A Software Framework for Automated Behavioral Modeling of Electronic Devices

Dirk Gorissen, *Student Member, IEEE*, Dirk Deschrijver, *Member, IEEE*, Tom Dhaene, *Senior Member, IEEE*, Daniel De Zutter, *Fellow, IEEE*

***Abstract*—As the fabrication of prototypes is very costly, system-level computer simulations have become commonplace alternatives for the design of electronic devices and systems. However, due to the computational cost of these simulations, the use of behavioral modeling techniques has become indispensable. Behavioral models can act as a surrogate for the expensive simulations, and effectively speed-up the simulations without sacrificing accuracy. Such models are compact and cheap to evaluate, and have proven very useful for tasks such as optimization, design space exploration, prototyping, and sensitivity analysis. Consequently, there is great interest in techniques that facilitate the construction of behavioral models, while minimizing the computational cost and maximizing model accuracy. This paper explores how such models can be calculated in an automated way by means of a unified software framework that integrates adaptive modeling and adaptive sampling methods. It generates global behavioral models that are accurate and valid over the design space of interest while minimizing the number of electromagnetic simulations. By placing a strong focus on adaptivity, flexibility, and self-tuning the burden on the designer is relieved and the total modeling turn-around time is reduced.**

I. Introduction

Behavioral models (such as macromodels, surrogate models, metamodels and response surface models) have many applications in diverse research domains such as aerodynamics [1], hydrology [2], mechanical engineering [3], and many more. When considering the design flow of electronic devices, these models are often used to characterize the time- or frequency-dependent behavior of an electronic component while taking all electromagnetic phenomena into account (crosstalk, attenuation, dispersion, coupling effects,...) [4]. Such models are of crucial importance for efficient design space exploration, design optimization and sensitivity analysis [5], [6]. A key advantage is that they are calculated independently of the device's physics and that they are valid of over a wide range of design variables, taking into account multiple geometrical layout or substrate features. Additionally, the models can easily be linked together in a model cascade.

Some examples of behavioral models include polynomial/rational functions [7], [8], [9], Kriging models [10], [11], Artificial Neural Networks (ANN) [12], [13], and Support Vector Machines (SVM) [14]. In order to obtain a reliable model that satisfies all design requirements, significant challenges need to be addressed: which data collection strategy to use, which model type is most applicable, how to rank different models according to quality, etc. At the same time, Electronic Design Automation (EDA) experts are typically not familiar with the intricacies of these design choices. Their primary concern is obtaining an accurate replacement model with minimal computational overhead. The selection of model types, model parameter optimization and sampling strategy are of lesser or no interest to them, since these are just necessary intermediate steps to solve the overall design problem.

In this paper, a unified and automated modeling framework is presented that can assist an EDA domain expert in generating accurate behavioral models [15]. It drives the underlying system-level simulator, and at the same time builds and tunes the model in such a way that the model accuracy and compactness is maximized. On the one hand it does not require particular assumptions about the device under test. However, on the other hand, as no algorithm is optimal for every problem, full control is still left with the EDA domain expert such that problem specific assumptions or customizations can easily be applied. Therefore, this work can lower the barrier of entry for domain experts, promote benchmarking between existing methods, and facilitate the transfer of knowledge from behavioral modeling researchers to EDA domain experts.

II. Global Behavioral Modeling

From an abstract level, the system-level computer simulator can be seen as an unknown multivariate function $f : \Omega \mapsto \mathbb{C}^q$, that is defined on some domain $\Omega \subset \mathbb{R}^d$, and whose function values $Y = \{f(\mathbf{x}_1), ..., f(\mathbf{x}_k)\} \subset \mathbb{C}^q$ are known at a fixed set of pairwise distinct sample points $X = \{\mathbf{x}_1, ..., \mathbf{x}_k\} \subset \Omega$ [16]. The behavioral model is then a suitable function $\tilde{f}$ that closely resembles f as measured by some criterion ξ, where ξ is defined as a triplet that consists of following parts:

$$\xi = (\Lambda, \varepsilon, \tau) \tag{1}$$

Λ is a model quality estimator, i.e. a function that assigns a positive score to a behavioral model where lower scores indicate a more desirable model. Many implementations of Λ have been described: the error in the samples, the hold-out, bootstrap, cross validation, jack-knife, Akaike's Information Criterion (AIC), etc. [17]. An implementation of Λ is typically associated with an error function ε. While Λ specifies the model quality estimation algorithm (e.g., cross validation), ε specifies what error function should be used to calculate the actual quality score (e.g., mean relative error, maximum

Dirk Gorissen, Dirk Deschrijver, Tom Dhaene and Daniel De Zutter are with Ghent University - IBBT, Department of Information Technology (INTEC), Gaston Crommenlaan 8, Bus 201, 9050 Ghent, Belgium. E-mail: {dirk.gorissen, dirk.deschrijver, tom.dhaene, daniel.dezutter}@intec.ugent.be.

absolute error, etc.). Finally, τ is the model quality target desired by the user.

The behavioral modeling problem (i.e., finding the best approximation $\tilde{f}^*$) for a given set of data points $D = \{(\mathbf{x}_1, f(\mathbf{x}_1)), ..., (\mathbf{x}_k, f(\mathbf{x}_k))\}$ can be formally defined as

$$\tilde{f}^* = \arg\min_{t \in T} \arg\min_{\theta \in \Theta} \Lambda(\varepsilon, \tilde{f}_{t,\theta}, D) \tag{2}$$

such that

$$\Lambda(\varepsilon, \tilde{f}^*_{t,\theta}, D) \leqslant \tau \tag{3}$$

where $\tilde{f}_{t,\theta}$ is the parametrization θ (from a parameter space Θ) of $\tilde{f}$, and $\tilde{f}_{t,\theta}$ is of model type t (from a set of model types T).

The first minimization over $t \in T$ is the task of selecting a suitable model type, i.e., a rational function, a neural network, a spline, etc. This is the model type selection problem. In practice, one typically considers only a single $t \in T$, though others may be included for comparison. In some cases, the choice of model type is also linked to physical properties of a system (e.g. causality, stability, passivity) [18], [19], [20]. Then given a particular model type t, the task is to find the model parameter assignment θ that minimizes the model quality measure Λ (e.g. determine the optimal order of a polynomial model or the optimal neural network topology). This is referred to as the hyperparameter optimization problem, though generally both minimizations (over t and over θ) are simply referred to as the model selection problem.

In order to construct $\tilde{f}$, the dataset needs to be populated. Traditionally the size and distribution of the data is chosen up-front using standard experimental designs (e.g., a latin hypercube design). However, since $f(\cdot)$ is expensive to compute it becomes important to avoid unnecessary simulations. Consequently, since the complexity of the response surface is not known up-front, defining an a priori data distribution is undesired. Instead data points should be selected iteratively, at locations where the benefit to the model will be the greatest.

One starts by constructing an *initial experimental design* using one of the many algorithms available from the theory of Design and Analysis of Computer Experiments (DACE) [21]. The task is then to generate a new set of maximally informative samples based on one or more criteria. Examples of such criteria include distance from other points (spacefilling-ness), distance from optima, prediction uncertainty, etc. In each iteration of sampling algorithm, a best approximation model is obtained, leading to a sequence of models. The overall number of data samples should be kept to a minimum, while at the same time maximizing the benefit to the model $\tilde{f}$ with respect to ξ.

This process is called adaptive sampling, but is also known as active learning [22], reflective exploration [7], Optimal Experimental Design [23], Sequential Exploratory Experimental Design [24], and sequential design [25].

Fig. 1 shows how the model generation and data collection fit together in a high level control flow.

There are two main parts to the flowchart: an outer, adaptive sampling loop and an inner adaptive modeling loop. Given a set of sample points, the inner loop will optimize the model hyperparameters (e.g., neural network topology) until no further improvement is possible on the given set of samples. Having reached a minimum of the hyperparameter space (e.g., the inner minimization in (2)), the algorithm then signals the sample selection loop to determine a new, maximally informative set of simulation points. These are then evaluated, intelligently merged with the existing sample points, and the adaptive modeling process is allowed to resume. This whole process continues until the user-defined accuracy has been reached or some user-defined stopping criterion is met. The final behavioral model is returned to the user.

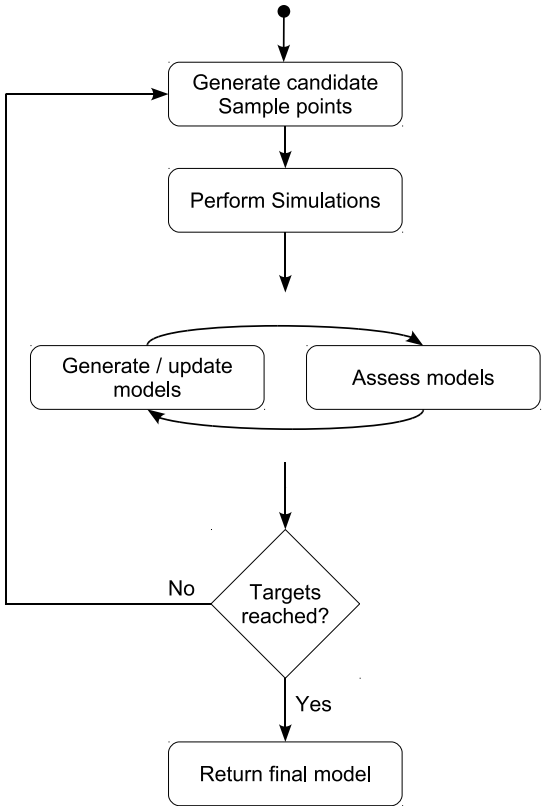

Figure 1. General flowchart for adaptive global behavioral modeling. The outer sampling loop will iteratively sample the design space while an inner modeling loop will generate models that accurately capture the data selected each iteration.

The flow of control in Fig. 1 forms the basis for every data based approximation methodology. For example, note the similarities with the work by Zhang in [26]. Of course, as-is, Fig. 1 has little practical value since it is very vague. Like the Expectation-Maximization algorithm [27] it is really a meta-algorithm that can have many different actual instantiations depending on what techniques are used for each step. However, it should be clear that there are a large number of modeling options and choices available to the designer: different model types, different experimental designs, different sample selection strategies, different model selection criteria, different hyperparameter optimization strategies, etc. All choices that are difficult to make in an a priori manner [28].

Consequently, this generic control flow is taken as a starting backbone, extended with pluggable components for each of the

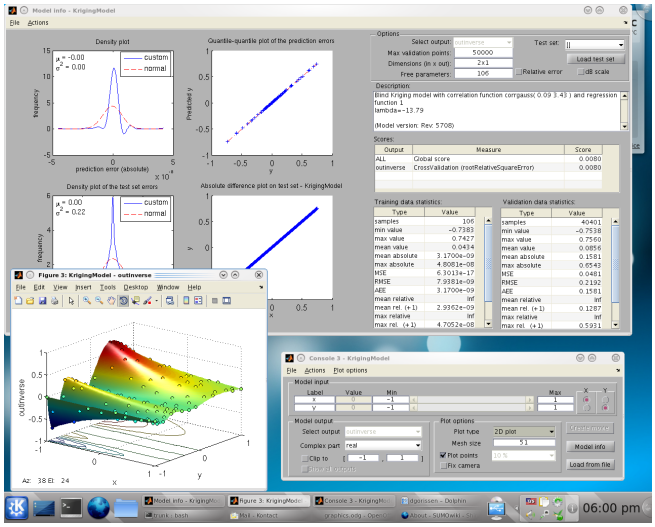

Figure 2. Screenshot of SUMO Toolbox.

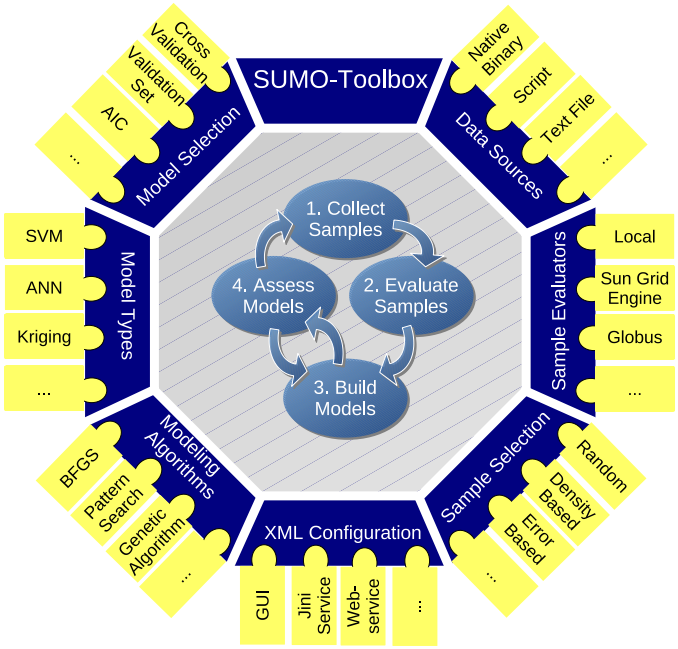

Figure 3. Illustration of how the SUMO Toolbox enables algorithms and methods to be combined in different ways through the use of a plugin-based infrastructure.

different steps. This results in a generic solution that can still be customized for a specific problem if needed. In addition, custom extensions ensure that the whole is more than a bag-of-tools but that some automation (e.g., for model type selection) is available if needed. The resulting modeling platform can easily help an EDA domain expert to choose between different techniques and help him to apply advanced modeling methods to his problem with minimal overhead. Depending on the configuration, the modeling platform can run autonomously (if just a 'quick-and-dirty' model is needed) or under full manual control with problem specific methods and customizations (to ensure optimal efficiency accuracy for a given problem). The framework in question is the **SU**rrogate **MO**deling MATLAB toolbox (SUMO Toolbox).

III. SUMO Toolbox

The SUMO Toolbox illustrated in Fig. 2 [29], [30] is a flexible tool that integrates different modeling approaches and implements an adaptive behavioral model construction algorithm based on the flowchart in Fig. 1. Given an EDA simulation engine (e.g., SPICE, SPECTRE, HFSS, Momentum, CST, etc.) the toolbox computes a model within the time and accuracy constraints set by the user. Different plugins are supported: model types (rational functions, Kriging, splines, SVM, etc.), model parameter optimization algorithms (Particle Swarm Optimization, Efficient Global Optimization, simulated annealing, etc.), sample selection (random, error based, density based, etc.), and sample evaluation methods (local, on a cluster or grid). The behavior of each software component is configurable through a central XML [31] configuration file and components can easily be added, removed or replaced by custom implementations. This is illustrated in Fig. 3. In addition the toolbox provides 'meta' plugins which include, e.g. evolutionary algorithms to automatically select the best model type for a given problem (see [28] for details), or the possibility to use multiple model selection criteria in concert [32].

Furthermore, there is built-in support for high performance computing. On the modeling side, the model generation process can take full advantage of multi-core CPUs (if the MATLAB Parallel Computing Toolbox is available) and even of a complete cluster or grid (if the MATLAB Distributed Scheduler is available). This can result in significant speedups for model types where the fitting process can be expensive (e.g., neural networks).

Likewise, sample evaluation (simulation) can occur locally (with the option to take advantage of multi-core architectures) or on a separate compute cluster or grid (possibly accessed through a remote head-node). All interfacing with the grid middleware (submission, job monitoring, rescheduling of failed/lost simulation points, etc.) is handled transparently and automatically (see [33] for more details). Also, the sample evaluation component runs in parallel with the other components (non-blocking) and not sequentially. This allows for an optimal use of computational resources.

The SUMO Toolbox has already been applied successfully to a very wide range of applications, including RF circuit block modeling [29], hydrological modeling [34], Electronic Packaging [35], aerodynamic modeling [36], and automotive data modeling [32]. Besides global modeling capabilities, the SUMO Toolbox also includes a powerful optimization framework based on the Efficient Global Optimization framework developed by Jones [37].

IV. Applications

To demonstrate the working and performance of the framework described above, two EM device modeling problems are discussed: a passive component (bandstop filter) and an active device (low noise amplifier).

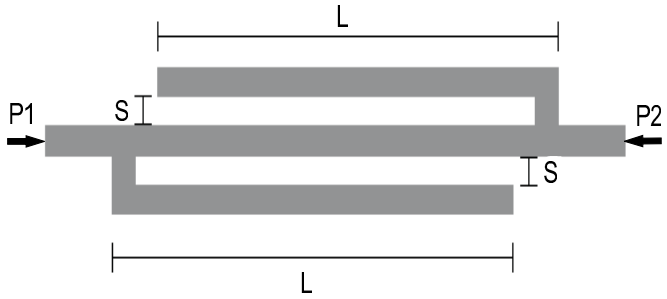

Figure 4. Double-folded microstrip stub bandstop filter

A. EM Behavioral Modeling of a Bandstop Filter

The first application concerns the modeling of a parametrized double-folded microstrip stub bandstop filter [6]. The filter with ports P_1 and P_2 is shown in Fig. 4.

The substrate is 0.1270 mm thick with a relative dielectric constant $\varepsilon_r = 9.9$ and a loss tangent $tan\,\delta = 0.003$. The lines are infinitely thin and perfectly conducting with $W = 0.1219$ mm. The parametric macromodel of the scattering matrix is built as a function of the varying length of each folded segment $L \in [1.97, 2.41]$ mm and varying spacing between a folded stub and the main line $S \in [0.06, 0.24]$ mm over the frequency range $[5, 20]$ GHz. The EM simulation engine used is ADS Momentum.

In order to objectively assess the accuracy of the models, a dense $50 \times 30 \times 151$ ($L \times S \times frequency$) reference grid was calculated. It is important to note that this dataset is not used during the modeling process in any way since typically such a reference grid is not available. It is simply used to objectively test the quality of the models a posteriori.

1) Modeling settings: The SUMO Toolbox v6.2.1 is configured with the generic ANN and multivariate rational modeling plugins. No problem specific tuning or settings were used. Thus the results will be indicative of how good a behavioral model can be obtained if the framework is applied without further problem specific knowledge, using just concepts from a data modeling perspective. The multivariate rational models are based on a custom implementation [38]. The order selection is performed using a genetic algorithm (population size: 30, number of generations: 20), thus this need not be done manually. The model quality estimator, $\Lambda(\cdot)$, is 5-fold cross validation [39], [17] configured with a Mean Square Error (MSE) error function (ε). The ANN models are based on the MATLAB Neural Network Toolbox and are trained with Levenberg-Marquardt backpropagation with Bayesian regularization [40], [41] (600 epochs). Since the ANN models do not support complex data directly, the real and imaginary components are fitted separately using an ANN model with two outputs. The topology and initial weights are determined by an evolutionary strategy-like algorithm, with 25 models being generated each modeling iteration. To assess the model quality and drive the topology selection, $\Lambda(\cdot)$ is taken as the sum of two criteria: the in-sample error (using a MSE) and the Linear Reference Model (LRM) score [42]. The combined scores for each output (real/imaginary) are then added together to obtain the overall score of the model. The LRM score penalizes a model if it exhibits unwanted bumps or 'ripples' between the sample points. It can be seen as a kind of smoothness penalty that has the added benefit of keeping the neural network model complexity low. We found that the advantage of using these two metrics together is that they produce better ANN models and are much faster to evaluate than cross validation.

2) Sampling settings: The modeling starts with an optimal Latin hypercube design of 4 points augmented with the corner points in the 2-dimensional $L \times S$ space. Each iteration a new sample is selected using the LOcal Linear Approximation-Voronoi (LOLA-Voronoi) adaptive sampling algorithm [43]. LOLA-Voronoi identifies new sample locations by performing a trade-off between exploration (covering the design space evenly) and exploitation (concentrating on regions where the actual response is nonlinear). LOLA-Voronoi's strengths are that it scales well with the number of dimensions, makes no assumptions about the underlying problem or behavioral model type, and works in both the $\mathbb{R}$ and $\mathbb{C}$ domains. LOLA can automatically identify nonlinear regions in the domain and sample these more densely than the more linear, 'flatter' regions. In order to do this, LOLA-Voronoi depends only on previously evaluated data points.

Because frequency is sampled automatically by ADS Momentum, LOLA-Voronoi samples in the 2-dimensional instance space defined by the geometric parameters L and S. New samples are submitted to ADS Momentum, which returns a set of S-parameters over the frequency range of interest. In order to select a sample in the reduced 2-dimensional design space (without the frequency parameter), slices are taken at multiple frequencies, and LOLA-Voronoi is used on each slice separately. This results in a nonlinearity estimation for each frequency slice, covering the entire 2-dimensional design space. These estimations are aggregated into one score, which is used to select new samples in locations with the highest nonlinearity over the entire frequency range.

Momentum is configured to return 31 frequency samples and the SUMO Toolbox is set to terminate after 136 instance simulations (=136×31=4216 data points). Thus instead of using an explicit target accuracy value, simulations are performed until the computational budget is exhausted ($\tau = -\infty$).

3) Results: Fig. 5 shows a plot of the final rational models for S_{11} and S_{12}. For conciseness the following discussion will only treat S_{11}. The results for S_{12} are completely analogous.

Fig. 6 shows that the ANN model generation code is able to reduce the true error (evaluated over a dense reference grid) quite effectively while minimizing the estimated error. This means that the in-sample error/LRM combination is a good approximator of the true accuracy. In line with previous results [42] we see that the decrease in true error is quite steady with no large jumps.

Fig. 7 shows the probability density function of the absolute errors (obtained using kernel density estimation [44]) for the final best ANN model found by the toolbox after 136 Momentum simulations. The figure shows that very good accuracy is achieved. Note also the small difference between the training and test curves, meaning there is no overfitting and the models show good generalization. The final ANN model

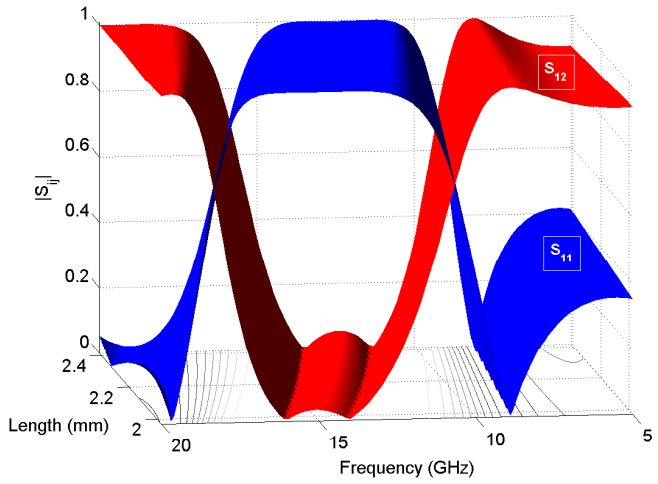

Figure 5. Plot of the final rational model for $|S_{11}|$ and $|S_{12}|$ at $S = 0.131$

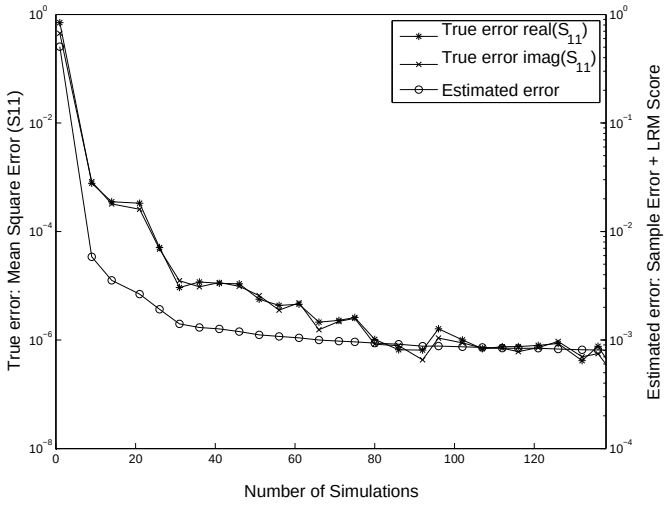

Figure 6. Evolution of the true and estimated error over the reference data during the ANN topology optimization. Note that the true error is not minimized directly, it is only shown here for reference.

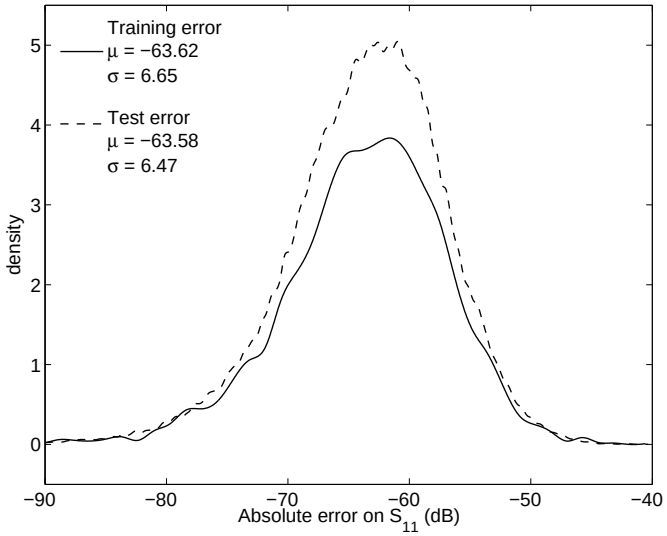

Figure 7. Probability density function of the absolute errors of the final ANN model over the training and reference data after 136 simulations.

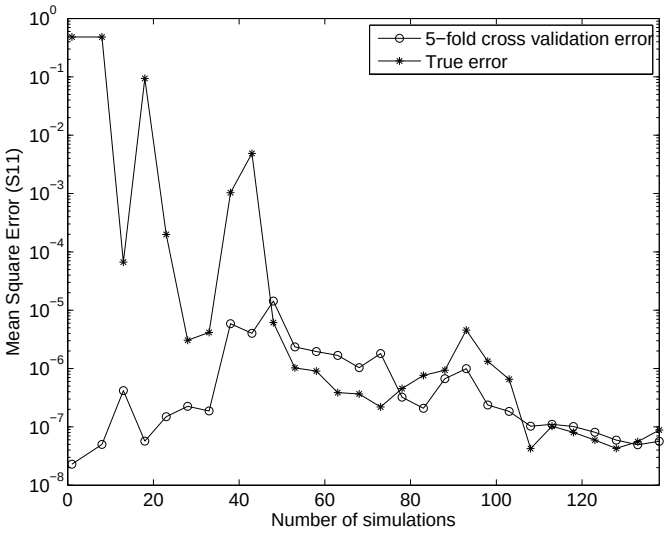

Figure 8. Evolution of the true error and estimated error over the reference data during the rational model order selection.

for S_{11} is a $3-8-16-2$ network (210 parameters) and for S_{12} a $3-12-15-2$ network (275 parameters). This notation denotes the number of elements of the input vector, and the number of neurons in each layer of the ANN.

Let us now regard the results for the multivariate rational functions. The evolution of the true error and estimated error during the model generation process is shown in Fig. 8. Compared to the ANN results in Fig. 6 we see that the error reduction is more erratic in the rational case. Particularly in the beginning, when only little data is available. This is due to the use of cross validation as the accuracy estimator. Even though we ensure an even distribution of the different folds, when data is relatively sparse cross validation is known to give biased results [17] and can mislead the order selection procedure.

Again, good accuracy is achieved on S_{11}, with the mean accuracy being better than for the ANN models. Given the rational nature of the underlying transfer function this should not be surprising. However, it should be noted that the modeling effort was not the same for both model types. Since the rational functions are fast to construct and train we can afford to build more of them during each modeling iteration than neural network models (since these are much slower to train). In this case 20×30=600 rational models are built each modeling iteration versus only 25 neural networks.

B. EM Behavioral Modeling of a Low Noise Amplifier

The second application is concerned with the accurate capturing of the non linear behavior of a Low Noise Amplifier (LNA). An LNA is characterized by performance figures (e.g., voltage gain, linearity, noise figure, etc.) which are functions of the design parameters (e.g., width and length of transistors, bias conditions, values of passive components) [45]. The goal of the design process is to figure out one or more sets of design parameters resulting in a circuit which fulfills the

specifications, i.e., constraints given on the performances. More information on the modeling of this problem can be found in [29].

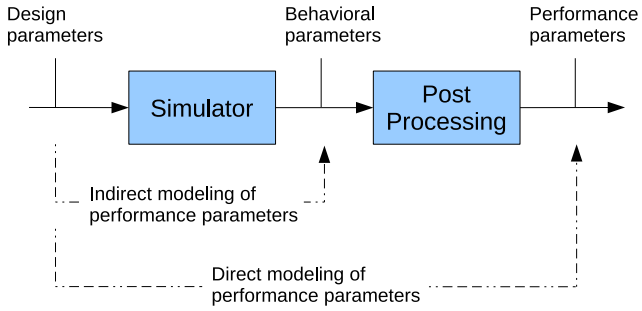

Figure 9. Direct and indirect modeling of the LNA performance parameters

Obtaining the required circuit design parameters can be done through an approximation of the circuit performance figures based on one or more behavioral models. This is referred to as 'forward model' of the circuit. A forward model can be either obtained via direct modeling of circuit performances (i.e., a one step approach) or by using intermediate surrogate models of a convenient set of behavioral parameters (e.g. admittances and noise functions) and by computing performances via analytical equations in a post-processing step (i.e., a two step approach). This is illustrated in figure 9.

To illustrate the indirect modeling approach we consider the LNA and attempt to reproduce the behavior of the input-noise current response variable $\sqrt{\overline{i_{in}^2}}$. The input parameters are the MOSFET width W, the inductances L_s, L_m, and the frequency, leading to a 4-dimensional (4D) problem [29]. To illustrate the direct modeling approach we model the $IIP2$ performance parameter (denoting the second order nonlinearity). We also use this approach to illustrate the scalability of the modeling algorithms used. Instead of fixing the number of inputs we vary them from 2 to 6. This will give insight in how the model accuracy and required number of data points changes as the dimensionality is increased (i.e. from 2D to 6D). The relevant input parameters are the transistor width W, the source inductance L_s, the load resistance RL, the voltage bias of the transistor V_{GS}, the transistor length L, and the resistance in series with the generator RS (the generator series resistance).

1) Modeling settings: We use the same setup as the previous section but now only use ANN models since experience showed these to perform best on this problem. To objectively assess the accuracy of the produced models reference test sets of size $51^2, 15^3, 11^4, 7^5, 5^6$ are available. Again remember that these do not influence the modeling process itself.

2) Sampling settings: The same sample selection algorithm is used as with the filter application. Only this time there is no autosampling (the frequency is not treated specially) and the sample budget is extended to 1580 points in the indirect case and 3000 points in the direct case. The data source is now a custom MATLAB script instead of ADS Momentum.

3) Results: Fig. 10 shows the evolution of the relative error histogram over the reference data for the indirect problem. After 1580 points the error histogram corresponds to a mean

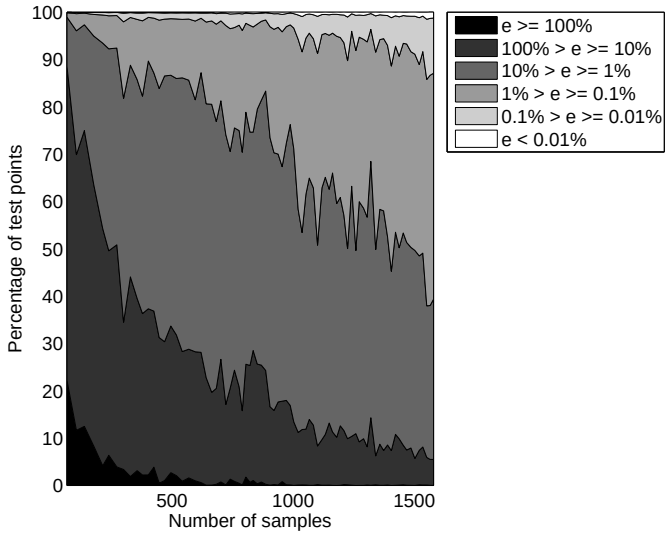

Figure 10. Evolution of the true error histogram (using a relative error) over the reference data during ANN model generation for the 4D indirect LNA modeling problem. More lighter regions mean a higher percentage of the reference points have a low error. The final model after 1580 points has mean relative error of 2.5%.

Table I
ELAPSED TIME TO COMPUTE 2D-6D MODELS FOR SECOND ORDER NONLINEARITY ($IIP2$) OF LNA PERFORMANCE PARAMETERS.

LNA	2D	3D	4D	5D	6D
mins	0.4	1.2	24	61	192

relative error (over all test data) of 2.5% for the final 4D model.

For the direct modeling problem then, Fig. 11 shows a visualization of the selected data samples and the model for the 2D case. Fig. 12 shows the true accuracy curve for each number of inputs from 2D to 6D. The true accuracy is calculated as the Root Relative Square Error (RRSE) on each test set. The RRSE is defined as

$$RRSE(\mathbf{y}, \tilde{\mathbf{y}}) = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}} \tag{4}$$

where $\mathbf{y}, \tilde{\mathbf{y}}, \bar{y}$ are the true, predicted and mean true response values respectively.

The curves in Fig. 12 depict how good the LRM-SampleError combination is at minimizing the error on the reference grid. Desirable features are a smooth, monotonic decrease of the error as a function of the number of data points. The steeper the descent the better. Erratic jumps should be avoided but temporary increases in error are permitted. The error may temporarily increase if adding new data points reveals new features in the data or a new interpretation. What was thought to be a good model may turn out to be less accurate given the new information.

Fig. 12 shows that satisfactory accuracy (which in this case is defined as a RRSE score of 0.05) can be reached in all cases, with convergence being particularly fast in the 2D and 3D case. A second observation is that the curves for 5D and 6D are rather erratic, much more so than the 2D-4D curves. The most likely reason for this lies in the fact that the LRM algorithm

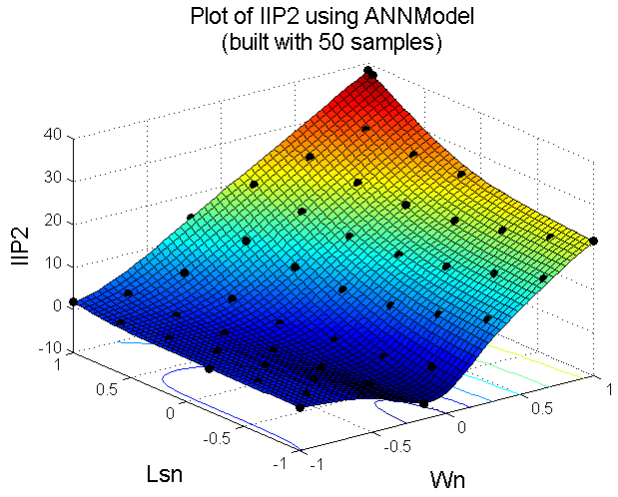

Figure 11. Plot of the final 2D ANN model for IIP2 with $W = 100 * 10^{-6} * 10^{Wn}$ m and $L_s = 0.2 * 10^{-9} * 10^{Lsn}$ H. Selected data samples are marked (•).

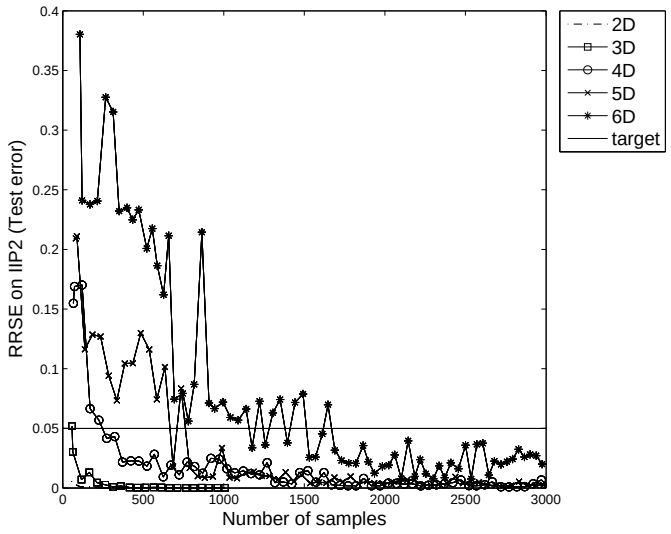

Figure 12. Evolution of the true error across different dimensions when modeling the second order nonlinearity (*IIP*2) of the LNA performance parameters.

uses too little test points to estimate the model smoothness in higher dimensions [42]. Table 1 shows the running times on a Laptop with Intel(R) Core(TM) i7-2760QP CPU at 2.4 GHz with 8 GB of RAM and a 64-bit operating system.

V. Conclusion

The continuous emergence of new devices and circuit design techniques has led to the development of a wide range of behavioral modeling methods that facilitate design space exploration, optimization, and sensitivity analysis. This paper discussed how the use of data modeling techniques can be leveraged in a flexible modeling framework to facilitate domain experts in their modeling task. By way of example, accurate behavioral models with good generalization and scalability were generated for electromagnetic bandstop filter and low noise amplifier modeling problems. Sample selection and model complexity setting were handled fully autonomously. Future work includes the integration of more specialized fitting algorithms (such as [46]) as plugins into the SUMO Toolbox so as to ensure maximum accuracy and interpretability of models for electronic devices and systems.

VI. Download Instructions

The SUMO Toolbox framework which was used for the tests (including all algorithms and features discussed here) is available under an open source license (AGPLv3) for download at http://www.sumo.intec.ugent.be. The software package contains a collection of demos, datasets and examples. Additional documentation is also included and a wiki page with information is available at http://www.sumowiki.intec.ugent.be.

References

[1] A. Forrester, A. Sobester, and A. Keane, *Engineering Design Via Surrogate Modelling: A Practical Guide*. Wiley, 2008.

[2] D. P. Solomatine and A. Ostfeld, "Data-driven modelling : some past experiences and new approaches," *Journal of hydroinformatics*, vol. 10, no. 1, pp. 3–22, 2008.

[3] H. Fang, M. Rais-Rohani, Z. Liu, and M. Horstemeyer, "A comparative study of metamodeling methods for multiobjective crashworthiness optimization," *Computers and Structures*, vol. 83, no. 25-26, pp. 2121–2136, 2005.

[4] A. Chinea, *Passive Macromodeling of Electrically Long Interconnects.* PhD thesis, Dept. of Electrical Engineering, Polytechnic University of Turin, Turin, Italy, 2010.

[5] M. Bakr, J. Bandler, K. Madsen, and J. Sondergaard, "Review of the space-mapping approach to engineering optimization and modeling," *Optimization and Engineering*, vol. 1, no. 3, pp. 241–276, 2000.

[6] J. Bandler, R. M. Biernacki, S. H. Chen, P. A. Grobelny, and R. H. Hemmers, "Space mapping technique for electromagnetic optimization," *IEEE Transactions on Microwave Theory and Techniques*, vol. 42, pp. 2536–2544, Aug. 1994.

[7] J. De Geest, T. Dhaene, N. Faché, and D. De Zutter, "Adaptive CAD-model building algorithm for general planar microwave structures," *IEEE Transactions on Microwave Theory and Techniques*, vol. 47, pp. 1801–1809, Sep. 1999.

[8] D. Deschrijver and T. Dhaene, "Stability and passivity enforcement of parametric macromodels in time and frequency domain," *IEEE Transactions on Microwave Theory and Techniques*, vol. 56, pp. 2435–2441, Nov. 2008.

[9] T. Dhaene and D. Deschrijver, "Generalised vector fitting algorithm for macromodelling of passive electronic components," *IEE Electronics Letters*, vol. 41, no. 6, pp. 299–300, 2005.

[10] J. Sacks, W. J. Welch, T. Mitchell, and H. P. Wynn, "Design and analysis of computer experiments," *Statistical science*, vol. 4, no. 4, pp. 409–435, 1989.

[11] E. S. Siah, M. Sasena, J. L. Volakis, P. Y. Papalambros, and R. W. Wiese, "Fast parameter optimization of large-scale electromagnetic objects using direct with kriging metamodeling," *IEEE Transactions on Microwave Theory and Techniques*, vol. 52, no. 1, pp. 276–285, 2004.

[12] Q. J. Zhang and K. C. Gupta, *Neural Networks for RF and Microwave Design (Book + Neuromodeler Disk)*. Norwood, MA, USA: Artech House, Inc., 2000.

[13] A. H. Zaabab, Q. J. Zhang, and M. Nakhla, "A neural network modeling approach to circuit optimization and statistical design," *IEEE Transactions on Microwave Theory and Techniques*, vol. 43, pp. 1349–1358, Jun. 1995.

[14] J. Suykens, T. V. Gestel, J. D. Brabanter, B. D. Moor, and J. Vandewalle, *Least Squares Support Vector Machines*. Singapore: World Scientific Publishing Co., Pte, Ltd., 2002.

[15] D. Gorissen, K. Crombecq, I. Couckuyt, P. Demeester, and T. Dhaene, "A surrogate modeling and adaptive sampling toolbox for computer based design," *Journal of Machine Learning Research*, vol. 11, pp. 2051–2055, 2010.

[16] D. Busby, C. L. Farmer, and A. Iske, "Hierarchical nonlinear approximation for experimental design and statistical data fitting," *SIAM Journal on Scientific Computing*, vol. 29, pp. 49–69, Jan. 2007.

[17] C. Cherkassky and F. M. Mulier, *Learning from Data: Concepts, Theory, and Methods*. Wiley-IEEE, 2007.

[18] P. Triverio, S. Grivet-Talocia, M. Nakhla, F. Canavero, and R. Achar, "Stability, causality and passivity in electrical interconnect models," *IEEE Transactions on Advanced Packaging*, vol. 30, pp. 795–808, Nov. 2007.

[19] T. Dhaene and D. Deschrijver, "Efficient algorithm for passivity enforcement of s-parameter based macromodels," *IEEE Transactions on Microwave Theory and Techniques*, vol. 57, pp. 415–420, Feb. 2009.

[20] D. Deschrijver and T. Dhaene, "Fast passivity enforcement of s-parameter macromodels by pole perturbation," *IEEE Transactions on Microwave Theory and Techniques*, vol. 57, pp. 620–626, Feb. 2009.

[21] J. P. C. Kleijnen, *DASE : Design and Analysis of Simulation Experiments*. Springer, May 2007.

[22] M. Ding and R. Vemur, "An active learning scheme using support vector machines for analog circuit feasibility classification," in *18th International Conference on VLSI Design*, pp. 528–534, Jan. 2005.

[23] T. G. Robertazzi and S. C. Schwartz, "An accelerated sequential algorithm for producing D-optimal designs," *Siam Journal on scientific Computing*, vol. 10, pp. 341–358, Mar. 1989.

[24] Y. Lin, *An Efficient Robust Concept Exploration Method and Sequential Exploratory Experimental Design*. PhD thesis, Georgia Institute of Technology, 2004.

[25] A. C. Keys and L. P. Rees, "A sequential-design metamodeling strategy for simulation optimization," *Computers and Operational Research*, vol. 31, no. 11, pp. 1911–1932, 2004.

[26] L. Zhang, Y. Cao, S. Wan, H. Kabir, and Q.-J. Zhang, "Parallel automatic model generation technique for microwave modeling," in *IEEE/MTT-S International Microwave Symposium*, pp. 103–106, Jun. 2007.

[27] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM Algorithm," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 39, no. 1, pp. 1–38, 1977.

[28] D. Gorissen, T. Dhaene, and F. De Turck, "Evolutionary model type selection for global surrogate modeling," *Journal of Machine Learning Research*, vol. 10, pp. 2039–2078, 2009.

[29] D. Gorissen, L. De Tommasi, K. Crombecq, and T. Dhaene, "Sequential modeling of a low noise amplifier with neural networks and active learning," *Neural Computing and Applications*, vol. 18, pp. 485–494, Jun. 2009.

[30] D. Gorissen, "Grid-enabled adaptive metamodeling and active learning for computer based design," in *Proceedings of The 22nd Canadian Conference on Artificial Intelligence (AI 2009), Kelowna*, vol. LNCS 5549 of *Lecture Notes in Artificial Intelligence*, pp. 266–269, May 2009.

[31] E. R. Harold and W. S. Means, *XML in a Nutshell*. O'Reilly Media, Inc., Cambridge, Massachusetts, 2004.

[32] D. Gorissen, I. Couckuyt, E. Laermans, and T. Dhaene, "Multiobjective global surrogate modeling,dealing with the 5-percent problem," *Engineering with Computers*, vol. 26, pp. 81–89, Jan. 2010.

[33] D. Gorissen, T. Dhaene, P. Demeester, and J. Broeckhove, *Handbook of Research on Grid Technologies and Utility Computing: Concepts for Managing Large-Scale Applications*, ch. Grid enabled surrogate modeling, pp. 249–258. IGI Global, May 2009.

[34] I. Couckuyt, D. Gorissen, H. Rouhani, E. Laermans, and T. Dhaene, "Evolutionary regression modeling with active learning: An application to rainfall runoff modeling," in *International Conference on Adaptive and Natural Computing Algorithms*, vol. LNCS 5495, pp. 548–558, Sep. 2009.

[35] T. Zhu and P. D. Franzon, "Application of surrogate modeling to generate compact and PVT-sensitive IBIS models," in *Proceedings of the 18th Conference on Electrical Performance of Electronic Packaging and Systems (EPEPS)*, Oct. 2009.

[36] D. Gorissen, K. Crombecq, I. Couckuyt, and T. Dhaene, *Foundations of Computational Intelligence, Volume 1: Learning and Approximation: Theoretical Foundations and Applications*, vol. 201, ch. Automatic Approximation of Expensive Functions with Active Learning, pp. 35–62. Springer Verlag, Series Studies in Computational Intelligence, 2009.

[37] D. R. Jones, M. Schonlau, and W. J. Welch, "Efficient global optimization of expensive black-box functions," *Journal of Global Optimization*, vol. 13, pp. 455–492, Nov. 1998.

[38] W. Hendrickx and T. Dhaene, "Sequential design and rational metamodelling," in *Proceedings of the 2005 Winter Simulation Conference* (M. Kuhl, S. N. M., F. B. Armstrong, and J. A. Joines, eds.), pp. 290–298, Dec. 2005.

[39] A. Krogh and J. Vedelsby, "Neural network ensembles, cross validation, and active learning," in *Advances in Neural Information Processing Systems 7, NIPS Conference, Denver, CO*, pp. 231–238, Dec. 1994.

[40] D. MacKay, "Bayesian model comparison and backprop nets," in *Advances in Neural Information Processing Systems 4* (J. E. Moody, S. J. Hanson, and R. P. Lippmann, eds.), pp. 839–846, Morgan Kaufmann, Dec. 1992.

[41] F. Foresee and M. Hagan, "Gauss-Newton approximation to bayesian regularization," in *Proceedings of the 1997 International Joint Conference on Neural Networks*, pp. 1930–1935, Jun. 1997.

[42] H. Nguyen, I. Couckuyt, L. Knockaert, T. Dhaene, D. Gorissen, and Y. Saeys, "An alternative approach to avoid overfitting for surrogate models," in *Winter Simulation Conference*, pp. 2760–2771, Dec. 2011.

[43] K. Crombecq, D. Gorissen, D. Deschrijver, and T. Dhaene, "A novel hybrid sequential design strategy for global surrogate modeling of computer experiments," *SIAM Journal on Scientific Computing*, vol. 33, pp. 1948–1974, Jul. 2011.

[44] B. W. Silverman, *Density Estimation for Statistics and Data Analysis*. Chapman & Hall/CRC, Apr. 1986.

[45] T. Lee, *The Design of CMOS Radio-Frequency Integrated Circuits (Second Edition)*. Cambridge University Press, 2003.

[46] D. Deschrijver, T. Dhaene, and D. De Zutter, "Robust parametric macromodeling using multivariate orthonormal vector fitting," *IEEE Transactions on Microwave Theory and Techniques*, vol. 56, pp. 1661–1667, Jul. 2008.

Dirk Gorissen was born on April 21, 1981. He received his M.Sc. degree in Computer Science from the University of Antwerp in 2004 and a he received a Masters degree in Artificial Intelligence from the Katholieke Universiteit Leuven in 2007. He continued on to the PhD level at the University of Antwerp and later at Gent University, Belgium where he obtained his PhD in Engineering Science in May 2010. During this time he also worked in research labs in Atlanta, USA and Ottawa, Canada, and he was a member of IBBT, an internationally recognized multidisciplinary ICT research center. Starting February 2010 he joined the Computational Engineering and Design Group at School of Engineering Sciences of Southampton University, UK. His research interests lie in the domain of computational engineering. Particular topics of interest include: global and local surrogate modeling for engineering design exploration and optimization, High Performance Computing (HPC), evolutionary computing, machine learning, and software engineering.

Dirk Deschrijver was born in Tielt, Belgium, on September 26, 1981. He received the Master degree (Licentiaat) in computer science and Ph.D. degree from the University of Antwerp, Antwerp, Belgium, in 2003 and 2007, respectively. He was with the Computer Modeling and Simula- tion (COMS) Group, University of Antwerp, where he was supported by a research project of the Fund for Scientific Research Flanders (FWO-Vlaanderen). From May to October 2005, he was a Marie Curie Fellow with the Scientific Computing Group, Eindhoven University of Technology, Eindhoven, The Netherlands. He is currently an FWO Post-Doctoral Research Fellow with the Department of Information Technology (INTEC), Ghent University, Gent, Belgium. His research interests include rational least squares approximation, orthonormal rational functions, system identification, and parametric macromodeling techniques.

Tom Dhaene (M'94-SM'05) received the Ph.D. degree in electrotechnical engineering from Ghent University, Ghent, Belgium, in 1993. From 1989 to 1993, he was a Research Assistant with the Department of Information Technology, Ghent University, where his research focused on different aspects of full-wave electromagnetic circuit modeling, transient simulation, and time-domain characterization of high-frequency and high-speed interconnections. In 1993, he joined the EDA company Alphabit (now part of Agilent). He was one of the key developers of the planar EM simulator ADS Momentum, ADS Model Composer, and ADS Broadband SPICE. Since 2007, he has been a Full Professor with the Department of Information Technology (INTEC), Ghent University - IBBT. He has authored or coauthored more than 250 peer-reviewed papers and abstracts in international conference proceedings, journals, and books. He is the holder of 5 U.S. patents.

Daniel De Zutter (F'00) was born in 1953. He received the M.Sc. degree in electrical engineering from Ghent University, Ghent, Belgium, in 1976. In 1981, and the Ph.D. degree from Ghent University, in 1984. He is currently a Full Professor of electromagnetics. As author or coauthor, he has contributed to more than 180 international journal papers (cited in the Web of Science) and 200 papers in conference proceedings. Between 2004 and 2008, he served as the Dean of the Faculty of Engineering, Ghent University, where he is currently the Head of the Department of Information Technology. His research interest focuses on all aspects of circuit and electromagnetic modeling of high-speed and high-frequency interconnections and packaging, and on electromagnetic compatibility and numerical solutions of Maxwell's equations. Dr. De Zutter was an Associate Editor of the IEEE TRANSACTIONS ON MICROWAVE THEORY AND TECHNIQUES.