Automatic Emulation by Adaptive Relevance Vector Machines

Luca Martino, Jorge Vicent, and Gustau Camps-Valls*

Image Processing Laboratory (IPL), Universitat de València, Spain
luca.martino@uv.es, http://isp.uv.es

Abstract. This paper introduces an automatic methodology to construct emulators for costly radiative transfer models (RTMs). The proposed method is sequential and adaptive, and it is based on the notion of the acquisition function by which instead of optimizing the unknown RTM underlying function we propose to achieve accurate approximations. The proposed methodology combines the interpolation capabilities of a modified Relevance Vector Machine (RVM) with the accurate design of an acquisition function that favors sampling in low density regions and flatness of the interpolation function. The proposed Relevance Vector Machine Automatic Emulator (RAE) is illustrated in toy examples and for the construction of an optimal look-up-table for atmospheric correction based on MODTRAN5.

Keywords: Radiative transfer model, Relevance Vector Machines, emulation, self-learning, look-up table, interpolation, MODTRAN

1 Introduction

In many fields of Science, Engineering and Technology, the mathematical and physical models are implemented in computer programs known as *simulators* [26]. Simulators are increasingly popular nowadays in social sciences, social network modeling and (electronic) commerce as well. These simulators aim to model and reproduce the complex real-world phenomena accurately. On the downside, simulators typically require large computational cost and memory resources, as well as the introduction of complicated *ad hoc* rules in the programs. Despite their good performance and reputation in the related fields, these shortcomings impede its wide practical adoption. In the last decades, machine learning has played a key role in the field by proposing surrogate models, known commonly as *emulators*, which try to reproduce (learn) the complex input-output mapping built by simulators from empirical data. This is typically done by running the simulator for a number of input factors and situations, thus yielding a training dataset, which is then used to fit regression models able to perform out-of-sample predictions.

* The research was funded by the European Research Council (ERC) under the ERC-CoG-2014 SEDAL project (grant agreement 647423), and the Spanish Ministry of Economy and Competitiveness (MINECO) through the project TIN2015-64210-R.

Statistical emulators were actually developed in the 1980s for general purposes [9, 22, 26]. Emulators typically rely on Gaussian Processes and neural networks because of their flexibility, accuracy, and computational efficiency. Once the emulator model is developed, one can run *approximate* simulations very efficiently because the testing time for machine learning models is commonly very low (often linear or sub-linear). This actually allows one to analyze the system for which the simulator was built with far more instantiations and situations, as well as to perform *sensitivity analysis* (that is, analyze the relative relevance of the drivers) in a more robust manner. We find interesting applications of Gaussian process emulators for input feature selection and sensitivity analysis, and uncertainty quantification of the outputs given the uncertainty of the inputs [5, 20].

In this paper, we will focus on the perhaps most active field nowadays building emulators, that of Earth Science. In Earth observation and climate science one typically has access to physical models encoding the variable relations. These physical models are either called process-based models in global change ecology, radiative transfer models (RTMs) in remote sensing, or climate models in detection-and-attribution schemes for climate science applications. RTMs simulate the system as $\mathbf{y} = f(\mathbf{x}, \boldsymbol{\omega})$, where $\mathbf{y}$ is a measurement obtained by the satellite (e.g. radiance); the vector $\mathbf{x}$ represents the state of the biophysical variables on the Earth; $\boldsymbol{\omega}$ contains a set of controllable conditions (e.g. wavelengths, viewing direction, time, Sun position, and polarization); and $f(\cdot)$ is a function which relates $\mathbf{y}$ with $\mathbf{x}$. Such a function f is typically considered to be nonlinear, smooth and continuous. The goal in inverse modeling is to obtain an accurate model $g(\cdot) \approx f^{-1}(\cdot)$, parametrized by $\boldsymbol{\theta}$, which approximates the biophysical variables $\mathbf{x}$ given the data $\mathbf{y}$ received by the satellite, i.e. $\hat{\mathbf{x}} = g(\mathbf{y}, \boldsymbol{\theta})$. In emulation mode, however, we are interested in approximating the RTM well, that is, obtain a machine learning model $\widehat{f}$ that approximates the RTM code f at a fraction of time and accurately. Obtaining such a model is a game changer since one can do model inversion, sensitivity analysis and parameter retrieval much more efficiently than with the original simulator.

Here we are concerned about using the emulator to replace RTMs and then perform model inversion. Such inversion typically requires large multi-dimensional look-up tables (LUTs), which are precomputed for their later interpolation [11]. However, the computation of these LUTs still imposes a large computation burden, requiring techniques of parallelization and execution in computer grids. In order to further reduce this computation burden, a possible strategy is to select an optimal subset of *anchor or landmark points* in order to reduce the error of the interpolation of LUTs. Compact and informative LUTs give raise, in turn, to interesting possibilities for emulating RTMs [25]. In this work, we address the problem of optimal selection of the points to be included in the LUT and the construction of the emulator simultaneously.

The field has received attention from (apparently unrelated) fields in statistical signal processing and machine learning. The problem has been cast as *experimental optimal design* [6, 18, 21, 27] of interpolators of arbitrary functions f. To reduce the number of direct runs of the system (evaluations of f), a possible approach is to construct an approximation of f starting with a set of support points. This approximation is then sequentially improved incorporating novel points given a suitable statistical rule. This topic is also related to different research areas: optimal nonuniform sampling, quantization and interpolation of continuous signals [8, 17], the so-called Bayesian Optimization (BO) problem [12, 19, 28], and active learning [2, 7, 10, 29]. Finally, an interesting alternative approach is based on *adaptive gridding*, where the aim is to construct a partitioning of $\mathcal{X}$ into cells of equal size, where the cell edges have different lengths depending on their spatial direction. This was the approached followed in [4]. In order to find these lengths, the proposed method uses a modification of the Relevance Vector Machines (RVMs) [3, 23].

In this paper we introduce a simpler and more general approach. The proposed method is a sequential, adaptive and automatic construction of the emulator based on the notion of the *acquisition function*, similarly to the BO approach [12, 19]. Unlike in BO, our goal is not the optimization of the unknown underlying function f but its accurate *approximation* $\widehat{f}$. Given a set of initial points, the emulator is built automatically with the online addition of new nodes maximizing the acquisition function at each iteration. Theoretically, the acquisition function should incorporate (a) geometric information of the unknown function f, and (b) information about the distribution of the current nodes. Indeed, areas of high variability of $f(\mathbf{x})$ require the addition of more points as well as areas with a small concentration of nodes require the introduction of new inputs. Thus, the experimental design problem is converted into a sequential optimization problem where the function to be optimized involves geometric and spatial information (regardless of the dimensionality of the input space).

The rest of the paper is outlined as follows. Next section §2 describes the general ideas behind the automatic emulation. In Section 3, we introduce the proposed automatic emulator scheme, and revises each of the building blocks (regression and acquisition functions). In Section §4, we give experimental evidence of performance of the proposal in several numerical examples, one of them involving the optimization of the nodes for the construction of an optimal LUT for atmospheric correction involving a computationally demanding physical model. We conclude in §5 with some remarks and outline of the further work.

2 Automatic Emulation

This section introduces the proposed scheme for automatic emulation. We start by fixing the notation and presenting the processing scheme. Then we will detail all the ingredients that allows to design an adaptive acquisition function in conjunction with the RVM model.

2.1 Emulation scheme

Let us consider a D-dimensional input space $\mathcal{X}$, i.e., $\mathbf{x} = [x_1, \ldots, x_D]^\intercal \in \mathcal{X} \subset \mathbb{R}^D$ and, for the sake of simplicity, we assume that $\mathcal{X}$ is bounded. Let us denote the system to be emulated as $f(\mathbf{x}) : \mathcal{X} \mapsto \mathbb{R}$, e.g. a complicated transfer equations

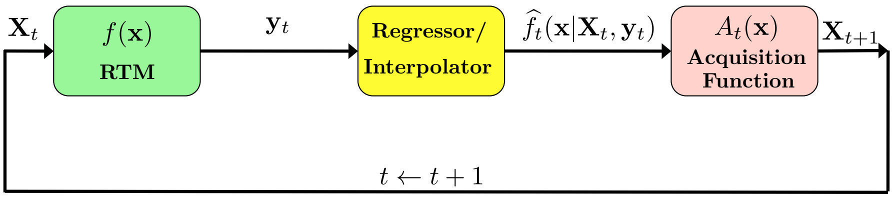


Fig. 1. Graphical representation of an automatic emulator. In this work, the function $f(\mathbf{x})$ represents RTM but, more generally, it can be any function costly to be evaluated.

modeled with an expensive RTM. Given an input matrix $\mathbf{X}_t = [\mathbf{x}_1, \cdots, \mathbf{x}_{m_t}]$ of dimension $D \times m_t$, we have a vector of outputs $\mathbf{y}_t = [y_1, \dots, y_{m_t}]^\intercal$, where $y_t = f(\mathbf{x}_t)$, where the index $t \in \mathbb{N}^+$ denotes the t-th iteration of the algorithm. Essentially, at each iteration t one performs an *interpolation* or a *regression*, providing $\widehat{f}_t(\mathbf{x}|\mathbf{X}_t, \mathbf{y}_t)$, followed by an *optimization* step that updates the acquisition function, $A_t(\mathbf{x})$, updates the set $\mathbf{X}_{t+1} = [\mathbf{X}_t, \mathbf{x}_{m_t+1}]$ adding a new node, set $m_{t+1} = m_t + 1$ and $t \leftarrow t + 1$. The procedure is repeated until a suitable stopping condition is satisfied. We assume scalar outputs in order to simplify the description of the technique, yet the algorithm can be easily extended to multi-output settings. The algorithm is outlined in Alg. 1 and Figure 1 shows a graphical representation of a generic automatic emulator. In this work, $f(\mathbf{x})$ represents RTM but, more generally, it could be any costly function.

Algorithm 1 Automatic Emulation.

```
1: Regression: Apply an regression procedure, following some pre-established
   method, providing $\widehat{f}_t(\mathbf{x}|\mathbf{X}_t, \mathbf{y}_t)$, given the current matrix of input/support points
   $\mathbf{X}_t = [\mathbf{x}_1, \dots, \mathbf{x}_{m_t}]^\intercal$, and $\mathbf{y}_t = [y_1, \dots, y_{m_t}]^\intercal$.
2: Optimization: Update the acquisition function obtaining $A_t(\mathbf{x})$ and set
       $\mathbf{x}_{m_t+1} = \max_{\mathbf{x} \in \mathcal{X}} A_t(\mathbf{x}).$
   Moreover, include the new point in the set of nodes, i.e., $\mathbf{X}_{t+1} = [\mathbf{X}_t;\ \mathbf{x}_{m_t+1}]$, and
   set $m_{t+1} = m_t + 1$.
3: Check Stop Condition: For instance, whether the number of desired points was
   reached, or if $\|\widehat{f}_t(\mathbf{x}) - \widehat{f}_{t-1}(\mathbf{x})\| \leq \epsilon$, then stop. Otherwise, set $t \leftarrow t + 1$ and come
   back to step 1.
```

2.2 Conceptual Definition of the Acquisition Function

Let us start describing the general properties that a generic acquisition function $A_t(\mathbf{x})$ should satisfy. Conceptually, an acquisition function can be defined as the product of two functions, a *geometry term* $G_t(\mathbf{x})$ and a *diversity term* $D_t(\mathbf{x})$:

$$A_t(\mathbf{x}) = G_t(\mathbf{x}) D_t(\mathbf{x}), \tag{1}$$

where $G_t(\mathbf{x}) : \mathcal{X} \mapsto \mathbb{R}^+$, $D_t(\mathbf{x}) : \mathcal{X} \mapsto \mathbb{R}^+$ and hence $A_t(\mathbf{x}) : \mathcal{X} \mapsto \mathbb{R}^+$ (i.e., $A(\mathbf{x}) \geq 0$). The first function $G_t(\mathbf{x})$ represents some suitable geometrical information of the hidden function f. The second function $D_t(\mathbf{x})$ depends on the distribution of the points in the current vector $\mathbf{X}_t$. More specifically, $D_t(\mathbf{x})$ will have a greater probability mass around empty areas within $\mathcal{X}$, whereas $D_t(\mathbf{x})$ will be approximately zero close to the support points and exactly zero at the support points, i.e., $D_t(\mathbf{x}_i) = 0$, for $i = 1, \ldots, m_t$ and $\forall t \in \mathbb{N}$. As a consequence, a suitable acquisition function satisfies the following condition,

$$A_t(\mathbf{x}_i) = 0, \quad \text{for} \quad i = 1, \ldots, m_t, \quad \text{and} \quad \forall t \in \mathbb{N}. \tag{2}$$

3 RVM Automatic Emulator (RAE)

In this section, we specify the implementation of an automatic emulator based on a variant of the RVM method, called RAE. Thus, we introduce the Adaptive RVM (A-RVM) (employed as regressor in RAE) and then we describe a suitable construction of an acquisition function $A_t(\mathbf{x})$, taking into account important information provided by the hyperparameters of A-RVM previously optimized.

3.1 Adaptive Relevance Vector Machine (A-RVM)

Let us consider the standard regression model

$$y = f(\mathbf{x}) + \epsilon,$$

where $\epsilon \sim \mathcal{N}(0, \sigma_e^2)$ and we observe N data pairs, $\{\mathbf{x}_n, y_n\}_{n=1}^N$. We also denote $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]$, $\mathbf{y} = [y_1, \ldots, y_N]^\intercal$, and define the following N basis functions

$$\boldsymbol{\phi}_n(\mathbf{x}|\mathbf{x}_n) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}_n\|^2}{2\delta_n^2}\right) : \mathcal{X} \times \mathcal{X} \to \mathbb{R} \tag{3}$$

Hence, we have N functions centered in the data inputs $\mathbf{x}_n$'s, each one with a different scale parameter δ_n^2. Moreover, let us define the $N \times N$ matrix $[\boldsymbol{\Phi}]_{i,j} := \boldsymbol{\phi}_j(\mathbf{x}_i|\mathbf{x}_j)$, and the $N \times 1$ vector $\boldsymbol{\phi}(\mathbf{x}, \mathbf{X}) = [\boldsymbol{\phi}_1(\mathbf{x}|\mathbf{x}_1), \ldots, \boldsymbol{\phi}_N(\mathbf{x}|\mathbf{x}_N)]^\intercal$. We assume that the hidden function f can be expressed as $f(\mathbf{x}) = \boldsymbol{\phi}(\mathbf{x}, \mathbf{X})^\intercal \mathbf{w}$, where $\mathbf{w}$ is an unknown vector. Furthermore, we consider a Gaussian prior over the $N \times 1$ weight vector $\mathbf{w}$, i.e., $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_p)$. The Minimum Mean Square Error (MMSE) solution is

$$\widehat{\mathbf{w}} = \frac{1}{\sigma_e^2}\left(\frac{1}{\sigma_e^2}\boldsymbol{\Phi}\boldsymbol{\Phi}^\intercal + \boldsymbol{\Sigma}_p^{-1}\right)^{-1}\boldsymbol{\Phi}\mathbf{y} = \boldsymbol{\Sigma}_p\boldsymbol{\Phi}\left(\boldsymbol{\Phi}^\intercal\boldsymbol{\Sigma}_p\boldsymbol{\Phi} + \sigma_e^2\mathbf{I}_N\right)^{-1}\mathbf{y}. \tag{4}$$

Hence, since $\widehat{f}(\mathbf{x}) = \boldsymbol{\phi}(\mathbf{x}, \mathbf{X})^\intercal\widehat{\mathbf{w}}$, then

$$\widehat{f}(\mathbf{x}) = \boldsymbol{\phi}(\mathbf{x}, \mathbf{X})^\intercal\boldsymbol{\Sigma}_p\boldsymbol{\Phi}\left(\boldsymbol{\Phi}^\intercal\boldsymbol{\Sigma}_p\boldsymbol{\Phi} + \sigma_e^2\mathbf{I}_N\right)^{-1}\mathbf{y} = \mathbf{k}^\intercal\left(\mathbf{K} + \sigma_e^2\mathbf{I}_N\right)^{-1}\mathbf{y}, \tag{5}$$

where we have set $\mathbf{K} = \boldsymbol{\Phi}^\intercal\boldsymbol{\Sigma}_p\boldsymbol{\Phi}$ and $\mathbf{k}^\intercal = \boldsymbol{\phi}(\mathbf{x}, \mathbf{X})^\intercal\boldsymbol{\Sigma}_p\boldsymbol{\Phi}$.

On the adaptive lengthscales. We remark again that the number of basis function $\boldsymbol{\phi}_n$ is exactly N as the number of data and $N + 1$ hyperparameters, $\boldsymbol{\theta} = [\delta_1, \ldots, \delta_N, \sigma_e^2]^\intercal$, N scale parameters δ_n (one per each function $\boldsymbol{\phi}_n$) and the variance of the measurement noise σ_e^2.

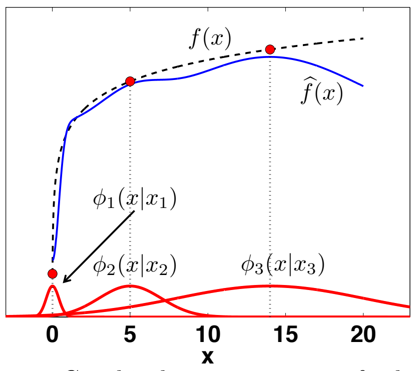


Fig. 2. Graphical representation of adaptive RVM regression model. The regressor $\widehat{f}(x) = \sum_{n=1}^{3} w_n \boldsymbol{\phi}_n(x|x_n)$ is a linear combination of $N = 3$ basis functions, each one with a different scale δ_n, $n = 1, 2, 3$.

Clearly, the use of different δ_n's fosters the flexibility of the regression method. Moreover, and more important for our purpose, each parameter δ_n contains *geometric information* about the hidden function $f(\mathbf{x})$. Indeed, the parameters δ_n's of the functions $\boldsymbol{\phi}_n$'s located in regions with a greater variation of $f(\mathbf{x})$ are smaller than the parameters δ_n's of the functions $\boldsymbol{\phi}_n$'s located in regions where $f(\mathbf{x})$ is flatter. Roughly speaking, a great value of δ_n means that $\boldsymbol{\phi}_n$ is located in an area where f is virtually flat, whereas a small value of δ_n is obtained when $\boldsymbol{\phi}_n$ is located in a region where f has an high derivative, for instance. This consideration is very useful in order to build a suitable acquisition function. An illustrative example of the adaptive property is shown in Fig. 2.

On hyperparameter tuning. The tuning of the hyperparameters is performed maximizing the log-marginal likelihood,

$$J(\boldsymbol{\theta}) = \log[p(\mathbf{y}|\boldsymbol{\theta})] = -\frac{1}{2}\mathbf{y}^\intercal(\mathbf{K} + \sigma_e^2\mathbf{I}_N)^{-1}\mathbf{y} - \frac{1}{2}\log\left[\det(\mathbf{K} + \sigma_e^2\mathbf{I}_N)\right] + c,$$

where c is a constant and $\mathbf{K} = \boldsymbol{\Phi}^\intercal \boldsymbol{\Sigma}_p \boldsymbol{\Phi}$. We consider a simulated annealing approach for finding the global maximum of $J(\boldsymbol{\theta})$ [13,15,24]. More sophisticated optimization methods could exploit information about the gradient of $J(\boldsymbol{\theta})$.

3.2 Proposed Acquisition Function

As we have observed before, the N learnt parameters δ_n's of M-RVM contain geometric information about the hidden function $f(\mathbf{x})$. Thus, in order to create a suitable acquisition function we need to take into account also the distribution of the other inputs in the current iteration of the algorithm. A possibility is to define the acquisition function as

$$A(\mathbf{x}) = \prod_{i=1}^{m_t} V_i(\mathbf{x}|\mathbf{x}_i), \tag{6}$$

where $V_i(\mathbf{x}|\mathbf{x}_i) = (\mathbf{x} - \mathbf{x}_i)^2 \exp(-\frac{(\mathbf{x}-\mathbf{x}_i)^2}{\ell\delta_i})$, being $\ell > 0$ a parameter chosen by the user. The function $V_i(\mathbf{x}|\mathbf{x}_i)$ has the analytical structure of a Nakagami density [14]. The Nakagami density has been widely studied from an analytical point of view. Thus, different features of $V_i(\mathbf{x}|\mathbf{x}_i)$ are known analytically and it can be easily normalized.

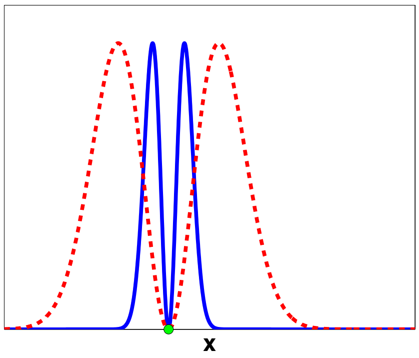


Fig. 3. Example of two functions $V_i(\mathbf{x}|\mathbf{x}_i)$ for the same $\mathbf{x}_i$ (circle) with a high (solid) and low (dashed) δ_n.

Note that each $V_i(\mathbf{x}|\mathbf{x}_i)$ depends on the parameter δ_i and $V_i(\mathbf{x}_i|\mathbf{x}_i) = 0$. Hence, $A(\mathbf{x}_i) = 0$ for all $i = 1, \dots, m_t$. The function $V_i(\mathbf{x}|\mathbf{x}_i)$ is symmetric with respect to $\mathbf{x}_i$ and it is bimodal with modes located at

$$\mathbf{x}^* = \mathbf{x}_i \pm \sqrt{\ell \delta_i}. \tag{7}$$

When δ_i is small, the two modes are closer to each other and closer to $\mathbf{x}_i$, whereas when δ_i is high the modes are far away from $\mathbf{x}_i$. Figure 4 shows a sequence of approximations $\{\widehat{f}_t(x)\}_{t=0}^3$ of $f(x)$, and the corresponding acquisition functions at each iteration.

4 Experimental Results

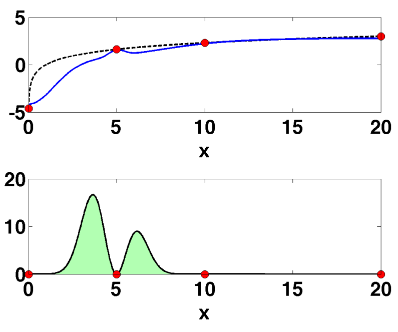
4.1 Synthetic Example

In order to test RAE, first we consider a toy example where we can compare the achieved approximation $\widehat{f}_t$ with the underlying function f which is unknown in the real-world applications. In this way, we can exactly check the true accuracy of the obtained approximation using different schemes. For the sake of simplicity, we consider the function

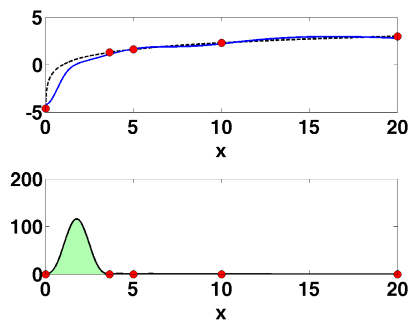
$$f(x) = \log(x), \tag{8}$$

with $x \in \mathcal{X} = (0, 20]$ hence $D = 1$ (see Fig. 5). Even in this simple scenario, the procedure used for selecting new points is relevant as confirmed the results provided below. We start with $m_0 = 3$ support points, $\mathbf{X}_0 = [0.01, 10, 20]$. We add sequentially and automatically other 10 points using different techniques: **(a)** completely random choice, uniformly within $\mathcal{X} = [0, 20]$, **(b)** a deterministic procedure filling the greatest distance between two consecutive points, adding the middle point of this interval, **(c)** Latin Hypercube (LHC) sampling [18], **(d)** RAE with $\ell = 6$ in the functions $V_i(\mathbf{x}|\mathbf{x}_i)$. In all cases, we use the M-RVM regression scheme (for simplicity, we also set $\sigma_e^2 = 0.1$). The optimization of the remaining hyperparameters is obtained by a parallel simulated annealing approach [13,15]. Note the deterministic procedure always adds sequentially the following points, obtaining $\mathbf{X}_{10} = [\mathbf{X}_0, 15, 5.005, 12.5, 17.5, 7.5025, 11.25, 13.75, 16.25, 18.75]$ (recalling that $\mathbf{X}_0 = [0.01, 10, 20]$ for all methods). At each run, the results can slightly vary even for the deterministic procedure due to the optimization of the hyperparameters (we use a simulated annealing approach that is a stochastic optimization technique [13]). We average all the results over 500 independent runs.

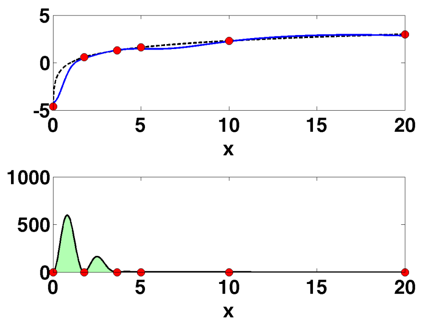
We compute the L_2 distance between $\widehat{f}_t(x)$ and $f(x)$ at each iteration, obtained by the different method We show the evolution of the averaged L_2 distance



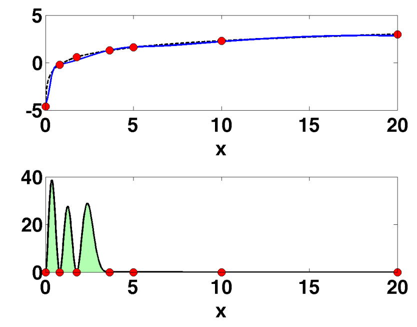
(a) $t = 0$



(b) $t = 1$



(c) $t = 3$



(d) $t = 4$

Fig. 4. A sequence of approximations $\widehat{f}_0(x)$, $\widehat{f}_1(x)$, $\widehat{f}_2(x)$ and $\widehat{f}_3(x)$ (top) of the function $f(x) = \log(x)$ (dashed line) obtained by RAE starting with 3 points and the corresponding acquisition functions $A_0(x)$, $A_1(x)$, $A_2(x)$, $A_3(x)$ and $A_4(x)$ (bottom). The nodes are depicted by circles. In this example, we have set $\ell = 6$.

versus the number of support points m_t (that is $m_t = t+m_0$) in Figure 5 (specifically, we show the median curves obtained over the 500 runs). We can observe that the RAE scheme outperforms the other methods, providing the smallest distances between f and $\widehat{f}_t$. The proposed technique gives a clear advantage in the first iterations: with only $m_t = 7$ points provides an error of $\approx 10^{-1}$, whereas the other methods require more than 12 points for obtaining the same error value. After adding a certain number of nodes (enough for covering the entire domain $\mathcal{X}$), clearly the results become similar. The difficulty of the automatic problem is clearly represented by the curve of the deterministic procedure: the attempt of filling as fast as possible all the domain $\mathcal{X}$ trying to maximizing the distance among the nodes is not the best strategy, in general. In some specific scenarios, after the addition of one point, the performance can even get worse (it can vary depending on the regression scheme employed). This happens with the deterministic procedure and the addition of the 6-th and 7-th node (see Fig.

5). Figure 4 depicts the function $f(x)$ in Eq. (8), the approximation $\widehat{f}_t(x)$, and the acquisition function $A_t(x)$ obtained in four consecutive iterations of RAE, considering a specific run of the algorithm.

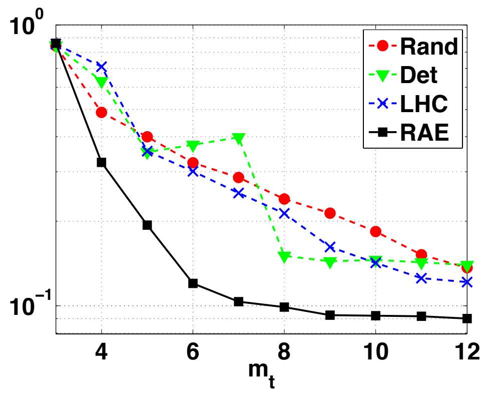


Fig. 5. The L_2 distance (in log-scale) between $f(x)$ and $\widehat{f}_t(x)$ versus the number of the number of support points m_t, that is $m_t = t + 3$ in this example. The results of RAE are shown with squares, the LHC curve is shown with x-marks, the curve of the deterministic procedure is depicted with triangles and the results of the completely random approach is illustrated by circles.

4.2 Emulation of Costly Radiative Transfer Codes

Our second example focuses on the optimization of selected points for a MODTRAN5-based LUT. MODTRAN5 is considered as *de facto* standard atmospheric RTM for atmospheric correction applications [1]. This RTM solves the radiative transfer equation in the atmosphere considering the effect of scattering and absorption by gasses and aerosols for a flexible configuration of viewing and illumination conditions and surface reflectance. In our test application, and for the sake of simplicity, we have considered $D = 2$ with the Aerosol Optical Thickness at 550 nm (τ) and ground elevation (h) as key input parameters. The underlying function $f(\mathbf{x})$ consists therefore on the execution of MODTRAN5 at given values of τ and h at the single output wavelength of 760 nm (i.e. bottom of the O_2-A band). As the parameters a_i, b_i and c in the toy example above, other MODTRAN5 input parameters are set to standard atmospheric and geometric conditions (e.g. mid-latitude summer atmosphere, rural aerosol, nadir view, 55° solar zenith angle, sensor height at 100 km). The input parameter space is bounded to 0.05-0.4 for τ and 0-3 km for h. In order to test the accuracy of the different schemes, we have evaluated $f(x)$ at all the possible 1750 combinations of 35 values of τ and 50 values of h. Namely, this thin grid represents the ground-truth in this example.

Table 1. Averaged number of nodes needed for obtaining a L_2 error $\approx \eta$.

L_2 **error** η	**Random**	**Latin Hypercube**	**RAE**
0.2	19.25	11.03	7.58
0.03	28.43	16.69	11.19

We test **(a)** a random approach choosing points uniformly within $\mathcal{X} = [0.05, 0.4] \times [0, 3]$, **(b)** the Latin Hypercube (LHC) sampling (see, e.g., [18]) and **(c)** RAE ($\ell = 3$). We use the simulated annealing algorithm [13, 15] for both, optimizing the hyper-parameters of M-RVM and finding the maximum of the acquisition function $A_t(\mathbf{x})$. We start with $m_0 = 5$ points $\mathbf{x}_1 = [0.05, 0]^\top$, $\mathbf{x}_2 = [0.05, 3]^\top$, $\mathbf{x}_3 = [0.4, 0]^\top$, $\mathbf{x}_4 = [0.4, 3]^\top$ and $\mathbf{x}_5 = [0.2, 1.5]^\top$ for all the techniques. We compute the final number of nodes m_t required to obtain an ℓ_2 distance between f and $\widehat{f}$ approximately of $\eta \in \{0.03, 0.2\}$, with the different methods. The results, averaged over 10^3 runs, are shown in Table 1. RAE requires the addition of ≈ 3 new points to obtain a distance ≈ 0.2 and ≈ 6 new points to obtain a distance ≈ 0.03.

5 Conclusions

We introduced an automatic method to construct surrogate models, also known as emulators, and optimal look-up-tables for costly physical models (as RTMs). We proposed an iterative scheme combining the interpolation capabilities of RVMs with the design of an acquisition function that fosters a suitable choice of the nodes and flatness of the interpolation function. We illustrated the good capabilities of the method in a synthetic example and a real example involving atmospheric correction based on the computationally expensive MODTRAN5 model. Future work is tied to the development of multi-output schemes and testing in other costly RTMs.

References

1. Berk, A., Anderson, G., Acharya, P., Bernstein, L., Muratov, L., Lee, J., Fox, M., Adler-Golden, S., Chetwynd, J., Hoke, M., Lockwood, R., Gardner, J., Cooley, T., Borel, C., Lewis, P., Shettle, E.: MODTRAN5: 2006 update. The International Society for Optical Engineering (2006)
2. Beygelzimer, A., Dasgupta, S., Langford, J.: Importance-weighted active learning. International Conference on Machine Learning (ICML) pp. 49–56 (2009)
3. Bishop, C.M.: Pattern recognition. Machine Learning 128, 1–58 (2006)
4. Busby, D.: Hierarchical adaptive experimental design for Gaussian process emulators. Reliability Engineering and System Safety 94, 1183–1193 (2009)
5. Camps-Valls, G., Verrelst, J., Muñoz Marí, J., Laparra, V., Mateo-Jiménez, F., Gomez-Dans, J.: A survey on gaussian processes for earth observation data analysis. IEEE Geoscience and Remote Sensing Magazine (6) (June 2016)
6. Chaloner, K., Verdinelli, I.: Bayesian experimental design: A review. Statistical Science 10(3), 237–304 (1995)

7. Cohn, D., Ghahramani, Z., Jordan, M.: Active learning with statistical models. Journal of Artificial Intelligence Research 4, 129–145 (1996)
8. Cover, T.M., Thomas, J.A.: Elements of Information Theory. Wiley-Interscience, New York (USA) (1991)
9. Currin, C., Mitchell, T., Morris, M., Ylvisaker, D.: A Bayesian approach to the design and analysis of computer experiments (Sep 1988)
10. Dasgupta, S.: Analysis of a greedy active learning strategy. In Advances inNeural Information Processing Systems (NIPS) 16(3), 337–344 (2004)
11. Guanter, L., Richter, R., Kaufmann, H.: On the application of the MODTRAN4 atmospheric radiative transfer code to optical remote sensing. International Journal of Remote Sensing 30(6), 1407–1424 (2009)
12. Gutmann, M.U., Corander, J.: Bayesian optimization for likelihood-free inference of simulator-based statistical models. Journal of Machine Learning Research 16, 4256–4302 (2015)
13. Kirkpatrick, S.K., Jr., C.D.G., Vecchi, M.P.: Optimization by simulated annealing. Science 220(4598), 671–680 (May 1983)
14. Luengo, D., Martino, L.: Almost rejectionless sampling from Nakagami-m distributions ($m \geq 1$). IET Electronics Letters 48(24), 1559–1561 (2012)
15. Martino, L., Elvira, V., Luengo, D., Corander, J., Louzada, F.: Orthogonal parallel MCMC methods for sampling and optimization. Digital Signal Processing 58, 64–84 (2016)
16. Marvasti, F.: Nonuniform sampling: Theory and Practice. Kluwer Academic Publishers (2001)
17. McKay, M.D., Beckman, R.J., Conover, W.J.: A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. Technometrics 21(2), 239–245 (1979)
18. Mockus, J.: Bayesian approach to global optimization. Kluwer Academic Publishers, Dordrecht (1989)
19. Oakley, J.E., O'Hagan, A.: Probabilistic sensitivity analysis of complex models: A bayesian approach. Journal of the Royal Statistical Society 66B, 751–769 (2004)
20. O'Brien, T.E., Funk, G.M.: A gentle introduction to optimal design for regression models. The American Statistician 57(4), 265–267 (2003)
21. O'Hagan, A.: Curve fitting and optimal design for predictions. Journal of the Royal Statistical Society 40B, 1–42 (1978)
22. Rasmussen, C.E., Williams, C.K.I.: Gaussian Processes for Machine Learning. The MIT Press (2005)
23. Read, J., Martino, L., Luengo, D.: Efficient Monte Carlo optimization for multi-label classifier chains. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP) pp. 1–5 (2013)
24. Rivera, J., Verrelst, J., Gómez-Dans, J., Muñoz Marí, J., Moreno, J., Camps-Valls, G.: An emulator toolbox to approximate radiative transfer models with statistical learning. Remote Sensing 7(7), 9347–9370 (2015)
25. Sacks, J., Welch, W.J., Mitchell, T.J., , Wynn, H.P.: Design and analysis of computer experiments. Statistical Science (4), 409423 (1989)
26. da Silva Ferreira, G., Gamerman, D.: Optimal design in geostatistics under preferential sampling. Bayesian Analysis 10(3), 711–735 (2015)
27. Snoek, J., Larochelle, H., Adams, R.P.: Practical Bayesian optimization of machine learning algorithms. Neural Information Processing Systems (NIPS) - arXiv:1206.2944 pp. 1–9 (2012)
28. Verrelst, J., Dethier, S., Rivera, J., Muñoz-Marí, J., Camps-Valls, G., Moreno, J.: Active learning methods for efficient hybrid biophysical variable retrieval. IEEE Geoscience and Remote Sensing Letters 13(7), 1012–1016 (2016)