 6 | 1, 2, 3, 4 | 2 | 1, 2, 3, 4 | 36 |
| | | | Network 4 | | | |
| 8 | 3, 4 | 4 | 4 | 2 | 2, 3, 4 | 36 |
| 10 | 1, 2 | 5 | 1, 2, 3 | 3 | 1 | |
| | | | Network 5 | | | |
| 8 | 2, 3, 4 | 4 | 4 | 2 | 2, 3, 4 | 36 |
| 10 | 1 | 5 | 3 | 3 | 1 | |
| | | 6 | 1, 2 | | | |

of $64$ and a learning rate of $5 \cdot 10^{-4}$. We trained the network $3$ times and picked the one which had the lowest testing error.

Next, for the third experiment, we tested both the feedforward and the convolutional neural network architectures on signals corrupted by multiplicative noise.

## 5 Results and discussion

### 5.1 Effect of additive noise

The relative prediction errors for the first experiment are given in Tables 3 and 4. We denote mean errors by $E_{ds,a}$ and maximum errors by $E_{ds,m}$, where the index $ds$ determines the dataset used: $tr$ – training set, $ts$ – test set, $rc$ – test set inside the reduced cube. We also give the amount of epochs and principal components (PCs) for each noise level $\sigma$ tested.

Results for batch mode show that in the noiseless case, the maximal errors are low and that they are even lower in the reduced cube. For noisy signals, the errors are larger and they are slightly lower in the reduced domain. The difference is smaller for the average errors. However, when the noise level $\sigma$ is reaching $0.03$, while the errors do decrease in reduced cube, they remain large, so that in the worst case, the biosensor can no longer make meaningful predictions. The best results were obtained for $80$ principal components.

Using the flow injection mode, the results for noiseless signals show that the errors decrease compared to the batch mode case, whereas the errors for noisy signals are

**Table 3.** Prediction errors for batch mode.

| $\sigma$ | Epochs | PCs | $E_{ts,a}$ | $E_{ts,m}$ | $E_{rc,a}$ | $E_{rc,m}$ |
|---|---|---|---|---|---|---|
| 0 | 3200 | 4 | 0.078 | 1.899 | 0.038 | 0.214 |
| 0 | 3200 | 80 | 0.050 | 1.093 | 0.039 | 0.269 |
| 0.01 | 2000 | 4 | 16.89 | 154.69 | 7.77 | 50.15 |
| 0.01 | 3200 | 80 | 1.04 | 9.88 | 0.92 | 6.64 |
| 0.02 | 3200 | 80 | 1.87 | 18.50 | 1.69 | 12.16 |
| 0.03 | 3200 | 80 | 2.66 | 26.62 | 2.38 | 17.52 |
| 0.04 | 3200 | 80 | 3.36 | 35.72 | 3.09 | 22.17 |
| 0.05 | 3200 | 80 | 4.08 | 39.75 | 3.77 | 26.50 |
| 0.10 | 3000 | 4 | 16.88 | 156.21 | 7.82 | 50.35 |
| 0.10 | 1000 | 80 | 8.12 | 107.02 | 7.20 | 52.22 |
| 0.20 | 450 | 80 | 12.54 | 163.91 | 8.77 | 74.70 |
| 0.40 | 3000 | 4 | 16.95 | 155.54 | 7.90 | 49.74 |
| 0.40 | 477 | 80 | 15.24 | 207.84 | 9.59 | 77.95 |

**Table 4.** Prediction errors for the flow injection mode.

| $\sigma$ | Epochs | PCs | $E_{ts,a}$ | $E_{ts,m}$ | $E_{rc,a}$ | $E_{rc,m}$ |
|---|---|---|---|---|---|---|
| 0 | 2200 | 4 | 0.030 | 0.273 | 0.021 | 0.068 |
| 0 | 3200 | 80 | 0.039 | 0.443 | 0.022 | 0.108 |
| 0.01 | 2200 | 4 | 2.52 | 35.42 | 2.43 | 23.97 |
| 0.01 | 1000 | 80 | 2.70 | 38.95 | 2.59 | 26.24 |
| 0.02 | 1200 | 4 | 3.49 | 49.56 | 3.53 | 34.92 |
| 0.03 | 1500 | 4 | 4.27 | 62.90 | 4.41 | 43.62 |
| 0.04 | 2000 | 4 | 4.98 | 76.12 | 5.15 | 52.20 |
| 0.05 | 2000 | 4 | 5.61 | 86.36 | 5.81 | 60.08 |
| 0.10 | 2000 | 4 | 8.37 | 139.32 | 7.38 | 76.99 |
| 0.10 | 260 | 80 | 9.41 | 148.92 | 8.66 | 81.60 |
| 0.20 | 2000 | 4 | 13.54 | 175.08 | 7.55 | 75.60 |
| 0.40 | 2000 | 4 | 15.03 | 161.68 | 7.76 | 49.77 |
| 0.40 | 150 | 80 | 16.64 | 198.22 | 10.35 | 80.29 |

larger. Inside the reduced cube, maximum errors do decrease, but remain so large that the biosensor can no longer make meaningful predictions in the worst case. The best results were obtained for $4$ principal components.

### 5.2 Effect of the convolutional architecture

Biosensor prediction errors for the convolutional architecture are given in Table 5. We also provide the neural network configuration for each noise level.

If we use a convolutional neural network, we can see that the testing errors are larger for the batch mode, but smaller for FIA compared to the FNN case. This suggests that such a network architecture is more suitable for the latter analysis mode.

When looking at the errors inside the reduced cube, we see the similar picture with the main exception being that the reduction in maximal errors is smaller in the flow injection case and larger in the batch case than for FNN networks.

However, again, at the noise level $\sigma$ larger than $0.01$, the maximum errors become so large that in the worst case, the biosensor can no longer make meaningful predictions.

**Table 5.** Prediction errors for the convolutional architecture.

| $\sigma$ | Config | Epochs | $E_{ts,a}$ | $E_{ts,m}$ | $E_{rc,a}$ | $E_{rc,m}$ |
|---|---|---|---|---|---|---|
| | | | Batch | | | |
| 0 | 1 | 990 | 0.392 | 4.420 | 0.261 | 1.812 |
| 0.01 | 1 | 936 | 2.09 | 25.75 | 1.69 | 13.70 |
| 0.05 | 1 | 979 | 5.01 | 72.97 | 4.50 | 38.08 |
| | | | FIA | | | |
| 0 | 3 | 340 | 0.095 | 0.945 | 0.111 | 0.470 |
| 0.01 | 5 | 277 | 1.25 | 17.35 | 1.16 | 12.19 |
| 0.05 | 5 | 393 | 3.25 | 52.10 | 3.07 | 38.97 |
| 0.10 | 7 | 200 | 6.03 | 101.26 | 5.48 | 65.81 |
| 0.40 | 7 | 200 | 12.74 | 182.75 | 8.24 | 88.29 |

## 5.3 Effect of multiplicative noise

Results for the multiplicative noise experiment are presented in Tables 6–8.

For the batch mode and multiplicative noise, the errors for noisy data are much larger than for additive noise.

Taking a larger number of principal components (PCs) has not reduced the errors in this case. These results suggest that the inverse biosensor problem in batch mode with

**Table 6.** Batch mode prediction errors for the FNN architecture and multiplicative noise.

| $\sigma$ | Epochs | PCs | $E_{ts,a}$ | $E_{ts,m}$ | $E_{rc,a}$ | $E_{rc,m}$ |
|---|---|---|---|---|---|---|
| 0 | 3200 | 4 | 0.078 | 1.899 | 0.038 | 0.214 |
| 0 | 3200 | 80 | 0.050 | 1.093 | 0.039 | 0.269 |
| 0.01 | 2200 | 4 | 5.34 | 82.11 | 5.17 | 42.39 |
| 0.01 | 1000 | 80 | 5.52 | 78.93 | 5.12 | 41.08 |
| 0.02 | 2200 | 4 | 9.64 | 128.39 | 8.88 | 71.27 |
| 0.02 | 700 | 80 | 9.61 | 136.20 | 8.50 | 67.72 |
| 0.03 | 2200 | 4 | 12.03 | 141.35 | 9.41 | 79.48 |
| 0.04 | 2200 | 4 | 13.48 | 155.49 | 9.51 | 86.14 |
| 0.05 | 2200 | 4 | 14.52 | 156.96 | 9.22 | 83.86 |

**Table 7.** FIA mode prediction errors for the FNN architecture and multiplicative noise.

| $\sigma$ | Epochs | PCs | $E_{ts,a}$ | $E_{ts,m}$ | $E_{rc,a}$ | $E_{rc,m}$ |
|---|---|---|---|---|---|---|
| 0 | 2200 | 4 | 0.030 | 0.273 | 0.021 | 0.068 |
| 0 | 3200 | 80 | 0.039 | 0.443 | 0.022 | 0.108 |
| 0.01 | 2000 | 4 | 0.14 | 1.25 | 0.11 | 0.58 |
| 0.01 | 400 | 80 | 0.22 | 2.23 | 0.16 | 0.83 |
| 0.02 | 2000 | 6 | 0.23 | 2.11 | 0.21 | 1.17 |
| 0.03 | 2000 | 4 | 0.36 | 3.18 | 0.31 | 1.78 |
| 0.04 | 2000 | 4 | 0.47 | 4.09 | 0.41 | 2.41 |
| 0.05 | 2000 | 4 | 0.58 | 5.02 | 0.52 | 3.02 |
| 0.10 | 2000 | 4 | 1.11 | 9.34 | 1.03 | 5.88 |
| 0.20 | 2000 | 4 | 2.19 | 19.75 | 2.06 | 11.49 |
| 0.40 | 2000 | 4 | 4.15 | 41.12 | 4.11 | 22.91 |
| 0.80 | 2000 | 4 | 7.67 | 75.68 | 7.98 | 45.23 |

**Table 8.** Prediction errors for the convolutional architecture and multiplicative noise.

| $\sigma$ | Config | Epochs | $E_{ts,a}$ | $E_{ts,m}$ | $E_{rc,a}$ | $E_{rc,m}$ |
|---|---|---|---|---|---|---|
| | | | Batch | | | |
| 0 | 1 | 990 | 0.392 | 4.420 | 0.261 | 1.812 |
| 0.01 | 2 | 459 | 6.13 | 82.67 | 6.01 | 45.89 |
| 0.05 | 2 | 131 | 14.83 | 180.28 | 9.80 | 94.08 |
| | | | FIA | | | |
| 0 | 3 | 340 | 0.095 | 0.945 | 0.111 | 0.470 |
| 0.01 | 4 | 335 | 0.26 | 2.25 | 0.19 | 1.40 |
| 0.05 | 3 | 312 | 0.64 | 6.81 | 0.59 | 3.99 |
| 0.10 | 5 | 200 | 1.25 | 10.00 | 1.16 | 7.26 |
| 0.20 | 5 | 210 | 2.39 | 17.71 | 2.23 | 13.61 |
| 0.40 | 5 | 240 | 4.19 | 34.23 | 3.97 | 24.33 |
| 0.80 | 4 | 308 | 6.01 | 51.47 | 5.91 | 37.86 |

multiplicative noise is very ill-posed. Mean and maximum errors are large even for the small noise level $\sigma$ and inside the reduced domain.

For the flow injection analysis mode and multiplicative noise, the errors for noisy data are much smaller than for the additive noise. These results suggest that the inverse biosensor problem in FIA mode with multiplicative noise is significantly less ill-posed. Mean and maximum errors are relatively small for a significant level of noise $\sigma = 0.2$ or lower.

Results obtained with the convolutional neural network are very similar to the results obtained with the feed-forward network, even though these architectures are completely different. This confirms our suggestion the inverse biosensor problem in FIA mode with multiplicative noise is significantly less ill-posed compared to the other operational modes. Initial concentrations in this case can be reliably recovered for higher levels of noise compared to all other cases considered in this study.

### 5.4 Results for the DIRECT method

In this section, we used the parallel master-worker DIRECT global optimization algorithm [30] to solve the inverse problem for a single concentration vector $\mathbf{c} = (4.5714, 10.0571, 3.2)$ for batch and FIA modes. The goal of this test is to see if biosensor precision in the FIA mode is better with this algorithm as with considered neural networks. The signals had no noise applied. To get acceptable results, we had to use a search region smaller than the original cube: $R = [3.4, 6] \times [7, 12.6] \times [3.2, 6]$. Computations were carried out on the VILNIUS TECH Vanagas cluster. We compared the performance of this algorithm to that of the convolutional neural network. Results are given in Table 9. Here the prediction error is calculated as the largest relative error among the three concentrations.

Table 9 shows that the DIRECT algorithm performs better for FIA mode than for batch mode as expected. Furthermore, for the FIA mode, DIRECT found a solution faster: it took 50 iterations, while the batch mode took 100, and increasing it to 200 yielded no benefit. Regarding scalability results (see Table 10), the DIRECT algorithm shows that the speedup is lower than the corresponding $p$ and the efficiency decreases as $p$ increases. This

**Table 9.** Prediction errors for the DIRECT algorithm.

| | Batch | | FIA | |
|---|---|---|---|---|
| | CNN | DIRECT | CNN | DIRECT |
| Error | 1.83 | 7.39 | 0.229 | 1.53 |

**Table 10.** Scalability results for the DIRECT algorithm (all cases had one master process).

| Number of worker processes $p$ | Time | Speedup | Efficiency |
|---|---|---|---|
| 1 | 1109 | – | – |
| 9 | 155.17 | 7.15 | 0.79 |
| 19 | 90.26 | 12.29 | 0.65 |

is due to the way DIRECT works: it generates a variable amount of tasks every iteration, which is then distributed among processes. Since we use a small amount of iterations, we have small amounts of tasks and cannot make full use of available processes. Therefore, increasing their amount leads to lowering of efficiency.

## 6 Conclusion

We have investigated the inverse biosensor problem – find initial concentrations of analytes in a mixture from its biosensor response when the latter is corrupted with noise. We have used a mathematical model to simulate biosensor signals for two analysis modes: batch mode and flow injection. The noise was modelled using additive and multiplicative Gaussian noises. To solve the problem, we employed neural networks of two different architectures: feed-forward and convolutional networks. Obtained results show that the largest errors of recovered initial concentrations are located on the edges of the training domain. We have found that the inverse biosensor problem is significantly less ill-posed in the flow injection analysis mode with the multiplicative noise. In this case, boundary concentrations of three analytes can be more accurately recovered for higher levels of noise compared to the other considered cases. This finding is confirmed by the application of the DIRECT global optimization method to the considered inverse biosensor problem.

Our paper provides recommendations to improve biosensor performance when neural networks are used. The prediction domain should be smaller than the training domain. In future work, we plan to investigate the problem of biosensor signal classification to introduce a mathematical biosensor model with fractional power diffusion and to test our methods on real biosensor signals.

## References

1. F. Alblehai, M.H. Mahmoud, M.S. Akhtar, A. Shaker, A.A. Mohamed, Analysing biosensor clinical pathogen information using mayfly optimized convolute neural network approach, *Expert Syst.*, p. e13027, 2022, https://doi.org/10.1111/exsy.13027.
2. R.C. Aster, B. Borchers, C.H. Thurber, *Parameter Estimation and Inverse Problems* 3rd ed., Elsevier, Amsterdam, 2018, https://doi.org/10.1016/C2015-0-02458-3.

3. R. Baronas, F. Ivanauskas, J. Kulys, *Mathematical Modeling of Biosensors*, Springer, Cham, 2021, https://doi.org/10.1007/978-3-030-65505-1.
4. R. Baronas, F. Ivanauskas, R. Maslovskis, P. Vaitkus, An analysis of mixtures using amperometric biosensors and artificial neural networks, *J. Math. Chem.*, **36**(3):281–297, 2004, https://doi.org/10.1023/B:JOMC.0000044225.76158.8e.
5. R. Baronas, J. Kulys, A. Lančinskas, A. Žilinskas, Effect of diffusion limitations on multianalyte determination from biased biosensor response, *Sensors*, **14**(3):4634–4656, 2014, https://doi.org/10.3390/s140304634.
6. R. Cartas, A. Mimendia, A. Legin, M. del Valle, Two analyte calibrations from the transient response of a single potentiometric sensor employed with the SIA technique, *Talanta*, **80**(3):1428–1435, 2010, https://doi.org/10.1016/j.talanta.2009.09.048.
7. F. Cui, Y. Yue, Y. Zhang, Z. Zhang, H.S. Zhou, Advancing biosensors with machine learning, *ACS Sens.*, **5**(11):3346–3364, 2020, https://doi.org/10.1021/acssensors.0c01424.
8. I. Dapšys, R. Čiegis, Numerical simulation of fractional power diffusion biosensors, *Math. Model. Anal.*, **28**(2):180–193, 2023, https://doi.org/10.3846/MMA.2023.17583.
9. S Deng, Y Hwang, Solution of inverse heat conduction problems using Kalman filter-enhanced Bayesian back propagation neural network data fusion, *Int. J. Heat Mass Transfer*, **50**(11–12):2089–2100, 2007, https://doi.org/10.1016/j.ijheatmasstransfer.2006.06.009.
10. N. Ding, Z. Yuan, X. Zhang, J. Chen, S. Zhou, Y. Deng, Programmable cross-ribosome-binding sites to fine-tune the dynamic range of transcription factor-based biosensor, *Nucleic Acids Res.*, **48**(18):10602–10613, 2020, https://doi.org/10.1093/nar/gkaa786.
11. S. Ghosh, D.K. Pratihar, B. Maiti, P.K. Das, Inverse estimation of location of internal heat source in conduction, *Inverse Probl. Sci. Eng.*, **19**(3):337–361, 2011, https://doi.org/10.1080/17415977.2011.551876.
12. A. Gulli, S. Pal, *Deep Learning with Keras*, Packt Publishing, Birmingham, 2017, https://www.oreilly.com/library/view/deep-learning-with/9781787128422.
13. H. Gutfreund, *Kinetics for the Life Sciences: Receptors, Transmitters, and Catalysts*, Cambridge Univ. Press, Cambridge, 1995.
14. A. Hassibi, H. Vikalo, A. Hajimiri, On noise processes and limits of performance in biosensors, *J. Appl. Phys.*, **102**(1):014909, 2007, https://doi.org/10.1063/1.2748624.
15. Q. Hu, S. Wang, H. Duan, Y. Liu, A fluorescent biosensor for sensitive detection of Salmonella typhimurium using low-gradient magnetic field and deep learning via faster region-based convolutional neural network, *Biosensors*, **11**(11):447, 2021, https://doi.org/10.3390/bios11110447.
16. S. Kamyab, Z. Azimifar, R. Sabzi, P. Fieguth, Deep learning methods for inverse problems, *PeerJ Comput. Sci.*, **8**:e951, 2022, https://doi.org/10.7717/PEERJ-CS.951/SUPP-1.
17. S. Kaziz, I.B. Romdhane, F. Echouchene, M.H. Gazzah, Numerical simulation and optimization of AC electrothermal microfluidic biosensor for COVID-19 detection through Taguchi method and artificial network, *Eur. Phys. J. Plus*, **138**(1):96, 2023, https://doi.org/10.1140/epjp/s13360-023-03712-z.

18. N.A. Khan, F.S. Alshammari, C.A. Tavera Romero, M. Sulaiman, G. Laouini, Mathematical analysis of reaction–diffusion equations modeling the Michaelis–Menten kinetics in a micro-disk biosensor, *Molecules*, **26**(23):7310, 12 2021, ISSN 1420-3049, https://doi.org/10.3390/molecules26237310.

19. M.K.H. Kumar, P.S. Vishweshwara, N. Gnanasekaran, Evaluation of artificial neural network in data reduction for a natural convection conjugate heat transfer problem in an inverse approach: experiments combined with CFD solutions, *Sādhanā*, **45**(1):1–15, 2020, https://doi.org/10.1007/s12046-020-1303-x.

20. L. Litvinas, R. Baronas, Application of artificial neural networks and biosensors to determine concentrations of mixture, *Liet. Mat. Rink., Ser. B*, **55**:78–83, 2014 (in Lithuanian), https://doi.org/10.15388/LMR.B.2014.15.

21. L. Litvinas, R. Baronas, The influence of the diffusion module to determination of two substrate concentrations by artificial neural network, *Computational Science and Techniques*, **3**(2):445–453, 2015, https://doi.org/10.15181/csat.v3i2.1109.

22. L. Lu, X. Meng, Z. Mao, G.E. Karniadakis, DeepXDE: A deep learning library for solving differential equations, *SIAM Rev.*, **63**(1):208–228, 2021, https://doi.org/10.1137/19M1274067.

23. E. Martynko, D. Kirsanov, Application of chemometrics in biosensing: A brief review, *Biosensors*, **10**(8):100, 2020, https://doi.org/10.3390/bios10080100.

24. M.H. Mozaffari, L.-L. Tay, A review of 1D convolutional neural networks toward unknown substance identification in portable Raman spectrometer, 2020, https://doi.org/10.48550/arxiv.2006.10575.

25. S. Pakravan, P.A. Mistani, M.A. Aragon-Calvo, F. Gibou, Solving inverse-PDE problems with physics-aware neural networks, *J. Comput. Phys.*, **440**:110414, 2021, https://doi.org/10.1016/j.jcp.2021.110414.

26. M.R. Romero, A.M. Baruzzi, F. Garay, Mathematical modeling and experimental results of a sandwich-type amperometric biosensor, *Sens. Actuators, B*, **162**(1):284–291, 2012, https://doi.org/10.1016/J.SNB.2011.12.079.

27. É.H. Shiguemori, J.D.S. Da Silva, H.F. de Campos Velho, Estimation of initial condition in heat conduction by neural network, *Inverse Probl. Sci. Eng.*, **12**(3):317–328, 2004, https://doi.org/10.1080/10682760310001598599.

28. A.N. Tikhonov, A.V. Goncharsky, V.V. Stepanov, A.G. Yagola, *Numerical Methods for the Solution of Ill-Posed Problems*, Springer, Dordrecht, 1995, https://doi.org/10.1007/978-94-015-8480-7.

29. H. Wang, Y. Wang, X. Hou, B. Xiong, Bioelectronic nose based on single-stranded DNA and single-walled carbon nanotube to identify a major plant volatile organic compound (P-ethylphenol) released by Phytophthora cactorum infected strawberries, *Nanomaterials*, **10**(3):479, 3 2020, https://doi.org/10.3390/nano10030479.

30. L.T. Watson, C.A. Baker, A fully-distributed parallel global search algorithm, *Eng. Comput. (Bradf.)*, **18**(1/2):155–169, 2001, https://doi.org/10.1108/02644400110365851.

31. A. Zhang, Z.C. Lipton, M. Li, A.J. Smola, Dive into deep learning, 2021, https://doi.org/10.48550/arXiv.2106.11342.

32. Y. Zhang, J. Sun, L. Liu, H. Qiao, A review of biosensor technology and algorithms for glucose monitoring, *J. Diabetes Complicat.*, **35**(8):107929, 8 2021, https://doi.org/10.1016/j.jdiacomp.2021.107929.

33. A. Žilinskas, D. Baronas, Optimization-based evaluation of concentrations in modeling the biosensor-aided measurement, *Informatica*, **22**(4):589–600, 2011, https://content.iospress.com/articles/informatica/inf22-4-08.