Machine Learning Regression Techniques for the Modeling of Complex Systems: An Overview

Machine Learning Regression Techniques for the Modeling of Complex Systems: an Overview

Riccardo Trinchero and Flavio Canavero

Dept of Electronics and Telecommunications, Politecnico di Torino, C.so Duca degli Abruzzi 24, 10129 Torino, Italy

Abstract—Recently, machine learning (ML) techniques have gained widespread diffusion, since they have been successfully applied in several research fields. This paper investigates the effectiveness of advanced ML regressions in two EMC applications. Specifically, support vector machine, least-squares support vector machine and Gaussian process regressions are adopted to construct accurate and fast-to-evaluate surrogate models able to predict the output variable of interest as a function of the system parameters. The resulting surrogates, built from a limited set of training samples, can be suitably adopted for both uncertainty quantification and optimization purposes. The accuracy and the key features of each of the considered machine learning techniques are investigated by comparing their predictions with the ones provided by either circuital simulations or measurements.

Index Terms—Surrogate model, uncertainty quantification, Machine Learning, Support Vector Machine, Least-Square Support Vector Machine, Gaussian Process regression.

I. Introduction

Understanding the link between the parameters and the responses of complex electronic systems and devices is a key aspect during the design phase. A deep knowledge of the system functioning can be used, along with optimization and uncertainty quantification (UQ) tools, to optimize the product performance, to meet the design constraints and to assess the product reliability [1,2].

Unfortunately, for realistic applications, the relationship between the parameters and the outputs of the system is rather complicated and usually not explicit, so it must be estimated via either physical (measurements) or computer experiments (simulations). Physical experiments can be expensive and time consuming, since they required the construction of several prototypes. As computing power increases, it has become possible to model the actual behavior of electronic circuits via sophisticated computer codes. Hence, computer experiments are now heavily adopted during the design phase [3].

Simulation experiments rely on the so-called *computational model*. The computational model must provide an accurate synthetic description of the actual behavior of the system under modeling, able to virtually compute, without the need of expensive prototypes, a prediction of the outputs of interest for any configuration of the system parameters. Such model can have different levels of fidelity going from a simple closed-form analytical solution (usually available for simple devices) to the more sophisticated cases of physical-based models (e.g., the ones based on 3D full-wave solvers). It goes without saying that the model complexity heavily impacts its computational cost. A detailed physical-based computational model can be complicated to be managed and analyzed, thus making the design process lengthy [2]-[6]. This is due to the fact that, UQ and optimization tasks usually require to run a relatively large number of simulations, making the overall computational cost unaffordable. As an example, if a single simulation with the computational model requires 1h, the computational cost of a Monte Carlo (MC) simulation with 10k samples will be 1 month!!!

Surrogate models, also known as *metamodels*, can be seen as an effective solution able to reduce the computational cost of the full models [3]. A surrogate model is "a model of a model", since it provides a closed-form and fast-to-evaluate approximation of the computational model. Such model is constructed from the data generated from a limited set of simulations with the expensive computational model by means of regression or interpolation techniques (see the illustration in Fig. 1). The dataset used for the training of the surrogate model is referred to as *training set.* Usually, the samples of the training set are carefully selected in order to explore the space of the input parameters as much as possible [3],[7]. The resulting surrogate model can be suitably employed by statistical simulations or optimization schemes to provide an efficient alternative to the expensive computational model [3]-[5].

Obviously, the accuracy of the predictions provided by the surrogate model strongly depends on the adopted regression or interpolation technique. Without loss of generality, the "best" or "ideal" technique for constructing the abovementioned surrogate models should be able to [3]:

- learn complex non-linear input-output relationship;
- handle a large number of input variables with large parameter variations;
- converge fast in terms of accuracy w.r.t. the number of training samples and provide

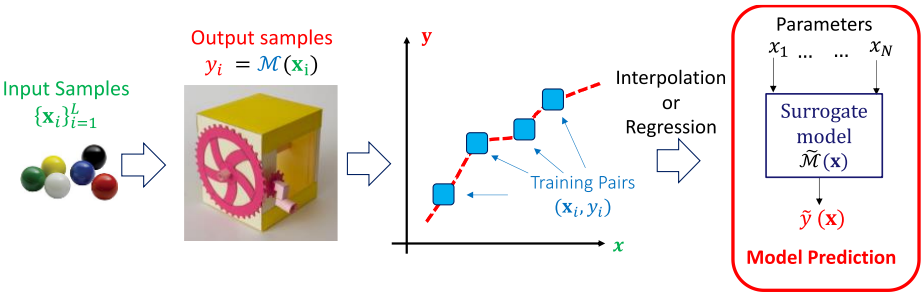

Fig. 1 – Illustration of the training process leading to a surrogate model.

information on the level of accuracy.
Unfortunately, no methodology exists complying with all the aforementioned features of our "ideal" modeling approach, since each technique has its own advantages and drawbacks.

In general, the structure of the surrogate models under consideration is a combination of basis functions, that can lead to two different categories. For the first category, we will refer to *parametric models*, if the number of parameters to be estimated via the regression is fixed by the number of basis functions. Members of this parametric category are the well-known ordinary least squares (OLS) regression and the Polynomial Chaos Expansion (PCE) [8]. OLS is conceptually very simple and intuitive and will be illustrated and compared with more sophisticated ML methods in Sect. II. The underlying idea of PCE, instead, is to represent the non-linear relationship between the system output and the stochastic parameter as a linear combination of orthonormal polynomial basis with suitable statistical properties [9]. For this reason, PCE is considered as reference technique for the UQ in different fields, including EMC applications [9]-[12]. However, for a polynomial expansion, the number of basis functions grows exponentially with the number of input parameter and the expansion order, leading to the infamous curse-of-dimensionality. For the above reasons, advanced sparse implementations of the PCE have been proposed [12]-[14]. Such implementations are rather complex, but when a low order expansion can be used, they allow to heavily reduce the impact of the curse of dimensionality and to deal with thousands of unknowns.

For the second category, most of ML regressions rely on a dual space formulation, which allows constructing a *non-parametric* surrogate model. For a non-parametric model the number of regression parameters to be estimated during the model training is independent from the number of basis functions and from the dimensionality of the input space [15]. Such feature sets the surrogate model free of the detrimental effects of the curse-of-dimensionality. Machine learning-based regressions (e.g., (Convolutional) Neural Networks [16], Support Vector Machine (SVM) regression [17],[18], Least Squares Support Vector Machine (LS-SVM) regression [19] and Gaussian Process Regression [20]) provide the users with a set of flexible and powerful alternatives to other state-of-the-art techniques for the construction of surrogate models with tunable complexity in several engineering fields, as well as for EMC applications [21]-[29]. Moreover, probabilistic ML regression techniques such as the Gaussian Process Regression (GPR) [20], can be adopted to train a probabilistic surrogate model able not only to predict the output of interest for any configuration of the input parameters, but also to provide by itself statistical information on the reliability of its predictions (e.g., in terms of confidence intervals (CI)) [20].

From the above discussion, ML approaches turn out to be promising candidates for metamodel construction, thus providing advanced tools for the UQ and the optimization of complex electronic systems. Without loss of generality, this paper investigates the performances of the SVM, LS-SVM regression and the GPR, comparing them with the well-known OLS regression with the help of a 1-D illustrative example. Finally, the proposed approaches are applied to the UQ of the spectrum envelop of a switching DC-DC buck converter as a function of 17 uncertain parameters [30].

II. Regression Techniques and Surrogate Modeling

This Section presents a quick overview of the mathematical background of the OLS, SVM, LS-SVM and GP regression techniques with specific emphasis on their application to surrogate model construction. Specifically, with reference to Fig. 1, we will address the problem of building a surrogate model $\widetilde{\mathcal{M}}$, starting from a set of training pairs $D = \{(\mathbf{x}_i, y_i)\}_{i=1}^{L}$ provided by the output of a computational model $y_i = \mathcal{M}(\mathbf{x}_i)$ as a function of the system parameters $\mathbf{x}_i \in \mathcal{X} \subseteq \mathbb{R}^d$. The obtained surrogate $\widetilde{\mathcal{M}}$ should allow to accurately predict the output y for any configuration of the input parameter $\mathbf{x} \in \mathcal{X}$, which has not been used during the training, a.k.a., the *test samples*. The extension to the multioutput formulation is available in [14],[24].

Ordinary Least Squares Regression

OLS regression undoubtedly represents the simplest and most common way to construct a metamodel. Starting from a set of training samples D, the OLS regression allows to train a generic metamodel in terms of the following linear expansion of basis functions [4],[31]:

$$\widetilde{\mathcal{M}}_{OLS}(\mathbf{x}; \mathbf{w}) = \sum_{i=1}^{n} w_i \, \phi_i(\mathbf{x}) = \langle \mathbf{w}, \boldsymbol{\phi}(\mathbf{x}) \rangle, \qquad (1)$$

where $\boldsymbol{\phi} = [\phi_1, \ldots, \phi_n]^T$ is a vector collecting the basis functions and $\mathbf{w} = [w_1, \ldots, w_n]^T$ is a vector of the regression coefficients [4].
The OLS regression looks for the best set of coefficients $\mathbf{w}^*$ by minimizing the following optimization problem, usually referred to as *empirical risk minimization*:

$$\mathbf{w}^* = \underset{\mathbf{w}}{\operatorname{argmin}} \frac{1}{L}\sum_{i=1}^{L}(y_i - \langle \mathbf{w}, \boldsymbol{\phi}(\mathbf{x}_i)\rangle)^2 . \quad (2)$$

In the above optimization, we estimate the best set of regression coefficients $\mathbf{w}^*$ by minimizing the squared of the model error computed on the training set. The solution of (2) leads to the well-known closed-form solution of the OLS regression based on the pseudoinverse matrix [4]:

$$\mathbf{w}^* = (\boldsymbol{\Phi}^T\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^T\mathbf{y} \quad (3)$$

where $\boldsymbol{\Phi}$ is a matrix collecting the basis functions evaluated on the training inputs (i.e., $\Phi_{ij} = \phi_j(\mathbf{x}_i)$) and $\mathbf{y} = [y_1, \ldots, y_L]^T$ is a vector collecting the training outputs.
Such regression approach has some limitations. First of all, the squared loss function unavoidably leads to a regression with low bias, but with high variance [32]. This means that the resulting regression is able to follow the training samples (i.e., the error on the training samples can be zero), but it might not generalize "well" on the test samples. This phenomenon is called *overfitting*. Moreover, the model constructed via the OLS regression is a parametric model, in which the number of regression coefficients (i.e., the dimensionality of the vector $\mathbf{w}$) is fixed by the number of basis functions (i.e., the dimensionality of $\boldsymbol{\phi}$).

Least-Squares Support Vector Machine Regression

The LS-SVM regression can be seen as an extension of the plain OLS regression, since it allows to construct both parametric and non-parametric models. Similar to the OLS, the *primal space* formulation of the LS-SVM regression can be written as [19]:

$$\widetilde{\mathcal{M}}_{LS-SVM}(\mathbf{x}; \mathbf{w}, b) = \langle \mathbf{w}, \boldsymbol{\phi}(\mathbf{x})\rangle + b, \quad (4)$$

where again the vectors $\boldsymbol{\phi}$ and $\mathbf{w}$ collect the basis functions and the regression coefficients, respectively, and b is a scalar bias term.
The LS-SVM regression optimizes the vector $\mathbf{w}$ and the bias b in (4), via the solution of the following optimization problem [20]:

$$\min_{\mathbf{w},b,\mathbf{e}} \frac{1}{2}\|\mathbf{w}\|^2 + \frac{\gamma}{2}\sum_{i=1}^{L} e_i^2$$
$$\text{subject to } e_i = y_i - (\langle \mathbf{w}, \boldsymbol{\phi}(\mathbf{x}_i)\rangle + b). \quad (5)$$

Different from the OLS, in the LS-SVM regression, the unknowns (i.e., $\mathbf{w}$ and b) are estimated by minimizing at the same time both the squared of the model error e_i on the training samples (see Fig. 2(a)), also known as squared error function, and the L2 norm of the regression coefficients collected in the vector $\mathbf{w}$. The latter term is the so-called Tikhonov regularizer. It is used within the optimization problem in (5) to penalize the accuracy of the model on the training samples (i.e., we are increasing the model bias). In this way we reduce the model variance, thus preventing overfitting [32]. The parameter γ, usually referred to as hyperparameter, helps providing a trade-off between the model bias and its variance. Such hyperparameter can be tuned either "manually" by the user or automatically via cross-validation (CV) algorithms (for those interested, please refer to [33]). The primal space formulation of the LS-SVM regression turns out to be equivalent to the Ridge regression [30].

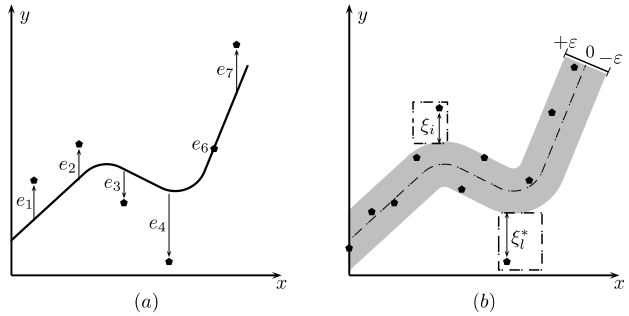

Fig. 2 – Graphical interpretation of the LS-SVM (panel (a)) and of the SVM (panel (b)) regression.

The optimization problem in (5) can be rewritten into its equivalent *dual* problem formulation. Such dual interpretation, along with the "kernel trick" [18],[19],[34], leads to the following dual space formulation of the LS-SVM regression:

$$\widetilde{\mathcal{M}}_{LS-SVM}(\mathbf{x}; \boldsymbol{\alpha}, b) = \sum_{i=1}^{L}\alpha_i K(\mathbf{x}_i, \mathbf{x}) + b , \quad (6)$$

where α_i are the dual parameters collected in the vectors $\boldsymbol{\alpha}$ and K is the kernel function defined as $K(\mathbf{x}, \mathbf{x}') = \langle \boldsymbol{\phi}(\mathbf{x}), \boldsymbol{\phi}(\mathbf{x}')\rangle$. It is important to remark, that the dual space formulation in (6) provides the users with a non-parametric regression, in which the number of coefficients α_i to be estimated during the model training turns out to be completely independent from both the number of basis functions (i.e., the dimensionality of the feature space) and the number of input parameters, but only depends on the number of training samples (i.e., L). Several kernel functions with different properties can be adopted (additional mathematical details are available in [18],[20], [34]). The most common ones are:

- Linear: $k(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T\mathbf{x}'$
- Polynomial of order q: $k(\mathbf{x}, \mathbf{x}') = (1 + \mathbf{x}^T\mathbf{x}')^q$
- Radial basis function (RBF): $k(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{2\sigma^2}\right)$, where σ is a kernel hyperparameter to be estimated during the model training (e.g., via CV).

For a given value of the regression hyperparameters, the

unknowns α_i and b can be suitably estimated by solving a linear system [19]. The LS-SVM regression is already available in MATLAB via the LS-SVMLab Toolbox [35].

Support Vector Machine Regression

The SVM is a well-consolidated technique for both classification and regression purposes. Similar to the LS-SVM, the primal space formulation of the SVM regression can be written in terms of the following linear model [17],[18]:

$$\widetilde{\mathcal{M}}_{SVM}(x;\mathbf{w},\mathrm{b}) = \langle \mathbf{w}, \boldsymbol{\phi}(\mathbf{x}) \rangle + b. \tag{7}$$

Even if the primal space formulation in (7) looks identical to the one of the LS-SVM regression, the "optimal" configuration of the vector $\mathbf{w}$ and the bias b in (7) is estimated via the solution of a completely different optimization problem [18]:

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{L} |y_i - (\langle \mathbf{w}, \boldsymbol{\phi}(\mathbf{x}_i) \rangle + b)|_{\varepsilon}, \tag{8}$$

where the Tikhonov regularization is yet applied and C is a free parameter, playing the same role of γ in the SVM regression. However, different from the LS-SVM, the above optimization adopts the so-called linear ε-insensitive loss function $|\ \ |_{\varepsilon}$, which writes [18]:

$$\left|y_i - \widetilde{\mathcal{M}}_{SVM}(\mathbf{x}_i)\right|_{\varepsilon} = \begin{cases} 0, & if\ \left|y_i - \widetilde{M}_{SVM}(\mathbf{x}_i)\right| \le \varepsilon \\ \left|y_i - \widetilde{\mathcal{M}}_{SVM}(\mathbf{x}_i)\right| - \varepsilon, & \text{otherwise.} \end{cases} \tag{9}$$

According to above loss function, if the absolute value of the model error is less than ε, the loss function does not add any penalization to the empirical risk functional. On the other hand, when the model error is larger than ε, the linear ε-insensitive loss function adds a penalization equal to the excess model $\ell 1$-error w.r.t. ε (see the terms ξ_i and ξ_i^* in Fig. 2(b)). The region $[-\varepsilon, +\varepsilon]$ is called ε-insensitive zone. The parameter ε can be seen as the regression tolerance, thus making such kind of regression extremely useful when we need to deal with noisy samples. It is important to remark that the tolerance ε, as well as the hyperparameter C, must be tuned by the user, as an example via CV [33].
Like the LS-SVM, also the SVM regression admits a dual formulation in the following form:

$$\widetilde{\mathcal{M}}_{SVM}(\mathbf{x};\boldsymbol{\beta},\mathrm{b}) = \sum_{i=1}^{L} \beta_i K(\mathbf{x}_i,\mathbf{x}) + b\,, \tag{10}$$

where β_i are the dual coefficients collected in the vector $\boldsymbol{\beta}$ with a different origin that the α_i of (6) and K is the kernel function previously defined for the LS-SVM. Similar to the case of the LS-SVM regression, the above dual formulation provides a non-parametric model. Moreover, it is important to point out that the optimization problem resulting from the dual problem formulation of the SVM regression cannot be solved in a closed-form and requires a numerical solution. The SVM regression algorithm is already available in MATLAB [36].

Gaussian Process Regression (GPR)

Gaussian process regression (GPR), also known as Kriging model, represents an interesting alternative to the deterministic regressions presented so far. Starting from a set of training samples D, the GPR allows building a *probabilistic* metamodel, which not only estimates the model output for any configuration of the input parameters, but it provides as output a Gaussian distribution. Such distribution can be used to provide useful statistical information on the reliability of the model predictions, without the need to run an equivalent simulation with the computational model [20]. Under the assumption that the computational model $y = \mathcal{M}(\mathbf{x})$ follows a Gaussian Process (GP) prior (i.e., the function $y = \mathcal{M}(\mathbf{x})$ has been drawn from a GP prior), the model obtained by the GPR writes:

$$y \sim \widetilde{M}_{GPR}(\mathbf{x}) = GP\big(m(\mathbf{x}), k(\mathbf{x},\mathbf{x}')\big), \tag{11}$$

where $m(\mathbf{x})$ and $k(\mathbf{x},\mathbf{x}')$ are the trend function and the covariance function (or kernel) of the GP, respectively. It is important to remark that the covariance function $k(.,.)$ specified in the GP prior, has the same mathematical properties of kernel $K(.,.)$ used in the dual space formulation of the SVM and LS-SVM regression [20], [37], [38]. Indeed, it is used to explain the correlation between each pair of points in the input space and characterizes the functions that can be described by the GP [38].
A GP process can be considered as an extension of the concept of Gaussian distribution from random numbers to random functions [20]. The trend $m(\mathbf{x})$ provides the average function among the ones drawn from the GP prior, while the covariance provides the correlation between the values of such functions at different point (i.e., $\mathbf{x}$ and $\mathbf{x}'$) in the

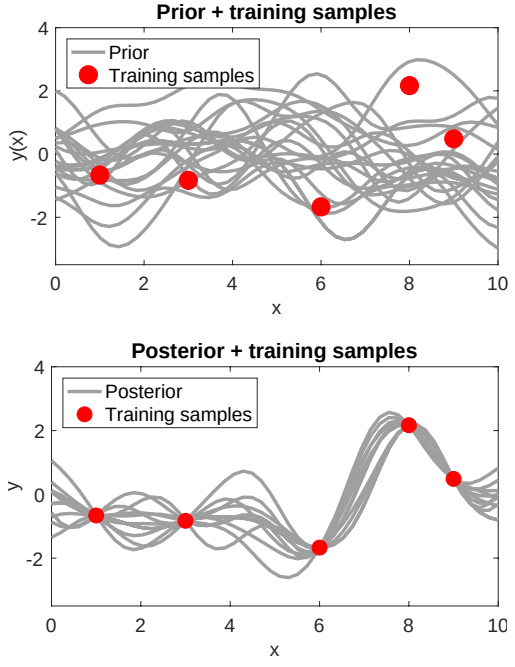

Fig. 3 – Illustration of 10 functions drawn from the prior (panel (a)) and of 10 functions drawn from the posterior (panel(b)).

parameters space. The prior mean $m(\mathbf{x})$ can be any deterministic function (e.g., a metamodel obtained via the any deterministic regression technique) [30]. Several covariance functions are available [20],[37], e.g., for the case of smooth functions the most common is the squared exponential function, i.e.,

$$k(\mathbf{x},\mathbf{x}') = \sigma_f^2 \exp\left(-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{2\sigma_l^2}\right). \quad (12)$$

The prior GP in (11) fixes the properties of the GPR model without using the information provided by the training samples. The information about the training set D can be included within the GPR by restricting the infinite set of functions drawn from the prior to the ones which agree with the output training samples, leading to the so-called posterior distribution. Thanks to the properties of the GP, such operation can be done analytically and corresponds to conditioning the Gaussian prior distribution on the observations [20],[37],[38]. Figure 3 provides a graphical interpretation of the above process.

Given the training set D, the posterior distribution allows to predict the probability of the output variable y for a generic test configuration of the input parameters $\mathbf{x}_*$ in terms of the following Gaussian distribution:

$$\mathrm{y}_* \sim \mathrm{p}(\mathrm{y}_*|\mathbf{x}_*, \mathrm{D}) = N\left(\mu_{\mathbf{x}_*}, \sigma_{\mathbf{x}_*}^2\right), \quad (13)$$

in which $\mu_{\mathbf{x}_*}$ and $\sigma_{\mathbf{x}_*}^2$ are the posterior mean and variance defined as [20],[38]:

$$\mu_{\mathbf{x}_*} = m(\mathbf{x}_*) + \mathbf{k}_*^T(\mathbf{K}+\sigma_n^2\mathbf{I})^{-1}\mathbf{y} \quad (14a)$$
$$\sigma_{\mathbf{x}_*}^2 = k_{**} - \mathbf{k}_*(\mathbf{K}+\sigma_n^2\mathbf{I})^{-1}\mathbf{k}_*^T \quad (14b)$$

where, $\mathbf{y} = [y_1, \dots, y_L]^T$, $\mathbf{K} \in \mathbb{R}^{L\times L}$ is the correlation matrix evaluated on the input samples such as the entries $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$, $\mathbf{k}_* = [k(\mathbf{x}_*, \mathbf{x}_1), \dots, k(\mathbf{x}_*, \mathbf{x}_L)] \in \mathbb{R}^{1\times \mathrm{L}}$ is a vector, $k_{**} = k(\mathbf{x}_*, \mathbf{x}_*)$ is a scalar and σ_n^2 is an hyperparameter representing the variance of a possible additive Gaussian noise corrupting the training set. The latter is set to 0 for noiseless cases. The above formulation can be extended to account for the correlation among a set of n_t test samples (i.e., $\mathbf{x}_*$ is a $(n_t \times d)$ matrix). In such a case, the posterior mean and variance in (14) become the posterior mean vector and the posterior covariance matrix, respectively (see [20],[38] for additional details).
The probabilistic interpretation in (2) allows computing for any configuration of the input parameters $\mathbf{x}_*$ the CI, such that:

$$y_* \in \left[\mu_{\mathbf{x}_*} - z_{1-\frac{\alpha}{2}}\sigma_{\mathbf{x}_*}, \mu_{\mathbf{x}_*} + z_{1-\frac{\alpha}{2}}\sigma_{\mathbf{x}_*}\right], \quad (15)$$

with a probability of $100(1-\alpha)\%$, where z denotes the $1-\alpha/2$ quantile of a standard Gaussian distribution [20].

Summary

Summarizing, all the presented regressions techniques have their own advantages and drawbacks. The OLS regression provides the most straightforward approach for the surrogate model construction, since the surrogate model can be trained directly via the solution of a linear system, without requiring the tuning of regression hyperparameters. However, the OLS regression suffers from the curse of dimensionality and overfitting. The dual formulation of the LS-SVM and SVM regressions allows mitigating the above issues, but the improvements w.r.t. to the OLS regression do not come for free. Indeed, due to the tuning of the regression hyperparameters, the training phase for the generation of LS-SVM or SVM surrogate models is computationally more demanding than the one required by the OLS regression. A similar conclusion can be drawn also for the probabilistic model provided by the GPR. The non-parametric model trained with the GPR is able to provide useful statistical information about the reliability of its prediction, but again the complexity of the optimization problem solved during the model training is higher compared to a standard OLS regression.

III.Application Examples

In this Section, the performances of the ML regressions presented in the previous Section and of state-of-the-art regression techniques, such as OLS and sparse PCE are investigated on two applications consisting of a real dataset for the wet human skin permittivity [39] and the UQ of the conducted emission (CE) generated by a buck converter with 17 uncertain parameters.

Skin permittivity Dataset

As a first example, the OLS, the LS-SVM, the SVM and the GP regressions are applied to construct a set of surrogate models able to approximate an experimental dataset in [39] collecting the measurements of the wet human skin permittivity as a function of frequency. Such dataset with a single and deterministic parameter (d=1), i.e., the frequency, has been selected for graphical reason, since it will allow to better illustrate the main features of each of the considered methods. However, it is important to remark that despite its simplicity, the problem at hand is particularly relevant for the EMC field, since its study identifies the conditions that maximize the power density transmitted into the skin, according to the most recent exposure guidelines issued by the ICNIRP [40] and the IEEE [41].
The dataset consists of 171 permittivity values with a large variability, in a bandwidth from ~20Hz to 20GHz. A subset of 10 measurement data - selected to highlight the features of each method - has been used as training samples, whilst the remaining 161 samples are used as test samples (i.e., unseen realizations used to evaluate the model accuracy). The selected samples have been organized in the training dataset $D = \{(x_i, y_i)\}_{i=1,\dots,10}$, in which x_i and y_i are the log base 10 of the frequency and permittivity values, respectively. However, thanks to the monotonicity of the log function, the original set of data can be easily reconstructed by means of the exponential function.

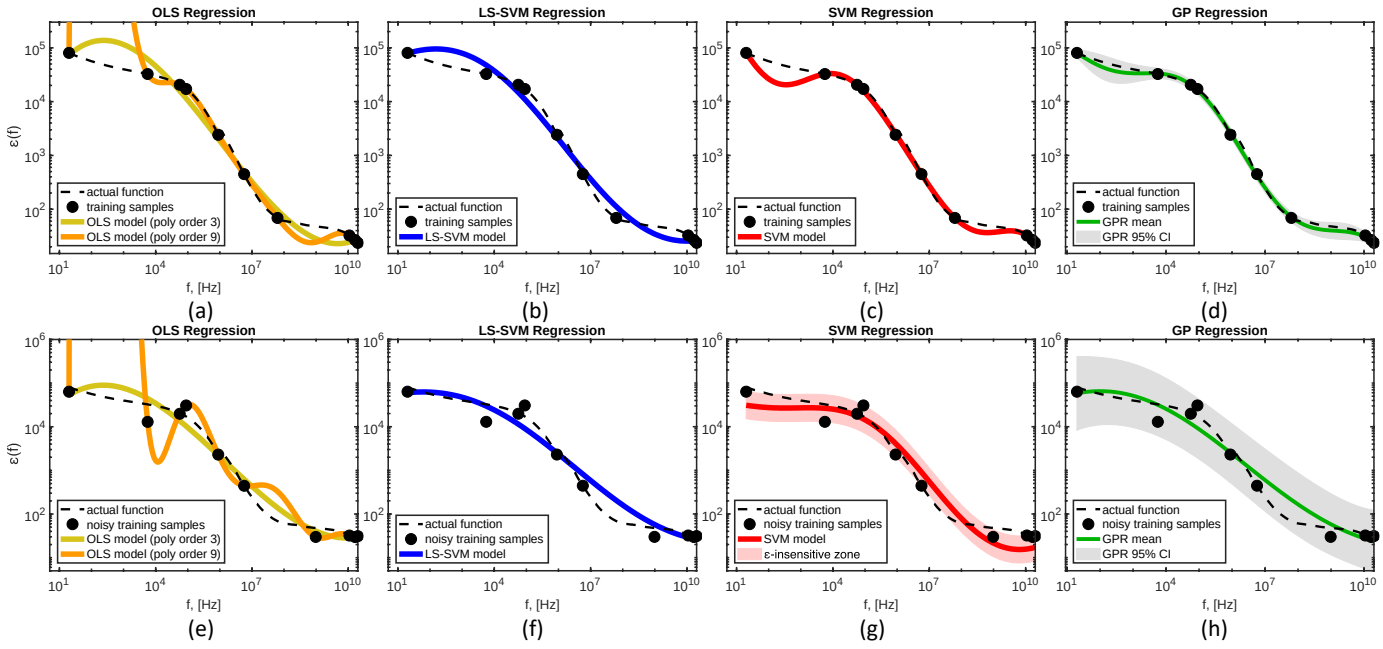

Fig. 4 – Permittivity of the wet human skin predicted by the four surrogate models presented in Sec. II. Panel (a)-(d) show the prediction obtained by the surrogate models trained with noiseless samples. Panel (e)-(h) show the prediction obtained by the surrogate models trained with noisy samples.

The training set D is then used to train five surrogate models based on order 3 and 9 polynomial OLS regression, LS-SVM and SVM regression with RBF kernel and the GPR with squared exponential covariance. Figure 4 (a)-(d) show the prediction obtained by the above surrogate models. The results highlight the detrimental effect of the overfitting due to the OLS surrogate when the order of the polynomial expansion is increased. Also, the plots show the capability of the SVM and LS-SVM to provide an accurate surrogate, since the overfitting issue is limited by the regularizer. Moreover, the results of the probabilistic GPR-based surrogate highlight its capability to provide a probabilistic interpretation of the uncertainty of its prediction calculated via the posterior mean in (14a) in terms of the 95% CI (see the gray area).

As a further validation, the output samples in the training set D are corrupted by an additive Gaussian noise mimicking measurement error, such that: $\tilde{y}_i = y_i \times (1 + \varepsilon)$, where $\varepsilon \sim \mathcal{N}(0, \sigma_\varepsilon^2)$ is a Gaussian variable with zero mean and standard deviation $\sigma_\varepsilon = 0.05$. Again, five surrogate models have been trained and their predictions are shown in Fig. 4 (e)-(h). The plots show again the loss of accuracy of the surrogate model based on OLS with order 9. Indeed, due to the overfitting, the model fits perfectly the noisy training samples, but it does not generalize well on the test samples. Also, the plots highlight that thanks to the ε-insensitive loss function (light red area in Fig. 4(g)), the SVM regression tuns out to be very effective and accurate with noisy samples. On the other hand, the predictions computed via the LS-SVM and the noisy GPR are almost equivalent (see [4] for additional details).

Conducted Emission and Switching Converter

As a second example, we will consider the UQ of the output current spectral envelope of the 12V:5V switching buck converter depicted in Fig. 5 [42]. Similar to [30], all the circuital elements specified in the schematic have been considered as Gaussian stochastic variables centered at their nominal value with a standard deviation of 20% around their mean value, leading to 17 uncorrelated Gaussian parameters (i.e., $x \in \mathbb{R}^{17}$).

The full-computational model adopted in this application is based on a parametric transient simulation in LTspice. For any configuration of the input parameters, a transient simulation has been run in the time window [0,3]ms with a time-step of 10ns. In order to ensure that all the waveforms have reached the steady-state, the FFT has been applied only to the last 3 switching periods of the current waveform. The resulting discrete spectral envelope $I_{out,E}(f_k; \mathbf{x})$ with $k = 1, \ldots, N_f$ covering a frequency bandwidth from DC to 30MHz via a set of $N_f = 91$ linearly spaced frequency samples.

For any discrete frequency f_k, the computational model is used to generate a set of L training samples $\{(\mathbf{x}_i, y_i(f_k))\}_{i=1,\ldots L}$ in which each configuration of input

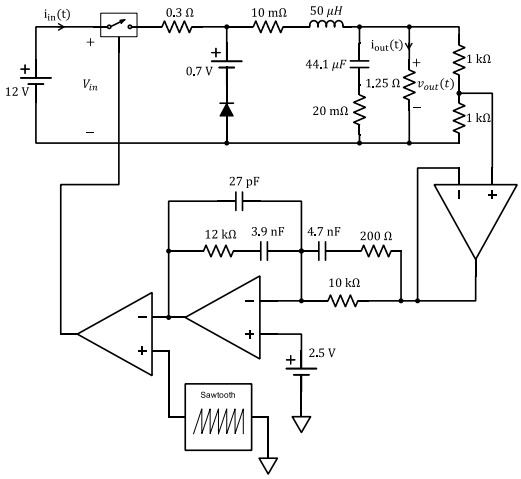

Fig. 5 – Schematic of the considered buck converter [42]. Nominal values of each component are indicated

parameters $\mathbf{x}_i$ have been drawn according to their distribution based on a Latin Hypercube Sampling scheme [7] and $y_i(f_k) = I_{out,E}(f_k; \mathbf{x}_i)$. The training samples have been used to train two surrogate models based on the LS-SVM regression with RBF kernel, a noiseless GPR with a squared exponential function and a second order sparse PCE built via the UQLab toolbox [43]. A second order PCE is considered since for the testcase at hand, the input-output relationship turns out to be slightly nonlinear. More complicated and advanced scenarios are available in [21], [22], [24].

The resulting surrogates will be then used to efficiently predict the effect of the uncertain parameters of the converter on the spectral envelope of the output current. In order to do this, the above regression techniques are used to build a set of N_f metamodels $\widetilde{\mathcal{M}}_k$, one for each of the considered frequency points f_k.

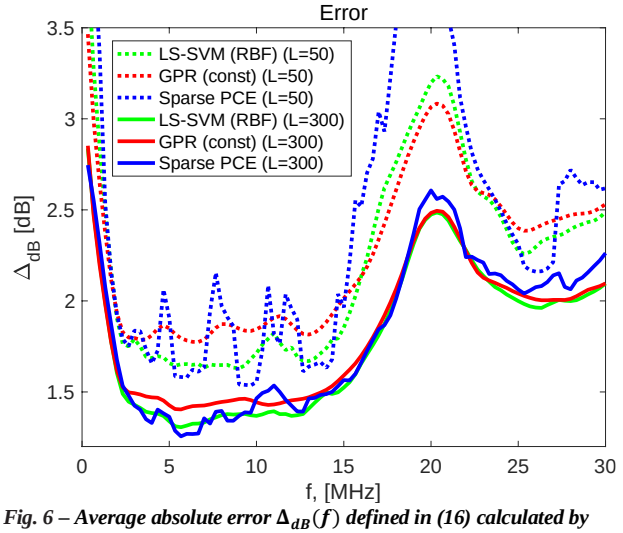

Fig. 6 – Average absolute error $\Delta_{dB}(f)$ defined in (16) calculated by comparing the predictions of the surrogates based on the LS-SVM regression (green lines), a second order PCE (blue lines) and the GPR (red lines) with the corresponding ones obtained via a MC simulation with 10,000 samples for an increasing number of training samples L=50 and 300.

The plots in Fig. 6 provide a comparison among the accuracy provided by each of the considered surrogate models in terms of the average absolute error spectrum $\Delta_{dB}(f_k)$ defined for $k = 1, \dots, N_f$, as follows

$$\Delta_{dB}(f_k) = \sum_{t=1}^{N_{MC}} \frac{\left| I_{out,E}(f_k; \mathbf{x}_t) - \tilde{I}_{out,E}(f_k; \mathbf{x}_t) \right|}{N_{MC}}, \qquad (16)$$

where $I_{out,E}(f_k; \mathbf{x}_i)$ corresponds to the envelope amplitude in dB obtained via the LTspice simulations for the configurations of the uncertain parameters $\mathbf{x}_t$ considered in a MC simulation with N_{MC}=10,000 samples, whilst $\tilde{I}_{out,E}(f_k; \mathbf{x}_t)$ is the corresponding value estimated by the considered surrogates. Such error is computed by considering an increasing number of training samples, i.e., L=50 and 300. The curves of Fig. 6 show that the accuracy achieved by three models is quite similar, even if the LS-SVM regression (green lines) seems to provide the most accurate surrogate models for L=50 and 300, since

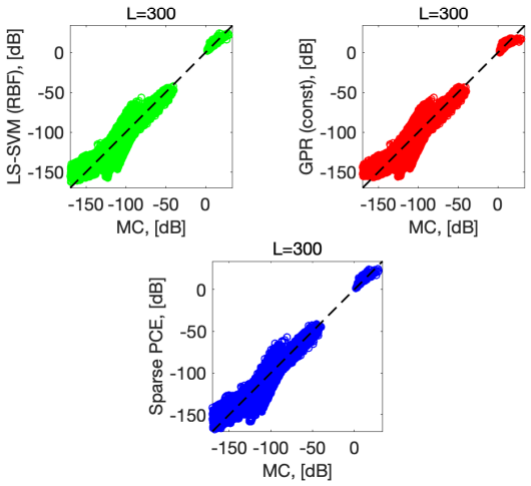

Fig. 7 – Scatter plot (10,000 samples) showing the correlation between the current spectral envelope predicted by the surrogate based on the LS-SVM regression (green dots), a second order sparse PCE (blue dots) and the mean value of the GPR model (red dots) with 10,000 MC samples computed via the computational model.

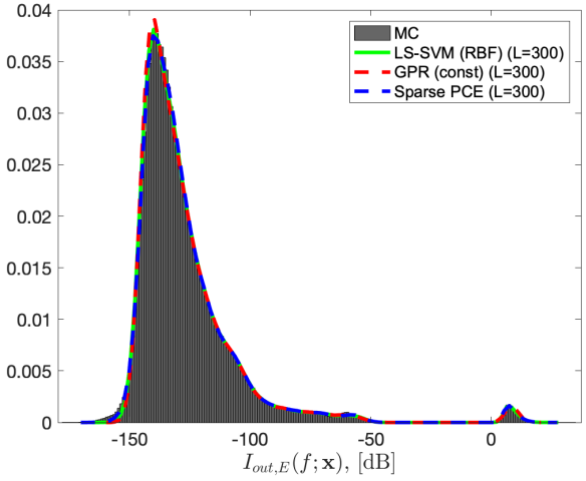

Fig. 8 – Comparison between the pdfs of the amplitude of the spectral envelope calculated from the results of 10,000 samples MC simulation (black bins) with the corresponding values estimated by the LS-SVM regression (solid green line) and a second order PCE (dashed blue line) and the mean values of the probabilistic model based on the GPR (solid red line) for L=300.

their curves are usually below the ones related to GPR (red lines) and sparse PCE (blue lines) surrogates.

As a further validation, Fig. 7 and 8 show the scatter plots and the probability density functions (pdfs) comparing the results of a 10,000 samples MC simulation for all the $N_f = 91$ frequency points, with the corresponding prediction provided by the deterministic LS-SVM surrogate, the mean values of the GPR (red dots) and a second order sparse PCE (dashed blue) built with L=300 samples. The plots highlight the capability of the proposed models to accurately predict, both in deterministic and statistical sense, the actual values calculated during the MC simulation.

As a final comparison between the two models, Fig. 9 shows one realization of the spectral envelope randomly selected

among the results of the MC simulation (black curve) along with the corresponding predictions provided by the surrogated models based on the LS-SVM regression (dashed green curve), the second order sparse PCE (dashed blue) and the probabilistic GPR (red vertical bars and dots) in terms of the mean values and the 95% CIs. Again, the surrogates have been trained with L=300 samples. The results show the capability of the probabilistic GPR model to provide useful statistical information on the model reliability in terms of the 95% CIs (red error bars), since the actual spectral envelope provided by the full-computational model (black curve) lays between the CIs estimated by the proposed models. Indeed, the CIs estimated by the GPR provide a conservative estimation for 85.5% of the 10k current spectra used in the train dataset. Concerning the computational cost required by each of the considered techniques for $L = 300$ training samples, the training cost is 200s for the LS-SVM regression, 11s for the GPR and 20s for the PCE of order 2. The latter grows to 100s for an order 4 expansion. After the training, the model evaluation is extremely fast, and for each of the three methods, requires less than 5s to evaluate the surrogate model for 10,000 configurations of the input parameters. On the other hand, the average cost of a simulation with the full-computational model is 11.7s and the cost of a MC simulation with 10,000 samples is 4h 43min. All the simulations have been performed on a MacBook Pro with an Intel Core i5 CPU running at 3.1GHz and 16GB of RAM. A more exhaustive and detailed comparison of the efficiency and the accuracy of the above modeling techniques for highly non-linear test cases is available in [21], [22] and [24].

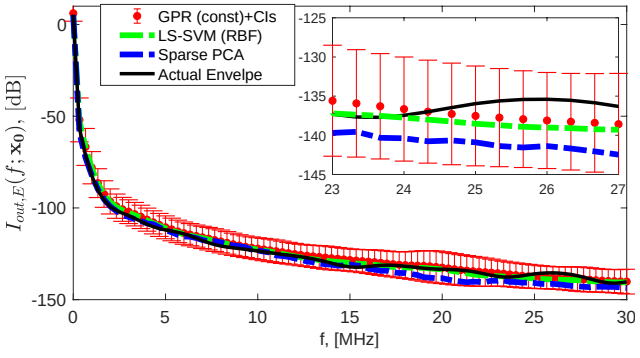

Fig. 9 – Comparison between the envelope spectra for a given configuration of the converter parameters (black curve) randomly chosen among the realizations of the MC simulation with the corresponding predictions of the deterministic models based on the LS-SVM regression with RBF kernel (dashed green curve) and a second order sparse PCE (dashed blue curve), and the mean values and 95% CI estimated by the GPR-based probabilistic model (red dots and vertical bars).

IV. Conclusions

This paper presented a quick overview of three ML regression techniques, which can be suitably adopted for the construction of fast-to-evaluate and accurate surrogate models. Specifically, the main features and the mathematical background of the SVM, LS-SVM and GP regressions have been presented and compared with a state-of-the-art technique such as the OLS regression and sparse PCE. The performances of the proposed techniques have been investigated by considering two illustrative examples.

References

[1] G R. Spence and R. S. Soin, Tolerance Design of Electronic Circuits. London, U.K.: Imperial College Press, 1997.

[2] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams and N. de Freitas, "Taking the Human Out of the Loop: A Review of Bayesian Optimization," in Proceedings of the IEEE, vol. 104, no. 1, pp. 148-175, Jan. 2016

[3] K. T. Fang, R. Li, A. Sudjianto, "Design and modeling for computer experiments", Taylor & Francis Group, LLC, London, 2016.

[4] Jean-Marc Bourinet. Reliability analysis and optimal design under uncertainty - Focus on adaptive surrogate-based approaches. Computation [stat.CO]. Université Clermont Auvergne, 2018.

[5] Sudret, B. Surrogate models for uncertainty quantification and design optimization (2019), Proc. 14 Colloque National en Calcul des Structures, Giens (France), May 13-17.

[6] R. Y. Rubinstein, Simulation and the Monte Carlo Methods. New York: Wiley, 1981.

[7] M. McKay, R.Beckmanand W.Conover, "A comparison of three methods for selecting values of input variables in the analysis of output from a computer code", Technometrics, vol. 42, no. 1, pp. 55–61, 2000.

[8] D. Xiu and G. E. Karniadakis, "The Wiener-Askey polynomial chaos for stochastic differential equations," SIAM J. Scientific Computing, vol. 24, no. 2, pp. 619-644, 2002.

[9] P. Manfredi, D. V. Ginste, I. S. Stievano, D. De Zutter and F. G. Canavero, "Stochastic transmission line analysis via polynomial chaos methods: an overview," in IEEE Electromagnetic Compatibility Magazine, vol. 6, no. 3, pp. 77-84, Third Quarter 2017.

[10] Z. Zhang, T. A. El-Moselhy, I. M. Elfadel, and L. Daniel, "Stochastic testing method for transistor-level uncertainty quantification based on generalized polynomial chaos," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 32, no. 10, pp. 1533-1545, Oct. 2013.

[11] D. Spina, F. Ferranti, T. Dhaene, L. Knockaert, G. Antonini, and D. Vande Ginste, "Variability analysis of multiport systems via polynomial-chaos expansion," in IEEE Transactions on Microwave Theory and Techniques, vol. 60, no. 8, pp. 2329-2338, Aug. 2012.

[12] M. Ahadi and S. Roy, "Sparse Linear Regression (SPLINER) Approach for Efficient Multidimensional Uncertainty Quantification of High-Speed Circuits," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 35, no. 10, pp. 1640-1652, Oct. 2016

[13] G. Blatman and B. Sudret, "Adaptive sparse polynomial chaos expansion based on least angle regression," Journal of Computational Physics, vol. 230, no. 6, pp. 2345–2367, Mar. 2011.

[14] V. Yaghoubi, S. Marelli, B. Sudret, and T. Abrahamsson, "Sparse polynomial chaos expansions of frequency response functions using stochastic frequency transformation," Probabilistic engineering mechanics, vol. 48, pp. 39–58, 2017.

[15] S. J. Russell, P. Norvig and E. Davis, Artificial Intelligence: A Modern Approach. Upper Saddle River, NJ: Prentice Hall, 2010.

[16] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT, 2017

[17] V. Vapnik, The Nature of Statistical Learning Theory, 2nd ed. New York, NY, USA: Springer-Verlag, 1999.

[18] V. Vapnik, Statistical Learning Theory. New York, NY, USA: Wiley, 1998.

[19] J. A. K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor. and J. Vandewalle, "Least Squares Support Vector Machines", World Scientific Pub Co Inc, 2002.

[20] C. E. Rasmussen and C. K. I. Williams, Gaussian Processes for Machine Learning, MIT Press. Cambridge, Massachusetts, 2006.

[21] R. Trinchero, M. Larbi, H. M. Torun, F. G. Canavero and M. Swaminathan, "Machine Learning and Uncertainty Quantification for Surrogate Models of Integrated Devices With a Large Number of Parameters," in IEEE Access, vol. 7, pp. 4056-4066, 2019.

[22] R. Trinchero, P. Manfredi, I. S. Stievano and F. G. Canavero, "Machine Learning for the Performance Assessment of High-

Speed Links," in IEEE Transactions on Electromagnetic Compatibility, vol. 60, no. 6, pp. 1627-1634, Dec. 2018.
[23] R. Trinchero, I. S. Stievano and F. G. Canavero, "Black-Box Modeling of the Maximum Currents Induced in Harnesses During Automotive Radiated Immunity Tests," in IEEE Transactions on Electromagnetic Compatibility, vol. 62, no. 2, pp. 627-630, April 2020.
[24] P. Manfredi and R. Trinchero, "A Data Compression Strategy for the Efficient Uncertainty Quantification of Time-Domain Circuit Responses," in IEEE Access, vol. 8, pp. 92019-92027, 2020
[25] H. Ma, E. Li, A. C. Cangellaris and X. Chen, "Support Vector Regression-Based Active Subspace (SVR-AS) Modeling of High-Speed Links for Fast and Accurate Sensitivity Analysis," in IEEE Access, vol. 8, pp. 74339-74348, 2020
[26] R. Medico, D. Spina, D. Vande Ginste, D. Deschrijver and T. Dhaene, "Machine-Learning-Based Error Detection and Design Optimization in Signal Integrity Applications," in IEEE Transactions on Components, Packaging and Manufacturing Technology, vol. 9, no. 9, pp. 1712-1720, Sept. 2019
[27] B. Li, B. Jiao, C. Chou, R. Mayder and P. Franzon, "Self-Evolution Cascade Deep Learning Model for High-Speed Receiver Adaptation," in IEEE Transactions on Components, Packaging and Manufacturing Technology, vol. 10, no. 6, pp. 1043-1053, June 2020
[28] H. Yu, T. Michalka, M. Larbi and M. Swaminathan, "Behavioral Modeling of Tunable I/O Drivers With Preemphasis Including Power Supply Noise," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 28, no. 1, pp. 233-242, Jan. 2020.
[29] H. Kabir, L. Zhang, M. Yu, P. H. Aaen, J. Wood and Q. Zhang, "Smart Modeling of Microwave Devices," in IEEE Microwave Magazine, vol. 11, no. 3, pp. 105-118, May 2010
[30] R. Trinchero and F. G. Canavero, "Combining LS-SVM and GP Regression for the Uncertainty Quantification of the EMI of Power Converters Affected by Several Uncertain Parameters," in IEEE Transactions on Electromagnetic Compatibility, vol. 62, no. 5, pp. 1755-1762, Oct. 2020.
[31] Hastie, T., Tibshirani, R., & Friedman, J. H. (2009). The elements of statistical learning: data mining, inference, and prediction. 2nd ed. New York: Springer.
[32] B. Ghojogh and M. Crowley. The theory behind overfitting, cross validation, regularization, bagging, and boosting: Tutorial. arXiv preprint arXiv:1905.12787, 2019.
[33] M. Kuhn and K. Johnson. Applied Predictive Modeling. Springer, New York, 2013.
[34] A. J. Smola and B. Scholkopf, “A tutorial on support vector regression”, Statistics and Computing, vol. 14, no. 3, pp. 199–222, 2004.
[35] LS-SVMlab, version 1.8; Department of Electrical Engineering (ESAT), Katholieke Universiteit Leuven: Leuven, Belgium, 2011. Available online: http://www.esat.kuleuven.be/sista/lssvmlab/.
[36] Statistics and Machine Learning Toolbox, version 11.7, The MathWorks,Inc., Natick, MA, USA.
[37] F. Lindsten, T.B. Schön, A. Svensson, N. Wahlström, Probabilistic modeling – linear regression & Gaussian processes, Uppsala Univ., Uppsala, 2017.
[38] F. Perez-Cruz, S. Van Vaerenbergh, J. J. Murillo-Fuentes, M. Lazaro-Gredilla and I. Santamaria, “Gaussian Processes for Nonlinear Signal Processing: An Overview of Recent Advances,” IEEE Signal Processing Magazine, vol. 30, no. 4, pp. 40–50, July 2013.
[39] C. Gabriel, Compilation of the dielectric properties of body tissues at RFand microwave frequencies. Air Force materiel command, Brooks AirForce Base, Texas: AL/OE-TR-1996-0037; 1996 (Appendix D)
[40] ICNIRP. Guidelines for limiting exposure to electromagnetic fields (100 kHz to 300 GHz). Health Phys 118(00):000–000; 2020. Pre-print. DOI: 10.1097/HP.0000000000001210.
[41] IEEE C95.1-2019 - IEEE Standard for Safety Levels with Respect to Human Exposure to Electric, Magnetic, and Electromagnetic Fields, 0 Hz to 300 GHz. https://standards.ieee.org/standard/C95_1-2019.html
[42] R. Muyshondt and P. T. Krein, “20 W benchmark converters for simulaWon and control comparisons,” in Record 6th Workshop on Computer in Power Electronics pp. 201–212, Cernobbio, Italy, 1998.
[43] S. Marelli and B. Sudret, “UQLab: A framework for uncertainty quantification in Matlab,” in Proc. 2nd Int. Conf. on Vulnerability, Risk Analysis and Management, Liverpool, United Kingdom, Apr. 2014, pp. 2554–2563.