electronics

Article

Compressed Complex-Valued Least Squares Support Vector Machine Regression for Modeling of the Frequency-Domain Responses of Electromagnetic Structures

Nastaran Soleimani and Riccardo Trinchero *

Department of Electronics and Telecommunications, Politecnico di Torino, 10129 Torino, Italy; nastaran.soleimani@polito.it
* Correspondence: riccardo.trinchero@polito.it

Abstract: This paper deals with the development of a Machine Learning (ML)-based regression for the construction of complex-valued surrogate models for the analysis of the frequency-domain responses of electromagnetic (EM) structures. The proposed approach relies on the combination of two-techniques: (i) the principal component analysis (PCA) and (ii) an unusual complex-valued formulation of the Least Squares Support Vector Machine (LS-SVM) regression. First, the training and test dataset is obtained from a set of parametric electromagnetic simulations. The spectra collected in the training set are compressed via the PCA by exploring the correlation among the available data. In the next step, the compressed dataset is used for the training of compact set of complex-valued surrogate models and their accuracy is evaluated on the test samples. The effectiveness and the performance of the complex-valued LS-SVM regression with three kernel functions are investigated on two application examples consisting of a serpentine delay structure with three parameters and a high-speed link with four parameters. Moreover, for the last example, the performance of the proposed approach is also compared with those provided by a real-valued multi-output feedforward Neural Network model.

Keywords: Least Squares Support Vector Machine; serpentine delay line; high-speed interconnect link; principal component analysis

Citation: Soleimani, N.; Trinchero, R. Compressed Complex-Valued Least Squares Support Vector Machine Regression for Modeling of the Frequency-Domain Responses of Electromagnetic Structures. *Electronics* **2022**, *11*, 551. https://doi.org/10.3390/electronics11040551

Academic Editor: Manuel Arrebola

Received: 29 December 2021
Accepted: 8 February 2022
Published: 11 February 2022

1. Introduction

In recent decades, Machine Learning (ML) methods have been widely applied to construct accurate and fast-to-evaluate surrogate models able to reproduce the input–output behavior of electromagnetic (EM) structures as a function of deterministic and uncertain parameters [1–14]. In the above scenario, advanced data-driven and ML-based regressions, such as Polynomial Chaos Expansion [1–4], Support Vector Machine (SVM) regression [5,6], Least-Squares Support Vector Machine (LS-SVM) regression [7], Gaussian Process regression (GPR) [8], and feedforward [9–12], deep [12,13], convolutional [12] and Long Short-Term Memory (LSTM) [14] neural networks (NNs), have been successful applied to uncertainty quantification (UQ) and optimization in EM applications. The common idea is to adopt the above methods to train a regression model by using a limited number of training samples generated via a set of expensive full-wave or circuital simulations with the so-called computational model. Then, since the resulting surrogate model is known in a closed form, it can be suitably embedded within UQ and an optimization scheme as an efficient and effective "surrogate" of the expensive computational model [15].

Despite the high performance shown in realistic electronic and EM applications, most of the ML techniques have been developed for dealing with real-valued data [16–18]. However, complex-valued data are widely used in electronic applications (e.g., for AC simulations and frequency-domain analysis). The simplest strategy for extending the applicability of real-valued ML techniques to the case of complex-valued data is based

on the so-called dual-channel formulation [19–21]. The underlaying idea is to recast the complex-valued problem into two uncorrelated real-valued ones, by stacking the real and imaginary part of the complex input and output values. The main advantage of the above procedure is that plain real-valued ML techniques can be directly adopted without requiring any generalization or improvement. However, such an approach completely ignores the possible correlation among the real and imaginary parts of the complex-valued output, thus leading to possible issues regarding accuracy and robustness to noise [19,22]. For the above reasons, alternative pure complex-valued formulations have been proposed for several ML techniques, such as NN [22], SVM regression [23], kernel Least Squares regression [19–21,24,25], LS-SVM regression [7] and GPR [26].

This paper investigates the performances of a pure-complex implementation of a kernel-based ML technique such as the LS-SVM regression. The proposed complex-value formulation is based on the theoretical framework developed in [19,21,24] for a plain kernel-based regression in which the regularization term is missing. Indeed, like the SVM regression, the LS-SVM regression is a powerful and well-consolidated regression technique combining the beneficial effect of the kernel and of the Tikhonov regularization [17,25,27]. Specifically, thanks to the kernel trick, the above regression allows the construction of a non-parametric model in which the number of unknowns to be estimated by the regression problem is independent from the dimensionality of the input space. Furthermore, the regularizer terms limits the overfitting issue and increases the model generalization.

The aim of this work is twofold. First, the mathematical formulation of the complex-value LS-SVM regression is developed, and the mathematical link between the pure complex formulation and the dual channel one is highlighted, where the latter is a special case of the more general complex-valued formulation. Second, the performance of compressed surrogate models constructed via the dual-channel and the pure complex-value LS-SVM regression with a complex and pseudo kernel are investigated by considering the frequency-domain responses of two test-cases consisting of a serpentine delay structure and a high-speed link, with three and four parameters, respectively. It is important to stress that, in the above applications, a compressed representation of the frequency-domain responses based on principal component analysis (PCA) [18,28–30] has been used to remove the redundant information and to reduce the number of LS-SVM-based models that need to be trained [31]. For the sake of completeness, for the second example, the results of the proposed approaches are compared with those provided by a real-valued multi-output feedforward NN model [11,12].

The paper is organized as follows. Section 2 describes the problem statement addressed during the paper, and its challenges. Section 3 provides an overview of the mathematical background of the PCA. Section 4 provides a self-contained formulation of the complex-valued LS-SVM regression with specific emphasis on the differences between the dual-channel and the pure complex implementation. In Section 5, the accuracy and the performance of the proposed complex-valued LS-SVM regression are investigated on two test cases consisting of a serpentine delay line structure and a transfer function of a high-speed interconnect link. Finally, the paper ends with the conclusions in Section 6.

2. Problem Statement and Challenges

Starting from a training set $D = \{(\mathbf{x}_i, y_i(f_k))\}_{i,k=1}^{L,K}$ and collecting the configuration of the input parameters $\mathbf{x}_\mathbf{i} = [x_{i,1}, \ldots, x_{i,d}]^T \in \mathcal{X}$, with $\mathcal{X} \in \mathbb{C}^d$ and the corresponding output $y_i(f_k) \in \mathbb{C}$ computed by a full-computational model (i.e., $y_i(f_k) = \mathcal{M}(f_k; \mathbf{x}_i)$) for a set of values of the independent variable f_k (e.g., the frequency), our goal is to build a surrogate model $\widetilde{\mathcal{M}}$ that approximates the training data and is able to generalize well on the "unseen" data, usually known as test samples, such as:

$$y(f_k; \mathbf{x}) \approx \widetilde{\mathcal{M}}(f_k; \mathbf{x}), \tag{1}$$

For any $\mathbf{x} \in \mathcal{X}$ and for $k = 1, \ldots, K$.

Without loss of generality, we seek a data-driven regression technique able to provide an accurate and efficient approximation of the actual behavior of the computational model $\mathcal{M}(f_k; \mathbf{x}_i)$. This means that the surrogate model $\widetilde{\mathcal{M}}$ should be constructed by using a small set of training samples (i.e., L should be as small as possible), since the training samples are usually generated via a set of computational expensive simulations based on the full-computational model $\mathcal{M}$.

The above modeling problem is rather challenging. First, the regression technique should be able to work in the complex domain. Moreover, the resulting surrogate model should be able to mimic the behavior of a multi-output complex-valued function provided by the computational model (i.e., the output values $y_i(f_k)$), in which its values unavoidably depend on both the configuration of the input parameters $\mathbf{x}_i$ and on the variable f_k.

A possible modeling scheme consists of considering the free variable f_k as an extra input parameter. This means that we are seeking a "single" advanced model able to represent in a closed-form the impact of both the parameter $\mathbf{x}$ and the frequency f_k on the system output. As an example, such a model can be obtained via a plain or recurrent NN [12,13]. However, due to the complexity of the problem at hand, the resulting neural network structure usually requires a large number of neurons and several hidden layers. Moreover, despite their flexibility, the training of NN-based structures requires the solution of a non-convex optimization leading to training issues (i.e., expensive training time and/or huge number of training samples [13,32]).

In order to partially overcome the above issues, we can think of using a single-output regression trained via the solution of a convex optimization problem, such as the plain kernel Least Squares regression [21], the LS-SVM regression [33], or the SVM regression [34], and build one model for each frequency point. In this way, the model generation requires less training samples; unfortunately, however, the overall model requires training K single-output surrogates, thus making the training process extremely expensive and cumbersome when a large number of frequency points are considered.

A data compression strategy can be seen as a good compromise between the two modeling schemes depicted in the previous section. As an example, PCA [18,28–30] allows the extraction of the inherent correlation existing among several realizations of output data samples at different frequency points, thus leading to a compressed representation of the frequency spectra. In such a scenario, the number of actual single-output models required to represent the data can be heavily reduced.

3. PCA Compression

This section briefly presents the mathematical background of the PCA [18,28–30]. Let us consider the output dataset $\{y_i(f_k)\}_{i,k=1}^{L,K}$ that collects L spectra computed for different configurations of the input parameters (i.e., the number of training outputs), each having K frequency points. The above dataset can be recast as a $K \times L$ matrix $\mathbf{Y}$, such that the element $\mathrm{Y}_{i,k} = y_i(f_k)$. For normalization purposes, a zero-mean matrix is considered:

$$\widetilde{\mathbf{Y}} = \boldsymbol{Y} - \boldsymbol{\mu}\,, \tag{2}$$

where $\boldsymbol{\mu}$ is the mean value of $\mathbf{Y}$, calculated row-wise and subtracted column-wise

The new rectangular matrix $\widetilde{\mathbf{Y}} \in \mathbb{C}^{K\times L}$ can be represented by means of the compact singular value decompositions (SVD) [28,30]:

$$\widetilde{\mathbf{Y}} = \mathbf{U}\mathbf{S}\mathbf{V}^{\mathrm{H}}, \tag{3}$$

where, assuming that there are L columns of $\mathbf{U}$ and $\mathbf{V}$ associated with non-zero singular values, $\mathbf{U} = [\mathbf{u}_1, \ldots \mathbf{u}_\mathbf{L}] \in \mathbb{C}^{K\times L}$ and $\mathbf{V} = [\boldsymbol{v_1}, \ldots, \boldsymbol{v}_L] \in \mathbb{C}^{L\times L}$ are orthogonal matrices, such that $\mathbf{U}^{\mathrm{H}}\mathbf{U} = \mathbf{V}^{\mathrm{H}}\mathbf{V} = \boldsymbol{I}_{L\times L}$, collecting the left and right singular vectors, and $\mathrm{S} = \mathrm{diag}\{(\sigma_1, \ldots \sigma_\mathrm{L})\} \in \mathbb{C}^{L\times L}$ is a diagonal matrix collecting the singular values sorted in

a decreasing order $\sigma_1 \gg \sigma_2 \gg \cdots \gg \sigma_L$. Now, a compressed approximation of the actual matrix $\widetilde{\mathbf{Y}}$ can be obtained as follows:

$$\widetilde{\mathbf{Y}} \approx \widetilde{\mathbf{U}}\widetilde{\mathbf{S}}\widetilde{\mathbf{V}}^{\mathrm{H}}, \tag{4}$$

where $\widetilde{\mathbf{U}} = [\mathbf{u}_1, \ldots, \mathbf{u}_n] \in \mathbb{C}^{K \times \tilde{n}}$ with $\mathbf{u}_i \in \mathbb{C}^{K \times 1}$ and $\widetilde{\mathbf{V}} = [\mathbf{v}_1, \ldots, \mathbf{v}_n] \in \mathbb{C}^{\tilde{n} \times L}$ with $\mathbf{v}_i \in \mathbb{C}^{L \times 1}$ are the reduced left and right-eigenvector matrices collecting only the first $\tilde{n}$ components (i.e., the first $\tilde{n}$ columns of the original matrices $\mathbf{U}$ and $\mathbf{V}$), and $\widetilde{\mathbf{S}} = diag\{(\sigma_1, \ldots, \sigma_{\tilde{n}})\}$ is a reduced diagonal matrix containing the first $\tilde{n}$ singular values.

The above relationship can be used to obtain a compressed representation $\mathbf{Z} \in \mathbb{C}^{\tilde{n} \times L}$ of the original matrix $\mathbf{Y}$, such that:

$$\mathbf{Z} = \widetilde{\mathbf{U}}^{H}\widetilde{\mathbf{Y}} = \widetilde{\mathbf{S}}\widetilde{\mathbf{V}}^{\mathrm{H}}, \tag{5}$$

It is important to note the resulting compressed matrix $\mathbf{Z} \in \mathbb{C}^{\tilde{n} \times L}$ is smaller than the original matrix $\widetilde{\mathbf{Y}}$, since usually $\tilde{n} \ll K$. Moreover, the rows of the compressed matrix $\mathbf{Z}$ can be considered to be the realizations of a new set of output variables $\{\mathbf{z}(\mathbf{x}_l)\}_{l=1}^{L}$, with $\boldsymbol{z}(\mathbf{x}_l) \in \mathbb{C}^{\tilde{n} \times 1}$, which can be considered as the collection of L samples of a compressed $\tilde{n}$-dimensional output variable. Specifically, the element (i, j) of the matrix $\mathbf{Z}$, corresponds to the i-th output of the compressed representation evaluated at the j-th configuration of the input parameters, i.e., $Z_{ij} = z_i(\mathbf{x}_j)$. This means that only $\tilde{n}$ single-output models need to be trained to represent the whole dataset provided by the matrix $\mathbf{Y}$, thus leading to a substantial improvement in the training time. Once a surrogate model for each of the $\tilde{n}$ components of the compressed multi-output representation in $\mathbf{Z}$ is available, the overall compress surrogate can be inexpensively used to predict the system output $\mathbf{y}\left(\bar{\mathbf{x}}\right) \in \mathbb{C}^{K \times 1}$ for a generic test sample $\bar{\mathbf{x}} \in \mathcal{X}$ as follows:

$$y\left(\bar{\mathbf{x}}\right) \approx \boldsymbol{\mu} + \widetilde{\mathbf{U}}\overline{\mathbf{Z}}\,, \tag{6}$$

where $\overline{\mathbf{Z}} = \left[z_1\left(\bar{\mathbf{x}}\right), \ldots, z_{\tilde{n}}\left(\bar{\mathbf{x}}\right)\right]^{\mathrm{T}} \in \mathbb{C}^{\tilde{n} \times 1}$.

It is important to remark that the compressed representation $\boldsymbol{z}(\mathbf{x})$ provided by the PCA compression allows approximating the actual data $\mathbf{y}(\mathbf{x})$ with a tunable accuracy depending on the number of PCA components $\tilde{n}$, such that [28]:

$$\left(\frac{\sigma_{\tilde{n}+1}}{\sigma_1}\right)^2 \leq \varepsilon^2, \tag{7}$$

where ε is a given error threshold tuned by the user.

4. Complex Valued Least-Square Support Vector Machine Regression

The LS-SVM regression is a kernel-based regression with an L2 regularizer, which allows constructing a non-parametric model in which the number of unknowns and complexity is independent from the number of input parameters via the solution of a simple linear system, i.e., by solving a convex optimization problem [33].

However, the plain formulation of such a technique has been developed for real-valued datasets only. The aim of this section is to extend the mathematical formulation of the LS-SVM regression to the more general case of the complex-data problem. The proposed formulation is based on the preliminary results presented in [25] and on the results presented in [19,21] for a classical reproducing kernel Hilbert space (RKHS) regression, in which the regularizer term is neglected. For the sake of simplicity, the following complex valued formulation is developed for the simplified case of single-output regression, but all the calculations can be easily extended to the case of a multi-output problem.

Starting from a set of complex-valued samples $\mathcal{D} = \{(\mathbf{x}_l, y_l)\}_{l=1}^{L}$ where $\mathbf{x}_l \in \mathbb{C}^d$ and $y_l = y(\mathbf{x}_l) \in \mathbb{C}$, the primal space formulation of the LS-SVM regression is written:

$$y \approx \widetilde{\mathcal{M}}(\mathbf{x}) = \sum_{i=1}^{N} w_i \phi_i^*(\mathbf{x}) + b = \mathbf{w}, \mathbf{\Phi}(\mathbf{x}) + b, \tag{8}$$

where $\mathbf{w} = [w_1, \ldots, w_N]^T = \mathbf{w}_R + j\mathbf{w}_I \in \mathbb{C}^N$ are the complex regression coefficients (i.e., $w_i = w_{i,R} + jw_{i,I}$), $\mathbf{\Phi}(\mathbf{x}) = \mathbf{\Phi}_R(\mathbf{x}) + j\mathbf{\Phi}_\mathrm{I}(\mathbf{x}) = [\phi_1(\mathbf{x}), \ldots, \phi_N(\mathbf{x})]^T$ is a vector-valued complex function $\mathbf{\Phi}(\cdot) : \mathbb{C}^d \to \mathbb{C}^N$ collecting the complex-valued basis functions $\phi_\mathrm{i}(\mathbf{x})$ that provide a map between the parameter space and the feature space, $b = b_R + jb_I$ is the bias term, and $\mathbf{w}, \mathbf{\Phi}(\mathbf{x}) = \mathbf{\Phi}^\mathrm{H}(\mathbf{x})\mathbf{w} = \mathbf{w}^T\mathbf{\Phi}^*(\mathbf{x})$ is the inner product in the complex domain.

The primal space formulation of the complex-valued regression in Equation (8) can be written in term of its real and imaginary components:

$$\widetilde{\mathcal{M}}(\mathbf{x}) = \widetilde{\mathcal{M}}_R(\mathbf{x}) + j\widetilde{\mathcal{M}}_I(\mathbf{x}) = \Big(\boldsymbol{w}_R\mathbf{\Phi}_R^T(\mathbf{x}) + \mathbf{w}_I\mathbf{\Phi}_I^T(\mathbf{x}) + b_R\Big) + j\Big(\mathbf{w}_\mathrm{I}\mathbf{\Phi}_R^T(\mathbf{x}) - \boldsymbol{w}_R\ \mathbf{\Phi}_I^T(\mathbf{x}) + b_i\Big)\ , \tag{9}$$

In the above primal space formulation, the regression unknowns (i.e., the coefficients in $\mathbf{w}_R$ and $\mathbf{w}_I$, and the bias term b_R and b_I, respectively) are estimated by solving the following convex optimization problem:

$$\min_{\mathbf{w}_R, \mathbf{w}_I, b_R, b_I, \mathbf{e}_R, \mathbf{e}_I} \frac{1}{2}\mathbf{w}_R^T\mathbf{w}_R + \frac{1}{2}\mathbf{w}_I^T\mathbf{w}_I + \frac{\gamma_R}{2}\sum_{l=1}^{L} e_{R,l}^2 + \frac{\gamma_I}{2}\sum_{l=1}^{L} e_{I,l}^2 \tag{10}$$

Such that

$$\mathbf{e_{R,l}} = \boldsymbol{Re}\Big\{y_l - \widetilde{\boldsymbol{\mathcal{M}}}(\mathbf{x}_l)\Big\} = \boldsymbol{y_{R,l}}\ - (\mathbf{w_R}\mathbf{\Phi_R}(\mathbf{x_l}) + \mathbf{w_I}\mathbf{\Phi_I}(\mathbf{x_l}) + \boldsymbol{b_R})$$

$$\boldsymbol{e_{R,l}} = \boldsymbol{Im}\Big\{\boldsymbol{y_l} - \widetilde{\boldsymbol{\mathcal{M}}}(\mathbf{x}_l)\Big\} = \boldsymbol{y_{I,l}}\ - (\mathbf{w_I}\mathbf{\Phi_R}(\mathbf{x_l}) - \mathbf{w_R}\mathbf{\Phi_I}(\mathbf{x_l}) + \boldsymbol{b_I}\Big)$$

For $l = 1, \ldots, L$, where $\boldsymbol{w_R^T w_R} + \boldsymbol{w_I^T w_I} = \boldsymbol{w_2^2}$ is the L2 regularizer and the terms $e_{R,l}^2$ and $e_{I,l}^2$ provide the error of a squared loss function.

The Lagrangian for the above constraint optimization problem is written:

$$\begin{aligned}
&\mathcal{L}(\boldsymbol{w}_R, \boldsymbol{w}_I, b_R, b_I, \boldsymbol{e}_R, \boldsymbol{e}_I, \boldsymbol{\alpha}_R, \boldsymbol{\alpha}_I) \\
&\qquad = \tfrac{1}{2}\boldsymbol{w}_R^T\boldsymbol{w}_R + \tfrac{1}{2}\boldsymbol{w}_I^T\boldsymbol{w}_I + \tfrac{\gamma_R}{2}\sum_{l=1}^{L} e_{R,l}^2 + \tfrac{\gamma_I}{2}\sum_{l=1}^{L} e_{I,l}^2 \\
&\qquad - \sum_{l=1}^{L} \alpha_{R,l}\big\{e_{R,l} - y_{R,l}\ + \big(\mathbf{w}_R^T\mathbf{\Phi}_R(\mathrm{x}_l) + \mathbf{w}_I^T\mathbf{\Phi}_I(\mathrm{x}_l) + b_R\big)\big\} \\
&\qquad - \sum_{l=1}^{L} \alpha_{I,l}\big\{e_{I,l} - y_{I,l}\ + \big(\mathbf{w}_I^T\mathbf{\Phi}_R(\mathrm{x}_l) - \mathbf{w}_R^T\mathbf{\Phi}_I(\mathrm{x}_l) + b_I\big)\big\}\ ,
\end{aligned} \tag{11}$$

where $\alpha_l = \alpha_{R,l} + j\alpha_{I,l}$ are the Lagrangian multipliers such that $\alpha_{R,l}, \alpha_{I,l} \geq 0$ for $l = 1, \ldots, L$.

By computing the partial derivatives of the Lagrangian $\mathcal{L}(\boldsymbol{w}_R, \boldsymbol{w}_I, b_R, b_I, \boldsymbol{e}_R, \boldsymbol{e}_I, \boldsymbol{\alpha}_R, \boldsymbol{\alpha}_I)$ with respect to its parameters:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}_R} = 0 \to \mathbf{w}_R = \sum_{l=1}^{L}[\alpha_{R,l}\mathbf{\Phi}_R(\mathrm{x}_l) - \alpha_{I,l}\mathbf{\Phi}_I(\mathrm{x}_l)], \tag{12}$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}_I} = 0 \to \mathbf{w}_I = \sum_{l=1}^{L}[\alpha_{R,l}\mathbf{\Phi}_I(\mathrm{x}_l) + \alpha_{I,l}\mathbf{\Phi}_R(\mathrm{x}_l)], \tag{13}$$

$$\frac{\partial \mathcal{L}}{\partial b_R} = 0 \to \sum_{l=1}^{L}\alpha_{R,l} = 0\ , \tag{14}$$

$$\frac{\partial \mathcal{L}}{\partial b_I} = 0 \rightarrow \sum_{l=1}^{L} \alpha_{I,l} = 0\,, \tag{15}$$

$$\frac{\partial \mathcal{L}}{\partial e_{R,l}} = 0 \rightarrow \gamma_R e_{R,l} = \alpha_{R,l}\,, \tag{16}$$

$$\frac{\partial \mathcal{L}}{\partial e_{I,l}} = 0 \rightarrow \gamma_I e_{I,l} = \alpha_{I,l}\,, \tag{17}$$

$$\frac{\partial \mathcal{L}}{\partial \alpha_{R,l}} = 0 \rightarrow e_{R,l} - y_{R,l} + \left(\mathbf{w}_R^T \mathbf{\Phi}_R(\mathrm{x}_l) + \mathbf{w}_I^T \mathbf{\Phi}_I(\mathrm{x}_l) + b_R\right) = 0, \tag{18}$$

$$\frac{\partial \mathcal{L}}{\partial \alpha_{I,l}} = 0 \rightarrow e_{I,l} - y_{I,l} + \left(\mathbf{w}_I^T \mathbf{\Phi}_R(\mathrm{x}_l) - \mathbf{w}_R^T \mathbf{\Phi}_I(\mathrm{x}_l) + b_I\right) = 0, \tag{19}$$

For $l = 1, \ldots, L$.
By substituting (12)–(17) in (18) and (19):

$$\begin{cases} \frac{\alpha_{R,l}}{\gamma_R} - y_{R,l} + \sum_{i=1}^{L} \left[\alpha_{R,i} \mathbf{\Phi}_{\mathrm{R}}(\mathbf{x}_i) - \alpha_{I,i} \mathbf{\Phi}_I(\mathbf{x}_i)\right]^T \mathbf{\Phi}_R(\mathbf{x}_l) + \sum_{i=1}^{L} \left[\alpha_{R,i} \mathbf{\Phi}_{\mathrm{I}}(\mathbf{x}_i) + \alpha_{I,i} \mathbf{\Phi}_R(\mathbf{x}_i)\right]^T \mathbf{\Phi}_I(\mathbf{x}_l) + b_R = 0 \\ \frac{\alpha_{I,l}}{\gamma_I} - y_{I,l} + \sum_{i=1}^{L} \left[\alpha_{R,i} \mathbf{\Phi}_{\mathrm{I}}(\mathbf{x}_i) + \alpha_{I,i} \mathbf{\Phi}_R(\mathbf{x}_i)\right]^T \mathbf{\Phi}_R(\mathbf{x}_l) - \sum_{i=1}^{L} \left[\alpha_{R,i} \mathbf{\Phi}_{\mathrm{R}}(\mathbf{x}_i) - \alpha_{I,i} \mathbf{\Phi}_I(\mathbf{x}_i)\right]^T \mathbf{\Phi}_I(\mathbf{x}_l) + b_I = 0 \\ \sum_{l=1}^{L} \alpha_{R,l} = 0 \\ \sum_{l=1}^{L} \alpha_{I,l} = 0 \end{cases} \tag{20}$$

For $l = 1, \ldots, L$.

The above system of equations is the dual-form representation of the optimization problem in Equation (10), in which the original regression coefficients collected in the vectors $\mathbf{w}_R$ and $\mathbf{w}_I$ are replaced by the Lagrangian multipliers $\boldsymbol{\alpha}_R$ and $\boldsymbol{\alpha}_I$. It is important to remark that, although the number of unknowns in the primal space (i.e., the dimensionality of $|\mathbf{w}| = N$) is given by the number of basis functions, in the dual formulation the number of unknowns (i.e., the Lagrangian multipliers collected in the vectors $\boldsymbol{\alpha} = \boldsymbol{\alpha}_R + j\boldsymbol{\alpha}_I$) is always equal to the number of the training samples L. This means the resulting model is non-parametric, i.e., a model in which its complexity is independent from the number of both the input parameters and the basis functions.

In order to define the kernel function and the dual formulation of the complex-valued LS-SVM, the first two equations in Equation (20) can be rewritten as follows:

$$\frac{\alpha_{R,l}}{\gamma_R} - y_{R,l} + \sum_{i=1}^{L} \alpha_{R,i} \left[\mathbf{\Phi}_{\mathrm{R}}^T(\mathbf{x}_i) \mathbf{\Phi}_R(\mathbf{x}_l) + \mathbf{\Phi}_{\mathrm{I}}^T(\mathbf{x}_i) \mathbf{\Phi}_{\mathrm{I}}(\mathbf{x}_l)\right] + \sum_{i=1}^{L} \alpha_{I,i} \left[\mathbf{\Phi}_R^{\mathrm{T}}(\mathbf{x}_i) \mathbf{\Phi}_I(\mathbf{x}_l) - \mathbf{\Phi}_{\mathrm{I}}^T(\mathbf{x}_i) \mathbf{\Phi}_R(\mathbf{x}_l)\right] + b_R = 0, \tag{21}$$

$$\frac{\alpha_{I,l}}{\gamma_I} - y_{I,l} + \sum_{i=1}^{L} \alpha_{R,i} \left[\mathbf{\Phi}_{\mathrm{I}}^{\mathrm{T}}(\mathbf{x}_i) \mathbf{\Phi}_R(\mathbf{x}_l) - \mathbf{\Phi}_{\mathrm{R}}^{\mathrm{T}}(\mathbf{x}_i) \mathbf{\Phi}_I(\mathbf{x}_l)\right] + \sum_{i=1}^{L} \alpha_{I,i} \left[\mathbf{\Phi}_R^{\mathrm{T}}(\mathbf{x}_i) \mathbf{\Phi}_R(\mathbf{x}_l) + \mathbf{\Phi}_I^T(\mathbf{x}_i) \mathbf{\Phi}_I(\mathbf{x}_l)\right] + b_I = 0, \tag{22}$$

For $l = 1, \ldots, L$.
Now, let us define a complex-valued kernel $k_c\left(\mathbf{x}, \mathbf{x}^{'}\right)$:

$$\begin{gathered} \boldsymbol{k_c}\left(\mathbf{x}, \mathbf{x}^{'}\right) = \mathbf{\Phi}(\mathbf{x}), \mathbf{\Phi}\left(\mathbf{x}^{'}\right) = \mathbf{\Phi}^{\mathbf{T}}(\mathbf{x}) \cdot \mathbf{\Phi}^*\left(\mathbf{x}^{'}\right) = \left(\mathbf{\Phi}_{\boldsymbol{R}}(\mathbf{x}) + \boldsymbol{j}\mathbf{\Phi}_{\boldsymbol{I}}(\mathbf{x})\right)^{\boldsymbol{T}} \cdot \left(\mathbf{\Phi}_{\boldsymbol{R}}\left(\mathbf{x}^{'}\right) - \boldsymbol{j}\mathbf{\Phi}_{\boldsymbol{I}}\left(\mathbf{x}^{'}\right)\right) = \\ \left[\mathbf{\Phi}_{\boldsymbol{R}}^T(\mathbf{x}) \mathbf{\Phi}_{\boldsymbol{R}}\left(\mathbf{x}^{'}\right) + \mathbf{\Phi}_{\boldsymbol{I}}^T(\mathbf{x}) \mathbf{\Phi}_{\boldsymbol{I}}\left(\mathbf{x}^{'}\right)\right] + \boldsymbol{j}\left[\mathbf{\Phi}_{\boldsymbol{I}}^T(\mathbf{x}) \mathbf{\Phi}_{\mathbf{R}}\left(\mathbf{x}^{'}\right) - \mathbf{\Phi}_{\boldsymbol{R}}^T(\mathbf{x}) \mathbf{\Phi}_{\boldsymbol{I}}\left(\mathbf{x}^{'}\right)\right] = \boldsymbol{k}_{\mathbb{R}}\left(\mathbf{x}, \mathbf{x}^{'}\right) + \boldsymbol{j}\boldsymbol{k}_{\mathbb{I}}\left(\mathbf{x}, \mathbf{x}^{'}\right) \end{gathered} \tag{23}$$

Similar to the real-valued formulation, the kernel function is defined as the inner product in the complex feature space of the basis functions evaluated at $\mathbf{x}$ and $\mathbf{x}'$, where: $k_{\mathbb{R}}\left(\mathbf{x}, \mathbf{x}^{'}\right) = \mathbf{\Phi}_R^T(\mathbf{x}) \mathbf{\Phi}_R\left(\mathbf{x}^{'}\right) + \mathbf{\Phi}_{\mathrm{I}}^T(\mathbf{x}) \mathbf{\Phi}_I\left(\mathbf{x}^{'}\right)$ and $k_{\mathbb{I}}\left(\mathbf{x}, \mathbf{x}^{'}\right) = \mathbf{\Phi}_I^T(\mathbf{x}) \mathbf{\Phi}_{\mathrm{R}}\left(\mathbf{x}^{'}\right) - \mathbf{\Phi}_{\mathrm{R}}^T(\mathbf{x}) \mathbf{\Phi}_I\left(\mathbf{x}^{'}\right)$.

Using the above definition, Equations (21) and (22) can be rewritten as follows:

$$\frac{\alpha_{R,l}}{\gamma_R} - y_{R,l} + \sum_{i=1}^{L}[\alpha_{R,i}k_{\mathbb{R}}(\mathbf{x}_i,\mathbf{x}_l) - \alpha_{I,i}k_{\mathbb{I}}(\mathbf{x}_i,\mathbf{x}_l)] + b_R = 0 \tag{24}$$

$$\frac{\alpha_{I,l}}{\gamma_I} - y_{I,l} + \sum_{i=1}^{L}[\alpha_{I,i}k_{\mathbb{R}}(\mathbf{x}_i,\mathbf{x}_l) + \alpha_{R,i}k_{\mathbb{I}}(\mathbf{x}_i,\mathbf{x}_l)] + b_I = 0 \tag{25}$$

For $l = 1, \ldots, L$.

In the above equations, the regression unknowns can be computed by solving the following linear system:

$$\begin{bmatrix} \boldsymbol{K}_{RR} + \frac{\boldsymbol{I}_L}{\gamma_R} & \boldsymbol{K}_{RI} & \mathbf{1} & \mathbf{0} \\ \boldsymbol{K}_{IR} & \boldsymbol{K}_{II} + \frac{\boldsymbol{I}_L}{\gamma_I} & \mathbf{0} & \mathbf{1} \\ \mathbf{1}^T & \mathbf{0}^T & 0 & 0 \\ \mathbf{0}^T & \mathbf{1}^T & 0 & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha}_R \\ \boldsymbol{\alpha}_I \\ b_R \\ b_I \end{bmatrix} = \begin{bmatrix} \mathbf{y}_R \\ \mathbf{y}_I \\ 0 \\ 0 \end{bmatrix} \tag{26}$$

where $\boldsymbol{I}_L$ is the identity matrix of size $L \times L$, $\mathbf{1}^T = [1, \ldots, 1]^T \in \mathbb{R}^{1\times L}$, $\mathbf{0}^T = [0, \ldots, 0]^T \in \mathbb{R}^{1\times L}$, $\boldsymbol{\alpha}_R$, $\boldsymbol{\alpha}_I \in \mathbb{R}^{L\times 1}$ are vectors collecting the real and imaginary part of the regression coefficients, $b = b_R + jb_I$ is the bias term and $\boldsymbol{K}_{RR}$, $\boldsymbol{K}_{RI}, \boldsymbol{K}_{IR}$, $\boldsymbol{K}_{II} \in \mathbb{R}^{L\times L}$ are kernel matrices defined as:

$$K_{RR}^{(i,j)} = K_{II}^{(i,j)} = k_{\mathbb{R}}(\mathbf{x}_i,\mathbf{x}_j) \tag{27}$$

$$K_{IR}^{(i,j)} = -K_{RI}^{(i,j)} = k_{\mathbb{I}}(\mathbf{x}_i,\mathbf{x}_j) \tag{28}$$

for any $i, j = 1, \ldots, L$. The parameters γ_R and γ_I are the regularizer hyperparameters tuned by the user and provide a trade-off between the model flatness and its accuracy [34].

It is important to remark that the above linear system is square and, therefore, if the determinant of the square matrix is different from 0, it always yields a unique solution, which leads to the following dual space formulation for complex-valued of LS-SVM regression:

$$\mathrm{y}(\mathrm{x}) = \sum_{l=1}^{L} \alpha_l \mathrm{k}_C(\mathbf{x}_l,\mathbf{x}) + b \tag{29}$$

By substituting Equation (23) in the above equation gives:

$$\mathrm{y}(\mathbf{x}) = \sum_{l=1}^{L}[(\alpha_{R,l}\mathrm{k}_{\mathbb{R}}(\mathbf{x}_l,\mathbf{x}) - \alpha_{I,l}\mathrm{k}_{\mathbb{I}}(\mathbf{x}_l,\mathbf{x}) + b_R) + j(\alpha_{R,l}\mathrm{k}_{\mathbb{I}}(\mathbf{x}_l,\mathbf{x}) + \alpha_{I,l}\mathrm{k}_{\mathbb{R}}(\mathbf{x}_l,\mathbf{x}) + b_I)] \tag{30}$$

From the above formulation, it is clear that by means of the complex kernel k_C, the complex-valued LS-SVM regression in the dual space is able to account for possible correlation between the real and imaginary part of $\mathrm{y}(\mathbf{x})$.

4.1. Complex-Valued Kernel

There are several strategies to construct a complex kernel k_C. Within this paper, we will investigate two of them, the independent kernel, referred to as the complex valued complex function (CVCF) [19], and the pseudo kernel, referred to as the pseudo complex-valued function (PCF) [35,36]. A generic CVCF kernel can be constructed starting from a real-valued kernel $\mathbf{k}_{\mathbb{R}}$ as follows:

$$\mathrm{k}_C\left(\mathbf{x},\mathbf{x}'\right) = \mathrm{k}_{\mathbb{R}}\left(\mathbf{x}_R,\mathbf{x}'_R\right) + \mathrm{k}_{\mathbb{R}}\left(\mathbf{x}_I,\mathbf{x}'_I\right) + j\left(\mathrm{k}_{\mathbb{R}}\left(\mathbf{x}_R,\mathbf{x}'_I\right) + \mathrm{k}_{\mathbb{R}}\left(\mathbf{x}_I,\mathbf{x}'_R\right)\right), \tag{31}$$

The above complex kernel is fully compliant with the definition provided in Equation (23). The real kernel $\mathrm{k}_{\mathbb{R}}$ can be any real kernel function, e.g., linear kernel, radial basis function

(RBF) kernel and polynomial kernel. Hereafter, in this paper, for the CVCF kernel we will use $\mathrm{k}_{\mathbb{R}}$ as the RBF kernel, i.e.,

$$\mathrm{k}_{\mathbb{R}}\left(\mathbf{x}, \mathbf{x}^{'}\right) = \exp\left(-\frac{1}{2\sigma^2}\mathbf{x} - \mathbf{x}\prime)^2\right), \tag{32}$$

where σ is the kernel hyperparameter, which in this work will be tuned, along with the regularizer hyperparameters, during the model training by combining cross validation (CV) with a Bayesian optimizer [37,38].

As an alternative, a family of kernels based on the PCF can be suitably generated from the isotropic complex covariance function, such that (additional mathematical details are provided in [19,36]):

$$\mathrm{k}_{\mathrm{c}}(\mathbf{x}, \mathbf{x}\prime) = \cos\left(c\mathbf{x} - \mathbf{x}^{'}\right) k_{\mathbb{R}}\left(\mathbf{x}, \mathbf{x}^{'}\right) + \mathrm{j}\ \sin(c\mathbf{x} - \mathbf{x}\prime)\mathrm{k}_{\mathbb{R}}\left(\mathbf{x}, \mathbf{x}^{'}\right), \tag{33}$$

where $\mathrm{k}_{\mathbb{R}}(\mathbf{x}, \mathbf{x}_{\boldsymbol{k}})$ can be selected as any kernel function and c is a new hyperparameter. In this specific case, a Rational Quadratic kernel is adopted, such as $\mathrm{k}_{\mathbb{R}}(\mathbf{x}, \mathbf{x}\prime) = \sigma^2\left(1 + \frac{\mathbf{x}-\mathbf{x}\prime^2}{2al^2}\right)^{-a}$, which is a Rational Quadratic kernel in which a, l and σ are additional hyperparameters. Moreover, in this case all the hyperparameters are tuned via CV and a Bayesian optimizer [37,38].

4.2. Dual Channel Kernel (DCK) LS-SVM for Complex-Valued Data

The dual channel kernel (DCK) formulation can be seen as a special case of the general mathematical framework presented in the previous section. The underlying idea is to recast the complex variables in terms of their real and imaginary part and to work with a standard real kernel, i.e., $k_c = k_{\mathbb{R}}$: $\mathbb{R}^{d\times d} \to \mathbb{R}$.

Under the above assumption, the regression problem in Equation (26) can be simplified as follows:

$$\begin{bmatrix} \boldsymbol{K}_{RR} + \frac{\boldsymbol{I}_L}{\gamma_R} & \mathbf{0}_L & \mathbf{1} & \mathbf{0} \\ \mathbf{0_L} & \boldsymbol{K}_{RR} + \frac{\boldsymbol{I}_L}{\gamma_I} & \mathbf{0} & \mathbf{1} \\ \mathbf{1^T} & \mathbf{0^T} & 0 & 0 \\ \mathbf{0^T} & \mathbf{1^T} & 0 & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha_R} \\ \boldsymbol{\alpha_I} \\ b_R \\ b_I \end{bmatrix} = \begin{bmatrix} \mathbf{y_R} \\ \mathbf{y_I} \\ 0 \\ 0 \end{bmatrix} \tag{34}$$

where $\mathbf{0_L}$ is the $L \times L$ null matrix, $\boldsymbol{K}_{RR} \in \mathbb{R}^{L\times L}$ such that $K_{RR}^{(i,j)} = k_{\mathbb{R}}\left(\mathbf{x}_i, \mathbf{x}_j\right)$, whilst the matrices $\boldsymbol{K}_{IR} = -\boldsymbol{K}_{RI} = \mathbf{0}_L$.

It is important to remark that, in the above formulation, there is no coupling between the real and imaginary coefficients $\boldsymbol{\alpha}_R$ and $\boldsymbol{\alpha}_I$, and bias terms b_R and b_I. Indeed, the solution of the above linear system is equivalent to solving two decoupled ones, accounting for the real and imaginary parts of the regression unknowns independently, such as:

$$\begin{bmatrix} \boldsymbol{K}_{RR} + \frac{\boldsymbol{I}_L}{\gamma_R} & \mathbf{1} \\ \mathbf{1}^T & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha_R} \\ b_R \end{bmatrix} = \begin{bmatrix} \boldsymbol{y_R} \\ 0 \end{bmatrix}, \tag{35}$$

$$\begin{bmatrix} \boldsymbol{K}_{RR} + \frac{\boldsymbol{I}_L}{\gamma_I} & \mathbf{1} \\ \mathbf{1}^T & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha_I} \\ b_I \end{bmatrix} = \begin{bmatrix} \boldsymbol{y_I} \\ 0 \end{bmatrix}, \tag{36}$$

In this work, a standard RBF kernel is considered as the real kernel $k_{\mathbb{R}}\left(\mathbf{x}_i, \mathbf{x}_j\right)$. In the above scenario, the dual space formulation of the LS-SVM is written [27]:

$$\mathrm{y_R}(\mathbf{x}) = \sum_{l=1}^{L} \alpha_{R,l} k_{\mathbb{R}}(\mathbf{x}_l, \mathbf{x}) + b_R, \tag{37}$$

$$\mathrm{y_I}(\mathbf{x}) = \sum_{l=1}^{L} \alpha_{I,l} k_{\mathbb{R}}(\mathbf{x}_l, \mathbf{x}) + b_I, \tag{38}$$

It is important to note that, in the above formulation, the model for the real and imaginary parts of y are built separately, thus ignoring any possible correlation between them. Furthermore, the above models can be trained using the LS-SVM Lab toolbox for the LS-SVM available in MATLAB [39].

5. Application Examples

This section compares the accuracy and the robustness against noise of the three implementations of the complex-valued LS-SVM regression provided in Section 4 by considering two different application examples. Specifically, the proposed approaches are applied to predict the scattering parameters of a serpentine structure with three parameters and the transfer function of a high-speed link with four parameters.

5.1. Example I

As a first test case, the complex-valued LS-SVM regression applied to calculate the scattering parameters of a serpentine delay line structure is presented. Serpentine lines are widely used in printed circuit board (PCB) design to compensate time delays introduced by the trace routing. However, the frequency-domain behavior of such a structure is heavily affected by its geometrical and electrical parameters and should be carefully assessed during the design phase to avoid signal and power integrity issues and to meet design constraints [40].

The structure of the serpentine delay line considered in this example is shown in Figure 1 (inspired by [40]). The S21 scattering parameters of the above structure are investigated as a function of three parameters (i.e., $\mathbf{x} = [\varepsilon_r, LL, SW]^T$) in a frequency bandwidth from 1 MHz to 1 GHz. Table 1 shows the range of variability serpentine delay line parameters that are simulated to produce the training and test data.

(**a**) (**b**)

Figure 1. (**a**) Design parameters of the serpentine line to be analyzed; (**b**) cross-sectional view of the serpentine line.

Table 1. Serpentine delay line parameters for the training and test dataset.

Training Ranges	Test Ranges
4.5 mm $\leq LL \leq$ 5.1 mm	4.5 mm $\leq LL \leq$ 5.1 mm
$3.9 \leq \varepsilon_r \leq 4.5$	$3.9 \leq \varepsilon_r \leq 4.5$
0.13 mm $\leq SW \leq$ 0.17 mm	0.13 mm $\leq SW \leq$ 0.17 mm
1 MHz $\leq f \leq$ 3 GHz	1 MHz $\leq f \leq$ 3 GHz
1000 samples for each frequency	2000 samples for each frequency

The whole dataset consists of 3000 samples (1000 training data and 2000 test samples). The samples were generated via Latin Hypercube Sampling (LHS) by assuming a uniform variability between their maximum and minimum value. For each configuration of the geometrical parameters, the corresponding scattering parameters were computed for 5000 linearly spaced frequency sample points. The samples were generated via a set of parametric simulations with the full-wave solver available in CST.

The PCA compression is then used to compress the data in the frequency domain and to reduce the computation for the model training. Figure 2 shows the behavior normalized

singular values of the frequency points of the training dataset. The plot shows that $\overline{n} = 9$ is enough to represent the whole training set with a 0.001% threshold.

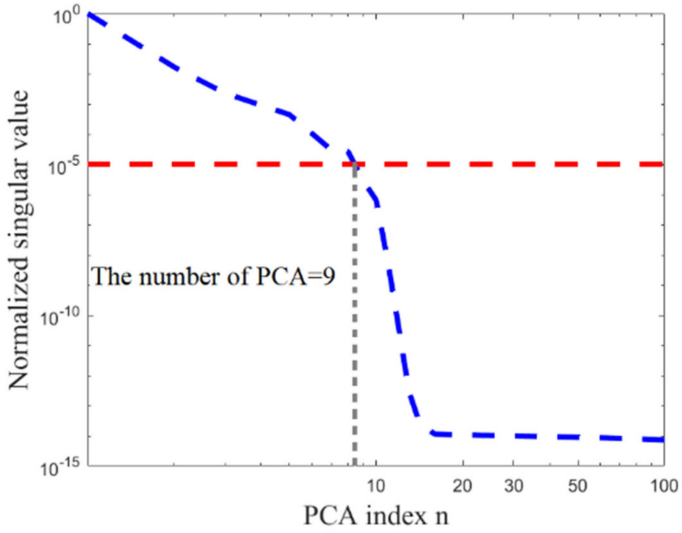

Figure 2. Normalized singular value plot of the serpentine delay line for the considered dataset with 5000 frequency points (blue line). The horizontal line shows the 0.001% threshold for the PCA truncation.

After applying the PCA, the performances of the LS-SVM regression using the PCF and the CVCF complex kernel function (see Section 4.1), and of the DCK LS-SVM regression (see Section 4.2), were assessed on the test samples for the S21 parameter. To investigate the performance of the mentioned methods, the relative Root Mean Square Error (NRMSE) is calculated as the following equation:

$$NRMSE\ \% = 100 \cdot \frac{\frac{1}{T}\sqrt{\sum_{t=1}^{T}\left(X_y - X_{\hat{y}}\right)^2}}{\frac{1}{T}\sqrt{\sum_{t=1}^{T} X_y^2}}, \tag{39}$$

where X_y can be either the real or imaginary part of the actual test samples, $X_{\hat{y}}$ is the corresponding prediction estimated via the proposed metamodels, and T is the number of test samples.

Figures 3 and 4 show the normalized error for real and imaginary parts of these three regression approaches for an increasing number of training samples (i.e., L = 50, 250, and 500).

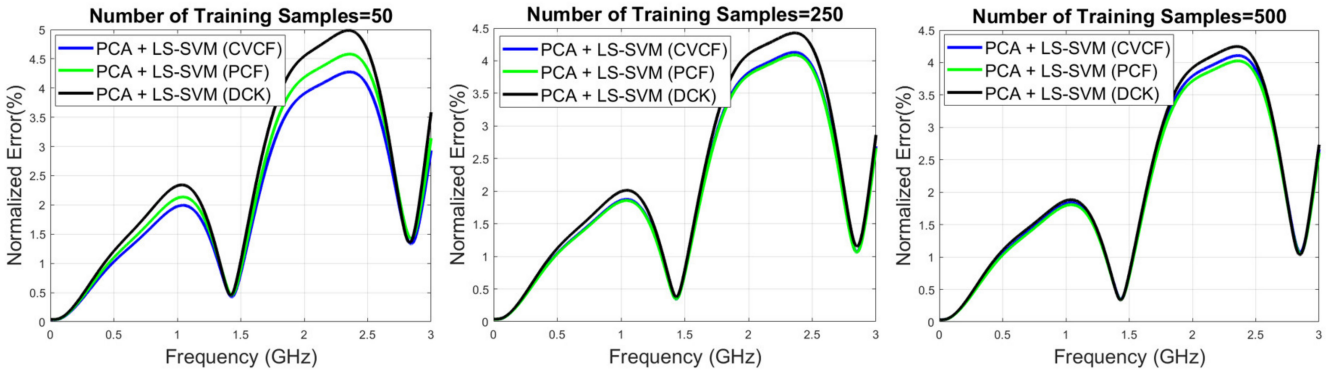

Figure 3. Comparison of the relative NRMSE values computed by the proposed approaches on the test samples by considering the real part of the S21 parameter of the serpentine delay line structure for an increasing number of training samples (i.e., L = 50, 250, 500).

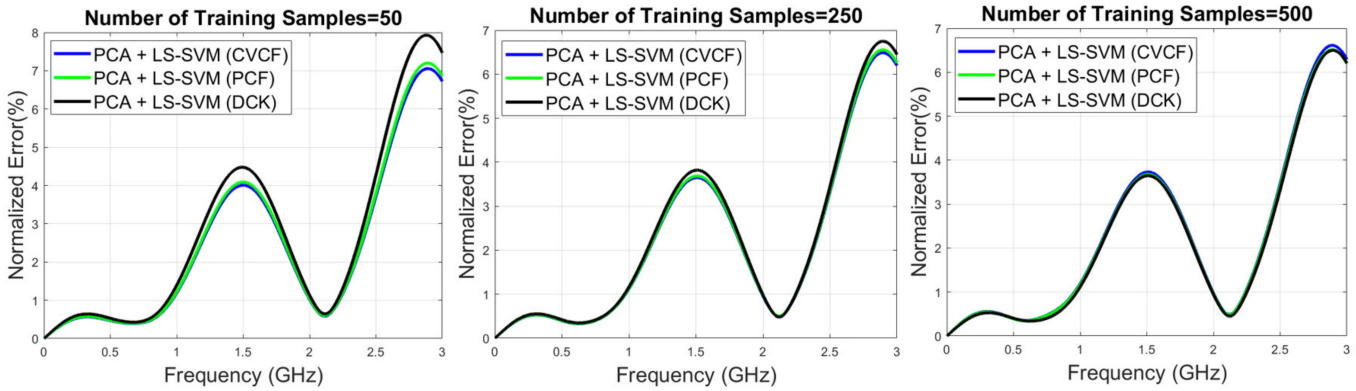

Figure 4. Comparison of the relative NRMSE values computed by the proposed approaches on the test samples by considering the imaginary part of the S21 parameter of the serpentine delay line structure for an increasing number of training samples (i.e., L = 50, 250, 500).

The plots highlight the improved accuracy of the CVCF and PCF with respect to the DCK. Indeed, with the DCK-based model, we are implicitly neglecting any kind of correlation between the real and imaginary parts of the S parameters, and this lack of complexity becomes even more evident when a low number of training samples is used to train the model.

5.2. Example II

The second application example is based on the high-speed link, as depicted in Figure 5, representing a signal distribution on a PCB. Similar to the previous example, the frequency response of the link, and thus its performance, can be greatly influenced by possible variations of its internal parameters [5].

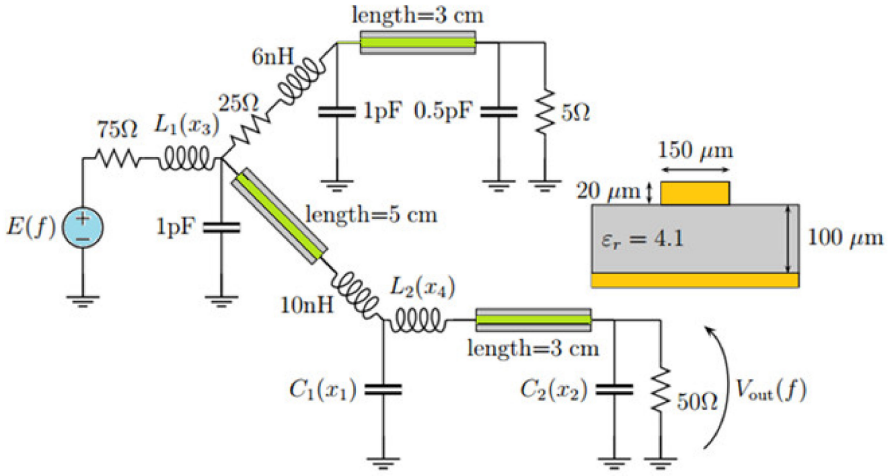

Figure 5. Schematic of the high-speed interconnect link.

Specifically, the proposed modeling approaches are here adopted to build a surrogate model for the frequency-domain behavior of the complex-valued transfer function, in which $y(\mathbf{x}; f) = \frac{Vout(f;\mathbf{x})}{E(f)}$, as a function of the values of four lumped components $C_1(x_1)$, $C_2(x_2)$, $L_1(x_3)$, and $L_2(x_4)$, is defined by four uniformly distributed random variables with a variation of $\pm 50\%$ around their central value collected in $\mathbf{x} = [x_1, x_2, x_3, x_4]^T$ (additional details are provided in Table 2).

Table 2. High-speed interconnect link parameters for the training and test datasets.

Training and Test Ranges	
C_1 (x_1)	$(1 \pm 0.5\, x_1)$ pF
C_2 (x_2)	$(0.5 \pm 0.25\, x_2)$ pF
L_1 (x_3)	$(10 \pm 5\, x_3)$ nH
L_2 (x_4)	$(10 \pm 5\, x_4)$ nH
i	1,2,3,4
x_i	Random variable in $[-1, 1]$

Four sets consisting of L = 20, 100, 150, and 500 training input configurations were generated via an LHS and used as input for a computational model implemented in MATLAB, which provides as output the corresponding transfer function evaluated at 200 frequency points. The PCA is applied to remove redundant information, leading to a compressed representation of the original dataset with only $\overline{n}$ = 10 components using a threshold of 0.01%.

The compressed training sets are then used to train three different surrogate models based on the DCK, PCF, and CVCF LS-SVM regressions. For the sake of completeness, the predictions of the above methods are compared with those provided by an additional surrogate model built via a multi-output feedforward NN structure [34,35] considering the real and imaginary parts of the considered transfer function (i.e., using the dual channel implementation). The NN is trained via the Gradient Decent backpropagation method implemented within the Neural Net Fitting Tool available in the MATLAB Deep Learning Toolbox. The network consists of three hidden layers with 50, 20, and 15 neurons, respectively. The activation function is the hyperbolic tangent sigmoid.

Figures 6 and 7 show the performance of each method on a test set consisting of 1000 samples assessed via the relative NRMSE in Equation (39) for the real and imaginary parts, respectively. The results clearly highlight the improved accuracy achieved via the PCF. Moreover, as expected, due to its simplified formulation, the DCK again provides the lowest accuracy. By comparison, NN shows the lower accuracy when a small number of training samples is used (i.e., up to 150 training samples). This low convergence with respect to the number of training samples is due to the inherent non-convex nature of the optimization problem solved during the model training [32].

Moreover, in order to stress the reliability of the proposed techniques, the training output y was corrupted with Gaussian noise, such that:

$$\mathbf{y}_{i,noisy}(\mathbf{x_i}) = \mathbf{y}_i(\mathbf{x_i}) \times (1 + \zeta_n), \tag{40}$$

where $\zeta_n \sim \mathcal{N}\left(0, \sigma_n^2\right)$ is a Gaussian random variable with standard deviation $\sigma_n = [0.01, 0.03]$.

Figures 8 and 9 compare the relative NRMSE computed at a single frequency point selected as the one providing the maximum error via the proposed approaches, and a multi-output feedforward NN for different values of the noise standard deviation σ_n and the number of training samples L for the real and imaginary parts, respectively. Among the introduced methods, CVCF shows the better performance and robustness against noise, both for real and imaginary parts.

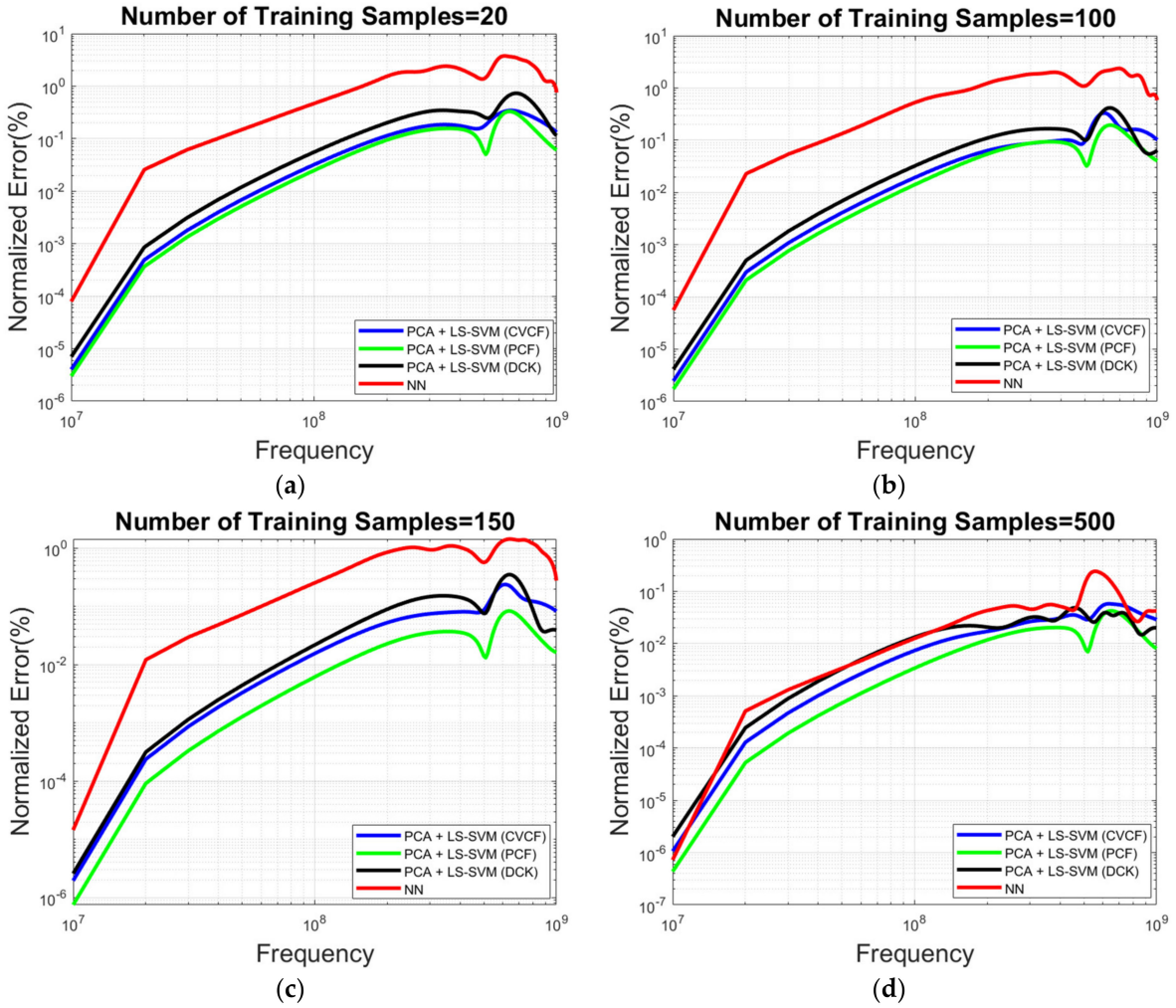

Figure 6. Comparison of the relative NRMSE values computed by the proposed approaches and a feedforward multi-output neural network on the test samples by considering the real part of the transfer function of a high-speed link for an increasing number of training samples (**a**) L = 20; (**b**) L = 100; (**c**) L = 150; (**d**) L = 500).

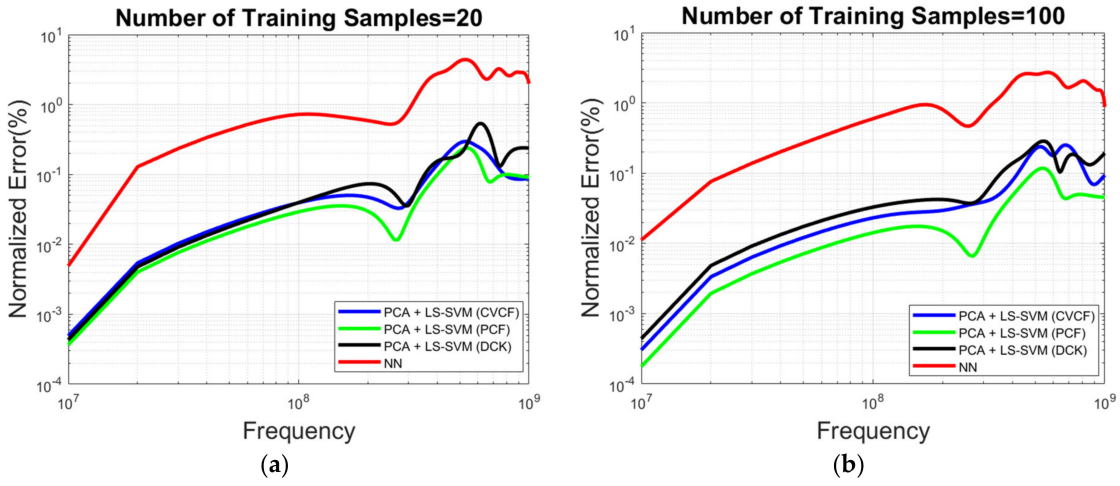

Figure 7. *Cont*.

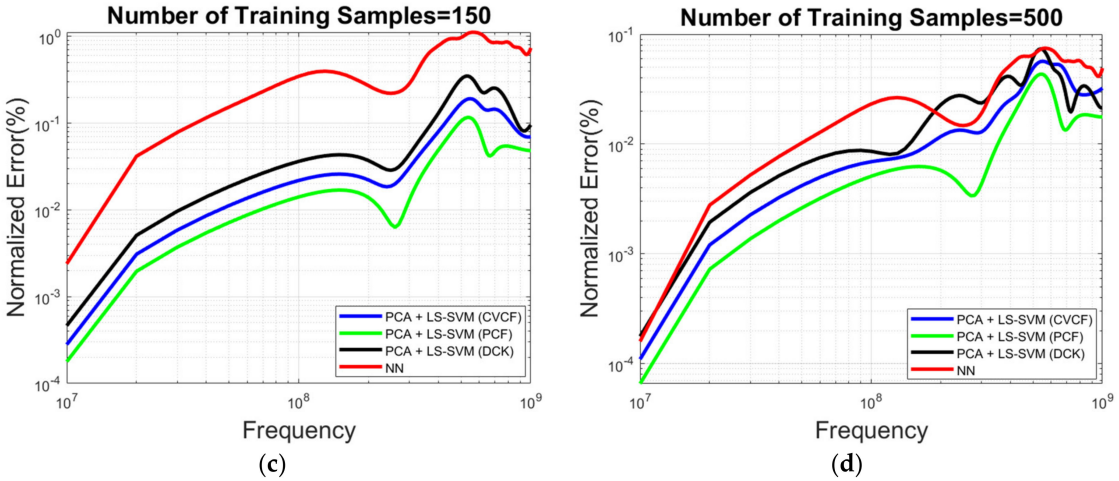

Figure 7. Comparison of the relative NRMSE values computed via the proposed approaches and a feedforward multi-output neural network on the test samples by considering the imaginary part of the transfer function of high-speed link for an increasing number of training samples. (**a**) L = 20; (**b**) L = 100; (**c**) L = 150; (**d**) L = 500).

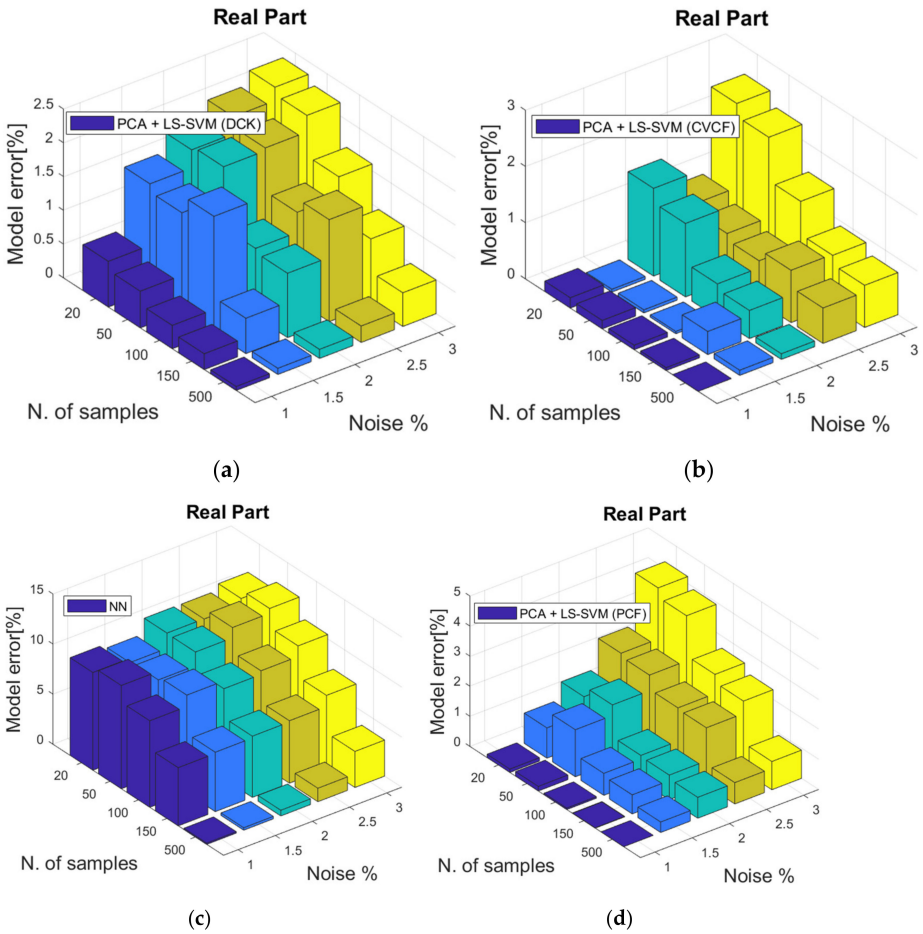

Figure 8. Three-dimensional plot of the relative NRMSE computed on the real part of the test samples at a single frequency point selected as the one providing the maximum error via the proposed approaches (see panel (**a**) for the DCK, panel (**b**) for CVCF and panel (**d**) for PCF), and a multi-output feedforward NN (see panel (**c**))for different values of the noise standard deviation σ_n and the number of training samples L.

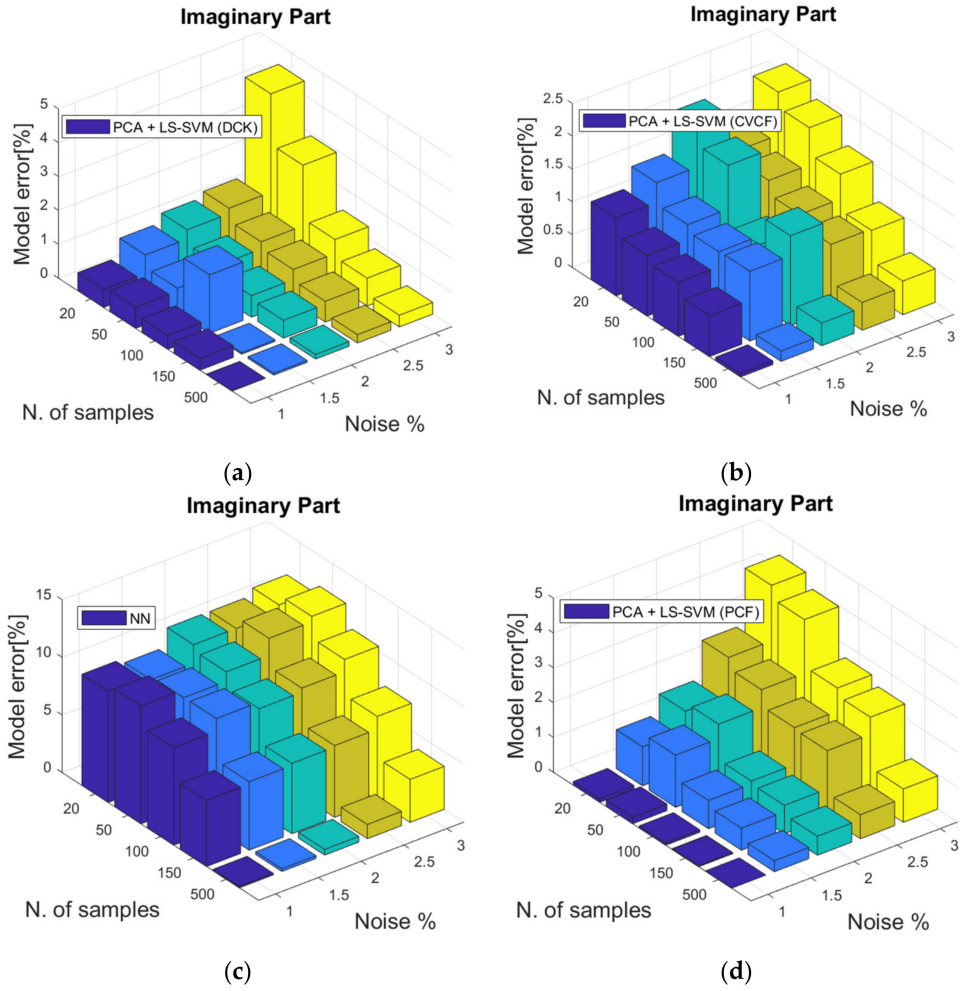

Figure 9. Three-dimensional plot of the relative NRMSE computed on the real part of the test samples at a single frequency point selected as the one providing the maximum error via the proposed approaches (see panel (**a**) for the DCK, panel (**b**) for CVCF and panel (**d**) for PCF), and a multi-output feedforward NN (see panel (**c**)) for different values of the noise standard deviation σ_n and the number of training samples L.

6. Conclusions

In this paper, the application of different methods based on LS-SVM regression for predicting the complex-valued frequency response was presented in a serpentine delay line and a high-speed link. The required data for training and examination were obtained from CST and MATLAB software, in terms of physical and geometrical parameters, including substrate dielectric constant and height and strip width of the serpentine delay line, and the capacitances and inductances of the high-speed link. PCA is shown to reduce the computational cost of the frequency samples. Then, the training data are used to create the model, and the test data are considered for the examination of the established model. The performance of the three proposed methods and of a multi-output feedforward NN are compared on the test dataset. It is shown that the CVCF method is suitable for the first complex electromagnetic problem when there are few training data points. In addition, PCF shows an acceptable performance under noise-free conditions at all frequency points for the second example. Finally, the results show that CVCF is the best method when the level of noise is increased.

Author Contributions: Conceptualization, N.S. and R.T.; methodology, N.S. and R.T.; software, data curation and validation, N.S.; writing original draft preparation, N.S.; writing review and editing, N.S. and R.T.; supervision, R.T. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Acknowledgments: The authors would like to thank Flavio Canavero, Politecnico di Torino, Italy, for his valuable and constructive suggestions during the planning and development of this manuscript.

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

References

1. Manfredi, P.; Ginste, D.V.; Stievano, I.S.; de Zutter, D.; Canavero, F.G. Stochastic transmission line analysis via polynomial chaos methods: An overview. *IEEE Electromagn. Compat. Mag.* **2017**, *6*, 77–84. [CrossRef]
2. Zhang, Z.; El-Moselhy, T.A.; Elfadel, I.M.; Daniel, L. Stochastic testing method for transistor-level uncertainty quantification based on generalized polynomial chaos. *IEEE Trans. Comput. -Aided Des. Integr. Circuits Syst.* **2013**, *32*, 1533–1545. [CrossRef]
3. Spina, D.; Ferranti, F.; Dhaene, T.; Knockaert, L.; Antonini, G.; Ginste, D.V. Variability analysis of multiport systems via polynomial-chaos expansion. *IEEE Trans. Microw. Theory Tech.* **2012**, *60*, 2329–2338. [CrossRef]
4. Ahadi, M.; Roy, S. Sparse Linear Regression (SPLINER) Approach for Efficient Multidimensional Uncertainty Quantification of High-Speed Circuits. *IEEE Trans. Comput. -Aided Des. Integr. Circuits Syst.* **2016**, *35*, 1640–1652. [CrossRef]
5. Trinchero, R.; Manfredi, P.; Stievano, I.S.; Canavero, F.G. Machine Learning for the Performance Assessment of High-Speed Links. *IEEE Trans. Electromagn. Compat.* **2018**, *60*, 1627–1634. [CrossRef]
6. Ma, H.; Li, E.; Cangellaris, A.C.; Chen, X. Support Vector Regression-Based Active Subspace (SVR-AS) Modeling of High-Speed Links for Fast and Accurate Sensitivity Analysis. *IEEE Access* **2020**, *8*, 74339–74348. [CrossRef]
7. Treviso, F.; Trinchero, R.; Canavero, F.G. Multiple delay identification in long interconnects via LS-SVM regression. *IEEE Access* **2021**, *9*, 39028–39042. [CrossRef]
8. Houret, T.; Besnier, P.; Vauchamp, S.; Pouliguen, P. Controlled Stratification Based on Kriging Surrogate Model: An Algorithm for Determining Extreme Quantiles in Electromagnetic Compatibility Risk Analysis. *IEEE Access* **2020**, *8*, 3837–3847. [CrossRef]
9. Watson, P.M.; Gupta, K.C.; Mahajan, R.L. Development of knowledge based artificial neural network models for microwave components. *IEEE Int. Microw. Symp. Baltim.* **1998**, *1*, 9–12.
10. Veluswami, A.; Nakhla, M.S.; Zhang, Q.-J. The application of neural networks to EM based simualtion and optimization of interconencts in high-speed VLSI circuits. *IEEE Trans. Microw. Theory Techn.* **1997**, *45*, 712–723. [CrossRef]
11. Kumar, R.; Narayan, S.L.; Kumar, S.; Roy, S.; Kaushik, B.K.; Achar, R.; Sharma, R. Knowledge-Based Neural Networks for Fast Design Space Exploration of Hybrid Copper-Graphene On-Chip Interconnect Networks. *IEEE Trans. Electromagn. Compat.* **2021**, 1–14, *Early Access Article*. [CrossRef]
12. Swaminathan, M.; Torun, H.M.; Yu, H.; Hejase, J.A.; Becker, W.D. Demystifying Machine Learning for Signal and Power Integrity Problems in Packaging. *IEEE Trans. Compon. Packag. Manuf. Technol.* **2020**, *10*, 1276–1295. [CrossRef]
13. Jin, J.; Zhang, C.; Feng, F.; Na, W.; Ma, J.; Zhang, Q. Deep Neural Network Technique for High-Dimensional Microwave Modeling and Applications to Parameter Extraction of Microwave Filters. *IEEE Trans. Microw. Theory Tech.* **2019**, *67*, 4140–4155. [CrossRef]
14. Moradi, M.; Sadrossadat, A.; Derhami, V. Long Short-Term Memory Neural Networks for Modeling Nonlinear Electronic Components. *IEEE Trans. Compon. Packag. Manuf. Technol.* **2021**, *1*, 840–847. [CrossRef]
15. Bourinet, J.-M. *Reliability Analysis and Optimal Design under Uncertainty—Focus on Adaptive Surrogate-Based Approaches*; Computation [stat.CO]; Université Clermont Auvergne: Clement Ferrand, France, 2018.
16. Scardapane, S.; van Vaerenbergh, S.; Hussain, A.; Uncini, A. Complex-valued neural networks with nonparametric activation functions. *IEEE Trans. Emerg. Top. Comput. Intell.* **2018**, *4*, 140–150. [CrossRef]
17. Adali, T.; Schreier, P.J.; Scharf, L.L. Complex-valued signal processing: The proper way to deal with impropriety. *IEEE Trans. Signal Processing* **2011**, *59*, 5101–5125. [CrossRef]
18. Papaioannou, A.; Zafeiriou, S. Principal Component Analysis with Complex Kernels, IEEE Transactions on Neural Networks and Learning Systems, 2013. pp. 1719–1726. Available online: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.402.3888&rep=rep1&type=pdf (accessed on 28 December 2021).
19. Boloix-Tortosa, R.; Murillo-Fuentes, J.J.; Santos, I.; Pérez-Cruz, F. Widely linear complex-valued kernel methods for regression. *IEEE Trans. Signal Processing* **2017**, *65*, 5240–5248. [CrossRef]
20. Tobar, F.A.; Kuh, A.; Mandic, D.P. A novel augmented complex valued kernel LMS. In Proceedings of the 2012 IEEE 7th Sensor Array and Multichannel Signal Processing Workshop (SAM), Hoboken, NJ, USA, 17–20 June 2012; pp. 473–476.
21. Boloix-Tortosa, R.; Murillo-Fuentes, J.J.; Tsaftaris, S.A. The Generalized Complex Kernel Least-Mean-Square Algorithm. *IEEE Trans. Signal Processing* **2019**, *67*, 5213–5222. [CrossRef]
22. Hirose, A. *Complex-Valued Neural Networks*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2013.
23. Bouboulis, P.; Theodoridis, S.; Mavroforakis, C.; Evaggelatou-Dalla, L. Complex Support Vector Machines for Regression and Quaternary Classification. *IEEE Trans. Neural Netw. Learn. Syst.* **2015**, *26*, 1260–1274. [CrossRef]

24. Scardapane, S.; van Vaerenbergh, S.; Comminiello, D.; Uncini, A. Widely Linear Kernels for Complex-Valued Kernel Activation Functions. In Proceedings of the ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brighton, UK, 12–17 May 2019; pp. 8528–8532.
25. Ogunfunmi, T.; Paul, T.K. On the complex kernel-based adaptive filter. In Proceedings of the 2011 IEEE International Symposium of Circuits and Systems (ISCAS), Rio de Janeiro, Brazil, 15–18 May 2011; pp. 1263–1266.
26. Boloix-Tortosa, R.; Murillo-Fuentes, J.J.; Payán-Somet, F.J.; Pérez-Cruz, F. Complex Gaussian processes for regression. *IEEE Trans. Neural Netw. Learn. Syst.* **2018**, *29*, 5499–5511. [CrossRef]
27. Soleimani, N.; Trinchero, R.; Canavero, F. Application of Different Learning Methods for the Modelling of Microstrip Characteristics. In Proceedings of the 2020 IEEE Electrical Design of Advanced Packaging and Systems (EDAPS), Shenzhen, China, 14–16 December 2020; pp. 1–3.
28. Manfredi, P.; Grivet-Talocia, S. Compressed Stochastic Macromodeling of Electrical Systems via Rational Polynomial Chaos and Principal Component Analysis. In Proceedings of the 2021 Asia-Pacific International Symposium on Electromagnetic Compatibility (APEMC), Nusa Dua-Bali, Indonesia, 27–30 September 2021.
29. Ahmadi, M.; Sharifi, A.; Fard, M.J.; Soleimani, N. Detection of brain lesion location in MRI images using 326 convolutional neural network and robust PCA. *Int. J. Neurosci.* **2021**, *131*, 1–12.
30. Jolliffe, I.T. *Principal Component Analysis*; Springer: New York, NY, USA, 2002.
31. Manfredi, P.; Trinchero, R. A data compression strategy for the efficient uncertainty quantification of time-domain circuit responses. *IEEE Access* **2020**, *8*, 92019–92027. [CrossRef]
32. Kushwaha, S.; Attar, A.; Trinchero, R.; Canavero, F.; Sharma, R.; Roy, S. Fast Extraction of Per-Unit-Length Parameters of Hybrid Copper-Graphene Interconnects via Generalized Knowledge Based Machine Learning. In Proceedings of the IEEE 30th Conference on Electrical Performance of Electronic Packaging and Systems (EPEPS), Austin, TX, USA, 17–20 October 2021.
33. Suykens, J.A.K.; van Gestel, T.; de Brabanter, J.; de Moor, B.; Vandewalle, J. *Least Squares Support Vector Machines*; World Scientific Publishing Company: Singapore, 2002.
34. Vapnik, V. *The Nature of Statistical Learning Theory*, 2nd ed.; Springer: New York, NY, USA, 1999.
35. Posa, D. Parametric families for complex valued covariance functions: Some results, an overview and critical aspects. *Spat. Stat.* **2020**, *39*, 100473. [CrossRef]
36. Iaco, D.S.; Palma, M.; Posa, D. Covariance functions and models for complex-valued random fields. *Stoch. Environ. Res. Risk Assess.* **2003**, *17*, 145–156. [CrossRef]
37. Snoek, J.; Larochelle, H.; Adams, R.P. Practical bayesian optimization of machine learning algorithms. *Adv. Neural Inf. Processing Syst.* **2012**, *25*, 1–10.
38. Geng, J.; Gan, W.; Xu, J.; Yang, R.; Wang, S. Support vector machine regression (SVR)-based nonlinear modeling of 354 radiometric transforming relation for the coarse-resolution data-referenced relative radiometric normalization (RRN). *Geo-Spat. Inf. Sci.* **2020**, *23*, 237–247. [CrossRef]
39. *LS-SVMlab*, Version 1.8 ed; Department of Electrical Engineering (ESAT), Katholieke Universiteit Leuven: Leuven, Belgium, 6 July 2011; Available online: https://www.esat.kuleuven.be/sista/lssvmlab/old/toolbox.html (accessed on 28 December 2021).
40. Soh, W.-S.; See, K.-Y.; Chang, W.-Y.; Oswal, M.; Wang, L.-B. Comprehensive analysis of serpentine line design. In Proceedings of the 2009 Asia Pacific Microwave Conference, Singapore, 7–10 December 2009; pp. 1285–1288.