SPICE-Based Behavioral Models of IC Buffers via Compact Kernel Regressions

SPICE-based Behavioral Models of IC Buffers via Compact Kernel Regressions

Marco Atlante, Riccardo Trinchero, Igor S. Stievano
Politecnico di Torino, 10129 Torino, Italy
E-mail: marco.atlante@polito.it
riccardo.trinchero@polito.it
igor.stievano@polito.it

Mihai Telescu, Noël Tanguy
Univ Brest, Lab-STICC, CNRS,
UMR 6285, F-29200, Brest, France
E-mail: mihai.telescu@univ-brest.fr
noel.tanguy@univ-brest.fr

***Abstract*—This work presents a method for generating compact SPICE-compliant IC buffer models using the vector-valued Kernel Ridge Regression (VV-KRR). Power supply variations are captured, and model complexity is reduced with a compression technique based on the random sampling or the Nyström approximation. The compressed models are then implemented in SPICE, and their performance is evaluated on a realistic test case.**

***Index Terms*—Digital integrated circuits, buffer modeling, signal and power integrity, high-speed interconnects, machine learning, kernel regression.**

I. Introduction

The availability of accurate and efficient models of digital IC buffers is a key resource for the simulation of critical interconnects and the design of high speed digital electronics. Over the years, the research community has worked toward the development of modeling techniques allowing to generate device models enabling reliable signal and power integrity assessments via numerical (e.g., via SPICE) simulations.

The golden reference is provided by the Input/Output Buffer Information Specification (IBIS), which has evolved significantly since its initial deployment in the late 1990s, incorporating numerous improvements over time. It is a standard in this field, offering a practical well defind procedure and modular modeling approach widely supported by electronic design automation (EDA) tools and silicon vendors [1]. Despite its success, the research community contributed to the study of alternative solutions aimed at improving some possible weakness of IBIS, mainly, better accuracy for some device technologies and parameter variation as well as simplifying model generation [2]–[4]. In parallel, machine learning (ML) has gained traction in this domain, with methods like recurrent neural networks (RNNs) demonstrating impressive flexibility and precision [5], [6]. However, such approaches often come with significant computational demands, requiring extensive training processes and intricate implementation in simulation environments. Moreover, due to their inherently nonlinear formulation and the complexity of RNN structures, their implementation in SPICE-like circuital solvers is rather complex and is usually done in advance hardware description languages, such as Verilog-A [6]. As a result, simpler yet effective alternatives have attracted growing attention. Kernel regression techniques, particularly those based on feedforward architectures, offer a viable pathway for balancing accuracy and simplicity in behavioral modeling. By leveraging a nonlinear autoregressive exogenous (NARX) representation, kernel-based models can achieve high fidelity while maintaining a structure that is readily integrated into simulation tools [7].

In this work, we propose the generation of compact SPICE-compliant models of single-ended IC buffers. A vector-valued kernel Ridge regression (VV-KRR) [8] is adopted, enabling the generation of accurate IC models including the effects of power supply variations. Model complexity, in terms of number of expansion terms, is then reduced using two different compression schemes based on a random sampling selection and the Nyström approximation. Their performance is thoroughly discussed. The model is implemented in SPICE and model performance, i.e., its accuracy and efficiency, is evaluated on a realistic SI/PI test case, stressing the beneficial effects of the proposed representation and model compression.

II. System Identification via NARX Model and Kernel Ridge Regression

A possible way to learn and model the actual nonlinear dynamic map describing the behavior of the port variables of a generic IC driver, like the one shown in Fig. 1, is to adopt a NARX model.

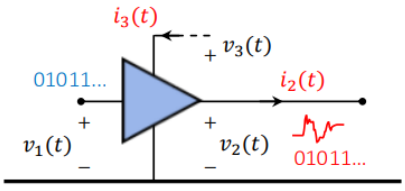

Fig. 1. Representation of a typical IC driver, including its relevant electrical input and output variables.

The NARX model is a feedforward representation of a nonlinear dinamic system in the following form [8]–[10]:

$$\hat{\boldsymbol{y}}_k = f(\boldsymbol{y}_{k-1}, \ldots, \boldsymbol{y}_{k-p}, \boldsymbol{u}_k, \ldots, \boldsymbol{u}_{k-p}), \tag{1}$$

where the input vector $\boldsymbol{u}_k = [v_{1,k}, v_{2,k}, v_{3,k}]$ collects the discrete time samples of the port voltages, the output vector

$\boldsymbol{y}_k = [y_k^{(1)}, y_k^{(2)}] = [i_{2,k}, i_{3,k}]$ collects the samples of the actual port currents considered as output variables, and $\hat{\boldsymbol{y}}_k$ collects the estimated discrete output values. The discrete time index k is defined as $t = kT_s$, with T_s as the sampling period. The parameter p represents the dynamic order, defining the number of past samples influencing the behavior of the device. The feedforward structure of the NARX model simplifies parameter estimation, particularly when used with plain kernel-based learning methods.

To learn the NARX model in (1) from data, the transient responses of the waveforms of the input and output variables must be organized in terms of the dataset $\mathcal{D} = \{(\tilde{\boldsymbol{x}}_l, \boldsymbol{y}_l)\}_{l=1+p}^{L}$, where each input vector $\tilde{\boldsymbol{x}}_l$ captures the temporal dependencies of the device by combining current and past signals [9], [10], such as

$$\tilde{\boldsymbol{x}}_l = [\boldsymbol{u}_l, \ldots, \boldsymbol{u}_{l-p}, \boldsymbol{y}_{l-1}, \ldots, \boldsymbol{y}_{l-p}]^T. \tag{2}$$

The VV-KRR presented in [8] with a block-diagonal structure can be used to the above nonlinear relationships between the input and output variables via the following kernel expansion:

$$\hat{y}_k^{(i)} = \sum_{l=1+p}^{L} \alpha_l^{(i)} K_{\boldsymbol{x}}(\tilde{\boldsymbol{x}}_l, [\boldsymbol{u}_k, \hat{\boldsymbol{y}}_{k-1}, \ldots, \hat{\boldsymbol{y}}_{k-p}]^T), \tag{3}$$

where $i = 1, 2$, $\{\alpha_l^{(i)}\}_{l=1+p}^{L}$ are model coefficients, and $K_{\boldsymbol{x}}(\cdot,\cdot)$ is a kernel function. The coefficients $\boldsymbol{\alpha}^{(i)} = [\alpha_{1+p}^{(i)}, \ldots, \alpha_L^{(i)}]^T$ are estimated by solving the following linear system:

$$\boldsymbol{\alpha}^{(i)} = (\mathbf{K}_{\boldsymbol{x}} + \lambda \mathbf{I})^{-1} \boldsymbol{y}^{(i)}, \tag{4}$$

where $\mathbf{K}_{\boldsymbol{x}}$ is the Gramian matrix with elements $K_{\boldsymbol{x}}(\tilde{\boldsymbol{x}}_k, \tilde{\boldsymbol{x}}_j)$, $\boldsymbol{y}^{(i)}$ contains the training samples, and λ is the Tikhonov regularization parameter. A Gaussian RBF kernel, with hyperparameters optimized on a validation set, is used [8].

The block-diagonal structure of VV-KRR offers an advantage in terms of hyperparameter tuning, as the number of parameters to optimize remains independent of the output dimensionality.

III. SPICE Implementation

This Section briefly outlines the netlist implementation of model equations in a SPICE-based solver. Specifically, the implementation is done via a suitable interpretation of the output variables in (3) (i.e., the output i_2 and the power supply i_3 currents) in terms of controlled current sources.

As an example, the HSPICE syntax for the output current which is assumed flowing out of the output terminal of the buffer (e.g. at node 2) writes: `G2 0 2 cur='g2'`. In the above syntax, g2 is replaced by the weighted sum of kernel functions:

$$\mathtt{g2} = \sum_{j=1}^{N} \alpha_j^{(1)} \exp\left(-\frac{[\cdot]_j}{2\sigma^2}\right), \tag{5}$$

where the argument of the j-th exponential term is

$$[\cdot]_j = (v_1(t) - \tilde{x}_{j1})^2 + \cdots + (v_1(t - pT_s) - \tilde{x}_{jp})^2 + \cdots \\ (v_2(t) - \tilde{x}_{j(p+1)})^2 + \cdots + (v_2(t - pT_s) - \tilde{x}_{j(2p)})^2 + \cdots \tag{6}$$

Such implementation is readily possible via the set of mathematical operators such as `EXP` (exponential function) or `POW` (power function) available in any SPICE engine.

It is important to notice the uniform sampling required by the model in (3) and (5), to probe the past samples of the voltage and current variables discretized at given time instants. Such discrete samples can be conveniently obtained using a chain of matched transmission lines with a delay corresponding to the sampling period T_s, as shown in Fig 2 for the variable $v_1(t)$. Similar structures are used for all the involved variables. A native option in SPICE is the so-called delay element in place of the explicit matched concatenated lines.

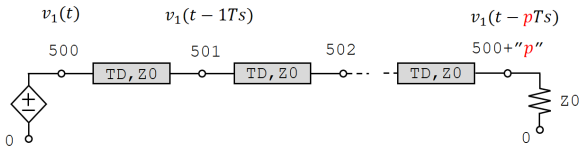

Fig. 2. Matched transmission lines chain used to generate the discrete time samples of the port voltage v_1 (`TD` = T_s and `Z0` = 50 Ω) .

It is important to remark that the SPICE implementation has a complexity in terms of number of components and/or operations which is directly related to the number of training samples and the dynamic order p. The number of training samples corresponds to the terms in the behavioral controlled sources for the output and power supply currents and the dynamic order impacts on the size of the argument of the exponential kernel functions in (5). The above observation highlights the key role of a possible model compression, which might have a strong beneficial role in terms of model compactness and CPU time [11].

IV. Model compression

The integration of VV-KRR into SPICE-like simulators offers significant advantages, but its scalability is limited by the size of the training dataset. As the dataset grows, the computational demands increase due to the linear scaling of kernel basis functions and $\boldsymbol{\alpha}$ coefficients. To address these challenges, this work explores two compression techniques: the random sampling and the Nyström approximation.

A. Compression with Random Sampling

Random sampling can be seen as the simplest scheme to reduce the complexity of kernel expansion in (3). The underlying idea is to train the model using a subset of m training samples randomly selected from the full training set with $N = (L - 1 - p)$ samples, with $m \ll N$ [11]. This results in a compressed model that accelerates both training

and inference. Random sampling is particularly effective when the dataset contains highly correlated samples, as discarding certain data points generally has a minimal impact on the overall performance of the model. On the other hand, due to the random nature of the selection of the training samples, there is the risk of choosing a subset with overly similar samples, potentially excluding valuable information. This can introduces variability in the performance of the model, making this compression scheme affected by high variance. Moreover, the samples that are not selected and used for the model training are completely discarded and do not contribute to the training process.

B. Compression with Nyström Approximation

The Nyström approximation provides a more robust and reliable alternative to the above pure random selection [12].

The underlying idea is to compress the Gramian matrix $\mathbf{K}_{\boldsymbol{x}}$ in (4) using a subset of $m \ll N$ representative columns, indexed by S while the columns indexed by R correspond to all the data. Specifically, the kernel matrix $\mathbf{K}_{\boldsymbol{x}}$ can be approximated as follows:

$$\mathbf{K}_{\boldsymbol{x}} \approx \mathbf{K}_{Nm}\mathbf{K}_{mm}^{-1}\mathbf{K}_{Nm}^{T}, \tag{7}$$

where $\mathbf{K}_{mm} = \{K_{\boldsymbol{x},ij} : i,j \in S\}$ is a $m \times m$ matrix representing a selection of the elements of the full Gramian matrix in which the kernel is evaluated on the pairs of input samples belonging to the subset S, while $\mathbf{K}_{Nm} = \{ K_{\boldsymbol{x},ij} : i \in R, j \in S\}$ is an $N \times m$ matrix represents again a selection of the elements of the full Gramian matrix in which the kernel is evaluated on the pairs of input samples belonging the selected subset S and R.

This decomposition results in an efficient approximation of the full kernel Gramian matrix, which is subsequently used to calculate the $\tilde{\boldsymbol{\alpha}}$ coefficients as follows:

$$\tilde{\boldsymbol{\alpha}}^{(i)} = (\mathbf{K}_{Nm}^{T}\mathbf{K}_{Nm} + \lambda\mathbf{K}_{mm})^{-1}\mathbf{K}_{Nm}^{T}\boldsymbol{y}^{(i)}. \tag{8}$$

Also in this case the quality of the approximation depends on which columns are selected from the original kernel matrix. In this study, we employ uniform column sampling, a simple yet effective method that involves randomly selecting columns from the matrix. However, different from the compression technique based on the random selection, the Nyström approximation uses all the data in the training set leading to a reduction of the model variance.

V. Numerical Results

This section collects the results obtained by using a commercial device, namely the Texas Instruments transceiver SN74ALVCH16973, powered with a nominal supply voltage $V_{DD} = 1.8\,\text{V}$. The analysis relies on a transistor-level HSPICE model provided by the manufacturer, which serves as the reference for both collecting device responses for training and for model validation. The dataset for model estimation is obtained from a set of simulations carried out when the device is driven by a bitstream with a small number of transition events and different loads are considered. The following loads are chosen to cover diverse realistic conditions, including lumped and distributed terminations, to enhance robustness: #1) an open-ended transmission line with $75\,\Omega$ impedance and a delay of 2 ns; #2) a $50\,\Omega$ resistor in parallel with a 10 pF capacitor; #3) the series connection of a $50\,\Omega$ resistor and the nominal supply V_{DD}; and #4) a lumped $150\,\Omega$ resistor. The power supply pin is connected to a nominal V_{DD} with 5% random variations applied as stepwise levels, distributed irregularly during normal buffer operation. The waveforms generated from these configurations are sampled using $T_s = 285\,\text{ps}$, leading to a training dataset with $N = 1679$ samples. The dynamic order is $p = 5$. A VV-KRR model is trained on this dataset and subsequently validated using a new setup (#5) involving a realistic interconnect with $50\,\Omega$ characteristic impedance and 2 ns delay, being this load unseen during training, to emphasize the model's generality. The power supply in this setup uses a nominal V_{DD} followed by a series resistor and a parallel combination of an inductor and resistor.

Table I summarizes the performance of the modeling framework under the two compression approaches. The first column in the table specifies the modeling technique (full model or models generated from random compression or via Nyström approximation). The second column collects the number of expansion terms, and thus of the coefficients $\boldsymbol{\alpha}$ actually used in the model (N for the full model and m for the compressed ones). The third column reports the CPU simulation time in HSPICE, based on the implementation done following the procedure in Sec. III. The last column includes the average absolute error, calculated as the mean absolute deviation of the predicted output current from the reference curve. For each compressed method and subset size ($m = 800$ and $m = 400$ samples), five simulations were performed and the best model is chosen. The training time of the VV-KRR model remains manageable, typically on the order of tens of seconds. For instance, in the FULL case, training takes approximately 50s, whereas for datasets with 400 samples, it requires around 15s. The comparison in Table I highlights that Nyström method consistently provides better accuracy compared to random sampling, particularly when smaller subsets are used. Models with compression demonstrate shorter simulation times. Although the complete model achieves the highest accuracy, the Nyström method offers a good trade-off between accuracy and efficiency.

In spite of the overall good accuracy of the compressed models highlighted in the table, to stress the impact of randomness, Figure 3 compares the range of variability of model responses (for the current $i_2(t)$) obtained from the two alternate compression schemes with the same $m = 400$ (top panel) and the mean absolute error (MAE) obtained for each time sample (bottom panel). This figure clearly highlights an unavoidable larger variability of model performance when a blind random selection approach is used. On the contrary, the Nyström method ensures closer predictions through its structured column selection, yielding a limited variability in

TABLE I
CPU TIME AND ACCURACY COMPARISON FOR VV-KRR MODELS WITH DIFFERENT COMPRESSION SIZE AND METHOD.

Model	Exp. Terms	CPU time HSPICE (s)	Ave error i_2
FULL	$N = 1679$	84.8	0.15
Random	$m = 800$	49.2	0.82
	$m = 400$	23.0	0.94
Nyström	$m = 800$	42.4	0.52
	$m = 400$	18.7	0.59

model responses.

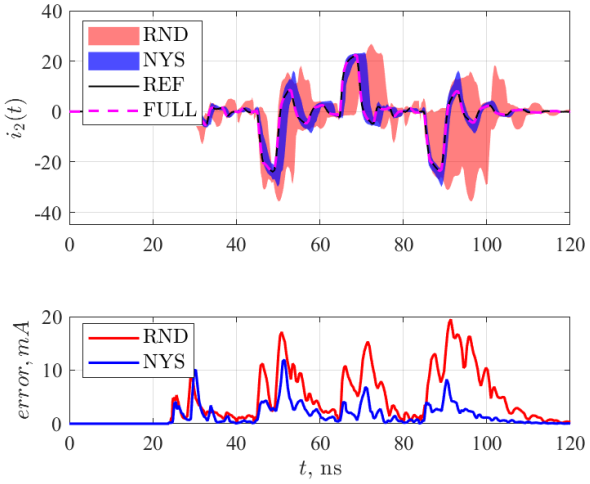

Fig. 3. Comparison of reference output voltages with model predictions and MAE analysis for different compression techniques.

As a final comparison, Figure 4 compares the predictions of the best-performing compressed models and of the full model. Even with the strongest compression ($m = 400$), the Nyström method maintains close agreement with the reference response, confirming its robustness and efficiency.

In conclusion, the proposed compression strategy effectively reduces the training set size and computational requirements while maintaining predictive accuracy. This efficiency translates into significantly faster SPICE simulation times. The method's scalability with respect to the number of expansion terms highlights its potential for modern signal and power integrity workflows, where the trade-off between accuracy and computational cost is critical.

VI. CONCLUSIONS

This work introduced a VV-KRR-based modeling framework for IC buffers, leveraging data compression to achieve a balance between reduced model complexity and maintained accuracy. The results confirm the method's ability to closely match reference responses even with significant compression. Its seamless integration into SPICE-based solvers highlights the approach's practicality and scalability for enhancing signal and power integrity simulations.

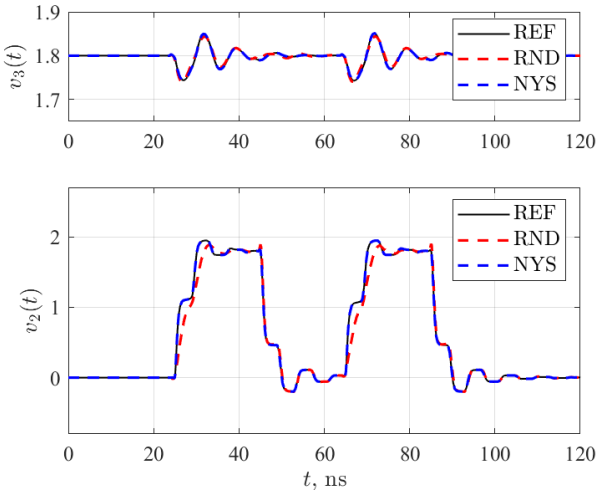

Fig. 4. Direct comparison of the best-performing Nyström and Random models with the reference responses in the scenario with the highest compression.

REFERENCES

[1] *I/O Buffer Information Specification, Ver. 7.2. Accessed: Dec. 12, 2023 [online]. Available: https://ibis.org/*

[2] G. Signorini, C. Siviero, M. Telescu, I.S. Stievano "Present and future of I/O-buffer behavioral macromodels," IEEE Electromagnetic Compatibility Magazine, vol. 5, no. 3, pp. 79–85, 2016.

[3] B. Mutnury, M. Swaminathan and J.P. Libous, "Macromodeling of nonlinear digital I/O drivers," IEEE Transactions on Advanced Packaging, vol. 29, no. 1, pp. 102–113, Feb. 2006.

[4] M. Souilem, N. Zgolli, T.R. Cunha, W. Dghais and H. Belgacem, "Signal and Power Integrity IO Buffer Modeling Under Separate Power and Ground Supply Voltage Variation of the Input and Output Stages," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 31, no. 6, pp. 874–886, June 2023.

[5] Y. Cao and Q. -J. Zhang, "A New Training Approach for Robust Recurrent Neural-Network Modeling of Nonlinear Circuits," IEEE Transactions on Microwave Theory and Techniques, vol. 57, no. 6, pp. 1539-1553, June 2009.

[6] H. Yu, T. Michalka, M. Larbi and M. Swaminathan, "Behavioral Modeling of Tunable I/O Drivers With Preemphasis Including Power Supply Noise," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 28, no. 1, pp. 233–242, Jan. 2020.

[7] R. Trinchero, T. Bradde, M. Telescu, I.S. Stievano "Modeling of IC Buffers from Channel Responses Via Machine Learning Kernel Regression", Proc. *28th IEEE Workshop on Signal and Power Integrity (SPI)*, Lisbon (Pt), May 12-15, 2024.

[8] N. Soleimani, R. Trinchero and F. G. Canavero, "Bridging the Gap Between Artificial Neural Networks and Kernel Regressions for Vector-Valued Problems in Microwave Applications," IEEE Trans. Microw. Theory Tech., vol. 71, no. 6, pp. 2319-2332, 2023.

[9] J.A.K. Suykens, et al., *Least Squares Support Vector Machines*, World Scientific Pub Co Inc, 2002.

[10] J. A. K. Suykens and J. Vandewalle, "Recurrent least squares support vector machines,"in IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, vol. 47, no. 7, pp. 1109–1114, July 2000.

[11] M. Atlante, R. Trinchero, T. Bradde, P. Manfredi and I.S. Stievano "Compressed SPICE-Compliant IC Models via Machine Learning Kernel Regression", Proc. *IEEE Electrical Design of Advanced Packaging and Systems (EDAPS)*, Bengaluru (India), December 17–19, 2024.

[12] A. J. Smola and B. Scholkop."Sparse Greedy Matrix Approximation for Machine Learning", in *Proc. of International Conference on Machine Learning (ICML)*, pag. 911–918, 2000.