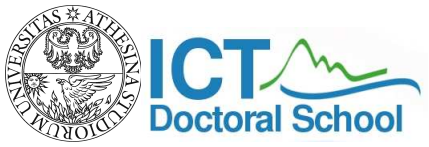
Ph.D. Dissertation



International Doctoral School in Information

and Communication Technology

DISI - University of Trento

An Innovative Learning-by-Example Methodological Strategy for Advanced Reflectarray Antenna Design

Lorenza Tenuti

Tutor:
Paolo Rocca, Professor
University of Trento

October 2018

To my parents Mara and Michele, to my colleagues and friends of the ELEDIA Research Center and to who has supported me during this period. Thank you all.

Abstract

Reflectarray antennas are reflector structures which combine characteristics of both reflector and array antennas. They exhibit electrically large apertures in order to generate significant gain as conventional metallic reflector antennas. At the same time they are populated by several radiating elements which can be controlled individually like conventional phased array antennas. They are usually flat and can be folded and deployed permitting important saving in terms of volume. For these reasons they have been considered since several years for satellite applications. Initially constituted by truncated metallic waveguides and mainly considered for radar applications, they are now mainly constituted by a dielectric substrate, backed by a metallic plane (groundplane) on which microstrip elements with variable shape/size/orientation are printed. These elements are illuminated by the primary feed. The reflected wave from each element has a phase that can be controlled by the geometry of the element itself. By a suitable design of the elements that make up the reflectarray, it is therefore possible to compose the phase front of the reflected waves in the desired direction (steering direction), and to ensure that the obtained overall radiation pattern exhibits a secondary lobe profile which meets the design specifications. Reflectarrays may be used to synthesize pencil or shaped beams. The synthesis methods commonly used to achieve this goal are based on three different steps: (a) calculation of the near field "phase distribution" that the wave reflected by the reflectarray must exhibit to get the desired far-field behaviour; (b) discretization of such distribution into cells of size comparable to that of the elements of interest (i.e., the patches); (c) calculation of the geometry of each elementary cell that will provide the desired reflection coefficient. The first step (a) is a Phase Only approach and permits already to achieve fast preliminary indications on the performance achievable. Accurate results require the implementation of the steps (b) and (c) as well and it is thus of fundamental importance to have techniques capable of efficiently and accurately calculating the reflection coefficient associated with a given geometry of the element [in order to efficiently solve the step (c)]. This coefficient is mathematically represented by a 2x2 complex matrix, which takes into account the relationships between co-polar and cross-polar components of the incident (due to the feed) and reflected field. This matrix naturally depends on the geometry of the element, the direction of incidence of the wave (azimuth and elevation) and the operating frequency of the system. The computation of the reflection coefficient is usually performed using electromagnetic full-wave (FW) simulators; the computation is however time consuming and the generation of the unit cell scattering response database becomes often unfeasible.

In this work, an innovative strategy based on an advanced statistical learning method is introduced to efficiently and accurately predict the electromagnetic re-

sponse of complex-shaped reflectarray elements. The computation of the scattering coefficients of periodic arrangements, characterized by an arbitrary number of degrees-of-freedom, is firstly recast as a vectorial regression problem, then solved with a learning-by-example strategy exploiting the *Ordinary Kriging* paradigm. A set of representative numerical experiments dealing with different element geometries is presented to assess the accuracy, the computational efficiency, and the flexibility of the proposed technique also in comparison with state-of-the-art machine learning methods.

Keywords
Reflectarrays, Scattering Matrix, Computational Electromagnetics, Statistical Learning, Ordinary Kriging.

Published Conference Papers

[C1] M. Salucci, L. Tenuti, C. Nardin, G. Oliveri, F. Viani, P. Rocca, and A. Massa, “Civil engineering applications of ground penetrating radars - Recent advances @ the ELEDIA Research Center,” 2014 European Geoscience Union General Assembly (EGU-GA 2014), Vienna, Austria, April 27 - May 02, 2014.

[C2] M. Salucci, L. Tenuti, C. Nardin, M. Carlin, F. Viani, G. Oliveri, and A. Massa, ”GPR survey thorugh a multi-resolution deterministic approach,” Proc. 2014 IEEE AP-S International Symposium and USNC-URSI Radio Science Meeting, Memphis, Tennessee, USA, pp. 882-883, July 6-12, 2014.

[C3] G. Oliveri, E. Bekele, L. Tenuti, J. Turpin, D. Werner, P. Werner, and A. Massa, “Metamaterial-enhanced arrays by innovative QTCO approaches,” Proc. 2014 IEEE AP-S International Symposium and USNC-URSI Radio Science Meeting, Memphis, Tennessee, USA, pp. 757-758, July 6-12, 2014.

[C4] A. Monti, L. Tenuti, G. Oliveri, A. Alù, A. Massa, A. Toscano, and F. Bilotti, "Design of multi-layer mantle cloaks," 8th International Congress on Advanced Electromagnetic Materials in Microwaves and Optics (Metamaterials 2014), Copenhagen, Denmark, pp. 214-216, August 25-30, 2014.

[C5] T. Moriyama, M. Salucci, G. Oliveri, L. Tenuti, P. Rocca, and A. Massa, ”Multi-scaling deterministic imaging for GPR survey,” Proc. 2014 IEEE Antenna Conference on Antenna Measurements and Applications (IEEE CAMA 2014), Antibes Juan-les-Pins, France, pp. 1-3, November 16-19, 2014.

[C6] G. Oliveri, E. Bekele, M. Carlin, L. Tenuti, J. Turpin, D. H. Werner, and A. Massa, “Extended QCTO for innovative antenna system designs,” Proc. 2014 IEEE Antenna Conference on Antenna Measurements and Applications (IEEE CAMA 2014), Antibes Juan-les-Pins, France, pp. 1-3, November 16-19, 2014.

[C7] M. Donelli, N. Anselmi, G. Oliveri, L. Poli, P. Rocca, L. Tenuti, M. Salucci, and A. Massa, “Imaging and inverse scattering @ ELEDIA Research Center,” Atti XX Riunione Nazionale di Elettromagnetismo (XX RiNEm), Padova, pp. 501-504, 15-18 Settembre 2014.

[C8] M. Carlin, M. Salucci, L. Tenuti, P. Rocca, and A. Massa, ”Efficient radome optimization through the system-by-design methodology,” 9th European Conference on Antennas and Propagation (EUCAP 2015), Lisbon, Portugal, April 12-17, 2015.

[C9] L. Tenuti, M. Salucci, L. Poli, G. Oliveri, and A. Massa, ”Physical-information exploitation in inverse scattering approaches for GPR survey,” 9th European Conference on Antennas and Propagation (EUCAP 2015), Lisbon, Portugal, April 12-17, 2015.

[C10] G Oliveri, M. Salucci, L. Tenuti, P. Calmon, C. Reboud, R. Miorelli, and A. Massa, "SVR-based inversion of eddy current probe impedance data for crack reconstruction within complex structures," Proc. URSI-France 2015 Scientific Days “Probing matter with Electromagnetic Waves”, Paris, France, pp. 241-243, March 24-25, 2015.

[C11] G. Oliveri, M. Salucci, L. Tenuti, and A. Massa, "Enhancing GPR microwave imaging through innovative information acquisition techniques," Proc. URSI-France 2015 Scientific Days ”Probing matter with Electromagnetic Waves”, Paris, France, pp. 225-227, March 24-25, 2015.

[C12] M. Carlin, M. Salucci, L. Tenuti, P. Rocca, F. Viani, and A. Massa, "Complex radome design through the system-by-design approach," Proc. 2015 IEEE AP-S International Symposium and USNC-URSI Radio Science Meeting, Vancouver, BC, Canada, pp. 1324-1325, July 19-25, 2015.

[C13] N. Anselmi, L. Poli, L. Tenuti, P. Rocca, F. Viani, and A. Massa, "Tolerance analysis of planar arrays through Minkowski-based interval analysis," Proc. 2015 IEEE AP-S International Symposium and USNC-URSI Radio Science Meeting, Vancouver, BC, Canada, pp. 2501-2502, July 19-25, 2015.

[C14] L. Tenuti, G. Oliveri, F. Viani, and A. Massa, "Spline-enhanced synthesis of metamaterial lenses for linear array miniaturization by the SbD-QCTO," Proc. 2015 IEEE AP-S International Symposium and USNC-URSI Radio Science Meeting, Vancouver, BC, Canada, pp. 951-952, July 19-25, 2015.

[C15] L. Tenuti, G. Oliveri, F. Viani, F. Bilotti, A. Toscano, and A. Massa, "A system-by-design approach for the synthesis of multi-layer mantle cloaks," Proc. 2015 IEEE AP-S International Symposium and USNC-URSI Radio Science Meeting, Vancouver, BC, Canada, pp. 59-60, July 19-25, 2015.

[C16] M. Salucci, L. Tenuti, L. Poli, G. Oliveri, and A Massa, "A comparative assessment of information-exploitation techniques for GPR data inversion," 5th International Workshop on New Computational Methods for Inverse Problems (NCMIP 2015), Cachan, France, May 29, 2015.

[C17] A. Monti, L. Tenuti, G. Oliveri, J. Soric, A. Alù, A. Massa, A. Toscano, and F. Bilotti, "Recent developments in the design of microwave mantle cloaks with improved performance and relative applications," 9th International Congress on Advanced Electromagnetic Materials in Microwaves and Optics (Metamaterials 2015), Oxford, United Kingdom, pp. 217-219, September 7-12, 2015.

[C18] L. Tenuti, G. Oliveri, F. Viani, A. Polo, M. Donelli, and A. Massa, "A frequency-hopping BCS strategy for imaging buried objects," Proc. 2015 IEEE Mediterranean Microwave Symposium (MMS'2015), Lecce, Italy, pp. 1-4, November 30 - December 2, 2015.

[C19] F. Viani, N. Anselmi, M. Donelli, P. Garofalo, G. Gottardi, G. Oliveri, L. Poli, A. Polo, P. Rocca, M. Salucci, L. Tenuti, and A. Massa, "On the role of information in inversion and synthesis: challenges, tools, and trends," Proc. 2015 IEEE Mediterranean Microwave Symposium (MMS'2015), Lecce, Italy, pp. 1-4, Novembre 30 - December 2, 2015.

[C20] L. Tenuti, M. Salucci, G. Oliveri, P. Rocca, and A. Massa, "Surrogate-assisted optimization of metamaterial devices for advanced antenna Systems," Proc. 2015 IEEE Symposium Series on Computational Intelligence (IEEE SSCI 2015), Cape Town, South Africa, pp. 1154-1156, December 8-10, 2015.

[C21] G. Oliveri, L. Tenuti, M. Salucci, and A. Massa, "Innovative antenna architectures exploiting metamaterials for new generation radars," 10th European Conference on Antennas and Propagation (EUCAP 2016), Davos, Switzerland, pp. 1-3, April 11-15, 2016.

[C22] L. Tenuti, G. Oliveri, A. Monti, F. Bilotti, A. Toscano, and A. Massa, "Design of mantle cloaks through a system-by-design approach," 10th European Conference on Antennas and Propagation (EUCAP 2016), Davos, Switzerland, pp. 1-3, April 11-15, 2016.

[C23] L. Tenuti, G. Oliveri, D. Bresciani, and A. Massa, “Fast design of next generation reflectarrays through advanced LBE strategies,” 10th European Conference on Antennas and Propagation (EUCAP 2016), Davos, Switzerland, pp. 1-4, April 11-15, 2016.

[C24] A. Massa, G. Oliveri, N. Anselmi, L. Poli, and L. Tenuti, "CS-based computational imaging at microwave frequencies," Proc. 2016 IEEE AP-S International Symposium and USNC-URSI Radio Science Meeting, Fajardo, Puerto Rico, pp. 1061-1062, June 26 - July 1, 2016.

[C25] L. Tenuti, G. Oliveri, D. Bresciani, and A. Massa, "Synthesis of next generation reflectarrays," Proc. 2016 IEEE AP-S International Symposium and USNC-URSI Radio Science Meeting, Fajardo, Puerto Rico, pp. 1417-1418, June 26 - July 1, 2016.

[C26] G. Oliveri, E. Bekele, M. Salucci, L. Tenuti, G. Gottardi, T. Moriyama, T. Takenaka, F. Bilotti, A. Toscano, A. Massa, “A system-by-sesign approach to the synthesis of mantle cloaks for large dielectric cylinders,” PIERS 2016 in Shangai, Shangai, China, August 8-11, pp. 3144-3145, 2016.

[C27] P. Rocca, G. Oliveri, L. Tenuti, M. Salucci, T. Moriyama, T. Takenaka, A. Massa, “Imaging complex targets through alphabet-based compressive sensing,” PIERS 2016 in Shangai, Shangai, China, August 8-11, pp. 943-944, 2016 .

[C28] M. Salucci, L. Tenuti, L. Poli, G. Oliveri, and A. Massa, "A computational method for the inversion of wide-band GPR measurements," 6th International Workshop on New Computational Methods for Inverse Problems (NCMIP 2016), Cachan, France, May 20, 2016.

[C29] N. Anselmi, M. Donelli, M. A. Hannan, G. Oliveri, L. Poli, P. Rocca, M. Salucci, L. Tenuti, and A. Massa, “Inverse scattering methodologies and applications @ ELEDIA Research Center,” Atti XXI Riunione Nazionale di Elettromagnetismo (XXI RiNEm), 12-14 Settembre 2016.

[C30] N. Anselmi, M. Donelli, A. Gelmini, G. Gottardi, G. Oliveri, L. Poli, P. Rocca, L. Tenuti, and A. Massa, “Design and optimization of advanced radar and communications systems and architectures @ ELEDIA Research Center,” Atti XXI Riunione Nazionale di Elettromagnetismo (XXI RiNEm), Parma, 12-14 Settembre 2016.

[C31] A. Massa, N. Anselmi, G. Gottardi, G. Oliveri, L. Poli, P. Rocca, M. Salucci, and L. Tenuti, “Unconventional techniques for the synthesis

of modern antenna arrays," 11th European Conference on Antennas and Propagation (EUCAP 2017), Paris, France, pp. 2843-2845, March 19-24, 2017.

[C32] M. Salucci, L. Tenuti, L. Poli, and A. Massa, "Buried-object detection and imaging through innovative processing of GPR data," 11th European Conference on Antennas and Propagation (EUCAP 2017), Paris, France, pp. 1703-1706, March 19-24, 2017.

[C33] L. Tenuti, G. Oliveri, D. Bresciani, and A. Massa, "Advanced learning-based approaches for reflectarrays design," 11th European Conference on Antennas and Propagation (EUCAP 2017), Paris, France, pp. 84-87, March 19-24, 2017.

[C34] L. Tenuti, P. Rocca, and A. Massa, "Optimization-based design of innovative grating-lobe free radar UWB architectures," 2017 International Applied Computational Electromagnetics Society Symposium, ACES 2017, Firenze, Italy, pp. 1-2, March 26-30, 2017.

[C35] M. Salucci, L. Tenuti, G. Oliveri, and A. Massa, "Frequency-hopping GPR prospecting of sparse scatterers through Bayesian compressive sensing," 2017 International Applied Computational Electromagnetics Society Symposium, ACES 2017, Firenze, Italy, pp. 1-2, March 26-30, 2017.

[C36] L. Tenuti, P. Rocca, M. Salucci, G. Gottardi, and A. Massa "Innovative optimization-based design of UWB planar arrays for grating lobes reduction," Proc. 2017 IEEE AP-S International Symposium and USNC-URSI Radio Science Meeting, San Diego, California, USA, July 9-15, 2017.

Published Journals Papers

[R1] G. Oliveri, L. Tenuti, E. Bekele, M. Carlin, and A. Massa," An SbD-QCTO approach to the synthesis of isotropic metamaterial lenses," IEEE Antennas and Wireless Propagation Letters - Special Cluster on 'Transformation Electromagnetics,' Invited Paper, vol. 13, pp. 1783-1786, 2014.

[R2] M. Salucci, L. Tenuti, L. Poli, G. Oliveri, and A. Massa, "A computational method for the inversion of wide-band GPR measurements," Journal of Physics, vol. 756, pp. 1-6, 2016.

[R3] L. Tenuti, N. Anselmi, P. Rocca, M. Salucci, and A. Massa, "Minkowski sum method for planar arrays sensitivity analysis with uncertain-but-bounded excitation tolerances," IEEE Transactions on Antennas and Propagation, vol. 65, pp. 167-177, January 2017.

[R4] L. Tenuti, M. Salucci, G. Oliveri, and A. Massa, "Efficient prediction of the EM response of reflectarray antennas by an advanced statistical learning method," IEEE Transactions on Antennas and Propagation, (under review).

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In the recent years, reflectarrays have emerged as a cost-effective and reliable technological solution in many applicative domains - including satellite communications [1][2], radar [3], and IoT [4] - where a radiating system characterized by a low profile, a light weight, a high gain/efficiency, and an accurate control of the beam contour is required [5][6]. Compared to traditional reflector antennas [7], these devices can guarantee several advantages including lower thickness, flat/conformal shapes, increased robustness, and (potentially) reconfigurability [8][9] thanks to the layout consisting of a feed that illuminates a passive array of microstrip patches, which in turn properly focuses/shapes the reflected beam by controlling the (non-uniform) scattering properties of the reflectarray surface [7][10]. However, the synthesis of high-performance reflectarrays is still a very challenging task from both the methodological and the practical viewpoint, even more when wideband operations and/or a careful control of the cross-polarization components of the reflected field are needed [11]. Generally speaking, complex patch shapes are usually adopted to fit these requirements because of the wider set of degrees-of-freedom (DoFs) potentially enabling an enhanced control of the antenna scattering properties [2][12][13][14][15]. Unfortunately, designing a reflectarray featuring complicated element geometries often turns out to be in practice a very challenging task. To determine the optimal shape of each reflectarray element (i.e., setting the DoFs of the reflectarray patches), the relationships between the descriptors of both the unit cell (e.g., geometry/size of the patch metallizations) and the illumination (e.g., the polarization/frequency/angle-of-arrival of the incident field) with the associated scattering coefficients must be known [11][15]. This knowledge is analytically available only for "simple" unit cells [16][17] described by few DoFs. Otherwise, *scattering matrix*-vs-*descriptors* look-up tables (LUTs), which are off-line computed through extensive full-wave (FW) simulations [18][19][20], are usually built [2][14][15][18]. Because of the exponential grow of the number of LUTs entries with the DoFs of the unit cells [20], advanced reflectarray geometries characterized by arbitrary variations of many descriptors are realistically impossible to handle because of the infeasible

generation and storage of the associated *unit cell scattering response* databases (*UCS-DB*s). To overcome these latter issues towards the fulfilment of advanced and more challenging telecommunication standards, an innovative methodology for the quasi- or real-time prediction of the electromagnetic response of complex reflectarray elements is hereinafter introduced. The evaluation of the scattering coefficients of generic reflectarray unit cells (i.e., featuring an arbitrary number of *DoF*s) is firstly re-cast as a regression problem and then solved with a learning-by-example (*LBE*) strategy able to exploit the information provided by a reduced set of *FW* simulations (namely the "examples") performed once and off-line. More specifically, the statistical learning method is based on the *Ordinary Kriging* (*OK*) [21][22][23] here customized to the vectorial problem at hand. Such guidelines and methodological choices have been motivated by the following considerations:

- the scattering features of complex reflectarray unit cells (e.g., the Phoenix unit cells [12][13][24]) are often smoothly dependent on their geometrical features [2][14][15]. Therefore, it is expected that a suitable equivalent *meta-model* may be deduced to reliably predict the scattering features associated to a unit-cell instead of computing and storing a huge *UCS-DB*;

- standard interpolation methods [25] cannot be employed for electromagnetic prediction purposes owing to the highly non-linear nature of the relation between the unit-cell descriptors and the corresponding electromagnetic response [2][14][15];

- among existing state-of-the-art *LBE* strategies, *OK* has emerged as a very competitive prediction tool when high-fidelity/noiseless input/training samples are available [21][22][23].

As for the main innovative contributions of this work, they include (*a*) the introduction of a computationally-efficient strategy for predicting the scattering response of reflectarray elements featuring arbitrarily complex unit cells; (*b*) the development of a numerical tool that, whether integrated within a system-by-design (*SbD*) loop [26], could enable the optimal synthesis of next-generation reflectarray antennas with controlled co- and cross-polar radiation patterns; (*c*) the customization of an advanced *LBE* technique based on the *OK* for the prediction of complex-valued scattering matrices of periodic planar structures, thus useful not only for reflectarrays, but also generalizable to analogous electromagnetic engineering problems (e.g., the analysis of *frequency-selective surfaces* and *metasurfaces*); (*d*) the derivation of practical guidelines (e.g., reference setups for various trade-offs between time saving and prediction accuracy) for an easy and reliable use of such an *OK*-based reflectarray meta-modeling technique.

Thesis outline

The outline of the thesis is as follows. Chapter 2 is dedicated to a deep analysis of some selected Learning-by-Example strategies which have been taken into account for the problem of efficiently computing the electromagnetic response of generic reflectarrays unit cells. More in details, the two-step prediction process is summarized and then a detailed analysis of the training phase and of the prediction phase is carried out. In Chapter 3 the computation of the scattering coefficients of simple rectangular unit cells is described. It is then explained the difficulty of computing the response for unit cells with more complex shapes. The necessity of an innovative methodology for the quasi or real-time prediction of the response of complex reflectarray elements is here emphasized. Chapter 4 describes the proposed methodology: the reflectarray modeling problem is mathematically formulated and the OK-based prediction method is deeply analyzed. Representative numerical results are reported to illustrate the features and to assess the potentialities of the proposed approach. Moreover, comparative analyses on the accuracy and the computational efficiency of this latter versus FW simulation methods and competitive/most-advanced state-of-the-art regression techniques are carried out. Eventually, some conclusions and final remarks follow.

Chapter 2

Learning-by-example (LBE) strategies

2.1 Motivation and objective of learning-by-examples strategies

Learning-by-examples strategies are computer-aided approaches which allow dealing with problems characterized by an uncountable number of features and variabilities.

In the field of EM engineering the computational cost of classic synthesis (or imaging) strategies are mainly linked to the great amount of time required by EM simulators. Full-wave solvers, based for example on the method of moments (MoM), finite element method (FEM) or finite-difference time-domain (FTDT) are time-consuming. As an example, considering evolutionary optimization problems where there is the need to repeat a large number of simulations, the total cost is directly related to the CPU time necessary to evaluate the fitness of a single trial solution. If we denote with

- P: the number of individuals;
- I: the number of iterations performed by the algorithm;
- Δt: the CPU time required to compute the fitness of a single individual/trial solution;

we have that the total cost of a single optimization is given by

$$Cost = P \times I \times \Delta t$$

In this framework LBE methods drastically reduce the computational effort by emulating or predicting the behaviour of the high-fidelity simulations.

Prediction as a "two-step process"

We can summarize the main loop involving a predictor as a substitute of the EM simulator in the following two steps

1. **Training**. This step is related to how select samples which are the most representative of the real function behavior. In other words, the aim is to reduce the number of simulations required to produce an accurate global representation of the function to predict over the whole domain. The training process is composed by three logical phases:

   (a) **Reduction of the Degrees of Freedom ($DoFs$) of the functional space**. Given a functional space of K variables, there is the possibility that the function doesn't depend equally on all the variables (i.e., some of them have more "impact" on the output w.r.t. others). This task is thus devoted to reduce the number of input variables (i.e., the number of $DoFs$) from K to H (with $H < K$).

   (b) **"Exhaustive" representation of the functional space**. Given a functional space of H ($\leq K$) variables, it is necessary to properly select samples in order to build a training set able to collect the most information from the function over all the input space. In addition the minimum number of training samples needed to accurately train the model has to be defined;

   (c) **Prediction Model building**: this step is strongly related to previous step and is aimed at building the surrogate model which will be used to map the input to the output space in order to emulate the behavior of a real system.

   In order to deal with step 1.(a) some of the existing methods for the reduction of the $DoFs$ are:

   - Sammon Mapping;
   - Principal Component Analysis (PCA);
   - Partial Least Squares (PLS);
   - Fisher Linear Discriminant Analysis (LDA);
   - Step-wise Dimension Reduction;
   - Sufficient Dimension Reduction;
   - Non-Linear PCA;

   In order to deal with step 1.(b) the proper method for sampling the functional space has to be chosen. Some known methods are:

   One-Shot sampling strategies

- Uniform Grid Sampling ($GRID$);
- Uniform Random Sampling (RND);
- Latin Hypercube Sampling (LHS);

Iterative adaptive sampling strategies

- $LOLA - Voronoi$ adaptive sampling;
- $MSE - Based$ adaptive sampling;
- $EIGF - Based$ adaptive sampling;

2. **Test or Prediction.** Starting from the training steps, this task is devoted to predict the function in every point of its domain/support. Methods of prediction can be classified into two main classes:

   - **Interpolation techniques**
     - Nearest neighbor interpolator;
     - Linear (Delaunay) interpolator;
     - Polynomial interpolator;
     - Spline interpolator;
   - **Learning-by-Example (LBE) techniques**
     - ANN (Artificial Neural Network);
     - $RBFN$ (Radial Basis Function Network);
     - SVR (Support Vector Regression);
     - GP (Gaussian Process, or "Kriging" when regression is considered);

2.2 Training phase: Reduction of the *DoFs* of the functional space

2.2.1 What is dimensionality reduction?

Main goal

The main goal of dimensionality reduction techniques is to reduce the number of unknowns of a given problem from K to H, with

$$H < K$$

Why should we use dimensionality reduction?

Reducing the number of unknowns has many benefits when dealing with regression:

1. **Reduce the computational complexity** (and consequently the time) of the training and test phase of a given predictor;

2. [Strongly related to point 1] **Enhance the prediction accuracy**. In general, all the prediction techniques work better if the number of unknowns is not too high;

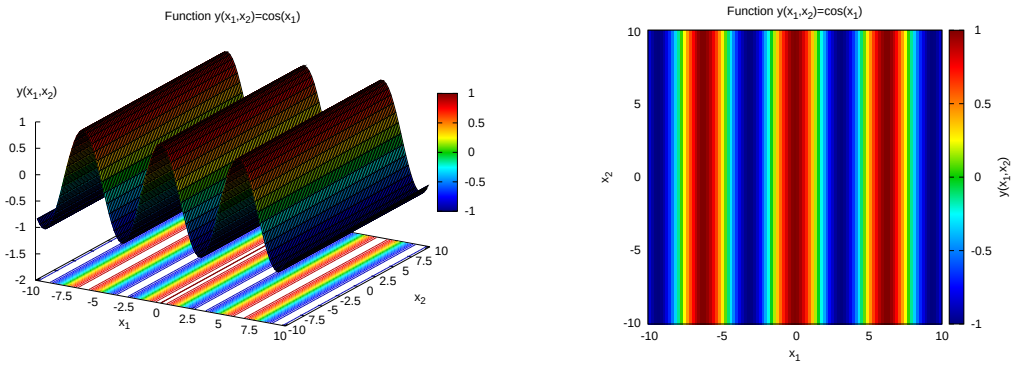
3. **Decrease the number of required training samples (N)**. One of the main problems when training a predictor is the so-called "curse of dimensionality", which causes an exponential growth of the number of training samples (N) required to model a function with the dimension of the input space (K).

It is important to observe that many space reduction techniques have not been introduced as tools for improving prediction. In many cases, space reduction is used to simply improve the way data is visualized, in order to make it more understandable by the interested user.

Where dimensionality reduction makes sense in regression?

This figure shows a particular benchmark function of $K = 2$ variables. It is defined as

$$y(x_1, x_2) = cos(x_1)$$



(*a*) (*b*)

Figure 2.1: Function $y(x_1, x_2) = cos(x_1)$, computed for $\{x_1, x_2\} \in [-10, +10]$. (a) 3D plot and (b) 2D plot.

In this case, variable x_2 has no impact on the output value (i.e., on the function value $y(x_1, x_2)$), since this latter depends only on the values assumed by the variable x_1. By considering this particular example, one should now have a more clear idea of what are the main steps that should be performed by a space reduction technique:

1. **Analyze the relationship between the input variables and the computed output**. In this case, analyze the impact of both x_1 and x_2 on $y(x_1, x_2)$;
2. **Determine what are the input variables showing the largest impact on the output**. In this case, understand that x_2 is meaningless;
3. **Reduce the number of variables**, keeping only the variables that impact on the function value. In this case, x_2 should be discarded.

Variable selection vs. variable extraction

Space reduction techniques can be classified into two main classes:

1. **Variable selection:** it is the process of selecting a subset of relevant variables for use in model construction;
2. **variable extraction:** is the process of creating new variables by combining the the original ones.

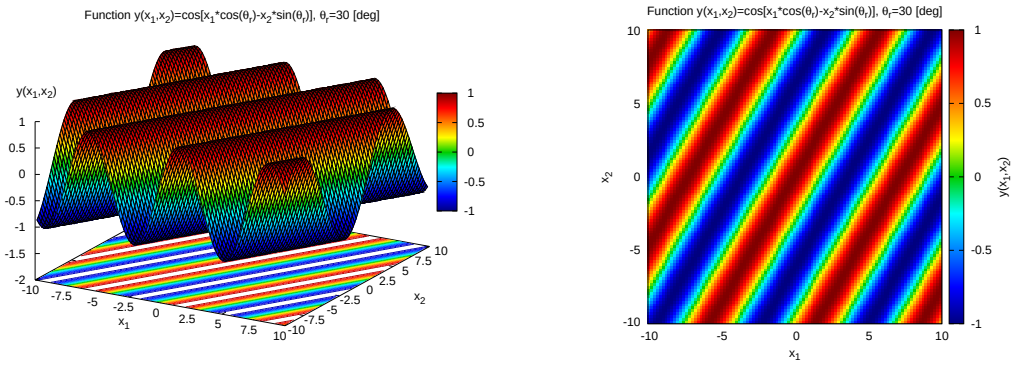
When variable selection and when variable extraction?

Variable selection should be performed when some of the input variables have a smaller impact on the function value with respect to the others. These variables can be discarded, by keeping the H most significant ones.

For example, considering the function reported in Fig. 2.1 the variable x_2 has no impact on the output, so it can be simply discarded. All the information is indeed contained in the value of variable x_1.

On the contrary, variable selection can fail if we consider the same function, but rotated by an angle ϑ_r

$$y\left(x_1, x_2\right) = cos\left[x_1 cos\left(\vartheta_r\right) - x_2 sin\left(\vartheta_r\right)\right]$$



(a) (b)

Figure 2.2: Function $y\left(x_1, x_2\right) = cos\left[x_1 cos\left(\vartheta_r\right) - x_2 sin\left(\vartheta_r\right)\right]$, computed for $\{x_1, x_2\} \in [-10, +10]$. $\vartheta_r = 30$ [deg]. (a) 3D plot and (b) 2D plot.

In this case, the function value depends on a direction which is different from x_1 and x_2 considered singularly. The goal of feature extraction is then to properly identify this direction as a function of x_1 and x_2, so that the function value will be expressed in terms of this new variable.

Function dependent space reduction vs. function independent space reduction

Space reduction techniques can be also classified into

1. **Function dependent ("supervised") techniques:** the reduction of the number of variables takes into account the relationship between input variables and associated function response. Only variables showing the largest impact on the function are considered. State-Of-The-Art techniques belonging to this category are:

   (a) Partial Least Squares (PLS);

2. **Function independent ("unsupervised") techniques:** the reduction of the number of variables is performed without analyzing/knowing the associated output. In other words, these techniques analyze the distribution

of the training samples in the input space, without considering their correlation with the function. State-Of-The-Art techniques belonging to this category are:

(a) Principal Component Analysis (PCA);

(b) Sammon Mapping.

Linear vs. non-linear feature extraction

1. **Linear techniques:** the new H variables are linear combinations of the old K variables. The transformation is expressed by the multiplication of the original N K-dimensional samples $\mathbf{X}$ for a transformation matrix $\mathbf{W}$

$$\mathbf{X}^{new} = \mathbf{X}\mathbf{W}$$
$$[N \times H] = [N \times K]\,[K \times H]$$

2. **Non-linear techniques**: The new H variables are not linear combinations of the old N variables

$$\mathbf{X}^{new} = \Im\{\mathbf{X}\}$$

where the transformation matrix $\Im\{.\}$ is non-linear.

2.3 Training phase: "Exhaustive" representation of the functional space

There exist different sampling strategies in order to build training sets with given dimensions (N). Two main classes can be identified:

1. One-shot sampling strategies;
2. Adaptive (or sequential) sampling strategies.

2.3.1 One-shot sampling strategies: overview

In this section, the following one-shot sampling strategies will be presented:

1. Uniform grid sampling ($GRID$);
2. Uniform random distribution sampling (RND);
3. Latin Hypercube Sampling (LHS);

2.3.1.1 One-shot sampling strategies: Uniform grid sampling ($GRID$)

Parameters:

- Variation range (min, max) for each variable (x_i, $i = 1, ..., K$);
- Number of quantization levels for each variable (Q_i, $i = 1, ..., K$);

The total number of generated samples is given by $N = Q^K$.

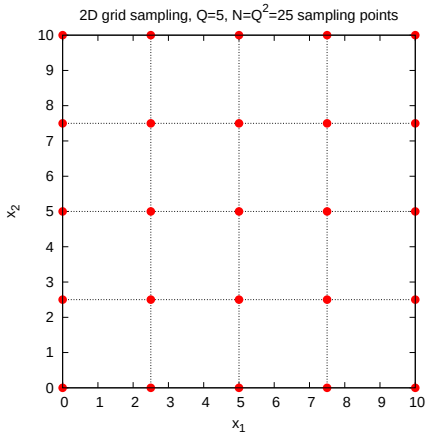


Figure 2.3: Uniform grid sampling for the $2D$ case, $N = 25$ samples. The number of quantization levels is set to $Q = 5$ both for x_1 and for x_2, thus generating a set of $N = Q^K = 5^2 = 25$ samples.

2.3.1.2 One-shot sampling strategies: Uniform random sampling (RND)

Parameters:

- Variation range (min, max) for each variable (x_i, $i = 1, ..., K$);
- Total number of samples to generate (N).

The N samples are selected according to a standard uniform distribution.

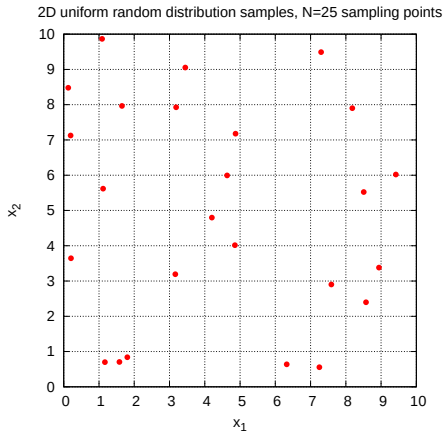


Figure 2.4: Uniform random sampling for the $2D$ case, $N = 25$ samples.

2.3.1.3 One-shot sampling strategies: Latin Hypercube Sampling (LHS)

In the context of statistical sampling, a square grid containing sample positions is a Latin square if (and only if) there is only one sample in each row and each column. A Latin hypercube is the generalization of this concept to an arbitrary number of dimensions, whereby each sample is the only one in each axis-aligned hyperplane containing it.

Parameters:

- Variation range (min, max) for each variable (x_i, $i = 1, ..., K$);
- Total number of samples to generate (N);

Construction:

The N samples in a K-dimensional input space are selected according to these simple steps:

1. Divide the range of each input variable (x_k, $k = 1, ..., K$) into N equally sized segments. Denote with Δ_k the length of each segment in the k-th dimension.

2. For each dimension k $(k = 1, ..., K)$ randomly select a point inside each of the N intervals. This means that on dimension k you will get a set of samples $\mathbf{x}_k = \left\{x_k^1, x_k^2, ..., x_k^N\right\}$, where $(n-1)\,\Delta_k \leq x_k^n \leq n\Delta_k$;

3. Randomly combine a selected point for each dimension $(k = 1, ..., K)$ to generate a new sample $(\mathbf{x}^n = \{x_1^n, x_2^n, ..., x_K^n\})$.

4. Repeat step 3 until all N combinations are generated.

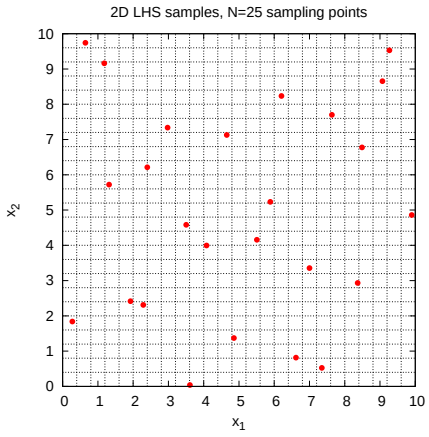


Figure 2.5: LHS sampling for the $2D$ case, $N = 25$ samples.

2.3.2 Adaptive (sequential) sampling strategies: overview

In this section, some state-of-the-art adaptive (or sequential) training techniques will be analyzed. In particular, the following strategies will be considered:

1. LOLA-Voronoi adaptive sampling (**LOLA-Voronoi**);

2. MSE-based adaptive sampling (maximum uncertainty selection criterion, **MSE**);

3. $EIGF$-based adaptive sampling (Expected Improvement For Global Fit, **EIGF**).

Sequential design strategies offer a huge advantage over one-shot experimental designs (such as the Latin Hypercube Sampling, LHS) because they can use information gathered from previous data points in order to determine the location of new data points.

2.3.2.1 Adaptive (sequential) sampling strategies: basic idea

First, an initial batch of data points is evaluated using a minimal experimental design. This design is usually one of the traditional designs from DOE (Design Of Experiments), such as a Latin Hypercube. The initial design must be large enough to guarantee a minimal coverage of the domain, but should be small enough so that there is room for improvement, allowing the sequential design strategy to do its work.

Based on the initial experimental design, a surrogate model is built and the accuracy of this model is estimated using one or more well-known error metrics. Then, the location of some additional samples are chosen by the adaptive sampling strategy. Finally a new surrogate model is built using all the data gathered so far, and the model accuracy is estimated again. If a stop criterion is not met, the entire sample selection process is started all over again. The goal is to reduce the overall number of samples, since evaluating the samples (running the simulations) is the dominant cost in the entire surrogate modeling process.

2.3.2.2 Adaptive (sequential) sampling strategies: $LOLA - Voronoi$ adaptive sampling

LOLA-Voronoi [27] is a novel hybrid sequential design technique that combines an exploration metric based on Voronoi tessellations with an exploitation metric using local linear approximations. Sequential design strategies offer a huge advantage over one-shot experimental designs (such as the Latin Hypercube Sampling, LHS) because they can use information gathered from previous data points in order to determine the location of new data points. The advantage of this method over other sequential design is that it is independent of the model type (Kriging, SVR, etc...). Its main disadvantage is its high computational complexity ($\mathcal{O}(N^2)$, where N is the training set dimension).

Steps:

1. Build an initial training set with N_1 samples using a single shot sampling technique (e.g., Latin Hypercube Sampling, LHS);

2. Analyze the available samples and generate ΔN additional samples by jointly maximizing the exploration and exploitation metrics. For each sample $\mathbf{x}_n$, $n = 1, ..., N_1$, compute the sample score H as:

$$H(\mathbf{x}_n) = V(\mathbf{x}_n) + \frac{E(\mathbf{x}_n)}{\sum_{j=1}^{N_1} E(\mathbf{x}_n)}$$

   where:

   (a) $V(\mathbf{x}_n)$ is the estimated Voronoi cell size associated to sample $\mathbf{x}_n$. This term is related to the **exploration** capability: small values of

$V(\mathbf{x}_n)$ indicate a high density of samples, while high values of $V(\mathbf{x}_n)$ indicate that the region surrounding sample $\mathbf{x}_n$ is characterized by a low density of samples. **New samples should be added where the current samples density is low**.

(b) $E(\mathbf{x}_n)$ is an estimation of how non-linear the function is around sample $\mathbf{x}_n$.

$$E(\mathbf{x}_n) = \sum_{i=1, i \neq n}^{Z} |y(\mathbf{x}_i) - (y(\mathbf{x}_n) + g \cdot (\mathbf{x}_i - \mathbf{x}_n))|$$

Where g is the estimated gradient at point $\mathbf{x}_n$ and $Z < N_1$ is the number of samples closer to $\mathbf{x}_n$ (they are called the "*neighbors*" of $\mathbf{x}_n$). This term is related to the **exploitation** capability: high values of $E(\mathbf{x}_n)$ indicate a high non-linearity of the real function in the region surrounding the sample $\mathbf{x}_n$. **New samples should be added where the function rapidly varies**.

3. Sort the input samples by H;
4. For $j = 1, ..., \Delta N$:
   (a) Select the $j-th$ highly ranked sample $(\mathbf{x}_j)$;
   (b) Generate a set $\{\Omega_j\}$ of R random samples inside the Voronoi cell of $\mathbf{x}_j$: $\{\Omega_j\} = \{\omega_{j,1}; ...; \omega_{j,r}; ...; \omega_{j,R}\}$;
   (c) Select the new j-th sample in $\{\Omega_j\}$ as the farthest sample from $\mathbf{x}_j$: $\mathbf{x}_{N_1+j} = arg\left\{\max_{\{\Omega_j\}}(\|\mathbf{x}_j - \omega_{j,r}\|)\right\}$;
   (d) Compute the function value associated to the new sample $y(\mathbf{x}_{N_1+j})$ and add the pair $\{\mathbf{x}_{N_1+j}; y(\mathbf{x}_{N_1+j})\}$ to the training set for the next LOLA-Voronoi iteration.
5. The new training set will be composed by $N_1 + \Delta N$ samples. Go to step (2) to generate new additional ΔN samples. Iterate until a maximum number of training samples (N_{max}) is reached.

For further details on how the exploration $(V(\mathbf{x}_n))$ and exploitation metrics $(E(\mathbf{x}_n))$ are computed, please refer to [27].

NOTE: the process of generating training sets with increasing dimensions is completely independent from the surrogate model. New samples are in fact added on the basis of the previously observed samples (observations).

Parameters:

- Dimension of the initial training set (N_1);
- Step ΔN: if ΔN is low, more resolution in acquiring information from previous samples but more computational time!

2.3.2.3 Adaptive (sequential) sampling strategies: $MSE-Based$ adaptive sampling

This strategy selects new samples where the highest prediction uncertainty (MSE) is observed.

Steps:

1. Build an initial training set with N_1 samples using a single shot sampling technique (e.g., Latin Hypercube Sampling, LHS) and create a Kriging surrogate model using these training samples;
2. Generate a set of C candidate points using a single shot sampling technique (e.g., Latin Hypercube Sampling, LHS);
3. Use the Kriging model to predict the C candidate solutions, and rank them according to the prediction uncertainty (MSE);
4. Select ΔN candidates showing the highest MSE;
5. Compute the real output of the new selected samples;
6. Add the new samples and their associated output to the original training set in order to generate a new training set. This will be composed by $N_2 = N_1 + \Delta N$ training samples;
7. Train a new surrogate model using the new training set;
8. Go to step (2) to generate additional ΔN samples, iterate until a maximum number of training samples (N_{max}) is reached.

Parameters:

- Dimension of the initial training set (N_1);
- Step ΔN; low ΔN implies more steps (= more computational time) to reach a given training dimension (N);
- Number of candidate points (C). Note that $C \geq \Delta N$. Authors in [28] indicate as a good choice setting the number of candidates $C = 200 \times K$, where K is the problem dimension.

2.3.2.4 Adaptive (sequential) sampling strategies: $EIGF-Based$ adaptive sampling

This strategy selects new samples where the highest Expected Improvement For Global Fit ($EIGF$) is observed. The $EIGF$ for a given point $\mathbf{x}$ is defined as [28, 29]:

$$EIGF(\mathbf{x}) = (\widetilde{y}(\mathbf{x}) - y(\mathbf{x}^*))^2 + MSE\,(\widetilde{y}(\mathbf{x}))$$

where:

- $\widetilde{y}(\mathbf{x})$: predicted output for the point $\mathbf{x}$;
- $MSE\,(\widetilde{y}(\mathbf{x}))$: prediction uncertainty (MSE) associated to the predicted value $\widetilde{y}(\mathbf{x})$;
- $y(\mathbf{x}^*)$: observed (real) output at the sampled point $\mathbf{x}^*$, that is closest in distance to the candidate point $\mathbf{x}$.

The $EIGF$ consists of two search components. The first (local) component will tend to be large at a point where it has the largest (response) increase over its nearest sampled point. The second (global) component is large for points with the largest prediction uncertainty (these tend to be far from existing sampled points).

Steps:

1. Build an initial training set with N_1 samples using a single shot sampling technique (e.g., Latin Hypercube Sampling, LHS) and create a Kriging surrogate model using these training samples;
2. Generate a set of C candidate points using a single shot sampling technique (e.g., Latin Hypercube Sampling, LHS);
3. Use the Kriging model to predict the C candidate solutions, and rank them according to the $EIGF$ metric;
4. Select ΔN candidates showing the highest $EIGF$;
5. Compute the real output of the new selected samples;
6. Add the new samples and their associated output to the original training set in order to generate a new training set. This will be composed by $N_2 = N_1 + \Delta N$ training samples;
7. Train a new surrogate model using the new training set;
8. Go to step (2) to generate additional ΔN samples, iterate until a maximum number of training samples (N_{max}) is reached.

Parameters:

- Dimension of the initial training set (N_1);

- Step ΔN; low ΔN implies more steps (= more computational time) to reach a given training dimension (N);
- Number of candidate points (C). Note that $C \geq \Delta N$. Authors in [28] indicate as a good choice setting the number of candidates $C = 200 \times K$, where K is the problem dimension.

2.3.2.5 Definition of the sampling metric (Λ)

Main idea:

A good sampling technique should have the following characteristics:

- Place **many samples where the function rapidly varies**, and less samples where the function is smooth (exploitation capability);
- **Cover the input space as much as possible** (exploration capability).

The following sampling metric can be defined, in order to measure the ability of a given sampling strategy to respect the above two conditions:

$$\Lambda = \frac{1}{N} \sum_{n=1}^{N} \left\{ \frac{\left[\frac{E(\mathbf{x}_n)}{\sum_{j=1}^{N} E(\mathbf{x}_j)} \right]}{V(\mathbf{x}_n)} \right\}$$

where:

- N is the number of training samples;
- $E(\mathbf{x}_n)$ is the LOLA-Voronoi metric for measuring the non-linearity of the function near the training sample $\mathbf{x}_n$. High values of $E(\mathbf{x}_n)$ indicate an high non-linearity near $\mathbf{x}_n$.
- $V(\mathbf{x}_n)$ is the estimated Voronoi cell size associated to the training sample $\mathbf{x}_n$. Small values of $V(\mathbf{x}_n)$ indicate a dense sampling near $\mathbf{x}_n$.

According to this metric, the following cases are penalized:

- Many samples where the function is smooth;
- Few samples where the function rapidly varies.

2.4 Training phase: Prediction model building

This sep is aimed at building the model that will be used in the test phase. The model is built starting from the training samples available. The problem of building the model can be stated as:

PREDICTION MODEL BUILDING: Given the training set $\mathbf{x}^{(i)} = \left[x_1^{(i)}, x_2^{(i)}, ..., x_K^{(i)}\right], i = 1, ..., N$ and selected the LBE technique define the estimation function $\overbrace{y}(\cdot)$ that better represents the behavior of the real system $y(\mathbf{x})$ for a specific and arbitrary input space $\mathbf{x}$.

Depending on the regression strategy chosen this step varies. In the following the prediciton model building is characterized for each of the learning-by-example techniques analyzed in next section:

- Kriging: estimate the set of hyperparameters used for the calculation of the correlation between the training samples;
- Support Vector Regression: define weights of the discriminant function in order to guarantee that the training samples deviate from the predicted function a maximum quantity ε;
- Radial Basis Funcion Networks: define the weights of the activation function.

A detailed analysis of these techniques is given in next sections.

2.5 Test or Prediction phase: Prediction through interpolation techniques

2.5.1 Nearest neighbor interpolator

Let be given

- $\mathbf{x}^{(i)} \in \Re^K$, $\mathbf{x}^{(i)} = \left\{x_k^{(i)}, k = 1, ..., K\right\}$: i-th training sample point in K-dimensional space, $i = 1, ..., N$;
- $y_i = y\left(\mathbf{x}^{(i)}\right)$: function value (output) associated to the i-th training sample point $\mathbf{x}^{(i)}$;
- $\mathbf{x}^* \in \Re^K$, $\mathbf{x}^* = \{x_k^*, k = 1, ..., K\}$: K-dimensional point at which we are performing the prediction (i.e., the output $y(\mathbf{x}^*)$ is unknown and has to be estimated given the available information from the N collected training samples).

Main idea:

The predicted value at position $\mathbf{x}^*$ is equal to the value assumed by the nearest (in terms of Euclidean distance) training sample.

Steps:

1. Calculate the Euclidean distance between the test sample point $\mathbf{x}^*$ and each i-th training sample as follows:

$$d_i = \left\|\mathbf{x}^{(i)} - \mathbf{x}^*\right\| = d\left(\mathbf{x}^{(i)}, \mathbf{x}^*\right) = \sqrt{\sum_{k=1}^{K}\left(x_k^{(i)} - x_k^*\right)^2}$$

2. The predicted value at position $\mathbf{x}^*$ is equal to the value assumed by the nearest (in terms of Euclidean distance) training sample

$$\tilde{y}(\mathbf{x}^*) = y\left(\arg\min_{\mathbf{x}^{(i)}}\{d_i\}\right)$$

Example: Ackley's function $K = 1$ variables

The following figure shows the predicted output made by a nearest neighbor interpolator when applied to the estimation of the 1-dimensional Ackley's function, when a set of $N = 9$ uniformly-spaced training samples are provided.

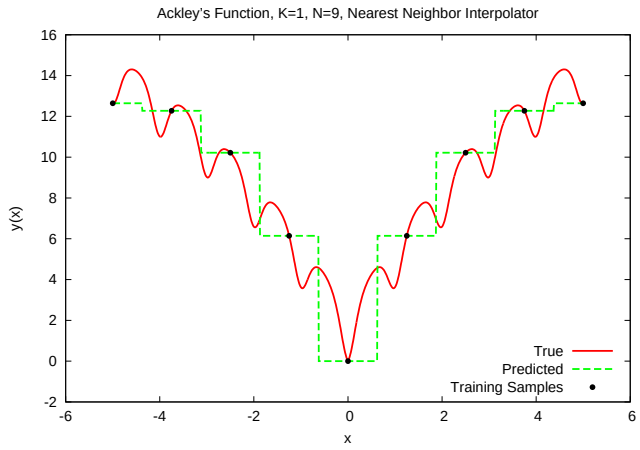
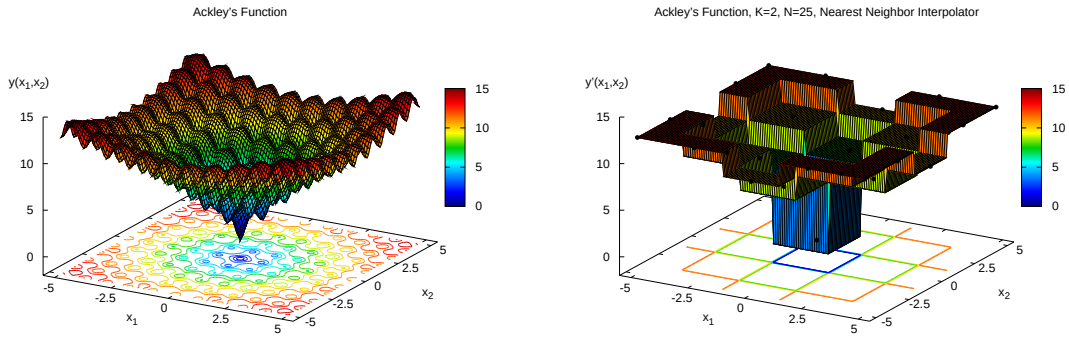


Figure 2.6: Ackley's function, $K = 1$ variables. True function vs prediction made by the nearest neighbor interpolator.

Example: Ackley's function $K = 2$ variables

The following figure shows the predicted output made by a nearest neighbor interpolator when applied to the estimation of the 2-dimensional Ackley's function, when a set of $N = 25$ uniformly-spaced training samples are provided.



(a) (b)

Figure 2.7: Ackley's function, $K = 2$ variables. (a) True function vs (b) prediction made by the nearest neighbor interpolator.

Extrapolation capabilities of the nearest neighbor interpolator

The nearest neighbor interpolator is able to do "extrapolation". This process is related to the capability of estimating the function value even beyond the original observation range. When estimating the function value at a position that lies outside the observed domain (i.e., the K-dimensional region identified

by the set of available training samples), the interpolator will simply use the value of the nearest (in terms of Euclidean distance) training sample.

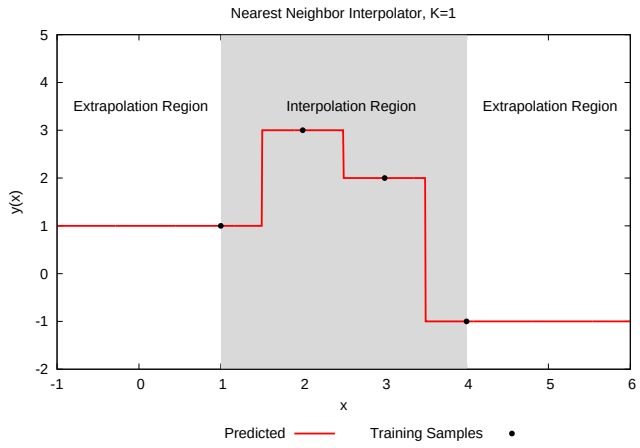


Figure 2.8: Extrapolation capabilities of the nearest neighbor interpolator (example $K = 1$).

2.5.2 Linear interpolation (based on Delaunay triangulation)

Let be given

- $\mathbf{x}^{(i)} \in \Re^K$, $\mathbf{x}^{(i)} = \left\{x_k^{(i)}, k = 1, ..., K\right\}$: i-th training sample point in K-dimensional space, $i = 1, ..., N$;
- $y_i = y\left(\mathbf{x}^{(i)}\right)$: function value (output) associated to the i-th training sample $\mathbf{x}^{(i)}$;
- $\mathbf{x}^* \in \Re^K$, $\mathbf{x}^* = \{x_k^*, k = 1, ..., K\}$: K-dimensional point at which we are performing the prediction (i.e., the output $y\left(\mathbf{x}^*\right)$ is unknown and has to be estimated given the available information from the N collected training samples);
- $\mathbf{X} = \left\{\mathbf{x}^{(i)}, i = 1, ..., N\right\}$: set of N , K-dimensional training sample points;

Main idea:

- In 1-dimensional case (i.e., $K = 1$), the function value at position $\mathbf{x}^* = x^*$ is given by the equation of the straight line passing between the two neighboring points.

- For higher dimensional spaces (i.e., $K > 1$), a more complex formulation is needed, which basically extends the basis idea for the 1-dimensional case. In particular, the choice of the neighboring training samples that should be used to predict the output at position $\mathbf{x}^*$ is performed by means of a Delaunay triangulation of the training samples in the input space. The prediction $\tilde{y}(\mathbf{x}^*)$ is then given by a weighted sum of the function values assumed by the samples in the neighborhood of $\mathbf{x}^*$.

Steps for the creation of the Delaunay graph (triangulation):

1. Let be given a set $\mathbf{X}$ of N K-dimensional training samples (in the following figures we will refer for simplicity to the case $K = 2$)

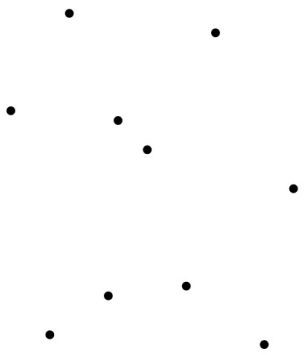


Figure 2.9: Delaunay triangulation: set of N K-dimensional points, $\mathbf{X}$ (i.e., the position of the training samples). Case $K = 2$.

2. Create the **Voronoi Diagram** $Vor(\mathbf{X})$, that is the subdivision of the plane into Voronoi cells $\mathcal{V}\left(\mathbf{x}^{(i)}\right)$ for all $\mathbf{x}^{(i)} \in \mathbf{X}$. The Voronoi cell associated to sample $\mathbf{x}^{(i)}$ is defined as the region of points whose distance (Euclidean) to sample $\mathbf{x}^{(i)}$ is lower than the distance to all other training samples

$$\mathcal{V}\left(\mathbf{x}^{(i)}\right) = \left\{\mathbf{x} | \, d\left(\mathbf{x}, \mathbf{x}^{(i)}\right) \leq d\left(\mathbf{x}, \mathbf{x}^{(j)}\right) \quad \forall i;\, j = 1, ..., N;\, i \neq j\right\}$$

where $d\left(\mathbf{x}, \mathbf{x}^{(i)}\right)$ denotes the Euclidean distance between positions $\mathbf{x}$ and $\mathbf{x}^{(i)}$

$$d\left(\mathbf{x}, \mathbf{x}^{(i)}\right) = \sqrt{\sum_{k=1}^{K} \left(x_k^{(i)} - x_k\right)^2}$$

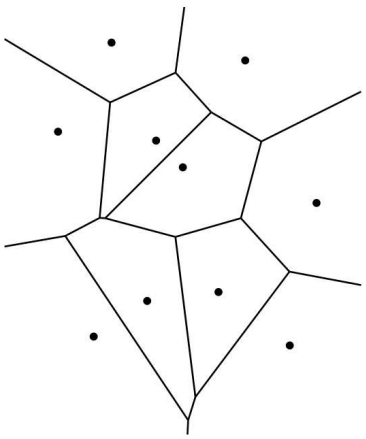


Figure 2.10: Delaunay triangulation: Voronoi Diagram $Vor\left(\mathbf{X}\right)$ of the training samples (case $K = 2$).

3. Create the dual graph of $Vor\left(\mathbf{X}\right)$, $\mathcal{G}\left(\mathbf{X}\right)$ of the Voronoi diagram. To create the dual graph, connect two samples if and only if there exists a path crossing only one Voronoi cell boundary.

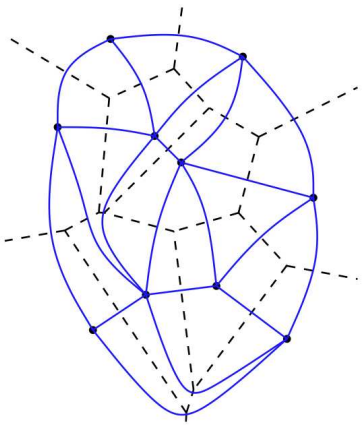


Figure 2.11: Delaunay triangulation: dual graph of the Voronoi graph $Vor\left(\mathbf{X}\right)$, $\mathcal{G}\left(\mathbf{X}\right)$ (case $K = 2$).

4. Create the Delaunay graph, $\mathcal{D}\left\{\mathcal{G}\left(\mathbf{X}\right)\right\}$, converting curved paths to straight lines. The Delaunay diagram (or triangulation) identifies the neighboring points that should be considered for the estimation of the function value at each test location $\mathbf{x}^{*}$.

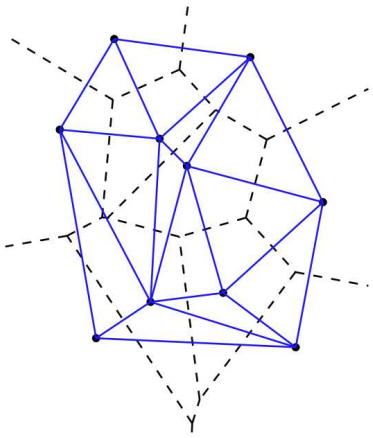


Figure 2.12: Delaunay triangulation: Delaunay graph, $\mathcal{D}\{\mathcal{G}(\mathbf{X})\}$

Predicting the function value at position $\mathbf{x}^*$:

The function at position $\mathbf{x}^*$ is computed as a weighted sum of $K+1$ independent samples belonging to the same K-dimensional "simplex"[1]:

$$\tilde{y}(\mathbf{x}^*) = \sum_{k=1}^{K+1} \alpha_k y_k$$

where

- α_k is the weight associated to the k-th neighboring training sample $\mathbf{x}^{(k)}$;
- $y_k = y\left(\mathbf{x}^{(k)}\right)$ is the function value at training sample $\mathbf{x}^{(k)}$.

The weight α_k associated to training sample $\mathbf{x}^{(k)}$ is computed as the ratio between the "volume" of the simplex including all remaining training samples and the test location $\mathbf{x}^*$ and the "volume" of the simplex containing $\mathbf{x}^*$:

$$\alpha_i = \frac{Vol\left\{\mathbf{x}^{(1)}, ..., \mathbf{x}^{(i-1)}, \mathbf{x}^*, \mathbf{x}^{(i+1)}, ..., \mathbf{x}^{(K+1)}\right\}}{Vol\left\{\mathbf{x}^{(1)}, ..., \mathbf{x}^{(K+1)}\right\}}$$

where the "volume" of a K-dimensional simplex made by $K+1$ points is computed as follows

$$Vol\left\{\mathbf{x}^{(1)}, ..., \mathbf{x}^{(K+1)}\right\} = \det\begin{pmatrix} \mathbf{x}^{(1)} & \cdots & \mathbf{x}^{(K+1)} \\ 1 & \cdots & 1 \end{pmatrix}$$

Note that

$$\sum_{i}^{K+1} \alpha_i = 1$$

Example with $K = 1$

[1] For $K = 1$ the simplex reduces to the straight line connecting two points, for $K = 2$ it corresponds to the Delaunay triangle connecting 3 points, for $K = 3$ it corresponds to a tetrahedron...

1. Let be given two training samples x_i and x_{i+1} and the associated outputs $f_i = y(x_i)$, $f_{i+1} = y(x_{i+1})$.

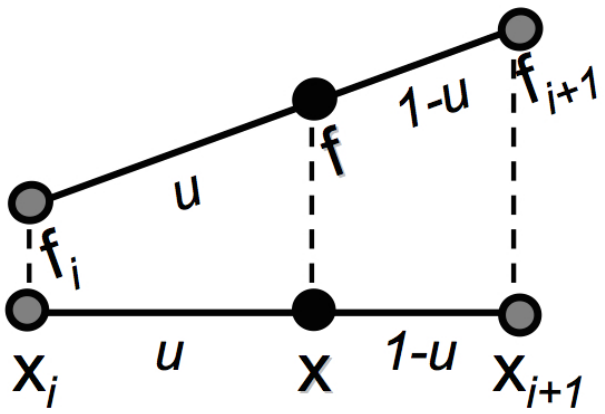


Figure 2.13: Delaunay triangulation: Example with $K = 1$.

2. Compute the "volumes" of the 1D simplex containing the location x at which we are doing the prediction. This corresponds (in absolute value) to the length of the segment between samples x_i and x_{i+1}:

$$Vol\{x_i, x_{i+1}\} = \det \begin{pmatrix} x_i & x_{i+1} \\ 1 & 1 \end{pmatrix} = x_i - x_{i+1}$$

3. Compute the weights associated to training samples x_i and x_{i+1}:

$$\alpha_i = \frac{Vol\{x, x_{i+1}\}}{Vol\{x_i, x_{i+1}\}} = \frac{x - x_{i+1}}{x_i - x_{i+1}}$$

$$\alpha_{i+1} = \frac{Vol\{x, x_i\}}{Vol\{x_i, x_{i+1}\}} = \frac{x - x_i}{x_i - x_{i+1}} = 1 - \alpha_i$$

Then, we have that

$$x = \alpha_i x_i + \alpha_{i+1} x_{i+1} = \alpha_i x_i + (1 - \alpha_i) x_{i+1}$$

and the prediction at point x is given by

$$\tilde{y}(x) = \alpha_i y(x_i) + (1 - \alpha_i) y(x_{i+1})$$

Example with $K = 2$

1. Let be given three training samples $\mathbf{x}^{(1)}$, $\mathbf{x}^{(2)}$ and $\mathbf{x}^{(3)}$ defining the Delaunay triangle for point $\mathbf{x}$ at which we are performing the prediction

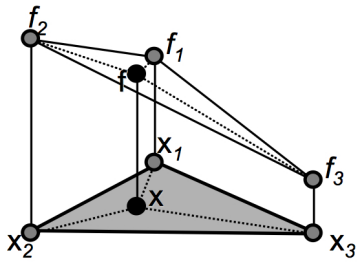


Figure 2.14: Delaunay triangulation: Example with $K = 2$.

2. Compute the "volumes" of the 2D simplex containing the location $\mathbf{x}$ at which we are doing the prediction

$$Vol\left\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \mathbf{x}^{(3)}\right\} = \det \begin{pmatrix} \mathbf{x}^{(1)} & \mathbf{x}^{(2)} & \mathbf{x}^{(3)} \\ 1 & 1 & 1 \end{pmatrix} = \pm 2A\left\{\triangle\left(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \mathbf{x}^{(3)}\right)\right\}$$

3. Compute the weights associated to the three training samples

$$\alpha_1 = \frac{Vol\left\{\mathbf{x}, \mathbf{x}^{(2)}, \mathbf{x}^{(3)}\right\}}{Vol\left\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \mathbf{x}^{(3)}\right\}}$$

$$\alpha_2 = \frac{Vol\left\{\mathbf{x}^{(1)}, \mathbf{x}, \mathbf{x}^{(3)}\right\}}{Vol\left\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \mathbf{x}^{(3)}\right\}}$$

$$\alpha_3 = \frac{Vol\left\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \mathbf{x}\right\}}{Vol\left\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \mathbf{x}^{(3)}\right\}}$$

Then, we have that

$$\mathbf{x} = \alpha_1 \mathbf{x}^{(1)} + \alpha_2 \mathbf{x}^{(2)} + \alpha_3 \mathbf{x}^{(3)}$$

and the prediction at point $\mathbf{x}$ is given by

$$\tilde{y}(\mathbf{x}) = \alpha_1 y\left(\mathbf{x}^{(1)}\right) + \alpha_2 y\left(\mathbf{x}^{(2)}\right) + \alpha_3 y\left(\mathbf{x}^{(3)}\right).$$

Example: Ackley's function $K = 1$ variables

The following figure shows the predicted output made by a linear interpolator when applied to the estimation of the 1-dimensional Ackley's function, when a set of $N = 9$ uniformly-spaced training samples are provided.

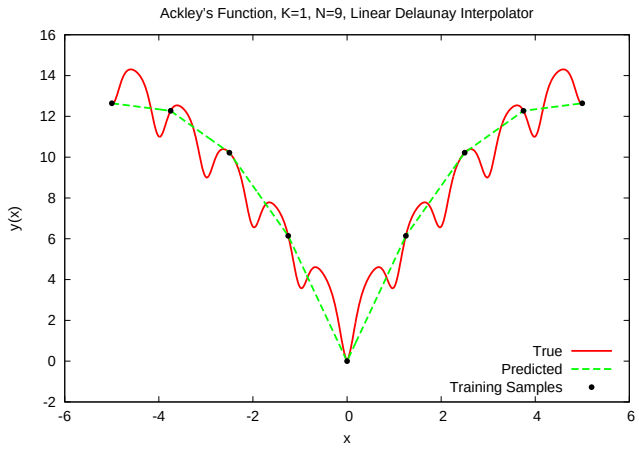
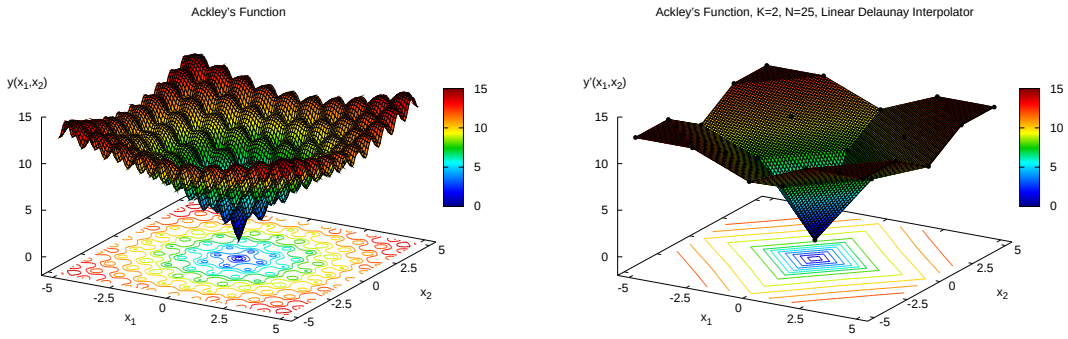


Figure 2.15: Ackley's function, $K = 1$ variables. True function vs prediction made by the linear interpolator.

Example: Ackley's function $K = 2$ variables

The following figure shows the predicted output made by a linear interpolator when applied to the estimation of the 2-dimensional Ackley's function, when a set of $N = 25$ uniformly-spaced training samples are provided.



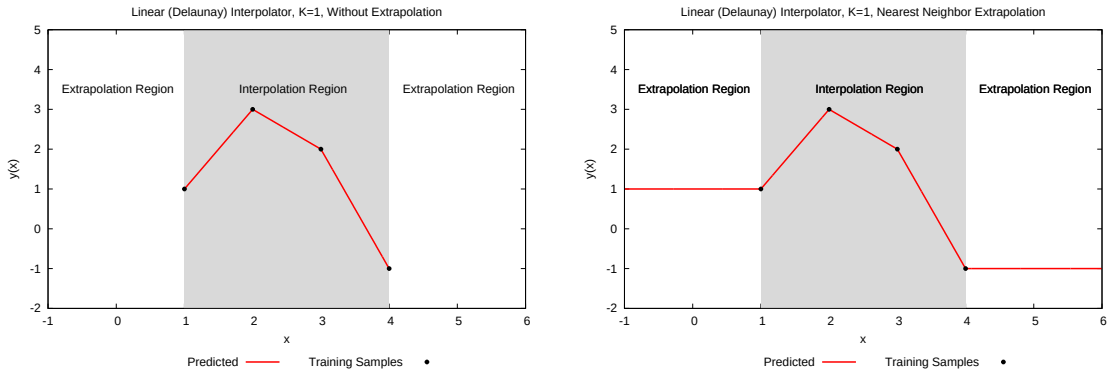
(a) (b)

Figure 2.16: Ackley's function, $K = 2$ variables. (a) True function vs (b) prediction made by the linear interpolator.

Extrapolation capabilities of the nearest neighbor interpolator

The linear Delaunay-based interpolator is not able to do "extrapolation". This process is related to the capability of estimating the function value even beyond the original observation range. In fact, the prediction at a given test position $\mathbf{x}^*$ is possible if and only if there exist a K-dimensional simplex of training samples surrounding it.

Linear interpolator can however be extended with a nearest neighbor interpolator in order to enable extrapolation, as shown by the above figures.



(a) (b)

Figure 2.17: Linear Delaunay interpolator (a) without extrapolation (standard) and (b) with nearest neighbor extrapolation (example $K = 1$).

2.6 Test or Prediction phase: prediction through learning-by-example techniques

2.6.1 Kriging: Gaussian Processes (GP) for regression

Mathematical formulation

The following formulation has been mainly derived by [21] and [30].

Suppose we have evaluated a deterministic function of K variables at N points. Denote the i-th sampled point by

$$\mathbf{x}^{(i)} = \left[x_1^{(i)}, x_2^{(i)}, ..., x_K^{(i)}\right] \tag{2.1}$$

and the associated function value by

$$y^{(i)} = y\left(\mathbf{x}^{(i)}\right) \tag{2.2}$$

for $i = 1, ..., N$.[2]

The classical linear regression model

The simplest and most familiar way to fit a response surface to such a data is linear regression. In this technique, the observations are treated as if they were generated from the following model

$$y\left(\mathbf{x}^{(i)}\right) = \left[\sum_{q=1}^{Q} \alpha_q \beta_q\left(\mathbf{x}^{(i)}\right)\right] + \epsilon^{(i)}, \qquad i = 1, ..., N \tag{2.3}$$

In this equation, each $\beta_q\left(\mathbf{x}^{(i)}\right)$ $(q = 1, ..., Q)$ is a linear or nonlinear function of $\mathbf{x}$, the α_q's $(q = 1, ..., Q)$ are unknown coefficients to be estimated and the $\epsilon^{(i)}$'s are normally distributed, *independent* error terms with mean zero and variance σ^2. The conceptual problem with linear regression is that the assumption of independent errors is clearly false when modeling a deterministic computer code. **If $\mathbf{x}^{(i)}$ and $\mathbf{x}^{(j)}$ are two points that are close together, then the errors terms $\epsilon\left(\mathbf{x}^{(i)}\right)$ and $\epsilon\left(\mathbf{x}^{(j)}\right)$ (i.e., their associated outputs $y\left(\mathbf{x}^{(i)}\right)$ and $y\left(\mathbf{x}^{(j)}\right)$) should also be close (correlated)**. In short, it makes no sense to assume that $\epsilon\left(\mathbf{x}^{(i)}\right)$ and $\epsilon\left(\mathbf{x}^{(j)}\right)$ are independent. Instead, it is more reasonable to assume that these error terms are related or "correlated", and that this correlation is high when $\mathbf{x}^{(i)}$ and $\mathbf{x}^{(j)}$ are close and low when the points are far apart.

In the stochastic process approach, we do not assume that the errors are independent, but rather assume that **the correlation between errors is related to the distance** between the corresponding points. As we will see, we do not

[2]**NOTE:** In the following, the addressed computer models are assumed deterministic, and thus a response from a model lacks random error (i.e., repeated runs for the same input parameters gives the same response from the model (e.g., the simulator)).

use the Euclidean distance, however, since this distance weights all the variables equally.

Kriging (Gaussian Process Regression): Fundamentals

Based on the Bayesian statistics, Kriging model treats the deterministic response of $y(\mathbf{x})$ as a realization of a stochastic process $Y(\mathbf{x})$

$$Y(\mathbf{x}) = \psi(\mathbf{x}) + Z(\mathbf{x})$$

where

- $\psi(\mathbf{x})$ is a regression (or "trend") function;
- $Z(\mathbf{x})$ is a Gaussian process.

The idea is that $\psi(\mathbf{x})$ captures the general trend of the real function, while $Z(\mathbf{x})$ models the errors (or "residuals") made by $\psi(\mathbf{x})$ w.r.t. the real function $y(\mathbf{x})$. The definition of the regression function $\psi(\mathbf{x})$ leads to different Kriging meta-models:

- $\psi(\mathbf{x}) = \sum_{q=1}^{Q} \alpha_q \beta_q(\mathbf{x})$: *Universal Kriging*;
- $\psi(\mathbf{x}) = \alpha_1 = \mu$: *Ordinary Kriging*;
- $\psi(\mathbf{x}) = 0$: *Simple Kriging*;

Note that α_q are unknown coefficients and should be estimated. For *Ordinary Kriging*, we have one single unknown coefficient $\alpha_1 = \mu$.

The following regression models can be defined:

- **Constant regression**, $Q = 1$:

$$\beta_1(\mathbf{x}) = 1$$

- **Linear regression**, $Q = (K+1)$:

$$\begin{gathered}\beta_1(\mathbf{x}) = 1 \\ \beta_2(\mathbf{x}) = x_1, ..., \beta_{K+1}(\mathbf{x}) = x_K\end{gathered}$$

- **Quadratic regression**, $Q = \frac{1}{2}(K+1)(K+2)$:

$$\begin{gathered}\beta_1(\mathbf{x}) = 1 \\ \beta_2(\mathbf{x}) = x_1, ..., \beta_{K+1}(\mathbf{x}) = x_K \\ \beta_{K+2}(\mathbf{x}) = x_1^2, ..., \beta_{2K+1}(\mathbf{x}) = x_1 x_K \\ \beta_{2K+2}(\mathbf{x}) = x_2^2, ..., \beta_{3K+1}(\mathbf{x}) = x_2 x_K \\ ... \\ \beta_Q(\mathbf{x}) = x_K^2\end{gathered}$$

The term $Z(\mathbf{x})$ is assumed to have the following stochastic behaviors

$$E[Z(\mathbf{x})] = 0$$
$$Cov\left[Z(\mathbf{x}), Z\left(\mathbf{x}^{'}\right)\right] = \sigma^2 Corr\left(\mathbf{x}, \mathbf{x}^{'}\right)$$

where

- σ^2 is the process variance;
- $Corr\left(\mathbf{x}, \mathbf{x}^{'}\right)$ is the correlation between any two locations $\mathbf{x}$ and $\mathbf{x}^{'}$.

The correlation function $Corr\left(\mathbf{x}, \mathbf{x}^{'}\right)$ is defined as a function of the distance $d\left(\mathbf{x}, \mathbf{x}^{'}\right)$ between samples $\mathbf{x}$ and $\mathbf{x}^{'}$ and satisfies the following conditions:

$$\lim_{d\left(\mathbf{x},\mathbf{x}^{'}\right)\to 0} Corr\left(\mathbf{x}, \mathbf{x}^{'}\right) = 1$$
$$\lim_{d\left(\mathbf{x},\mathbf{x}^{'}\right)\to \infty} Corr\left(\mathbf{x}, \mathbf{x}^{'}\right) = 0$$

More in detail, the correlation function used for Kriging meta-modeling is defined as

$$Corr\left(\mathbf{x}, \mathbf{x}^{'}\right) = \prod_{k=1}^{K} Corr\left(x_k, x_k^{'}\right)$$

The most common definition of the correlation between the k-th variable of $\mathbf{x}$ and the k-th variable of $\mathbf{x}^{'}$ is the following

$$Corr\left(x_k, x_k^{'}\right) = exp\left(-\theta_k \left|x_k - x_k^{'}\right|^{p_k}\right)$$

where $\theta_k \geq 0$ and $p_k \in [1, 2]$, for $k = 1, ..., K$. This leads to the following definition of correlation between samples $\mathbf{x}$ and $\mathbf{x}^{'}$

$$Corr\left(\mathbf{x}, \mathbf{x}^{'}\right) = \prod_{k=1}^{K} exp\left(-\theta_k \left|x_k - x_k^{'}\right|^{p_k}\right)$$

Note that, rather than using the Euclidean distance, the following weighted distance is employed

$$d\left(\mathbf{x}, \mathbf{x}^{'}\right) = \sum_{k=1}^{K} \theta_k \left|x_k - x_k^{'}\right|^{p_k} \tag{2.4}$$

and

$$Corr\left(\mathbf{x}, \mathbf{x}^{'}\right) = exp\left(-d\left(\mathbf{x}, \mathbf{x}^{'}\right)\right) = exp\left(-\sum_{k=1}^{K} \theta_k \left|x_k - x_k^{'}\right|^{p_k}\right) \tag{2.5}$$

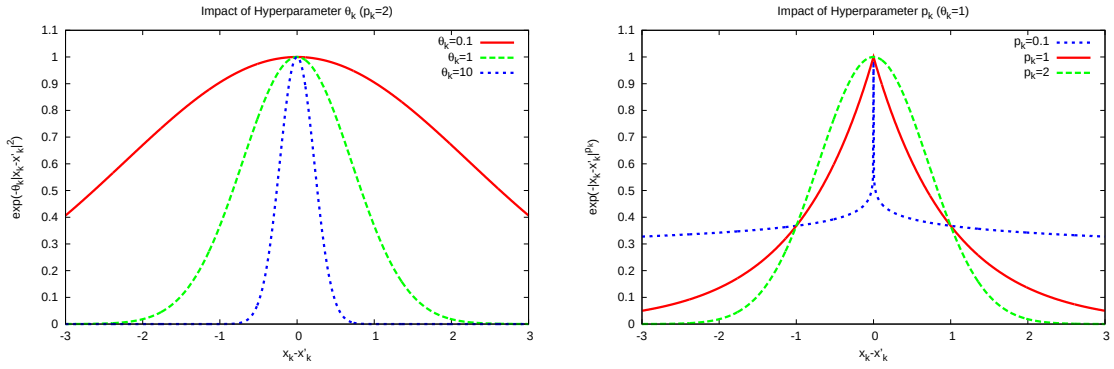
Thus, the correlation between two samples $\mathbf{x}$ and $\mathbf{x}^{'}$ is a function of both their distance in all K dimensions and of a set of $2K$ *hyper-parameters* $\mathbf{\Theta}$

$$\mathbf{\Theta} = \{\theta_1, ..., \theta_K; \, p_1, ..., p_K\}$$

whose values are unknown and should be estimated. Given the correlation function defined in (2.4) and (2.5), when the distance between $\mathbf{x}$ and $\mathbf{x}^{'}$ is small, the correlation is near one. Similarly, when the distance between the points is large, the correlation will approach to zero:

$$0 < Corr\left(\mathbf{x}, \mathbf{x}^{'}\right) \leq 1$$

The parameter θ_k in the distance formula (2.4) can be interpreted as measuring the importance or "activity" of the k-th variable x_k. To see this, note that saying "variable k is active" means that even small values of $\left|x_k - x_k^{'}\right|$ may lead to large differences in the function values at $\mathbf{x}$ and $\mathbf{x}^{'}$. This means that even small values of $\left|x_k - x_k^{'}\right|$ should imply a low correlation between the function values $y(\mathbf{x})$ and $y(\mathbf{x}^{'})$. If θ_k is very large, then it will indeed be true that small values of $\left|x_k - x_k^{'}\right|$ translate into large "distances" and hence low correlation. The exponent p_k is related to the smoothness of the function in coordinate direction k, with $p_k = 2$ corresponding to smooth functions and values near $p_k = 1$ corresponding to less smoothness.



(a) (b)

Figure 2.18: Effect of (a) hyper-parameter θ_k and of (b) hyper-parameter p_k on the correlation function between the k-th coordinate of points $\mathbf{x}$ and $\mathbf{x}^{'}$.

A common choice to reduce the number of unknowns (and consequently, the computational load of the training phase) is to fix the value of p_k to a given value, for $k = 1, ..., K$. The following correlation models can be derived:

- **Exponential correlation**: $p_k = 1$, $k = 1, ..., K$

$$Corr\left(x_k, x_k^{'}\right) = exp\left(-\theta_k \left|x_k - x_k^{'}\right|\right) \tag{2.6}$$

- **Gaussian correlation**: $p_k = 2$, $k = 1, ..., K$

$$Corr\left(x_k, x_k^{'}\right) = exp\left(-\theta_k \left|x_k - x_k^{'}\right|^2\right) \tag{2.7}$$

while if p_k is not fixed we get the so-called

- **Generalized exponential correlation**:

$$Corr\left(x_k, x_k^{'}\right) = exp\left(-\theta_k \left|x_k - x_k^{'}\right|^{p_k}\right)$$

NOTE: the DACE toolbox doesn't allow the estimation of different exponents p_k along the K dimensions.

Other correlation models can be defined. In particular, the Kriging MATLAB *DACE* toolbox [31] supports the following alternative correlation function

- **Linear correlation**:

$$Corr\left(x_k, x_k^{'}\right) = \max\left\{0,\, 1 - \theta_k \left|x_k - x_k^{'}\right|\right\} \tag{2.8}$$

- **Spherical correlation**:

$$Corr\left(x_k, x_k^{'}\right) = 1 - 1.5\xi_k + 0.5\xi_k^3 \tag{2.9}$$

where

$$\xi_k = min\left\{1,\, \theta_k \left|x_k - x_k^{'}\right|\right\}$$

- **Cubic correlation**:

$$Corr\left(x_k, x_k^{'}\right) = 1 - 3\xi_k^2 + 2\xi_k^3 \tag{2.10}$$

where

$$\xi_k = min\left\{1,\, \theta_k \left|x_k - x_k^{'}\right|\right\}$$

- **Spline correlation**:

$$Corr\left(x_k, x_k^{'}\right) = \zeta\left(\xi_k\right) \tag{2.11}$$

where

$$\xi_k = \theta_k \left| x_k - x_k^{'} \right|$$

and

$$\zeta(\xi_h) = \left\{ \begin{array}{cc} 1 - 15\xi_k^2 + 30\xi_k^3 & for\, 0 \leq \xi_k \leq 0.2 \\ 1.25\left(1 - \xi_k\right)^3 & for\, 0.2 < \xi_k < 1 \\ 0 & for\, \xi_k \geq 1 \end{array} \right\}.$$

[3]If the underlying phenomenon is continuously differentiable, the correlation function will likely show a parabolic behaviour near the origin, which means that the Gaussian, the cubic or the spline function should be chosen. Conversely, physical phenomena usually show a linear behavior near the origin, and exponential, generalized exponential, linear or spherical would usually perform better. Also note that for large distances the correlation is 0 according to the linear, cubic, spherical and spline functions, while it is asymptotically 0 when applying the other functions. Often the phenomenon is anisotropic. This means that different correlations are identified in different directions. This is accounted in the above correlation functions, since different parameters θ_k are allowed in the K dimensions of the input space.

Gaussian and exponential correlation functions are the most used in practical applications, since they represent a good choice for most of the conventional physical processes.

We assume that the correlation between two samples x and x$^{'}$is stationary, meaning that the set of hyper-parameters $\mathbf{\Theta}$ is invariant with respect to $\mathbf{x}$.

Now consider that the real value of $y\left(\mathbf{x}\right)$ is given at N sample points (training locations):

$$\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, ..., \mathbf{x}^{(N)}$$

Kriging supposes that the stochastic process $Y\left(\mathbf{x}\right)$ realizes all the N given samples:

$$Y\left(\mathbf{x}^{(i)}\right) = \psi\left(\mathbf{x}^{(i)}\right) + Z\left(\mathbf{x}^{(i)}\right) = y\left(\mathbf{x}^{(i)}\right) \tag{2.12}$$

Assuming a constant regression function (i.e., *Ordinary Kriging*)

$$\psi\left(\mathbf{x}\right) = \alpha_1 = \mu$$

[3]**NOTE:** The choice of the correlation function should be motivated by the underlying phenomenon, e.g., a function we want to optimize or a physical process we want to model.

the probability density function distribution conditioned on these realizations (called *Likelihood Function*) is obtained in logarithmic form as:

$$\begin{array}{l} Ln\left(\mu,\sigma^2,\boldsymbol{\Theta}\right)=ln\left\{PDF\left(\mathbf{y}/\,\mathbf{x}^{(1)},\mathbf{x}^{(2)},...,\mathbf{x}^{(N)}\right)\right\}= \\ =-\frac{N}{2}ln\left(2\pi\right)-\frac{N}{2}ln\sigma^2-\frac{1}{2}ln\left|\mathbf{R}\right|-\frac{(\mathbf{y}-\mathbf{1}\mu)^T\mathbf{R}^{-1}(\mathbf{y}-\mathbf{1}\mu)}{2\sigma^2} \end{array} \tag{2.13}$$

where

$$\mathbf{y}=\left[y\left(\mathbf{x}^{(1)}\right),y\left(\mathbf{x}^{(2)}\right),...,y\left(\mathbf{x}^{(N)}\right)\right]^T$$

$\mathbf{1}$ is a N-dimensional vector of ones ($\mathbf{1}=[1,1,...,1]^T$), and $\mathbf{R}$ is an $N\times N$ correlation matrix whose entries are represented by the **correlation between training samples** ($R_{ij}=Corr\left(\mathbf{x}^{(i)},\mathbf{x}^{(j)}\right)$). Note that the dependence of the *Likelihood Function* on the hyper-parameters $\boldsymbol{\Theta}$ is via the correlation matrix $\mathbf{R}$:

$$\mathbf{R}=\begin{bmatrix} Corr\left(\mathbf{x}^{(1)},\mathbf{x}^{(1)}\right) & ... & Corr\left(\mathbf{x}^{(1)},\mathbf{x}^{(N)}\right) \\ ... & \cdot & ... \\ Corr\left(\mathbf{x}^{(N)},\mathbf{x}^{(1)}\right) & \cdot & Corr\left(\mathbf{x}^{(N)},\mathbf{x}^{(N)}\right) \end{bmatrix} \tag{2.14}$$

The *Ordinary Kriging* model needs to estimate the values of μ, σ^2 and $\boldsymbol{\Theta}$ based on the *Maximum Likelihood Estimation* (*MLE*). The values of μ and σ^2 that maximize $Ln\left(\mu,\sigma^2,\boldsymbol{\Theta}\right)$ are solved in closed form as

$$\hat{\mu}=\frac{\mathbf{1}^T\mathbf{R}^{-1}\mathbf{y}}{\mathbf{1}^T\mathbf{R}^{-1}\mathbf{1}} \tag{2.15}$$

and

$$\hat{\sigma}^2=\frac{\left(\mathbf{y}-\mathbf{1}\hat{\mu}\right)^T\mathbf{R}^{-1}\left(\mathbf{y}-\mathbf{1}\hat{\mu}\right)}{N}. \tag{2.16}$$

Substituting (2.15) and (2.16) in (2.13) the following *Concentrated Likelihood Function* is obtained

$$Ln\left(\boldsymbol{\Theta}\right)=-\frac{N}{2}ln\left(2\pi\right)-\frac{N}{2}ln\hat{\sigma}^2-\frac{1}{2}ln\left|\mathbf{R}\right|$$

which depends only on the set of hyper-parameters $\boldsymbol{\Theta}$. This function should be maximized to get an estimate of $\boldsymbol{\Theta}$, and hence an estimate of the correlation matrix $\mathbf{R}$. Multiple optimization algorithms can be used (e.g., gradient descent, *GA*, *PSO*, etc.). Then, equations (2.15) and (2.16) are used to get an estimate of $\hat{\mu}$ and $\hat{\sigma}^2$. Note that when we estimate these parameters by maximum likelihood, we are essentially finding values of the parameters that best describe the behavior of the true function (we do not know them exactly, that's why we should use the hats).

Finally, consider the linear predictor $\hat{y}\left(\mathbf{x}\right)$ which estimates $y\left(\mathbf{x}\right)$ at location $\mathbf{x}$ (and $y\left(\mathbf{x}\right)$ is unknown), defined as

$$\hat{y}(\mathbf{x}) = \mathbf{c}^T(\mathbf{x})\,\mathbf{Y} \tag{2.17}$$

where

$$\mathbf{Y} = \left[Y\left(\mathbf{x}^{(1)}\right), ..., Y\left(\mathbf{x}^{(N)}\right)\right]^T = \left[y\left(\mathbf{x}^{(1)}\right), ..., y\left(\mathbf{x}^{(N)}\right)\right]^T$$

by hypothesis (2.12). The Kriging model obtains the *best linear unbiased predictor* ($BLUP$) by choosing the N-dimensional vector $\mathbf{c}(\mathbf{x})$ to minimize the following mean squared error (MSE):

$$s^2(\mathbf{x}) = Var\left[\hat{y}(\mathbf{x}) - Y(\mathbf{x})\right] = Var\left[\hat{y}(\mathbf{x}) - y(\mathbf{x})\right] \tag{2.18}$$

subject to the following *unbiasedness* constraint:

$$E\left[\hat{y}(\mathbf{x})\right] = E\left[Y(\mathbf{x})\right] = E\left[y(\mathbf{x})\right]$$

Then, $\mathbf{c}(\mathbf{x})$ is solved in closed form as

$$\hat{\mathbf{c}}(\mathbf{x}) = \mathbf{R}^{-1}\mathbf{r}(\mathbf{x}) + \frac{\mathbf{R}^{-1}\mathbf{1}\left(1 - \mathbf{1}\mathbf{R}^{-1}\mathbf{r}(\mathbf{x})\right)}{\mathbf{1}^T\mathbf{R}^{-1}\mathbf{1}} \tag{2.19}$$

where $\mathbf{r}(\mathbf{x})$ is an N-dimensional vector containing the correlation between the sample $\mathbf{x}$ at which we are making the prediction and the N training samples

$$\mathbf{r}(\mathbf{x}) = \begin{bmatrix} Corr\left(\mathbf{x}, \mathbf{x}^{(1)}\right) \\ \\ ... \\ \\ Corr\left(\mathbf{x}, \mathbf{x}^{(N)}\right) \end{bmatrix} \tag{2.20}$$

Substituting $\mathbf{r}(\mathbf{x})$, we get the final expression of the *Ordinary Kriging* predictor

$$\hat{y}(\mathbf{x}) = \hat{\mu} + \mathbf{r}^T(\mathbf{x})\,\mathbf{R}^{-1}\left(\mathbf{y} - \mathbf{1}\hat{\mu}\right) \tag{2.21}$$

This function models the estimate of $y(\mathbf{x})$ at any location $\mathbf{x}$ by interpolating the sample points with real values of $y(\mathbf{x})$. On the right-hand side of equation (2.21), the first term, $\hat{\mu}$, is the result of simply plugging $\mathbf{x}$ into the regression equation, and the second term represents the "adjustment" to this prediction based on the correlation of $\mathbf{x}$ with the N sampled points (which are known). Similarly, substituting (2.17) and (2.19) in (2.18) the MSE in final form results in

$$s^2(\mathbf{x}) = MSE(\mathbf{x}) = \hat{\sigma}^2\left[1 - \mathbf{r}^T(\mathbf{x})\,\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}) + \frac{\left(1 - \mathbf{1}^T\mathbf{R}^{-1}\mathbf{r}(\mathbf{x})\right)^2}{\mathbf{1}^T\mathbf{R}^{-1}\mathbf{1}}\right] \tag{2.22}$$

This function models the uncertainty expected in $\hat{y}(\mathbf{x})$. It indicates that the accuracy of $\hat{y}(\mathbf{x})$ depends largely on the distance from the given sampling points (i.e., the training samples). Intuitively, the closer $\mathbf{x}$ is to the training points, the less uncertain the prediction $\hat{y}(\mathbf{x})$.

Note that:

- If there is no correlation with training samples ($\mathbf{r}(\mathbf{x}) = 0$), then we just predict $\hat{y}(\mathbf{x}) = \hat{\mu}$;

- If we are making a prediction at the i-th sampled point ($\mathbf{x} = \mathbf{x}^{(i)}$), then $\hat{y}\left(\mathbf{x}^{(i)}\right) = y\left(\mathbf{x}^{(i)}\right)$ and $s^2\left(\mathbf{x}^{(i)}\right) = 0$.

- The predictor in equation (2.21) can be also written as:

$$\hat{y}(\mathbf{x}) = \hat{\mu} + \mathbf{w}^T \mathbf{r}(\mathbf{x}) = \hat{\mu} + \sum_{i=1}^{N} w_i r_i(\mathbf{x}) \tag{2.23}$$

  where $\mathbf{w} = \mathbf{R}^{-1}(\mathbf{y} - \mathbf{1}\hat{\mu})$ is a vector of constants and $r_i(x) = Corr\left[\mathbf{x}, \mathbf{x}^{(i)}\right]$, for $i = 1, ..., N$. Thus, we see that the Kriging predictor is a linear combination of "basis functions" $r_i(x)$, for $i = 1, ..., N$ that interpolate the data. The basis functions depend upon the correlation parameters θ_k and p_k for $k = 1, ..., K$, and these are "tuned" to the training data during the *Maximum Likelihood Estimation*.

Interpretation of s^2 (prediction uncertainty, MSE)

The correlation between the new sample $\mathbf{x}$ and the training samples affects our estimate of prediction accuracy. In fact, it makes intuitive that, if $\mathbf{x}$ is very close to a training sample $\mathbf{x}^{(i)}$, we should be much more confident in our prediction of $y(\mathbf{x})$ than we would be if $\mathbf{x}$ were far away from all the sampled points. This intuition is reflected in the general formula for the uncertainty $s^2(\mathbf{x})$ of the predictor.

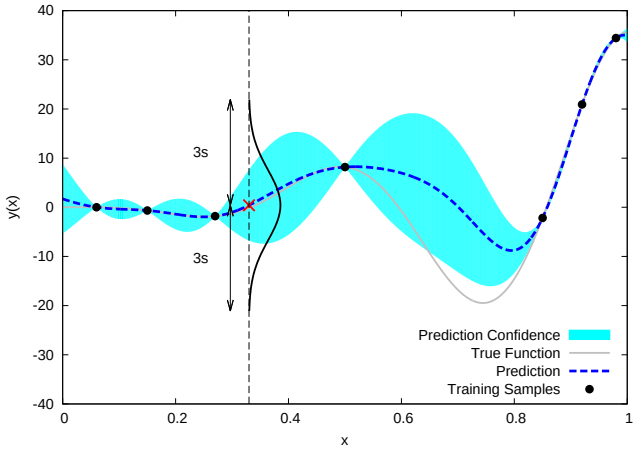


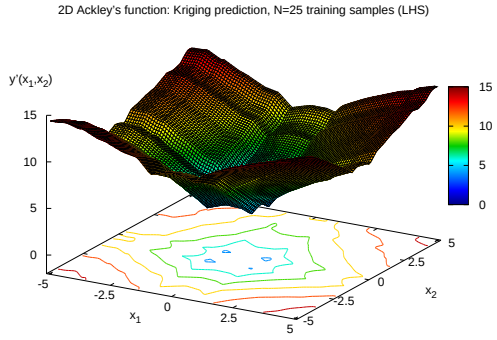
Figure 2.19: Example in 1D case.

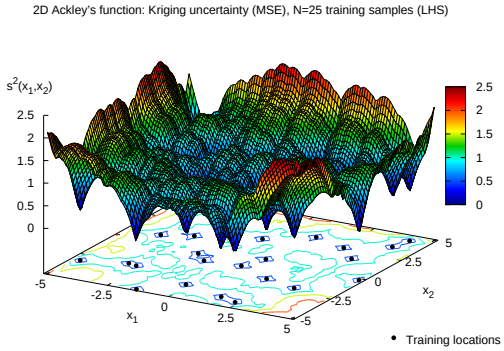
Let us model the uncertainty at $\mathbf{x}$ by treating the function value $y(\mathbf{x})$ as the realization of a normally distributed random variable Y with mean $\hat{y}(\mathbf{x})$ and standard deviation given by $s(\mathbf{x}) = \sqrt{s^2(\mathbf{x})}$. Then, the Kriging model is approximately 99.7% confident that $\hat{y}(\mathbf{x})$ lies inside the interval defined by

$$\hat{y}(\mathbf{x}) \pm 3s$$

Note that if we are making a prediction at the i-th sampled point ($\mathbf{x} = \mathbf{x}^{(i)}$), we get $s^2\left(\mathbf{x}^{(i)}\right) = 0$. This is as it should be: with a deterministic function, once we have sampled a point, we know its value there. Thus, our uncertainty, as measured by s^2, should be zero. In conclusion, s^2 gives us a measure of how accurate and "reliable" is a given predicted value (it shouldn't be confused with the prediction error, that is computed knowing the real value of $y(\mathbf{x})$). In Fig. 2.19 it is reported the prediction confidence interval of a 1D function. It is defined as $\hat{y} \pm s$ and it is null at observed points, while it grows with the distance w.r.t. the nearest training point. Fig. 2.20 shows the Prediction and the prediction uncertainty (MSE) of the Ackley's function when $N = 25$ and $N = 250$ training samples and are used. The prediction uncertainty is null at observed points, while it grows with the distance w.r.t. the nearest training point. Increasing the number of training samples leads to an higher prediction accuracy and to a lower uncertainty.

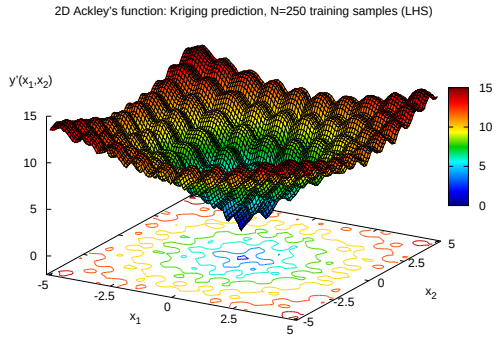
$N = 25$ training samples

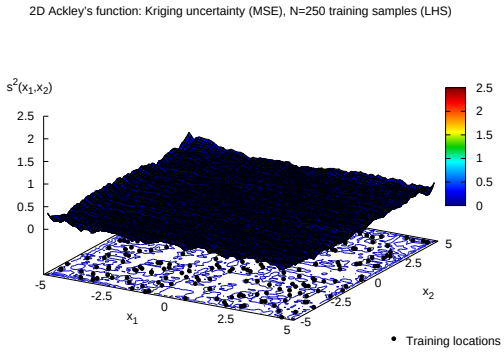




(a) Prediction (b) Prediction uncertainty (MSE)

$N = 250$ training samples





(c) Prediction (d) Prediction uncertainty (MSE)

Figure 2.20: Example in 2D case (Ackley's function). (a) Prediction and (b) prediction uncertainty (MSE) of the Ackley's function using $N = 25$ training samples. (c) Prediction and (d) prediction uncertainty (MSE) of the Ackley's function using $N = 250$ training samples.

Universal Kriging

While *Ordinary Kriging* assumes that the stochastic process $Y(\mathbf{x})$ has the form

$$Y(\mathbf{x}) = \mu + Z(\mathbf{x})$$

more generally, we can write

$$Y(\mathbf{x}) = \psi(\mathbf{x}) + Z(\mathbf{x})$$

Universal Kriging assumes that the regression (or "trend") function is computed as the weighted sum of known basis functions of different order

$$\psi(\mathbf{x}) = \sum_{q=1}^{Q} \alpha_q \beta_q(\mathbf{x})$$

Then, we can define the $N \times Q$ regression matrix $\mathbf{B}$ as

$$\mathbf{B} = \begin{bmatrix} \beta_1\left(\mathbf{x}^{(1)}\right) & \ldots & \beta_Q\left(\mathbf{x}^{(1)}\right) \\ \ldots & \ldots & \ldots \\ \beta_1\left(\mathbf{x}^{(N)}\right) & \ldots & \beta_Q\left(\mathbf{x}^{(N)}\right) \end{bmatrix}$$

The maximization of the *Likelihood Function* leads to the following definition of the estimator

$$\hat{y}(\mathbf{x}) = \mathbf{M}\boldsymbol{\alpha} + \mathbf{r}^T(\mathbf{x})\,\mathbf{R}^{-1}(\mathbf{y} - \mathbf{B}\boldsymbol{\alpha})$$

where $\mathbf{M}$ is a Q-dimensional vector

$$\mathbf{M} = [\beta_1(\mathbf{x}), ..., \beta_Q(\mathbf{x})]$$

and the coefficients are estimated as

$$\boldsymbol{\alpha} = \frac{\mathbf{B}^T\mathbf{R}^{-1}\mathbf{y}}{\mathbf{B}^T\mathbf{R}^{-1}\mathbf{B}}$$

The prediction uncertainty (MSE) is then obtained as

$$s^2(\mathbf{x}) = MSE(\mathbf{x}) = \hat{\sigma}^2\left[1 - \mathbf{r}^T(\mathbf{x})\,\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}) + \frac{\left(1 - \mathbf{B}^T\mathbf{R}^{-1}\mathbf{r}(\mathbf{x})\right)^2}{\mathbf{B}^T\mathbf{R}^{-1}\mathbf{B}}\right]$$

Note that in the case of constant regression (i.e., *Ordinary Kriging*), we have

$$\psi(\mathbf{x}) = \alpha_1 = \mu$$

$$\mathbf{B} = \begin{bmatrix} 1 \\ 1 \\ \ldots \\ \ldots \\ 1 \end{bmatrix} = \mathbf{1}$$

$$\mathbf{M} = [1, ..., 1] = \mathbf{1}^T$$

$$\boldsymbol{\alpha} = \hat{\mu} = \frac{\mathbf{1}^T\mathbf{R}^{-1}\mathbf{y}}{\mathbf{1}^T\mathbf{R}^{-1}\mathbf{1}}$$

Going back to the *Universal Kriging* formulation, if we predict at one training location (i.e., $\mathbf{x} = \mathbf{x}^{(i)}$), we get that $\mathbf{r}\left(\mathbf{x}^{(i)}\right)$ is the i-th column of the correlation matrix $\mathbf{R}$ ($\mathbf{R}_i$)

$$\mathbf{r}\left(\mathbf{x}^{(i)}\right) = \begin{bmatrix} Corr\left(\mathbf{x}^{(i)}, \mathbf{x}^{(1)}\right) \\ \\ ... \\ \\ Corr\left(\mathbf{x}^{(i)}, \mathbf{x}^{(N)}\right) \end{bmatrix} = \mathbf{R}_i$$

Then,

$$\mathbf{r}^T\left(\mathbf{x}^{(i)}\right)\mathbf{R}^{-1} = \left(\mathbf{R}^{-1}\mathbf{r}\left(\mathbf{x}^{(i)}\right)\right)^T = \left(\mathbf{R}^{-1}\mathbf{R}_i\right)^T = \mathbf{e}_i^T$$

where $\mathbf{e}_i$is the i-th unit vector (having all zeros and a one at the i-th position). Finally, we obtain

$$\mathbf{r}^T\left(\mathbf{x}^{(i)}\right)\mathbf{R}^{-1}\left(\mathbf{y} - \mathbf{B}\boldsymbol{\alpha}\right) = \mathbf{e}_i^T\left(\mathbf{y} - \mathbf{B}\boldsymbol{\alpha}\right) = y\left(\mathbf{x}^{(i)}\right) - \mathbf{B}\boldsymbol{\alpha}$$

Since $\mathbf{B}\boldsymbol{\alpha} = \mathbf{M}\boldsymbol{\alpha}$, we get

$$\hat{y}\left(\mathbf{x}^{(i)}\right) = \mathbf{M}\boldsymbol{\alpha} + y\left(\mathbf{x}^{(i)}\right) - \mathbf{M}\boldsymbol{\alpha} = y\left(\mathbf{x}^{(i)}\right)$$

meaning that the Kriging regressor exactly interpolates the training data. This is of course also valid for *Ordinary Kriging*. At point $\mathbf{x} = \mathbf{x}^{(i)}$ we get

$$\hat{y}\left(\mathbf{x}^{(i)}\right) = \hat{\mu} + \mathbf{r}^T\left(\mathbf{x}^{(i)}\right)\mathbf{R}^{-1}\left(\mathbf{y} - \mathbf{1}\hat{\mu}\right) = \hat{\mu} + \mathbf{e}_i^T\left(\mathbf{y} - \mathbf{1}\hat{\mu}\right) = \hat{\mu} + f\left(\mathbf{x}^{(i)}\right) - \hat{\mu} = f\left(\mathbf{x}^{(i)}\right)$$

If we look at the uncertainty, we have

$$s^2\left(\mathbf{x}^{(i)}\right) = MSE\left(\mathbf{x}^{(i)}\right) = \hat{\sigma}^2\left[1 - \mathbf{r}^T\left(\mathbf{x}^{(i)}\right)\mathbf{R}^{-1}\mathbf{r}\left(\mathbf{x}^{(i)}\right) + \frac{\left(1 - \mathbf{1}^T\mathbf{R}^{-1}\mathbf{r}\left(\mathbf{x}^{(i)}\right)\right)^2}{\mathbf{1}^T\mathbf{R}^{-1}\mathbf{1}}\right]$$

with

$$\mathbf{r}^T\left(\mathbf{x}^{(i)}\right)\mathbf{R}^{-1}\mathbf{r}\left(\mathbf{x}^{(i)}\right) = \mathbf{r}^T\left(\mathbf{x}^{(i)}\right)\mathbf{e}_i = r_i\left(\mathbf{x}^{(i)}\right) = Corr\left(\mathbf{x}^{(i)}, \mathbf{x}^{(i)}\right) = 1$$
$$\mathbf{1}^T\mathbf{R}^{-1}\mathbf{r}\left(\mathbf{x}^{(i)}\right) = \mathbf{1}^T\mathbf{e}_i = 1$$

resulting in

$$s^2\left(\mathbf{x}^{(i)}\right) = \left(\mathbf{x}^{(i)}\right) = \hat{\sigma}^2\left[1 - 1 + \frac{\left(1-1\right)^2}{\mathbf{1}^T\mathbf{R}^{-1}\mathbf{1}}\right] = 0$$

Note that if the point $\mathbf{x}$ is very far from all training samples, we have

$$\mathbf{r}\left(\mathbf{x}\right) \to \mathbf{0}$$
$$s^2\left(\mathbf{x}\right) \to \hat{\sigma}^2$$
$$\hat{y}\left(\mathbf{x}\right) \to \hat{\mu}$$

Dimension of the output vector y

The MATLAB DACE Toolbox [31] is able to handle models with G-dimensional responses $\left(\mathbf{y}\left(\mathbf{x}\right) : \mathbb{R}^K \to \mathbb{R}^G\right)$

$$\mathbf{y} = [y_1, y_2, ..., y_G]$$

All the previous equations can be easily expanded to a G-dimensional vector predicted output.

2.6.2 Support Vector Regression

The SVR prediction technique is based on the Support Vector Machines (SVM) [32]-[34] theory. The fundamental characteristic of the SVR technique is that it allows defining an "error" margin ε on the training samples; this means that it is assumed that the training samples can contain a certain amount of uncertainty but this does not interfer on the efficiency and accuracy of the prediction process. All the techniques based on the SVM theory are thus suited for dealing with noisy training data, as for example the leaning problems based on observations/measurements.

Within the SVR theory, the training samples which lie outside the $\pm\varepsilon$ band (called $\pm\varepsilon$-tube) are called support vectors (see. Fig 2.21) . These samples are fundamental for the training process of the SVR.

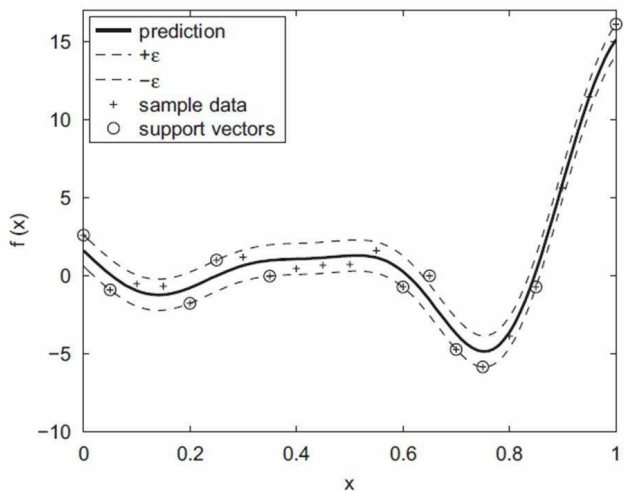


Figure 2.21: SVR prediction of a 1D test function. Among all the training samples, only some of them are support vectors.

Independently from the training technique used for the SVR, the expression of the SVR predictor is the following:

$$\hat{y}(\mathbf{x}) = \mu + \sum_{i=1}^{N} w^{(i)} \psi\left(\mathbf{x}, \mathbf{x}^{(i)}\right) \tag{2.24}$$

where $\psi^{(i)} = \psi\left(\mathbf{x}, \mathbf{x}^{(i)}\right)$ is the $i-$th basis function, $w^{(i)}$ is the $i-$th weight and μ is a bias coefficient which can be computed exploiting the Karish-Kuhn-Tucker conditions [35] starting from the support vectors. The goal of the training phase is to find the best prediction function which deviate from the training samples a maximum quantity ε. The standard form for the training of the SVR model is the following:

$$\begin{aligned} \min \quad & \left(\frac{1}{2}|\mathbf{w}|^2 + C\frac{1}{N}\sum_{i=1}^{N}\xi^{+(i)} + \xi^{-(i)}\right) \\ subj.to. \quad & \begin{cases} y^{(i)} - \mathbf{w}\mathbf{x}^{(i)} - \mu & \leq \varepsilon + \xi^{+(i)} \\ \mathbf{w}\mathbf{x}^{(i)} + \mu - y^{(i)} & \leq \varepsilon + \xi^{-(i)} \\ \xi^{+(i)}, \xi^{-(i)} \geq 0 \end{cases} \end{aligned} \tag{2.25}$$

where $\xi^{+(i)}, \xi^{-(i)}$ are the so called slack-variables [33]. It can be noticed that the training process illustrated in previous equation allows controlling the tradeoff of the final result in terms of complexity of the model and accuracy by opportunely choosing the C parameter. The problem in Eq.2.25 can be solved using the Lagrange multipliers technique and obtaining the canonical solution [34]:

$$\hat{y}(\mathbf{x}) = \mu + \sum_{i=1}^{N}\left(\alpha^{+(i)} + \alpha^{-(i)}\right)\left(\mathbf{x}^{(i)} \cdot \mathbf{x}\right) \tag{2.26}$$

The equation 2.26 is valid under linear regression hypothesis. The result can be extended to the case where non-linear basis function are used, obtaining

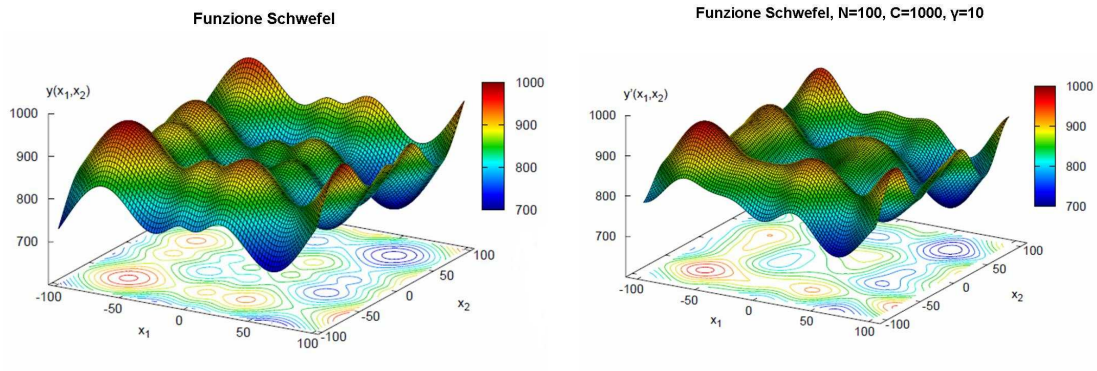
$$\hat{y}(\mathbf{x}) = \mu + \sum_{i=1}^{N}\left(\alpha^{+(i)} + \alpha^{-(i)}\right)\psi^{(i)} \tag{2.27}$$

It must be pointed out that Eq. 2.26 is valid only if the following assumptions hold true:

- ψ is continuous,
- ψ is symmetric, which means that $\psi\left(\mathbf{x}, \mathbf{x}^{(i)}\right) = \psi\left(\mathbf{x}^{(i)}, \mathbf{x}\right)$
- ψ is positive definite function, which means that the correlation function $\boldsymbol{\psi}^T = \boldsymbol{\psi}$ and has eigenvalues which are strictly positive.

The most common choices for the kernel function ψ are:

- Linear kernel: $\psi\left(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}\right) = \left(\mathbf{x}^{(i)} \cdot \mathbf{x}^{(j)}\right)$;
- d-grade homogeneous polynomial kernel: $\psi\left(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}\right) = \left(\mathbf{x}^{(i)} \cdot \mathbf{x}^{(j)}\right)^d$;
- Gaussian kernel (the most used): $\psi\left(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}\right) = e^{-\gamma\left\|\mathbf{x}^{(i)} \cdot \mathbf{x}^{(j)}\right\|^2}$;
- d-grade inhomogeneous polynomial kernel, etc..



(a) (b)

Figure 2.22: Two-dimensional Schwefel function. True function (a) and predicted function (b) using $N = 100$ training samples ($C = 100$, $\gamma = 10$)

With the goal of illustrating the performances of the SVR, Fig. 2.22 shows the comparison between the predicted output [Fig. 2.22(b)] and the corresponding benchmark function [Fig. 2.22(a)]. The benchmark function used is the 2-D Schwefel function. The number of training samples has been fixed to $N = 100$. It can be noticed that the accuracy in the prediction is acceptable: the position and amplitude of the maxima and minima is correctly predicted, as well as as the general behaviour of the function. However, due to the considered formulation, the predicted function appears to be smoother with respect to the original (see Fig 2.22(b) for $x_2 = -100$). This phenomenon is due to the fact that SVR does not interpolate the training samples; this can be suited for noisy training samples but is less suited for deterministic data because it can introduce noise in the prediction of the "exact" training samples. In addition a further drawback with respect to other techniques (such as Kriging) is the necessity to preliminary calibrate the training samples C (called penalty factor) and γ (for the Gaussian kernel).

2.6.3 Parameter selection via Cross-Validation (CV)

Many predictors, such as SVR, need a preliminary calibration phase in order to estimate the best combination of parameters which will be used during the training and testing phases. Often, such a calibration is performed by applying a cross-validation approach on a given set of known input/output pairs (i.e., a training set). Many cross-validation approaches exist, but the main two techniques are

1. V-fold cross-validation;

2. Leave-one-out cross-validation ($LOO - CV$);

On the one hand, V-fold cross-validation is considered a **non-exhaustive cross-validation method**, since it does not compute all ways of splitting the given training set. Even if it is an approximated technique, it is computationally faster. On the other hand, $LOO - CV$ is an exhaustive cross-validation method, since it requires the training and test on all possible ways to divide the original set into a training and a validation set.

2.6.3.1 V-fold Cross-Validation

More in details, SVR (with RBF kernel) are characterized by two main parameters, namely C and γ. C is often called "penalty factor" and controls the trade-off between the training error and the model complexity, while γ represents the exponent in the RBF kernel. RBFN are characterized by one parameter, the spread S. This parameter controls the smoothness of the Gaussian basis functions used in the hidden layer of the network.

Let us indicate with ($\boldsymbol{\alpha}$) the vector of parameters that should be calibrated:

- $\boldsymbol{\alpha} = (C, \gamma)$ for the SVR model with RBF kernel;
- $\boldsymbol{\alpha} = S$ for the RBFN model.

In order to identify the best parameters, a classical $V-$fold cross-validation approach is employed.

A given training set of N samples is divided into V subsets of approximately equal size. Then, for each $v-$th subset, a prediction model is trained using the remaining $V-1$ subsets. The resulting model is then used to test the prediction accuracy on the $v-$th subset, and the estimation error is computed by means of the Mean Squared Error, defined as follows:

$$MSE_v(\boldsymbol{\alpha}) = \frac{1}{T_v} \sum_{i=1}^{T_v} \{y_i - \widetilde{y}_i(\boldsymbol{\alpha})\}^2$$

where

- T_v is the number of samples inside the $v-$th subset;
- y_i is the real output associated to the $i-$th sample;
- $\boldsymbol{\alpha}$ correspond to the considered vector of parameters;
- $\widetilde{y}_i(\boldsymbol{\alpha})$ is the predicted output associated to the $i-$th sample for a given $\boldsymbol{\alpha}$.

Then, the cross-validation MSE for a given vector of parameters $\boldsymbol{\alpha}$ is computed as the average MSE obtained over all the V subsets

$$\eta(\boldsymbol{\alpha}) = \frac{1}{V} \sum_{v=1}^{V} MSE_v(\boldsymbol{\alpha})$$

The best configuration is finally identified as the one minimizing the cross-validation MSE

$$(\boldsymbol{\alpha}*) = \arg\{\min \eta(\boldsymbol{\alpha})\}$$

2.6.3.2 Kriging vs. Support Vector Regression (SVR)

"The lunch is not free" theorem is still valid when selecting the proper predictor. The choice of the prediction model is crucial and for sure it depends on the specific application. The following list of features may be drawn in order to enable a direct comparison between Kriging (Simple, Ordinary or Universal) and SVRs.

I denote with symbol $\uparrow$ features that appear as advantages, while symbol $\downarrow$ indicates a disadvantage. Symbol $\updownarrow$ indicates that the given feature may represent either an advantage or a disadvantage, depending on the application.

Feature	Kriging	SVR
Auto-tuning of hyper-parameters	YES $\uparrow$	NO $\downarrow$
Multi-dimensional output	YES $\uparrow$	NO $\downarrow$
Uncertainty measure	YES (MSE) $\uparrow$	NO $\downarrow$
Interpolates training data	YES $\updownarrow$	NO $\updownarrow$
Can handle noisy training data	NO $\downarrow$	YES $\uparrow$
Computational efficiency	LOW $\downarrow$	YES $\uparrow$
Can handle large number of variables ($K \geq 100$)	NO $\downarrow$	YES $\uparrow$

Table 2.1: Direct comparison between features of Kriging and SVR.

Tuning of the hyper-parameters

One of the most important advantages of Kriging over classical Support Vector Regression is the auto-tuning of the hyper-parameters. When using an ε-SVR with Radial Basis Function (RBF) kernel, a preliminary tuning of the hyper-parameters (namely, the penalty factor C and the Kernel coefficient γ) must be done in order to obtain good predictions.

More on noiseless/noisy training data

In its original formulation, **Kriging is intended to work with deterministic data**, meaning that the same output is always computed/measured for a given input vector. On the contrary, **Support Vector Regression (SVR) is able to manage noisy training samples**, where both the following conditions may happen:

1. The output of a given training sample may be not corresponding to the real output of the undergoing phenomenon/process. In other words, some noise can be added to the real output value;

2. More that one output value is present for the same input vector (i.e., we have multiple noisy realizations of a given training sample).

In other words, Kriging interpolates training data, meaning that the predicted output for a training location corresponds to the training output. On the other hand, no guarantees are given that the predictions made by an SVR interpolate the training observations.

In the following, we directly compare the predictions made by both Ordinary Kriging and ε-SVR (with RBF kernel) for the 1-D Ackley's function. $N = 21$ training samples are uniformly distributed inside the considered input range $-5 \leq x \leq 5$.

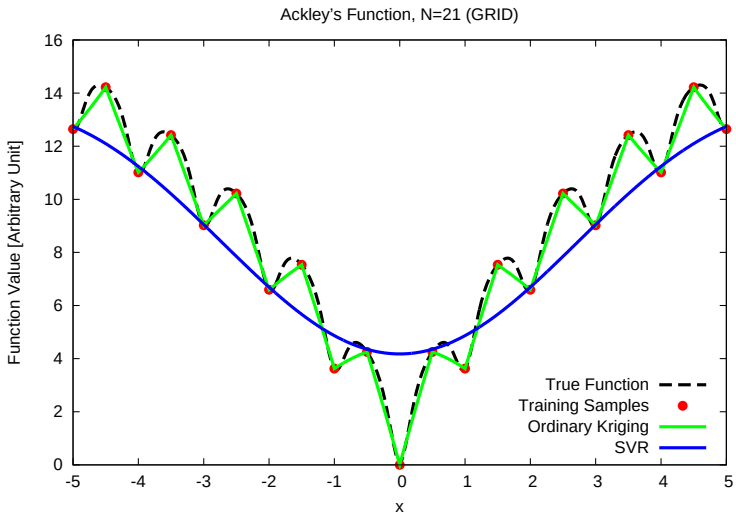


Figure 2.23: Ackley's function, $K = 1$ variables. Direct comparison between the predictions made by Ordinary Kriging and ε-SVR.

Observations

- predictions made by Kriging are much more accurate than those made by SVR. Normalized Mean Error is:
  - Ordinary Kriging: $NME = 4.95 \times 10^{-2}$
  - SVR: $NME = 5.95 \times 10^{-1}$
- Kriging interpolates training data, forcing its prediction to perfectly match training data, which are assumed to be deterministic realizations of the function to predict;
- SVR doesn't force the prediction to interpolate the observations. Different values are then predicted also when estimating the output at a training location.

In the following figure, a different example is given. In this case, our goal is to predict the function

$$y(x) = x$$

within the range $0 \leq x \leq 10$, by exploiting a set of $N = 21$ uniformly spaced training samples computed inside the same interval. Moreover, training samples are corrupted by an additive white Gaussian noise, with $SNR = 5$ [dB].

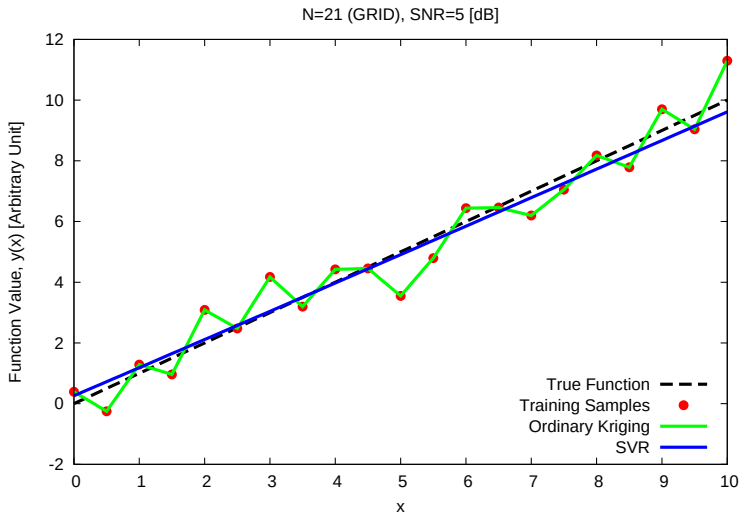


Figure 2.24: Ackley's function, $K = 1$ variables. Direct comparison between the predictions made by Ordinary Kriging and ε-SVR.

Observations

- In this case, predictions made by the SVR are much better than those made by the Ordinary Kriging; the computed Normalized Mean Error is:
  - Ordinary Kriging: $NME = 1.73 \times 10^{-1}$
  - SVR: $NME = 9.92 \times 10^{-2}$
- given its interpolating nature, Ordinary Kriging forces predictions to match all observed samples. However, this is not a desirable feature when treating noisy realization of the underlying function to predict.

Final considerations and guidelines

Given the above two examples, the following final observations may be considered when selecting the predictor, problem at hand:

- In case of noiseless deterministic simulations, it makes sense to use a Kriging predictor;
- **when samples are corrupted by noise** (both during the training and during the test phases), the SVR is not only is a good choice, but also **the only choice** (between the two considered predictors), since Kriging - in its standard implementation - is not able to manage noisy data.
- a new version of Kriging, called "stochastic Kriging" has been introduced in in order to enable the use of noisy training sets.

2.6.4 Radial Basis Function Networks ($RBFN$)

The idea of Radial Basis Function Networks (RBFN) derives from the theory of function approximation. The main features of a $RBFN$ [36]-[40] model are:

- They are two-layer feed-forward networks.
- The hidden nodes implement a set of radial basis functions (e.g. Gaussian functions).
- The output nodes implement linear summation functions.
- The network training is divided into two stages:
  1. first the weights from the input to hidden layer are determined,
  2. and then the weights from the hidden to output layer.
- The training/learning is very fast.
- They are very good at interpolation.

$RBFN$ with exact interpolation at training samples: theory

$RBFN$s are a special class of single hidden-layer feed forward neural networks for application to problems of supervised learning (i.e., those problems where the function value associated to training samples is assumed to be known during the training phase).

Let's suppose also in this case to have N training samples available, where:

- $\mathbf{x}^{(i)} \in \Re^K, \mathbf{x}^{(i)} = \left\{x_k^{(i)}, k = 1, ..., K\right\}$: $i-$th training sample point in $K-$dimensional space, $i = 1, ..., N$;
- $y_i = y\left(\mathbf{x}^{(i)}\right)$: function value (output) associated to the $i-$th training sample point $\mathbf{x}^{(i)}$.

The exact interpolation of the set of N data points in a multidimensional space requires that all the N K-dimensional input vectors $\mathbf{x}^{(i)} = \left\{x_k^{(i)}, k = 1, ..., K\right\}, i = 1, ..., N$ are mapped onto the corresponding outputs $y_i, i = 1, ..., N$. In other words, the goal of exact interpolation is to find a function $h(.)$ such that $h\left(\mathbf{x}^{(i)}\right) = y_i, i = 1, ..., N$.

The radial basis function approach introduces a set of N basis functions (one for each training sample) of the form:

$$\phi_i(\mathbf{x}) = \phi(\left\|\mathbf{x} - \mathbf{x}^{(i)}\right\|)$$

where $\phi(\cdot)$ is some nonlinear function and $\left\|\mathbf{x} - \mathbf{x}^{(i)}\right\|$ denotes the Euclidean distance between the generic input $\mathbf{x}$ and the i-th training point $\mathbf{x}^{(i)}$.

The characteristic feature of radial functions is that their response decreases (or increases) monotonically with distance from a central point. The centre, the distance scale, and the precise shape of the radial function are parameters of the model.

The most common choice is to consider the case of Gaussian basis function (and this is the choice adopted for the results in this report):

$$\phi_i(\mathbf{x}) = \phi(\left\|\mathbf{x} - \mathbf{x}^{(i)}\right\|) = exp\left(-\frac{\left\|\mathbf{x} - \mathbf{x}^{(i)}\right\|^2}{2\sigma^2}\right) = exp\left(-\frac{\left\|\mathbf{x} - \mathbf{x}^{(i)}\right\|^2}{2S}\right)$$

The term $\sigma^2 = S$ is often called "spread" and controls the smoothness properties of the interpolating function, as shown in next Figure:

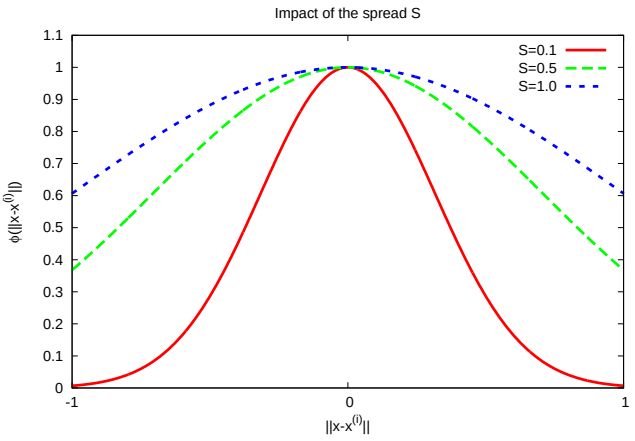


Figure 2.25: Impact of the spread value S on the width of the Gaussian basis functions.

The prediction made by the $RBFN$ model at a generic test location $\mathbf{x}^{(m)}$ is given by a linear combination of the basis functions

$$\hat{y}_m = h(\mathbf{x}^{(m)}) = \sum_{j=1}^{N} w_j \phi_{mj}$$

where

$$\phi_{mj} = \phi(\left\|\mathbf{x}^{(m)} - \mathbf{x}^{(j)}\right\|) = exp\left(-\frac{\left\|\mathbf{x}^{(m)} - \mathbf{x}^{(j)}\right\|^2}{2S}\right)$$

and w_j, $j = 1, ..., N$ are weights that must be estimated during the training phase. Thus, the interpolation condition at a generic training sample $\mathbf{x}^{(i)}$ can be expressed as

$$h(\mathbf{x}^{(i)}) = \sum_{j=1}^{N} w_j \phi_{ij} = y_i \, i = 1, ..., N$$

where

$$\phi_{ij} = \phi(\left\|\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\right\|) = exp\left(-\frac{\left\|\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\right\|^2}{2S}\right)$$

The above condition can be expressed also in a matrix form, $\mathbf{\Phi w^T} = \mathbf{y}$:

$$\begin{bmatrix} \Phi_{11} & \Phi_{12} & \cdots & \Phi_{1N} \\ \Phi_{21} & \Phi_{22} & \cdots & \Phi_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ \Phi_{N1} & \Phi_{N2} & \cdots & \Phi_{NN} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_N \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}$$

where $\mathbf{\Phi}$ is a matrix of dimension $N \times N$ of components $\Phi_{ij} = \phi(\left\|\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\right\|)$ and $\mathbf{w}$ is $1 \times N$ and $\mathbf{y}$ is $N \times 1$.

If $\mathbf{\Phi}$ is a non singular matrix the solution for the parameters (i.e., the vector of weights $\mathbf{w}$) can be found simply inverting the above relationship

$$\mathbf{w} = \mathbf{\Phi}^{-1}\mathbf{y}$$

The network looks like the following:

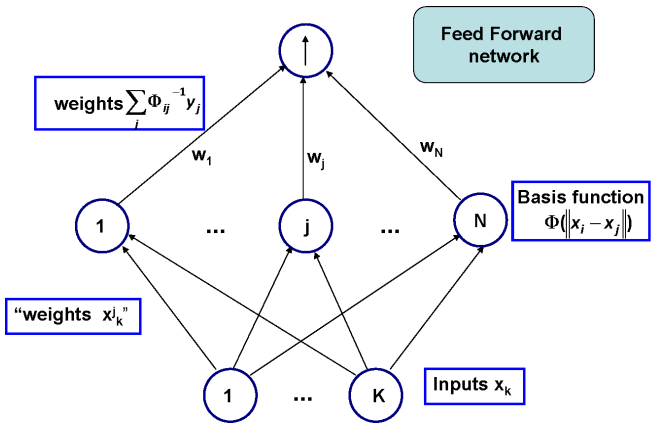


Figure 2.26: RBFN Network.

As can be noticed, the network has an input layer, a hidden layer and an output layer. The input layer broadcasts the coordinates of the input vector to each of the nodes in the hidden layer. Each node in the hidden layer then produces an activation based on the associated radial basis function. The dimensionality of the radial functions is the same as the input data. Finally, the node in the output layer computes a linear combination of the activations of the hidden nodes. How an RBFN reacts to a given input stimulus is completely determined by the activation functions associated with the hidden nodes and the weights associated with the links between the hidden layer and the output layer.

In practice, exact modeling of the training data is not always wanted because in this way a very poor predictive ability would be reached, due to the fact that all details, noise, outliers are modeled.

To have a smooth interpolating function in which the number of basis functions is determined by the fundamental complexity of the data structure, some modifications to the exact interpolation method are required.

1. The number of basis functions, M, is reduced to a lower number, $M \ll N$.

2. Bias parameters are included in the linear sum. These will compensate for the difference between the average value over the data set of the basis function activations and the corresponding average value of the targets.

3. The determination of suitable centers becomes part of training process.

4. Instead oh having a common spread parameter, σ^2, each basis function is given its own width σ_j, whose value is also determined during training.

The main problem with RBFN is that, since they perform exact interpolation, they perform poorly with noisy data. In addition they are not computationally efficient when many training samples are available. Indeed, the network requires

one hidden unit (i.e. one basis function) for each training data. The matrix inversion cost is typically $O(N^3)$.

Chapter 3

Reflectarray antennas

3.1 Introduction

The design, fabrication, and maintenance of large phased arrays for satellite and terrestrial applications is a very challenging and expensive task, especially if very high directivities are required. As a consequence, many techniques have been developed in the scientific and industrial communities in order to simplify the array architecture, reduce the number of control points, or ease the fabrication of the feed network. In this scenario, reflectarrays (antennas in which an active feeder illuminates a large set of passive resonant patches, that collectively scatter the desired beam) have emerged as a powerful and flexible solution to achieve effective beam control capabilities without requiring complex, expensive, and bulk feed networks (unlike phased arrays), and also without yielding the non-conformal geometries of standard parabolic reflector antennas. However, the design of a reflectarray is still a very complex task, especially if high performance in terms of bandwidth and polarization purity is required. As a matter of fact, the design of a reflectarray requires a very accurate knowledge of the relations between the elementary reflectarray antenna (i.e., its shape), the frequency/angle of incidence/polarization of the incoming wave, and the features (magnitude, phase, polarization) of the reflected wave. If the elementary antenna has a complex shape (required to achieve effective cross-polarization control and large bandwidth), no approximate formulas exist to predict such a relation, and expensive full-wave methods or ad hoc numerical techniques are currently required. Unfortunately, the number of simulations required to characterize a single reflectarray cell grows exponentially with its number of degrees-of-freedom, therefore making this approach numerically unfeasible when the reflectarray cell has more than 2/3 geometrical degrees of freedom. In the design methodology, it is therefore of fundamental importance to have techniques capable of efficiently and accurately calculate the reflection coefficient associated with a given geometry of the element in order to calculate the geometry of the element that will provide the desired reflection coefficient. This coefficient is mathematically represented by a

2x2 complex matrix (each entry has a magnitude and a phase), which takes into account the relationships between co-polar and cross-polar components of the incident (due to the feeder) and reflected field. This matrix naturally depends on the geometry of the element, the direction of incidence of the wave (azimuth and elevation) and the operating frequency of the system.

For the design of the reflectarray any possible value of phase-shift must be implemented by varying one parameter in the unit cell (such as the patch size or rotation angle) in order to be able to accurately predict the phase shift and dissipative losses. One of the most important parts in reflectarray analysis is the accurate characterization of the reflective elements (accurate knowledge of phase-shift and polarization losses for each polarization of the field). Curves which relate the phase of the radiated field with certain geometrical parameters of the reflectarray elements are usually adopted.

If the literature, when the arrays had too many elements, the analysis of arrays of rectangular microstrip patches has been carried out assuming the infinite array model and by applying Floquet's theorem, thus reducing the analysis to one periodic cell. This analysis can be used if all the elements in the reflectarray have the same shape. If elements with variable size are used, the reflectarray must be analyzed assuming local periodicity (which is accurate for neighboring patches with smooth variations, assumption which is normally true). In the following the analysis of a microstrip reflectarray with rectangular patches of variable size is carried out. The limitations of this geometry are then highlighted: in order to increase the performance of the reflectarray it is often necessary to use noncanonical patches. However the analysis of more complex shaped requires an increase in CPU time. The proposed method based on an innovative learning-by-example strategy allows analyzing patches with arbitrarily complex shape in an efficient and accurate way. The method is presented in Section 4.

3.2 Single layer reflectarray of rectangular patches

In the following the problem of predicting the power pattern generated by a reflectarray when the dimensions of the patches as well as the substrate thickness and permittivity vary is presented and mathematically modeled.

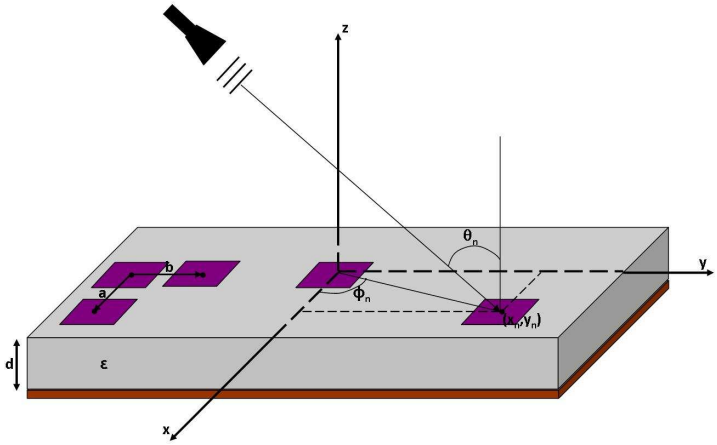


Figure 3.1: Geometry of the reflectarray antenna.

Let us consider a simple reflectarray with N rectangular patches of variable size arranged over a regular lattice, as shown in Fig. 3.1. The origin of the z axis is located at the interface between the dielectric substrate and the ground plane, while the reference point $(x,\, y) = (0, 0)$ is along the perpendicular direction between the feeder and the reflector surface. Both the ground plane and the patches are assumed to be made of perfectly electric conductor PEC. The substrate is assumed to be lossy, homogeneous and isotropic with complex permittivity $\varepsilon = \varepsilon_0 \varepsilon_r \left(1 - j \tan\delta\right)$ and thickness d. The incident plane wave generated by the feeder at an angle (θ_i, ϕ_i) with respect to the reference system, has the following expression

$$\underline{E}_i = \underline{E}_0 e^{jk_0(xu_i + yv_i + z\cos\theta_i)} = \begin{bmatrix} E_0^\theta \\ E_0^\phi \end{bmatrix} e^{jk_0(xu_i + yv_i + z\cos\theta_i)} \tag{3.1}$$

where $\underline{E}_0$ defines the amplitude and polarization of plane wave in free space, $u_i = \sin\theta_i \cos\phi_i$ and $v_i = \sin\theta_i \sin\phi_i$, $k_0 = 2\pi/\lambda_0$ with λ_0 free space wavelength. The total electric field in the region described by the coordinate $z > d$ is given by the sum of the incident field, of the reflected field and of the scattered field:

$$\underline{E}_t = \underline{E}_i + \underline{E}_r + \underline{E}_s \tag{3.2}$$

$\underline{E}_r$ indicates the field reflected from an infinite grounded dielectric slab without the microstrip patches and can be express as

$$\begin{bmatrix} E_r^\theta \\ E_r^\phi \end{bmatrix} = \begin{bmatrix} R_{\theta,\theta} & 0 \\ 0 & R_{\phi,\phi} \end{bmatrix} \begin{bmatrix} E_0^\theta \\ E_0^\phi \end{bmatrix} e^{jk_0(xu_i+yv_i-z\cos\theta_i)} \tag{3.3}$$

where $R_{jj}, j = \theta, \phi$ are the reflection coefficients defined in [41].

When the microstrips are present, a surface current $\underline{J}$ is induced on each conducting element and a scattered field is produced, which can be expressed in terms of the incident field $\underline{E}_i$ and of the scattering coefficients $S_{jk}, j, k = \{\theta, \phi\}$ as

$$\begin{bmatrix} E_s^\theta \\ E_s^\phi \end{bmatrix} = \begin{bmatrix} S_{\theta,\theta} & S_{\theta,\phi} \\ S_{\phi,\theta} & S_{\phi,\phi} \end{bmatrix} \begin{bmatrix} E_0^\theta \\ E_0^\phi \end{bmatrix} e^{jk_0(xu_i+yv_i-z\cos\theta_i)} \tag{3.4}$$

In 3.4 each scattering coefficient S_{jk} is given by $S_{j,k} = \frac{E_s^j(z=0)}{E_i^k(z=0)}; j, k = \{\theta, \phi\}$. If we assume that the reflectarray is subdivided into unit cells and that each cell radiates a spherical wave proportional to the sum of the reflected $\underline{E}_r$ and scattered $\underline{E}_s$ field, the radiation pattern of an N elements reflectarray in the direction (θ, ϕ) is defined as [16]

$$\begin{aligned} \underline{E}(\theta,\phi) &= \frac{e^{-jk_0r}}{r} \sum_{n=1}^{N} \underline{\underline{Q}}(\theta,\phi;\theta_n,\phi_n) \cdot \left[\underline{\underline{R}}(\theta_n,\phi_n) + \underline{\underline{S}}(\theta_n,\phi_n)\right] \\ &\quad \cdot \underline{E}_f(\theta_n,\phi_n)\, e^{jk_0(x_n\sin\theta\cos\phi+y_n\sin\theta\sin\phi)} \end{aligned} \tag{3.5}$$

where $\underline{E}_f$ is the feed pattern function,(x_n, y_n) are the Cartesian coordinates of the n-th patch center (see Fig. 1), (θ_n, ϕ_n) is the direction of arrival of the wave impinging on the n-th patch center, $\underline{\underline{Q}}$ is a term which accounts for the transformation from plane to spherical wave [16]. The jk-th scattering coefficient of the scattering matrix $\underline{\underline{S}}(\theta_n, \phi_n)$ of the n-th patch is defined as [17]:

$$\begin{aligned} S_{\theta,k}(\theta_n,\phi_n) &= -\frac{1}{ab\cos\theta_n} \Big[\underline{\underline{G}}(k_0u_r^n, k_0v_r^n)\, J_x^{(n,\underline{\hat{k}})}(k_0u_r^n, k_0v_r^n)\cos\phi_n\underline{\hat{x}} + \\ &\quad \underline{\underline{G}}(k_0u_r^n, k_0v_r^n)\, J_y^{(n,\underline{\hat{k}})}(k_0u_r^n, k_0v_r^n)\sin\phi_n\underline{\hat{y}}\Big] e^{jk_0\cos\theta_n d} \end{aligned} \tag{3.6}$$

$$\begin{aligned} S_{\phi,k}(\theta_n,\phi_n) &= -\frac{1}{ab} \Big[\underline{\underline{G}}(k_0u_r^n, k_0v_r^n)\, J_x^{(n,\underline{\hat{k}})}(k_0u_r^n, k_0v_r^n)\sin\phi_n\underline{\hat{x}} - \\ &\quad \underline{\underline{G}}(k_0u_r^n, k_0v_r^n)\, J_y^{(n,\underline{\hat{k}})}(k_0u_r^n, k_0v_r^n)\cos\phi_n\underline{\hat{y}}\Big] e^{jk_0\cos\theta_n d} \end{aligned} \tag{3.7}$$

$(k = \{\theta, \phi\})$. ab is the area of the lattice cell (see Fig. 1), $\underline{\underline{G}}$ is the dyadic Green's function in spectral domain, $\underline{J}^n = \sum_{k=\{\theta,\phi\}} J_x^{(n,\underline{\hat{k}})}\underline{\hat{x}} + \sum_{k=\{\theta,\phi\}} J_y^{(n,\underline{\hat{k}})}\underline{\hat{y}}$ is

the current induced on the n-th patch in spectral domain and $\underline{\hat{k}}$ indicates the polarization of the incident field. The jk-th scattering coefficient can be easily computed in an analytical way as indicated in (3.5) for the simple reflectarray reported in Fig. 4.1 (single dielectric substrate, cells with rectangular shapes) by expanding the induced current $\underline{J}^n$ in a set of basis functions $\underline{D}_i^{\left(n,\underline{\hat{k}}\right)}, i = 1, ..., I$ with unknown coefficients $\underline{C}_i^{\left(n,\underline{\hat{k}}\right)}, i = 1, ..., I$

$$J_\chi^{\left(n,\underline{\hat{k}}\right)} = \sum_{i=1}^{I} C_{i,\chi}^{\left(n,\underline{\hat{k}}\right)} D_{i,\chi}^{(n)}, \chi = \{x, y\} \tag{3.8}$$

and by computing the unknown coefficient vector solving the following system

$$\underline{C}^{\left(n,\underline{\hat{k}}\right)} = \left(\underline{\underline{Z}}^{(n)}\right)^{-1} \underline{V}^{\left(n,\underline{\hat{k}}\right)} \tag{3.9}$$

where

$$Z_{il}^{(n)} = -\frac{1}{A} \sum_{m=-\infty}^{\infty} \sum_{u=-\infty}^{\infty} \widetilde{D}_i^{(n)H} \left(-k_{x,m}^{(n)}, -k_{y,mu}^{(n)}\right) \underline{\underline{G}} \left(k_{x,m}^{(n)}, k_{y,mu}^{(n)}\right) \widetilde{D}_l \left(k_{x,m}^{(n)}, k_{y,mu}^{(n)}\right) \tag{3.10}$$

are the elements of the impedance matrix $\underline{\underline{Z}}^{(n)}$, H stands for the conjugate trasnspose and i, l are number of expansions; the i-th entry of the voltage vector $\underline{V}^{\left(n,\underline{\hat{k}}\right)}$ can be computed as described in [41]:

$$V_i^{\left(n,\underline{\hat{k}}\right)} = J_{s0}^{\left(n,\underline{\hat{k}}\right)} \underline{\underline{G}} \left(-k_x^{(n)}, -k_y^{(n)}\right) \widetilde{D}_i^{(n)} \left(-k_x^{(n)}, -k_y^{(n)}\right) e^{jk_0 \cos\theta_n} \tag{3.11}$$

If we consider a more general case in which (a) the substrate may be multilayer, with each layer characterized by a complex permittivity $\varepsilon_w = \varepsilon_0 \varepsilon_{rw} \left(1 - j \tan\delta_w\right)$ and by a thickness d_w (with $w = 1, ..., W$) and (b) the unit cell may have arbitrary complex shape and orientation angle, it is not possible to define a suitable set of basis functions $\underline{D}_i^{\left(n,\underline{\hat{k}}\right)}, i = 1, ..., I$ because they are not available. A full-wave method or an ad-hoc technique for the computation of $\underline{\underline{S}}\left(\theta_n, \phi_n\right)$, based for example on the definition of subdomain basis functions, are required [42].

3.3 How to deal with the limitations of single-layer reflectarray of rectangular patches

The main reason why we have to consider other structures than the single-layer reflectarray of rectangular patches is because of its inherent narrow bandwidth performance [43, 44]. This problem is due to the strongly nonlinear relation existing between the rectangular patches' size and the reflected field. A significant effort has been made in recent years in order to mitigate this problem and elements with linear phase response and broadband behavior have been designed; on one side stacked patches of variable size (see Fig. 3.2) have shown increased bandwidth achieved by combining the resonances of each patch.

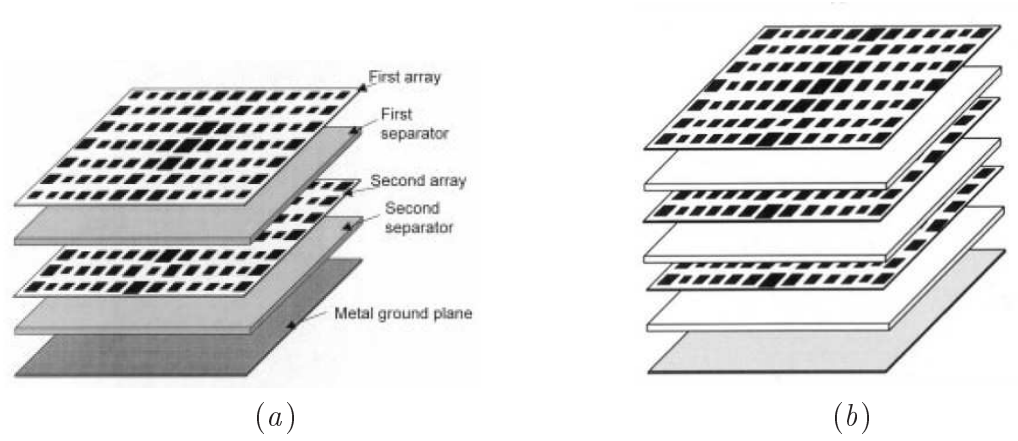


Figure 3.2: (a) Two-layer reflectarray using patches of variable size [45] and (b) three-layer reflectarray using patches of variable size [46].

However, these elements exhibit an increase in the complexity of the manufacturing process [45, 46]. On the other side elements with multiple resonances printed on a single dielectric layer have demonstrated to provide wider bandwidth without complicating the realization process [12, 47], see Fig. 3.3.

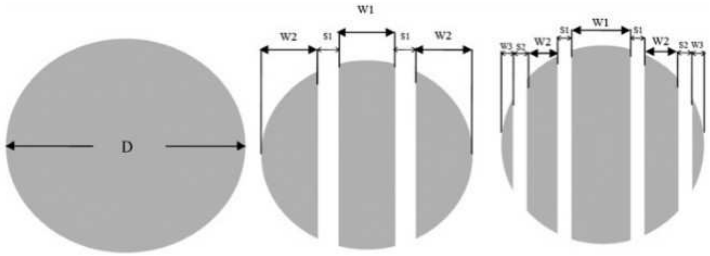


Figure 3.3: Sliced circular fractal derived from the circular patch in [47].

In [48] a three-layer square patch element of variable size was designed to achieve simultaneous coverage at the *Ku* receiving (13.75-14.25 GHz) and transmitting (1.7-12.2 GHz) bands. Multilayer structures have been proposed to achieve larger frequency ratios (ratios of the center frequencies of the upper and lower bands): in this scenario on each layer a different set of elements which operates at a specific frequency band is used. FSS structures have then been introduced to reduce the mutual interference between the elements of different frequency bands [49]. A schematic view of the antenna where an FSS backed reflectarray which works in the Ka-band is located on top of a metal-backed reflectarray working in X-band is reported in Fig. 3.4.

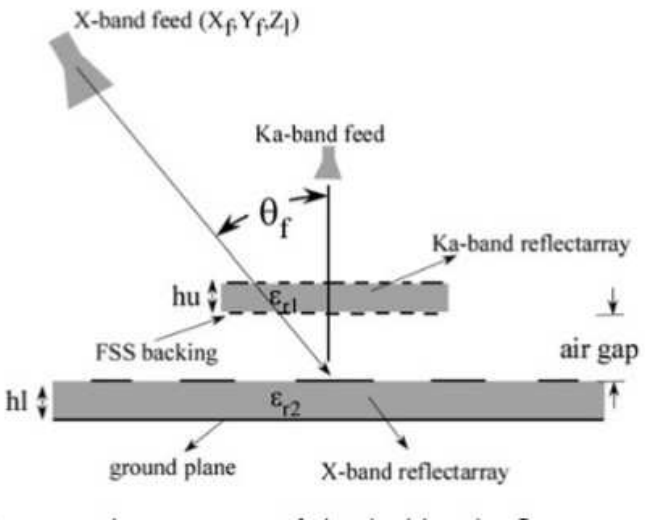


Figure 3.4: Schematic view of dual-band reflectarray presented in [49].

This latter solution becomes however costly and complex to be fabricated; moreover, the gain and efficiency of the lower layer are reduced by the upper layer. Single-layer configurations with different sets of elements displaced on interlaced array grids appear to be the most viable solution to overcome these issues but electromagnetic coupling between elements becomes a problem. For this reason, single layer configurations including a single set of elements capable of achieving a dual-band phase response have been investigated. In [50] a single-layer design with two different sets of elements has been studied; mutual coupling was also taken into account by considering all possible element combinations in a unit cell. The price to be payed was the increased time consumption and design complexity.

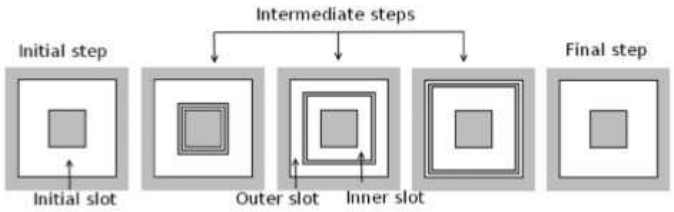


Figure 3.5: Phoenix cycle: evolution of the cell geometry over a complete 360 [deg] cycle [12].

In recent years a new reflectarray cell with cycle evolution and with linear phases with variations in a range bigger than 360 [deg] has been introduced [12]. The problem to be overcome was that at phase transition the geometrical differences of neighboring cells could be particularly sharp, making coupling effects difficult to be accounted for. The new element presented in [12], and reported in Fig. allows achieving a full 360 [deg] phase coverages at both bands with smooth phase responses and low elements loss. It is composed of two square loops and a square patch and it has the capability of coming back to its initial shape after a 360° phase cycle. Differently from the triple square loop element the sizes of the outermost loop and innermost patches are fixed. In [15] a dual-frequency phase-only synthesis method has been applied to the Phoenix element to obtain wider frequency ratio and higher aperture efficiencies at both bands.

The phase response of the "Square Phoenix" and "Rectangular Phoenix" unit cells will be deeply analyzed in Section 4.

3.4 Adopted solution for the computation of scattering coefficients of generic reflectarray unit cells

Many analytical relations based on an equivalent circuit analysis have been derived for the correct computation of the phase of the field radiated by patches with noncanonical shapes [51, 52]. However, these formulas turn out to be rather complicate and analytically onerous, thus scarcely attractive. The Method of Moments in the spectral domain has been demonstrated to be the best approach in terms of efficiency and accuracy, under the assumption of local periodicity (this assumption is valid when variations in neighboring cells are smooth) [53, 42]. This method allows analyzing every configuration and is computationally more efficient compared to other three-dimensional full-wave FEM and FDTD full-wave codes. It assumes that the scattered field can be expressed as a function of the current distribution on each cell, which can be expanded as a summation of basis-function:

$$J_{\chi}^{\left(n,\underline{\hat{k}}\right)} = \sum_{i=1}^{I} C_{i,\chi}^{\left(n,\underline{\hat{k}}\right)} D_{i,\chi}^{(n)} \tag{3.12}$$

where all the terms have been already described in Section 3.2. In general, there are two categories of basis functions $D_{i,\chi}^{(n)}$ used to represent the unknown function, the entire domain and the subdomain basis functions. Entire domain basis function have been derived for dipole, square patch, circular patch, cross, and Jerusalem cross geometries. The most important advantage of entire domain basis functions is that the size of the resulting moment method matrix is usually small and it is thus possible to solve problems for electrically large structures. In contrast, the number of subdomain basis functions required to accurately represent the current is often much larger compared to entire domain basis functions. Moreover, the Fourier transforms of the subdomain basis functions do not decay very rapidly.

For these reasons the Method of Moments may fail whenever subdomain basis functions have to be used to expand the unknown current induced on the metallic cell (this happens for non-canonical shapes), thus not minimizing the number of Floquet harmonics [42]. In [54] a full-wave method based on the transmission line technique has been proposed, which allows the definition of a generalized scattering matrix of each grid and embedding layers (in case of stacked patches). In [55] a simple equivalent-circuit model has been derived in order to compute the response of generic frequency-selective-surface with low computational effort. However, this approach shows good agreement with the more computationally expensive MoM approach only up to the frequency at which grating lobes occur, thus limiting the periodicity of the cells. To the author's best knowledge the full-wave method or ad-hoc numerical technique presented in the literature may

become an unaffordable solution in terms of computational time whenever high performances in terms of bandwidth, radiation efficiency, polarization purity are required. The reason why this happens is because the more complex the problem to be addressed (in terms of antenna requirements), the higher the number of degrees of freedom of the unit cell required. However, when the number of degrees of freedom increases (i.e., parameters describing the shape of the patch, frequency and angles of incidence on the incoming plane waves), the number of simulations required increases, too.
The method proposed in order to address the problem of efficiently computing the response of arbitrary complex reflectarray elements is based on an innovative and customized statistical learning (SL) technique. The idea is to recast the problem of computing the scattering matrix of a patch element given its specific features as a regression problem, by processing the information embedded in a set of I/O pairs in order to predict the output of unknown configurations.

The evaluation of the scattering coefficients of generic reflectarray unit cells (i.e., featuring an arbitrary number of DoFs) is firstly re-cast as a regression problem and then solved with a learning-by-example (LBE) strategy able to exploit the information provided by a reduced set of FW simulations (namely the "examples") performed once and off-line. In order to compute the "examples" an EM analysis tool based on a mode-matching method between the free space Floquet's mode and the aperture or patch modes of the single cell elements is used as a FW solver [56][57]. The idea beyond this method is that each layer of the reflectarray can be seen as a capacitive (periodic distribution of metallic patches on dielectric layers), inductive (periodic distribution of holes on metallic sheets, with or without dielectric support), or mixed (multi-layer constituted by capacitive as well as inductive grids) structure. The generalized scattering matrix of each structure becomes huge if the number of interacting Floquet modes increases (as is the case of mixed structures) but the method presented in [56] allows to obtain a linear matrix system by expressing the voltages and currents on all the grid generators and by simultaneously applying the boundary conditions in the spectral domain on each of them.

Chapter 4

Efficient prediction of the EM response of reflectarray antennas by an advanced statistical learning method

The following work has been submitted for publication in the IEEE Transactions on Antennas and Propagation. The problem being addressed is the efficient and accurate prediction of the electromagnetic response of complex-shaped reflectarray elements. The addressed problem is important to the Antennas and Propagation community since the synthesis of high performance reflectarrays, even more when wideband operations and/or a careful control of the cross-polarization components of the reflected field are needed, needs complex patch shapes because of the wider set of degrees of freedom (DoFs) potentially enabling an enhanced control of the antenna scattering properties. Unfortunately, designing a reflectarray featuring complicated element geometries often turns out to be a very challenging task in practice. To determine the optimal shape of each reflectarray element (i.e., setting the DoFs of the reflectarray patches), the relationships between the descriptors of both the unit cell (e.g., geometry/size of the patch metallizations) and of the illumination (e.g., the polarization/frequency/angle-of-arrival of the incident field) with the associated scattering coefficients must be known, but this knowledge is analytically available only for "simple" unit cells described by few DoFs. Otherwise, scattering matrix-vs-descriptors look-up tables (LUTs), which are off-line computed through extensive full-wave (FW) simulations, are usually built, but the exponential grow of the number of entries of these latter with the DoFs of the unit cells, prevent such an approach when dealing with advanced reflectarray geometries characterized by arbitrary variations of many descriptors because of the infeasible generation and storage of the associated unit cell scattering response databases (UCS-DBs). Therefore, innovative methodologies for the quasi- or real-time prediction of the electromagnetic response of complex

reflectarray elements are necessary.

The novelties of the presented work over the existing work comprise (i) the introduction of a computationally efficient, reliable/accurate, and flexible strategy to predict the scattering response of reflectarray elements featuring arbitrarily complex unit cells that potentially enables their use in next-generation and more demanding reflectarray designs; (ii) the development and customization to the vectorial case of an advanced OK technique for the prediction of complex valued scattering matrices of periodic EM planar structures, thus useful not only for reflectarrays, but also generalisable to analogous electromagnetic engineering problems (e.g., the analysis of frequency-selective surfaces and metasurfaces); (iii) the development of a numerical tool that, whether integrated within a system-by-design (SbD) loop, could enable the optimal synthesis of next-generation reflectarray antennas with controlled co- and cross-polar radiation patterns; and (iv) the derivation of operative guidelines on the achievable time saving and the arising prediction accuracy vs. the training set size for the exploitation of such a OK meta-modeling in reflectarray response prediction.

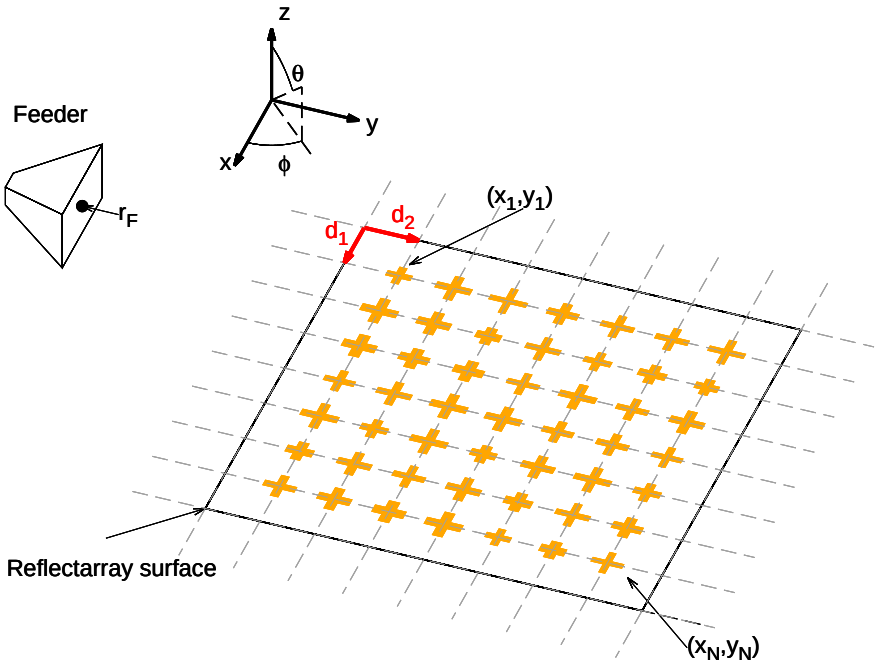


Figure 4.1: Sketch of the reflectarray antenna.

4.1 Problem Statement

Let us consider a microstrip reflectarray consisting of a planar array of N patches displaced over the xy-plane in a regular lattice with unit cell $\mathbf{d}_1 \times \mathbf{d}_2$ (Fig. 4.1) on a grounded multilayer substrate. Each n-th $(n = 1, ..., N)$ array element is described by B $DoFs$ $\mathbf{g}(n) \triangleq \left\{ g^{(b)}(n) ; b = 1, ..., B \right\}$. The design of the patch arrangement is usually carried out as the synthesis of the set of N descriptor vectors, $\mathbf{G} = \{\mathbf{g}(n) \in \wp ; n = 1, ..., N\}$, $\wp$ being the set of admissible variations of the unit-cell geometry with respect to a reference one, such that the field radiated by the reflectarray, $\mathbf{E}(\theta, \varphi; f)$, is as close as possible to a user-defined one, $\mathbf{E}^{ref}(\theta, \varphi; f)$. More in detail, the field distribution $\mathbf{E}(\theta, \varphi; f)$ is given by [16][17][41]

$$\begin{aligned} \mathbf{E}(\theta, \varphi; f) \;=\; & \sum_{n=1}^{N} \{ [\mathcal{R}(\theta_n, \varphi_n; f) + \mathcal{S}(\theta_n, \varphi_n; f, \mathbf{g}(n))] \\ & \cdot \mathbf{E}_F(\theta_n, \varphi_n; f) \exp(jk_0 \mathbf{r}_n \cdot \hat{\mathbf{r}}) \} \end{aligned} \tag{4.1}$$

where f is the working frequency, $\hat{\mathbf{r}} \triangleq (\sin\theta\cos\varphi,\ \sin\theta\sin\phi,\ \cos\theta)$, $\mathbf{r}_n = (x_n, y_n, 0)$ is the location of the n-th patch element, $k_0 = \frac{2\pi f}{c_0}$ is the free-space wavenumber (c_0 being the speed of light), and (θ_n, φ_n) are the elevation angle and the azimuth one of the direction of incidence from the feed to the n-th element, respectively,

while

$$\mathbf{E}_F(\theta_n,\varphi_n;f) = \frac{|\mathbf{r}_F|}{|\mathbf{r}_n-\mathbf{r}_F|}\frac{E_F(\theta_n,\varphi_n;f)}{E_F(0,0;f)}\exp\left(jk_0|\mathbf{r}_n-\mathbf{r}_F|-|\mathbf{r}_F|\right) \left[\cos\varphi_n\widehat{\boldsymbol{\theta}}+\sin\varphi_n\widehat{\boldsymbol{\varphi}}\right] \quad (4.2)$$

is the field pattern radiated by the feed on the n-th element, $\mathbf{r}_F$ and $E_F(\theta,\varphi;f)$ being the feeder position and the element factor, respectively. Therefore, the synthesis of a reflectarray layout radiating a field distribution (4.1) fitting the desired one $\mathbf{E}^{ref}(\theta,\varphi;f)$ requires, for each n-th ($n=1,...,N$) layout element, the knowledge of both the plane wave reflection matrix,
$\mathcal{R}(\theta_n,\varphi_n;f)=\{R_{pq}(\theta_n,\varphi_n;f);\, p,q=\{\theta,\varphi\}\}$, and the scattering matrix,
$\mathcal{S}(\theta_n,\varphi_n;f,\mathbf{g}(n))=\{S_{pq}(\theta_n,\varphi_n;f,\mathbf{g}(n));\, p,q=\{\theta,\varphi\}\}$ as suggested by (4.1). Towards this end, let us notice that the entries of the matrix $\mathcal{R}$ do not generally depend on the patch elements and they are usually available in closed-form [16][17][41]. Otherwise, the scattering matrix $\mathcal{S}$ heavily depends on the shape/layout of the reflectarray unit cells and there are no available closed-form expressions for the associated entries except for simple geometries (e.g., rectangular patches [16][17]). Thus, it is generally needed to solve the following estimation problem

> **Scattering Matrix Estimation Problem**. Find the estimation function $\widehat{\mathcal{S}}(\mathbf{z})$ such that $\widehat{\mathcal{S}}(\mathbf{z})\approx\mathcal{S}(\mathbf{z})$, $\mathbf{z}\in Z$.

where

$$\mathbf{z}\triangleq[\theta,\varphi,f,\mathbf{g}] \quad (4.3)$$

is an *input* vector of dimension $B+3$ in the feasibility space Z ($Z\triangleq\{\theta\in[\theta_{\min},\theta_{\max}];\ \varphi\in[\varphi_{\min},\varphi_{\max}];\ f\in[f_{\min},f_{\max}];\ \mathbf{g}\in\wp\}$).

4.2 *LBE*-Based Prediction of the Reflectarray Unit-Cell Response

A direct approach to address the "*Scattering Matrix Estimation Problem*" when dealing with reflectarray elements for which no analytical models are available is that of exploiting *FW* numerical solvers to exhaustively populate huge *LUT*s - to be used in the design phase [15][18][20] - mapping the I/O relationship between the *input* $\mathbf{z}$ and the electromagnetic response function $\widehat{\mathcal{S}}(\mathbf{z})$ (i.e., the *output*). Towards this end, the following two steps are carried out:

- the elevation θ, the azimuth φ, the working frequency f, and the *DoF*s of the array element $\mathbf{g}$ are first discretized in V $[\theta_v = \theta_{\min} + (v-1)\,\Delta\theta;\ v = 1,...,V;\ \Delta\theta = \frac{\theta_{\max}-\theta_{\min}}{V-1}]$, H $[\varphi_h = \varphi_{\min} + (h-1)\,\Delta\varphi;\ h = 1,...,H;\ \Delta\varphi = \frac{\varphi_{\max}-\varphi_{\min}}{H-1}]$, W $[f_w = f_{\min} + (w-1)\,\Delta f;\ w = 1,...,W;\ \Delta f = \frac{f_{\max}-f_{\min}}{W-1}]$, and L $[g_{l_b}^{(b)} = g_{\min}^{(b)} + (\Im\{l_b\}-1)\,\Delta g^{(b)},\ l_b = 1,...,L_b;\ b = 1,...,B;\ L = \prod_{b=1}^{B} L_b;\ \Delta g^{(b)} = \Delta g^{(1)},\ \Delta g^{(1)} \triangleq \frac{g_{\max}^{(1)}-g_{\min}^{(1)}}{L_1-1}]$ quantized values, respectively;

- a *FW* simulation for each m-th ($m = 1,...,M;\ M = V \times H \times W \times L$) setup of the *input* vector $\mathbf{z}_m$ ($\mathbf{z}_m \triangleq [\theta_v, \varphi_h, f_w, \mathbf{g}_l]$, $m = H \times W \times L \times (v-1) + W \times L \times (h-1) + L \times (w-1) + l;\ v = 1,...,V;\ h = 1,...,H;\ l = 1,...,L;\ w = 1,...,W$) is carried out to determine the corresponding *output* function $\mathcal{S}(\mathbf{z}_m)$, thus filling the m-th entry of the I/O *UCS-DB* $\mathcal{D} \triangleq \{\mathbf{z}_m, \mathcal{S}(\mathbf{z}_m);\, m = 1,...,M\}$ [20].

Despite the simplicity and the accuracy in computing the function $\mathcal{S}(\mathbf{z})$ thanks to the use of *FW* solvers, such an approach has actually a limited applicability since the size of the resulting database, M, increases proportionally with the number of *DoF*s describing the shape of the reflectarray cell-element, thus making both the storage and the computation time, T_{tot}^{FW} ($T_{tot}^{FW} \triangleq M \times T_{sim}^{FW}$, T_{sim}^{FW} being the *CPU*-time for the computation of a single $\mathcal{S}$ matrix) unmanageable when complex geometries are at hand.
To deal with complex patch shapes, suitable for fitting more challenging radiation constraints, thus overcoming the storage/computational-issues of database-based methods, the use of a statistical *LBE* method based on *OK* [21] is proposed hereinafter. Such a choice is motivated by several reasons, the most important ones being (*i*) the generalization capabilities of *LBE* strategies that theoretically enable an accurate prediction of the *output* function $\widehat{\mathcal{S}}(\mathbf{z})$ just starting from *few* I/O "examples", $\mathcal{T} \triangleq \{\mathbf{z}_u, \mathcal{S}(\mathbf{z}_u);\, u = 1,...,U\}$, collectively indicated as "*training set*", of dimension significantly lower than that of a standard I/O database (i.e., $U \ll M$). This latter feature guarantees a non-negligible time-saving with respect to the whole filling of a *FW*-based database; (*ii*) unlike standard interpolation techniques, reliable predictions also without the *a-priori* knowledge of the

functional properties of $\mathcal{S}(\mathbf{z})$; (*iii*) the capability of the *OK* to deal with noiseless training data such as for the estimation of the scattering matrix problem [21][22][23]; (*iv*) unlike other *LBE* methods, the effective self-calibration/setup of the *OK* control hyper-parameters during the training phase [21]; (*v*) the good generalization capabilities and the numerical efficiency of the *OK* already assessed in very large problems [21][22][23], as well.

The *Scattering Matrix Estimation Problem* is solved with the *OK* method according to the following guidelines. First the entries of the estimated scattering matrix $\widehat{\mathcal{S}}^{OK}(\mathbf{z})$ are expressed in terms of the $I = 8$ components of the vectorial *auxiliary OK* prediction function, $\boldsymbol{\chi}(\mathbf{z}) \triangleq \{\chi_i(\mathbf{z}), i = 1, ..., I\}$: $\widehat{S}^{OK}_{\theta\theta}(\mathbf{z}) = \chi_1(\mathbf{z}) + j\chi_2(\mathbf{z})$, $\widehat{S}^{OK}_{\theta\varphi}(\mathbf{z}) = \chi_3(\mathbf{z}) + j\chi_4(\mathbf{z})$, $\widehat{S}^{OK}_{\varphi\theta}(\mathbf{z}) = \chi_5(\mathbf{z}) + j\chi_6(\mathbf{z})$, and $\widehat{S}^{OK}_{\varphi\varphi}(\mathbf{z}) = \chi_7(\mathbf{z}) + \chi_8(\mathbf{z})$. Such an auxiliary function is defined as follows [21][22]

$$\boldsymbol{\chi}(\mathbf{z}) = \boldsymbol{\beta}(\boldsymbol{\eta}) + [\boldsymbol{\gamma}(\mathbf{z};\boldsymbol{\eta})]^{*} [\Gamma(\boldsymbol{\eta})]^{-1} (\Psi - \mathbf{1}_U \boldsymbol{\beta}(\boldsymbol{\eta})), \tag{4.4}$$

where $\mathbf{1}_U$ is an all-ones column vector of length U, $\boldsymbol{\beta}(\boldsymbol{\eta})$ is the vector of the *OK* regression parameters given by

$$\boldsymbol{\beta}(\boldsymbol{\eta}) = \left(\mathbf{1}_U^{*} [\Gamma(\boldsymbol{\eta})]^{-1} \mathbf{1}_U\right)^{-1} \mathbf{1}_U^{*} [\Gamma(\boldsymbol{\eta})]^{-1} \Psi \tag{4.5}$$

where Ψ

$$\Psi \triangleq \{\mathbb{R}[S_{pq}(\mathbf{z}_u)], \mathbb{I}[S_{pq}(\mathbf{z}_u)]; p, q = \{\theta, \varphi\}, u = 1, .., U\} \tag{4.6}$$

is the matrix comprising the real part, $\mathbb{R}[.]$, and the imaginary one, $\mathbb{I}[.]$, of the off-line *FW* computed scattering matrix coefficients belonging to the *training* set $\mathcal{T}$. Moreover, $\boldsymbol{\gamma}(\mathbf{z};\boldsymbol{\eta})$ is a U-dimensional vector whose u-th ($u = 1, ..., U$) entry, $\gamma_u(\mathbf{z};\boldsymbol{\eta})$, is the correlation value between the reflectarray unit-cell descriptor $\mathbf{z}$ and the u-th "example" input setup $\mathbf{z}_u$ given by

$$\gamma_u(\mathbf{z};\boldsymbol{\eta}) \triangleq \exp(-\boldsymbol{\eta}^{*} \cdot |\mathbf{z} - \mathbf{z}_u|) \tag{4.7}$$

when an *exponential* correlation model is assumed, $\boldsymbol{\eta} \triangleq \{\eta_b, b = 1, ..., B+3\}$ being the set of the *OK* control coefficients [21][22][23]. Furthermore, $\Gamma(\boldsymbol{\eta})$is the $U \times U$ matrix of the auto-correlation values, whose u-th ($u = 1, ..., U$) column is the vector $\boldsymbol{\gamma}(\mathbf{z}_u;\boldsymbol{\eta})$.

The entries of $\widehat{\mathcal{S}}^{OK}(\mathbf{z})$ are then inferred as a function of the vectorial predictor function $\boldsymbol{\chi}(\mathbf{z})$ in (4.4) from the knowledge of the training set $\mathcal{T} \triangleq \{\mathbf{z}_u, \mathcal{S}(\mathbf{z}_u); u = 1, ..., U\}$ once the optimal value of the control vector $\boldsymbol{\eta}$ in (4.7) is specified [21][22][23]. Unlike many popular *LBE* techniques, which need time-consuming trial-and-error calibration procedures [33][32][34][58], the calibration step in the *OK* comes from an effective self-tuning process [21][22] where the optimal setup, $\boldsymbol{\eta}^{opt}$, is determined by looking for the minimum of the *concentrated likelihood* function $\Phi(\boldsymbol{\eta})$ defined as

$$\Phi(\boldsymbol{\eta}) = \left\{ \frac{\sqrt[U]{det[\Gamma(\boldsymbol{\eta})]}\, tr[\mu(\boldsymbol{\eta})^{*} \mu(\boldsymbol{\eta})]}{U} \right\} \tag{4.8}$$

where $\mu(\boldsymbol{\eta}) \triangleq \{\kappa[\Gamma(\boldsymbol{\eta})]\}^{-1}[\Psi - \mathbf{1}_U \boldsymbol{\beta}(\boldsymbol{\eta})]$, $\kappa[\cdot]$ being the Cholesky factorization operator, while $det[\cdot]$ and $tr[\cdot]$ stand for the *determinant* and the *trace* operators, respectively. Finally, the search for $\boldsymbol{\eta}^{opt} = \arg\min_{\boldsymbol{\eta}}\{\Phi(\boldsymbol{\eta})\}$ is efficiently carried out by means of a standard technique such as the *BOXMIN* multivariate dichotomy algorithm [59].

It is worth pointing out that such an *OK*-based procedure for solving the *Scattering Matrix Estimation Problem* presents some key features/advantages that include (a) the straightforwardly exploitation of the multi-dimensional nature of (4.4) for the prediction of the scattering matrix $\mathcal{S}(\mathbf{z})$, (b) the self-setup of the *OK* control parameters (4.8) that avoids expensive trial-and-error calibration procedures, and (c) an implicit and effective processing of noiseless data, since the *OK* predictor exactly fits the training samples (i.e., $\widehat{\mathcal{S}}^{OK}(\mathbf{z}_u) = \mathcal{S}(\mathbf{z}_u)$, $u = 1, ..., U$ [21][22][23]).

4.3 Numerical Results

This section is aimed at numerically validating the proposed *OK*-based approach for the solution of the *Scattering Matrix Estimation Problem* as well as evaluating its performance in comparison with state-of-the-art prediction methods, as well. Towards this end, the *OK* performance will be assessed by means of the *matrix norm* error Ξ_1 and the the *phase mean squared* error Ξ_2 defined as follows

$$\Xi_1 \triangleq \frac{1}{M}\sum_{m=1}^{M}\frac{\left\|\widehat{\mathcal{S}}^{OK}(\mathbf{z}_m)-\mathcal{S}(\mathbf{z}_m)\right\|^2}{\left\|\mathcal{S}(\mathbf{z}_m)\right\|^2} \tag{4.9}$$

where $\mathcal{S}$ denotes the exact *FW*-computed scattering matrix/entries, $\|\cdot\|$ being ℓ_2-norm, and

$$\Xi_2 = \frac{1}{4M}\sum_{m=1}^{M}\sum_{p,q=\{\theta,\varphi\}}\left|\frac{1}{\pi}\arg\left[\frac{\widehat{S}_{pq}^{OK}(\mathbf{z}_m)}{S_{pq}(\mathbf{z}_m)}\right]\right|^2 \tag{4.10}$$

where the π normalization accounts for the fact that the phase is expressed in radians, while the coefficient $\frac{1}{4}$ refers to the four entries of the scattering matrix. The values of these metrics allow one to quantitatively evaluate the prediction accuracy of the method (4.9) and its reliability in estimating the phase of the entries of the scattering matrix (4.10), which is the key parameter in state-of-the-art reflectarray design methods [11][15]. On the other hand, a success index of using a *LBE*-based strategy is its computational efficiency for a given degree of prediction accuracy. More specifically, the time saving with respect to the time required by the *FW* approach to fill the same size *I/O UCS-DB* $\mathcal{D}$

$$\Delta T \triangleq \left|\frac{T_{tot}^{FW}-T_{tot}^{OK}}{T_{tot}^{FW}}\right| \tag{4.11}$$

where T_{tot}^{OK} is the time required by the *OK* to determine the M entries of $\mathcal{D}$ given by

$$T_{tot}^{OK} \triangleq T_{set}^{FW} + T_{train}^{OK} + T_{test}^{OK} \tag{4.12}$$

where $T_{set}^{FW} \triangleq U \times T_{sim}^{FW}$ is the time for *FW*-computing the U entries of the training set $\mathcal{T}$, T_{train}^{OK} is the time for the *OK* training process [i.e., the computation of (4.8) to be substituted in (4.4)], and $T_{test}^{OK} \triangleq (M-U)\times T_{sim}^{OK}$ is the time needed by the *OK*-based approach to predict the remaining $M-U$ entries of $\mathcal{D}$, T_{sim}^{OK} being the time of a single *OK* prediction.[1]

The first experiment is concerned with the scattering matrices of a reflectarray unit cell printed on a multi-layer dielectric substrate (Tab. 4.I) with square lattice periodicity ($\mathbf{d}_1 = \frac{\lambda_0}{3}\widehat{\mathbf{x}}$, $\mathbf{d}_2 = \frac{\lambda_0}{3}\widehat{\mathbf{y}}$, λ_0 being the wavelength at the central

[1] For the sake of fairness, all the simulation time refer to non-optimized Matlab implementations executed on a single-core CPU running at 2.20 GHz.

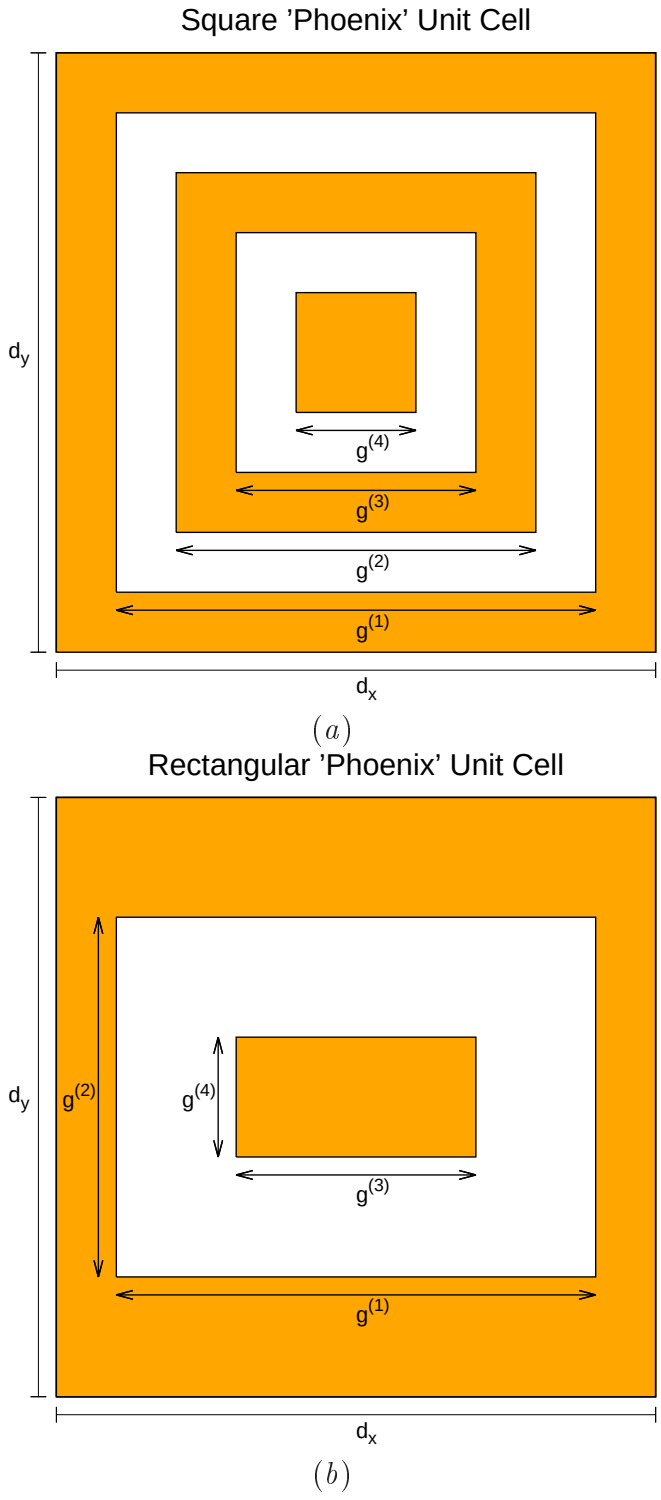


Figure 4.2: Geometry of (a) the $B = 4$ "*Square Phoenix*" unit cell and (b) the $B = 4$ "*Rectangular Phoenix*" unit cell.

Table 4.1: *Numerical Assessment* (*Square/Rectangular Phoenix unit cell*, $d_x = d_y = \frac{\lambda_0}{3}$) - Multilayer dielectric substrate features.

Layer	*Thickness* $[\lambda_0]$	*Relative Permittivity*
Bottom	5.84×10^{-3}	$2.8 - j1.96 \times 10^{-2}$
Middle	2.61×10^{-1}	$1.03 - j3.09 \times 10^{-3}$
Top	5.84×10^{-3}	$2.8 - j1.96 \times 10^{-2}$

frequency f_0) and comprising multiple concentric square metallic rings/slots [i.e., the "Square Phoenix" cell - Fig. 4.2(a)] [2][12][13][14]. Such a geometry, which features $B = 4$ geometrical DoFs [Fig. 4.2(a)], is known to guarantee wide phase variations with smooth geometrical changes and to be suitable for wideband applications [2][12][13][14]. Since no analytical model is available [2][12][13][14], the electromagnetic response of the corresponding reflectarray is numerically computed by first discretizing its DoFs according to the following setup: $\theta_{\min} = 0$ [deg], $\theta_{\max} = 40$ [deg], $V = 9$, $\varphi_{\min} = 0$, $\varphi_{\max} = 45$, $H = 4$, $f_{\min} = 0.9 f_0$, $f_{\max} = 1.1 f_0$, $W = 3$, $L_b = 32$, $g_{\min}^{(b)} = 0$, $g_{\max}^{(b)} = \frac{\lambda_0}{3}$ $(b = 1, ..., B)$, and

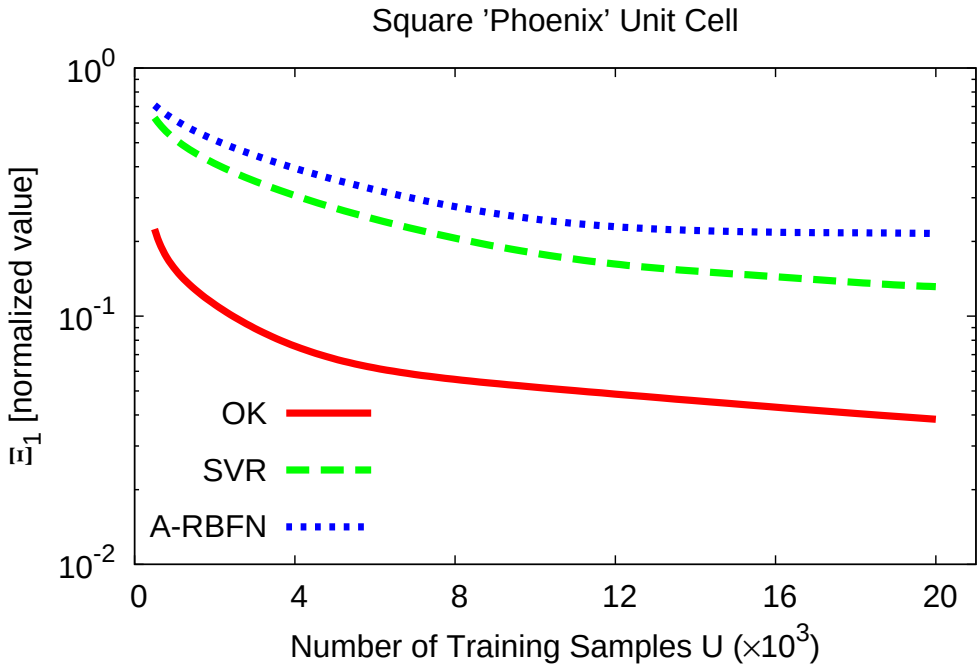
$$\Im\{l_b\} = \begin{cases} l_b & b = 1 \\ l_b \times \mathcal{H}(l_{b-1} - l_b) & b = 2, ..., B \end{cases}, \tag{4.13}$$

$\mathcal{H}(\cdot)$ being the Heaviside function, then applying an EM analysis tool based on Floquet hypotheses as a FW solver [56][57] to fill the whole database of $L \approx 2.85 \times 10^4$ different cell descriptor configurations. It is worth remarking that, despite the coarse sampling of the solution space (only 9 angles in elevation and 4 angles in azimuth) and the choice of an efficient FW method (i.e., $T_{sim}^{FW} \approx 1.20 \times 10^2$ [s]), the computation of the $M \approx 3.1 \times 10^6$ entries of $\mathcal{D}$ would require $T_{tot}^{FW} \approx 3.69 \times 10^8$ [s] (i.e., ≈ 11.7 years).
In order to predict the entries of $\mathcal{S}(\mathbf{z})$, the preliminary *offline* step (likewise any other LBE method) is the choice of the U entries of the training set $\mathcal{T}$ that populate Ψ in (4.6). Towards this end, several advanced algorithms (e.g., exploiting feature extraction and adaptive selection of the U configurations [58]) could be adopted in principle. Owing to the focus of this validation (i.e., the analysis of the potentialities of a "bare" implementation of the proposed OK strategy), a *uniform random* sampling approach has been adopted and the U entries of Ψ have been randomly selected from the M configurations in $\mathcal{D}$. The next step has been the OK self-calibration, which has been performed according to (4.8) to deduce $\boldsymbol{\eta}^{opt}$. This actually completed the *training* phase of the method, since the prediction (4.4) has been then carried out by simple substitution (see Sect. 4.2).
The plot of the resulting matrix norm error with respect to U shows that, as expected, the OK accuracy monotonically improves with the size of $\mathcal{T}$ [e.g.,

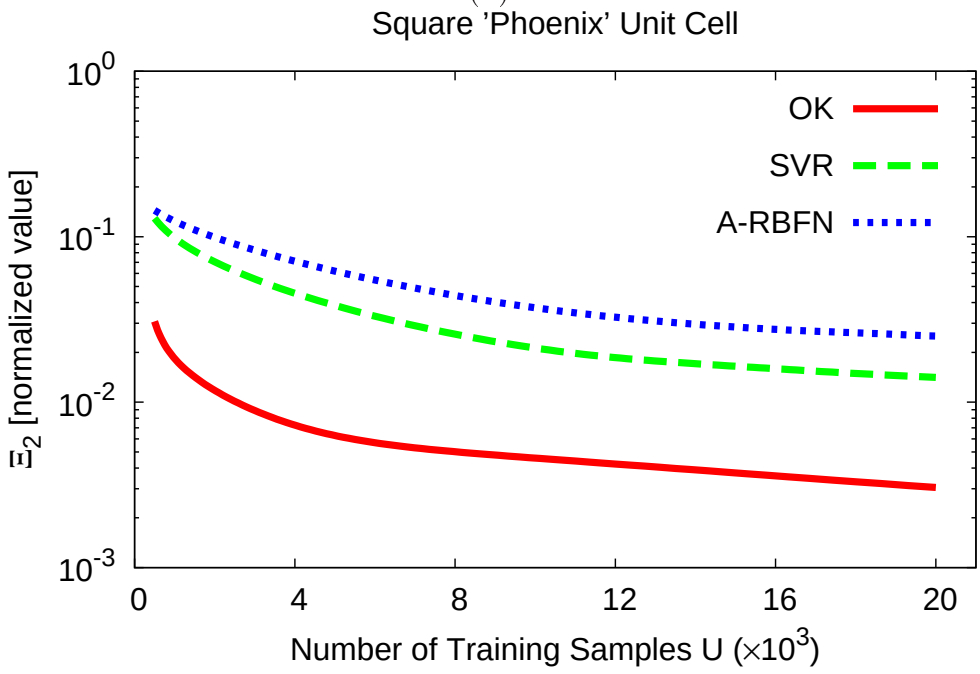
$\frac{\Xi_1^{OK}\big|_{U=2.0\times10^4}}{\Xi_1^{OK}\big|_{U=5.0\times10^2}} \approx 16.9\%$ - Fig. 4.3(a)]. Moreover, the arising value of Ξ_1 is low [i.e., $\Xi_1^{OK}\big|_{U=2.0\times10^4} \approx 3.8\times10^{-2}$ - Fig. 4.3(a)] even though $U \ll M$ (i.e., $\frac{U}{M} \approx 6.4\times10^{-3}$). Such a result is even more impressive when compared to the accuracy level for the same setup when applying competitive state-of-the-art *LBE* strategies based on Support Vector Regression (*SVR*) [33][32][34][58][60] and Augmented Radial Basis Function Network (*A-RBFN*) paradigms [13][36]. Indeed, both techniques yield a significantly worse Ξ_1 regardless of U [e.g., $\frac{\Xi_1^{SVR}}{\Xi_1^{OK}}\Big|_{U=2.0\times10^4} \approx 342\%$ and $\frac{\Xi_1^{A-RBFN}}{\Xi_1^{OK}}\Big|_{U=2.0\times10^4} \approx 552\%$ - Fig. 4.3(a)]. This outcome can be theoretically motivated from the fact that (a) unlike *OK*, *SVR* strategies do not guarantee to fit the training samples (i.e., in general $\widehat{\mathcal{S}}^{SVR}(\mathbf{z}_u) \neq \mathcal{S}(\mathbf{z}_u)$, $u = 1, ..., U$), thus they are less effective when noiseless deterministic data (such as those produced by a *FW* solver) are at hand [33][32][34][58]; (b) thanks to its semi-parametric nature and the self-tuned configuration parameters, *OK* affords a greater flexibility than *A-RBFN* and this results in more accurate predictions [61].

Now, let us analyze the capabilities of the *OK*-based method in predicting $\angle\mathcal{S}(\mathbf{z})$ in view of its exploitation for the reflectarrays synthesis [11][15]. The plot of Ξ_2 versus U in Fig. 4.3(b) shows that the error is smaller than those from the *SVR* and the *A-RBFN* for any size of the training set [e.g., $\frac{\Xi_2^{SVR}}{\Xi_2^{OK}}\Big|_{U=5.0\times10^2} \approx 420\%$ and $\frac{\Xi_2^{A-RBFN}}{\Xi_2^{OK}}\Big|_{U=5.0\times10^2} \approx 469\%$; $\frac{\Xi_2^{SVR}}{\Xi_2^{OK}}\Big|_{U=2.0\times10^4} \approx 466\%$ and $\frac{\Xi_2^{A-RBFN}}{\Xi_2^{OK}}\Big|_{U=2.0\times10^4} \approx 833\%$]. Moreover, the phase behaviour turns out to be more accurately (in percentage) estimated than the $\mathcal{S}(\mathbf{z})$ matrix [i.e., $\Xi_2^{OK}\big|_{U=2.0\times10^4} \approx 3.05\times10^{-3}$ vs. $\Xi_1^{OK}\big|_{U=2.0\times10^4} \approx 3.8\times10^{-2}$ - Fig. 4.3(b)]. In order to give the interested readers an idea of the correspondence between the figures of merit in Fig. 4.3 and the associated prediction capabilities, the plots of the magnitude and phase of $S_{\theta\theta}(\mathbf{z})$ versus θ when $f = f_0$ and $\varphi = 45$ [deg], $U = 2.0\times10^4$ being the size of the training set, for two sample geometries - not belonging to $\mathcal{T}$ - of the unit cell of the reflectarray [i.e., *Square Phoenix Cell Config. 1* - Fig. 4.4(a); *Square Phoenix Cell Config. 2* - Fig. 4.4(b)] with descriptors in Tab. 4.II are reported in Fig. 4.4(c) - Fig. 4.4(e) and Fig. 4.4(d) - Fig. 4.4(f), respectively.

As expected, the *OK* strategy outperforms other state-of-the-art techniques in predicting the scattering magnitude [i.e., $\left|S_{\theta\theta}^{FW}(\mathbf{z}) - S_{\theta\theta}^{OK}(\mathbf{z})\right| < 0.9$ [dB], $\left|S_{\theta\theta}^{FW}(\mathbf{z}) - S_{\theta\theta}^{SVR}(\mathbf{z})\right| < 2.4$ [dB], $\left|S_{\theta\theta}^{FW}(\mathbf{z}) - S_{\theta\theta}^{A-RBFN}(\mathbf{z})\right| < 6.2$ [dB] - Figs. 4(c)-4(d)] and the phase [i.e., $\left|\angle S_{\theta\theta}^{FW}(\mathbf{z}) - \angle S_{\theta\theta}^{OK}(\mathbf{z})\right| < 0.5$ [deg], $\left|\angle S_{\theta\theta}^{FW}(\mathbf{z}) - \angle S_{\theta\theta}^{SVR}(\mathbf{z})\right| < 15$ [deg], $\left|\angle S_{\theta\theta}^{FW}(\mathbf{z}) - \angle S_{\theta\theta}^{A-RBFN}(\mathbf{z})\right| < 26$ [deg] - Figs. 4(e)-4(f)]. These results, besides visually confirming the quantitative indications coming from Figs. 3(a)-3(b), also highlight the effectiveness of the *OK*-based predictor to reliably model the $\mathcal{S}(\mathbf{z})$ variations [e.g., Fig. 4.4(e)] with negligible [e.g., Fig. 4.4(c)] or only slight [i.e., Fig. 4.4(d)] deviations



(a)



(b)

Figure 4.3: *Numerical Assessment* (*Square Phoenix unit cell*, $d_x = d_y = \frac{\lambda_0}{3}$, $U \in [5.0 \times 10^2, 2.0 \times 10^4]$) - Behavior of (a) Ξ_1 and (b) Ξ_2 versus the size of the training set U.

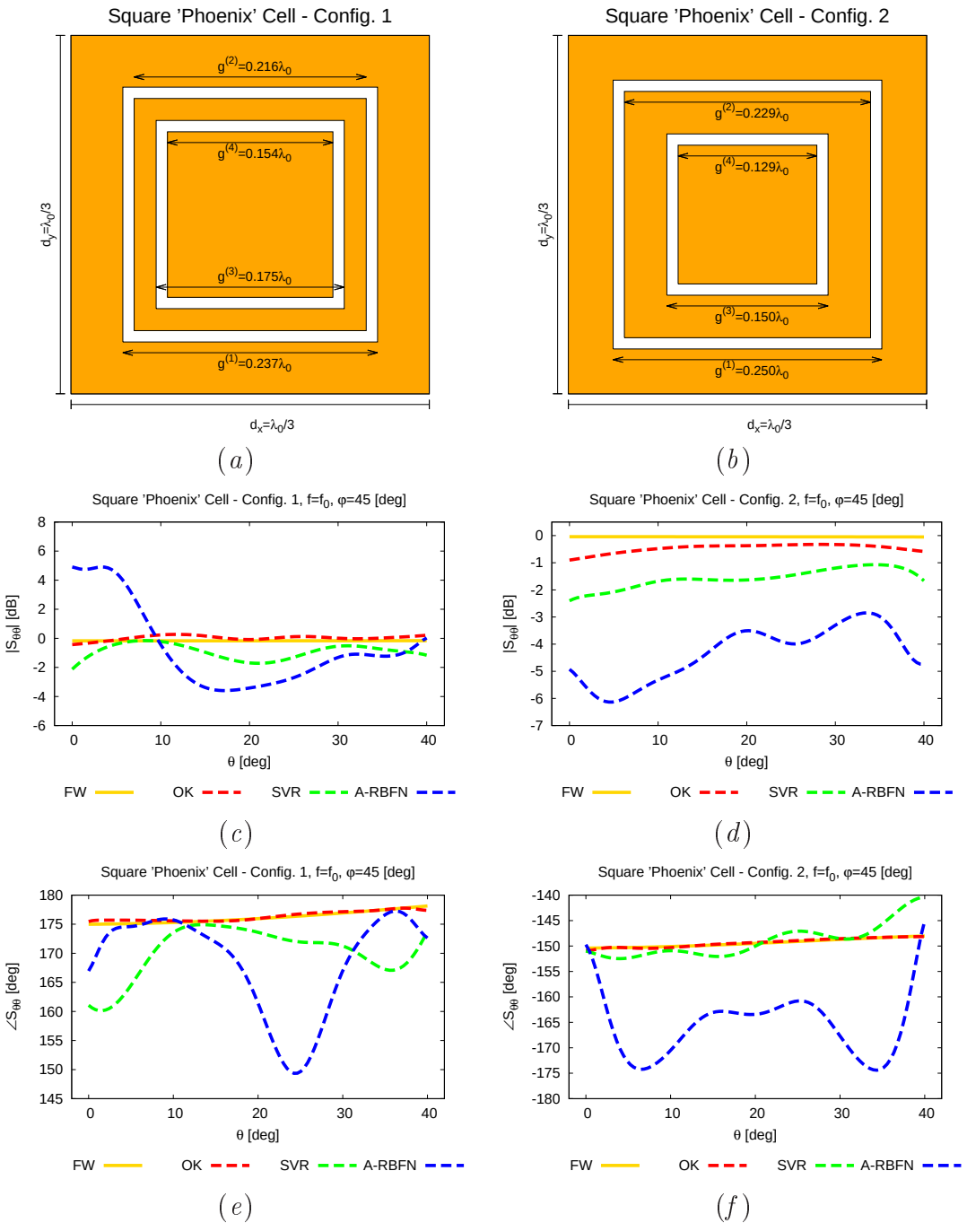


Figure 4.4: *Numerical Assessment* (*Square Phoenix unit cell*, $d_x = d_y = \frac{\lambda_0}{3}$, $f = f_0$, $\varphi = 45$ [deg], $U = 2.0 \times 10^4$) - Unit cell geometry $(a)(b)$ and behaviour of $(c)(d)$ the magnitude and $(e)(f)$ the phase of $S_{\theta\theta}(\mathbf{z})$ versus θ for $(a)(c)(e)$ "*Config.* 1" and $(b)(d)(f)$ "*Config.* 2".

Table 4.2: *Numerical Assessment* (*Square/Rectangular Phoenix unit cell*, $d_x = d_y = \frac{\lambda_0}{3}$, $B = 4$) - Geometrical descriptors of sample unit cell layouts.

Unit Cell	$g^{(1)}$ $[\lambda_0]$	$g^{(2)}$ $[\lambda_0]$	$g^{(3)}$ $[\lambda_0]$	$g^{(4)}$ $[\lambda_0]$
Square Phoenix - Config. 1	0.237	0.216	0.175	0.154
Square Phoenix - Config. 2	0.250	0.229	0.150	0.129
Rect. Phoenix - Config. 3	0.266	0.237	0.152	0.185
Rect. Phoenix - Config. 4	0.156	0.262	0.135	0.158

from the actual values despite the complexity of the geometry at hand [Fig. 4.2(a)] [2][12][13][14] and unlike the other state-of-the-art *LBE* methods [e.g., Fig. 4.4(e)].

To further assess and generalize these positive observations, Figure 4.5 reports the scatter plots of the real and imaginary parts of $S_{\theta\theta}(\mathbf{z}_m)$, $m = 1, ..., M$ (Fig. 4.5). As it can be inferred, the *OK* plots are closer to the *ideal* bisector behavior than the *SVR* and the *A-RBFN* ones [Fig. 4.5(a) vs. Fig. 4.5(b) and Fig. 4.5(c)]. The same conclusions hold true for the cross-polar component $S_{\theta\phi}(\mathbf{z}_m)$, $m = 1, ..., M$ (Fig. 4.6), as well. Indeed, notwithstanding the weaker magnitude [Fig. 4.6(a) vs. Fig. 4.5(a)], which is physically motivated by the square symmetric nature of the considered element [Fig. 4.2(a)], the proposed method is able to perform a quite reliable prediction [e.g., Fig. 4.6(a)], while the scatter clouds of the *SVR* [e.g., Fig. 4.6(b)] and the *A-RBFN* [e.g., Fig. 4.6(c)] significantly deviate from the ideal curve.

As for the computational issues and overall efficiency in dealing with the "*Square Phoenix*" unit cells, the plots of T_{train} vs. U in Fig. 4.7 show that the training phase for the *OK* -based approach is slightly more expensive than those of the *SVR* and the *A-RBFN* ones [solid lines - Fig. 4.7(a)]. This was theoretically expected because of the need of determining the autocorrelation matrix $\Gamma(\boldsymbol{\eta})$, not required by the other state-of-the-art techniques, whose computational load grows quadratically with the size of the training set, U. On the other hand, the time spent for the testing phase, T_{test} [dashed lines - Fig. 4.7(a)], is quite similar for all the considered *LBE* methods. Anyway, both T_{train} and T_{test} are always negligible when compared to the time for building the U-entries training set $\mathcal{T}$, T_{set}^{FW} [Fig. 4.7(a)], even though an highly efficient *FW* solver has been used [56][57]. Thus, it turns out that the overall computational cost, T_{tot}, is dominated by the simulation time for the training set creation regardless of the *LBE* technique at hand ($T_{tot} \approx T_{set}^{FW}$). Consequently, the behaviour of the time saving ΔT^{OK} versus U is almost identical to ΔT^{SVR} and ΔT^{A-RBFN} [Fig. 4.7(b)] and it always complies with the condition $\Delta T > 99.3\%$ [Fig. 4.7(b)]. Such an outcome, jointly with the results on the prediction accuracy from the analysis

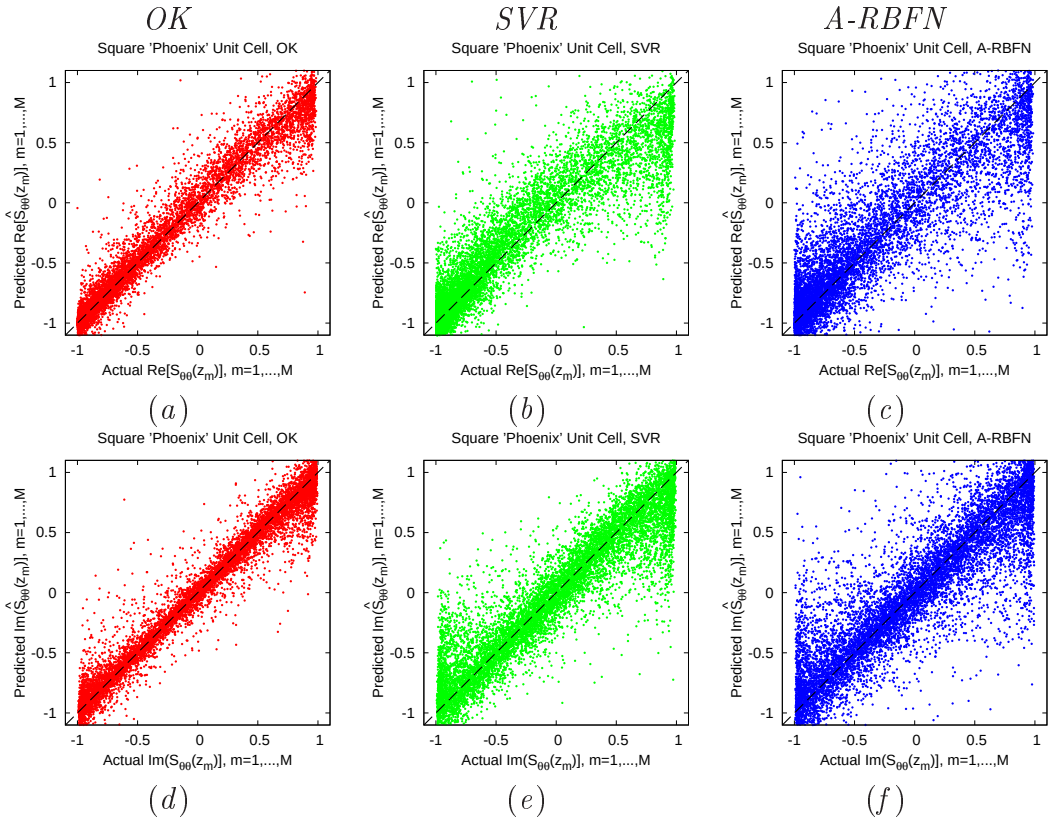


Figure 4.5: *Numerical Assessment* (*Square Phoenix unit cell*, $d_x = d_y = \frac{\lambda_0}{3}$, $U = 2.0 \times 10^4$) - Actual versus estimated values of $(a)(b)(c)$ $Re\{S_{\theta\theta}(\mathbf{z}_m)\}$, $m = 1, ..., M$, and $(d)(e)(f)$ $Im\{S_{\theta\theta}(\mathbf{z})\}$ when using $(a)(d)$ the *OK*, $(b)(e)$ the *SVR*, and $(c)(f)$ the *A-RBFN* prediction methods.

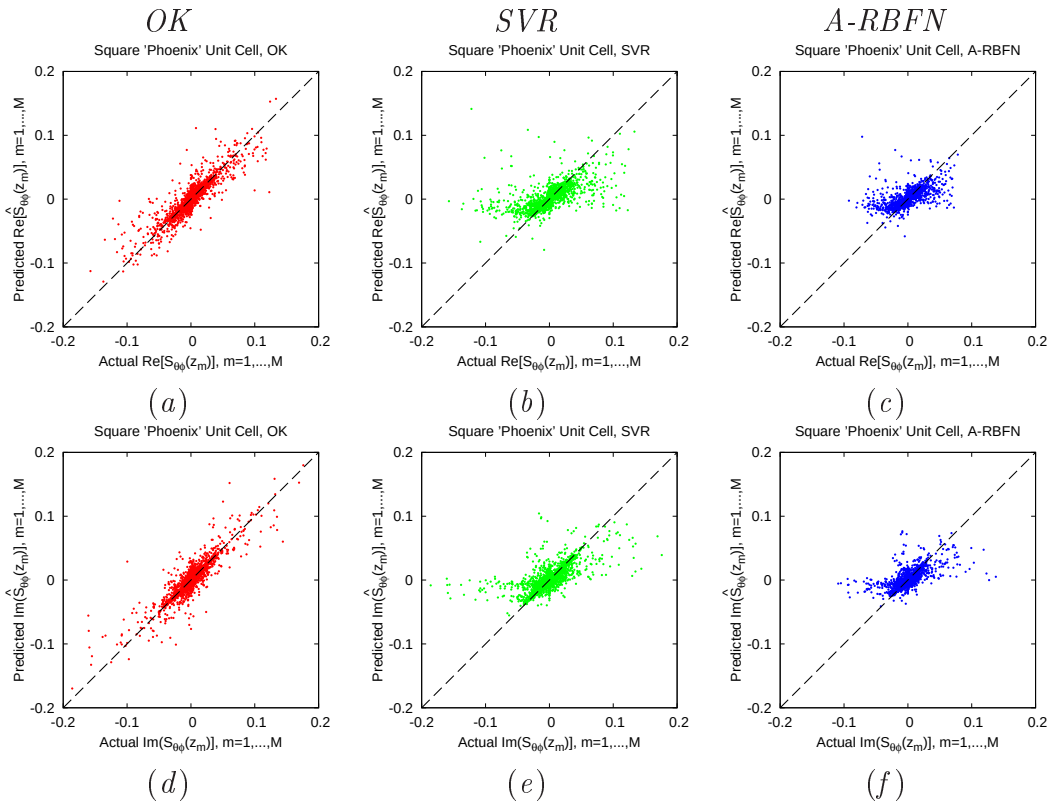


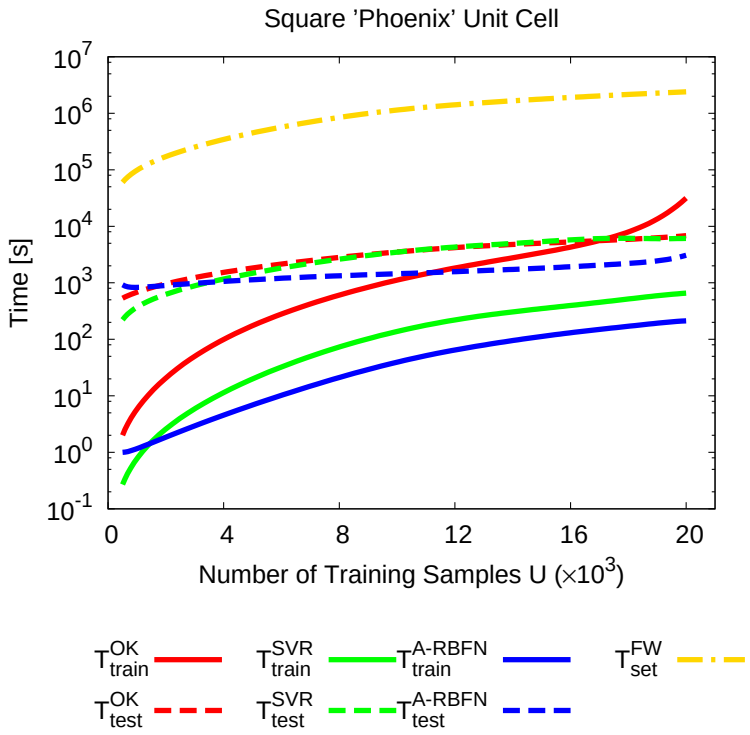
Figure 4.6: *Numerical Assessment* (*Square Phoenix unit cell*, $d_x = d_y = \frac{\lambda_0}{3}$, $U = 2.0 \times 10^4$) - Actual versus estimated values of $(a)(b)(c)$ $Re\{S_{\theta\varphi}(\mathbf{z}_m)\}$, $m = 1, ..., M$, and $(d)(e)(f)$ $Im\{S_{\theta\varphi}(\mathbf{z})\}$ when using $(a)(d)$ the *OK*, $(b)(e)$ the *SVR*, and $(c)(f)$ the *A-RBFN* prediction methods.

of the error figures Ξ_1 and Ξ_2 (Fig. 4.2), proves that the proposed *OK* strategy can be reliably and efficiently exploited for filling a very huge scattering matrix database ($M \approx 3.1 \times 10^6$) with a considerable time saving with respect to an heavy use of an albeit efficient *FW* technique ($T_{tot}^{OK} \approx 2.43 \times 10^6$ [s] ≈ 28 days vs. $T_{tot}^{FW} \approx 3.69 \times 10^8$ [s] ≈ 11.7 years - Fig. 4.7), while guaranteeing a faithful estimation of the scattering matrix $\mathcal{S}(\mathbf{z})$ (e.g., Fig. 4.4).
But what's about the prediction of the electromagnetic response from reflectarray unit cells with stronger cross-polar scattering matrix entries? To give some feedbacks about this question, the $B = 4$ "*Rectangular Phoenix*" unit cell [12][13] in Fig. 4.2(b) has been considered as the next benchmark by discretizing its descriptors/*DoF*s analogously to the unit cell in Fig. 4.2(a), but considering

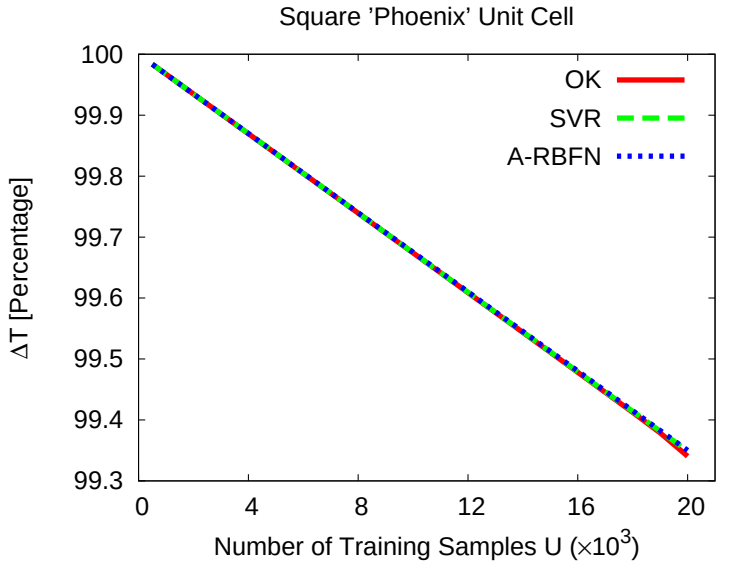
$$\Im\{l_b\} = \begin{cases} l_b & b = 1, 2 \\ l_b \times \mathcal{H}(l_{b-2} - l_b) & b = 3, 4 \end{cases} . \tag{4.14}$$

This choice corresponds to $L \approx 2.57 \times 10^5$ different geometrical configurations[2] yielding to $M \approx 2.7 \times 10^7$ entries of $\mathcal{D}$, which correspond to $T_{tot}^{FW} \approx 5.56 \times 10^8$ [s] (i.e., ≈ 17.6 years) since $T_{sim}^{FW} \approx 2.00 \times 10^1$ [s]. By comparing the plots of the matrix norm errors of the *OK*, the *SVR*, and the *A-RBFN* methods [Fig. 4.8(a)], it turns out that the former once again outperforms the others in terms of fidelity [e.g., $\left.\frac{\Xi_1^{SVR}}{\Xi_1^{OK}}\right\rfloor_{U=2.0\times10^4} \approx 467\%$ and $\left.\frac{\Xi_1^{A-RBFN}}{\Xi_1^{OK}}\right\rfloor_{U=2.0\times10^4} \approx 402\%$ - Fig. 4.8(a)] with a prediction accuracy enhancement with the size U [e.g., $\frac{\Xi_1^{OK}\rfloor_{U=2.0\times10^4}}{\Xi_1^{OK}\rfloor_{U=5.0\times10^2}} \approx 15.8\%$ - Fig. 4.8(a)]. Moreover, the error behavior is very close to that for the simpler unit cell in Fig. 4.2(a) [e.g., $\Xi_1^{OK}\rfloor_{U=2.0\times10^4}^{square} \approx 3.8 \times 10^{-2}$ vs. $\Xi_1^{OK}\rfloor_{U=2.0\times10^4}^{rect} \approx 3.7 \times 10^{-2}$ - Fig. 4.3(a) vs. Fig. 4.8(a)] even though $U \lll M$ (i.e., $\frac{U}{M} \approx 7.4 \times 10^{-4}$).
For illustrative purposes, the plots of $|S_{\theta\theta}(\mathbf{z})|$ and $|S_{\theta\varphi}(\mathbf{z})|$ for two sample unit cell geometries [Tab. 4.II] not belonging to $\mathcal{T}$ are shown in Fig. 4.9 ["*Rectangular Phoenix Cell - Config. 3* " - Fig. 4.9(a) and Tab. 4.II] and Fig. 4.10 ["*Rectangular Phoenix - Cell Config. 4*" - Fig. 4.10(a) and Tab. 4.II]. As it can be observed, the comparisons among the *OK*, the *SVR*, and the *A-RBFN* predictions show that (i) the behaviour and the values of $\left|S_{\theta\theta}^{OK}(\mathbf{z})\right|$ [Fig. 4.9(b) and Fig. 4.10(b)] and $\left|S_{\theta\varphi}^{OK}(\mathbf{z})\right|$ [Fig. 4.9(c) and Fig. 4.10(c)] match very well the corresponding *FW* results with a maximum deviation smaller than 0.6 dB [Fig. 4.10(b)], (ii) the *SVR* and the *A-RBFN* predictions often turn out to be inaccurate [e.g., Fig. 4.9(b)] even providing qualitatively different trends with respect to the actual electromagnetic response. For instance, $\left|S_{\theta\theta}^{SVR}(\mathbf{z})\right|^{Config-3}$ increases with θ until $\theta = 30$ [deg], while $\left|S_{\theta\theta}^{FW}(\mathbf{z})\right|^{Config-3}$ decreases in the same range [Fig. 4.9(b)].

[2]Thanks to the lower degree of symmetry of the layout in Fig. 4.2(b) than that in Fig. 4.2(a) (i.e., one-axis symmetry vs. two-axes symmetry), a significantly greater number of geometrical variations is feasible.

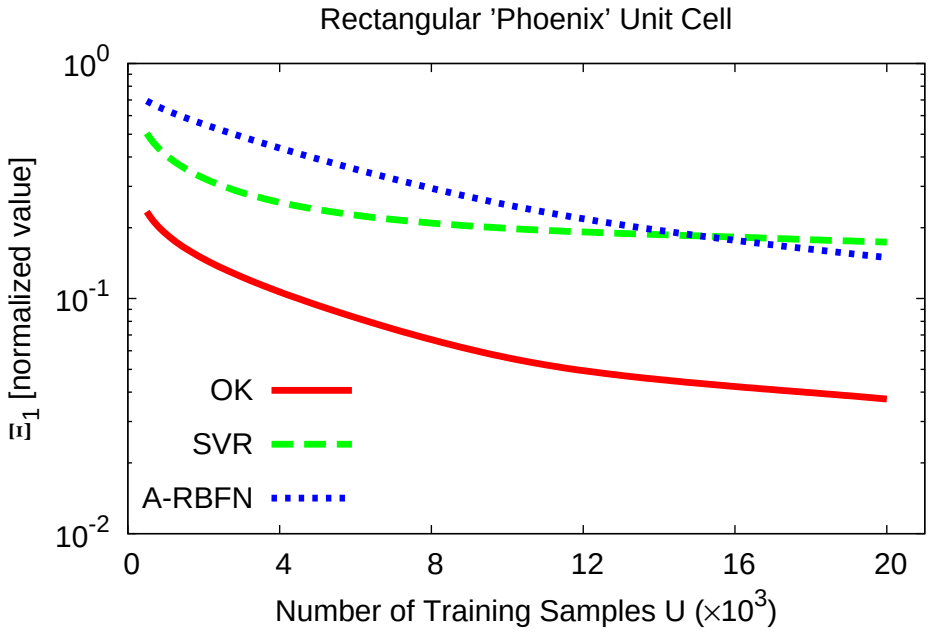


(a)

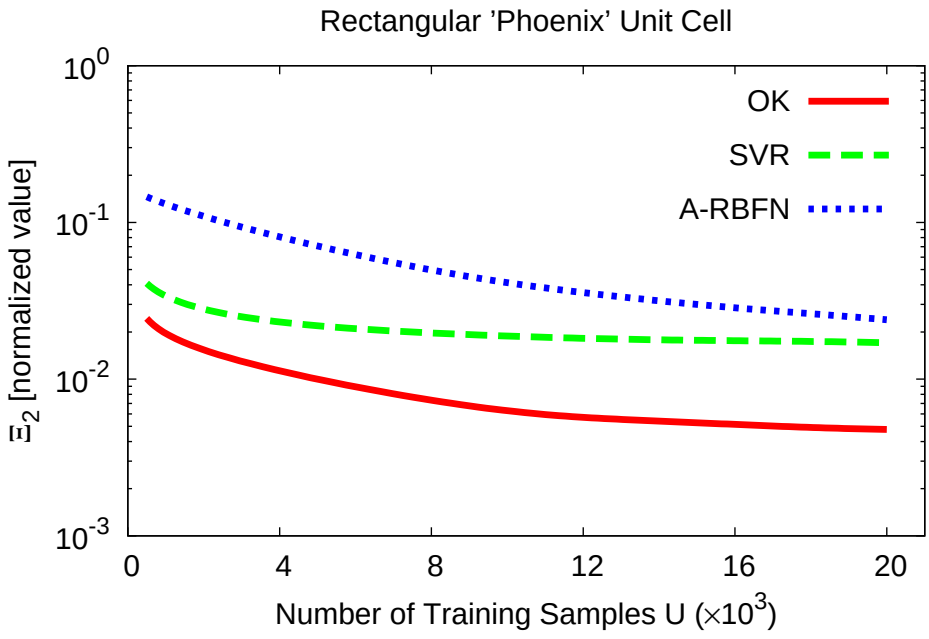


(b)

Figure 4.7: *Numerical Assessment* (*Square Phoenix unit cell*, $d_x = d_y = \frac{\lambda_0}{3}$, $U \in [5.0 \times 10^2, 2.0 \times 10^4]$) - Behaviour of (a) T_{train}andT_{test}, and (b) ΔT when using the *OK*, the *SVR*, and the *A-RBFN* predictors.



(a)



(b)

Figure 4.8: *Numerical Assessment* (*Rectangular Phoenix unit cell*, $d_x = d_y = \frac{\lambda_0}{3}$, $U \in [5.0 \times 10^2, 2.0 \times 10^4]$) - Behavior of (a) Ξ_1 and (b) Ξ_2 versus the size of the training set U.

Moreover, $\left|S_{\theta\varphi}^{A-RBFN}(\mathbf{z})\right|^{Config-4}$ exhibits an oscillating behaviour which is not present in $\left|S_{\theta\varphi}^{FW}(\mathbf{z})\right|^{Config-4}$ [Fig. 4.10(c)]. Similar considerations hold true for the plots of $\angle S_{\theta\theta}(\mathbf{z})$ and $\angle S_{\theta\varphi}(\mathbf{z})$ in correspondence with the same sample configurations [i.e., "*Config. 3*" - Figs. 4.9(d)-4.9(e); "*Config. 4*" - Figs. 4.10(d)-4.10(e)]: $\left|\angle S_{\theta\theta}^{FW}(\mathbf{z}) - \angle S_{\theta\theta}^{OK}(\mathbf{z})\right| < 0.8$ [deg], $\left|\angle S_{\theta\theta}^{FW}(\mathbf{z}) - \angle S_{\theta\theta}^{SVR}(\mathbf{z})\right| < 14.5$ [deg], $\left|\angle S_{\theta\theta}^{FW}(\mathbf{z}) - \angle S_{\theta\theta}^{A-RBFN}(\mathbf{z})\right| < 4.0$ [deg] for "*Config. 3*" [Figs. 4.9(d)-4.9(e)] as it can be also inferred from the behaviour of Ξ_2 in Fig. 4.8(b).

Such outcomes on the reliability of the *OK*-based approach in handling reflectarray elements featuring non-negligible cross-polar entries are further assessed by the scatter plots of the real and imaginary parts of $S_{\theta\theta}(\mathbf{z}_m)$, $m = 1, ..., M$ (Fig. 4.11) and of $S_{\theta\varphi}(\mathbf{z}_m)$, $m = 1, ..., M$ (Fig. 4.12).

As for the efficiency/time saving when addressing such a benchmark, Figure 4.13(a) confirms that (i) as expected, the *OK* training phase is more expensive than the *SVR* and the *A-RBFN* ones [solid lines - Fig. 4.13(a)], (ii) the testing phases of all considered *LBE* methods have analogous durations [dashed lines - Fig. 4.13(a)], but it points out that (iii) although T_{train} and T_{test} are smaller than T_{set}^{FW} regardless of the adopted method - as in the previous benchmark example [Fig. 4.7(a)] - their values are no more negligible [e.g., $T_{train}^{OK}\big\rfloor_{U=2.0\times10^4} \approx 3.16 \times 10^4$ vs. $T_{test}^{OK}\big\rfloor_{U=2.0\times10^4} \approx 6.15 \times 10^4$ [s] vs. $T_{set}^{FW}\big\rfloor_{U=2.0\times10^4} \approx 4.00 \times 10^5$ [s] - Fig. 4.13(a)]. This is due to the fact that T_{sim}^{FW} is significantly smaller than in the previous test case (i.e., $T_{sim}^{FW}\big\rfloor_{rect.} \approx 20$ [s] vs. $T_{sim}^{FW}\big\rfloor_{square} \approx 120$ [s]) as a consequence of the higher efficiency of the *FW* technique in handling the reference electrical layout [i.e., 1 slot vs. 2 concentric slots - Fig. 4.2(b) vs. Fig. 4.2(a)]. Therefore, ΔT^{OK} is here slightly lower than ΔT^{SVR} and ΔT^{A-RBFN} [Fig. 4.13(b)], even though it must be noticed that $\Delta T^{OK} > 99.9\%$ even when $U = 2.0 \times 10^4$ [Fig. 4.13(b)], which turns out in $T_{tot}^{OK} \approx 4.93 \times 10^5$ [s] ≈ 5.7 days vs. $T_{tot}^{FW} \approx 5.56 \times 10^8$ [s] ≈ 17.6 years [Fig. 4.13], while guaranteeing excellent estimation accuracies (Fig. 4.8).

The last numerical experiment is devoted to the assessment of the performance of the proposed *LBE* method when handling geometries with a wider lattice periodicity [$\mathbf{d}_1 = 0.7\lambda_0\widehat{\mathbf{x}}$, $\mathbf{d}_2 = 0.7\lambda_0\widehat{\mathbf{y}}$ - Figs. 4.14(a)-4.14(b) vs. $\mathbf{d}_1 = \frac{\lambda_0}{3}\widehat{\mathbf{x}}$, $\mathbf{d}_2 = \frac{\lambda_0}{3}\widehat{\mathbf{y}}$ - Fig. 4.2, Figs. 4.4(a)-4.4(b), Fig. 4.9(a), and Fig. 4.10(a)] [62]. Moreover, two different $B = 3$ unit cells featuring either a single "*Square Ring Slot*" [*SRS* - Fig. 4.14(a)] or a "*Cross Slot*" [*CS* - Fig. 14.4(b)] have been considered [62][20]. By setting $V = 18$, $H = 10$, $W = 6$, $L_b = 16$, $g_{\min}^{(b)} = 0$, $g_{\max}^{(b)} = 0.2\lambda_0$ ($b = 1, ..., B$),

$$\Im^{SRS}\{l_b\} = \begin{cases} l_b & b = 1 \\ l_b \times \mathcal{H}(l_{b-1} - l_b) & b = 2, 3 \end{cases} \tag{4.15}$$

$$\Im^{CS}\{l_b\} = \begin{cases} l_b & b = 1 \\ l_b \times \mathcal{H}(l_1 - l_b) & b = 2, 3 \end{cases} \tag{4.16}$$

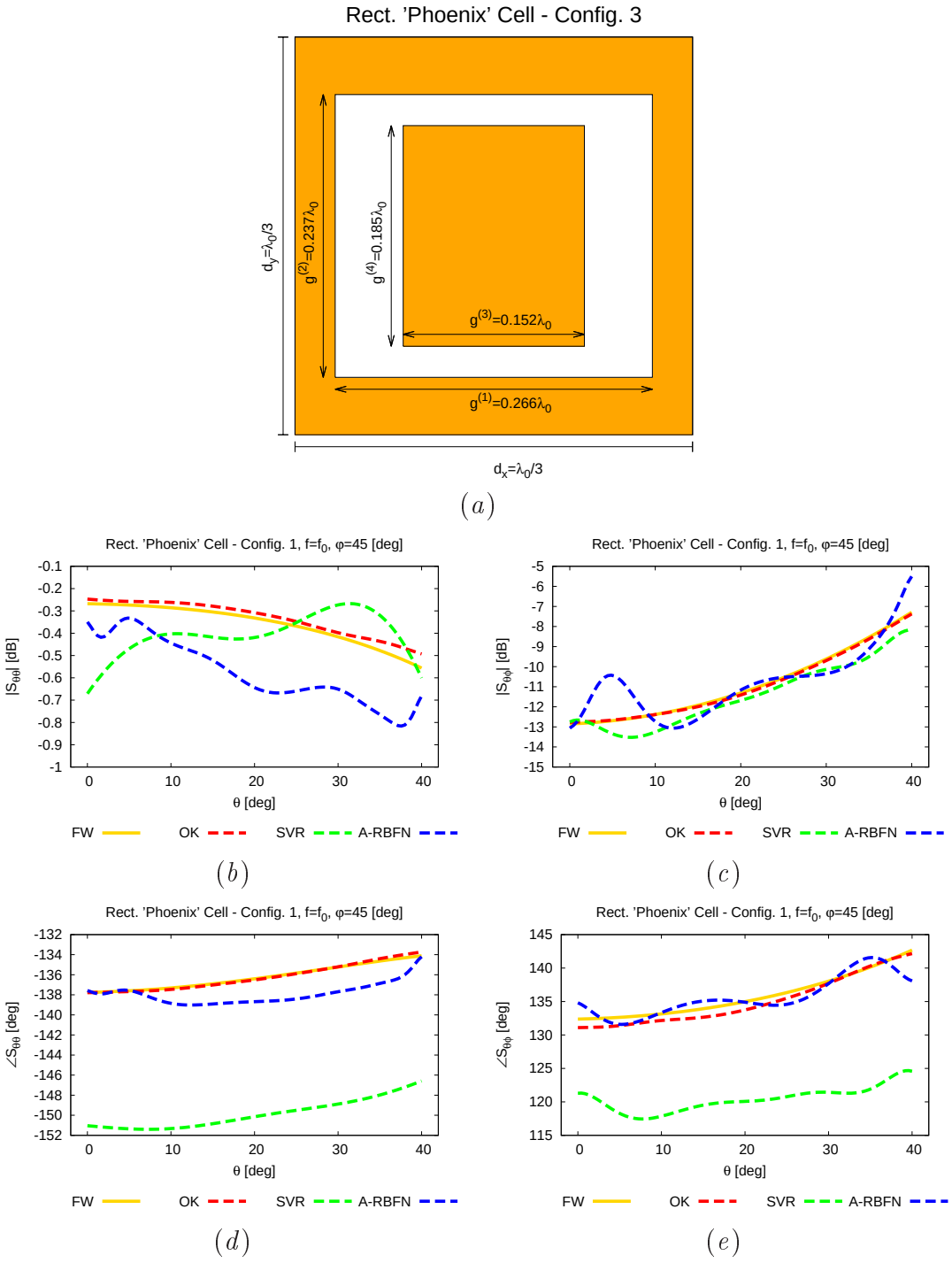


Figure 4.9: *Numerical Assessment* (*Rectangular Phoenix unit cell* - "*Config.* 3", $d_x = d_y = \frac{\lambda_0}{3}$, $f = f_0$, $\varphi = 45$ [deg], $U = 2.0 \times 10^4$) - Unit cell geometry (a) and behaviour of $(b)(c)$ the magnitude and $(d)(e)$ the phase of $(b)(d)$ $S_{\theta\theta}(\mathbf{z})$ and $(c)(e)$ $S_{\theta\varphi}(\mathbf{z})$ versus θ.

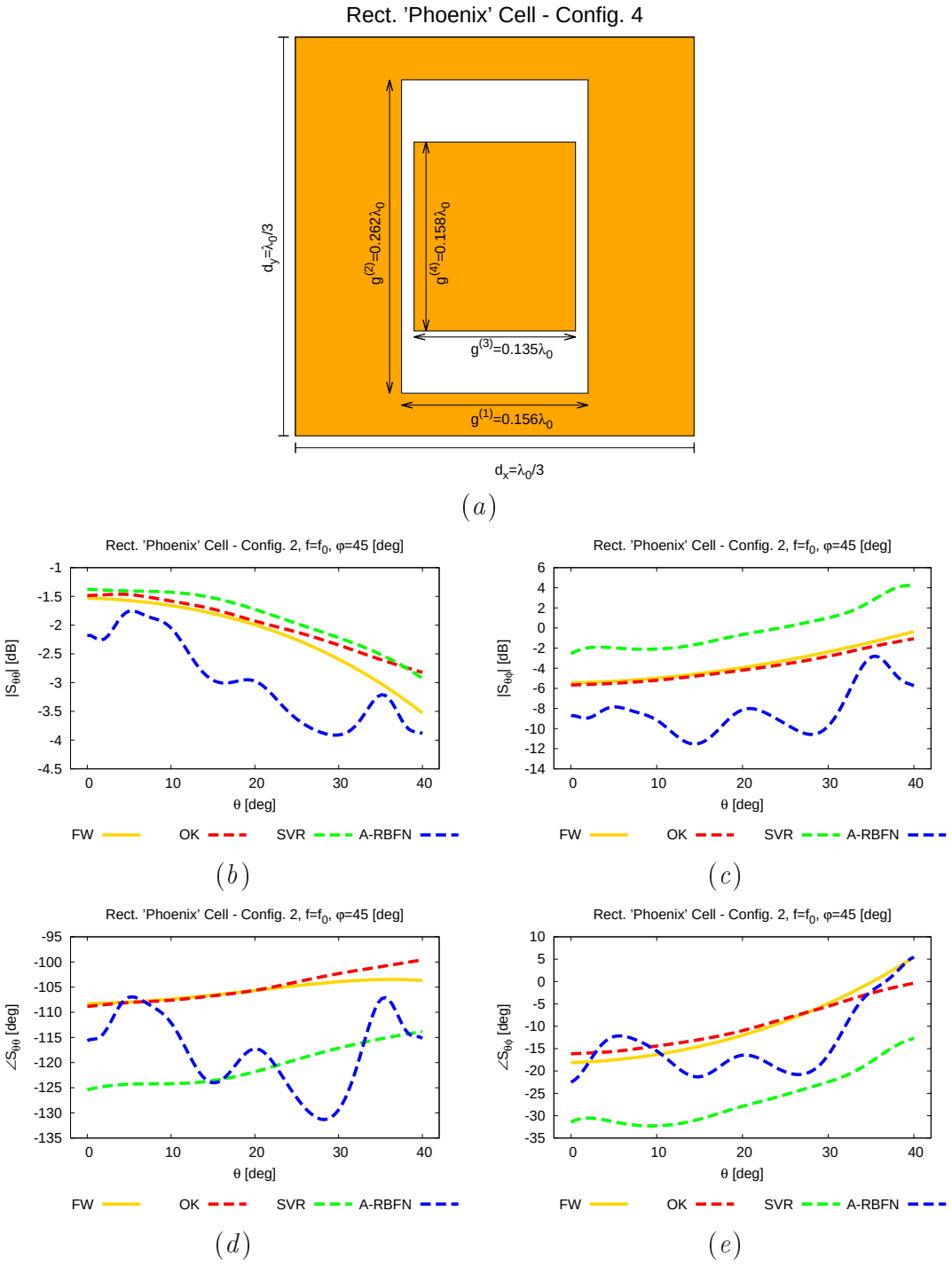


Figure 4.10: *Numerical Assessment* (*Rectangular Phoenix unit cell* - "*Config.* 3", $d_x = d_y = \frac{\lambda_0}{3}$, $f = f_0$, $\varphi = 45$ [deg], $U = 2.0 \times 10^4$) - Unit cell geometry (a) and behaviour of $(b)(c)$ the magnitude and $(d)(e)$ the phase of $(b)(d)$ $S_{\theta\theta}(\mathbf{z})$ and $(c)(e)$ $S_{\theta\varphi}(\mathbf{z})$ versus θ.

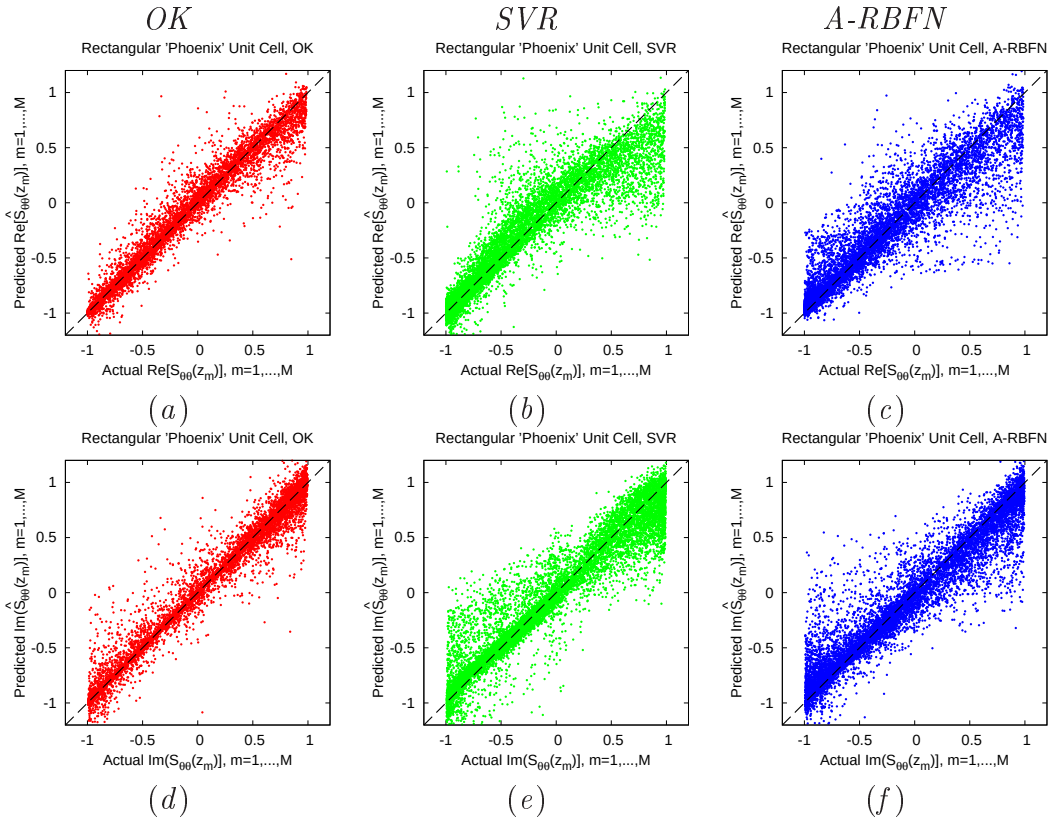


Figure 4.11: *Numerical Assessment* (*Rectangular Phoenix unit cell*, $d_x = d_y = \frac{\lambda_0}{3}$, $U = 2.0 \times 10^4$) - Actual versus estimated values of $(a)(b)(c)$ $Re\{S_{\theta\theta}(\mathbf{z}_m)\}$, $m = 1, ..., M$, and $(d)(e)(f)$ $Im\{S_{\theta\theta}(\mathbf{z})\}$ when using $(a)(d)$ the *OK*, $(b)(e)$ the *SVR*, and $(c)(f)$ the *A-RBFN* predictors.

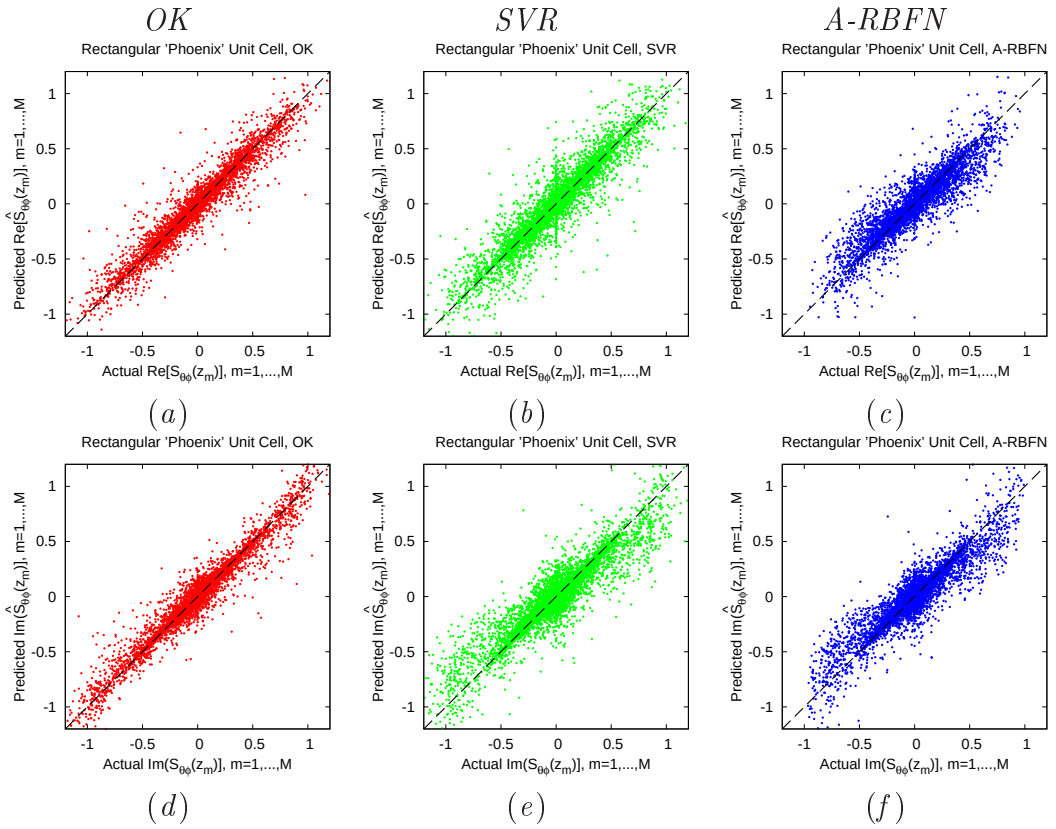
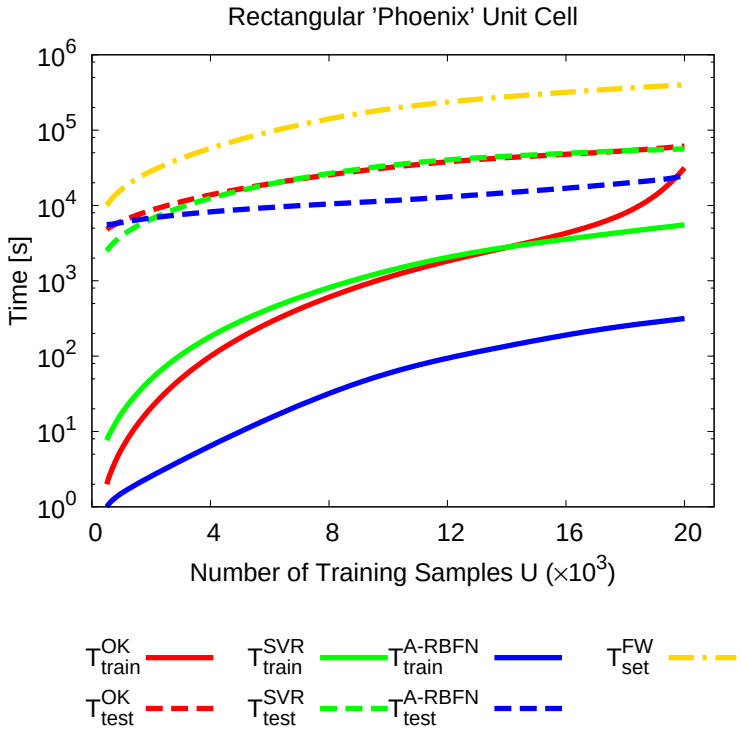
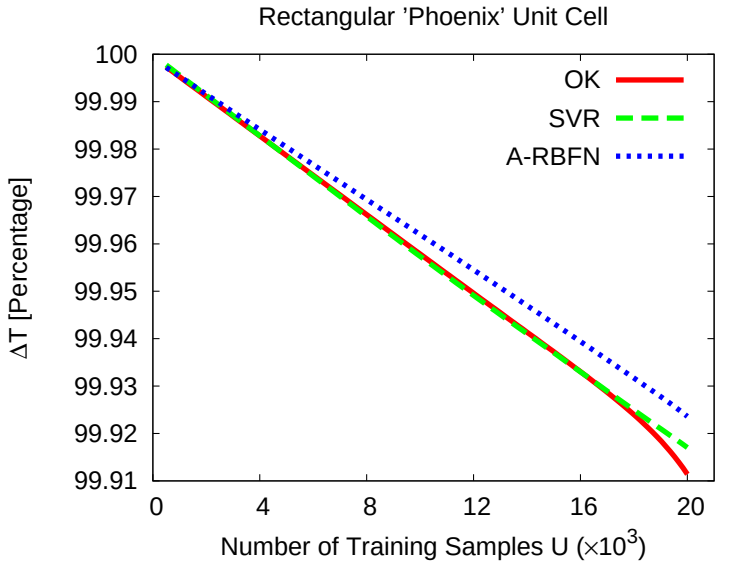


Figure 4.12: *Numerical Assessment* (*Rectangular Phoenix unit cell*, $d_x = d_y = \frac{\lambda_0}{3}$, $U = 2.0 \times 10^4$) - Actual versus estimated values of $(a)(b)(c)$ $Re\{S_{\theta\varphi}(\mathbf{z}_m)\}$, $m = 1, ..., M$, and $(d)(e)(f)$ $Im\{S_{\theta\varphi}(\mathbf{z})\}$ when using $(a)(d)$ the *OK*, $(b)(e)$ the *SVR*, and $(c)(f)$ the *A-RBFN* predictors.



(a)



(b)

Figure 4.13: *Numerical Assessment* (*Rectangular Phoenix unit cell*, $d_x = d_y = \frac{\lambda_0}{3}$, $U \in [5.0 \times 10^2, 2.0 \times 10^4]$) - Behaviour of (a) $T_{train}$and T_{test}, and (b) ΔT when using the *OK*, the *SVR*, and the *A-RBFN* predictors.

resulting in $L = 5.25 \times 10^2$ different admissible geometrical configurations[3], it turns out that $\mathcal{D}$ comprises $M \approx 5.7 \times 10^5$ entries, which correspond to a computational load of $T_{tot}^{FW} \approx 1.4 \times 10^7$ [s] (i.e., ≈ 164 days) when using a FW solver ($T_{sim}^{FW} \approx 2.50 \times 10^1$ [s]) to fill the whole LUT.
Although the lattice periodicity is different and qualitatively less smooth phase variations arise [62], analogous feedbacks on the higher accuracy of the OK strategy can be drawn in terms of both magnitude [e.g., $\left.\frac{\Xi_1^{SVR}}{\Xi_1^{OK}}\right|_{U=2.0\times10^4} \approx 623\%$ and $\left.\frac{\Xi_1^{A-RBFN}}{\Xi_1^{OK}}\right|_{U=2.0\times10^4} \approx 1535\%$ - Fig. 4.14(c); $\left.\frac{\Xi_1^{SVR}}{\Xi_1^{OK}}\right|_{U=2.0\times10^4} \approx 235\%$ and $\left.\frac{\Xi_1^{A-RBFN}}{\Xi_1^{OK}}\right|_{U=2.0\times10^4} \approx 659\%$ - Fig. 4.14(d)] and phase prediction [e.g., $\left.\frac{\Xi_2^{SVR}}{\Xi_2^{OK}}\right|_{U=2.0\times10^4} \approx 328\%$ and $\left.\frac{\Xi_2^{A-RBFN}}{\Xi_2^{OK}}\right|_{U=2.0\times10^4} \approx 899\%$ - Fig. 4.14(c); $\left.\frac{\Xi_2^{SVR}}{\Xi_2^{OK}}\right|_{U=2.0\times10^4} \approx 253\%$ and $\left.\frac{\Xi_2^{A-RBFN}}{\Xi_2^{OK}}\right|_{U=2.0\times10^4} \approx 551\%$ - Fig. 4.14(d)]. Concerning the computational costs, once again it is verified that the OK approach is able to yield the best trade-off between time saving [$\Delta T^{OK} > 96\%$ - Figs. 4.15(c)-4.15(d)] and accuracy, thus its candidature as a suitable and competitive tool for efficiently [$T_{tot}^{OK} \approx 5.17 \times 10^5$ [s] ≈ 5.9 days vs. $T_{tot}^{FW} \approx 1.4 \times 10^7$ [s] ≈ 164 days] and faithfully generating large reflectarray scattering matrix databases ($M \approx 5.7 \times 10^5$).

[3]The number of geometrical variations L is significantly smaller than in the previous examples since $B = 3$ (while $B = 4$ for the geometries in Fig. 4.2).

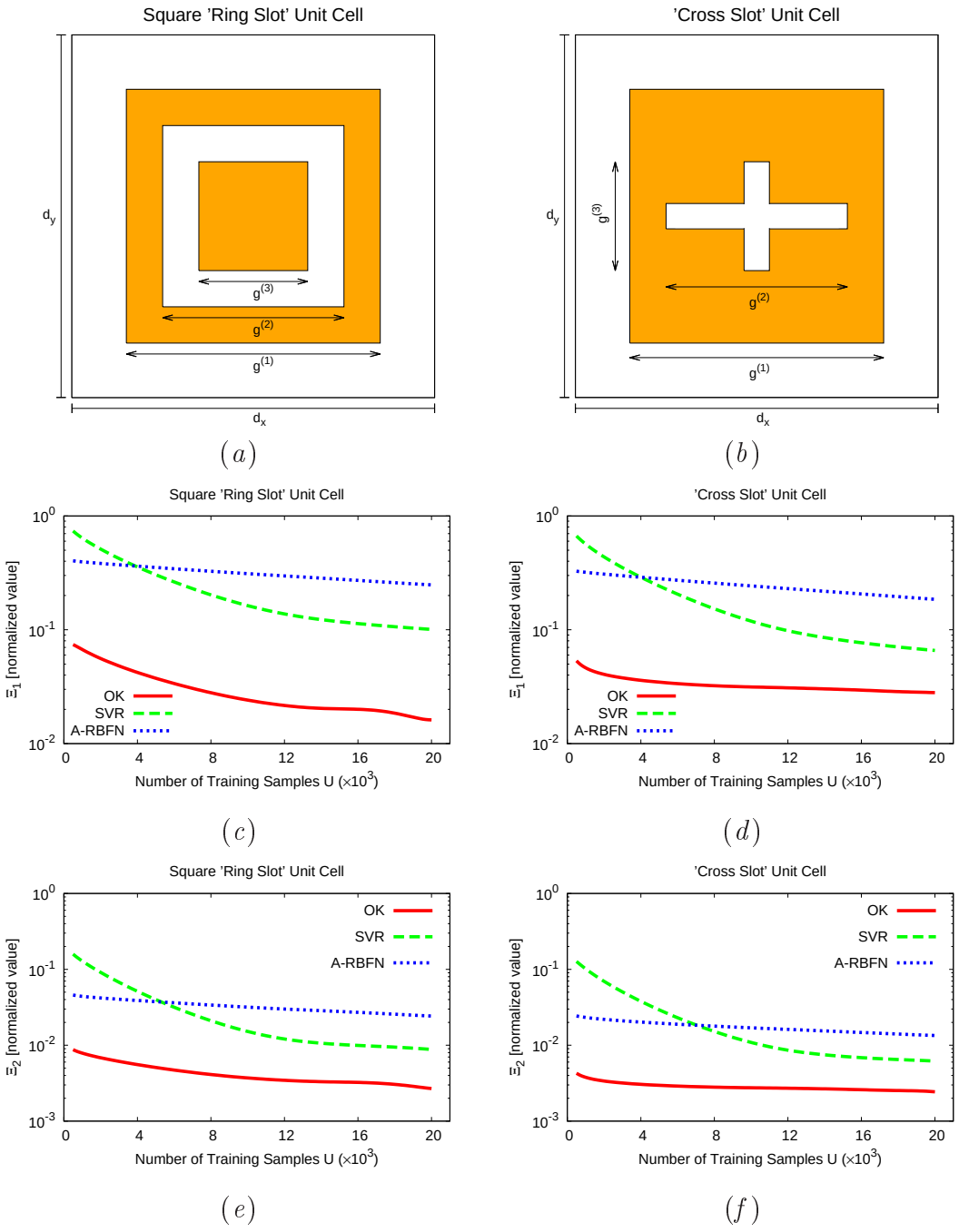


Figure 4.14: *Numerical Assessment* ($d_x = d_y = 0.7\lambda_0$, $B = 3$, $U \in [5.0 \times 10^2, 2.0 \times 10^4]$) - Geometries $(a)(b)$ and behavior of $(c)(d)$ Ξ_1 and $(e)(f)$ Ξ_2 versus the size of the training set U for $(a)(c)(e)$ the "*Square Ring Slot*" unit cell and $(b)(d)(f)$ the "*Cross Slot*" unit cell.

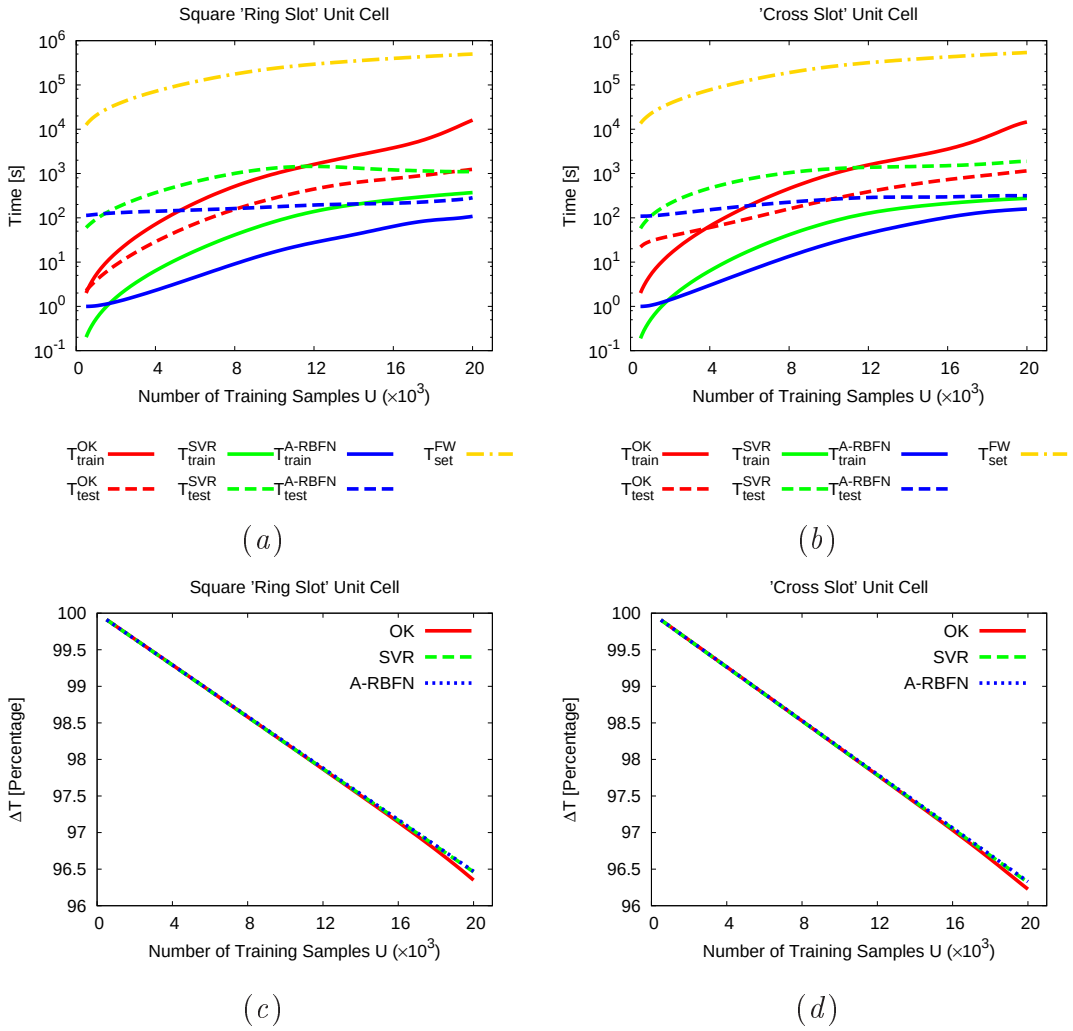


Figure 4.15: *Numerical Assessment* ($d_x = d_y = 0.7\lambda_0$, $B = 3$, $U \in [5.0 \times 10^2, 2.0 \times 10^4]$) - Behaviour of $(a)(b)$ $T_{train}$and T_{test}, and $(c)(d)$ ΔT when using the *OK*, the *SVR*, and the *A-RBFN* prediction methods for $(a)(c)$ the "*Square Ring Slot*" unit cell and $(b)(d)$ the "*Cross Slot*" unit cell.

Chapter 5

Conclusions

An innovative *LBE* method based on an *Ordinary Kriging* strategy has been proposed to efficiently and accurately model the scattering response of complex-shaped reflectarray unit cells. Towards this end, the evaluation of the scattering coefficients of passive elements with an arbitrary number of (geometrical and electrical) *DoF*s has been formulated as a vectorial regression problem, which has been then solved through a customized *OK* technique. Selected and representative results from numerical experiments dealing with different unit cell geometries (including *cross-slot*, *ring-slot*, and *square/rectangular Phoenix* shapes) have been reported to assess the accuracy, the numerical efficiency as well as the achievable time-saving, and the flexibility of the proposed approach also in comparison with other competitive state-of-the-art machine learning methods based on *SVR* and *A-RBFN* algorithms.
From the numerical analysis, the following main outcomes can be drawn:

- thanks to the *OK* formulation, the optimal values of the control hyper-parameters are reliably self-configured during the training phase to provide a faithful prediction of the magnitude [e.g., Fig. 4.4(c)] and the phase [e.g., Fig. 4.4(e)] of the scattering coefficients of complex reflectarray unit cells;
- the prediction accuracy guaranteed by the proposed methodology turns out to be higher than that from *SVR* and *A-RBFN* methods in all considered benchmark configurations [e.g., Fig. 4.3 and Fig. 4.8];
- although the *OK* training phase is slightly more time-expensive than that for *SVR* and *A-RBFN* [e.g., solid lines - Fig. 4.7(a)], the arising time saving ($\Delta T^{OK} > 96\%$ - Fig. 4.15) is always very similar to that yielded with the *SVR* and the *A-RBFN* strategies [e.g., Fig. 4.7(b)];
- thanks to the excellent trade-off between accuracy and computational efficiency, the proposed prediction method can be profitably used to fill very huge scattering matrix databases and it represents a very competitive alternative to the heavy use of efficient *FW* solvers (e.g., $T_{tot}^{OK} \approx 5.7$ days vs. $T_{tot}^{FW} \approx 17.6$ years - Fig. 4.13).

In addition to these key-features, the main methodological advances of this research work comprise (*i*) the introduction of a flexible strategy to efficiently model the scattering response of arbitrarily complex reflectarray unit cells, thus potentially enabling their use in next-generation and more demanding reflectarray designs, (*ii*) the development and the customization to the vectorial case of an advanced *OK* technique for the prediction of complex-valued scattering matrices of periodic *EM* structures, and (*iii*) the derivation of operative guidelines on the achievable time saving and the arising prediction accuracy vs. the training set size for the exploitation of such an *OK* meta-modeling in reflectarray response prediction.
Future works will be aimed at combining the proposed *OK* algorithm with advanced approaches for the selection of the training samples and/or the reduction of the feature space [58]. Moreover, the integration of an *OK*-based meta-model in the *SbD* framework for the automated synthesis of large reflectarrays is under development. Finally, thanks to its generality (*ii*), the extension of the same *OK* paradigm to other popular periodic *EM* structures (such as *frequency-selective surfaces* and *metasurfaces*) is on-going.

Bibliography

[1] S. Montori, F. Cacciamani, R. Vincenti Gatti, R. Sorrentino, G. Arista, C. Tienda, J. A. Encinar, and G. Toso , "A transportable reflectarray antenna for satellite Ku-band emergency communications," *IEEE Trans. Antennas Propag.*, vol. 63, no. 4, pp. 1393-1407, Apr. 2015.

[2] R. Deng, F. Yang, S. Xu, and M. Li, "A low-cost metal-only reflectarray using modified slot-type phoenix element with 360° phase coverage," *IEEE Trans. Antennas Propag.*, vol. 64, no. 4, pp. 1556-1560, Apr. 2016.

[3] A. Tamminen, S. Makela, J. Ala-Laurinaho, J. Hakli, P. Koivisto, P. Rantakari, J. Saily, A. Luukanen, and A. V. Raisanen, "Reflectarray design for 120-GHz radar application: measurement results," *IEEE Trans. Antennas Propag.*, vol. 61, no. 10, pp. 5036-5047, Oct. 2013.

[4] J. G. D. Hester and M. M. Tentzeris, "Inkjet-printed flexible mm-Wave Van-Atta reflectarrays: A solution for ultralong-range dense multitag and multisensing chipless RFID implementations for IoT smart skins," *IEEE Trans. Microw. Theory Tech.*, vol. 64, no. 12, pp. 4763-4773, Dec. 2016.

[5] J. A. Encinar, C. Tienda, M. Barba, E. Carrasco and M. Arrebola, "Analysis, design and prototyping of reflectarray antennas for space applications," in *2013 Loughborough Antennas & Propagation Conference*, Loughborough, UK, 2013, pp. 1-5.

[6] Y. Rahmat-Samii and A. C. Densmore, "Technology trends and challenges of antennas for satellite communication systems," *IEEE Trans. Antennas Propag.*, vol. 63, no. 4, pp. 1191-1204, Apr. 2015.

[7] J. Huang and J. A. Encinar, *Reflectarray antennas*. Piscataway, NJ, USA: IEEE Press/Wiley, 2008.

[8] H. Yang, F. Yang, S. Xu, Y. Mao, M. Li, X. Cao, and J. Gao, "A 1-bit 10×10 reconfigurable reflectarray antenna: design, optimization, and experiment," *IEEE Trans. Antennas Propag.*, vol. 64, no. 6, pp. 2246-2254, Jun. 2016.

[9] H. Yang, F. Yang, S. Xu, M. Li, X. Cao, J. Gao, and Y. Zheng, "A study of phase quantization effects for reconfigurable reflectarray antennas," *IEEE Antennas Wireless Propag. Lett.*, vol. 16, pp. 302-305, May 2016.

[10] D. G. Berry, R. G. Malech, and W. A. Kennedy, "The reflectarray antenna," *IEEE Trans. Antennas Propag.*, vol. 11, no. 6, pp. 645-651, Nov. 1963.

[11] Y. Mao, S. Xu, F. Yang and A. Z. Elsherbeni, "A novel phase synthesis approach for wideband reflectarray design," *IEEE Trans. Antennas Propag.*, vol. 63, no. 9, pp. 4189-4193, Sep. 2015.

[12] L. Moustafa, R. Gillard, F. Peris, R. Loison, H. Legay, and E. Girard, "The Phoenix cell: a new reflectarray cell with large bandwidth and rebirth capabilities," *IEEE Antennas Wireless Propag. Lett.*, vol. 10, pp. 71-74, Jan. 2011.

[13] A. Vosoogh, K. Keyghobad, A. Khaleghi, and S. Mansouri, "A high efficiency Ku-band reflectarray antenna using single-layer multiresonance elements," *IEEE Antennas Wireless Propag. Lett.*, vol. 13, pp. 891-894, Apr. 2014.

[14] R. Deng, S. Xu, F. Yang and M. Li, "A single-layer high-efficiency wideband reflectarray using hybrid design approach," *IEEE Antennas Wireless Propag. Lett.*, vol. 16, pp. 884-887, Sep. 2016.

[15] R. Deng, S. Xu, F. Yang and M. Li, "Single-layer dual-band reflectarray antennas with wide frequency ratios and high aperture efficiencies using Phoenix elements," *IEEE Trans. Antennas Propag.*, vol. 65, no. 2, pp. 612-622, Feb. 2017.

[16] D. M. Pozar, S. D. Targonski, and H. D. Syrigos, "Design of millimiter wave microstrip reflectarrays," *IEEE Trans. Antennas Propag.*, vol. 45, no. 2, pp. 287-296, Feb. 1997.

[17] P. Rocca, L. Poli, N. Anselmi, M. Salucci, and A. Massa, "Predicting antenna pattern degradations in microstrip reflectarrays through interval arithmetic," *IET Microwave Antennas Propag.*, vol. 10., no. 8, pp. 817-826, Mar. 2016.

[18] R. Deng, F. Yang, S. Xu and M. Li, "An FSS-Backed 20/30-GHz dual-band circularly polarized reflectarray with suppressed mutual coupling and enhanced performance," *IEEE Trans. Antennas Propag.*, vol. 65, no. 2, pp. 926-931, Feb. 2017.

[19] P. Nayeri, A. Z. Elsherbeni and F. Yang, "Radiation analysis approaches for reflectarray antennas," *IEEE Antennas Propag. Mag.*, vol. 55, no. 1, pp. 127-134, Feb. 2013.

[20] L. Marnat, R. Loison, R. Gillard, D. Bresciani and H. Legay, "Accurate synthesis of a dual linearly polarized reflectarray," *Proc. 3rd European Conf. Antennas Propag.* (*EuCAP 2009*), Berlin, pp. 2523-2526, 2009.

[21] D. R. Jones, M. Schonlau, and W. J. Welch, "Efficient global optimization of expensive black box functions," *J. Global Optim.*, vol. 13, no.4, pp. 455-492, Jun. 1998.

[22] J. Sachs, W. J. Welch, T. J. Mitchell, and H. P. Wynn, "Design and analysus of computer experiments," *Statistical Science*, vol. 4, no. 4, pp. 409-435, 1989.

[23] A. Forrester, A. Sobester, and A. Keane, *Engineering Design via Surrogate Modelling: A Practical Guide*. New York, NY, USA: John Wiley & Sons, 2008.

[24] T. Makdissy, R. Gillard, E. Fourn, M. Ferrando Rocher, E. Girard, H. Legay, and L. Le Coq, "'Phoenix' reflectarray unit cell with reduced size and inductive loading," *IET Microwave Antennas Propag.*, vol. 10., no. 12, pp. 1363-1370, May 2016.

[25] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars, *Computational Geometry: Algorithms and Applications*. Santa Clara, CA, USA: Springer-Verlag, 2008.

[26] A. Massa, G. Oliveri, P. Rocca, and F. Viani, "System-by-design: a new paradigm for handling design complexity," *Proc. 8th European Conf. Antennas Propag. (EUCAP 2014)*, The Hague, The Netherlands, pp. 1180-1183, April 6-11, 2014.

[27] 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, no. 4, pp. 1948-1974, 2011.

[28] D. Maljovec, B. Wang, A. Kupresanin, G. Johannesson, V. Pascucci and P. Bremer, "Adaptive sampling with topological scores," *International Journal for Uncertainty Quantification*, vol. 3, no. 2, pp. 119-141, 2013.

[29] C. Q. Lam and W. I. Notz, "Sequential adaptive designs in computer experiments for response surface model fit", *Statistics and Applications*, vol. 6, no. 1-2, pp. 207-233, 2008.

[30] K. Shimoyama, S. Kawai and J. J. Alonso, "Dynamic adaptive sampling based on Kriging surrogate models for efficient uncertainty quantification," *54th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Material Conference*, April 8-11, Boston, Massachusetts, 2013.

[31] DACE website: *http://www2.imm.dtu.dk/~hbni/dace/.*

[32] J. C. Platt, "Fast training of support vector machines using sequential minimal optimization," *Advances in Kernel Methods-Support Vector Learning*, B.

Scholkopf, C. J. C. Burges, and A. J. Smola, Eds. Cambridge, MA, USA: MIT Press, 1999.

[33] V. N. Vapnik, *The Nature of Statistical Learning Theory*. New York, NY, USA: Springer-Verlag, 1995.

[34] A. J. Smola and B. Scholkopf, "A tutorial on support vector regression," *Statist. Comput.*, vol. 14, no. 3, pp. 199-222, Aug. 2004.

[35] H. W. Kuhn and A. W. Tucker, "Nonlinear Programming," in Proc. 2nd Berkley Symp. Math. Statist. Prob., 1951, pp. 481-492.st squares SVR inversion approach," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 11, pp. 6818-6832, Nov. 2016.

[36] Y. K. Sui, S. P. Li, and Y. Q. Guo, "An efficient global optimization algorithm based on augmented radial basis function," *Int. J. Simul. Multisci. Des. Optim.*, vol. 2, pp. 49-55, Feb. 2008.

[37] D. S. Broomhead, D. Lowe, "Radial basis function, multi-variable functional interpolation and adaptive networks," Technical Report, RSRE, 4148.

[38] M. D. Buhmann, Radial basis functions: theory and implementations, Cambridge University Press, 2003.

[39] N. B. Karayiannis, and M. Randolph-Gips, "On the construction and training of reformulated radial basis function neural networks," *IEEE Trans. on Neural Network*, vol. 14, no.4, Jul. 2003.

[40] Z. R. Yang, "A novel radial basis function neural network for discriminant analysis," *IEEE Trans. on Neural Network*, vol. 17, no.3, May. 2006.

[41] T. A. Metzler, "Design and analysis of a microstrip reflectarray," PhD Thesis, Univ. Massachusetts, Amherst, 1992.

[42] R. Mittra, C. H. Chan, and T. Cwik, "Techniques for analyzing frequency selective surfaces - a review," *Proc. IEEE*, vol. 76, no. 12,. pp.1593-1615, Dec. 1988.

[43] D. M. Pozar, "Radiation and scattering from a microstrip patch on a uniaxial substrate," *IEEE Trans. Antennas Propag.*, vol. 35, no. 6, pp. 613-621, Jun. 1987.

[44] D. M. Pozar, "Bandwidth of reflectarrays," *Electr. Lett.*, vol. 39, no. 21, pp. 1490-1491, Oct. 2003.

[45] J. A. Encinar, "Design of two-layer printed reflectarray using patches of variable size," *IEEE Trans. Antennas Propag.*, vol. 49, no. 10, pp. 1403-1410, Oct. 2001.

[46] J. A. Encinar and J. A. Zornoza, "Broadband design of three-layer printed reflectarrays," *IEEE Trans. Antennas Propag.*, vol. 51, no. 7, pp. 1662-1664, Jul. 2003.

[47] K. H. Sayidmarie and M. E. Bialkowski, "Fractal unit cells of increased phasing range and low slopes for single-layer microstrip reflectarrays," *IET Microw. Antennas Propag.*, vol. 5, no. 11, pp. 1371-1379, Aug. 2011.

[48] J. A. Encinar, M. Arrebola, L. F. de la Fuente, and G. Toso, "A transmit-receive reflectarray antenna for direct broadcast satellite applications," *IEEE Trans. Antennas Propag.*, vol. 59, no. 9, pp. 3255-3264, Sep. 2011.

[49] M. R. Chaharmir, J. Shaker, and H. Legay, "Dual-band Ka/X reflectarray with broadband loop elements," *IET Microwave Antennas Propag.*, vol. 4., no. 2, pp. 225-231, Feb. 2010.

[50] R. Deng, Y. Mao, S. Xu, and F. Yang, "A single-layer dual-band circularly polarized reflectarray with high aperture efficiency," *IEEE Trans. Antennas Propag.*, vol. 63, no. 7, pp. 3317-3320, Jul. 2015.

[51] R. J. Langley and E. A. Parker, "Double-square frequency selective surfaces and their equivalent circuit," *Electron. Lett.*, vol. 19, no. 17, pp. 675-677, Aug. 1983.

[52] S. Monni, G. Gerini, A. Neto, and A. G. Tijhuis, "Multimode equivalent networks for the design and analysis of frequency selective surfaces," *IEEE Trans. Antennas Propag.*, vol. 55, no. 10, pp. 2824-2835, Oct. 2007.

[53] R. Florencio, R. R. Boix, and J. A. Encinar, "Fast and accurate MoM analysis of periodic arrays of multilayered stacked rectangular patches with applications to the design of reflectarray antennas," *IEEE Trans. Antennas Propag.*, vol. 63, no. 6, pp. 2558-2570, Jun. 2015.

[54] C. Wan and J. A. Encinar, "Efficient computation of generalized scattering matrix for analyzing multilayered periodic structures," *IEEE Trans. Antennas Propag.*, vol. 43, no. 11, pp. 1233-1242, Nov. 1995.

[55] F. Costa, A. Monorchio, and G. Manara, "Efficient analysis of frequency-selective surfaces by a simple equivalent-circuit model," *IEEE Antennas Propag. Mag.,* vol. 54, no. 4, pp. 35-48, Aug. 2012.

[56] D. Bresciani, "A unified approach to the characterization of frequency and polarization selective surfaces," in *Proceedings of IEEE Antennas and Propagation Society International Symposium*, Ann Arbor, MI, USA, 1993, pp. 1960-1963, vol. 3.

[57] S. Contu and R. Tascone, "Scattering from passive arrays in plane stratified regions," *Electromagnetics*, vol. 5, no. 4, pp. 285-306, 1985.

[58] M. Salucci, N. Anselmi, G. Oliveri, P. Calmon, R. Miorelli, C. Reboud, and A. Massa, "Real-time NDT-NDE through an innovative adaptive partial least squares SVR inversion approach," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 11, pp. 6818-6832, Nov. 2016.

[59] S. N. Lophaven, H. B. Nielsen, and J. Sondergaard, "DACE, a MATLAB Kriging toolbox," Technical University of Denmark, Aug. 2002.

[60] F. Viani, P. Rocca, M. Benedetti, G. Oliveri, and A. Massa, "Electromagnetic passive localization and tracking of moving targets in a WSN-infrastructured environment," *Inverse Probl.,* vol. 26, pp. 1-15, May 2010.

[61] J.-P. Costa, L. Pronzato, and E. Thierry, "A comparison between Kriging and Radial Basis Function networks for nonlinear prediction," *Proceedings of the IEEE-EURASIP Workshop on Nonlinear Signal and Image Processing*, Antalya, Turkey, June 20-23, 1999.

[62] D. Cadoret, L. Marnat, R. Loison, R. Gillard, H. Legay, and B. Salome, "A dual linear polarized printed reflectarray using slot loaded patch elements," *II European Conference on Antennas and Propagation*, Edinburgh, 2007, pp. 1-5.