sensors

MDPI

Article

Rational Design of Field-Effect Sensors Using Partial Differential Equations, Bayesian Inversion, and Artificial Neural Networks

Amirreza Khodadadian [1,*], Maryam Parvizi [1,2], Mohammad Teshnehlab [3], Clemens Heitzinger [4,5]

[1] Institute of Applied Mathematics, Leibniz University Hannover, Welfengarten 1, 30167 Hannover, Germany; parvizi@ifam.uni-hannover.de
[2] Cluster of Excellence PhoenixD (Photonics, Optics, and Engineering-Innovation Across Disciplines), Leibniz University Hannover, 30167 Hannover, Germany
[3] Faculty of Electrical Engineering, K. N. Toosi University of Technology, Tehran 19697, Iran; teshnehlab@eetd.kntu.ac.ir
[4] Institute of Analysis and Scientific Computing, TU Wien, Wiedner Hauptstrasse 8–10, 1040 Vienna, Austria; clemens.heitzinger@tuwien.ac.at
[5] Center for Artificial Intelligence and Machine Learning (CAIML), TU Wien, 1040 Vienna, Austria
* Correspondence: khodadadian@ifam.uni-hannover.de

Abstract: Silicon nanowire field-effect transistors are promising devices used to detect minute amounts of different biological species. We introduce the theoretical and computational aspects of forward and backward modeling of biosensitive sensors. Firstly, we introduce a forward system of partial differential equations to model the electrical behavior, and secondly, a backward Bayesian Markov-chain Monte-Carlo method is used to identify the unknown parameters such as the concentration of target molecules. Furthermore, we introduce a machine learning algorithm according to multilayer feed-forward neural networks. The trained model makes it possible to predict the sensor behavior based on the given parameters.

Keywords: field-effect sensors; biosensors; charge transport; neural networks; Bayesian inversion; inverse modeling

check for updates

Citation: Khodadadian, A.; Parvizi, M.; Teshehlab, M.; Heitzinger, C. Rational Design of Field-Effect Sensors Using Partial Differential Equations, Bayesian Inversion, and Artificial Neural Networks. *Sensors* **2022**, *22*, 4785. https://doi.org/10.3390/s22134785

Academic Editors: Michael J. Schöning and Sven Ingebrandt

Received: 31 May 2022
Accepted: 21 June 2022
Published: 24 June 2022

1. Introduction

Silicon nanowire (SiNW) field-effect transistors (FETs) are typically used to detect proteins [1], cancer cells [2], DNA and miRNA strands [3,4], enzymes [5], and toxic gases such as carbon monoxide [6,7]. The sensors have several advantages including fast response, very high sensitivity, and low power consumption; they do not need labeling and can be used to detect subpicomolar concentrations of biological species [8–13]. The functioning of the sensors is based on the field effect due to the (partial) charges of the target molecules. When they are selectively bound to probe molecules and close enough to the semiconducting transducer, they affect the charge concentration inside the nanowire, which changes the current through the nanowire.

Using mathematical models based on partial differential equations (PDEs) enables us to model physically relevant quantities such as electrostatic potential, electron and hole current density, device sensitivity to the target molecule and signal-to-noise ratio [14–18]. The three-dimensional simulations give rise to more reliable models compared to two-dimensional cross-sections, since all target molecules bound to bio-receptors will be included [19,20]. We couple a charge transport model (the drift-diffusion equations) and the nonlinear Poisson–Boltzmann equation (PBE) for fully self-consistent simulations. The system of equations is a comprehensive model to compute the electrical current and study the nonlinear effects of different semiconductor parameters (e.g., doping concentration) and device parameters such as nanowire type (radial, trapezoidal, radial, or rectangular), its dimensions, contact voltages, and insulator thickness on device performance (output and sensitivity).

Having an accurate model enables the rational design of field-effect sensors. However, in the model equations, there are several material parameters that cannot be (easily) measured. The surface charge density of the insulator has an essential effect on the device and also affects the probe and target molecules. The doping concentration has a crucial effect on the device and the model. Due to the nonlinear effect of these parameters, an efficient parameter estimation framework will enhance the accuracy and reliability of the model.

Markov-chain Monte-Carlo (MCMC) techniques are among the most efficient probabilistic methods to extract information by comparison between measurements and simulations by updating available prior knowledge and estimating the posterior densities of unknown quantities of interest. Here, we use a forward model, and a backward, inverse setting is used to determined the unknown parameters using the experiments. The classical algorithm was introduced in 1970 and is called the *Metropolis–Hastings (MH) algorithm* [21]. There are several improvements in the algorithm, e.g., adaptive-proposal Metropolis [22], delayed-rejection Metropolis [23], and delayed rejection adaptive Metropolis (DRAM) [24], as well as using ensemble Kalman filters [25]. In all techniques, different candidates are proposed based on a proposal distribution, and the algorithm decides whether they are rejected or accepted. A review of the MCMC methods is given in [26]. For SiNW-FETs, the DRAM algorithm has been used to identify the doping concentration and the amount of target molecules [14]. Considering the selective functionalization of SiNW, the authors of [1] used the MH algorithm to estimate the probe-target density at the surface.

Neural networks (also known as artificial neural networks (ANNs)) as the subset of machine learning are frameworks to analyze the available data and discover patterns that can not be observed independently. The ANNs have been inspired by the human brain and are suitable for complicated and nonlinear cases. Here, we split the prior data into two categories, namely training and testing data. The training set (between 60% and 80%) is used to extract useful information from the data, and the test set (between 20% and 40%) is employed to monitor the algorithm performance. In SiNW-FETs, there are a large amount of simulation and experimental data concerning different input (physical, chemical, and device) parameters that should be analyzed to ensure their accuracy and reliability. Of course, this process is time consuming and reduces the efficiency. Furthermore, the sensors are developed to detect specific biological species with the highest sensitivity. In the design process, using neural networks enables us to optimize the design parameters to enhance the sensor performance [27–32].

This article is structured as follows. In Section 2, we present the model equations and explain how the electrical current is computed. In Section 3, we discuss the parameter estimation methods and explain how MCMC can be used to determine the unknown parameters. In Section 4, we introduce the developed neural networks algorithm for SiNW-FETs. In Section 5, we first verify the model response with the experimental data; then, Bayesian inversion is used to identify the material parameters. Afterward, the developed machine-learning algorithm is employed in training and testing. Finally, the conclusions are summarized in Section 6.

2. The Model Equations

The drift–diffusion–Poisson system is used to describe the electrochemical interactions (Poisson–Boltzmann equation) and the charge transport (drift–diffusion equations) in field-effect sensors. The convex and bounded domain $\Omega \subset \mathbb{R}^3$ consists of four subdomains, namely the insulator (SiO_2, Ω_{Si}), the silicon substrate and transducer (Ω_{Si}), the aqueous solution (Ω_{liq}), and the charged molecules (Ω_{mol}). To model the potential interactions, we use the Poisson–Boltzmann equation

$$-\nabla\cdot(A(x)\nabla V(x)) = \begin{cases} q(C_{\text{dop}}(x)+p(x)-n(x)) & \text{in } \Omega_{\text{Si}}, \\ 0 & \text{in } \Omega_{\text{ox}}, \\ \rho(x) & \text{in } \Omega_{\text{M}}, \\ -2\varphi(x)\sinh(\beta(V(x)-\Phi_F)) & \text{in } \Omega_{\text{liq}}, \end{cases} \tag{1}$$

where A indicates the dielectric constant, which is a function of the material, V is the electrostatic potential, C_{dop} is the doping concentration, ρ is the surface charge of the molecules, ϕ_F denotes the Fermi level, and φ is the ionic concentration. Regarding the electrical constants, we use the relative values $A_{\text{Si}} = 11.7$, $A_{\text{ox}} = 3.9$, $A_{\text{M}} = 3.7$, and $A_{\text{liq}} = 78.4$. Considering the Boltzmann constant k_B, the temperature T and the elementary charge q, we define $\beta = q/(k_B T)$. In the simulations, a thermal voltage of 0.021 V will be used.

A two-dimensional cross-section of the device is given in Figure 1.

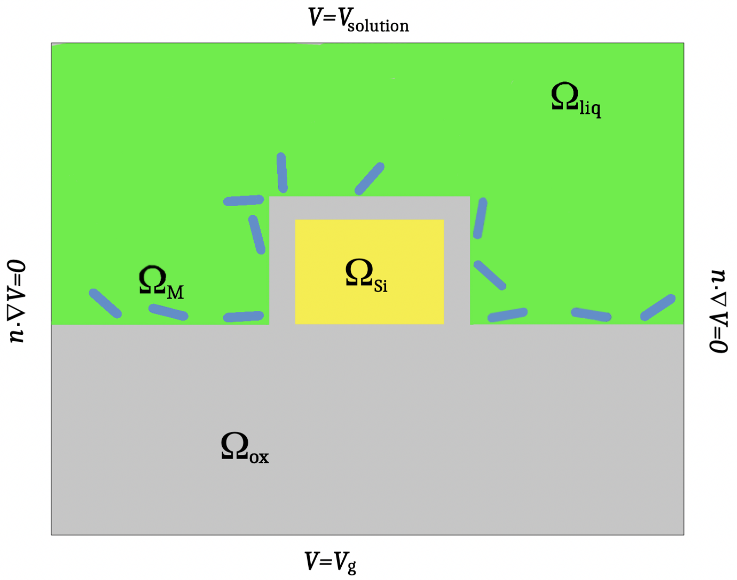

Figure 1. A schematic cross-section of a SiNW-FET depicting the subdomains, i.e., the transducer Ω_{Si}, SiO$_2$ insulator (Ω_{ox}), the aqueous solution Ω_{liq}, the binding of the target molecules to the immobilized receptor molecules (Ω_{mol}), and the boundary conditions.

At the interface between the insulator and the liquid (i.e., $\Gamma := \Omega_{\text{ox}} \cap \Omega_{\text{liq}}$), we impose the interface conditions

$$A(0+)(V(0+,y,z) - V(0-,y,z)) = \alpha(y,z) \qquad \text{on } \Gamma, \tag{2a}$$

$$A(0+)\partial_x V(0+,y,z) - A(0-)\partial_x V(0-,y,z) = \gamma(y,z) \qquad \text{on } \Gamma \tag{2b}$$

for V_I. Here, $0+$ and $0-$ denote the limit at the interface on the side of liquid and insulator. Furthermore, α is macroscopic dipole moment density, and γ is the macroscopic surface-charge density.

In Ω_{Si}, we solve the drift–diffusion system

$$-\nabla\cdot(A\nabla V) = q(p(x) - n(x) + C_{\text{dop}}(x)), \tag{3a}$$

$$\nabla\cdot J_n = qR(n,p), \tag{3b}$$

$$\nabla\cdot J_p = -qR(n,p), \tag{3c}$$

$$J_n = q(D_n\nabla n - \mu_n n\nabla V), \tag{3d}$$

$$J_p = q(-D_p\nabla p - \mu_p p\nabla V) \tag{3e}$$

to model the charges in the transistor, where D_n and D_p are the electron and hole diffusion coefficients. The concentrations of electrons and holes are given by

$$p =: n_i \exp\left(\frac{q}{K_B T}(\Phi_1 - V)\right), \qquad n =: n_i \exp\left(\frac{-q}{K_B T}(\Phi_2 - V)\right), \tag{4}$$

where n_i is the intrinsic carrier density and Φ_1 and Φ_1 are the Fermi levels. In order to compute the electron and hole current densities, we use the Shockley–Read–Hall recombination rate, i.e.,

$$R(n,p) := \frac{np - n_i^2}{\tau_n(p + n_i) + \tau_p(n + n_i)},$$

where τ_n and τ_p denote the lifetimes of the electrons and holes.

For solving the nonlinear system of equations, we use the Scharfetter–Gummel iteration. For this, we write the concentrations n and p in terms of the two Slotboom variables u and v as

$$n(x,\omega) =: n_i \mathrm{e}^{V(x,\omega)/U_T} u(x,\omega), \tag{5a}$$
$$p(x,\omega) =: n_i \mathrm{e}^{-V(x,\omega)/U_T} v(x,\omega). \tag{5b}$$

Therefore, the model problem (3) can be rewritten as

$$-\nabla \cdot (A(x)\nabla V(x)) = q\Big(C_{\text{dop}}(x) - n_i\Big(\mathrm{e}^{V(x)/U_T} u(x) - \mathrm{e}^{-V(x)/U_T} v(x)\Big)\Big), \tag{6a}$$
$$U_T n_i \nabla \cdot (\mu_n e^{V/U_T} \nabla u(x)) = R(x), \tag{6b}$$
$$U_T n_i \nabla \cdot (\mu_p e^{-V/U_T} \nabla v(x)) = R(x), \tag{6c}$$

where U_T is the thermal voltage and the Shockley–Read–Hall recombination rate takes the form

$$R_{\text{SRH}}(x) = n_i \frac{u(x)v(x) - 1}{\tau_p(e^{V/U_T} u(x) + 1) + \tau_n(e^{-V/U_T} v(x) + 1)}.$$

At the ohmic contacts (backgate, source, and drain) and the solution gate, we have a Dirichlet boundary condition $V_{\partial\Omega} = V_D$ consisting of

$$V|_{\partial\Omega_{\text{G}}} = V_{\text{g}} \qquad V|_{\partial\Omega_{\text{S}}} = V_{\text{S}} \qquad V|_{\partial\Omega_{\text{D}}} = V_{\text{D}} \qquad V|_{\partial\Omega_{\text{sol}}} = V_{\text{solution}}. \tag{7}$$

At the source and drain contacts (on $\partial\Omega_{\text{Si}}$), we apply

$$u(x) = u_D(x), \quad v(x) = v_D(x). \tag{8}$$

For the remaining part of the domain, we impose a zero Neumann boundary condition to guarantee the self-isolation. We refer the interested reader to [15,19,33,34] for theoretical discussions about the model including the Slotboom variables. The existence and uniqueness of the solutions for deterministic and stochastic model problems are given in [15,35]. Finally, the computation of J_n and J_p enables us to calculate the electrical current as

$$\mathcal{I} := \int (J_n + J_p)\,\mathrm{d}x, \tag{9}$$

where we take the integral on a cross-section of the transducing part.

In this work, we use the finite element method (FEM) to solve the coupled system of equations. We define the spaces

$$X^1 = \Big\{ V \in H^1(\Omega) \quad | \quad V|_{\partial\Omega} = V_D\,, \;\; V|_\Gamma = V_I \Big\}, \tag{10a}$$

$$X^2 = \Big\{ u \in H^1(\Omega_{\mathrm{Si}}) \quad | \quad u|_{\partial\Omega_{\mathrm{Si}}} = u_D \Big\}, \tag{10b}$$

$$X^3 = \Big\{ v \in H^1(\Omega_{\mathrm{Si}}) \quad | \quad v|_{\partial\Omega_{\mathrm{Si}}} = v_D \Big\}. \tag{10c}$$

Therefore, we define the continuous solution space $X := X^1 \times X^2 \times X^3$ for the DDP system. Regarding the space discretization, we assume $\mathcal{T}_h = \{T_1, T_2, \ldots, T_n\}$ denotes a quasi-uniform mesh defined on $\Omega_h \approx \Omega$ with mesh width $h := \max_{T_j \in \mathcal{T}_h} \mathrm{diam}(T_j)$. We define

$$\mathcal{S}^1_V(\mathcal{T}_h) := \{ V \in H^1(\Omega) \quad | \quad V|_T \in \mathcal{P}_1(T) \;\; \forall T \in \mathcal{T}_h \},$$
$$\mathcal{S}^1_u(\mathcal{T}_h) := \{ u \in H^1(\Omega) \quad | \quad u|_T \in \mathcal{P}_1(T) \;\; \forall T \in \mathcal{T}_h \},$$
$$\mathcal{S}^1_v(\mathcal{T}_h) := \{ v \in H^1(\Omega) \quad | \quad v|_T \in \mathcal{P}_1(T) \;\; \forall T \in \mathcal{T}_h \},$$

where $\mathcal{P}_1$ is the space of first-order polynomials. Then, we have

$$X^1_h := \Big\{ V_h \in \mathcal{S}^1_V(\mathcal{T}_h) \quad | \quad V_h|_{\partial\Omega} = V_D\,, \;\; V_h|_\Gamma = V_I \Big\}, \tag{11a}$$

$$X^2_h := \Big\{ u_h \in \mathcal{S}^1_u(\mathcal{T}_h) \quad | \quad u_h|_{\partial\Omega_{\mathrm{Si}}} = u_D \Big\}, \tag{11b}$$

$$X^3_h := \Big\{ v_h \in \mathcal{S}^1_v(\mathcal{T}_h) \quad | \quad v_h|_{\partial\Omega_{\mathrm{Si}}} = v_D \Big\}. \tag{11c}$$

The discrete solution is defined as $X_h := X^1_h \times X^2_h \times X^3_h$, which is a subset of X. The weak form of the model equations can be found in [15,33]. The a prior and a posterior estimations are proved in [33]. More theoretical works regarding the finite elements analysis are given in [36–38].

3. Parameter Estimation Based on Bayesian Inference

In different experimental situations, an accurate estimation of the effective parameters and constants cannot be easily estimated. Bayesian inversion techniques based on Markov chain Monte Carlo methods are efficient and straightforward probabilistic techniques to estimate these unknowns. We initiate the algorithm using the available information, named *prior* knowledge (which may not be sufficiently accurate), and during several iterations, we can update the information and provide more reliable data (i.e., the posterior density). Then, we can extract valuable information from the posterior density, and its mean/median can be used as the solution of the interference. A very strong agreement with the experimental values and the model response can be achieved. We start a statistical model

$$\mathcal{M} = \mathcal{P}(x, \chi) + \varepsilon, \tag{12}$$

where $\mathcal{M}$ is the experimental observation (normally $n-$ dimensional), while $\mathcal{P}$ is the solution of the model problem which depends on the set of parameters χ (i.e., $\chi = (\chi_1, \chi_2, \ldots, \chi_k)$ and the Cartesian coordinates x. Here, ε is the measurement error, and we assume that it is normally distributed, i.e., $\varepsilon \sim \mathcal{N}(0, \sigma^2 I)$, including the parameter σ^2. Having an experimental observation, for instance electrical current (i.e., $\mathcal{M} = \mathtt{obs}$), we define the probability function

$$\pi(\mathtt{obs}) = \int_{\mathbb{R}^n} \pi(\mathtt{obs}|\chi)\pi_0(\chi)\,\mathrm{d}\chi. \tag{13}$$

Our aim is to estimate the posterior density $\pi(\chi|m)$, considering the measured observation m and the available prior information. For this, we compute the likelihood function

$$\pi(\mathcal{M}|\chi) = L(\chi, \sigma^2|\mathcal{M}) = \frac{1}{(2\pi\sigma^2)^{n/2}} \exp\left(-\mathcal{M}_{\mathcal{P}}/2\sigma^2\right) \tag{14}$$

where

$$\mathcal{M}_{\mathcal{P}} = \sum_{j=1}^{n} [\mathcal{M}_j - \mathcal{P}_j(x, \chi)]^2 \tag{15}$$

is the sum of square errors. Obviously, if the model response with respect to the (set of) parameters χ will be closer to the measured value, the square error (15) will converge to zero, and its relative probability (computed by the likelihood function) will converge to 1. Inaccurate estimation of χ will increase the error term, and the probability will converge to zero.

In the Metropolis algorithm, we initiate the process using an initial guess χ^0 based on the prior density. According to the proposal distribution, a new candidate $\chi^\star$ is proposed. We compute the acceptance rate by

$$\lambda(\chi^{j-1}, \chi^\star) = \min\left(1, \frac{\pi(\chi^\star)}{\pi(\chi^{j-1})}\right). \tag{16}$$

If the new candidate $\chi^\star$ is accepted, we continue the MCMC chain with that; otherwise, (χ^{j-1} has a higher probability concerning $\chi^\star$), we follow the chain with the previous candidate. Using a non-symmetric proposal density is a generalization of the Metropolis algorithm, introduced by Hastings [21], where the probability of the forward jump is not equal to the backward one. A summary of the algorithm is given in Algorithm 1.

Algorithm 1 The Metropolis–Hastings algorithm.

```
Initialization: Start the process with the initial guess $\chi^0$ and number of samples $N$.
while $j < N$
  1. Propose a new sample according to the proposal density $\chi^* \sim \mathcal{T}(\chi^*|\,\chi^{j-1})$.
  2. Compute the acceptance/rejection ratio
       $\zeta(\chi^*|\,\theta^{j-1}) = \min\left(1, \frac{\pi(\chi^*|\,m)}{\pi(\chi^{j-1}|\,m)}\,\frac{\mathcal{T}(\chi^{j-1}|\,\chi^*)}{\mathcal{T}(\chi^*|\,\chi^{j-1}))}\right).$
  3. Sample $\mathcal{R} \sim$ Uniform $(0,1)$.
  4. if $\mathcal{R} < \zeta$ then
       accept $\chi^*$ and set $\chi^j := \chi^*$
     else
       reject $\chi^*$ and set $\chi^j := \chi^{j-1}$
     end if
  5. Set $j = j + 1$.
```

The Metropolis–Hastings algorithm is a simple and versatile technique and has been widely used for several problems in applied science. However, for the high-dimensional cases (different parameters should be inferred simultaneously), the algorithm does not work appropriately, since the rejection rate increases significantly. To improve its computational drawbacks, different improvements, such as the adaptive Metropolis algorithm [22], delayed rejection Metropolis [23], and their combination, namely delayed rejection adaptive

Metropolis (DRAM) [24]. We refer the interested readers to [26] as a review paper about the methods.

Mcmc with Ensemble-Kalman Filter (EnKF-MCMC)

In EnKF-MCM [25], we use a Kalman gain employing the mean and the covariance of the prior distribution and the cross-covariance between parameters and observations. It will be used to compute the proposal distribution and make the convergence to the target density faster. Here, the new candidate is computed as the jump of the Kalman-inspired proposal $\Delta\chi$ as

$$\chi^{\star} = \theta^{j-1} + \Delta\chi. \tag{17}$$

In order to update the candidates, we compute $\Delta\chi$ by

$$\Delta\chi = \mathcal{K}\left(y^{j-1} + s^{j-1}\right), \tag{18}$$

where $\mathcal{K}$ denotes the so-called Kalman gain,

$$\mathcal{K} = \mathcal{C}_{\chi M}(\mathcal{C}_{MM} + \mathcal{R})^{-1}. \tag{19}$$

Here, $\mathcal{C}_{\theta M}$ indicates the covariance matrix between the identified unknowns and model response, $\mathcal{C}_{MM}$ points out the covariance matrix of the model response, and $\mathcal{R}$ denotes the measurement noise covariance matrix [39]. In addition, y^{j-1} is the residual of the proposed values concerning the model and $s^{j-1} \sim \mathcal{N}(0, \mathcal{R})$ relates to the density of measurement. A summary of the relative algorithm is given in Algorithm 2. Finally, Figure 2 shows the implementation of EnKF-MCMC and Schafetter–Gummel iteration for parameter estimation and solving the model equations.

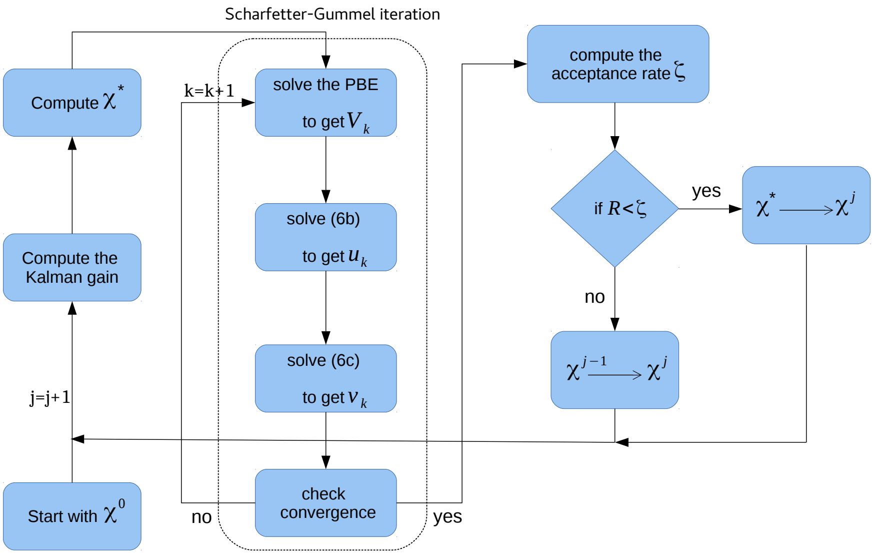
Scharfetter-Gummel iteration
Compute χ^*
k=k+1
solve the PBE to get V_k
solve (6b) to get u_k
solve (6c) to get v_k
check convergence
no
yes
Compute the Kalman gain
j=j+1
Start with χ^0
compute the acceptance rate ζ
if $R<\zeta$
yes
no
$\chi^* \longrightarrow \chi^j$
$\chi^{j-1} \longrightarrow \chi^j$

Figure 2. Bayesian inversion using EnKF-MCMC to identify the unknown material parameters, where the Scharfetter–Gummel iteration is used to solved the coupled system of equations.

Algorithm 2 Bayesian inference using EnKF-MCMC

```
Initialization ($j = 0$): Start the process with the initial guess $\chi^0$ and number of samples $N$.
while $j < N$
    1. Estimate the model response with respect to $\chi^{j-1}$
    2. Compute the Kalman gain    $\mathcal{K} = \mathcal{C}_{\chi M}(\mathcal{C}_{MM} + \mathcal{R})^{-1}$
    3. Produce the new proposal using the shift    $\chi^{\star} = \chi^{j-1} + \mathcal{K}\left(y^{j-1} + s^{j-1}\right)$
    4. Accepted/rejected $\chi^{\star}$
    5. Set $j = j + 1$.
```

4. Multilayer Feed-Forward Neural Networks

Neural networks are efficient, flexible, and robust simulation tools specifically for nonlinear and complicated problems. They consist of three effective components, including neurons, structures, and weights, which all affect the response and behavior of the network. Artificial neural networks (ANNs) are supervised machine learning algorithms consisting of neurons and hidden layers. The input data are processed into the hidden layers, the output is compared with the target trajectory, and the relative error is computed. The neural networks strive to minimize this error.

Typically, there are two common classes of neural networks, namely feed-forward neural networks (single or multilayers) and recurrent dynamics neural networks. Single-layer neural networks [40] have less complexity; however, they are more suitable for linear problems. In multilayer feed-forward neural networks (MFNNs) [41,42], more than one layer of the artificial neurons will be used to enhance the capability to learn nonlinear patterns, which is more appropriate for BIO-FETs. In MFNNs, the neurons are organized in different non-recurrent layers, where in the first layer, we have the input vector (here are the parameters of the sensor), and the output is given to the first hidden layer. After the data processing, the data are transferred to the next layers using the weights; the procedure is followed until the latest MFNNs layer. These networks are also named multilayer perceptrons, and their structure is shown in Figure 3.

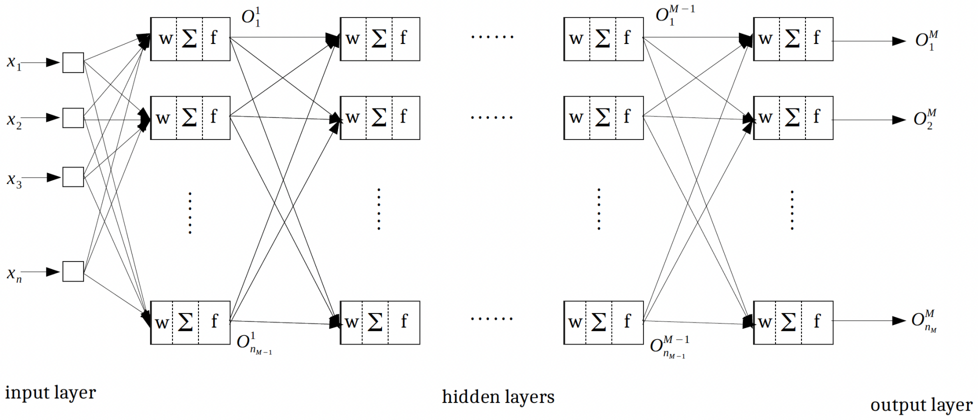

Figure 3. The structure of multilayer feed-forward neural networks (MFNNs).

Let us assume d denotes the desired trajectory (i.e., the device output); for M-layer neural networks, we have

$$
\begin{aligned}
\underset{(n_{s-1})\times 1}{\nabla w_j^s(k)} &= \eta^s \frac{\partial E}{\partial w_j^s}(k) = -\eta^s \delta_i^s(k) \underset{(n_{s-1})\times 1}{\nabla w_j^s(k)} \\
&= -\eta^s e_j^s(k) f_j^{s'}\left(net_j^s(k)\right) \underset{(n_{s-1})\times 1}{x^{s-1}} \qquad s = 1,2,\ldots,M, \quad j = 1,2,\ldots,n_s, \qquad (20)
\end{aligned}
$$

$$
\delta_j^s(k) := -\frac{\partial E}{\partial net_j^s}(k) = e_j^s(k) f_j^{s'}\left(net_j^s(k)\right), \tag{21}
$$

$$
e_j^s(k) = \sum_{l=1}^{n_{s+1}} \delta_j^{s+1}(k) w_{lj}^{s+1}(k), \tag{22}
$$

where w is the weights, η is the training rate, E is the network mean square error (MSE), δ is the sensitivity function (here, δ^s indicates the network error in the jth layer), net^s is the weighted input, n_s is the number of neurons in the sth layer, x^0 is the network input, x^{s-1} is the output of the $s-1$th layer, and it is also the input of the sth layer. We also have the following initial conditions for the recurrent process

$$
\delta_j^M(k) = e_j^M(k) f_j^{M'}\left(net_j^M(k)\right), \tag{23}
$$

$$
e_j^M(k)) \triangleq d_j(k) - O_j^M(k). \tag{24}
$$

Figure 4 shows the jth neuron in the ith layer in the learning algorithm. In the recurrent process, in order to adjust the weights from the first layer, we follow as

$$
\delta_{l_i}^s(k) = -\frac{\partial E(k)}{\partial net_{l_i}^s} \sum_{l_m=1}^{n_M} \sum_{l_{m-1}=1}^{n_{M-1}} \cdots \sum_{l_{i+2}=1}^{n_{i+2}} \sum_{l_{i+1}=1}^{n_{i+1}} \frac{\partial E}{\partial net_{l_m}^s} \frac{\partial net_{l_m}^s}{\partial net_{l_m-1}^{s-1}} \cdots \frac{\partial net_{l_{i+2}}^{s+2}}{\partial net_{l_{i+1}}^{s+1}} \frac{\partial net_{l_{i+1}}^{s+1}}{\partial net_{l_i}^s}(k) \tag{25}
$$

For $i = 1,2,\ldots,M-1$ and $s = 1,2,\ldots,M$, the relation $net_{l_i}^s$ and $net_{l_{i+1}}^{s+1}$ takes

$$
net_{l_{i+1}}^{s+1}(k) = \sum_{p=1}^{n_i} w_{l_{i+1}p}^{s+1}(k) f_p^s\left(net_p^s(k)\right), \tag{26}
$$

therefore

$$
\frac{\partial net_l^{s+1}}{\partial net_{l_i}^s}(k) = w_{l_{i+1}l_i}^{s+1}(k) f_{l_i}^{s'}\left(net_{l_i}^{s'}(k)\right). \tag{27}
$$

So, we can write $\delta_{l_i}^s$ as

$$
\delta_{l_i}^s(k) = \left(\sum_{l=1}^{n_{i+1}} \delta_l^{s+1}(k) w_{l\,l_i}^{s+1}(k)\right) f_{l_i}^{s'}\left(net_{l_i}^s(k)\right) = e_{l_i}^s(k) f_{l_i}^{s'}\left(net_{l_i}^s(k)\right), \tag{28}
$$

where

$$
e_{l_i}^s(k) = \sum_{l=1}^{n_{i+1}} \delta_l^{s+1}(k) w_{l\,l_i}^{s+1}(k). \tag{29}
$$

The gradient of E (the difference between desired trajectory and the neural networks's output) with respect to the weight vector is given by

$$
\frac{\partial E}{\partial w_{l_i}^s}(k) = \sum_{l=1}^{n} \frac{\partial E}{\partial net_{l_i}^s}(k) \frac{\partial net_{l_i}^s}{\partial w_{l_i}^s}(k), \tag{30}
$$

where the second term depends only on the neurons features and takes

$$\frac{\partial net_l^s}{\partial w_{l_i}^s}(k) = \begin{cases} x^{s-1}(k) & l = l_i, \\ 0 & \text{otherwise,} \end{cases} \tag{31}$$

$$\frac{\partial E}{\partial w_{l_i}^s}(k) = -\delta_{l_i}^s(k) x^{s-1}(k). \tag{32}$$

Using the back-propagation error algorithm enables us to adjust the weight functions in order to minimize the network error. This training process is also named the supervised learning algorithm.

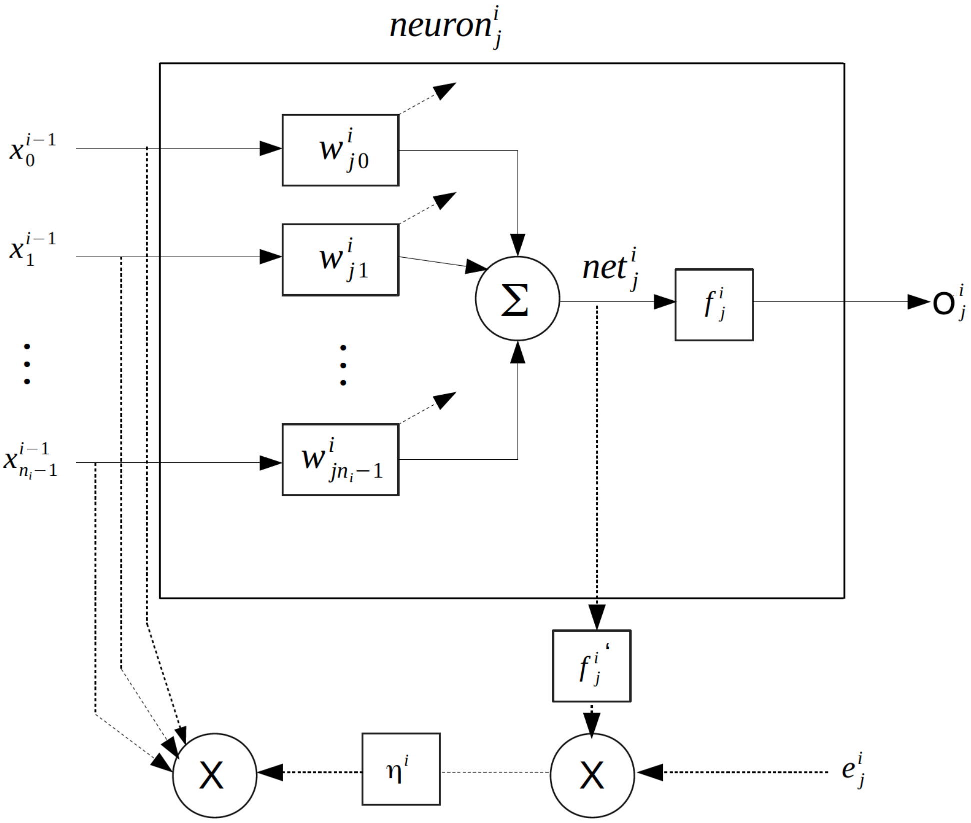

Figure 4. The back-propagation algorithm for the adjustment of neuron weights.

5. Numerical Experiments

As we already mentioned, the DDP system is a roust and reliable system of equations to model the electrical behavior of the FET devices. We use a prostate-specific antigen (PSA) sensitive sensor which is used to diagnose prostate cancer. For the simulations, we use a sensor device with the nanowire length of 1000 nm, width of 100 nm and height of 50 nm, which is coated with SiO_2 with 8 nm thickness. We use the P_1 finite element to solve the model problem, and tetrahedral meshes are employed to discretize the domain. A schematic of the bio-FET including dimensions using 6622 nodes and 45,735 tetrahedra is shown in Figure 5. The sensor is developed for the detection of 2ZCH (https://www.rcsb.org/structure/2ZCH). The PROPKA algorithm predicts the pK_a values of ionizable groups in proteins and protein–ligand complexes based on the 3D structure. The values are the basis for understanding the pH-dependent characteristics of proteins and catalytic mechanisms of many enzymes [43]. To compute the net charge, we performed a PROPKA algorithm [44–46] to detect the net charge for different pH values. The simulations are completed using a pH value of 9, giving rise to the net charge of $-15\,\mathrm{q}$ [14]. In field-effect sensors, surface reactions at the oxide surface depending on the pH value and the binding of charged target molecules result in changes in the charge concentration at and near the surface, and subsequently in changes in the electrostatic potential, which then modulates the current through the transducer. Since the molecules are negatively charged, the binding of the target molecules to the bio-receptors will enhance the charge conductance and increase the response of the sensor (i.e., the electrical current).

The system of equations is capable of modeling the surface charges at the surface. In a previous work, we developed a Monte-Carlo approach to simulate the charges around a charged biomolecule at a charged surface [47]. Furthermore, in [48], a nonlinear Poisson model was used to calculate the free energies of various molecule orientations in dependence of the surface charge. Based on the free energies, the probabilities of the orientations were calculated, and hence, the biological noise was simulated.

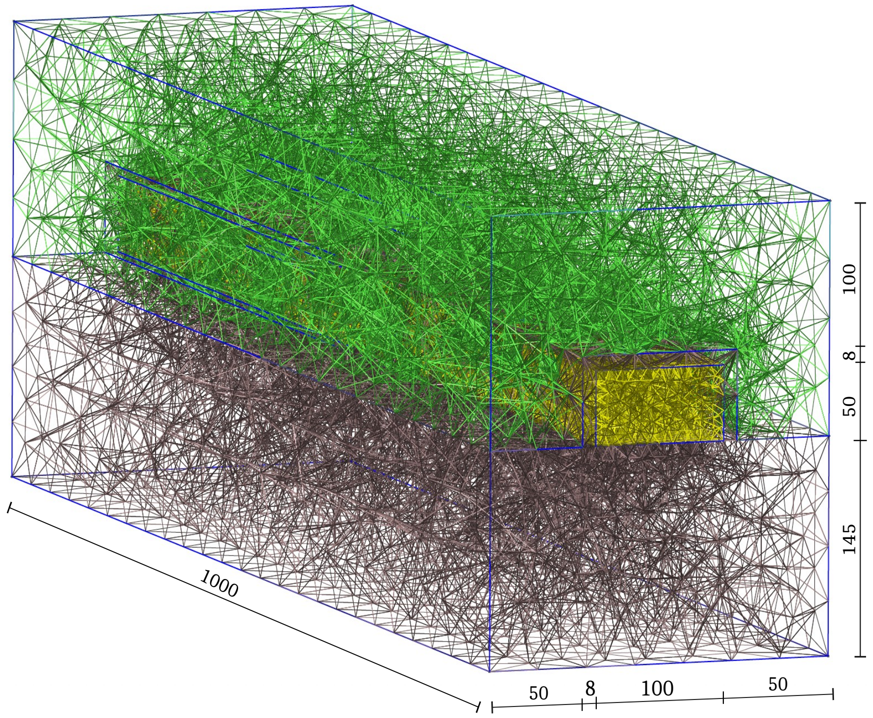

Figure 5. A 3D schematic of the sensor device including the dimensions and tetrahedral meshes for the discretization. All values are in nanometers.

5.1. Model Verification

As the first step, we verify the model accuracy with the experiments. We compute the electrical current I with respect to different gate voltages V_G where the source-to-drain voltage $V_\mathrm{SD} = 0.2$ V, doping concentration $C_\mathrm{dop} = 1 \times 10^{16}\,\mathrm{cm}^{-3}$, and the thermal voltage $U_T = 0.021$ V. The experimental data are taken form [20]. In order to solve the nonlinear coupled system of equations, a Scharfetter–Gummel-type iteration is used. Figure 6 shows the current as a function gate voltage varying between $V_\mathrm{G} = -1\,\mathrm{V}$ and $V_\mathrm{G} = -3.5\,\mathrm{V}$ for experimental and simulation values. These results indicate that the DDP system is reliable and will be used for the next simulations.

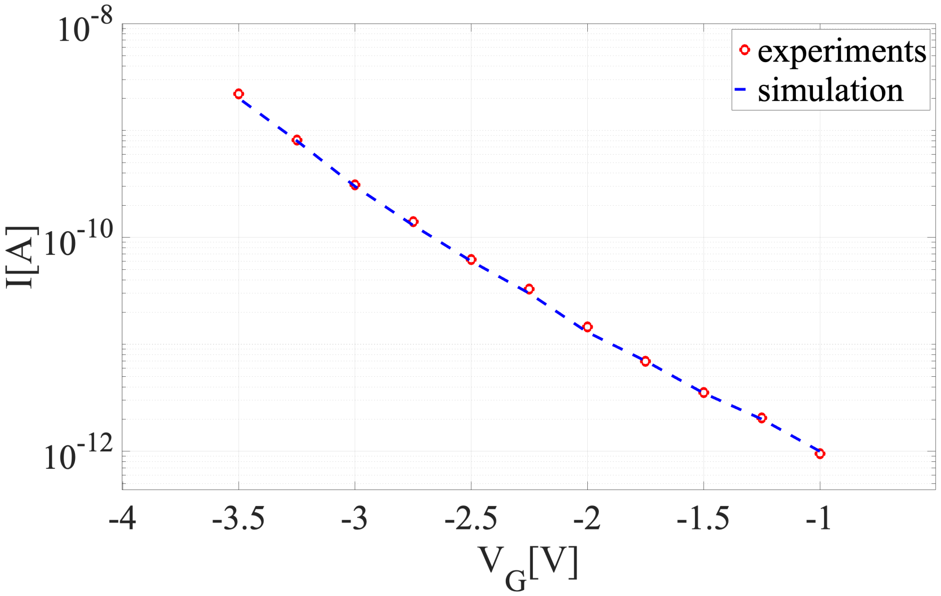

Figure 6. A comparison between the experimental [20] and simulation current.

5.2. Bayesian Inversion

The molecules are negatively charged (here, −15 q is used); however, an accurate estimation of the molecule charge density will be necessary. In semiconductor devices, in order to enhance the conductivity, impurity atoms are added to the silicon lattice, namely the doping process. Higher doping concentration will improve the transistor conductivity; however, the device will be less sensitive to the charged molecules. Physically, doping concentration (as a macroscopic quantity) denotes the average amount of the dopants. We implemented a delayed rejection adaptive Metropolis (DRAM) [14] and the Metropolis–Hastings algorithm [1] to infer doping concentration, molecule charge density, and probe–target density. The efficiency of the EnKF-MCMC compared to these algorithms is studied in [26]. Therefore, we employ the Kalman filter for the proposal adaptation. We performed the MCMC algorithm with N = 10,000 iterations, and a uniform prior density is used. The computational aspects are summarized in Table 1.

The back-propagation error is an efficient algorithm for the training of neural networks where we compute the gradient of the loss function with respect to the weights of the network.

Table 1. The computational features and the results of the Bayesian inversion.

Parameter	Min	Max	EnKF (Median)	True Values	Acceptance Rate
C_{dop} (cm^3)	1×10^{15}	5×10^{16}	9.4×10^{15}	1×10^{16}	91%
ρ (q/nm^2)	−5	1	−1.55	−1.5	86%

Employing a footprint of 10 nm for the molecules [20,49] gives rise to a surface charge of −1.5 q/nm^2. In the experiments, a doping concentration of 1×10^{16} is used in the transducer (both values are selected as the true values). The posterior densities are shown in Figure 7. As expected, the posterior densities are around the true values. Regarding the surface charge, we have a normal distribution, and the charge cannot be positive (which is reasonable due to using P-type FET). For the doping concentration, the distribution points out that for C_{dop} more than 2×10^{16}, the sensitivity will reduce significantly, and almost all of the candidates are rejected.

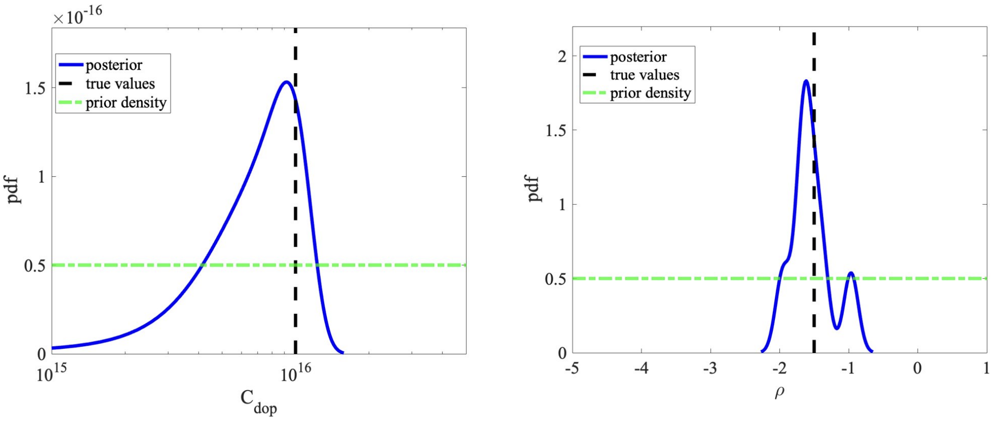

Figure 7. The posterior density of doping concentration (**left**) and surface charge density (**right**) using EnKF-MCMC. The units are C_{dop} (cm^3) and ρ (q/nm^2).

5.3. Machine Learning Based on MFNNs

In this section, we employ MFNNs to train the machine according to available information from the sensors. The effective physical/geometrical parameters will have a nonlinear effect on the device output. For instance, for a doping concentration of more than $C_{dop} = 2 \times 10^{16}$, the current will increase sharply, which is compatible with the results in Bayesian inversion (Figure 7). Due to this nonlinear behavior, the MFNNs algorithm is chosen to monitor the data accuracy and reliability and predict the sensor behavior.

More hidden layers will facilitate the convergence to the desired trajectory; however, it will increase dramatically the computational costs (e.g., computational time). In this work, we use two hidden layers for the MFNNs algorithm to strike a balance between complexity and efficiency. The procedure is shown in Figure 8. We define five specific scenarios according to the number of inputs. In Case 1, we only have one input (V_g) varying between −1 V and −5 V, where other parameters including insulator thickness, nanowire width (N_W), doping concentration, and nanowire height (N_H) are constant. In Case 5, we have five inputs, and the output is the calculated electrical current. Table 2 shows the range of the parameters used for different cases.

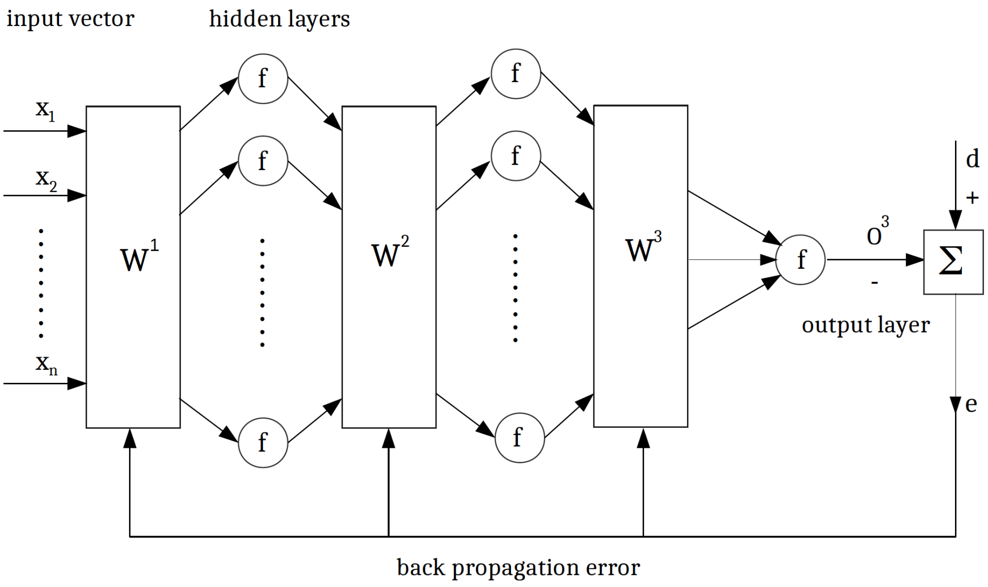

Figure 8. The structure of the MFNNs algorithm.

Table 2. The range of parameters used to compute the electrical current in different cases.

Cases	Inputs	V_g [V]	SiO$_2$ [nm]	N$_W$ [nm]	C_{dop} [cm^3]	N$_H$ [nm]
Case 1	1	$\mathcal{U}(-1,-5)$	8	100	1×10^{16}	50
Case 2	2	$\mathcal{U}(-1,-5)$	$\mathcal{U}(5,15)$	100	1×10^{16}	50
Case 3	3	$\mathcal{U}(-1,-5)$	$\mathcal{U}(5,15)$	$\mathcal{U}(80,120)$	1×10^{16}	50
Case 4	4	$\mathcal{U}(-1,-5)$	$\mathcal{U}(5,15)$	$\mathcal{U}(80,120)$	$\mathcal{U}(1\times 10^{15}, 5\times 10^{16})$	50
Case 5	5	$\mathcal{U}(-1,-5)$	$\mathcal{U}(5,15)$	$\mathcal{U}(80,120)$	$\mathcal{U}(1\times 10^{15}, 5\times 10^{16})$	$\mathcal{U}(40,60)$

The MFNNs algorithm is trained with two learning rates (i.e., $\eta = 0.1$ and $\eta = 0.2$) and different numbers of epochs. Here, we use 75% of the samples for data training and 25% of the samples for data testing. The numbers of epochs and neurons in the 1st and 2nd hidden layers are given in Table 3. The sigmoid function is used as an activation function in hidden and output layers. In order to verify the efficiency/accuracy of the MFNNs structure algorithm, for different cases, we compare the output of the machine learning algorithm with the desired trajectories (computed currents). We have the relative MSE for the test and training process and performed a linear regression test to explain the relation between the targets and MFNNs output. Figures 9 and 10 show the results for Cases 1–5, where in all cases, there is a good agreement between the machine learning output and the sensor data.

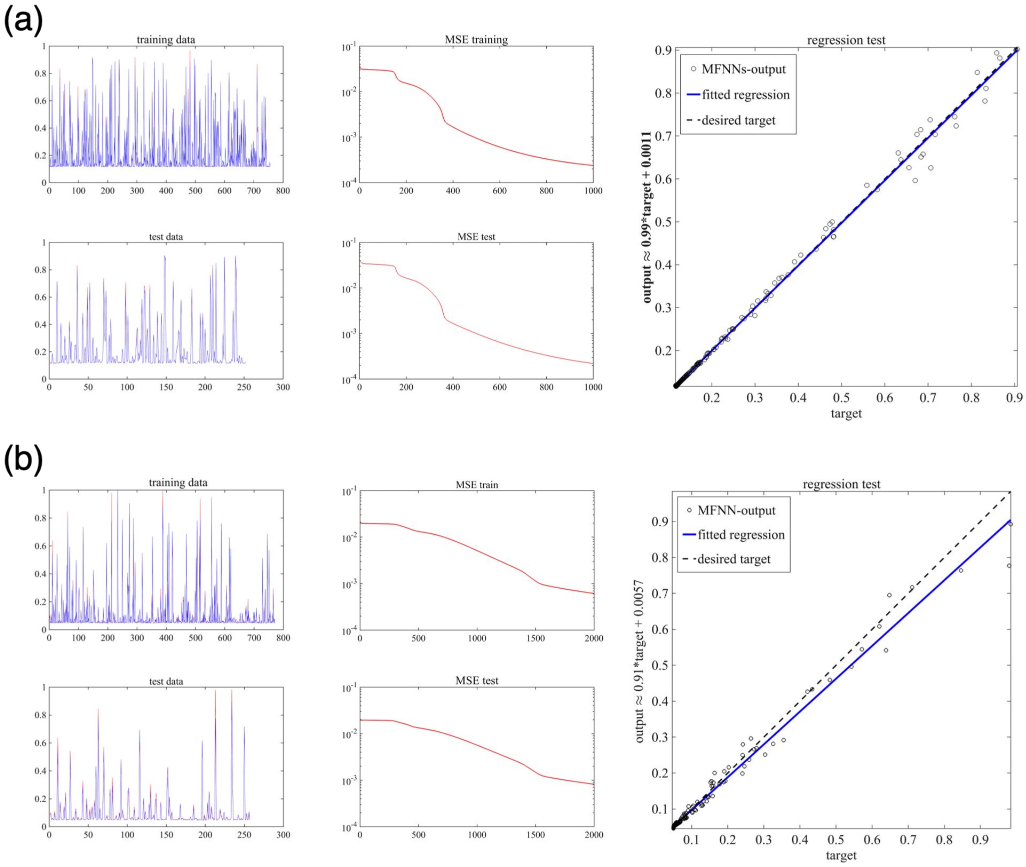

Figure 9. *Cont.*

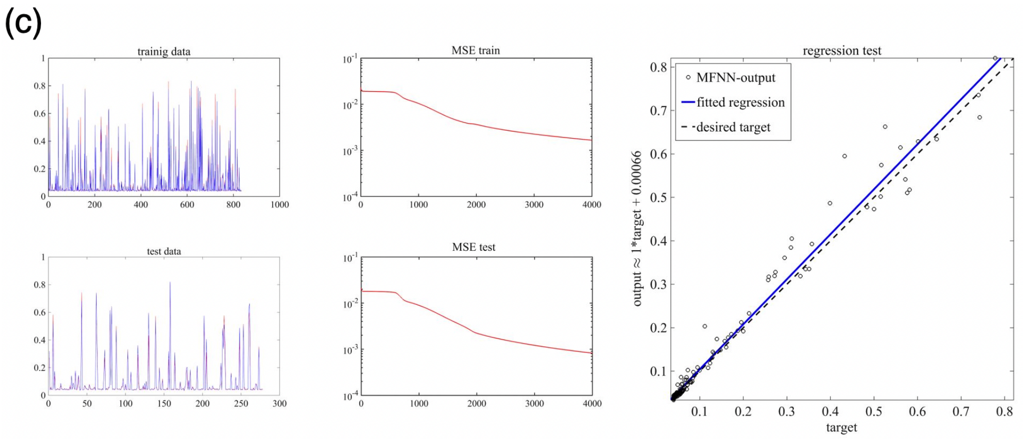

Figure 9. The performance of MFNNs algorithm for Case 1 (**a**), Case 2 (**b**), and Case 3 (**c**). In the first column, the desired trajectories (shown in blue) are compared with the MFNN output (shown in red). In the second column, we have the relative MSE, and the regression test is given in the third column.

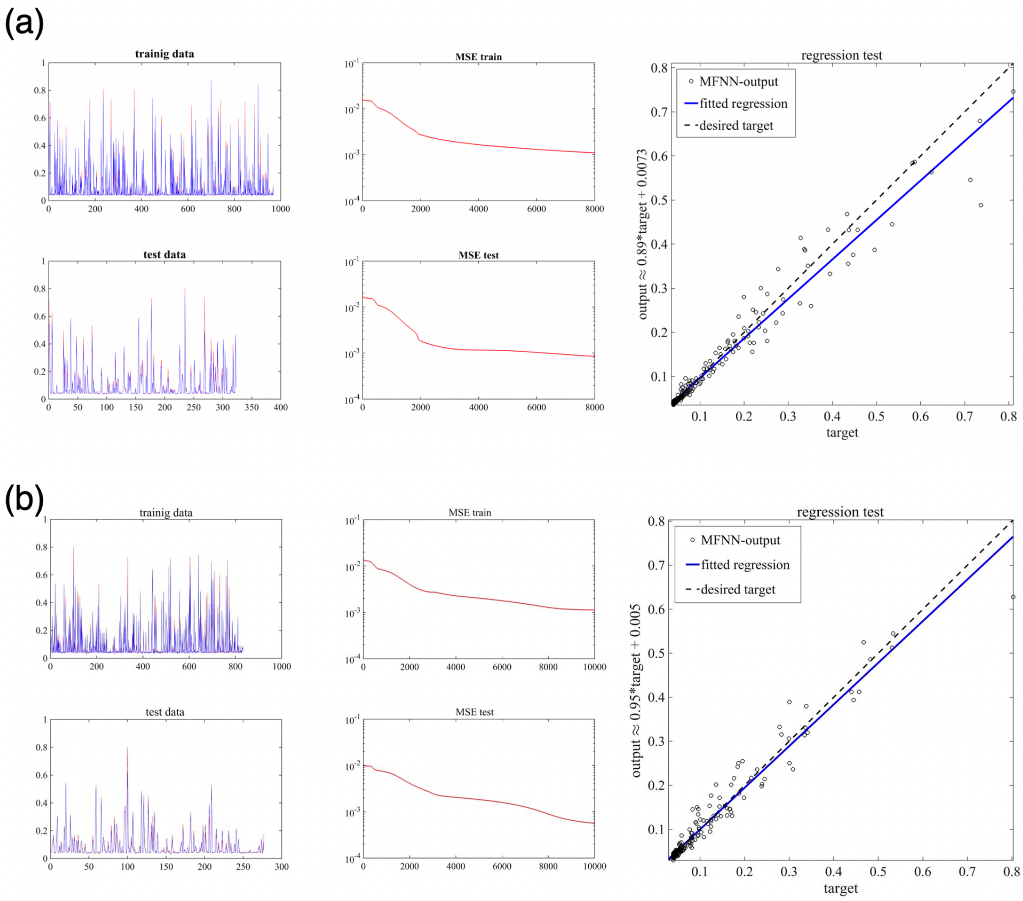

Figure 10. The performance of the MFNNs algorithm for Case 4 (**a**) and Case 5 (**b**). In the first column, the desired trajectories (shown in blue) are compared with the MFNN output (shown in red). In the second column, we have the relative MSE, and the regression test is given in the third column.

Table 3. The features of the MFNNs algorithm including the MSE of training and test processes.

Case	No. Neurons in 1st Hidden Layer	No. Neurons in 2nd Hidden Layer	MSE-Train	MSE-Test	No. Epochs	η
1	10	4	0.00057	0.00061	1000	0.1
2	20	7	0.00147	0.00184	2000	0.2
3	20	7	0.00181	0.000836	4 000	0.2
4	20	7	0.000842	0.000517	8 000	0.2
5	20	7	0.0011	0.000058	10 000	0.2

6. Conclusions

In this work, we introduced a computational framework for modeling charge transport and electrostatic potential distribution in SiNW-FETs in order to enable the rational design of this sensor technology. The PDE-based model has been verified with the experimental data and showed its accuracy. Bayesian inversion can be used to determine quantities of interest such as molecule concentrations, surface charges, and doping concentrations.

Our approach and results can be extended to different types of sensors including plasma resonance-based biosensors, fluorescence-based sensors, and electrochemiluminescence-based biosensors that are used to detect biomarkers.

Finally, machine learning algorithms based on MFNNs have been developed for SiNW-FETs. Here, we use two hidden layers to deal with the nonlinear behavior of the current (with respect to the input parameters), where the method shows its computational efficiency. We used 75% of the data to train the machine and the remaining 25% for testing. In both cases, the obtained MSE shows the convergence to the desired trajectory. The results indicate that MFNNs are a suitable machine learning algorithm for SiNW-FETs and can be used to predict the sensor output behavior as a compact model.

Author Contributions: Writing—review & editing, A.K., M.P., M.T. and C.H. All authors have read and agreed to the published version of the manuscript.

Funding: C. Heitzinger and A. Khodadadian acknowledge support by FWF START project no. Y660 *PDE Models for Nanotechnology*. M. Parvizi acknowledges the financial support of the *Alexander von Humbold Foundation* project named $\mathcal{H}-$matrix approximability of the inverses for FEM, BEM and FEM–BEM coupling of the electromagnetic problems. She is affiliated to the Cluster of Excellence PhoenixD (EXC 2122, Project ID 390833453).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Mirsian, S.; Khodadadian, A.; Hedayati, M.; Manzour-ol Ajdad, A.; Kalantarinejad, R.; Heitzinger, C. A new method for selective functionalization of silicon nanowire sensors and Bayesian inversion for its parameters. *Biosens. Bioelectron.* **2019**, *142*, 111527. [CrossRef]
2. Kuang, T.; Chang, L.; Peng, X.; Hu, X.; Gallego-Perez, D. Molecular beacon nano-sensors for probing living cancer cells. *Trends Biotechnol.* **2017**, *35*, 347–359. [CrossRef]
3. Hahm, J.i.; Lieber, C.M. Direct ultrasensitive electrical detection of DNA and DNA sequence variations using nanowire nanosensors. *Nano Lett.* **2004**, *4*, 51–54. [CrossRef]
4. Zhang, G.J.; Chua, J.H.; Chee, R.E.; Agarwal, A.; Wong, S.M. Label-free direct detection of MiRNAs with silicon nanowire biosensors. *Biosens. Bioelectron.* **2009**, *24*, 2504–2508. [CrossRef] [PubMed]
5. Choi, J.H.; Kim, H.; Kim, H.S.; Um, S.H.; Choi, J.W.; Oh, B.K. MMP-2 detective silicon nanowire biosensor using enzymatic cleavage reaction. *J. Biomed. Nanotechnol.* **2013**, *9*, 732–735. [CrossRef] [PubMed]
6. De Santiago, F.; Trejo, A.; Miranda, A.; Salazar, F.; Carvajal, E.; Pérez, L.; Cruz-Irisson, M. Carbon monoxide sensing properties of B-, Al-and Ga-doped Si nanowires. *Nanotechnology* **2018**, *29*, 204001. [CrossRef]
7. Song, X.; Hu, R.; Xu, S.; Liu, Z.; Wang, J.; Shi, Y.; Xu, J.; Chen, K.; Yu, L. Highly sensitive ammonia gas detection at room temperature by integratable silicon nanowire field-effect sensors. *ACS Appl. Mater. Interfaces* **2021**, *13*, 14377–14384. [CrossRef]

8. Duan, X.; Li, Y.; Rajan, N.K.; Routenberg, D.A.; Modis, Y.; Reed, M.A. Quantification of the affinities and kinetics of protein interactions using silicon nanowire biosensors. *Nat. Nanotechnol.* **2012**, *7*, 401–407. [CrossRef]
9. Patolsky, F.; Zheng, G.; Lieber, C.M. Fabrication of silicon nanowire devices for ultrasensitive, label-free, real-time detection of biological and chemical species. *Nat. Protoc.* **2006**, *1*, 1711–1724. [CrossRef]
10. Stern, E.; Vacic, A.; Rajan, N.K.; Criscione, J.M.; Park, J.; Ilic, B.R.; Mooney, D.J.; Reed, M.A.; Fahmy, T.M. Label-free biomarker detection from whole blood. *Nat. Nanotechnol.* **2010**, *5*, 138–142. [CrossRef]
11. Chua, J.H.; Chee, R.E.; Agarwal, A.; Wong, S.M.; Zhang, G.J. Label-free electrical detection of cardiac biomarker with complementary metal-oxide semiconductor-compatible silicon nanowire sensor arrays. *Anal. Chem.* **2009**, *81*, 6266–6271. [CrossRef]
12. Chen, K.I.; Li, B.R.; Chen, Y.T. Silicon nanowire field-effect transistor-based biosensors for biomedical diagnosis and cellular recording investigation. *Nano Today* **2011**, *6*, 131–154. [CrossRef]
13. Gao, A.; Lu, N.; Wang, Y.; Dai, P.; Li, T.; Gao, X.; Wang, Y.; Fan, C. Enhanced sensing of nucleic acids with silicon nanowire field effect transistor biosensors. *Nano Lett.* **2012**, *12*, 5262–5268. [CrossRef]
14. Khodadadian, A.; Stadlbauer, B.; Heitzinger, C. Bayesian inversion for nanowire field-effect sensors. *J. Comput. Electron.* **2020**, *19*, 147–159. [CrossRef]
15. Taghizadeh, L.; Khodadadian, A.; Heitzinger, C. The optimal multilevel Monte-Carlo approximation of the stochastic drift–diffusion-Poisson system. *Comput. Methods Appl. Mech. Eng.* **2017**, *318*, 739–761. [CrossRef]
16. Khodadadian, A.; Hosseini, K.; Manzour-ol Ajdad, A.; Hedayati, M.; Kalantarinejad, R.; Heitzinger, C. Optimal design of nanowire field-effect troponin sensors. *Comput. Biol. Med.* **2017**, *87*, 46–56. [CrossRef]
17. Pittino, F.; Selmi, L. Use and comparative assessment of the CVFEM method for Poisson–Boltzmann and Poisson–Nernst–Planck three dimensional simulations of impedimetric nano-biosensors operated in the DC and AC small signal regimes. *Comput. Methods Appl. Mech. Eng.* **2014**, *278*, 902–923. [CrossRef]
18. Khodadadian, A.; Heitzinger, C. Basis adaptation for the stochastic nonlinear Poisson–Boltzmann equation. *J. Comput. Electron.* **2016**, *15*, 1393–1406. [CrossRef]
19. Khodadadian, A.; Taghizadeh, L.; Heitzinger, C. Three-dimensional optimal multi-level Monte–Carlo approximation of the stochastic drift–diffusion–Poisson system in nanoscale devices. *J. Comput. Electron.* **2018**, *17*, 76–89. [CrossRef]
20. Baumgartner, S.; Heitzinger, C.; Vacic, A.; Reed, M.A. Predictive simulations and optimization of nanowire field-effect PSA sensors including screening. *Nanotechnology* **2013**, *24*, 225503. [CrossRef]
21. Hastings, W.K. Monte Carlo sampling methods using Markov chains and their applications *Biometrika* **1970**, *57*, 97–109. [CrossRef]
22. Haario, H.; Saksman, E.; Tamminen, J. Adaptive proposal distribution for random walk Metropolis algorithm. *Comput. Stat.* **1999**, *14*, 375–395. [CrossRef]
23. Green, P.J.; Mira, A. Delayed rejection in reversible jump Metropolis–Hastings. *Biometrika* **2001**, *88*, 1035–1053. [CrossRef]
24. Haario, H.; Laine, M.; Mira, A.; Saksman, E. DRAM: Efficient adaptive MCMC. *Stat. Comput.* **2006**, *16*, 339–354. [CrossRef]
25. Evensen, G. The ensemble Kalman filter for combined state and parameter estimation. *IEEE Control Syst. Mag.* **2009**, *29*, 83–104. [CrossRef]
26. Noii, N.; Khodadadian, A.; Ulloa, J.; Aldakheel, F.; Wick, T.; François, S.; Wriggers, P. Bayesian Inversion with Open-Source Codes for Various One-Dimensional Model Problems in Computational Mechanics. *Arch. Comput. Methods Eng.* **2022**, 1–34. [CrossRef]
27. Schackart, K.E.; Yoon, J.Y. Machine learning enhances the performance of bioreceptor-free biosensors. *Sensors* **2021**, *21*, 5519. [CrossRef] [PubMed]
28. Cui, F.; Yue, Y.; Zhang, Y.; Zhang, Z.; Zhou, H.S. Advancing biosensors with machine learning. *ACS Sensors* **2020**, *5*, 3346–3364. [CrossRef] [PubMed]
29. Albrecht, T.; Slabaugh, G.; Alonso, E.; Al-Arif, S.M.R. Deep learning for single-molecule science. *Nanotechnology* **2017**, *28*, 423001. [CrossRef] [PubMed]
30. Jin, X.; Liu, C.; Xu, T.; Su, L.; Zhang, X. Artificial intelligence biosensors: Challenges and prospects. *Biosens. Bioelectron.* **2020**, *165*, 112412. [CrossRef] [PubMed]
31. Raji, H.; Tayyab, M.; Sui, J.; Mahmoodi, S.R.; Javanmard, M. Biosensors and machine learning for enhanced detection, stratification, and classification of cells: A review. *arXiv* **2021**, arXiv:2101.01866.
32. Rivera, E.C.; Swerdlow, J.J.; Summerscales, R.L.; Uppala, P.P.T.; Maciel Filho, R.; Neto, M.R.; Kwon, H.J. Data-driven modeling of smartphone-based electrochemiluminescence sensor data using artificial intelligence. *Sensors* **2020**, *20*, 625. [CrossRef]
33. Khodadadian, A.; Parvizi, M.; Heitzinger, C. An adaptive multilevel Monte Carlo algorithm for the stochastic drift–diffusion–Poisson system. *Comput. Methods Appl. Mech. Eng.* **2020**, *368*, 113163. [CrossRef]
34. Khodadadian, A.; Taghizadeh, L.; Heitzinger, C. Optimal multilevel randomized quasi-Monte-Carlo method for the stochastic drift–diffusion-Poisson system. *Comput. Methods Appl. Mech. Eng.* **2018**, *329*, 480–497. [CrossRef]
35. Baumgartner, S.; Heitzinger, C. Existence and local uniqueness for 3d self-consistent multiscale models of field-effect sensors. *Commun. Math. Sci.* **2012**, *10*, 693–716. [CrossRef]
36. Cockburn, B.; Triandaf, I. Convergence of a finite element method for the drift-diffusion semiconductor device equations: The zero diffusion case. *Math. Comput.* **1992**, *59*, 383–401. [CrossRef]
37. Chen, Z.; Cockburn, B. Analysis of a finite element method for the drift-diffusion semiconductor device equations: The multidimensional case. *Numer. Math.* **1995**, *71*, 1–28. [CrossRef]

38. Zlámal, M. Finite element solution of the fundamental equations of semiconductor devices. I. *Math. Comput.* **1986**, *46*, 27–43. [CrossRef]
39. Zhang, J.; Vrugt, J.A.; Shi, X.; Lin, G.; Wu, L.; Zeng, L. Improving Simulation Efficiency of MCMC for Inverse Modeling of Hydrologic Systems with a Kalman-Inspired Proposal Distribution. *Water Resour. Res.* **2020**, *56*, e2019WR025474. [CrossRef]
40. Bebis, G.; Georgiopoulos, M. Feed-forward neural networks. *IEEE Potentials* **1994**, *13*, 27–31. [CrossRef]
41. Svozil, D.; Kvasnicka, V.; Pospichal, J. Introduction to multi-layer feed-forward neural networks. *Chemom. Intell. Lab. Syst.* **1997**, *39*, 43–62. [CrossRef]
42. Sazli, M.H. A brief review of feed-forward neural networks. *Commun. Fac. Sci. Univ. Ank. Ser. A2-A3 Phys. Sci. Eng.* **2006**, *50*. [CrossRef]
43. Dolinsky, T.J.; Czodrowski, P.; Li, H.; Nielsen, J.E.; Jensen, J.H.; Klebe, G.; Baker, N.A. PDB2PQR: Expanding and upgrading automated preparation of biomolecular structures for molecular simulations. *Nucleic Acids Res.* **2007**, *35*, W522–W525. [CrossRef] [PubMed]
44. Li, H.; Robertson, A.D.; Jensen, J.H. Very fast empirical prediction and rationalization of protein pKa values. *Proteins: Struct. Funct. Bioinform.* **2005**, *61*, 704–721. [CrossRef] [PubMed]
45. Søndergaard, C.R.; Olsson, M.H.; Rostkowski, M.; Jensen, J.H. Improved treatment of ligands and coupling effects in empirical calculation and rationalization of pKa values. *J. Chem. Theory Comput.* **2011**, *7*, 2284–2295. [CrossRef] [PubMed]
46. Olsson, M.H.; Søndergaard, C.R.; Rostkowski, M.; Jensen, J.H. PROPKA3: Consistent treatment of internal and surface residues in empirical pKa predictions. *J. Chem. Theory Comput.* **2011**, *7*, 525–537. [CrossRef] [PubMed]
47. Bulyha, A.; Heitzinger, C. An algorithm for three-dimensional Monte-Carlo simulation of charge distribution at biofunctionalized surfaces. *Nanoscale* **2011**, *3*, 1608–1617. [CrossRef]
48. Heitzinger, C.; Liu, Y.; Mauser, N.J.; Ringhofer, C.; Dutton, R.W. Calculation of fluctuations in boundary layers of nanowire field-effect biosensors. *J. Comput. Theor. Nanosci.* **2010**, *7*, 2574–2580. [CrossRef]
49. Punzet, M.; Baurecht, D.; Varga, F.; Karlic, H.; Heitzinger, C. Determination of surface concentrations of individual molecule-layers used in nanoscale biosensors by in situ ATR-FTIR spectroscopy. *Nanoscale* **2012**, *4*, 2431–2438. [CrossRef]