

AN ALTERNATIVE APPROACH TO AVOID OVERFITTING FOR SURROGATE MODELS

Huu Minh Nguyen
Ivo Couckuyt
Luc Knockaert
Tom Dhaene

Ghent University - IBBT
Gaston Crommenlaan 8
Gent, 9050, BELGIUM

Dirk Gorissen

University of Southampton
Room 2041, Building 25, Highfield Campus
School of Engineering Sciences
University of Southampton
SO17 1BJ, UK

Yvan Saeys

Ghent University - VIB
Technologiepark 927
Gent, 9052, BELGIUM

ABSTRACT

Surrogate models are data-driven models used to accurately mimic the complex behavior of a system. They are often used to approximate computationally expensive simulation code in order to speed up the exploration of design spaces. A crucial step in the building of surrogate models is finding a good set of hyperparameters, which determine the behavior of the model. This is especially important when dealing with sparse data, as the models are in that case more prone to overfitting. Cross-validation is often used to optimize the hyperparameters of surrogate models, however it is computationally expensive and can still lead to overfitting or other erratic model behavior. This paper introduces a new auxiliary measure for the optimization of the hyperparameters of surrogate models which, when used in conjunction with a cheap accuracy measure, is fast and effective at avoiding unexplained model behavior.

1 INTRODUCTION

In engineering design, many routine tasks such as design space exploration, sensitivity analysis or optimization can quickly become impractical due to the (relative) high cost of computing a single design point. One way of speeding up the design process is by using surrogate models, also referred to as meta-models or response surface models. Surrogate models are data-driven black box models, which accurately approximate the input-output behavior of the real simulator over the entire design space, using as little data as possible. Examples of surrogate models are Artificial Neural Networks (ANN), Support Vector Machines (SVM), Kriging models and Radial Basis Function (RBF) models. Because they can be evaluated almost instantaneously, they can replace, in a first design step, the real simulator in order to quickly explore the design space.

The quality of the surrogate model is in a great deal dependent on the hyperparameters that govern its behavior. High quality models are not only able to accurately approximate the data samples that were used to construct it, but also to accurately predict the values of unseen data points. Optimizing these hyperparameters, however, is a non-trivial problem, since this requires an accurate estimation of the model quality. Finding a good set of hyperparameters becomes even more difficult when dealing with sparse

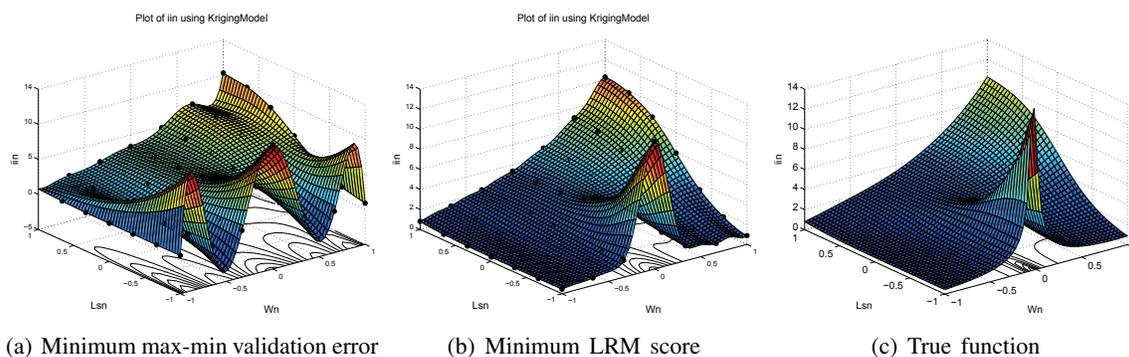


Figure 1: Surrogate models of the input noise current ($\sqrt{i_{in}^2}$) of a Low Noise Amplifier generated with different model selection criteria. The dots represent a sparse intermediate training during model construction (7×7 samples).

data, as the models are more prone to overfitting in such cases, especially when high in-sample accuracy is required. Overfitted models can for example exhibit artificial ripples and bumps while in reality the true response is smooth and there is no data to support this highly nonlinear behavior. This is illustrated in Figure 1. Figure 1(a) shows a surrogate model, built with sparse data, with low in-sample error, but which is unable to capture the true behavior shown in Figure 1(c).

Cross-validation is a common technique to optimize the hyperparameters of surrogate models, especially when little data is available. It is simple to implement and can be applied to any model type. However it can be quite time and resource consuming, especially if the cost of building one model is high (e.g. when working with large ANNs), as each model has to be rebuilt several times for each set of hyperparameters. Moreover models optimized with cross-validation can still exhibit artificial ripples and bumps when the approximating models have high complexity (Gorissen et al. 2009). This paper presents a new generic auxiliary measure model selection measure called the Linear Reference Model (LRM) which is designed to be fast and to avoid spurious model behavior in such cases. The LRM measure identifies regions where the model exhibits complex behavior (such as oscillations) but lacks the data to support this and penalizes the model accordingly. When used in conjunction with a cheap accuracy measure, the LRM measure is able to achieve competitive results with cross-validation at much reduced computational cost. Figure 1(b) show the effect of the LRM measure on the earlier example, where optimizing solely using LRM results in a model with worse in-sample error, but which is able to capture the trend of the true behavior more accurately.

The paper is organized as follows: Section 2 presents related works relevant to the LRM measure. The LRM measure itself is discussed in Section 3. Section 4 applies LRM on a mathematical example and a real-world application. Section 5 concludes with a summary and pointers for future work.

2 RELATED WORK

Several model selection criteria have been discussed in the literature: re-sampling methods such as cross-validation, bootstrap sampling or hold-out sets (Cherkassky and Mulier 2007, Meckesheimer et al. 2002, Bengio and Chapados 2003), information theoretic methods such as the Akaike information criterion, Minimum description length, Bayesian information criterion, etc. (Anderson and Burnham 2003), and methods from statistical learning theory (Vapnik 1995, Cherkassky and Ma 2003, Bartlett et al. 2002).

Most of this work has focused on model complexity measures. These measures penalize the complexity of the model itself, as opposed to the complexity of its behavior. Examples include discrepancy-based measures, Rademacher complexity measures, and Gaussian complexity measures (Bartlett and Mendelson 2003, Bartlett, Bousquet, and Mendelson 2005). Usually these measures are used to build bounds driving

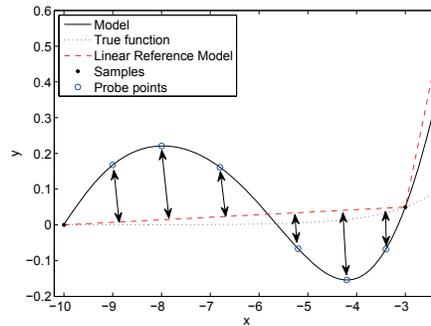


Figure 2: Intuitive 1D illustration of the LRM measure. The goal is to minimize the deviation between the surrogate model and the linear fit. The LRM measure penalizes the models proportionally to the distance between a number of probe points and the linear fit (these distances are denoted by the arrows).

model selection, and may serve as regularization terms in a structural risk minimization approach. An overview of these and other regularization related topics is given in (Cherkassky and Mulier 2007, Chen and Haykin 2002).

Less work has been done on measuring the behavioral of a model directly. In (Girosi et al. 1995) the function smoothness is estimated, based on techniques from signal analysis. Recently, (Koo and Kil 2008) proposes a new model selection measure based on the modulus of continuity of a function (Lorentz 1986) and provide upper bounds for the modulus of continuity for different estimation functions. From Genetic Programming (GP), a new complexity measure is introduced in (Vladislavleva, Smits, and den Hertog 2009) that measures the order of non-linearity of a given GP tree. It is based on the degree of the best fitting polynomial of the symbolic equation represented by the tree. Thus it is a kind of hybrid between structural and observable complexity. However, due to numerical constraints the algorithm does not scale well beyond two dimensions and is thus applied on each sub-tree independently (univariate case only).

In general, though, there is no universally optimal model selection procedure and much depends on the model type and on the amount and distribution of data points (Lin 2004).

3 LRM MEASURE FOR RESPONSE NONLINEARITY ESTIMATION

The LRM algorithm scores a model on the basis of its behavioral complexity, as opposed to its structural complexity. The LRM measure is based on the assumption that, all other factors being equal and if no other information is available, the model with the most parsimonious behavior between any two neighboring points should be chosen. In other words, the model whose behavior is the most linear-like. Any complex behavior (such as oscillations) exhibited by the surrogate model has to be supported by data. Therefore LRM penalizes a surrogate model by giving it a score proportional to its deviation from a local linear interpolation (large deviations indicating high behavioral complexity). Note that although the LRM measure does not penalize the structural complexity explicitly, some correlation might exist.

Optimizing only the LRM measure will usually not result in a good model, since a LRM score of zero corresponds to a local linear interpolation. The LRM measure is therefore only intended as an auxiliary measure to be used along side some (cheap) accuracy measure, pushing the hyperparameters optimization towards models with a more parsimonious behavior. The risk of underfitting due to LRM is usually negligible as the high accuracy typically required of the surrogate models can only be achieved by models with high complexity, in which case the LRM measure will only reduce their tendency to overfit but never to the extent that could cause underfitting.

The deviation from the linear interpolation can be quantified in many ways, for example as the (hyper)volume between the surrogate model and the linear fit. It turns out that a simple estimation of this deviation is sufficient to drive the optimization in the right direction. Thus, in order to limit the

computational burden, the deviation is quantified as the average distance between a number of probe points \mathbf{p}_j of the candidate model and the interpolating hyperplanes (shown in Figure 2). The LRM score for a model with d input values and 1 output value is calculated as follows. Denote the $k \times d$ matrix X and the $k \times 1$ matrix Y , as the respective input and corresponding output variables used to construct $\tilde{f}(\mathbf{x})$, where k is the number of samples. Together, using the block matrix notation, they form the data samples matrix $D = (X|Y)$, given by:

$$D = \left(\begin{array}{ccc|c} x_{1,1} & \dots & x_{1,d} & y_1 \\ \vdots & \ddots & \vdots & \vdots \\ x_{k,1} & \dots & x_{k,d} & y_k \end{array} \right) \quad (1)$$

The first step is to construct a $(d + 1)$ -dimensional local linear interpolation of D . The LRM algorithm starts by constructing a Delaunay tessellation (Hjelle and Daehlen 2006) of the input values X . A graphical illustration of the resulting tessellation of the input space for the 2D case is shown in Figure 3.

Let s_i be a simplex ($i = 1..l \leq k^{\lceil \frac{d}{2} \rceil}$) given by the Delaunay tessellation of X and defined by the input values of data samples $V_i = (\mathbf{v}_1^i, \dots, \mathbf{v}_{d+1}^i)$, where $\mathbf{v}_n^i = (v_{n,1}^i, \dots, v_{n,d}^i, y_{in}^i)$ is one of the data samples in D . The samples V_i are then used to construct a unique interpolating hyperplane for simplex s_i , H_i with following equation:

$$a_{i_1}x_1 + \dots + a_{i_d}x_d + a_{i_{d+1}}y + a_{i_{d+2}} = 0 \quad (2)$$

Next for each simplex s_i a number of probe points are chosen to calculate the deviation. For that purpose the centroid \mathbf{p}_i^c belonging to simplex s_i , which is given by:

$$\mathbf{p}_i^c = \frac{\sum_{j=1}^{d+1} \mathbf{v}_j^i}{d+1} \quad (3)$$

and $d + 1$ inner points \mathbf{p}_j^{ih} , which are given by:

$$\mathbf{p}_j^{ih} = \alpha \mathbf{v}_j^i + (1 - \alpha) \mathbf{p}_i^c, j = 1 \dots d + 1 \quad (4)$$

are chosen. There are many ways to choose α , for example as discussed in (Hammer and Stroud 1956), however in the rest of this paper we keep $\alpha = 0.5$. The probe points are then placed into a matrix P_i (corresponding to simplex s_i) together with their output values given by the surrogate model $\tilde{f}(\mathbf{x})$.

$$P_i = \left(\begin{array}{ccc|c|c} p_{i,1}^c & \dots & p_{i,d}^c & \tilde{f}(\mathbf{p}_i^c) & 1 \\ p_{1,1}^{ih} & \dots & p_{1,d}^{ih} & \tilde{f}(\mathbf{p}_1^{ih}) & 1 \\ \vdots & \ddots & \vdots & \vdots & \vdots \\ p_{d+1,1}^{ih} & \dots & p_{d+1,d}^{ih} & \tilde{f}(\mathbf{p}_{d+1}^{ih}) & 1 \end{array} \right) \quad (5)$$

The LRM score for simplex s_i is then equal to the average perpendicular distance between the hyperplane H_i with coefficients $\mathbf{a}_i = (a_{i_1}, \dots, a_{i_{d+2}})$ and P_i^j , where P_i^j is the j^{th} row of P_i :

$$\mathcal{S}_i = \frac{1}{d+2} \sum_{j=1}^{d+2} \frac{|\mathbf{a}_i \cdot P_i^j|}{\|\mathbf{a}_i\|_2} \quad (6)$$

The scores \mathcal{S}_i for each simplex s_i are then averaged to obtain the overall LRM score for surrogate model $\tilde{f}(\mathbf{x})$.

$$LRM(\tilde{f}(\mathbf{x})) = \frac{1}{l} \sum_{i=1}^l \mathcal{S}_i \quad (7)$$

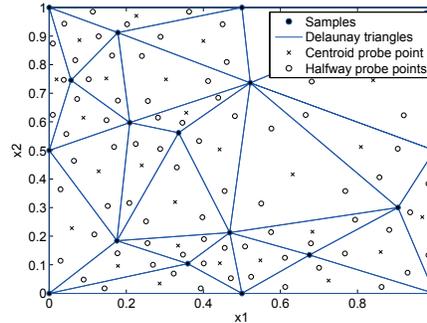


Figure 3: 2D Delaunay tessellation of the input space. The location of the probe points in the input space are denoted by the circles and crosses. The location of the samples are denoted by the black dots.

Models with a large deviation from the local linear fit will have higher LRM scores.

The dominating cost of computing the LRM score is performing the tessellation of X . The implementation used in this paper is based on QHull (Barber, Dobkin, and Huhdanpaa 1996), which has a computational complexity in the order of $O(\frac{k \cdot f_v}{v})$, where k is the number of points, v is the number of output vertexes and f_v is the maximum number of facets for a convex hull of v vertexes. The latter, f_v , is tightly bound to the dimension of the problem and grows rapidly as the number of dimensions increase.

4 APPLICATIONS

4.1 Configuration

Both examples are modeled using the SURrogate MODELing (SUMO) Toolbox (Gorissen et al. 2010). The models are built with an increasing number of samples, which are selected by an adaptive sampling algorithm, called LOLA-Voronoi (Crombecq et al. 2009). LOLA-Voronoi determines the non-linear regions of the true response and samples those more densely. It depends only on the true data and not on the surrogate model. LOLA-Voronoi starts from a small optimized Latin hypercube design augmented with the corner points of the domain. Each sampling iteration LOLA-Voronoi selects a small number of new samples until a pre-determined limit is reached.

ANNs are used as the approximating models. A genetic algorithm (GA) is used to optimize the topology and the initial weights of the ANNs. The GA is run for 10 generations between each sampling iteration with a population size of 10. The networks are trained using Levenberg-Marquardt backpropagation in conjunction with Bayesian regularization (Foresee and Hagan 1997, MacKay 1995) for 300 epochs.

The tests were performed using the LRM measure in conjunction with the in-sample error as the model selection criterion and using 5-fold cross-validation. In addition, the tests were also performed using only the in-sample error as model quality measure to show the influence of LRM. The LRM measure is combined with the in-sample error by taking the average of their output values for these experiments. The total score T is thus given by:

$$T = \frac{1}{2}(LRM + SampleError) \quad (8)$$

The cross-validation score and the in-sample error score are calculated using the Root Relative Square Error (RRSE) which is given by

$$RRSE(y, \tilde{y}) = \sqrt{\frac{\sum_{i=1}^n (y_i - \tilde{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2}} \quad (9)$$

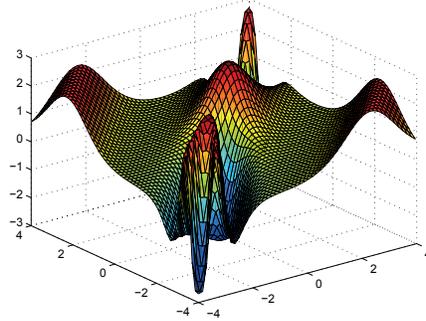


Figure 4: A plot of the bird function.

where $y_i, \tilde{y}_i, \bar{y}$ are the true, predicted and mean true response values respectively. The tests were repeated to smooth out random effects. To estimate the generalization ability of the models, a dense independent grid test set was used.

4.2 The Bird Function

The first example is the modeling of the bird function (Ryu et al. 2002) given by the following formula:

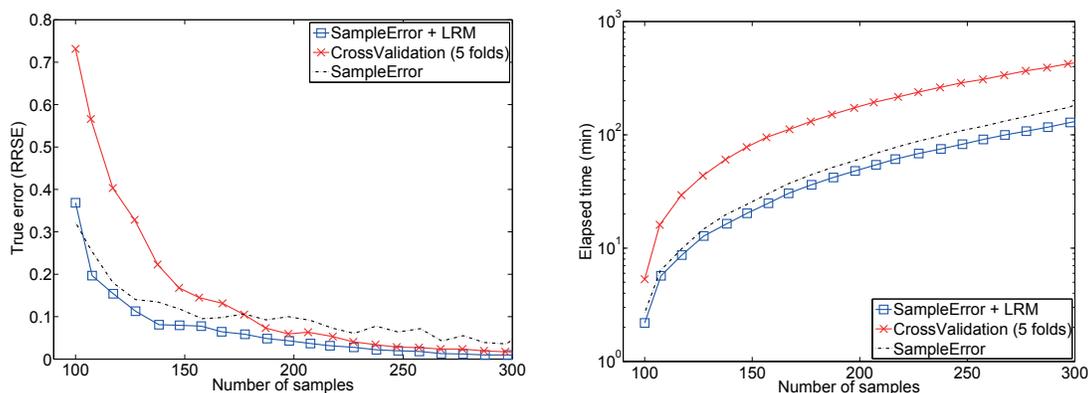
$$\text{bird}(x,y) = \exp(\cos(x-y)) \sin\left(\frac{(x-y)^2 + x + y}{1 + (x-y)^2}\right) \quad (10)$$

with $x, y \in [-4, 4]$. A plot of this function is given in Figure 4. The adaptive sampling algorithm starts from 100 samples and stops once the number of samples is greater than 300. The experiments were repeated 20 times on an Intel Xeon CPU E5504@2.00GHz machine with 6GB total memory. The average results of the different repeat runs are shown in Figure 5. Figure 5(a) shows the error on a dense independent test set (with 401×401 samples) as function of the samples chosen by the adaptive sample selection algorithm. This plot shows that the LRM measure in conjunction with the in-sample error (SampleError+LRM) outperforms cross-validation, especially for smaller number of samples. As the number of samples grows, the difference in accuracy between cross-validation and SampleError+LRM becomes much smaller because the cross-validation estimate of the quality of the model becomes more accurate. SampleError+LRM also achieves lower errors than when using only the in-sample error (SampleError) as a measure for model quality. As the number of samples increases the test error of SampleError remains worse than that of cross-validation and SampleError+LRM.

When looking at the running time for the tests shown in Figure 5(b), it is clear that cross-validation always takes considerably more time to compute, this despite favoring less complex ANNs (Figure 5(c)). SampleError+LRM is also slightly faster SampleError, because of the (slightly) higher complexity of the models favored by SampleError (Figure 5(c)). This also suggests that models chosen using in-sample error are overfitting, since the models are more complex and have worse test errors. SampleError+LRM can therefore achieve better accuracy than the two other model measure and in less time.

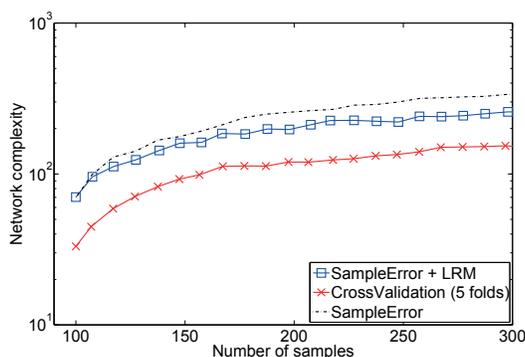
4.3 Low Noise Amplifier (LNA)

The second example is taken from (Gorissen et al. 2009) where the goal was to accurately model the non-linear behavior of a Low Noise Amplifier (LNA) while at the same time minimizing the number of samples used. An LNA is the first stage of a receiver, used to provide the gain needed to suppress the noise of subsequent stages. The resulting surrogate model can then be used for example to find one or more sets of device parameters resulting in a circuit which fulfills the specifications, i.e., constraints given on the performances.



(a) The RRSE error on the independent test set as function of the number of sample.

(b) Modeling run time.



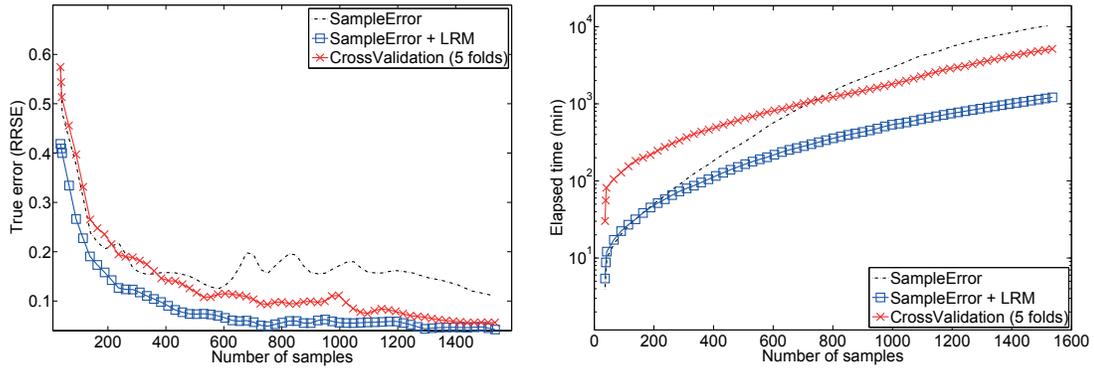
(c) Complexity of the ANNs measured as the number of weights used by the network.

Figure 5: Comparison between cross-validation and SampleError+LRM on the bird example.

The behavior LNA can be calculated in two different ways: by using an analytical small-signal model of the device or by directly simulating the device (we refer to Gorissen et al. (2009) for more details on obtaining the small-signal representations). Subsections 4.3.1 and 4.3.2 will model both approaches respectively using ANNs trained both with the LRM measure (in conjunction with the in-sample error) and cross-validation. The results of using only the in-sample error to guide the model selection process will also be reported. The number of samples is limited to 1500 for the small-signal approach and to 3000 for the direct simulation approach. The experiments for both approaches are repeated 15 times to account for stochastic effects. All tests were run on CalcUA, the cluster available at the University of Antwerp, which consists of 256 Sun Fire V20z nodes (dual AMD Opteron with 4 or 8GB RAM), running SUSE linux, and Matlab 7.7 R2008b.

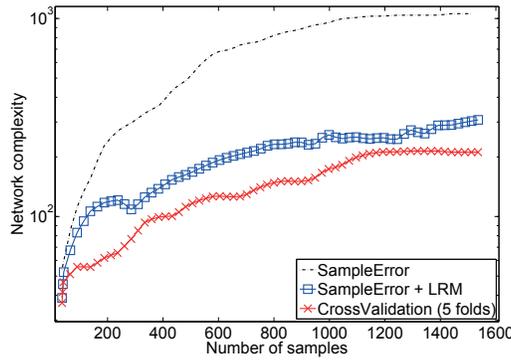
4.3.1 Small-signal Example

The goal here is to build a surrogate model of the input-noise current response variable $\sqrt{i_m^2}$ using the 4D formulation of the LNA problem in (Gorissen et al. 2009). The input parameters are the MOSFET width W , the inductances L_s, L_m (we refer to (Gorissen, De Tommasi, Crombecq, and Dhaene 2009) for further details). Figure 6 summarizes the results of these experiments. Figure 6(a) shows how the true error



(a) Error on the independent test set as function of the number of sample.

(b) Modeling run time.



(c) Complexity of the ANNs measured as the number of weights used by the network.

Figure 6: Comparison between cross-validation and SampleError+LRM for the LNA small-signal example.

decreases when using the LRM measure in conjunction with the in-sample error (SampleError+LRM), in-sample error (SampleError) and cross-validation model selection measures as more data points are added. The true error is calculated as the RRSE (see Eq. 9) on a test set of 11^4 points.

Like in Section 4.2 it can be seen that using SampleError+LRM yields better results than using 5-fold cross-validation for all number of samples and in-sample error (SampleError). The difference between in-sample error and SampleError+LRM however is more pronounced as more samples are added. Figure 6(c) shows the complexity of the ANN models chosen by each model measure. Using SampleError results in more complex models than when using SampleError+LRM and cross-validation, which is likely an indication that those models are overfitting. Figure 6(b) shows that also in this example SampleError+LRM runs faster than the other measures.

4.3.2 Direct Modeling of the LNA

An alternative approach to using a small-signal formulation is to model the performance parameters directly. Note that this is a completely different modeling problem with completely different response behaviors.

The relevant input parameters here are the transistor width W , the transistor length L , the source inductance L_s , the load resistance RL , the voltage bias of the transistor V_{GS} , and the resistance in series

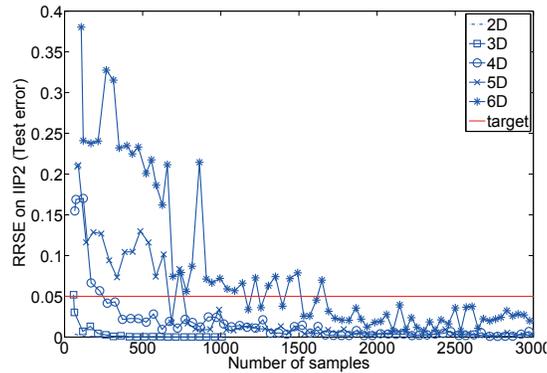


Figure 7: Evolution of the true error when modeling the second order nonlinearity (*IIP2*) of the LNA performance parameters. The LRM measure and the in-sample error are used as the model selection criterion.

with the generator *RS* (the generator series resistance). The second order nonlinearity *IIP2* is taken as the output parameter. The dimensionality is varied from two to six. A comparison of the three model selection criteria (as in the previous section) is not done for computational cost reasons. The goal here is to see if applying the LRM measure can lead to satisfactory ANN models within the sample budget for an increasing number of input variables. Reference test sets of size 51^2 , 15^3 , 11^4 , 7^5 , 5^6 are available for this purpose.

Figure 7 shows the curve for each number of inputs. The curves again depict how good the SampleError+LRM combination is at minimizing the error on the reference grid. Ideally, the graph should display a smooth, monotonic decrease of the error in function of the number of data points. The steeper the descent the better. Erratic jumps should be avoided but temporary increases in error are acceptable, as long as the error continues to decrease globally. The error can temporarily increase if adding new data points reveals new features in the data or a new interpretation, allowing for a more accurate estimation of the model quality.

Satisfactory accuracy can be reached for all number of input dimensions, with particularly fast convergence in the 2D and 3D case. The curves for 5D and 6D much more erratic than the 2D-4D curves. This is confirmed by preliminary results on the other performance parameters (Gorissen, Couckuyt, and Dhaene 2009). The most likely reason for this lies in the fact that for more than 4 dimensions the number of LRM probe points chosen per simplex is too small to allow for an accurate estimation of the deviation. The average simplex volume grows exponentially with the number of dimensions while the number of probe points only grows linearly. Thus these results seem to imply that a different test point distribution is needed in more than four dimensions in order to more accurately guide the hyperparameter optimization.

4.4 Summary

The motivation for the LRM measure stemmed from the need for a fast auxiliary parsimony promoting measure that was just as generic and easy to apply as cross-validation, and that was faster and better at suppressing erratic model behavior in a sparse sample, high accuracy requirement context. The two examples show that the LRM measure is able to achieve this, at least for the ANN models employed. The accuracy curve is comparable or better than that of 5-fold cross-validation with a lower computational cost.

5 CONCLUSION

Surrogate models are commonly used to speed up many routine engineering tasks such as optimization, design space exploration or sensitivity analysis. The hyperparameters of the surrogate model have to be optimized however, if a high quality model is to be obtained.

This work presented a new model selection criterion (LRM) that, when combined with some cheap accuracy criterion, can be used to quickly find accurate surrogate models while at the same time avoiding erratic model behavior. The LRM measure penalizes the surrogate model if there are regions in the design space where it exhibits complex behavior (oscillations) although there is no data to support this. When used in an applications with high risk of overfitting, using the LRM measure leads to equal or better models as when using cross-validation, but at much reduced computational cost.

Various extensions to the basic LRM idea are also possible. For example one could also include the volume of the simplex in the LRM equation and thus give more (or less) weight to certain areas of the model. The same applies to the prediction uncertainty. Also instead of aggregating the LRM measure with some other accuracy measure using a simple average, weighted averages can also be used. Pareto-based optimization can also be used to create a Pareto-optimal front with LRM and the accuracy measure.

The full LRM algorithm is implemented as a plugin in the SUMO Toolbox (Gorissen et al. 2010) which is available for download from <http://www.sumo.intec.ugent.be>.

ACKNOWLEDGMENTS

The authors would like to thank Jeroen Croon and Joost Rommes from the NXP-TSMC Research Center, Device Modeling Department, Eindhoven, The Netherlands, for the LNA simulation code. Ivo Couckuyt and Huu Minh Nguyen are funded by the Institute for the Promotion of Innovation by Science and Technology in Flanders (IWT-Vlaanderen). Yvan Saeys would like to thank the Research Foundation Flanders (FWO-Vlaanderen) for funding his research. We would also like to thank the reviewers for their constructive comments.

REFERENCES

- Anderson, D., and K. Burnham. 2003. *Model Selection and Multi-Model Inference*. Springer.
- Barber, C. B., D. P. Dobkin, and H. Huhdanpaa. 1996, December. "The quickhull algorithm for convex hulls". *ACM Trans. Math. Softw.* 22:469–483.
- Bartlett, P. L., S. Boucheron, and G. Lugosi. 2002. "Model Selection and Error Estimation". *Machine Learning* 48 (1-3): 85–113.
- Bartlett, P. L., O. Bousquet, and S. Mendelson. 2005. "Local Rademacher Complexities". *The Annals of Statistics* 33 (4): 1497–1537.
- Bartlett, P. L., and S. Mendelson. 2003. "Rademacher and gaussian complexities: risk bounds and structural results". *J. Mach. Learn. Res.* 3:463–482.
- Bengio, Y., and N. Chapados. 2003. "Extensions to metric based model selection". *J. Mach. Learn. Res.* 3:1209–1227.
- Chen, Z., and S. Haykin. 2002. "On Different Facets of Regularization Theory". *Neural Computation* 14 (12): 2791–2846.
- Cherkassky, C., and F. M. Mulier. 2007. *Learning from Data: Concepts, Theory, and Methods*. Wiley-IEEE.
- Cherkassky, V., and Y. Ma. 2003. "Comparison of model selection for regression". *Neural Comput.* 15 (7): 1691–1714.
- Crombecq, K., D. Gorissen, L. De Tommasi, and T. Dhaene. 2009, December. "A Novel Sequential Design Strategy for Global Surrogate Modeling". In *Proceedings of the 2009 Winter Simulation Conference*, edited by M. D. Rossetti, R. R. Hill, B. Johansson, A. Dunkin, and R. G. Ingalls, 731–742. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.
- Foresee, F., and M. Hagan. 1997, Jun.. "Gauss-Newton approximation to Bayesian regularization". In *Proceedings of the 1997 International Joint Conference on Neural Networks*, 1930–1935.
- Girosi, F., M. Jones, and T. Poggio. 1995. "Regularization theory and neural networks architectures". *Neural Computation* 7 (2): 219–269.

- Gorissen, D., I. Couckuyt, and T. Dhaene. 2009. “A Linear Reference Model (LRM) Metric for Model Selection and Sequential Design”. Technical report, University of Antwerp.
- Gorissen, D., K. Crombecq, I. Couckuyt, P. Demeester, and T. Dhaene. 2010. “A Surrogate Modeling and Adaptive Sampling Toolbox for Computer Based Design”. *Journal of Machine Learning Research* 11:2051–2055.
- Gorissen, D., L. De Tommasi, K. Crombecq, and T. Dhaene. 2009, Jun.. “Sequential Modeling of a Low Noise Amplifier with Neural Networks and Active Learning”. *Neural Computing and Applications* 18 (5): 485–494.
- Hammer, P. C., and A. H. Stroud. 1956, Jul.. “Numerical Integration Over Simplexes”. *Mathematical Tables and Other Aids to Computation* 10 (55): 137–139.
- Hjelle, O., and M. Daehlen. 2006. *Triangulations and Applications (Mathematics and Visualization)*. Springer.
- Koo, I., and R. M. Kil. 2008. “Model Selection for Regression with Continuous Kernel Functions Using the Modulus of Continuity”. *Journal of Machine Learning Research* 9:2607–2633.
- Lin, Y. 2004. *An Efficient Robust Concept Exploration Method and Sequential Exploratory Experimental Design*. Ph. D. thesis, Georgia Institute of Technology.
- Lorentz, G. 1986. *Approximation of Functions*. Chelsea Publishing Company, New York.
- MacKay, D. J. C. 1995. “Probable Networks and Plausible Predictions — A Review of Practical Bayesian Methods for Supervised Neural Networks”. *Network: Computation in Neural Systems* 6:469–505.
- Meckesheimer, M., A. J. Booker, R. Barton, and T. Simpson. 2002. “Computationally Inexpensive Metamodel Assessment Strategies”. *AIAA Journal* 40 (10): 2053–2060.
- Ryu, J.-S., M.-S. Kim, K.-J. Cha, T. H. Lee, and D.-H. Choi. 2002. “Kriging Interpolation Methods in Geostatistics and DACE Model”. *KSME International Journal* 16 (5): 619–632.
- Vapnik, V. N. 1995. *The nature of statistical learning theory*. New York, NY, USA: Springer-Verlag New York, Inc.
- Vladislavleva, E., G. Smits, and D. den Hertog. 2009, April. “Order of Nonlinearity as a Complexity Measure for Models Generated by Symbolic Regression via Pareto Genetic Programming”. *Evolutionary Computation, IEEE Transactions on* 13 (2): 333–349.

AUTHOR BIOGRAPHIES

HUU MINH NGUYEN received his Erasmus Mundus M.Sc. degree in Photonics from Ghent University (UGent) in 2008. Since April 2009 he is active as a PhD student in the research group Internet Based Communication Networks and Services (IBCN) at Ghent University, working on machine learning problems. His e-mail is huuminh.nguyen@intec.ugent.be.

IVO COUCKUYT received his M.Sc. degree in Computer Science from the University of Antwerp (UA) in 2007. He is active as a PhD student in the research group Internet Based Communication Networks and Services (IBCN) at Ghent University. At the same time he is also affiliated with the Computational Modeling and Programming (COMP) research group (UA). His e-mail is ivo.couckuyt@intec.ugent.be.

DIRK GORISSEN received the M.Sc. degree in computer science from the University of Antwerp, Antwerp, Belgium, in 2004, the Masters degree in artificial intelligence from the Katholieke Universiteit Leuven, Leuven, Belgium, in 2007, and the Ph.D. degree in engineering science from Ghent University, Ghent, Belgium, in 2010. While working toward the doctoral degree, he also worked with research laboratories in Atlanta, GA and Ottawa, ON, Canada, and was a member of the IBBT, an internationally recognized multidisciplinary ICT Research Center. In February 2010, he joined the Computational Engineering and Design Group, School of Engineering Sciences, Southampton University, Southampton, U.K. His research interests lie in the domain of computational engineering. His particular topics of interest include global and local surrogate modeling for engineering design exploration and optimization, high-

performance computing, evolutionary computing, machine learning, and software engineering. His e-mail is dirk.gorissen@soton.ac.uk.

YVAN SAEYS received his diploma of Master in Computer Science from Ghent University in 2000, with a Master thesis entitled “A study and enhancement of the genetic algorithm in the CAM-Brain Machine”. After doing research on digital neural networks at the “Centre for Evolutionary Language Engineering” (CELE) he started working towards a PhD degree in Bioinformatics in the Bioinformatics group of the Department of Plant Systems Biology at Ghent University. His PhD work was entitled “Feature selection for classification of nucleic acid sequences”. Currently he is a postdoctoral researcher in the Bioinformatics group, doing research on applying machine learning algorithms to gene prediction and gene network analysis. His e-mail is yvan.saeys@ugent.be.

LUC KNOCKAERT is a research professor affiliated with the Department of Information Technology (INTEC-IBCN) of the Faculty of Engineering at Ghent University. He received the M. Sc. Degree in physical engineering, the M. Sc. Degree in telecommunications engineering and the Ph. D. Degree in electrical engineering from Ghent University, Belgium, in 1974, 1977 and 1987, respectively. From 1979 to 1984 and from 1988 to 1995 he also acquired a longtime experience in North-South cooperation and development projects at the Universities of the Democratic Republic of the Congo and Burundi. His current research interests include computational electromagnetics, model reduction, machine learning, signal processing, and scientific computing. He has authored or co-authored over 120 papers in international journals and conference proceedings, and he is the holder of 2 U.S. patents. His e-mail is luc.knockaert@intec.ugent.be.

TOM DHAENE is professor in Distributed Scientific Computing within the department of information technology (INTEC-IBCN) of the faculty of Engineering at Ghent University. In 1993, after his Ph.D. on electromagnetic modeling at INTEC, Ghent University, he joined the EDA startup company Aphabet, later acquired by Hewlett-Packard, and now part of Agilent Technologies. In 2000, he went back to academia. His current research interests include machine learning, surrogate modeling, bioinformatics and distributed computing. He has published over 200 papers in international journals and conference proceedings, and he is the holder of 5 U.S. patents. His e-mail is tom.dhaene@intec.ugent.be.