Computers and Chemical Engineering 165 (2022) 107952

Contents lists available at ScienceDirect

Computers and Chemical Engineering

journal homepage: www.elsevier.com/locate/compchemeng

A general deep hybrid model for bioreactor systems: Combining first principles with deep neural networks

José Pinto [a], Mykaella Mestre [a], J. Ramos [a], Rafael S. Costa [a], Gerald Striedner [b], Rui Oliveira [a,*]

[a] LAQV-REQUIMTE, NOVA Science and Engineering School, NOVA University of Lisbon, Portugal
[b] Department of Biotechnology, BOKU University, Vienna, Austria

ARTICLE INFO

Keywords:
Hybrid semiparametric modeling
Deep neural networks
ADAM algorithm
Stochastic regularization
Bioprocess dynamics
Pichia pastoris

ABSTRACT

Numerous studies have reported the use of hybrid semiparametric systems that combine shallow neural networks with First Principles for bioprocess modeling. Here we revisit the general bioreactor hybrid model and introduce some deep learning techniques. Multi-layer networks with varying depths were combined with First Principles equations in the form of deep hybrid models. Deep learning techniques, namely the adaptive moment estimation method (ADAM), stochastic regularization and depth-dependent weights initialization were evaluated in a hybrid modeling context. Modified sensitivity equations are proposed for the computation of gradients in order to reduce CPU time for the training of deep hybrid models. The methods are illustrated with applications to a synthetic dataset and a pilot 50 L MUT+ *Pichia pastoris* process expressing a single chain antibody fragment. All in all, the results point to a systematic generalization improvement of deep hybrid models over its shallow counterpart. Moreover, the CPU cost to train the deep hybrid models is shown to be lower than for the shallow counterpart. In the pilot 50L MUT+ *Pichia* pastoris data set, the prediction accuracy was increased by 18.4% and the CPU decreased by 43.4%.

1. Introduction

The first steps towards the integration of mechanistic abstraction and neural networks in process systems engineering were taken in the early 90's with the pioneering works of Psichogios and Ungar (1992), Su and Mcavoy (1993), Schubert et al. (1994) and Thompson and Kramer (1994). The main motivation was to overcome neural networks limitations, namely the (i) inability to comply with process constraints, (ii) the tendency for data overfitting, and (iii) the poor predictive power outside the training-validation domain. Thompson and Kramer (1994) framed this problem as hybrid semiparametric systems, whereby parametric functions with fixed structure stemming from prior process knowledge (e.g., macroscopic material balance equations) are combined in series or in parallel with nonparametric functions (e.g., neural networks) identified from process data. Numerous bioprocess modeling studies followed (e.g., Preusting et al. (1996), van Can et al. (1998), Chen et al. (2000), Galvanauskas et al. (2004), Oliveira (2004), Teixeira et al. (2007), Fiedler and Schuppert (2008), von Stosch et al. (2011), Ferreira et al. (2014), Pinto et al. (2019), O'Brien et al. (2021) and Bayer et al. (2021)) highlighting the advantages of the hybrid technique, which may be summarized as a more rational usage of prior knowledge eventually translating into more accurate, transparent and robust process models.

The vast majority of hybrid modeling studies explored the combination of conservation laws and shallow neural networks (see review by von Stosch et al., 2014). Recent advances in deep learning have however demonstrated that neural networks with multiple hidden layers (deep networks) are advantageous over their shallow counterparts. Shallow and deep networks are both universal function approximators, but deep networks are able to approximate compositional functions with exponentially lower number of parameters and sample complexity (Delalleau and Bengio, 2011; Eldan and Shamir, 2016; Liang and Srikant, 2017) and are less prone to overfitting (Mhaskar and Poggio, 2016). The shift from shallow to deep network architectures has been triggered by the development of stochastic gradient descent training algorithms, particularly the ADAM method (Kingma and Ba, 2014). ADAM is a first-order gradient-based method for stochastic objective functions based on adaptive estimates of lower-order moments. The data subsampling along with the learning rate adaptation at each iteration resulted in a simple and robust training method that is less sensitive to gradient attenuation and to the convergence to local optima. Stochastic regularization based on weights dropout has been shown to effectively avoid overfitting in deep learning (Hinton et al., 2012; Srivastava et al.,

* Corresponding author.
E-mail address: rmo@fct.unl.pt (R. Oliveira).

https://doi.org/10.1016/j.compchemeng.2022.107952
Received 17 June 2022; Received in revised form 31 July 2022; Accepted 8 August 2022
Available online 9 August 2022

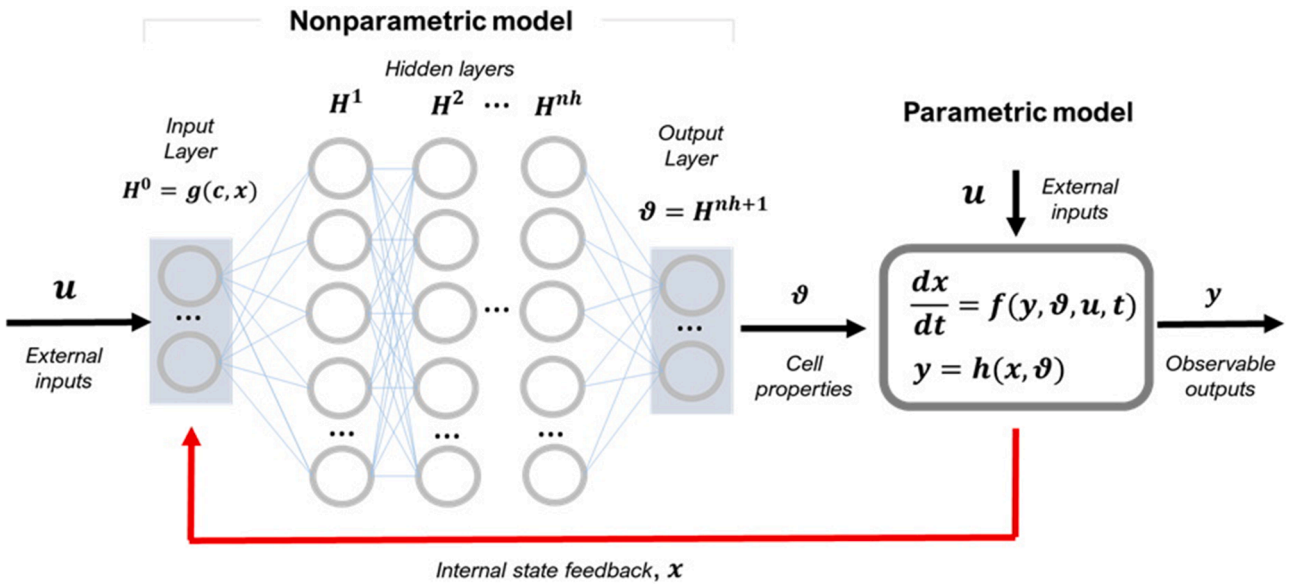

Fig. 1. Schematic representation of the general deep hybrid model for bioreactor systems. The model is dynamic in nature with state vector, **x,** and observable outputs **y**. The model has a parametric component (functions f(.) and h(.)) with fixed mathematical structure determined by First Principles (typically material/energy balance equations). Some cellular properties are modelled by a deep feedforward neural network with multiple hidden layers as function of the process state, **x**, and external inputs, **u**. The deep neural network is a nonparametric model component with loose structure that must be identified from process data given the absence of explanatory mechanisms for that particular part of the model.

2014). Stochastic regularization is frequently associated with stochastic gradient descent methods to prevent overfitting and to improve generalization properties (Koutsoukas et al., 2017).

Only very recently the deep learning advances are penetrating the hybrid modeling field. Bangi and Kwon (2020) proposed a hybrid model for a hydraulic fracturing process that combines a First Principles model with a deep neural network. A fully connected network with 5 layers (1 × 20 × 20 × 20 × 1), hyperbolic tangent activation (*tanh*) in the 3 hidden layers and linear activation in the input/output layers, was adopted. The Levenberg–Marquardt algorithm and finite difference-based sensitivity analysis were adopted to train the hybrid model. The resulting hybrid model had superior extrapolation properties compared to a purely data-driven deep neural network model. Following a similar approach, Shah et al. (2022) developed a deep hybrid model for an industrial fermentation process. Lee et al. (2020) developed of a hybrid deep model of an intracellular signaling pathway using a neural network with 2 hidden layers. Bangi et al. (2022) proposed the Universal Differential Equations (UDE) formalism for mixing the information of physical laws and scientific models with data-driven machine learning approaches. They applied it to a *Saccharomyces cerevisiae* batch fermentation process. Merkelbach et al. (2022) have develop a software package called HybridML that uses TensorFlow for artificial neural network training and Casadi to integrate ordinary differential equations.

In this study, we revisit the general bioreactor hybrid model (Oliveira, 2004; Teixeira et al., 2007; von Stosch et al., 2011; Ferreira et al., 2014; Pinto et al., 2019) and extend it to deep learning. More specifically, we explore deep learning techniques in a hybrid semiparametric modeling context, such as deep feedforward neural networks with varying depths, the rectified linear unit (ReLU) activation function, dropout regularization of network weights, and stochastic training with the ADAM method. These techniques are applied to two case studies and are benchmarked against the traditional shallow hybrid modeling approach.

2. Materials and methods

2.1. General deep hybrid model for bioreactor systems

A stirred tank bioreactor can be generically represented by the hybrid model structure of Fig. 1. The dynamics of state variables are modelled by a system of ordinary differential equations (ODEs) derived from macroscopic material balances and/or intracellular material balances and/or other physical assumptions. These equations take the following general form:

$$\frac{dx}{dt} = f(x, \vartheta, u, t) \tag{1a}$$

$$y = h(x, \vartheta) \tag{1b}$$

with t the independent variable time, $x(t)$ the process state vector, $u(t)$ the vector of external inputs (feed rates, temperature, pH, etc), ϑ a vector of process variables with unknown defining functions, and y the vector of measured variables. Eqs. (1a) and (1b) are the state-space model and measurement model, respectively. The functions $f(.)$ and $h(.)$ are of parametric nature thus with fixed structure stemming from prior knowledge. They are typically set by material and/or energy balance equations of extracellular and intracellular variables (as shown in the case studies). Some relevant bioprocess variables may be less defined in terms of explanatory mechanisms and/or rely on loose assumptions. Typical examples are biological reaction kinetics or product quality attributes, which are difficult to establish on a mechanistic basis. In the general hybrid model, such properties are defined as loose functions, $\vartheta(\cdot)$ (typically of the process state and external inputs), with unknown structure, i.e. nonparametric functions without physical meaning. Among the many possibilities to define $\vartheta(\cdot)$, the preferred approach (in a hybrid modeling context) has been by far the feedforward perceptron networks with 3 layers only (see review by von Stosch et al., 2014). In the present study, the more general case of deep multi-layer perceptron networks with arbitrary number of *nh* hidden layers is explored, stated as follows:

$$H^0 = g(x, u, t) \tag{2a}$$

$$H^i = \sigma\left(w^i \cdot H^{i-1} + b^i\right),\ i = 1, \ldots,\ nh \tag{2b}$$

$$\vartheta(\cdot) = w^{nh+1} \cdot H^{nh} + b^{nh+1} \tag{2c}$$

The input layer Eq. (2a)) typically receives information of the state variables, x, and/or external inputs, u (temperature, pH, etc,…) and/or process time, t. An optional non-linear pre-processing function, $g(x,u,t)$, may sometimes facilitate the identification of $\vartheta(\cdot)$, as for example concentration ratios are set as inputs to the neural network or other normalization rules (see von Stosch et al. (2016), Gnoth et al. (2008) and Gnoth et al. (2010)). Then, follows nh hidden layers (Eq. (2b)) with $\sigma(\cdot)$ the nodes transfer function (in this study either the *tanh* or the *ReLU*). Finally, the output layer has linear nodes (Eq. (2c)). The parameters $w = \{w^1, w^2, \ldots, w^{nh+1}\}$ and $b = \{b^1, b^2, \ldots, b^{nh+1}\}$ are the nodes connection weights that need to be identified from data during the training process. Presuming that initial conditions $x(t) = x_0$ and network weights $\omega = \{w,\ b\}$ are given, the deep hybrid model can be solved by numerical integration as an Initial Value Problem (IVP). In the present study, a Runge–Kutta 4th order ODE solver was adopted to integrate the system (Eqs. ((1) and (2)) and compute x, y and ϑ over time. All the code was implemented in MATLAB on a computer with Intel(R) Core(TM) i5-8265U CPU @ 1.60 GHz 1.80 GHz, and 24 GB of RAM.

2.2. Training method

2.2.1. Standard non-deep method

Hybrid bioreactor models are typically trained by indirect supervised learning with cross-validation to avoid overfitting (e.g., Psichogios and Ungar (1992), Oliveira (2004), Pinto et al. (2019) and von Stosch et al. (2014)). The data are partitioned in a training subset (for parameter estimation), a validation subset (stop criterion to avoid overfitting) and a test subset (to assess the predictive power). Partitioning is typically performed batch wise with the amount of data allocated in each partition depending on the objective of the study and on the amount of data available. The optimization of network parameters is performed over the training set only in a weighted least-squares sense:

$$WSSE = \frac{1}{T}\sum_{t=1}^{T}\frac{\left(y_t^* - y_t\right)^2}{\sigma_t^2} \tag{3}$$

with T the number of training patterns, y_t^* the measured variables at time t, y_t the corresponding model prediction and σ_t the measurement standard deviation. This method is called indirect because the loss function is not directly linked to the neural network outputs, ϑ. The Levenberg–Marquardt method (LMM), has been shown to solve very effectively the indirect training problem (Eqs. (1)–(3)) in the case of shallow hybrid models (Schubert et al., 1994; Oliveira, 2004). The LMM has also been used in a recent deep hybrid modeling study (Bangi and Kwon, 2020). The LMM convergence is improved if the sensitivity equations are applied to calculate the loss function gradients instead of numerical gradients (e.g., Psichogios and Ungar (1992), Schubert et al. (1994) and Oliveira (2004)). The sensitivity equations for the general hybrid have the following structure (for simplicity it is assumed that $y = x$):

$$g = \frac{\partial WSSE}{\partial \omega} = -2\sum_{t=1}^{T}\frac{y_t^* - y_t}{\sigma_t^2}\left(\frac{\partial x_t}{\partial w}\right) \tag{4a}$$

$$\frac{d\left(\frac{\partial x}{\partial w}\right)}{dt} = \left(\frac{\partial f}{\partial x}\right)\left(\frac{\partial x}{\partial w}\right) + \left(\frac{\partial f}{\partial x}\right) \tag{4b}$$

$$\left(\frac{\partial c}{\partial w}\right)|_{t=0} = 0 \tag{4c}$$

The sensitivity equations are obtained by differentiation of the state-space model (Eq. (1a)) in relation to the network parameters, w. For more details regarding the sensitivity equations in a hybrid modeling context see Psichogios and Ungar (1992) and Oliveira (2004). The integration of the sensitivity equations was performed in this study with a Runge–Kutta 4th order ODEs solver.

2.2.2. Stochastic adaptive moment estimation (ADAM) with semi-direct sensitivities

An important goal of this study is to compare the standard training method with state-of-the-art deep learning techniques in the context of hybrid modeling. Particularly, ADAM is considered a landmark in deep learning and was implemented here to train hybrid models. The ADAM method estimates the network parameters, $\omega = \{w,\ b\}$, through the first and second moments of the gradients of the loss function and a set of hyperparameters α, β_1 and β_2, representing the step size and exponential decays of the moment estimations (for details see Kingma and Ba (2014)). The loss function is the same as in the previous method (Eq. (3)). This results in the following implementation:

$$m_k = \frac{\beta_1 \cdot m_{k-1} + (1-\beta_1)\cdot g_k}{\left(1-\beta_1^k\right)} \tag{5a}$$

$$v_k = \frac{\beta_2 \cdot v_{k-1} + (1-\beta_2)\cdot g_k^2}{\left(1-\beta_2^k\right)} \tag{5b}$$

$$w_k = w_{k-1} - \frac{\alpha \cdot m_k}{\left(\sqrt{v_k} + \varepsilon\right)} \tag{5c}$$

with k the iteration number, m_k the first order moment of gradients, g_k the loss function gradients, v_k the second order moment of gradients. For the present study, the suggested default parameters of $\alpha = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\varepsilon = 10^{-8}$ were adopted (Kingma and Ba, 2014).

The gradients at each iteration are obtained by solving the sensitivity Eqs. (4a)–(4c). Because the CPU scales exponentially with the size of the network, a different approach to calculate the gradients was explored. Instead of computing the sensitivities of state variables in relation to network parameters, $\left(\frac{\partial x}{\partial w}\right)$, a semidirect approach was implemented where the sensitivities of state variables in relation to network outputs, $\left(\frac{\partial c}{\partial \vartheta}\right)$, are computed. The semidirect sensitivity equations are as follows (again assuming $y = x$):

$$\frac{\partial WSSE}{\partial \vartheta} = -2\sum_{t=1}^{T}\frac{y_t^* - y_t}{\sigma_t^2}\left(\frac{\partial x}{\partial \vartheta}\right) \tag{6a}$$

$$\frac{d\left(\frac{\partial x}{\partial \vartheta}\right)}{dt} = \left(\frac{\partial f}{\partial x}\right)\left(\frac{\partial x}{\partial \vartheta}\right) + \left(\frac{\partial f}{\partial \vartheta}\right) \tag{6b}$$

$$\left(\frac{\partial x}{\partial \vartheta}\right)|_{t=0} = 0 \tag{6c}$$

Finally, the loss function gradients $g = \frac{\partial WSSE}{\partial \omega}$ can be computed from the $\frac{\partial WSSE}{\partial \vartheta}$ sensitivity (Eq. (6a)) by the well-known error backpropagation algorithm through the network (Werbos, 1974). The main advantage of the semidirect method is that the number of ODEs for calculating the sensitivities is massively reduced and are independent of the size of the network. This results in a sizable CPU reduction as shown in the results section.

2.3. Case studies

2.3.1. Lee & Ramirez synthetic data set

A synthetic data set was generated based on the Lee & Ramirez bioreactor model (Lee and Ramirez, 1994). This model is frequently adopted as a benchmark to test different optimal control methods (e.g., Banga et al. (2005)). The objective in this case study is to train hybrid models on an information rich data set (time series data generated by statistical design of experiments) and then to assess if the trained hybrid

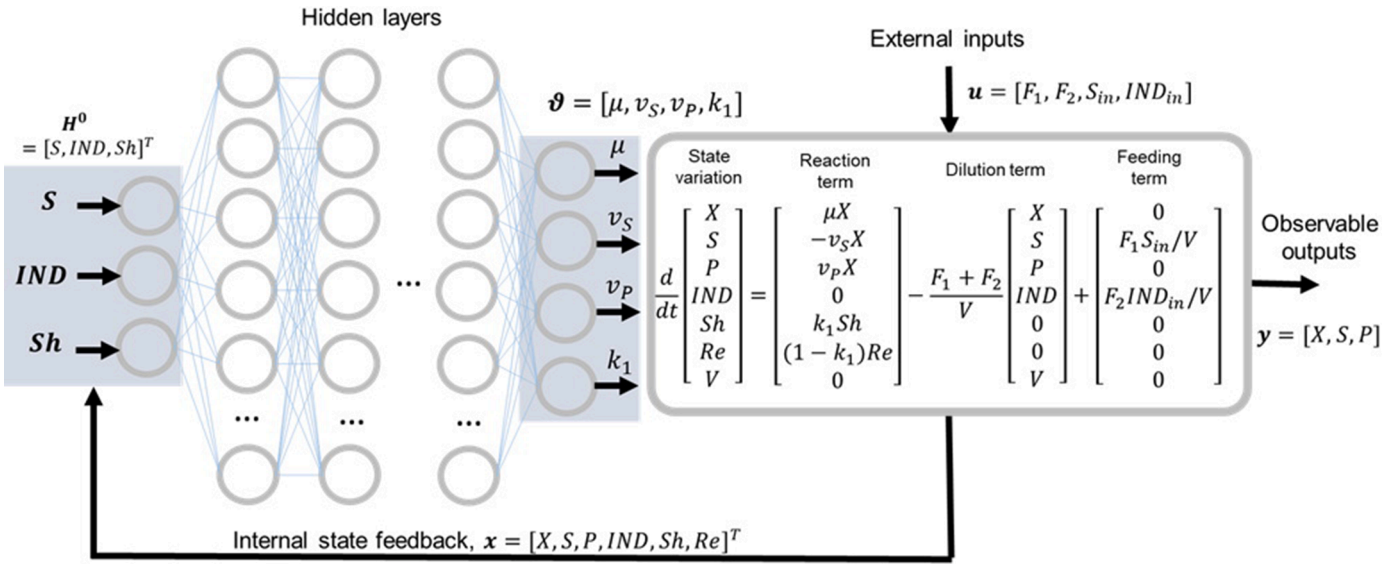

Fig. 2. Deep hybrid model structure for the Lee & Ramirez data set. The parametric component is established by a system of ODEs as described in Lee & Ramirez (1994). The specific biologic kinetics are considered mechanistically unknown thus modelled by a deep feedforward network. The job of this model is thus to "learn" from data the biologic kinetics under the constraint of dynamic material balance equations.

models are able to describe (extrapolate) the maximum productivity fed-batch obtained by optimal control studies (Lee and Ramirez, 1994).

The Lee & Ramirez model describes the dynamics of biomass (X), substrate concentration (S), inducer concentration (IND), product concentration (P), shock factor (Sh), recovery factor (Re) and reactor volume (V) in a recombinant *Escherichia coli* fed-batch process. Experiments were simulated dynamically for different conditions (see below) applying a Runge–Kutta 4th order ODEs solver. Samples were simulated with 1 h sampling time. Gaussian noise was added to "sampled" variables with standard deviations of 1.5 (X), 5(S) and 0.3(P) (10% of maximum concentration). As shown in the results section, modeling errors were calculated based on the noisy data (noisy weighted mean squared error (WMSE)) and also on the noise free data (noise free WSSE).

A central composite design (CCD) was applied to the process degrees of freedom, namely the induction time between 5 and 9 h, pre-induction substrate feed rate between 0 and 0.8 h^{-1}, post-induction substrate feed rate between 0 and 0.8 h^{-1} and Inducer feed rate between 0 and 1 h^{-1}. This resulted in 25 fed-batch experiments. The 25 fed-batch experiments were included in the training data partition (297 training data points). The validation data set (used only as training stop criterium) was obtained by adding Gaussian noise with standard deviations of 1.5 (X), 5 (S) and 0.3(P) to the training data set resulting in 297 validation data points. In our experience, this partition method maximizes data usage for training and also effectively prevents model overfitting. For the test data set (used to assess the model generalization capacity), the optimal fed-batch with optimized feeding and maximum product concentration of 3.16 g/L (Lee and Ramirez, 1994) was adopted (15 data points). In summary, the models were trained/validated with the 25 DoE experiments and then set to predict the dynamic profiles of the optimal production fed-batch. The optimal production fed-batch delivers a final product mass, which is 34.4% higher than the best DoE fed-batch. The details of the dataset are provided as supplementary material A.

The hybrid model structure adopted for this problem is shown in Fig. 2. The reactor has 7 internal sate variables $\boldsymbol{x} = [X, S, P, IND, Sh, Re]^T$ of which only 3 are measured, thus $\boldsymbol{y}$=[*X, S, P*]^T. The system of ODEs are derived from mass conservation laws and are the same as in Lee and Ramirez (1994). The neural network computes 4 reaction terms $\vartheta = [\mu, v_S, v_P, k_1]^T$, taken as unknown cellular features that need to be learned from data. The neural network has only 3 inputs $H^0 = [S, IND, Sh]^T$ which were pre-selected based on prior knowledge of the reaction kinetics for this problem (Lee and Ramirez, 1994). Hybrid models with different network depths and sizes were evaluated, with the best hybrid model discriminated on the basis of the Akaike Information Criterion with second order bias correction (AICc) computed for the training data partition as follows:

$$AICc = T\ ln(WSSE) + 2\ nw + \frac{2\ nw\ (nw+1)}{T - nw - 1} \quad (7).$$

AICc includes an overparameterization penalty and is commonly used to discriminate between empirical model candidates with different number of parameters, *nw*, and to select a parsimonious model for small sample sizes (Li et al., 2002).

2.3.2. *MUT± Pichia pastoris pilot data set*

A MUT+ *Pichia pastoris* expressing a single chain antibody (scFv) was cultivated in a Lab Pilot Fermenter Type LP351, 50 L, Bioengineering, Switzerland with standard instrumentation to measure on-line pH, temperature, pressure, stirrer, airflow and pO2. The wet cell weight and scFv titer were measured off-line. All the details of the experimental procedure are given elsewhere (Teixeira et al., 2006). The reactor operation is divided in three phases: glycerol batch (GB) phase, glycerol fed-batch (GFB) phase and methanol fed-batch (MFB) phase (or post-induction phase). In the GB phase, the initial glycerol level was set at 4%, taking approximately 30 h for complete depletion. Thereupon, the GFB phase starts, following an exponential feeding profile. At the end of the GFB, a transition to the MFB phase is implemented in order to minimize the adaptation time of cells to methanol. After the transition phase, the methanol feeding rate, the pH and the temperature were designed in order to generate process data to optimize scFv productivity (see Teixeira et al. (2006) for details). A total of 9 experiments were performed with varying methanol feed rate, temperature and pH. In this study, only the MFB phase was considered for hybrid modeling. The data set with the 9 experiments has 207 measurements of biomass wet cell weight in triplicate and 207 measurements of scFv in triplicate. The training-validation partition included 8 experiments and the test partition 1 experiment. All possible training-validation/test permutations were evaluated. The hybrid model structure adopted for this problem is similar to that of Fig. 2 with a few adaptations (discussed in the results sections). The training and model discrimination methods were as for the Park & Ramirez case study.

Table 1
Training results of shallow hybrid models for the Lee & Ramirez data set with 25 training batches (Training WSSE), 25 validation batches (Validation WSSE) and a single test batch with the highest possible productivity obtained by optimal control(Test WSSE noisy/noise free are computed with noisy or noise free target concentrations, respectively). The AICc is computed for the training data set only. Each row represents a different model with a given number of hidden nodes (between 1 and 15) in a single hidden layer with tanh activation function. The hybrid models were trained with the standard nondeep method (LMM optimization with 20,000 iterations + cross-validation + random weights initialization between [-0.1, 0.1] from the uniform distribution). The training was repeated 10 times with different weights initialization and only the best result is kept for each model.

Number of hidden nodes	Training WSSE	Validation WSSE	Test WSSE (noisy)	Test WSSE (noise free)	AICc	CPU time	Number of weights
1	20.2	20.3	42.2	2.1	2490	776	10
2	2.57	2.77	7.53	8.12	810	1320	17
3	1.16	1.31	1.08	1.39	172	1780	24
4	1.1	1.29	1.34	1.01	146	1560	31
5	2.77	3.07	6.56	5.42	922	1390	38
6	1.78	1.94	1.22	2.11	570	1730	45
7	1.40	1.70	7.87	7.31	389	1870	52
8	1.09	1.32	1.14	0.76	200	2050	59
9	1.01	1.21	1.04	0.68	150	2250	66
10	0.941	1.16	1.05	0.54	111	2250	73
11	0.949	1.22	1.33	0.83	134	2360	80
12	0.914	1.11	0.86	0.75	121	2290	87
13	0.935	1.07	1.03	0.69	154	2280	94
14	0.944	1.15	1.10	0.93	183	2230	101
15	0.899	1.11	0.937	0.62	152	2670	108

3. Results and discussion

3.1. Development of a shallow hybrid model: Lee & Ramirez case study

A traditional shallow hybrid model was first developed for the Lee & Ramirez data set. A shallow feedforward network with a single hidden layer with *tanh* activation function was employed. The hybrid model was trained with the standard nondeep method (LMM optimization + cross-validation + random weights initialization from the uniform distribution). The training and validation partition comprehended 25 experiments (825 training patterns) designed by statistical DoE (see methods section). The test partition included a single experiment with the highest protein production (optimal batch obtained by dynamic optimization as reported in (Lee and Ramirez, 1994). The test experiment has a final product mass 34.4% higher than the best training/validation experiment. For a given network size, the training was always repeated 10 times with different weights initialization between [-0.1, 0.1] and only the best result was kept (lower validation error). This procedure was repeated for hybrid models with varying number of nodes in the hidden layer keeping the same data partition and maximum number of iterations of 20,000 for comparability. The overall results are shown in Table 1. From these results, it is possible to conclude that the optimal number of hidden nodes is 10 corresponding to the lowest corrected Akaike information criterion (AICc) value (111). Of note, the AICc criterion, which is calculated for the training partition only, coincided with the lowest noise free test error (0.54 noise free WSSE; to note that the noise free WSSE is computed on process data uncorrupted by experimental noise, thus a better metric for accessing the predictive power). Despite the coincident outcome in this case, the AICc sometimes fails to discriminate the structure with the highest predictive power as shown in the next sections. Moreover, the noisy test error of the selected model with 10 hidden nodes (noisy WSSE = 1.05) is only moderately higher (11,6%) than the corresponding training error (WSSE = 0.941).

3.2. Comparing the deep and shallow hybrid modeling approaches

Several hybrid structures with varying neural network depths (2-4 hidden layers) were compared with the shallow network case (1 hidden layer). The same Lee & Ramirez data set and data partition were kept as in the previous section. We first focused on the *tanh* activation (in the hidden layers), which has been the standard for nonlinear regression problems with shallow neural networks (Cybenko, 1989). Every model structure was trained with two different methods: the traditional LMM+CV+tanh and ADAM+CV+tanh. The training was always repeated 10 times and only the best solution (lowest validation error) was kept, as before. The number of iterations for the ADAM method was

Table 2
Comparison of deep and shallow hybrid models for the Lee & Ramirez data set (same data partition as in Table 1) trained either by the LMM algorithm or by the ADAM algorithm. In all cases cross-validation (CV) and indirect sensitivities were applied. Each row represents a different shallow or deep hybrid model structure using either *tanh* or *ReLU* in the hidden layers. The training was repeated 10 times with different weights initialization and only the best result is kept.

Hybrid model	Training method	Hidden layer type	Training WSSE (noisy)	Validation WSSE (noisy)	Testing WSSE (noisy)	Testing WSSW (noise free)	AICc	CPU time	Weights
Shallow 5	LMM+CV	tanh	2.77	3.07	6.56	5.42	922	1390	38
Shallow 10	LMM+CV	tanh	0.941	1.16	1.05	0.54	111	2250	73
Deep 5 × 5	LMM+CV	tanh	1.06	1.31	1.40	1.05	198	1674	68
Deep 5 × 5 × 5	LMM+CV	tanh	0.921	1.17	1.13	0.72	154	74,892	98
Deep 5 × 5 × 5 × 5	LMM+CV	tanh	0.835	1.09	0.915	0.32	155	81,430	128
Shallow 5	ADAM+CV	tanh	1.22	1.32	1.20	0.66	242	33,476	38
		ReLu	1.02	1.05	1.03	0.35	98	33,410	
Shallow 10	ADAM+CV	tanh	1.60	1.21	0.91	0.24	547	30,376	73
		ReLu	1.34	1.13	0.94	0.14	352	30,200	
Deep 5 × 5	ADAM+CV	tanh	0.937	1.15	0.82	0.14	95	28,567	68
		ReLu	0.926	1.08	0.923	0.05	90	28,122	
Deep 5 × 5 × 5	ADAM+CV	Tanh	0.936	1.16	0.81	0.09	168	32,285	98
		ReLu	0.886	1.04	0.96	0.04	87	32,174	
Deep 5 × 5× 5 × 5	ADAM+CV	tanh	0.870	1.11	1.05	0.28	189	40,570	128
		ReLu	0.841	1.07	0.942	0.16	152	40,514	

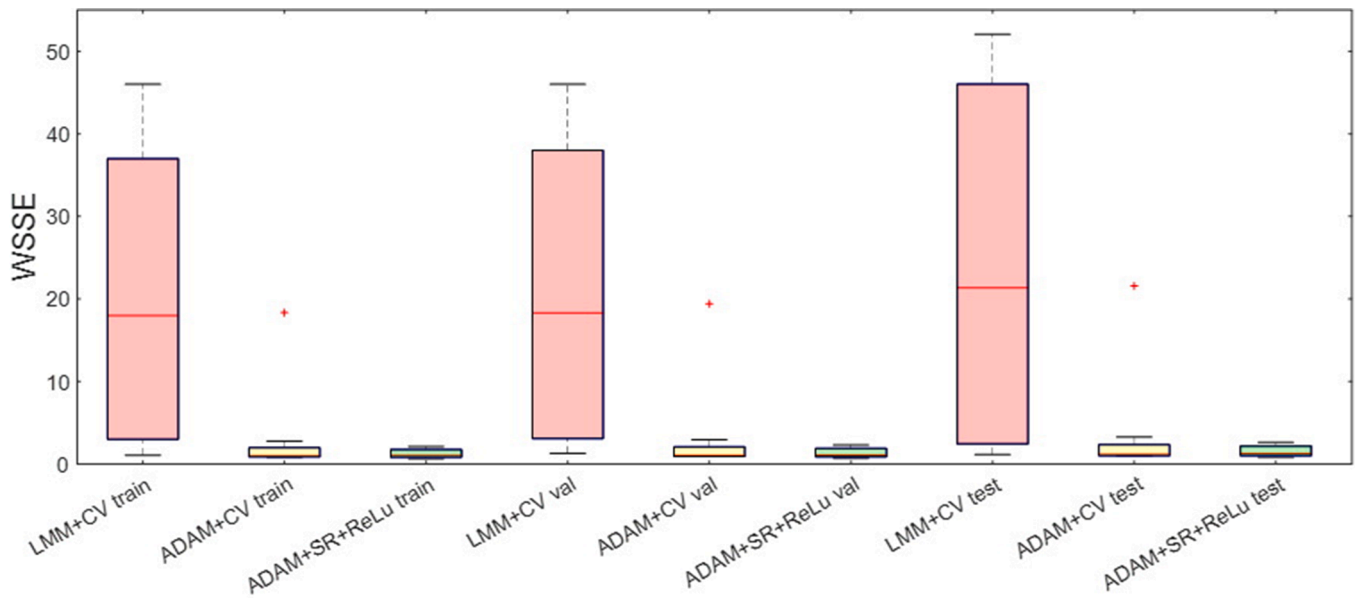

Fig. 3. Boxplot of training, validation and testing WSSE for 10 training repetitions of the deep hybrid structure 5 × 5 × 5 trained by different training approaches either using the LMM or the ADAM method. Ten sets of initial weights were randomly generated (one per repletion) and kept the same in all tests performed for comparability. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

20,000 as for the LMM method. The overall results are shown in Table 2.

The results in Table 2 clearly show ADAM to outperform the LMM method in what concerns the predictive power of the final model (the noise free test WSSE; to note that the AICc is not an adequate metric to compare models of equals sizes). This conclusion is valid for deep or shallow hybrid structures. The best shallow structure with 10 hidden nodes (identified in the previous section) improved the noise free test error from 0.54 to 0.24 (>2 fold decrease) with ADAM+CV+tanh. The same conclusions can be taken for the deep structures, without exception. The key conclusion is that the ADAM method systematically increases the predictive power of the final hybrid model for the Lee & Ramirez data set.

The best model (with *tanh* activation function) among the deep and shallow structures is the 5 × 5 × 5 deep hybrid model with 98 weights, showing a noise free test error (WSSE = 0.09) 2.7 fold lower than the best hybrid shallow case (WSSE = 0,24). The AICc miss spotted the best deep model. It identified the 2nd best model (5 × 5 structure) with, however, comparable performance. In terms of CPU, the ADAM method is generally more expensive than the LMM method for small size networks. This pattern reverses for large size networks (e.g., the best 5 × 5 × 5 structure decreased CPU by 2,3 fold with ADAM in comparison to LMM). Thus, the CPU scales more steeply with the network size in the case of LMM training when compared to ADAM training. This favors ADAM for deep hybrid structures, both in terms of predictive power and

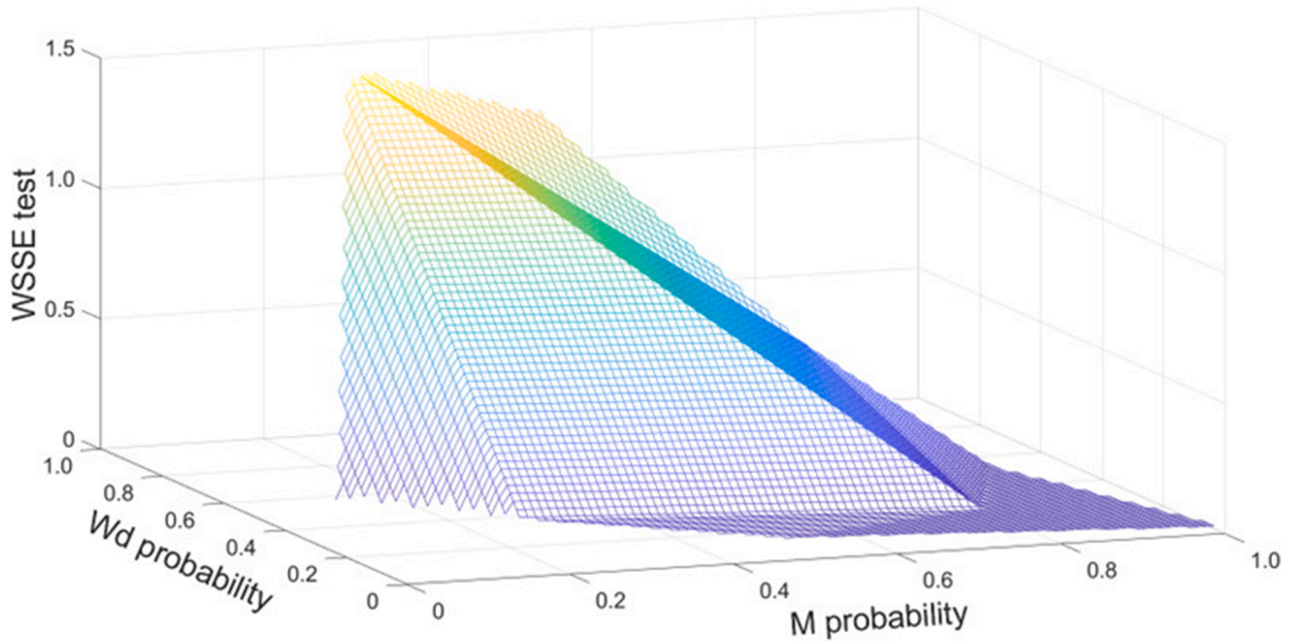

Fig. 4. Effect of stochastic regularization (SR) on the predictive power of the hybrid model configuration 5 × 5 × 5 trained with ADAM + SR + indirect sensitivities with 20,000 iterations for the Lee & Ramirez data set. Obtained noise free test WSSE over minibatch probability (M probability) and weights dropout probability (Wd probability). (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

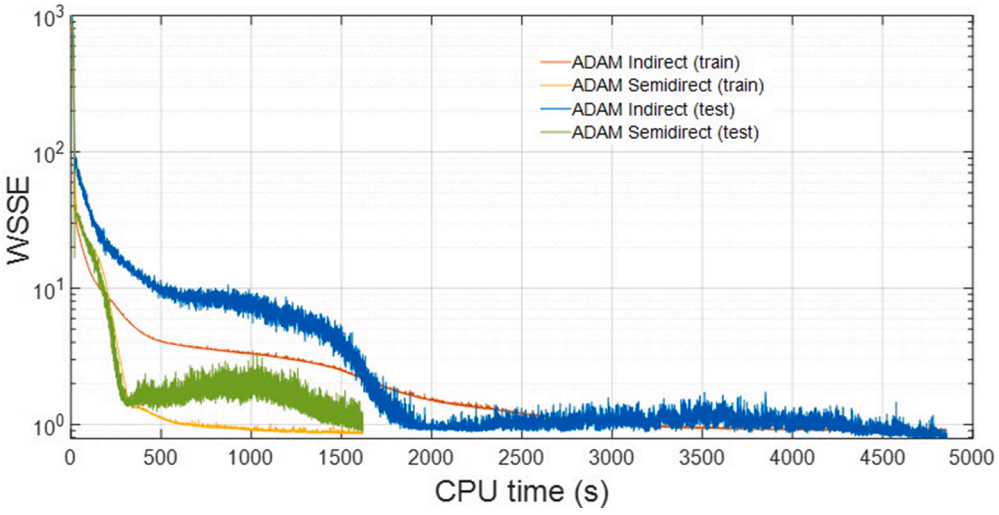

Fig. 5. Training and testing error (WSSE) over CPU time for 1) shallow hybrid model {10} + LMM +CV with ten repetitions (blue line) 2) the hybrid model 5 × 5 × 5 trained with ADAM + stochastic regularization + indirect sensitivities (red line) and 3) ADAM + stochastic regularization + semidirect sensitivities (yellow line). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

CPU time for training.

Fig. 3 shows the effect of weights initialization on the final training, validation and testing error for the best deep configuration 5 × 5 × 5 when the model is trained with LMM or with ADAM. The initial weights values were kept the same for LMM and ADAM training for comparability. Interestingly, the dispersion of the errors for 10 repetitions with different weights initialization is significantly lower for ADAM in comparison to LMM, irrespective of the data partition (train, validation or testing). There is an outlying point with significantly higher final errors for both the LMM and ADAM trainings. Concordant results were obtained for the other model configurations (results not shown). This suggests the ADAM method to be less sensitive to weights initialization. Similar conclusions were reported by Hiscock (2019) for standalone deep neural networks, who showed that gradient descent training methods with variable learning rate (such as the ADAM method) are less prone to be trapped in local optima thus less sensitive to weights initialization. The key conclusion to be taken, is that the number of repetitions for different weights initialization may be mitigated in the case of ADAM training. This represents a potential 10 fold cut in CPU time in comparison to the LMM method for the case of 10 repetitions.

The *ReLU* activation function in the hidden layers has been a key achievement in deep learning, outperforming the *tanh* function for standalone deep neural networks (Nair and Hinton, 2010). The use of ReLU was investigated comparatively with tanh in a hybrid modelling context. Table 2 compares hybrid model performances using the one or the other activation function in the hidden layers trained by ADAM + CV using the same training procedure. The key conclusion to be taken is that the *ReLU* further improved the training and test error in all cases without exception. The best 5 × 5 × 5 structure further decreased the noise free test WSSE from 0.09 (with *tanh*) to 0.04 (with *ReLU*) at comparable CPU cost. Our results clearly show the *ReLU* to be advantageous in a deep hybrid modeling context as previously shown for (standalone) deep neural networks (Nair and Hinton, 2010). The *ReLU* activation function was thus adopted in all proceeding studies. These results might be related to the problem of gradients vanishing/exploding in deep networks. Typically, the *tanh* activation function is associated with vanishing gradients whereas the reLU is associated with exploding gradients (Ding et al., 2018). The ADAM training is invariant to diagonal rescaling of the gradients. It does not completely avoid the problem of gradient vanishing when tanh is used. The use of ADAM with reLU is however very efficient at avoiding gradient explosion since it performs dynamic scaling of the learning rate (down) when the gradients become very large.

3.3. Introducing stochastic regularization

Stochastic regularization (SR) has been reported as an effective method to avoid overfitting in deep learning (Srivastava et al., 2014). Here we study the ADAM method with stochastic regularization in replacement of the cross-validation technique. More specifically, ADAM was implemented with the minibatch technique and the weights dropout technique. The minibatch technique consists in a random selection of the training patterns from the uniform distribution using a cutoff probability parameter. Similarly, the weights dropout technique used random weights selection according to a cutoff probability parameter. Fig. 4 shows the effect of the minibatch size probability and of the weights dropout probability on the testing error for the deep configuration 5 × 5 × 5. The hybrid 5 × 5 × 5 model was trained 10 times with different weights initialization with varying minibatch and weights dropout probabilities. Fig. 4 shows the lowest WSSE test among the 10 repetitions as function of the minibatch size probability and of the weights dropout probability. The training performance is indeed very sensitive to the choice of these two parameters. The optimal minibatch probability is ~90% and the optimal dropout probability is ~50%. The final noise free test WSSE was 0.0258, which is 35.5% lower than the corresponding solution without stochastic regularization (Table 2, ADAM+CV+ReLU). The final train and test errors among the 10 repetitions are shown in Fig. 3. Interestingly the stochastic regularization eliminated the outlying training result obtained by LMM+CV and ADAM+CV in the previous section. This result is promising because it shows the weights initialization to have practically no influence on the final training outcome. If repetitions are not needed, the CPU cost may be significantly reduced in relation to the LMM+CV or ADAM+CV methods.

3.4. Speeding up hybrid deep learning by semidirect sensitivities

The results above support ADAM + deep networks + stochastic regularization to produced hybrid models with higher predictive power in comparison to the traditional shallow hybrid approach. Nevertheless, deep models tend to have large networks with the CPU time increasing with the network size (Luo et al., 2005). Solving the sensitivity equations is responsible for a significant part of the CPU cost. Taking the 5 × 5 × 5 hybrid structure as example, solving the sensitivity equations implies integrating 98 × 5 = 490 ODEs along with the hybrid model ODEs for the computation of the objective function and objective function gradients. Such a large number of ODEs represents a significant CPU

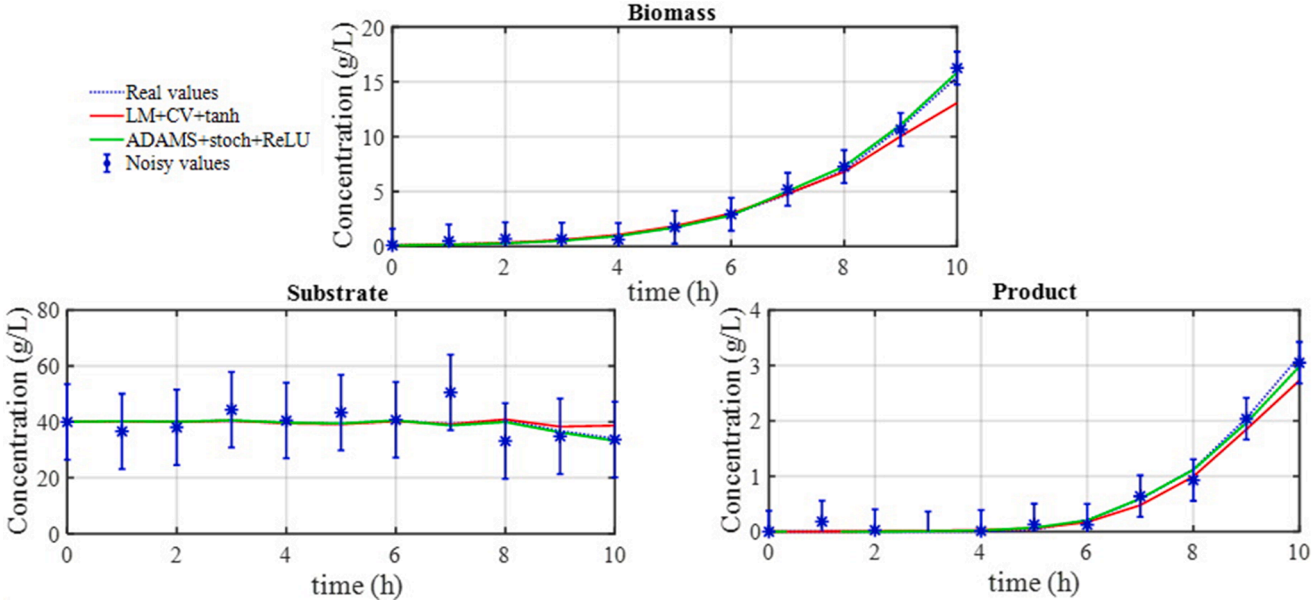

Fig. 6. Prediction of the dynamic profiles of observable variables (biomass -X, substrate– S and product–P) of the test batch (Lee & Ramirez dataset) by the best shallow hybrid model trained with the standard method (10 hidden nodes) and by the best deep hybrid model (5 × 5 × 5). Asterisks represented observations and respective ± standard deviation. The dashed line represents the "true" noise-free process behavior (hidden to the training of the hybrid models). The red line represents the predictions of the shallow hybrid model. The green line represents the prediction by the deep hybrid model. The shallow hybrid model used the tanh function and was trained by the traditional non-deep method (LMM algorithm + CV + indirect sensitivities + 10 repetitions and only the best result is kept). The deep hybrid model used the ReLU activation function and was trained by the novel method (ADAM + SR + semidirect sensitivities + no repetitions). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

burden. A different implementation of the sensitivity method was investigated, namely the semidirect sensitivity equations (see methods section) in an attempt to reduce CPU time. In the semidirect approach, a much lower number of $\left(\frac{\partial c}{\partial v}\right)$ sensitivity equations are integrated over time. For the same 5 × 5 × 5 hybrid structure, the $\left(\frac{\partial c}{\partial v}\right)$ sensitivities only require 5 × 4 = 20 ODEs to be integrated over time. Furthermore, the semidirect sensitivity equations are independent of the number and size of hidden layers (they depend only on the number of network inputs and outputs). Fig. 5 shows the variation of the train and test cost function over CPU for the configuration 5 × 5 × 5. This result shows that the semidirect sensitivity equations produced a comparable final training WSSE in relation to the indirect sensitivity equations. The convergence is however much faster. The CPU time could be reduced by 77.4% when adopting the semidirect sensitivity equations in comparison with the indirect approach. Furthermore, the test error follows similar patterns for both methods reaching a comparable final value.

Fig. 6 shows the prediction of the optimal batch dynamics by the hybrid 5 × 5 × 5 model trained with ADAM+SR+ReLU+semidirect compared to the standard shallow model with 10 hidden nodes (LMM+CV+tanh+indirect). The noise free test WSSE was 0.03 and 0.54, respectively (94,4% reduction). It may be seen that both modes are able to describe fairly well the dynamics of the test experiment up to 7.5 h. There are however some visible differences towards the end of the cultivation. The shallow hybrid model underestimated the final biomass and final product by 15.3% and 13.8%, respectively, whereas the deep hybrid model overestimated the final biomass by 2.7% and underestimated the final product by 5.8% only.

3.5. Pilot scale Pichia pastoris case study

Hybrid models were developed for the *P. pastoris* process with a similar structure to the Lee & Ramirez model. The biomass and product material balance equations, and the shock factor ODEs are kept the same in both models. A few modifications were however required as follows:

- The inducer material balance equation was removed because in the MUT+ *P. pastoris* expression system the methanol is simultaneously the main carbon source and the inducer of foreign protein expression.
- The substrate material balance equation was also removed because methanol concentration (the substrate) was not measured. This is a limitation imposed by the experimental protocol. Instead, the measured volumetric methanol feed rate (F_{met}, g/Lh) and the measured total methanol fed to the reactor (g) were set as external inputs to the neural network.
- Temperature (T) and pH were also added as external inputs to the neural network as these two parameters varied between 17.2 and 30.1 °C and pH 4.0–7.0 in the experiments performed as part of a design of experiments to study the influence of these two parameters in the protein expression.
- The neural network computed the volumetric protein production rate (output) instead of the specific protein production rate as in the case of *Lee & Ramirez*. It is known that *Pichia pastoris* secretes proteases that hydrolyses the target product on certain experimental conditions (Cereghino and Cregg, 2000). The neural network is thus set to calculate the apparent volumetric production rate of the scFv, which lumps the synthesis and hydrolysis in the same kinetic term.

We have investigated the optimal hybrid structures and concluded that the two best shallow and deep hybrid structures previously identified for the Lee & Ramirez case study (namely the shallow structure with 10 nodes in the hidden layer and the deep 5 × 5 × 5 structure) also apply for the *Pichia pastoris* case study (results not shown). The number of parameters in both the shallow and deep models is the same, namely 123. The shallow hybrid structure was trained with the traditional method (LM+CV+*tanh*+direct, 10 repetitions with random weights initialization from the uniform distribution) whereas the deep hybrid structure was trained with the new method (ADAM+SR+ReLU+semidirect, weight dropout probability of 0.5, minibatch probability of 0.9 and no repetitions). Eight reactor experiments were used for training-validation (validation data points were

Table 3
Comparison of deep and shallow hybrid models for the pilot reactor MUT+ *Pichia pastoris* data set. Each row represents a hybrid model obtained by training over a different training/testing data permutation (Test batch ID refers to the batch used for testing while the remaining 8 batches were used for training/validation). Shallow hybrid models had tanh activation function and were trained by the traditional non-deep method (LMM algorithm + CV + indirect sensitivities + 10 repetitions and only the best result is kept). Deep hybrid models used the ReLU activation function and were trained by the novel method (ADAM + SR + semidirect sensitivities + no repetitions).

Test batch ID	Model type	Training WSSE (noisy)	Testing WSSE (noisy)	AICc	CPU time
F037	Shallow 10	2.18	2.58	664	19,560
	Deep 5 × 5 × 5	1.79	2.13	587	13,980
F044	Shallow 10	2.42	3.94	700	46,440
	Deep 5 × 5 × 5	2.14	3.73	633	19,980
F048	Shallow 10	2.01	2.55	626	15,060
	Deep 5 × 5 × 5	1.96	2.28	618	12,000
F061	Shallow 10	2.65	4.69	738	22,860
	Deep 5 × 5 × 5	1.98	4.05	620	14,520
F066	Shallow 10	2.54	2.82	722	13,680
	Deep 5 × 5 × 5	1.59	1.86	542	9660
F007	Shallow 10	2.79	4.13	752	23,640
	Deep 5 × 5 × 5	2.24	2.98	663	13,320
F009	Shallow 10	2.82	4.72	754	30,180
	Deep 5 × 5 × 5	2.62	3.76	730	12,480
F018	Shallow 10	2.48	3.28	710	15,900
	Deep 5 × 5 × 5	2.31	2.67	684	10,200
F072	Shallow 10	3.15	4.85	791	26,100
	Deep 5 × 5 × 5	3.01	3.98	775	14,820
Sum	**Shallow 10**	**23.04**	**33.6**	**6457**	**213,420**
	Deep 5 × 5 × 5	**19.64**	**27.4**	**5852**	**120,960**

obtained by adding Gaussian noise to the training data points as in the Lee & Ramirez case study) and just one experiment for testing. All possible training-validation/testing permutations were evaluated. The overall results are shown in Table 3 where each row represents a different training-validation/testing permutation. As illustrative example, Fig. 7 shows the measured and predicted dynamic profiles of biomass and product for the case of experiment F66 used for testing. The key conclusions to be taken is that both the training and testing WSSEs were lower for the deep hybrid structure in relation to the shallow structure, in all data partitions tested without exception. The AICc criteria also points out to the same conclusion. The differences between the dynamic profiles of biomass and scFv are clearly visible in Fig. 7. The predicted final scFv titer by the shallow hybrid model is 17.5% below the experimental value whereas the deep hybrid model overestimated the experimental value by 4.2% only. Taking all data partitions together (last row in Table 3), the average training WSSE decreased by 14.8% whereas the average testing WSSE decreased by 18.4% for the deep hybrid structure in relation to the shallow hybrid structure. Moreover, the average CPU time decrease by 43.4% when applying the deep methodology in comparison to the standard methodology.

4. Conclusions

In this study the general bioreactor hybrid model was revisited and some recent deep learning techniques were investigated in the context of hybrid modeling. The effect of increasing the depth of the neural network resorting to two different training approaches was investigated. The traditional approach uses the Levenberg–Marquardt optimization coupled with the indirect sensitivities, cross-validation and *tanh* activation function. The novel hybrid deep approach uses the adaptive moment estimation method (ADAM), semidirect sensitivities, stochastic regularization and ReLU activation functions in the hidden layers. Two applications were addressed, one with a synthetic data set, the other with an experimental dataset collected in a pilot 50 L bioreactor. The key conclusion to be taken is that there is a clear advantage of adopting hybrid deep models both in terms of predictive power and in terms of computational cost in relation to the shallow hybrid case. In the

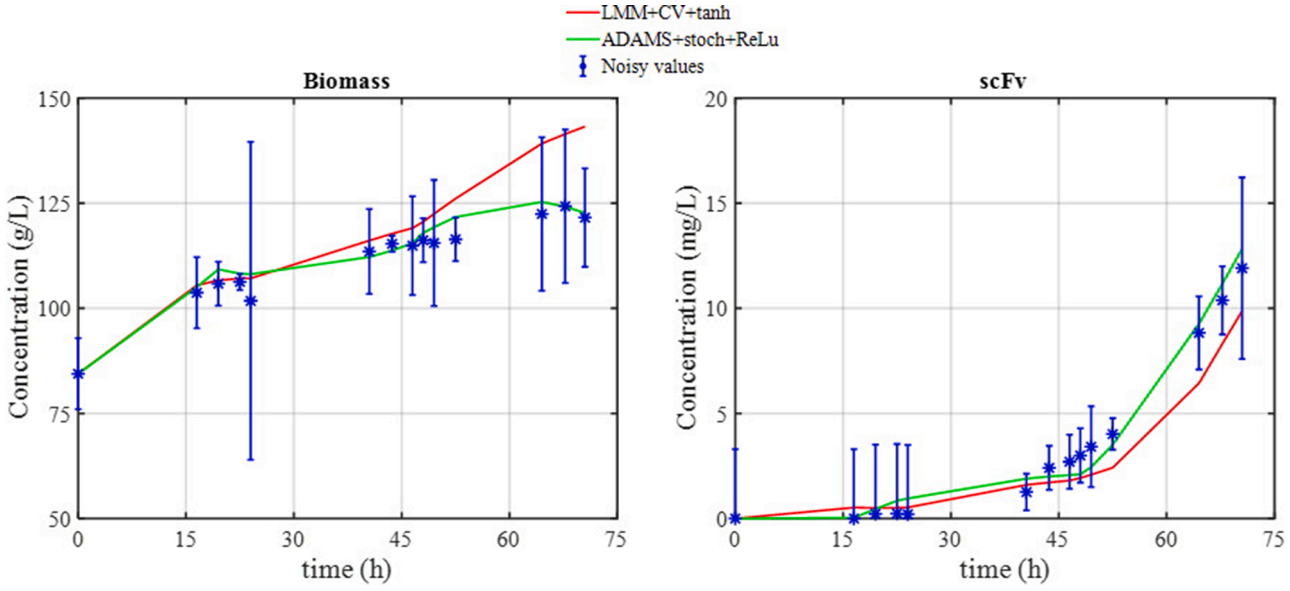

Fig. 7. Prediction of the dynamic profiles of observable variables (biomass-X, and product–scFv) by the shallow (10) hybrid model and by the deep (5 × 5 × 5) hybrid model for the test batch F066 of the MUT+ *Pichia pastoris* pilot data set. Asterisks represent observations and respective ± standard deviation. The red line represents the predictions of the shallow hybrid model. The green line represents the prediction of the deep hybrid model. The shallow hybrid model used the tanh activation function and was trained by the traditional non-deep method (LMM algorithm + CV + indirect sensitivities + 10 repetitions and only the best result is kept). The deep hybrid model used the ReLU activation function and was trained by the novel method (ADAM + SR + semidirect sensitivities + no repetitions). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Lee&Ramirez case study, the prediction error decreased 94.4% and the CPU decreased 29%. In the case of the *P. Pastoris* case study, the prediction error decreased 18.4% and the CPU decreased 43,3%. The ADAM method coupled with stochastic regularization shows two significant advantages. First, it is practically insensitive to weight initialization thereby eliminating the need for training repetitions. Second, the stochastic nature of the method is less sensitive to experimental noise eliminating the need for cross-validation. Lastly, the introduction of semidirect sensitives, further decreases the CPU time particularly for large deep structures as the number of sensitivity equations (that need to be integrated over time) becomes independent of the number of hidden layers.

JP Collected the data and performed the analysis. MM and JR and RC contributed to the data collection and to data analysis. GS and RO conceived and design the analysis. JP, RC and RO contributed to write the paper.

We confirm that the manuscript has been read and approved by all named authors and that there are no other persons who satisfied the criteria for authorship but are not listed. We further confirm that the order of authors listed in the manuscript has been approved by all of us.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data Availability

Data will be made available on request.

Acknowledgments

JP acknowledges PhD grant SFRD/BD14610472019, Fundação para a Ciência e Tecnologia (FCT). RSC also acknowledges for the contract CEECIND/01399/2017. This work has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement no 101000733 (PROMICON).

Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.compchemeng.2022.107952.

References

Banga, J.R., Balsa-Canto, E., Moles, C.G., Alonso, A.A., 2005. Dynamic optimization of bioprocesses: efficient and robust numerical strategies. J. Biotechnol. 117, 407–419.
Bangi, M.S.F., Kao, K., Kwon, J.S.I., 2022. Physics-informed neural networks for hybrid modeling of lab-scale batch fermentation for ss-carotene production using Saccharomyces cerevisiae. Chem. Eng. Res. Des. 179, 415–423.
Bangi, M.S.F., Kwon, J.S.I., 2020. Deep hybrid modeling of chemical process: application to hydraulic fracturing. Comput. Chem. Eng. 134.
Bayer, B., Diaz, R.D., Melcher, M., Striedner, G., Duerkop, M., 2021. Digital twin application for model-based doe to rapidly identify ideal process conditions for space-time yield optimization. Processes 9 (7), 1–16, 1109.
Cereghino, J.L., Cregg, J.M., 2000. Heterologous protein expression in the methylotrophic yeast *Pichia pastoris*. FEMS Microbiol. Rev. 24 (1), 45–66.
Chen, L., Bernard, O., Bastin, G., Angelov, P., 2000. Hybrid modelling of biotechnological processes using neural networks. Control Eng. Pract. 8 (7), 821–827.
Cybenko, G., 1989. Approximation by superpositions of a sigmoidal function. Math. Control Signal Syst. 2, 303–314.
Delalleau, O., Bengio, Y., 2011. Shallow vs. deep sum-product networks. In: Proceedings of the International Conference on Neural Information Processing Systems, pp. 666–674.
Ding, B., Qian;, H., Zhou;, J, 2018. Activation functions and their characteristics in deep neural networks. In: Proceedings of the Chinese Control and Decision Conference (CCDC).
Eldan, R.; Shamir, O., 2016. The power of depth for feedforward neural networks, arXiv: 1512.03965.
Ferreira, A.R., Dias, J.M.L., von Stosch, M., Clemente, J., Cunha, A.E., Oliveira, R., 2014. Fast development of *Pichia pastoris* GS115 Mut(+) cultures employing batch-to-batch control and hybrid semi-parametric modeling. Bioprocess Biosyst. Eng. 37 (4), 629–639.
Fiedler, B., Schuppert, A., 2008. Local identification of scalar hybrid models with tree structure. Ima J Appl Math 73 (3), 449–476.
Galvanauskas, V., Simutis, R., Lubbert, A., 2004. Hybrid process models for process optimisation, monitoring and control. Bioprocess Biosyst. Eng. 26 (6), 393–400.
Gnoth, S., Jenzsch, M., Simutis, R., Lubbert, A., 2008. Product formation kinetics in genetically modified E-coli bacteria: inclusion body formation. Bioprocess Biosyst. Eng. 31 (1), 41–46.
Gnoth, S., Simutis, R., Lubbert, A., 2010. Selective expression of the soluble product fraction in Escherichia coli cultures employed in recombinant protein production processes. Appl. Microbiol. Biotechnol. 87 (6), 2047–2058.
Hinton, G., Srivastava, N., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.R., 2012. Improving neural networks by preventing co-adaptation of feature detectors. arxiv: 1207.0580.
Hiscock, T.W., 2019. Adapting machine-learning algorithms to design gene circuits. BMC Bioinform. 20, 1–13, 214.
Kingma, D.P., Ba, J., 2014. Adam:. A method for stochastic optimization, arXiv: 14126980.
Koutsoukas, A., Monaghan, K.J., Li, X.L., Huan, J., 2017. Deep-learning: investigating deep neural networks hyper-parameters and comparison of performance to shallow methods for modeling bioactivity data. J. Cheminform. 9, 1–13, 42.
Lee, J., Ramirez, W.F., 1994. Optimal fed-batch control of induced foreign protein-production by recombinant bacteria. AiChE J. 40 (5), 899–907.
Lee, D., Jayaraman, A., Kwon, J.S., 2020. Development of a hybrid model for a partially known intracellular signaling pathway through correction term estimation and neural network modeling. PLOS Comput. Biol. 16 (12), 1–31 e1008472.
Liang, S.; Srikant, R., 2017. Why deep neural networks for function approximation?, arXiv:1610.04161.
Li, B.B., Morris, J., Martin, E.B., 2002. Model selection for partial least squares regression. Chemometr. Intell. Lab. 64 (1), 79–89.
Luo, Z.W., Liu, H.Z., Wu, X.C., 2005. Artificial neural network computation on graphic process unit. In: Proceedings of the IEEE IJCNN, pp. 622–626.
Mhaskar, H.N., Poggio, T., 2016. Deep vs. shallow networks: an approximation theory perspective. Anal. Appl. 14 (6), 829–848.
Merkelbach, K., et al., 2022. HybridML: open source platform for hybrid modeling. Comput. Chem. Eng. 160, 107736.
Nair, V., Hinton, G.E., 2010. Rectified linear units improve restricted Boltzmann machines Vinod Nair. In: Proceedings of the International Conference on International Conference on Machine Learning. Haifa, pp. 807–814.
O'Brien, C.M., Zhang, Q., Daoutidis, P., Hu, W.S., 2021. A hybrid mechanistic-empirical model for in silico mammalian cell bioprocess simulation. Metab. Eng. 66, 31–40.
Oliveira, R., 2004. Combining first principles modelling and artificial neural networks: a general framework. Comput. Chem. Eng. 28 (5), 755–766.
Pinto, J., de Azevedo, C.R., Oliveira, R., von Stosch, M., 2019. A bootstrap-aggregated hybrid semi-parametric modeling framework for bioprocess development. Bioprocess Biosyst. Eng. 42 (11), 1853–1865.
Preusting, H., Noordover, J., Simutis, R., Lubbert, A., 1996. The use of hybrid modelling for the optimization of the penicillin fermentation process. Chimia 50 (9), 416–417.
Psichogios, D.C., Ungar, L.H., 1992. A hybrid neural network-1st principles approach to process modeling. AlChE J. 38 (10), 1499–1511.
Schubert, J., Simutis, R., Dors, M., Havlik, I., Lubbert, A., 1994. Bioprocess optimization and control - application of hybrid modeling. J. Biotechnol. 35 (1), 51–68.
Shah, P.R., et al., 2022. Deep neural network-based hybrid modeling and experimental validation for an industry-scale fermentation process: Identification of time-varying dependencies among parameters. Chem. Eng. J. 441, 135643.
Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R., 2014. Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. 15, 1929–1958.
Su, H.T., Mcavoy, T.J., 1993 Sep. Integration of multilayer perceptron networks and linear dynamic-models - a Hammerstein modeling approach. Ind. Eng. Chem. Res. 32 (9), 1927–1936.
Teixeira, A.P., Clemente, J.J., Cunha, A.E., Carrondo, M.J.T., Oliveira, R., 2006. Bioprocess iterative batch-to-batch optimization based on hybrid parametric/ nonparametric models. Biotechnol. Progress 22 (1), 247–258.
Teixeira, A.P., Alves, C., Alves, P.M., Carrondo, M.J., Oliveira, R., 2007. Hybrid elementary flux analysis/nonparametric modeling: application for bioprocess control. BMC Bioinform. 8, 30.
Thompson, M.L., Kramer, M.A., 1994. Modeling chemical processes using prior knowledge and neural networks. AlChE J. 40 (8), 1328–1340.
van Can, H.J.L., teBraake, H.A.B., Dubbelman, S., Hellinga, C., Luyben, K.C.A.M., Heijnen, J.J., 1998. Understanding and applying the extrapolation properties of serial gray-box models. AlChE J. 44 (5), 1071–1089.
von Stosch, M., Oliveira, R., Peres, J., de Azevedo, S.F., 2011. A novel identification method for hybrid (N)PLS dynamical systems with application to bioprocesses. Expert Syst. Appl. 38 (9), 10862–10874.
von Stosch, M., Oliveira, R., Peres, J., de Azevedo, S.F., 2014. Hybrid semi-parametric modeling in process systems engineering: past, present and future. Comput. Chem. Eng. 60, 86–101.
von Stosch, M., Hamelink, J.M., Oliveira, R., 2016. Hybrid modeling as a QbD/PAT tool in process development: an industrial E-coli case study. Bioprocess Biosyst. Eng. 39 (5), 773–784.
Werbos, P., 1974. Beyond Regression New Tools for Prediction and Analysis in Behavioral Sciences. Harvard University.