POLITECNICO DI TORINO
Repository ISTITUZIONALE

IC Modeling via Machine Learning Regressions: A Data-Driven Approach to SPICE Integration

Original
IC Modeling via Machine Learning Regressions: A Data-Driven Approach to SPICE Integration / Atlante, M., Trinchero, R., Stievano, I.S., Telescu, M., Tanguy, N.. - In: IEEE TRANSACTIONS ON COMPONENTS, PACKAGING, AND MANUFACTURING TECHNOLOGY. - ISSN 2156-3950. - ELETTRONICO. - 15:9(2025), pp. 1814-1822. [10.1109/TCPMT.2025.3584470]

Availability:
This version is available at: 11583/3003138 since: 2025-10-30T11:47:06Z

Publisher:
IEEE

Published
DOI:10.1109/TCPMT.2025.3584470

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright
IEEE postprint/Author's Accepted Manuscript

©2025 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collecting works, for resale or lists, or reuse of any copyrighted component of this work in other works.

(Article begins on next page)

18 November 2026

IC Modeling via Machine Learning Regressions: a Data-Driven Approach to SPICE Integration

M. Atlante, *Student Member, IEEE,* R. Trinchero, *Member, IEEE,* I. S. Stievano, *Senior Member, IEEE,* M. Telescu, *Member, IEEE,* N. Tanguy

***Abstract*—This paper presents a method for generating accurate and efficient macromodels of high-speed I/O buffers. Extending existing techniques, the proposed approach enables a modular and scalable model generation tool based on machine-learning. Given the limitations of traditional methods, this work leverages kernel regression to develop SPICE-compliant models. Two compression schemes, random selection and Nyström approximation are used and thoroughly compared to reduce the number of expansion terms, with beneficial effects in terms of compactness of the SPICE implementation. The effectiveness of the method in terms of model accuracy and efficiency is stressed through real devices and typical signal and power integrity co-simulations.**

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

I. Introduction

THE design of modern high-performance electronic systems is increasingly challenged by stringent performance and technological constraints, which demand more complex evaluations to ensure system reliability. Engineers must conduct comprehensive Signal and Power Integrity (SIPI) and Electromagnetic Compatibility (EMC) assessments to mitigate critical issues such as crosstalk, simultaneous switching noise, immunity, and radiation. Advanced simulation techniques are increasingly necessary to predict power bouncing and signal propagation across interconnects, considering complex geometries and nonlinear device behavior.

In this framework, traditional transistor-level models based on internal physical descriptions of devices offer high accuracy, but are impractical due to their potential complexity and the resulting CPU time demand. Additionally, they risk disclosing proprietary device information or encrypted features specific to a particular SPICE solver. On the other hand, black-box models, such as behavioral models or macromodels, offer a promising and efficient alternative to transistor-level models [1]. Behavioral models infer mathematical relationships from observable responses at device ports or assume simplified circuital equivalents [2]–[11].

The most well-known example of behavioral modeling using equivalent circuits is the Input/Output Buffer Information Specification (IBIS) [12]. IBIS has been the gold standard in the industry for over thirty years, offering a structured framework enabling the derivation of IC port models with high numerical efficiency and broad availability. Many papers have been published in literature, offering enhancements aligned with the IBIS idea (e.g., see [3] and references therein). However, its reliance on predefined physical effects can constrain its flexibility, leading to a highly structured model construction and potentially compromising its performance in devices with a strong nonlinear nature.

In the above modeling framework, system identification techniques have emerged as a feasible and effective solution for the black-box modeling of ICs from recorded waveforms at device ports [4], [5], [13]. Among the various approaches, machine learning (ML) techniques, particularly recurrent neural networks (RNNs), have demonstrated high accuracy and adaptability in this context [9], [14]–[16]. However, their high computational demands and complex integration into simulation environments present significant challenges. Consequently, simpler yet effective alternatives are gaining attention.

Kernel regression techniques, particularly those based on feedforward architectures, offer a compelling balance between accuracy and simplicity. Using a nonlinear autoregressive exogenous (NARX) representation, kernel-based models achieve high fidelity while remaining compatible with standard simulation tools. This paper extends the published results and proposes compact SPICE-compliant models for IC buffers, employing kernel ridge regression (KRR) to accurately capture IC behavior, including power supply fluctuations [17]. This work presents a more rigorous formulation of the proposed NARX modeling approach, an in-depth discussion of the SPICE implementation, and an improved compression strategy based on adaptive Nyström approximation. Furthermore, the validation campaign is significantly expanded to include comparisons with IBIS models and a second test case featuring a custom low-voltage buffer with a non negligible I/O delay, thereby confirming the generality and scalability of the proposed method.

The remainder of this paper is organized as follows. We begin by introducing the problem under investigation in Section II, outlining its challenges. Next, we provide an overview of the mathematical background of KRR in Section III. Section IV presents a discussion on the implementation of the model equations in a SPICE environment. To improve efficiency, Section V explores two different approaches for compressing the kernel matrix. We then assess the accuracy and performance of the proposed regression model on two

M. Atlante, R. Trinchero, and I. S. Stievano are with the EMC Group, Department of Electronics and Telecommunications, Politecnico di Torino, 10129 Torino, Italy (e-mail: marco.atlante@polito.it, riccardo.trinchero@polito.it, igor.stievano@polito.it).

M. Telescu and N. Tanguy are with University of Brest, Lab-STICC, CNRS, UMR 6285, F-29200, Brest, France (e-mail: mihai.telescu@univ-brest.fr, noel.tanguy@univ-brest.fr).

test cases in Section VI. Finally, the paper concludes with a summary of key findings and potential directions for future work in Section VII.

II. Problem Statement

This section introduces the modeling problem addressed in this paper. When it comes to IC modeling, only the external layer – also known as the input/output (I/O) layer – is typically modeled. This is because it serves as the interface between the external interconnect and the internal core, which is more conveniently described using high-level description languages. The I/O layer is responsible for both loading and terminating the interconnects.

For simplicity, the discussion focuses on single-ended output buffers, like the one shown in Fig. 1, even if the proposed approach is general and it can be applied to more complex structures and alternative device technologies.

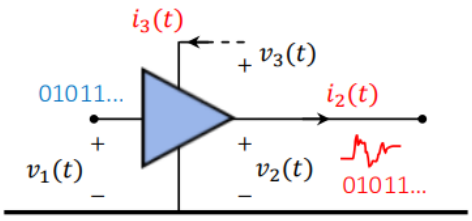


Fig. 1. Representation of a typical IC output buffer (driver) including its relevant electrical input and output variables.

The dynamic behavior of digital IC drivers arises from nonlinear interactions involving voltages, currents, and external factors such as supply variations and load conditions. Without loss of generality, the relationship among the *input*, *output*, and *power supply* ports can be described via the following implicit dynamic and nonlinear characteristic:

$$g\left(\boldsymbol{u}(t), \boldsymbol{y}(t), t, \frac{d}{dt}, \ldots, \frac{d^p}{dt^p}\right) = 0, \tag{1}$$

where $g(\cdot)$ is a general nonlinear dynamic function, the input vector $\boldsymbol{u}(t) = [v_1(t), v_2(t), v_3(t)]$ represents port voltages (input, output, supply), while the output vector $\boldsymbol{y}(t) = [i_2(t), i_3(t)]$ includes the currents at the output and supply ports, respectively.

To enable numerical modeling, all time-domain signals are sampled uniformly with period T_s, resulting in the corresponding discrete-time variables:

$$\boldsymbol{u}_k = [v_{1,k}, v_{2,k}, v_{3,k}] \text{ and } \boldsymbol{y}_k = [i_{2,k}, i_{3,k}], \tag{2}$$

where k is the discrete-time index, such that $t = kT_s$, $\mathbf{u}_k$ represents the input vector (e.g., the voltages at the device ports), while $y_k^{(i)}$ (for $i = 1, 2$) denotes each output variable at the k-th time step, corresponding to $i_2(kT_s)$ and $i_3(kT_s)$.

After discretization, the dynamic equation in (1) is reformulated in terms of a Nonlinear Output Error (NOE) model given by the following recursive equation [18], [19]:

$$\hat{y}_k^{(i)} = f_i(\boldsymbol{u}_k, \ldots, \boldsymbol{u}_{k-p}, \hat{y}_{k-1}^{(i)}, \ldots, \hat{y}_{k-p}^{(i)}), \tag{3}$$

where $\mathbf{u}_k$ is the input vector, $\hat{y}_k^{(i)}$ is the i-th output at time step k, and the parameter p denotes the model order. Whilst f_i represents a generic nonlinear recurrent map defining the NOE modeling in which the model prediction at a given time step depends on the predictions of the model at previous p time steps.

In the above scenario, building a model of an IC driver turns out to be equivalent to learning the recurrent nonlinear map f_i associated with each output from observed input-output data samples. The latter will be the goal of this paper, along with the implementation of the resulting model into a SPICE-like solver.

III. System Identification via NARX models with Kernel Ridge Regression

Accurately modeling the nonlinear dynamics of the port variables of IC buffer, and in general of electronic circuits, requires the availability of flexible regression techniques.

Recursive regression techniques, such as RNNs [16], [18], [19], can be applied to learn the recurrent map f_i of the NOE model in (3) from the training set $\mathcal{D}_{NOE} = \{(\boldsymbol{x}_j, y_j^{(i)})\}_{j=1+p}^{L}$, collecting the observed input-output data samples, in which the vector $\boldsymbol{x}_j = [\boldsymbol{u}_j, \ldots, \boldsymbol{u}_{j-p}]$ collects the current and past values of the model inputs and $y_j^{(i)}$ represents the corresponding current output value.

However, the learning problem associated with the NOE model is not fully compatible with the inherent feedforward structure of conventional kernel regression [18], [19]. While a recurrent formulation of kernel regression is theoretically possible, its training would require solving a non-convex optimization problem. This significantly limits its advantages over alternative regression techniques, such as RNNs, leading to a complex learning process and requiring a large number of training samples [18]–[21].

NARX models allow overcoming the above limitation by eliminating recursion from the learning phase, making them fully compatible with the standard feedforward structure used by kernel regressions [17]–[19], [22]. The NARX model can be formalized as:

$$\hat{y}_k^{(i)} = f_i(\boldsymbol{x}_k, y_{k-1}^{(i)}, \ldots, y_{k-p}^{(i)}), \tag{4}$$

where at each time step the model prediction $\hat{y}_k^{(i)}$ is obtained by evaluating f_i on the input $\boldsymbol{x}_k$ and on the true output values $y_{k-1}^{(i)}, \ldots, y_{k-p}^{(i)}$.

To learn the NARX model in (4) from data, the transient responses of the input and output waveforms must be organized into the dataset $\mathcal{D}_{NARX} = \{(\tilde{\boldsymbol{x}}_l, y_l^{(i)})\}_{l=1+p}^{L}$. Unlike the NOE dataset $\mathcal{D}_{NOE}$, where past predicted outputs are used as inputs, the NARX dataset incorporates the true output values. Specifically, the column vector $\tilde{\boldsymbol{x}}_l = [\boldsymbol{x}_l, y_{l-1}^{(i)}, \ldots, y_{l-p}^{(i)}]^T$ defines the input training features which include the input samples in $\boldsymbol{x}_l$ and the true past outputs $y_{l-1}^{(i)}, \ldots, y_{l-p}^{(i)}$.

Given the dataset $\mathcal{D}_{\text{NARX}}$, the KRR can be used to approximate the NARX model in (4) as:

$$\hat{y}_k^{(i)} = \sum_{l=1+p}^{L} \alpha_l^{(i)} k_i(\tilde{\boldsymbol{x}}_l, [\boldsymbol{x}_k, \hat{y}_{k-1}^{(i)}, \ldots, \hat{y}_{k-p}^{(i)}]^T)), \quad (5)$$

where $k_i(\cdot,\cdot)$ is the chosen kernel function measuring the correlation in the input space, and $\alpha_l^{(i)}$ are the regression coefficients learned during the training phase.

The regression coefficients $\boldsymbol{\alpha}^{(i)}$ are calculated as the ones that minimize the following regularized least-squares problem, which in matrix form is written as:

$$\min_{\boldsymbol{\alpha}^{(i)}} \|\boldsymbol{y}^{(i)} - \mathbf{K}^{(i)}\boldsymbol{\alpha}^{(i)}\|^2 + \lambda^{(i)}(\boldsymbol{\alpha}^{(i)})^\top \mathbf{K}^{(i)}\boldsymbol{\alpha}^{(i)}, \quad (6)$$

where $\boldsymbol{y}^{(i)} = [y_{1+p}^{(i)}, \ldots, y_L^{(i)}]^\top$ denotes the vector of target outputs, $\mathbf{K}^{(i)}$ is the Gramian kernel matrix obtained by evaluating the kernel on the input training pairs, and $\lambda^{(i)}$ is the hyperparameter associated with the regression regularizer [23], [24].

The regression unknowns $\boldsymbol{\alpha}^{(i)}$ can be computed by solving the learning problem in (6) in closed form. This leads to the following equation, which involves solving a linear system:

$$\boldsymbol{\alpha}^{(i)} = (\mathbf{K}^{(i)} + \lambda^{(i)}\mathbf{I})^{-1}\boldsymbol{y}^{(i)}, \quad (7)$$

where $\mathbf{I}$ is the identity matrix.

In this work, we use the Radial Basis Function (RBF) kernel, defined as:

$$k^{(i)}(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{x}'\|^2}{2\sigma_i^2}\right), \quad (8)$$

where σ_i is a kernel hyperparameter controlling the smoothness of the mapping. The Gaussian RBF kernel is chosen for its simplicity, making implementation in circuit solvers easier, and for its flexibility across various applications.

For each considered output, the hyperparameters to be optimized are the regularization coefficient $\lambda^{(i)}$ and the kernel parameter σ_i. Those parameters are optimized on the validation set using a Bayesian optimization scheme. The search ranges are defined logarithmically as $\lambda^{(i)} \in [10^{-11}, 10^1]$ and $\sigma_i \in [10^{-2}, 10^5]$.

IV. SPICE Implementation

This section describes a circuit implementation of the KRR model in (5) within SPICE-like solvers. The netlist defining the subcircuit that encapsulates the buffer model is outlined below, using HSPICE syntax.

```
* BUFFER MODEL
* Sampling time = TS
* MODEL ORDER = p
  .subckt KRR_buffer_model 1 2 3 4
  .param TS = ...
  .param p = ...
* MODEL IMPLEMENTATION (output, i2)
  G22 4 22 cur = 'g2(v1,v2,v3,i2)'
  V2  22 2 0 *i2
* MODEL IMPLEMENTATION (power supply, i3)
  G33 33 4 cur = 'g3(v1,v2,v3,i3)'
  V3  3 33 0 *i3
* I/O variables u=(v1,v2,v3); y=(i2,i3)
  E5 500 0 1 0 1 * v1
  E6 600 0 2 0 1 * v2
  E7 700 0 3 0 1 * v3
  H8 800 0 V2  1 * i2
  H9 900 0 V3  1 * i3
  .ends
```

The external pin labeled as `1`, `2`, `3` and `4`, correspond to the pertinent physical terminals allowing the definition of the input port voltage $v_1(t) =$ `V(1,4)`, output port voltage $v_2(t) =$ `V(2,4)` and power supply voltage $v_3(t) =$ `V(3,4)`, with `4` being the reference node. The above definition is coherent to the voltage variables defined in the schematic of Fig. 1. Also, the port currents $i_2(t)$ and $i_3(t)$ are defined in the subcircuit as the currents flowing through the probing voltage sources `V2` and `V3`, respectively.

Based on the above discussion, the involved electrical variables can be conveniently reinterpreted as the voltages at nodes in the range between `500` and `900` in the bottom part of the netlist. For this, voltage-controlled (element `E`) or current-controlled (element `H`) voltage sources are used, with a unitary scaling factor. The core of the netlist is represented by the behavioral controlled current sources `G22` and `G33`, one for each model's output variables i_2 and i_3, respectively.

As an example, the NARX model in (5) associated to i_2 at anytime instant t is explicitly defined via the behavioral current source `G22`, where the function `g2` implements the weighted sum of the kernel functions, such that:

$$i_2(t) = \sum_{l=1+p}^{L} \alpha_l^{(1)} \exp\left(-\frac{[\cdot]_l}{2\sigma_1^2}\right), \quad (9)$$

where according to the definition of the Gaussian RBF kernel in (8), the terms $[\cdot]_l$ provide the square of the Euclidean distance between the current regressor vector and the l-th training sample, which can be expressed as:

$$[\cdot]_l = (v_1(t) - \tilde{x}_{l1})^2 + \cdots + (v_1(t - pT_s) - \tilde{x}_{l(p+1)})^2 + \cdots (v_2(t) - \tilde{x}_{l(p+2)})^2 + \cdots + (v_2(t - pT_s) - \tilde{x}_{l(2p+2)})^2 + \cdots, \quad (10)$$

where all the training samples $\tilde{x}_{lj}$ used in the computation are stored in the matrix $\tilde{\boldsymbol{x}}$ constructed during the training phase of the model. The subscripts of $\tilde{\boldsymbol{x}}$ allows picking the entries of the l-th row of the matrix, and specifically the coefficients corresponding to the columns from 1 to $(4p + 3)$, being the latter index due to the present and past samples, i.e., $(p + 1)$, of each input (i.e., v_1, v_2, and v_3) and the past p samples of the output, i_2. The above expansion has been implemented using the set of standard mathematical operators such as `EXP` (exponential function) and `POW` (power function) available in any SPICE engine.

The SPICE implementation in (9) makes use of past samples of the voltage and current variables discretized at specific time instants. These samples can be conveniently obtained using a chain of matched transmission lines, each introducing a delay

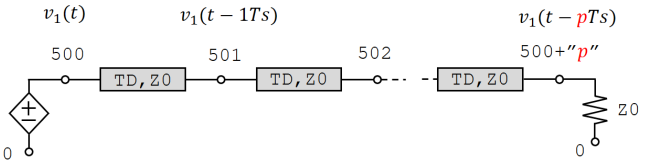


Fig. 2. Matched transmission line chain used to implement a delay of p samples for the port voltage v_1, with TD $= T_s$ and Z0 $= 50\,\Omega$.

equal to the sampling period T_s. As an example, Fig. 2 shows the schematic of the delay line associated to the voltage v_1. Similar structures are used for all the involved variables. The HSPICE implementation of above structure is shown below.

```
* Transmission lines for the variable v1
* (e.g. with p=4)
  T51 500 0 501 0 Z0=50 TD=TS
  T52 501 0 502 0 Z0=50 TD=TS
  T53 502 0 503 0 Z0=50 TD=TS
  T54 503 0 504 0 Z0=50 TD=TS
  R55 504 0  50
```

It is important to remark that the computational complexity of the proposed SPICE implementation grows with the number of expansion terms in the kernel model of (5), which is equal to the number of training samples, and with the dynamic order p. The latter determines the actual dimension of the input feature space, as it defines how many past samples should be considered for each variable.

These parameters directly affect the terms in the behavioral sources g2 and g3. Among these parameters, and for a given device, the only one that can be possibly adjusted is the number of expansion terms. This observation underscores the significance of model compression in achieving a compact and computationally efficient implementation. This aspect is thoroughly examined in Section V, where the effects of compression techniques on accuracy and performance are assessed, building on the analysis previously presented in [17], [22].

V. Compression Strategies for Kernel-based NARX Models

The modeling of dynamical systems from transient data poses a challenge due to the potentially long sequence of system response samples. As noted in Section IV, this results in a large number of training samples, which can reduce the efficiency of implementing the model in a SPICE-like solver. Indeed, the number of regression coefficients and kernel function evaluations $K_{\boldsymbol{x}}$ in (9) is directly proportional to the number of training samples. To mitigate the above issues, this Section presents two different compression strategies: the random sampling and Nyström approximation. As a general rule, the reduced number of coefficients (hereafter denoted as m) is initially set to a relatively small value -typically a few dozen - and gradually increased, provided that the accuracy on the validation set of the reduced model is maintained.

A. Random Selection

Random sampling simplifies the kernel expansion in (5), thereby accelerating both training and inference. This approach is particularly effective when the dataset contains highly correlated samples, as discarding some points has minimal impact on overall model performance. The idea is to train the model on a randomly selected subset of $m \ll L$ samples from the full dataset of size L [22].

We denote by $\mathcal{R} = \{r_1, \ldots, r_m\} \subset \{1, \ldots, L\}$ a set of m distinct indices selected uniformly at random without replacement. Each index r_j is drawn from a discrete uniform distribution, i.e.,

$$r_j \sim \mathcal{U}_d(1, L), \quad \text{with } r_i \neq r_j \text{ for } i \neq j. \tag{11}$$

The corresponding reduced dataset is defined as $\mathcal{D}_{\text{RND}} = \{(\tilde{\boldsymbol{x}}_l, y_l^{(i)})\}_{l \in \mathcal{R}} \subset \mathcal{D}_{\text{NARX}}$.

From this subset, the matrix $\bar{\mathbf{K}}^{(i)}$ is built, and the new coefficients $\bar{\boldsymbol{\alpha}}^{(i)}$ are computed using (7), where the bar notation indicates quantities derived from the reduced set. These coefficients are then used in (5) for predicting the system output.

$$\hat{y}_k^{(i)} = \sum_{l \in \mathcal{R}} \bar{\alpha}_l^{(i)} k_i(\tilde{\boldsymbol{x}}_l, [\boldsymbol{x}_k, \hat{y}_{k-1}^{(i)}, \ldots, \hat{y}_{k-p}^{(i)}]^T). \tag{12}$$

However, the randomness of the selection process may lead to subsets with overly similar samples, potentially omitting critical information and increasing performance variability. Moreover, the samples that are not selected for training are completely discarded, meaning they do not contribute to the training process in any way.

B. Nyström Compression

The Nyström approximation offers a more advanced and reliable alternative to random sampling by constructing a low-rank approximation of the kernel matrix using a subset of strategically chosen data points, known as landmarks.

Rather than computing the full kernel matrix explicitly, the Nyström method approximates it by selecting a small number of representative columns, $m \ll L$, that capture the essential structure of the dataset. This significantly reduces computational complexity, making it especially suitable for large-scale applications where efficiency is critical.

Unlike random sampling, which may overlook key data regions, the Nyström method provides a more structured and robust approximation by focusing on informative columns [25]. The kernel matrix $\mathbf{K}^{(i)}$ is approximated by selecting a subset of m columns indexed by $\mathcal{N}$, while the remaining columns correspond to the rest of the data:

$$\mathbf{K}^{(i)} \approx \tilde{\mathbf{K}}^{(i)} = \mathbf{K}_{Lm}^{(i)} (\mathbf{K}_{mm}^{(i)})^{-1} (\mathbf{K}_{Lm}^{(i)})^{\top}, \tag{13}$$

where $\mathbf{K}_{mm}^{(i)}$ is an $m \times m$ submatrix containing kernel evaluations between the selected points, and $\mathbf{K}_{Lm}^{(i)}$ is an $L \times m$ submatrix containing kernel evaluations between all data points and the selected ones.

The objective function in (6) can be reformulated as:

$$\min_{\boldsymbol{\alpha}^{(i)}} \|\boldsymbol{y}^{(i)} - \mathbf{K}_{Lm}^{(i)} \boldsymbol{\alpha}^{(i)}\|^2 + \lambda^{(i)} (\boldsymbol{\alpha}^{(i)})^{\top} \mathbf{K}_{mm}^{(i)} \boldsymbol{\alpha}^{(i)}, \tag{14}$$

leading to the closed-form solution for the coefficients $\tilde{\boldsymbol{\alpha}}^{(i)}$:

$$\tilde{\boldsymbol{\alpha}}^{(i)} = ((\mathbf{K}_{Lm}^{(i)})^\top \mathbf{K}_{Lm}^{(i)} + \lambda^{(i)} \mathbf{K}_{mm}^{(i)})^{-1} (\mathbf{K}_{Lm}^{(i)})^\top \boldsymbol{y}^{(i)}. \quad (15)$$

The quality of the approximation depends heavily on the selected columns. A basic approach is to sample them randomly [17]. Even though this simple approach does not guarantee that the minimal and optimal subset of landmarks is considered, potentially compromising the effectiveness of the compression.

Therefore, a more advanced selection scheme is considered in this work. Starting from a small initial set, columns are iteratively added by greedily selecting those that most reduce the current approximation error. The process repeats until m columns are selected. The above adaptive strategy is summarized in Alg. 1.

Algorithm 1 Nyström Approximation with Greedy Adaptive Column Selection

Require: Data matrix $\tilde{\boldsymbol{x}}$, kernel function $k_i(\cdot,\cdot)$, initial subset size n, number of iterations $T = (m - n)$
Ensure: Approximated kernel matrix $\tilde{\mathbf{K}}^{(i)}$
  Select n columns randomly from $\mathbf{K}^{(i)}$
  Compute initial approximation $\tilde{\mathbf{K}}^{(i)}$ using eq. (13)
  for $t = 1$ to T **do**
    Compute error matrix $\mathbf{E} = |\mathbf{K}^{(i)} - \tilde{\mathbf{K}}^{(i)}|$
    Compute column-wise error: $\mathbf{e} = \sum \mathbf{E}$
    Set $e_j = 0$ for all $j \in \mathcal{N}$ to avoid reselection
    Identify $j^* = \arg\max_j(e_j)$
    Add j^* to $\mathcal{N}$ and update the landmark set
    Recompute $\tilde{\mathbf{K}}^{(i)}$ using eq. (13) with the new set
  end for

Let us $\mathcal{N}$ denote the set of indices selected iteratively by the adaptive procedure in Algorithm 1. The associated reduced dataset is $\mathcal{D}_{\text{NYS}} = \{(\tilde{\boldsymbol{x}}_l, y_l^{(i)})\}_{l \in \mathcal{N}} \subset \mathcal{D}_{\text{NARX}}$.

From this subset, the final approximated matrix $\tilde{\mathbf{K}}^{(i)}$ is built, and the coefficients $\tilde{\boldsymbol{\alpha}}^{(i)}$ are computed using (15). These are then used in (5) to generate the model predictions:

$$\hat{y}_k^{(i)} = \sum_{l \in \mathcal{N}} \tilde{\alpha}_l^{(i)} k_i(\tilde{\boldsymbol{x}}_l, [\boldsymbol{x}_k, \hat{y}_{k-1}^{(i)}, \ldots, \hat{y}_{k-p}^{(i)}]^T). \quad (16)$$

While both random sampling and Nyström approximation reduce the number of coefficients $\boldsymbol{\alpha}^{(i)}$, the Nyström method provides a more structured compression by approximating the kernel matrix directly, preserving its internal structure.

VI. Application Example

To assess the proposed modeling approach, we test it on two buffer circuits with distinct features, serving as controlled benchmarks for accuracy, robustness, and generalization.

All simulations and model training are carried out on a Dell Precision 3490 equipped with an Intel Core Ultra 7 processor and 64 GB of RAM. MATLAB R2024b is used for training and optimization, while circuit-level simulations are performed using HSPICE U-2023.03-SP1 (WIN64).

A. Device A: Commercial Buffer

The first test case is a legacy 8-bit bus transceiver from Texas Instruments (model SN74ALVCH16973), operating at a nominal supply voltage of $V_{DD} = 1.8\,\text{V}$. This device is particularly suitable for benchmarking purposes due to the availability of both its SPICE-level transistor description and corresponding IBIS model, provided by the vendor. The transistor-level implementation serves as a trusted reference for all subsequent validations.

Custom scripts are used to generate training data, preprocess waveforms, and configure the model. The input signal $v_1(t)$ is defined by the predefined bit sequence "010100", with 20 ns bit period and 1 ns rise/fall time. To introduce some variability, the timing of the first transition event is stochastically perturbed across data subsets.

Training is performed across four distinct load conditions to capture a range of realistic scenarios. To emulate non-ideal power delivery, the supply voltage is forced by a source to have a flat behavior in the initial part of the transient with a stepwise waveform with $\pm 5\%$ relative fluctuations around the nominal power supply in the second part (see Fig. 3). The four loads are:

(1) A matched transmission line with a characteristic impedance of $60\,\Omega$ and a delay of 10 ns, terminated by a $100\,\Omega$ resistor. The power supply pin is connected to a lumped RL equivalent of a typical power net driven by the above mentioned voltage source;
(2) An open-ended transmission line with $75\,\Omega$ impedance and 4 ns delay. The power supply source is directly connected to the supply pin;
(3) A lumped RC load comprising a $50\,\Omega$ resistor in series with a 10 pF capacitor to ground. The power supply configuration is the same as in (2);
(4) A lumped resistive load of $150\,\Omega$. The power supply configuration is the same as in (1);

KRR models are generated by feeding the proposed tool with the above waveforms sampled at $T_s = 285\,\text{ps}$, yielding $L = 1680$ time-domain samples. A dynamic model order of $p = 4$ is employed. Model performance is tested on a fifth, independent configuration (see the last part of the transient curves in Fig. 3), featuring a $50\,\Omega$ transmission line with a 3 ns delay and an open-ended termination. The power network included a decoupling capacitor, followed by a lumped RL power supply network connected to the nominal power supply.

As an initial comparison, Fig. 4 shows the validation responses of the example device, alongside those generated using the full KRR model. The error, calculated as the difference between the reference and model responses, is also presented. Figure 5 shows a similar comparison in which the IBIS v3.2 model of the example device is used, as this is the only version available on the vendor's webpage. The IBIS model responses include the typical, slow, and fast corner cases. On one hand, this comparison demonstrates how the proposed *automatic* procedure, which involves feeding the KRR estimation with a set of transient responses, allows for the generation of accurate KRR models for both the output and power supply currents. Additionally, despite some non-negligible differences, IBIS

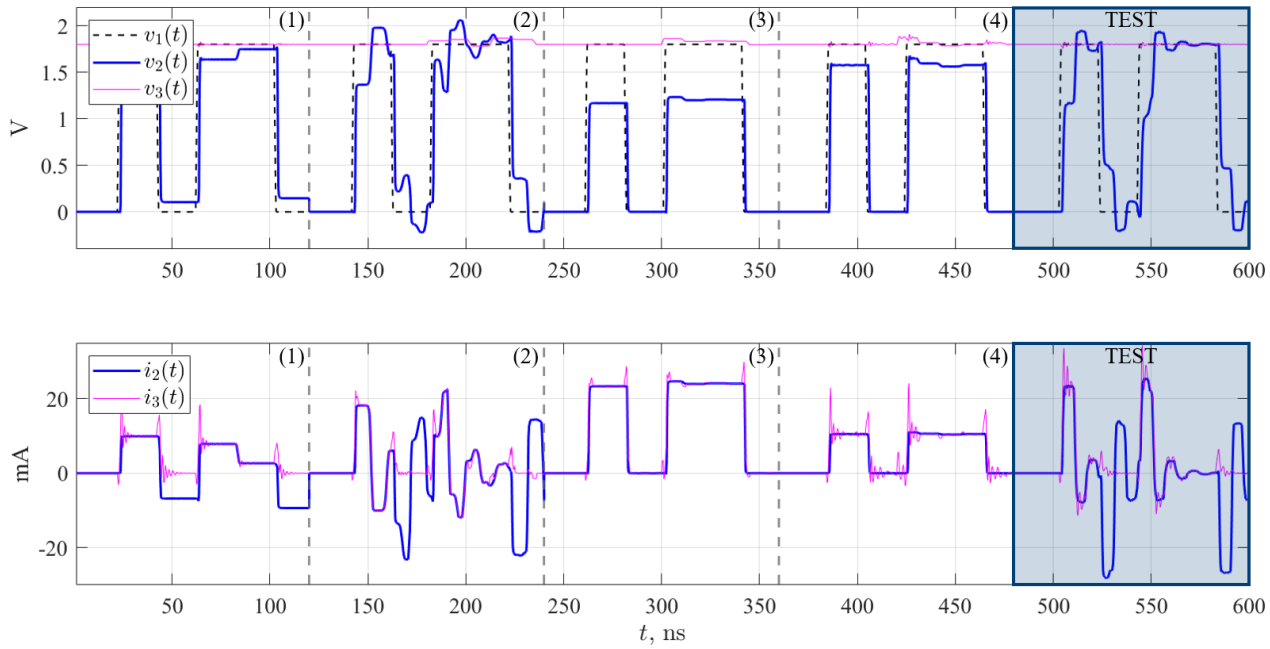


Fig. 3. Voltage and current responses across different load configurations (Device A). Each section, delimited by vertical dashed gray lines, corresponds to a distinct load condition used during training. The final segment, highlighted with a shaded blue background and labeled TEST, represents the configuration reserved for model validation. The top subplot shows the port voltages (in V): the input voltage $v_1(t)$ (black dashed line), the output gate voltage $v_2(t)$ (blue line), and the supply voltage $v_3(t)$ (magenta line). The bottom subplot shows the corresponding port currents (in mA): the output current $i_2(t)$ (blue line) and the supply current $i_3(t)$ (magenta line).

offers a functional model that offers reasonable performance. Discrepancies may arise when IBIS models are used without custom precise modeling or golden validation. The goal of this comparison is not to highlight the inaccuracies of the IBIS approach, but rather to showcase the viability of the proposed tool as a reliable alternative.

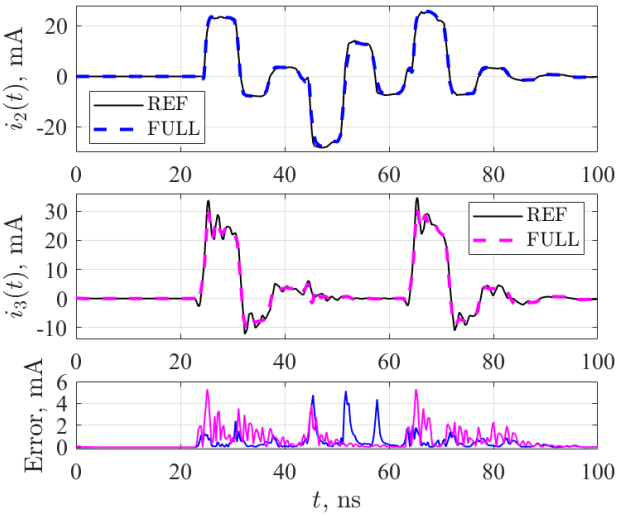


Fig. 4. Comparison between the reference simulation (REF) and the full kernel ridge regression model (FULL) for the port currents $i_2(t)$ and $i_3(t)$ of Device A during the validation phase. The bottom plot shows the absolute prediction errors.

To further dig into the potential of compressed KRR modeling, a comprehensive summary of model performance under different compression strategies is reported in Table I. The

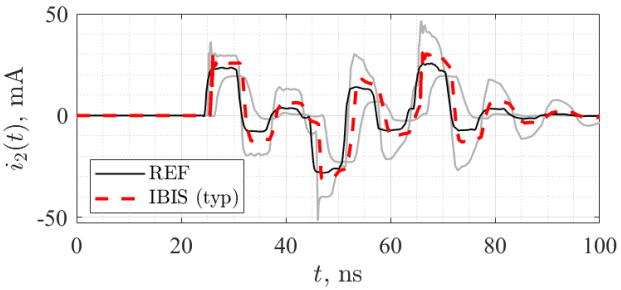


Fig. 5. Transient response of the port current $i_2(t)$ of Device A during the validation scenario. The reference simulation (REF) is shown as a solid black line, while the IBIS model (version 3.2) typical case is represented by a dashed red line. The gray lines correspond to the IBIS fast and slow corner cases.

table details: (i) the modeling technique employed (FULL dataset, i.e., no compression, Random sampling, or Nyström), (ii) the number of expansion terms L or m, (iii) the CPU simulation time in HSPICE for reproducing the validation test, and (iv, v) the average absolute error in predicting the output and the power supply currents. For the compressed models ($m = 700$ and $m = 200$), five independent simulations are executed, and the best-performing configuration is retained.

The training time of the KRR model remains manageable overall, although it varies significantly depending on the dataset size and the approximation method employed. Specifically, the uncompressed model requires approximately 70 seconds for training. When using random subset selection, training takes around 20 seconds for $m = 200$, and about 30 seconds for $m = 700$. In contrast, the Nyström method with column selection is more computationally demanding, though

it still executes within tens of seconds, requiring approximately 50 seconds for $m = 200$ and up to 200 seconds for $m = 700$.

TABLE I
PERFORMANCE OF COMPRESSION STRATEGIES (DEVICE A)

Model	Exp. Terms	CPU time HSPICE (s)	Avg err i_3 (mA)	Avg err i_2 (mA)
FULL	$L = 1680$	35.18	0.53	0.71
Random	$m = 700$	15.12	1.91	1.51
	$m = 200$	4.92	3.02	3.45
Nyström	$m = 700$	15.73	0.64	0.72
	$m = 200$	5.02	0.82	0.85

The results indicate that the Nyström method yields consistently superior accuracy relative to random sampling, especially for smaller dataset sizes. Although the full model achieves the best performance overall, the Nyström approach emerges as a compelling compromise between precision and computational efficiency, offering substantial reductions in simulation time with only marginal accuracy losses. It is important to point out that the table above presents results for only two values of m, which are fixed in the quantitative analysis for this example and the next one to ensure a fair comparison between the two compression strategies.

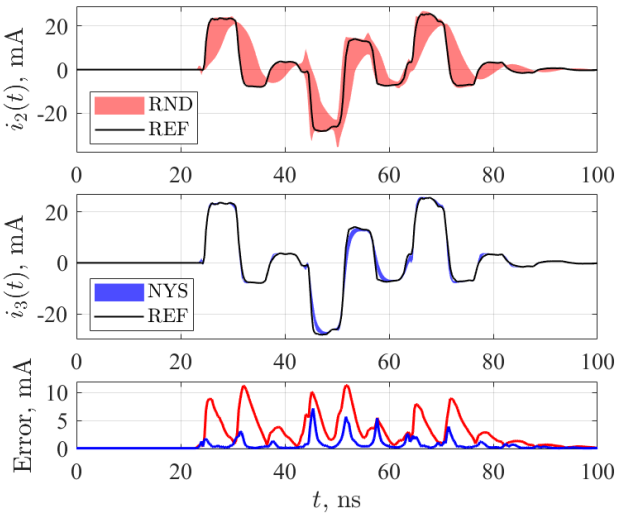


Fig. 6. Validation results for the output current $i_2(t)$ of Device A using compressed models. The top and middle panels compare the predictions obtained with random (RND) and Nyström (NYS) compression methods, respectively, against the reference simulation (REF). The bottom panel shows the absolute error in each case.

To further emphasize the implications of compression, Fig. 6 illustrates the variability of model predictions under both sampling techniques with $m = 200$ and five runs of compression. The top panel shows the envelope of predicted current waveforms $i_2(t)$, while the bottom panel quantifies the mean absolute error (MAE) over time. This comparison highlights how random sampling introduces greater response dispersion and model variance, due to its unstructured selection process, as evidenced by the wide red band in the figure. In contrast, Nyström-based models demonstrate significantly lower variance, thanks to their structured column selection strategy.

B. Device B: Custom Low-Voltage Buffer with Delay

A second custom device targeting high-speed, low-power memory interfaces is selected to further stress the modeling pipeline. This buffer operates with a nominal core voltage of $V_{DD} = 1\,\text{V}$ and an I/O voltage of $V_{\text{DD_IO}} = 1.2\,\text{V}$. For this device, we choose a sampling interval of $T_s = 42\,\text{ps}$ and a dynamic model order of $p = 6$. A peculiar feature of this device is the presence of a non-negligible delay between the input voltage $v_1(t)$ and the output response $v_2(t)$. This delay is quantified and discretized into d discrete time samples ($d = 12$). As a result, the model structure in (4) undergoes a slight modification, where the present and past samples of the variables v_2, v_3, i_2 and i_3, which feed the model equation, are all delayed by the same amount d. The SPICE implementation is accordingly adjusted to reflect this modification.

To preserve consistency and comparability, we retain the overall structure adopted for the previous device. Input stimuli are again synthesized in MATLAB, using logical bitstreams with $T_{\text{BIT}} = 5\,\text{ns}$ and $T_{\text{RF}} = 0.5\,\text{ns}$, and including stochastic perturbations on the initial transition. Power supply variations of $\pm 5\%$ are introduced as before, targeting the $V_{\text{DD_IO}}$ rail.

A KRR model is trained using the resulting dataset and validated via HSPICE simulations, in line with the previous example. Comparison with the transistor-level reference confirmed the model's ability to faithfully reproduce the device behavior. Table II reports quantitative results for this buffer, including the full model and compressed variants based on random selection and the Nyström method. The same structure of the table discussed above is used, thus confirming the same outcome already discussed for the first example device, highlighting that the Nyström method offers a win-win solution for generating accurate and faster IC models.

TABLE II
PERFORMANCE OF COMPRESSION STRATEGIES (DEVICE B)

Model	Exp. Terms	CPU time HSPICE (s)	Avg err i_3 (mA)	Avg err i_2 (mA)
FULL	$L = 2866$	80.64	0.58	0.23
Random	$m = 700$	19.87	0.65	0.36
	$m = 200$	6.55	1.05	0.63
Nyström	$m = 700$	18.12	0.61	0.25
	$m = 200$	7.12	0.64	0.27

Figure 7 compares the reference validation responses of Device B for both output and power supply signals with the predictions from the full model and the most compressed Nyström model. The Nyström method maintains excellent agreement with the reference simulation, even at the highest compression level ($m = 200$), confirming its robustness and efficiency.

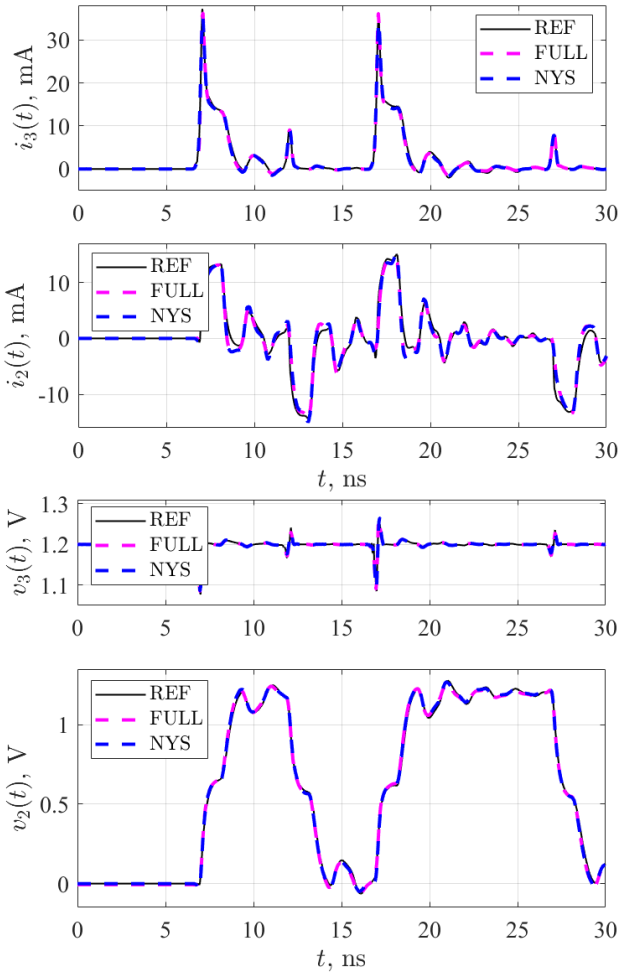


Fig. 7. Validation results for Device B showing output and supply voltage waveforms. Each panel compares the reference simulation (REF) with the predictions of the full model (FULL) and the most compressed Nyström model (NYS). The plotted signals are the output currents $i_3(t)$ and $i_2(t)$, the supply voltage $v_3(t)$, and the output voltage $v_2(t)$.

VII. Conclusions

This study presented a data-efficient modeling strategy for IC buffers based on KRR. By incorporating a compression scheme, the approach effectively reduces model complexity while preserving high accuracy. The results demonstrate that the proposed method maintains strong agreement with reference simulations, even under substantial data reduction. Additionally, its compatibility with SPICE-like environments confirms its suitability for scalable and practical integration of this class of behavioral IC models into signal and power integrity workflows.

References

[1] 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.

[2] T. Zhu, M.B. Steer and P.D. Franzon, "Accurate and Scalable IO Buffer Macromodel Based on Surrogate Modeling," IEEE Trans. Compon. Packag. Manuf. Technol., vol. 1, no. 8, pp. 1240–1249, Aug. 2011.

[3] 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 Trans. Very Large Scale Integration (VLSI) Systems, vol. 31, no. 6, pp. 874–886, June 2023.

[4] I.S. Stievano, I.A. Maio, F.G. Canavero, "Behavioral models of I/O ports from measured transient waveforms", IEEE Transactions on Instrumentation and Measurement, Vol. 51, No. 6, pp. 1266–1270, Dec. 2002.

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

[6] H. Yu and M. Swaminathan, "A Bit-Time-Dependent Model of I/O Drivers for Overclocking Analysis," IEEE Trans. Very Large Scale Integration (VLSI) Systems, vol. 28, no. 7, pp. 1630–1637, 2020.

[7] M. Souilem, J.N. Tripathi, W. Dghais and H. Belgacem, "An IBIS-like Modelling for Power/Ground Noise Induced Jitter under Simultaneous Switching Outputs (SSO)," *2019 IEEE 23rd Workshop on Signal and Power Integrity (SPI)*, Chambéry, France, 2019, pp. 1–4.

[8] Yi Cao, Runtao Ding and Qi-Jun Zhang, "State-space dynamic neural network technique for high-speed IC applications: modeling and stability analysis," in IEEE Trans. Microw. Theory Tech., vol. 54, no. 6, pp. 2398–2409, June 2006.

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

[10] T. Bradde, S. Grivet-Talocia, A. Zanco and G.C. Calafiore, "Data-driven extraction of uniformly stable and passive parameterized macromodels," *IEEE Access*, vol. 10, pp. 15786–15804, 2022.

[11] A. Carlucci, T. Bradde and S. Grivet-Talocia, "Addressing load sensitivity of rational macromodels," *IEEE Trans. Compon. Packag. Manuf. Technol.*, vol. 13, no. 10, pp. 1591–1602, 2023.

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

[13] J. Sjöberg, Q. Zhang, L. Ljung, A. Benveniste, B. Delyon, P.-Y. Glorennec, H. Hjalmarsson, and A. Juditsky, "Nonlinear black-box modeling in system identification: a unified overview," *Automatica*, vol. 31, no. 12, pp. 1691–1724, Dec. 1995.

[14] T. Bradde, S. Grivet-Talocia, P. Toledo, A.V. Proskurnikov, A. Zanco, G.C. Calafiore and P. Crovetti, "Fast Simulation of Analog Circuit Blocks Under Nonstationary Operating Conditions," *IEEE Trans. Compon. Packag. Manuf. Technol.*, vol. 11, no. 9, pp. 1355–1368, 2021.

[15] A. Carlucci, I.V. Gosea and S. Grivet-Talocia, "Data-Driven Modeling of Weakly Nonlinear Circuits via Generalized Transfer Function Approximation," *IEEE Access*, vol. 13, pp. 2746–2762, 2025.

[16] 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.

[17] M. Atlante, R. Trinchero, I.S. Stievano, M. Telescu, N. Tanguy "SPICE-based Behavioral Models of IC Buffers via Compact Kernel Regressions", Proc. *29th IEEE Workshop on Signal and Power Integrity (SPI)*, Gaeta (Italy), May 11-14, 2025.

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

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

[20] A. Rudi, L. Carratino, and L. Rosasco, "Falkon: An optimal large scale kernel method," Advances in neural information processing systems, vol. 30, 2017.

[21] 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.

[22] 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.

[23] J. Shawe-Taylor and N. Cristianini, *Kernel methods: an overview.* Kernel Methods for Pattern Analysis, Cambridge: Cambridge University Press, 2004, pp. 25–46.

[24] B. Scholkopf, R. Herbrich, and A. J. Smola, "A generalized representer theorem",in *Proc. 14th Annu. Conf. Comput. Learn. Theory*, 2001, pp. 416–426.

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



Marco Atlante received the M.Sc. degree in Biomedical Engineering (Instrumentation track) from Politecnico di Torino, Italy, in 2024. During his Master's studies, he spent one semester as an international exchange student at the University of Bilbao, Spain. He is currently a first-year Ph.D. student involved in a joint doctoral project between Italy and France, in collaboration with the University of Brest (UBO). His research focuses on modeling techniques and machine learning methods applied to electrical and microwave systems and devices.



Riccardo Trinchero (M'16) received the M.Sc. and the Ph.D. degrees in Electronics and Communication Engineering from Politecnico di Torino, Torino, Italy, in 2011 and 2015, respectively. He is currently an Associate Professor within the EMC Group with the Department of Electronics and Telecommunications at the Politecnico di Torino. His research interests include the analysis of switching DC-DC converters, machine learning, and statistical simulation of circuits and systems.



Igor S. Stievano (M'98–SM'08) received the master's degree in electronic engineering and the Ph.D. degree in electronics and communication engineering from the Politecnico di Torino, Turin, Italy, in 1996 and 2001, respectively. He is currently a Professor of Electrical Engineering with the Department of Electronics and Telecommunications, Politecnico di Torino. He has authored or co-authored more than 140 papers published in international journals and conference proceedings. His research interests mainly relate to modeling and simulation of electrical and electronic systems, with emphasis on behavioral modeling of digital integrated circuits, signal and power integrity, energy networks, statistical and machine learning methods.



Mihai Telescu obtained his Engineer's Degree in Electronics and Telecommunications from the Polytechnic University of Timisoara in 2003. He obtained a Master of Science degree in France in 2004, jointly delivered by the ENSSAT of Lannion and the University of Brest. He received a Doctorate in Electronics from the University of Brest, France in 2007. He was a post doc researcher with the EMC Group at the Politecnico di Torino, Italy between 2008 and 2009. Since 2009 he is an Associate Professor with the UBO. His research interests include linear and non-linear macromodeling, automated design, model order reduction, system identification, uncertainty quantification.



Noël Tanguy received the M.Sc. and the Ph.D. degrees (Thèse de Doctorat) in electronics from the University of Brest (UBO), Brest, France, in 1989, and 1994, respectively, where he received the HDR degree (accreditation to supervise research) in 2006. He was an Associate Professor between 1995 and 2008 and has been a Full Professor with UBO ever since. He currently conducts his research with the Lab-STICC (Laboratoire des sciences et technologies de l'information, de la communication et de la connaissance, UMR CNRS). His research interests include signal integrity, modeling, and approximation theory, model order reduction, linear and nonlinear macromodeling, impairments estimation and compensation techniques in optical communications, stochastic analysis, and related topics