Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2016, Article ID 8241275, 17 pages
http://dx.doi.org/10.1155/2016/8241275

Research Article

Two Simulated Annealing Optimization Schemas for Rational Bézier Curve Fitting in the Presence of Noise

Andrés Iglesias,[1,2] Akemi Gálvez,[1] and Carlos Loucera[1,3]

[1]*Department of Applied Mathematics and Computational Sciences, E.T.S.I. Caminos, Canales y Puertos, University of Cantabria, Avenida de los Castros, s/n, 39005 Santander, Spain*
[2]*Department of Information Science, Faculty of Sciences, Toho University, 2-2-1 Miyama, Funabashi 274-8510, Japan*
[3]*State Meteorological Agency (AEMET), 28071 Madrid, Spain*

Correspondence should be addressed to Andrés Iglesias; iglesias@unican.es

Received 31 August 2015; Revised 18 November 2015; Accepted 24 November 2015

Academic Editor: Anna Pandolfi

Fitting curves to noisy data points is a difficult problem arising in many scientific and industrial domains. Although polynomial functions are usually applied to this task, there are many shapes that cannot be properly fitted by using this approach. In this paper, we tackle this issue by using rational Bézier curves. This is a very difficult problem that requires computing four different sets of unknowns (data parameters, poles, weights, and the curve degree) strongly related to each other in a highly nonlinear way. This leads to a difficult continuous nonlinear optimization problem. In this paper, we propose two simulated annealing schemas (the all-in-one schema and the sequential schema) to determine the data parameterization and the weights of the poles of the fitting curve. These schemas are combined with least-squares minimization and the Bayesian Information Criterion to calculate the poles and the optimal degree of the best fitting Bézier rational curve, respectively. We apply our methods to a benchmark of three carefully chosen examples of 2D and 3D noisy data points. Our experimental results show that this methodology (particularly, the sequential schema) outperforms previous polynomial-based approaches for our data fitting problem, even in the presence of noise of low-medium intensity.

1. Introduction

(1) Motivation. This paper deals with the problem of fitting a collection of noisy data points by means of a rational curve. This problem arises in several scientific and applied fields. It is an important issue, for instance, in regression analysis for statistics and machine learning and in approximation theory for numerical analysis. It also plays a key role in several industrial fields, most prominently in reverse engineering. In its most comprehensive meaning, reverse engineering consists of obtaining a digital replica of an already existing physical object or component. This is a typical procedure in medical and health areas, where noninvasive techniques such as magnetic resonance imaging (MRI) or computer tomography (CT) are commonly used to visualize inner organs or different parts of the human body for medical check, diagnosis, and therapy. Reverse engineering is also a common practice in consumer products, microchips, and other electronic components for different purposes. For instance, to analyze how a new device or machine in the market is built or how a particular component works. Another relevant application arises in automotive and aerospace industries, where prototypes are built on clay, foam rubber, wood, and other materials to help designers and engineers explore new ideas about shape or size and get a visual insight of a new part to be designed. This is a challenging task, since nowadays designs are becoming more and more organic in shape, making them more difficult to be replicated by computer from scratch. A typical approach in this regard is to obtain a set of measurements of the object or workpiece and then reconstruct it as a 3D model. Typical ways to measure include scanning technologies such as 3D laser scanners, touch scanners, coordinate measuring machines, light digitizers, and industrial computer tomography. Typically, the process yields a cloud of data points, which have to be fitted in order to recover the topological information of the original model.

When data come from accurate sources (i.e., with minimal measurement errors) and smooth shapes, curve reconstruction can be done through well-known curve interpolation techniques such as spline and Bézier curve interpolation. This approach is characterized by imposing the parametric curve to pass through all data points. However, most reverse engineering applications retrieve data from scanning, digitizing [1], or some mechanical device, such as pressure machines in the shoemaking industry [2], thus being subjected to measurement errors. In case of high-intensity noise, interpolation methods tend to fail since they force the curve to pass through all points, including the noisy outliers. This undesirable situation can be overcome by *data approximation*, where the fitting curve is only required to be sufficiently *close* to the original data according to some prescribed metrics. Owing to its ability to cope with noise inherent to data in real-world settings, in this paper we will use a curve approximation method.

Several families of functions can be used for this curve approximation task. Among them, the free-form parametric functions (such as Bézier and B-splines) are the most common in CAD/CAM (Computer-Aided Design/Manufacturing), computer graphics and animation, virtual reality, and other fields. In particular, Bézier curves have been intensively used in automotive industry for decades. Mathematically, they are given by a linear combination of basis functions called the Bernstein polynomials with vector coefficients called poles or control points. The curve follows approximately the shape of its control polygon (the collection of segments joining the poles), and, hence, it reacts to the movement of its poles by following a push-pull effect. This nice feature was fundamental for the popularization of free-form curves for interactive design. Although nowadays Bézier curves have been mostly deprecated in that field, being overtaken by the more powerful B-splines, they are still in use in many other areas. For instance, True Type fonts use composite curves comprised of quadratic Bézier curves. Similarly, all Postscript font outlines are defined in terms of cubic and linear Bézier curves. Other recent applications include, for instance, robot path planning [3] and the determination of the airfoil geometry from a given C_p-distribution [4].

A recent paper by the authors addressed the problem of curve approximation with polynomial Bézier curves [5]. The method was based on a hybrid scheme combining a popular single-particle metaheuristic called simulated annealing for global search and a local search optimizer for further refinement of the global solution. This hybrid scheme was applied to compute all relevant parameters of the approximating function with satisfactory results.

Although the polynomial representation is very easy to deal with (and, therefore, very convenient for many purposes) it is still limited in some ways. For example, it is well known that the polynomial-based schemes cannot represent accurately some important shapes such as the conics (e.g., circles, ellipses, and hyperbolas). A way to overcome this drawback is to consider the polynomial curve in homogeneous coordinates, leading to rational functions (i.e., functions that are quotients of two polynomials). In this sense, it is important to remark that conics can be canonically described as rational functions. Many other complicated shapes can be dealt with more efficiently if the (more limited) polynomial scheme is extended to a rational one. It is not a trivial task, however, since the rational scheme includes additional degrees of freedom that have also to be computed. As a consequence, our previous method is no longer valid and must be substantially improved. This is actually the goal of the present contribution.

(2) Aims and Structure of This Paper. In this paper we focus on obtaining an accurate fitting of a given set of noisy data points by using a rational Bézier curve. To this aim, we consider a modification of the standard simulated annealing (SA), a popular probabilistic method for global optimization based on thermodynamical principles. In our approach, the basic SA algorithm is further improved by considering two larger global optimization schemas. They are applied to compute all relevant parameters of our curve fitting problem. A typical shortcoming in this regard is that the accuracy of the model increases as the number of poles increases, eventually leading to overfitting. To prevent this undesirable effect, we also aim to seek a suitable trade-off between data fidelity and model complexity. To attain this goal we apply the Bayesian Information Criterion (BIC), a model-selection technique widely used in the context of information sciences. The basic idea of BIC is to combine a penalty term with the fitness function. This penalty term increases with the number of free parameters of the model, thus penalizing any unnecessarily large complexity of the model.

The structure of this paper is as follows: Section 2 summarizes briefly the previous work in the field. Some basic mathematical concepts and definitions and the curve fitting problem from noisy data points with rational Bézier curves are described in Section 3. Then, the simulated annealing algorithm is described in detail in Section 4. Section 5 describes all steps of our proposed methodology along with the two schemas used in this paper. The experimental results of the application of our method to three illustrative examples are reported in Section 6. The section also discusses the robustness of the method in the presence of noise. The paper concludes with the main conclusions of this work and some future lines of research.

2. Previous Work

The problem of data approximation with free-form curves has been a classical subject of research for many years. The first research works addressing this issue were published in the 60s and 70s and were mostly based on numerical procedures [6–8]. The classical methods of this period select the free parameters by inferring some geometric properties from data in order to meet specific constraints [1, 9, 10]. A major problem in this regard is to perform data parameterization, that is, to determine suitable parameter values associated with the data points. Typical choices are the uniform parameterization and the arc-length parameterization. Unfortunately, classical numerical methods are not able to obtain optimal solutions in the general case. Subsequent attempts to address

this problem during the next decades provided solutions for several particular cases but were not successful in solving the general problem [9, 11]. More recent methods apply alternative approaches such as line use error bounds [12], curvature-based squared distance minimization [13], or dominant points [14]. Although they perform well, they are still limited to particular cases and/or require strong constraints in terms of differentiability or noiseless data, assumptions that are not reasonable for real-world instances.

Mathematically speaking, curve fitting is usually addressed as a continuous optimization problem [6, 8, 9]. However, traditional mathematical optimization techniques have had little success in solving this problem. In the last few years, the field has taken a new impetus with the application of several powerful artificial intelligence and soft computing techniques. Some of these methods are based on neural networks, such as the standard neural networks [15], Bernstein basis function networks [16], and Kohonen's SOM (Self-Organizing Maps) nets [17]. In some other cases, the neural approach is combined with partial differential equations [18]. This scheme has been extended to the more general functional networks in [19, 20]. More recently, powerful metaheuristic techniques have been applied to this problem, including swarm intelligence [21], artificial immune systems [22], genetic algorithms [23, 24], physics-inspired thermodynamics [5], cuckoo search [25], support vector machines [26], firefly algorithm [27], and hybrid techniques [28, 29]. These methods obtain remarkable results for polynomial curves, but more complicate shapes (e.g., conics) are still not properly fitted. This is a good evidence of the interest and difficulty of this data fitting problem, which is described in detail in the next section.

3. Description of the Problem

3.1. Basic Concepts and Definitions. In this section we assume that the reader is familiar with the concept of parametric curves. Bézier curves are a particular case of free-form parametric curves, introduced in the 60s for interactive design in the automotive industry. A *nonrational* (i.e., polynomial) *parametric Bézier curve of degree n* in $\mathbb{R}^d$ is given by

$$\mathbf{B}(t) = \sum_{j=0}^{n} \mathbf{b}_j B_j^n(t) \quad \text{with } t \in [0,1], \tag{1}$$

where $\{\mathbf{b}_j\}_{j=0}^n \subset \mathbb{R}^d$ are vector coefficients (usually called *poles* or *control points*) and $B_j^n(t)$ are the *Bernstein polynomials of index j and degree n*, given by

$$B_j^n(t) = \binom{n}{j} t^j (1-t)^{n-j} \quad \text{with } \binom{n}{j} = \frac{n!}{j!(n-j)!},\ 0! = 1. \tag{2}$$

Note that in this paper vectors are denoted in bold. The polynomial representation is not powerful enough to represent a variety of shapes, particularly the conics (e.g., circles, ellipses, and hyperbolas). One way to overcome this limitation is to use homogeneous coordinates (see [10, 30] for details). The basic idea is to consider the projection of the standard polynomial Bézier curve in $\mathbb{R}^{d+1}$, with new poles $\mathbf{b}_j^h$. The resulting curve in $\mathbb{R}^d$ is called a rational Bézier curve. Mathematically, this curve can be described as a quotient of two polynomials, or as a linear combination of rational basis functions:

$$\mathbf{C}(t) = \sum_{j=0}^{n} \mathbf{b}_j R_j^n(t) \quad \text{with } t \in [0,1], \tag{3}$$

where the rational basis functions are defined by

$$R_j^n(t) = \frac{w_j B_j^n(t)}{\sum_{k=0}^{n} w_k B_k^n(t)}, \tag{4}$$

where n is the curve degree and w_j is the last coordinate of the homogeneous control point $\mathbf{b}_j^h$. This set of new scalar parameters $\{w_j\}_{j=0}^n$, called *weights*, provides us with additional degrees of freedom for better shape approximation. They also increase the model complexity, however, as we introduce a new set of parameters that have to be computed as well.

3.2. The Fitting Problem. Let now $\{\mathbf{Q}_i\}_{i=1}^m$ be a set of data points in $\mathbb{R}^d$. The problem consists of obtaining the rational Bézier curve, $\mathbf{C}(t)$, of a certain degree n providing the best least-squares fitting of the data points. This leads to a minimization problem of the least-squares error Θ defined as the weighted sum of squares of the residuals:

$$\Theta = \sum_{i=1}^{m} \mu_i \left(\mathbf{Q}_i - \frac{\sum_{j=0}^{n} w_j \mathbf{b}_j B_j^n(t_i)}{\sum_{j=0}^{n} w_j B_j^n(t_i)} \right)^2, \tag{5}$$

where $\{\mu_i\}_{i=1}^m$ are scalar numbers used in situations when it may be reasonable to assume that sampled data should not be treated equally. In order to reflect faithfully the most common situation in real-world problems, in this paper we will assume that no information about the problem is available beyond the data points. This means that all data points must be treated equally; that is, $\mu_i = 1$, for all i. Note, however, that our method is independent on the values of μ_i. To represent the geometrical distribution of the data we need to associate a parameter t_i for each input point $\mathbf{Q}_i$. Therefore, our goal is to obtain the three sets of parameters $\{t_i\}_{i=1}^m$, $\{w_j\}_{j=0}^n$, and $\{\mathbf{b}_j\}_{j=0}^n$. It is obvious that since each blending function in (2) and (4) is nonlinear in t, system (5) is also nonlinear. As a consequence, we have to deal with a multivariate continuous nonlinear minimization problem. In this paper we solve this problem by applying two different schemas of the simulated annealing optimization method, which is described in the next section.

4. Simulated Annealing

One of the major trends in global optimization during the last few decades has been to build algorithms trying to mimic certain efficient optimization patterns observed in natural processes. As a result, a series of very powerful

nature-inspired optimization algorithms (e.g., particle swarm optimization, genetic algorithms, or ant colony optimization) have been devised. Very often, they provide better solutions than previous traditional mathematical algorithms to several hard optimization problems. Although they are very diverse, all of them share two common features: to be inspired by real-world observation and to search for solutions in a stochastic way. Most of them are also *derivative-free*, meaning that they can be applied to problems where it is not possible to compute the derivatives of the objective function (or they are very expensive computationally). In this paper we apply simulated annealing, one of the most popular nature-inspired optimization algorithms.

4.1. Background. The *simulated annealing* (SA) is a nature-inspired metaheuristic optimization algorithm introduced by Kirkpatrick et al. in the context of combinatorial optimization [31]. The algorithm is inspired in the annealing process in metallurgy, where metals are heated at very high temperatures and then slowly cooled down to reach a state of lower energy. During the process, atoms tend to move to configurations that minimize the system energy even if during such migration certain configurations rise the system overall energy (when it stabilizes for a fixed temperature, we call it *thermal equilibrium*). Such moves are more prominent at the beginning of the process than at the end, when the particles loose thermal mobility in order to polish the system inner structure to finally produce a better metal. As a result, the metals become stronger and with better properties, specially if the process is conducted several consecutive times (a process called *reannealing*).

The original SA algorithm is an advanced interpretation of the *Metropolis-Hastings algorithm* [32] to generate sample states of a thermodynamic system, showing the deep connections between statistical mechanics and combinatorial optimization. Given an initial (usually random) *state* in the solution domain, the algorithm iteratively perturbs it. Whenever a better solution is found, the change is always accepted; otherwise, it is accepted only with a certain probability. This probability is higher at the beginning (mimicking what happens in the thermodynamic process at high temperatures) than at the end. In other words, this idea of slow cooling is translated as a slow decrease of the probability of accepting such worse solutions. So essentially the system evolves from a free exploration of the search space at initial stages to a stochastic hill-climbing at latter stages.

Since its publication the algorithm has received a lot of attention from the scientific community, with many real-world applications in the most diverse fields, ranging from the classical NP-hard combinatorial *travelling salesman* problem [33] to the minimization of highly multimodal real-valued functions [34]; see [35] for an in-depth review of several simulated annealing applications.

4.2. The Algorithm. The simulated annealing algorithm was originally designed to compute a good approximation of the global optimum of a fitness function (usually called the *system energy*) within a problem domain $\mathscr{D}$, assumed to be continuous in this paper. Each point $\mathbf{x} \in \mathscr{D}$ is a *state* of some physical system. Given an initial (usually random) state $\mathbf{x}_0$, the goal of the SA is to obtain the state with the minimum energy (associated with the best solution of the optimization problem). In our case, we have a real-valued function $f : \mathscr{D} \subseteq \mathbb{R}^d \to \mathbb{R}$ that we want to minimize. The algorithm performs an iterative process; at each iteration step, a new state $\mathbf{x}_{\text{new}}$ is generated from the current one, $\mathbf{x}_{\text{old}}$, through a *neighborhood function*, denoted by $\mathfrak{N} : \mathscr{D} \to \mathscr{D}$, that is, $\mathbf{x}_{\text{new}} = \mathfrak{N}(\mathbf{x}_{\text{old}})$. Let now $f_{\text{old}} \equiv f(\mathbf{x}_{\text{old}})$, $f_{\text{new}} \equiv f(\mathbf{x}_{\text{new}})$ be their associated energies, respectively. The algorithm probabilistically decides between moving the system to the new state $\mathbf{x}_{\text{new}}$ or staying in current state $\mathbf{x}_{\text{old}}$. This new state is chosen with a probability function $\mathfrak{P} : \mathscr{D} \times \mathscr{D} \to [0, 1]$, called the *acceptance function*, which depends on two factors as follows:

(i) on one hand, on the difference between their energy values, $\Delta = f_{\text{old}} - f_{\text{new}}$,

(ii) on the other hand, on a global parameter called *temperature*, denoted by T, which varies according to a strictly decreasing function $\mathfrak{T} : \mathbb{R}^+ \to \mathbb{R}^+$ called the *cooling function*.

The probability function is not arbitrary, but must comply with certain requirements. One of them is that $\mathfrak{P} > 0$ if $\Delta < 0$, meaning that the system may move to the new state even if it is worse (has a higher energy) than the current one. The primary goal of this condition is to prevent *stagnation* (when the system gets trapped in the neighborhood of local optima, leading to premature convergence). Under the same condition $\Delta < 0$, we also impose that $\mathfrak{P} \to 0$ as $T \to 0$, while $\mathfrak{P} \to \eta > 0$ otherwise. Basically, these conditions state that, for sufficiently small values of T, the system will increasingly promote "downhill" changes (i.e., changes leading to lower energy values) and avoid "uphill" changes. In other words, the lower the temperature is, the easier it is to reject a worse solution. In fact, in the particular case $T = 0$, the procedure will only allow downhill moves, meaning that the algorithm reduces to a greedy search algorithm. Another desirable feature is to promote small uphill moves over large ones. This effect can be obtained by modulating the probability as a function of the parameter Δ so that $\mathfrak{P}$ decreases as Δ increases. Under these conditions, the temperature T becomes a critical parameter in describing the evolution of the system, as its value determines the sensitivity of the system to energy variations.

In this paper, we consider the classical acceptance function derived from the Metropolis-Hastings sampling algorithm, first introduced by Metropolis et al. in [32] as a Monte-Carlo method to simulate the creation of new states in a thermodynamic system and given by

$$\mathfrak{P}(\Delta) = \begin{cases} 1 & \text{if } \Delta \leq 0 \\ \exp\left(-\dfrac{\Delta}{T}\right) & \text{otherwise,} \end{cases} \tag{6}$$

where T represents the system temperature at the iteration where $\mathbf{x}_{\text{new}}$ has been generated. With this function, a better candidate solution is always accepted, but even worse solutions have a chance to be accepted. It is clear that

Require: (Initial Parameters)
(1) Initial Temperature T_0
(2) Schedule Criteria $\mathcal{S}_c$
(3) Stopping Criteria SC
(4) Neighbourhood function $\mathfrak{N}$: $\mathscr{D} \to \mathscr{D}$
(5) System Energy f: $\mathscr{D} \subseteq \mathbb{R}^d \to \mathbb{R}$
(6) Cooling Schedule $\mathfrak{T}$: $\mathbb{R}^+ \to \mathbb{R}^+$ (strictly decreasing function)
(7) $T \leftarrow T_0$ ▷ Temperature initialization
(8) Randomly select start state $\mathbf{x}_0 \in \mathscr{D}$
(9) $\mathbf{x}_{\text{old}} \leftarrow \mathbf{x}_0$
(10) **repeat** ▷ External loop: stopping criteria
(11) **repeat** ▷ Internal loop: thermal equilibrium
(12) $\mathbf{x}_{\text{new}} \leftarrow \mathfrak{N}(\mathbf{x}_{\text{old}})$ ▷ New state generated
(13) $f_{\text{old}} \leftarrow f(\mathbf{x}_{\text{old}})$ ▷ Energy evaluation of current state
(14) $f_{\text{new}} \leftarrow f(\mathbf{x}_{\text{new}})$ ▷ Energy evaluation of new state
(15) $\Delta \leftarrow f_{\text{old}} - f_{\text{new}}$
(16) **if** $\Delta < 0$ **then** ▷ Acceptance criterion
(17) $\mathbf{x}_{\text{old}} \leftarrow \mathbf{x}_{\text{new}}$
(18) **else**
(19) Randomly compute $u \in \text{Rand}(0, 1)$ ▷ Metropolis procedure
(20) **if** $u \leq \exp(-\Delta/T)$ **then** ▷ Equation (6)
(21) $\mathbf{x}_{\text{old}} \leftarrow \mathbf{x}_{\text{new}}$
(22) **end if**
(23) **end if**
(24) **until** $\mathcal{S}_c$ == true
(25) $T \leftarrow \mathfrak{T}(T)$ ▷ Temperature update
(26) **until** SC == true
(27) **return** $\mathbf{x}_{\text{new}}$ ▷ Best final solution

Algorithm 1: Simulated annealing.

(6) meets all required conditions for a proper *acceptance function* indicated above. It also provides an adequate trade-off between exploration and exploitation: at initial stages (i.e., higher temperatures) the algorithm explores the search space while at latter stages it resembles a hill-climbing algorithm, with the difference that now there is always the possibility to accept a worse state.

The corresponding pseudocode of the simulated annealing algorithm used in this paper is shown in Algorithm 1. A typical execution begins with a randomly chosen state $(\mathbf{x}_0, f_0)$ and an initial (very high) temperature T_0. The algorithm generates new states, according to $\mathfrak{N}$, at each iteration and probabilistically decides whether or not the new state is accepted according to the probability $\mathfrak{P}$. According to the pseudocode, the algorithm can be summarized as the interaction between two cycles: the external one controls the temperature updating and the inner one performs the Metropolis procedure for a given outer iteration. The temperature T is updated through the cooling function $\mathfrak{T}$ only when the thermal equilibrium is reached. This workflow is repeated until a *stopping criterion* is met. Classical stopping criteria are as follows: the system reaches a state good enough for the specific application, the method reaches a prescribed number of iterations, or the solution does not improve after a prescribed number of consecutive iterations. Note that both the cooling schedule criterion (line 24) and the stopping criterion (line 26) can be defined in many different ways, depending on the problem at hand. In this sense, these criteria can be either variables or functions or even rules. In our implementation it is assumed that they are Boolean functions that return true or false depending on whether or not the given conditions for each particular problem are met. The specific conditions used for our problem are explained in Section 5.

5. The Proposed Method

5.1. Overview of the Method. As discussed above, our problem consists of reconstructing the underlying shape of a given set of noisy data points by using a rational Bézier curve. This implies solving a nonlinear least-squares minimization problem while simultaneously minimizing the required number of free parameters. Solving this problem requires computing four different sets of unknowns: data parameters, poles, weights (represented in this section by vectors **p**, **w**, and **b**, resp.), and the curve degree, n. Our approach to tackle this issue is a hybrid strategy combining classical methods (least-squares minimization), modern stochastic methods (simulated annealing), and information science metrics (Bayesian Information Criterion (BIC)).

Before explaining how our method works, let us introduce the following notation: from now on, we will use the subindex $(\cdot)_w$ when searching for the curve weights, $(\cdot)_p$ when searching for the curve parameters, and $(\cdot)_{w,p}$ when

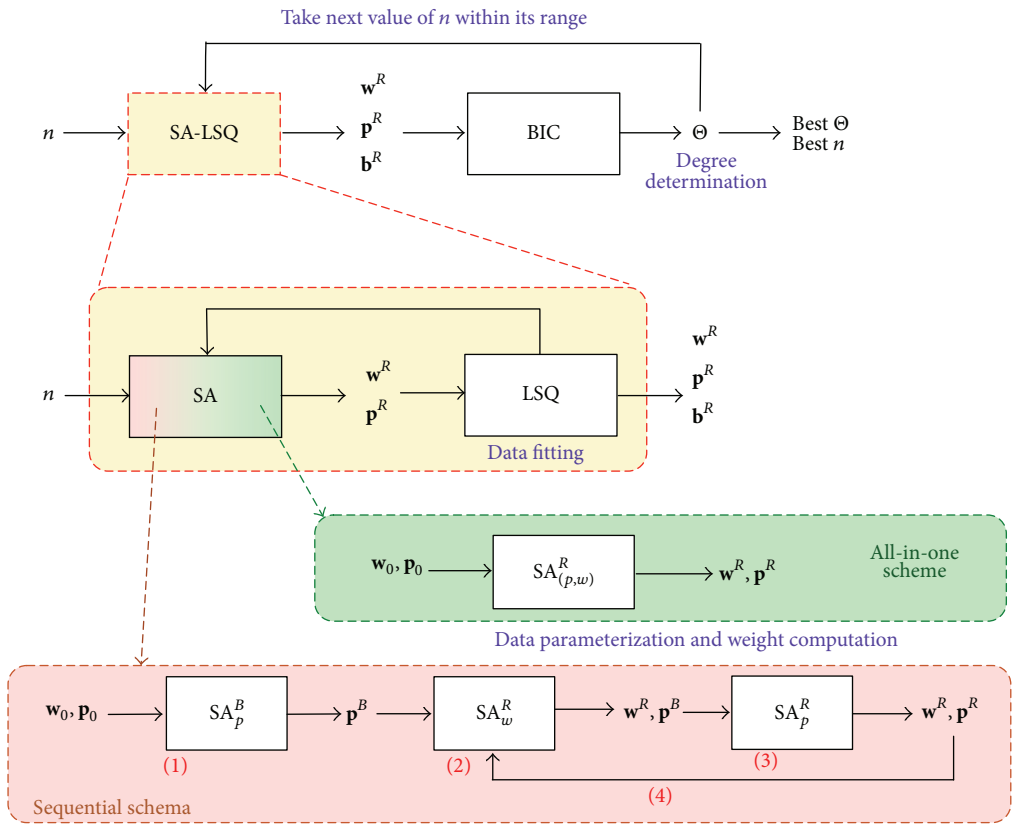

FIGURE 1: Workflow of the proposed method described in three layers (from upper to lower layer): the general workflow; decomposition of the SA-LSQ procedure; the two different SA schemas introduced in this paper: all-in-one schema (top) and sequential schema (bottom).

searching for both parameters and weights. Thus, SA_p stands for simulated annealing applied to parameter search, $f_{w,p}$ means the objective function with domain of definition $\mathbf{y} = (\mathbf{w}, \mathbf{p}) \in \mathscr{W} \times \mathscr{D} \subset \mathbb{R}^{n+1} \times [0,1]^m$, and so on. The superindex $(\cdot)^B$ denotes the search for a nonrational Bézier curve and $(\cdot)^R$ stands for a rational one. Hereafter, consider $\mathscr{W} = (0,100]^{n+1}$ and $\mathscr{D} = [0,1]^m$. Without loss of generality, we can also assume that $w_0 = w_n = 1$, $t_1 = 0$, and $t_m = 1$.

Figure 1 shows the main steps of our method. Basically, it consists of four major tasks: data parametrization, weight computation, data fitting, and degree determination. Upper part of this figure summarizes the method: we initially set a range for the curve degree n; then, for each value of this parameter n within that range, we apply a combination of simulated annealing and least-squares optimization (box SA-LSQ) to perform the first three tasks and compute the data parameters, weights, and poles of the best fitting rational Bézier curve for this value of n. Then, the BIC value of the resulting curve (corresponding to the last task, degree determination) is obtained (box BIC). At its turn, the SA-LSQ can be decomposed into two steps (middle layer of Figure 1): SA performs data parameterization and weight computation (see Section 5.2 for details), while LSQ is used to compute the poles (see Section 5.3). This combination of SA and LSQ is repeated iteratively until no further improvement is reached. In this paper, we introduce two different SA schemas, shown graphically in the lower layer of Figure 1: the sequential schema and the all-in-one schema. They are explained in detail in the next section.

5.2. Data Parameterization and Weight Computation. In this step we perform two different (but intertwined) tasks: data parameterization and weight computation. The former consists of obtaining a discrete association between the set of parameters $\{t_i\}_{i=1}^m$ and the noisy data points $\{\mathbf{Q}_i\}_{i=1}^m$ to be fitted, while the latter computes the weights. Both tasks are performed simultaneously by using the simulated annealing approach described in the previous section. The input for the SA method is given by the following:

(i) the curve degree, n,

(ii) initial random parameter vectors $\mathbf{p}_0$, $\mathbf{w}_0$, and $\mathbf{b}_0$,

(iii) the energy function, given by (5),

(iv) a neighborhood function (described in Section 5.2.1),

(v) a cooling schedule (described in Section 5.2.2),

(vi) a stopping criterion (described in Section 5.2.3).

In this work, two different simulate annealing schemas are considered: the *sequential schema* and the *all-in-one schema*. Basically, the former calculates the different sets of unknowns of our optimization problem in a sequential way (i.e., only some parameters are initially computed and subsequently

used to compute the remaining parameters), while the latter computes all unknowns at once. Let us analyze them in detail.

Using the notation introduced above, the sequential schema (SEQ) begins by finding the best nonrational Bézier fitting curve through SA_p^B, then it performs the following sequence iteratively:

$$\mathrm{SA}_w^R \longrightarrow \mathrm{SA}_p^R. \tag{7}$$

until there is no further improvement in the final solution of either SA procedure in comparison with the previous one. Each routine takes the preceding solution as an input parameter. The schema can be summarized as follows (see also the lower layer of Figure 1, where the numbers in red indicate the different steps of the algorithm):

(1) Apply the SA_p^B with a random initial guess $\mathbf{p}_0 \in \mathscr{D}$ to find a non-rational Bézier curve that fits the data better. Let $\mathbf{p}^B$ be the resulting solution.

(2) Search for a rational Bézier curve through SA_w^R with a random initial guess $\mathbf{w}_0 \in \mathscr{W}$ and fixed parameters $(\mathbf{p}^B)$. Let $(\mathbf{w}^R, \mathbf{p}^B)$ be the resulting solution.

(3) Apply the SA_p^R optimization with $\mathbf{p}^B$ as the initial guess and fixed weights $\mathbf{w}^R$. Let $(\mathbf{w}^R, \mathbf{p}^R)$ be the resulting solution.

(4) Repeat (2)-(3) iteratively until there is no improvement in the resulting solution.

In general, the energy function for the simulated annealing procedures SA^R in steps (2) and (3) above is that in (5). However, a different energy function is required for the nonrational case in step (1), SA^B, given by

$$\Theta = \sum_{i=1}^{m} \mu_i \left(\mathbf{Q}_i - \sum_{j=0}^{n} \mathbf{b}_j B_j^n (t_i) \right)^2 \tag{8}$$

that corresponds to the case $w_j = 1$ for all j.

The *all-in-one schema* (AIO) corresponds to the minimization of functional

$$f_{w,p} : \mathscr{W} \times \mathscr{D} \longrightarrow \mathbb{R}, \tag{9}$$

where for each state vector $(\mathbf{w}, \mathbf{p})$ we compute $f(\mathbf{w}, \mathbf{p})$ as the least-squares solution of (5) through $\mathrm{SA}_{(p,w)}^R$.

5.2.1. Neighborhood Function. The *neighborhood function* is one of the key components of the SA algorithm. Furthermore, its role becomes even more important for multimodal optimization problems, where the objective function is of the *many local peaks surrounded by deep valleys* type. This is exactly what happens in this paper. There are several alternatives for the neighborhood function. A very popular choice is the *fast neighborhood function*, which builds the new solution by modifying the previous one in steps proportional to the system temperature:

$$\mathbf{x}_{\text{new}} = \mathbf{x}_{\text{old}} + T \cdot \mathbf{v}, \tag{10}$$

where $\mathbf{v} \in \mathscr{D}$ is a random vector holding $\|\mathbf{v}\|_2 = 1$. This is one of the tested functions in [5] but it required another support function (a local search method) in order to exploit the neighborhood of a solution. In this paper we remove the supporting local method by maintaining two sets of controlling parameters: firstly, the global *real* temperature following the classical SA temperature. Then, the algorithm builds a second set of *virtual* temperatures (one for each spatial dimension) that are restarted at the beginning of each inner cycle. The resulting equation becomes

$$\mathbf{x}_{\text{new}} = \mathbf{x}_{\text{old}} + \mathbf{T} \odot \mathbf{v}, \tag{11}$$

where $\odot$ represents the Hadamard product of two vectors and $\mathbf{T}$ has all its components set to the current system temperature at the beginning of each outer cycle. Thus, whenever $\mathbf{x}_{\text{new}}$ is accepted through the *acceptance criterion*, we compute the absolute difference between the *old* and *new* solutions. For each component that behaves better than the previous one in the comparison, we update the corresponding component on $\mathbf{T}$ according to the cooling schedule. The proposed method is based on the *Adaptive Simulated Annealing* (ASA) algorithm but there is one key difference: our method does not require computing the gradient of the function. A further analysis about the performance of this neighborhood function for the third example of our benchmark is described in Section 6.3. The reader is kindly referred to that section for further details on this issue.

5.2.2. Cooling Schedule. By the *cooling schedule* we refer to a triplet $(T_0, T, \mathfrak{T})$ accounting for the selection of the initial temperature, the temperature parameter, and the cooling function $(\mathfrak{T})$, respectively, along with the *thermal equilibrium criterion*. The cooling schedule governs the pace at which the temperature is updated during the execution of the SA. Therefore, its choice is of primary importance for the good performance of the algorithm. Because of this reason, numerous cooling schedules have been proposed over the years, such as the linear schedule, the logarithmic schedule, and many others. A very common one is the *power schedule*, given by $T_{\text{new}} = \alpha^k T_0$, where typically $0.9 < \alpha < 0.99$ and $k > 0$ is a scalar parameter. Another popular schedule is the *Boltzmann schedule*, given by $T_{\text{new}} = T_0/\log(k)$. Finally, we also consider the *fast schedule*, governed by the law $T_{\text{new}} = T_0/k$, which provided a good balance among simplicity, speed, and good performance for other data fitting problems [5]. In this paper we considered initially the power, Boltzmann, and fast schedules. However, our computational experiments showed that the fast schedule provides more visually appealing results and runs faster than the other two alternatives. Since these are two particularly valuable features in the context of data fitting, we eventually selected the fast schedule as the best choice for this work.

The cooling schedule needs to accomplish two additional goals. On one hand, the starting temperature needs to be sufficiently high in order to let the algorithm *freely* explore the search space $\mathscr{D}$ but not so much that the system behaves like a random search for a large number of iterations. On the other hand, the cooling function must perform a slow reduction of the system temperature to prevent premature convergence.

These goals are achieved by letting the acceptance function to initially accept worse states frequently, which in turn requires high temperatures. There is not known general way to select a good starting temperature for all problems. A common practice consists of selecting a very high temperature to rapidly cool the system after a certain proportion of worse solutions are accepted (this is called the *acceptance ratio*); then, begin to *slowly* cool the system, thus beginning with the proper SA algorithm. Another common technique consists of spending most of the computation time in running fast sample runs of the entire algorithm for different starting temperatures in order to determine the *ideal* temperature: one that promotes better solutions but still manages to explore the space avoiding local minima. This last practice was proposed in [36] as a way to avoid wasting computation time when running the SA. In [35] there is a brief, although insightful, overview of the different approaches on how to select the initial temperature.

In order to select the initial temperature we have discarded the time-consuming restarting strategy used in our previous work in favor of the method presented in [37]. It consists of setting an acceptance ratio, χ_0, and determining, through an iterative algorithm, a *compatible* starting temperature. Among the several possible ways to define χ_0, in this work we choose the classical one of the quotient between the number of bad transitions accepted and the attempted ones, already proposed in [38]. According to this definition, we set $\chi_0 = 0.8$, a typical value in previous literature in the field.

5.2.3. Stopping Criteria. Similar to the choice of the initial temperature, there is not a general set of stopping conditions suitable for all problems. There are, however, two common practices: the first one is to set a maximum number of iterations; the second one is to stop the execution of the algorithm when the system is *frozen*, that is, when no new solutions (either better or worse states) are accepted for a predefined number of iterations. Since the former can waste a lot of computation time with no further improvements on the solution, in practice the stopping criterion is often a combination of the two. In our implementation, the algorithm stops whenever one of the following conditions is met: either $n_{f\text{eval}} > 1000 \cdot \xi$ or $\sharp[\partial(\Delta_{\text{global}})] = 10 \cdot \xi$, where $n_{f\text{eval}}$ denotes the total number of evaluations of the energy function, ξ denotes the number of free parameters of the system, and $\partial(\cdot)$ denotes the lack of changes of the mean variation of the energy function.

5.2.4. Further Improvements. In addition to the previous SA components, three other important modifications of the original SA algorithm have been included in our approach:

(i) We improve the memory capacities of the method through *elitism*: the best state from the current iteration is encoded as a vector $(\mathbf{x}^M_{\text{best}}, f^M_{\text{best}})$ and stored in a temporal buffer. Obviously, this "best so far" solution is updated whenever a better solution is achieved during SA execution. We remark however that this $(\mathbf{x}^M_{\text{best}}, f^M_{\text{best}})$ vector is not used to drive the SA execution; instead, it is only used as a memory effect, with the role to (possibly) improve the convergence rate with respect to the standard (nonelitist) version of SA.

(ii) We add a new operator, related to the domain of the problem, to work in combination with the neighborhood function. This extra functionality checks whether a new generated solution goes outside the search domain of the problem and sends it back into the search space whenever it goes away. To this purpose, we apply the classical *cast back* operator, a widely accepted routine in numerical methods. Suppose that the SA returns a new solution $\mathbf{x}_{\text{new}}$ outside the problem domain $\mathscr{D}$, obtained from a previous solution $\mathbf{x}_{\text{old}}$ within $\mathscr{D}$. The cast back procedure replaces $\mathbf{x}_{\text{new}}$ by a new value $\mathbf{x}_{\text{cb}}$ given by the convex combination: $\mathbf{x}_{\text{cb}} = \alpha\text{Proj}(\mathbf{x}_{\text{new}}) + (1 - \alpha)\mathbf{x}_{\text{old}}$, where $\text{Proj}(\cdot)$ is the operator that projects any point outside the domain onto its closest point on the boundary of $\mathscr{D}$ and α is a uniform random number in the interval $(0, 1)$. This procedure returns a new point $\mathbf{x}_{\text{cb}}$ that is well within the search domain while simultaneously ensuring that the probability of the boundary is not increased by this operator.

(iii) Finally we improve the cooling schedule with extra conditions for the thermal equilibrium. In our implementation, the inner cycle stops if the value of χ_0 is reached after ξ iterations.

These new features improve the performance of our approach significantly in terms of computational time and quality of results.

5.3. Data Fitting. With the parameterization and weights calculated in previous steps, we compute the curve coefficients $\{\mathbf{b}_j\}_{j=0}^n$. Using (3), (5) can be rewritten as

$$\begin{bmatrix} \mathbf{R}_0^T\mathbf{R}_0 & \cdots & \mathbf{R}_n^T\mathbf{R}_0 \\ \vdots & \vdots & \vdots \\ \mathbf{R}_0^T\mathbf{R}_n & \cdots & \mathbf{R}_n^T\mathbf{R}_n \end{bmatrix} \begin{bmatrix} \mathbf{b}_0 \\ \vdots \\ \mathbf{b}_n \end{bmatrix} = \begin{bmatrix} \mathbf{Q}\mathbf{R}_0 \\ \vdots \\ \mathbf{Q}\mathbf{R}_n \end{bmatrix}, \tag{12}$$

where $\mathbf{R}_j = (R_j^n(t_1), \ldots, R_j^n(t_m))^T$ represents the column vector of rational basis functions given by (4) at the best parameter values, $\mathbf{Q} = (\mathbf{Q}_1, \ldots, \mathbf{Q}_m)$ is a row vector, and $(\cdot)^T$ represents the transpose of a vector or matrix. System (12) is overdetermined, meaning that no analytical solution can be obtained. Instead, we solved it numerically through least-squares minimization. If $\mathbf{R}^+$ denote the generalized inverse (also known as Moore-Penrose pseudoinverse) of $\mathbf{R} = (\mathbf{R}_i)^T$, $\mathbf{P} = \mathbf{R}^+\mathbf{Q}$ is the least-squares solution of this data fitting problem. From a computational point of view, it can be obtained by either LU decomposition or singular value decomposition (SVD). In this work, we have used the SVD solver from the popular numerical program *Matlab*.

5.4. Degree Determination. The three previous steps assumed a given degree for the rational fitting curve. However, the optimal degree is problem-dependent, and therefore we need

a method to compute it. This is not an easy task. It is well known that increasing the number of poles increases the accuracy of the model, because we have more degrees of freedom to adjust the data. However, this process can eventually lead to overfitting. In order to prevent this undesirable effect, some kind of trade-off between these two competing factors (the accuracy of the fitting and the complexity of the model) is required. To address this issue, in this paper we compute the Bayesian Information Criterion (BIC) for the resulting model. The BIC is an information criterion providing a good compromise between data fidelity and model complexity [39]. This is done by introducing a penalty term for complex models into the target function:

$$\mathrm{BIC} = \zeta \log(\Theta) + \xi \log(\zeta), \tag{13}$$

where Θ, ζ, and ξ refer to the energy function, number of sampled points, and the number of free parameters, respectively. Note that, fixing ζ and ξ, the BIC behaves like the error function and if we fix a value of Θ and ζ, the criterion penalizes those models with a higher number of parameters. Therefore, BIC provides us with a powerful procedure to compute the optimal value for the degree of the fitting curve. It always corresponds to the parameter value n with the *smallest value* for the BIC.

6. Experimental Results

In this section, we analyze the performance of our method by applying it to a benchmark of three illustrative examples of 2D and 3D noisy data points. These examples have been carefully chosen so that they exhibit challenging features such as self-intersections or strong changes of slopes and curvatures. The first two examples correspond to real-world instances so that we can replicate the usual conditions of real-world applications, including the presence of noise of low-medium intensity. The last one is an academic example designed to analyze the effect of different levels of noise on our method. For each example, we report the results of three different schemas used for comparative purposes: nonrational, rational all-in-one, and rational sequential. Finally we analyze the robustness of the method in the presence of noise by comparing each schema against the same data perturbed with different levels of noise for the last example.

Regarding the implementation issues, all the experiments were run on a AMD-FX-4100 Quad-Core Processor at 3600 Mhz with 8 GB DDR3 RAM running *Linux 3.14.x LTS kernel* and *MATLAB 2012a*. For each dataset and schema, the experiment has been executed 26 times. Then, 20 executions are finally selected (after removing the three bast and three worst executions) to provide statistical evidence for the results presented and assert the experiment reproducibility.

Table 1 reports our results for each dataset and experiment. The following items are arranged (in columns): dataset examined, the *type* of curve reconstructed (NR: nonrational; *R*: rational), the *schema* executed for rational curves (AIO: all-in-one; SEQ: sequential), the total number of calls to the energy function in the best case (represented by $n_{f\mathrm{eval}}$), the best and average BIC, the number of poles for the best BIC (represented by n_{pol}), and the relative mean error for each component (x_{mean}, y_{mean}, and z_{mean}). When used, the acronym N.A. stands for not applicable. The *logo*, *shoe,* and *torus* names refer to the *scanned logo*, *shoe profile*, and *curve on a torus* datasets described below, respectively. The number after *torus* refers to the *signal-to-noise ratio* (SNR) applied. For the nonrational examples, a local search was performed in order to refine the SA solution. Our results show the good performance of the method even in highly noisy situations, which are the common case in real-world applications.

6.1. Example 1: A Scanned Logo. The first example corresponds to the shape of a digitally scanned logo. The dataset consists of a set of 190 noisy 2D data points, represented by black $\times$ symbols in Figure 2. The figure shows our experimental results for the three cases analyzed: nonrational case (top), the all-in-one rational case (middle), and the sequential rational case (bottom). The figures on the left show the reconstructed points, represented as red empty circles. On the right, the best fitting curve is displayed as a red solid line. This example has been chosen because it represents a common real-world scenario: a scanned figure with the typical noise introduced during the scanning process. In addition to the high-intensity noise (clearly visible in all instances of Figure 2), this shape is also challenging because it includes difficult geometric features, such as several self-intersections and strong changes of slope and curvature.

As the reader can see from the figures, the method is able to recover the general shape of the data points with good accuracy. This is a very remarkable result because the original data points are highly noisy. Best results correspond to the sequential rational schema, while the nonrational and the AIO rational schemas perform almost similarly in this case. This fact is clearly visible in the topmost loop of the figures in right column, best fitted through the sequential schema (bottom figure). These visual results are in good agreement with the numerical results reported in Table 1. The best and average BIC for the sequential schema are approximately -770 and -701, respectively, while they are both -691 for the nonrational schema and -661 and -658 for the AIO rational schema. Note also that the three methods obtain the same optimal number of poles $n_{\mathrm{pol}} = 13$. In other words, the good accuracy of our method is not at the expense of a very large number of free variables.

6.2. Example 2: Shoe Profile. The second example corresponds to a shoe profile obtained from a pressure-mechanical method without filtering, leading to a set of 400 three-dimensional noisy data points. Figure 3 shows our experimental results. The interpretation of this figure is similar to that of previous example and, hence, it is omitted here to avoid redundant information. Once again, the best fitting is obtained with the sequential rational schema, although in this case, the visual and numerical results are closer for the two rational schemas and significantly worse for the nonrational one. Note also that we obtained a similar parameter value, $n_{\mathrm{pol}} = 24$, for the number of poles in all cases.

6.3. Example 3: Curve on a Torus. Last instance in our benchmark corresponds to an academic example. It has been

Table 1: Experimental results on the three examples in our benchmark for the simulated annealing schemas discussed in this paper.

	Model		Statistical results						
Dataset	Type	Schema	n_{feval}	BIC (best)	BIC (avg)	n_{pol}	x_{mean}	y_{mean}	z_{mean}
logo	NR		10010	−691.913	−691.830	13	0.0005	0.0001	N.A.
logo	R	AIO	4208	−661.908	−658.804	13	0.0002	0.0003	N.A.
logo	R	SEQ	13031	−770.212	−701.338	13	0.0001	0.0001	N.A.
shoe	NR		16624	2915.200	3020.101	24	0.0191	0.0152	0.0076
shoe	R	AIO	6648	2098.859	2470.634	24	0.0012	0.0061	0.0053
shoe	R	SEQ	20202	1996.226	2289.712	24	0.0024	0.0061	0.0052
torus	NR		18546	−2204.821	−2301.567	20	$1.06e-05$	$1.23e-05$	$2.12e-05$
torus	R	AIO	1529	−6067.465	−6001.324	21	$6.42e-06$	$5.39e-06$	$4.32e-06$
torus	R	SEQ	5286	−7798.643	−7796.846	20	$1.48e-06$	$1.48e-06$	$1.35e-06$
torus10	NR		13276	4307.409	4390.492	17	0.0311	0.0342	0.0458
torus10	R	AIO	3768	4354.618	4414.523	19	0.0300	0.0304	0.0401
torus10	R	SEQ	23119	4341.604	4381.367	17	0.0312	0.0323	0.0402
torus20	NR		12027	3887.839	4991.823	20	0.0222	0.0201	0.0364
torus20	R	AIO	3778	3474.463	3479.345	20	0.0130	0.0137	0.0181
torus20	R	SEQ	45775	3446.804	3465.673	20	0.0128	0.0139	0.0182
torus30	NR		16174	3033.538	3561.901	20	0.0109	0.0094	0.0172
torus30	R	AIO	5546	3060.624	3066.298	20	0.0097	0.0089	0.0151
torus30	R	SEQ	48511	3021.939	3035.634	20	0.0098	0.0089	0.0148
torus40	NR		11015	6662.487	6700.738	40	0.0183	0.0144	0.0346
torus40	R	AIO	5111	2740.208	2788.422	20	0.0074	0.0073	0.0094
torus40	R	SEQ	53640	2708.680	2729.980	20	0.0074	0.0072	0.0096
torus50	NR		14101	2386.160	2501.900	25	0.0057	0.0059	0.0076
torus50	R	AIO	4205	2357.285	2408.811	21	0.0054	0.0053	0.0076
torus50	R	SEQ	45206	2344.050	2378.280	20	0.0055	0.0052	0.0076
torus60	NR		12052	3574.142	4713.190	20	0.0179	0.0121	0.0340
torus60	R	AIO	4012	2116.800	2145.321	20	0.0041	0.0045	0.0620
torus60	R	SEQ	45191	2051.304	2080.235	20	0.0031	0.0033	0.0038
torus70	NR		8860	3574.142	3866.889	21	0.0058	0.0057	0.0116
torus70	R	AIO	4205	2210.012	2284.590	21	0.0039	0.0040	0.0054
torus70	R	SEQ	45797	1963.317	1995.759	20	0.0038	0.0039	0.0056
torus80	NR		14100	3574.142	3934.296	25	0.0038	0.0044	0.0052
torus80	R	AIO	4002	1814.950	2008.400	21	0.0032	0.0036	0.0044
torus80	R	SEQ	13366	1803.356	1931.721	20	0.0032	0.0036	0.0043
torus90	NR		14100	1828.028	1981.534	25	0.0031	0.0041	0.0048
torus90	R	AIO	4204	1671.145	1799.627	21	0.0030	0.0030	0.0042
torus90	R	SEQ	13350	1654.944	1786.873	20	0.0030	0.0031	0.0044

carefully designed to analyze the performance of our method against noise of different intensities. To this aim, we consider the parametric:

$$\begin{aligned} x(t) &= [7 + 2\cos(5t)]\cos(2t) \\ y(t) &= [7 + 2\cos(5t)]\sin(2t) \\ z(t) &= 3\sin(5t) \end{aligned} \tag{14}$$

$$t \in [0, 2\pi],$$

which corresponds to a curve on a torus. We consider a set of 202 three-dimensional data points with uniform sampling in the interval domain $[0, 2\pi]$. This dataset, labelled as *torus* in Table 1, is then perturbed with additive white noise of different intensities, modulated by a signal-to-noise ratio (SNR) ranging from 10 (very high intensity) to 90 (low intensity), with step-size 10. The corresponding datasets are labelled as *torusN*, where *N* indicates the SNR intensity. The simulation results with our method for the resulting 10 datasets are reported in the last 10 horizontal blocks of Table 1. Some important observations can be obtained from the numerical data in the table. The most important one is that the sequential rational schema outperforms the others in terms of BIC value, meaning that it provides the best trade-off between accuracy and complexity for all instances in this example. According to our results, the optimal number of

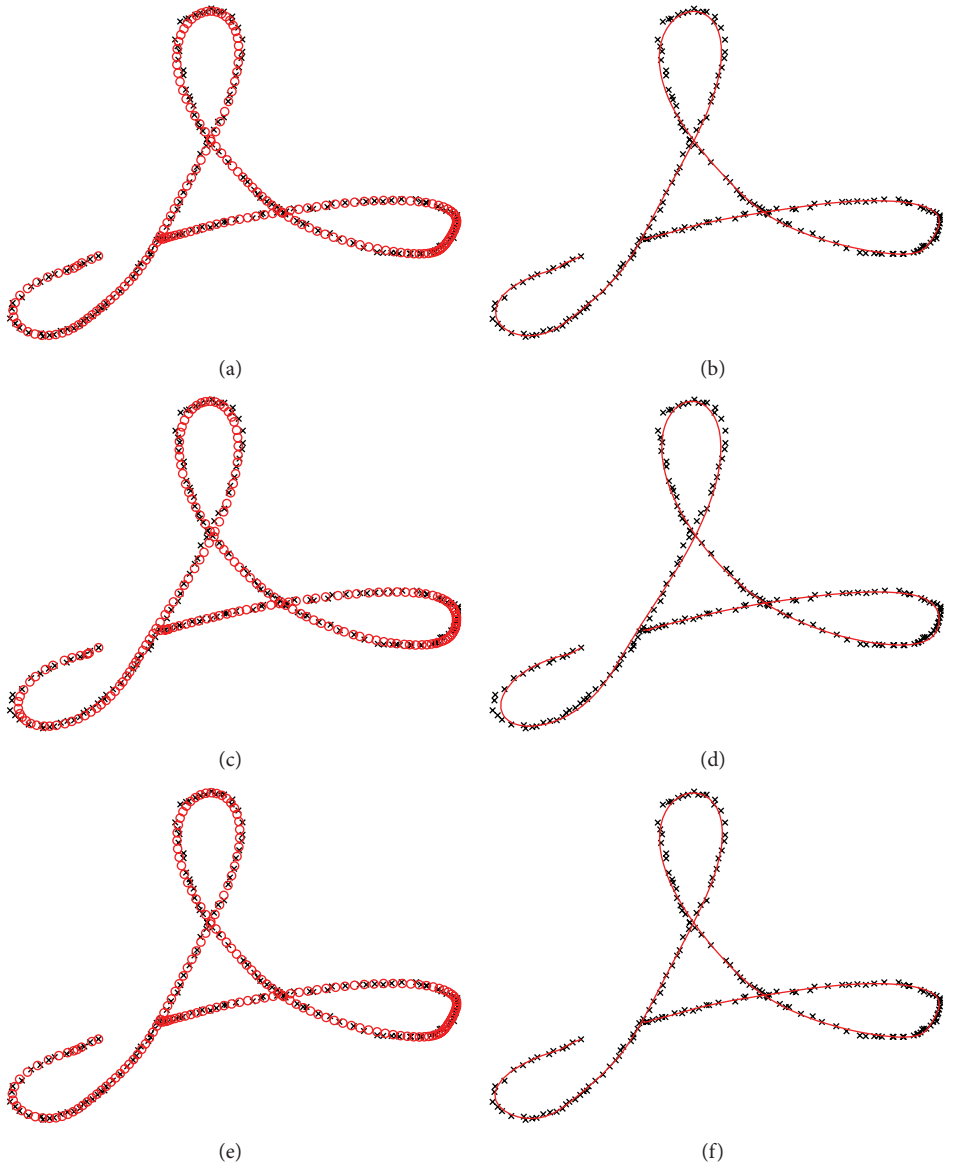

FIGURE 2: Experimental results for Example 1: left: reconstructed points (red circles); right: best fitting curve (red solid line); top: nonrational case; middle: AIO rational case; bottom: sequential rational case. In all cases, $n_{\text{pol}} = 13$.

poles for this example is $n_{\text{pol}} = 20$ in all cases, except for the instance *torus10*, which corresponds to a case of very high noise intensity. In other words, this schema is able to capture the optimal number of poles in cases of noise of low and medium intensity. Furthermore, the method fits the data points very accurately. For instance, the relative fitting error for the noiseless case is as good as 10^{-6} for each coordinate. These striking results are visually confirmed in Figure 4. Note, for instance, the very good fitting for the sequential rational schema (bottom row).

On the other hand, as expected, the BIC increases as the noise level increases, meaning that the method is affected by the noise intensity, but not drastically. In fact, the method is very resilient against noise, as it still yields very reasonable relative fitting errors of order 10^{-2} for high-intensity noise (for instance, of SNR $= 10$) and 10^{-3} for SNR $= 30$. For example, the visual quality of the fitting is clearly visible for the case SNR $= 50$, as shown in Figure 5. We remark, however, that in this case, the nonrational and AIO rational schemas require extra parameters to obtain their best fitting. Note, for instance, that $n_{\text{pol}} = 25$ for the nonrational schema in this example. This effect can be explained by the fact that the nonrational curve has less degrees of freedom because no weights are available. As a consequence, more poles are usually required to compensate this limitation. But even in this case, the value of this parameter is lower or equal to 25.

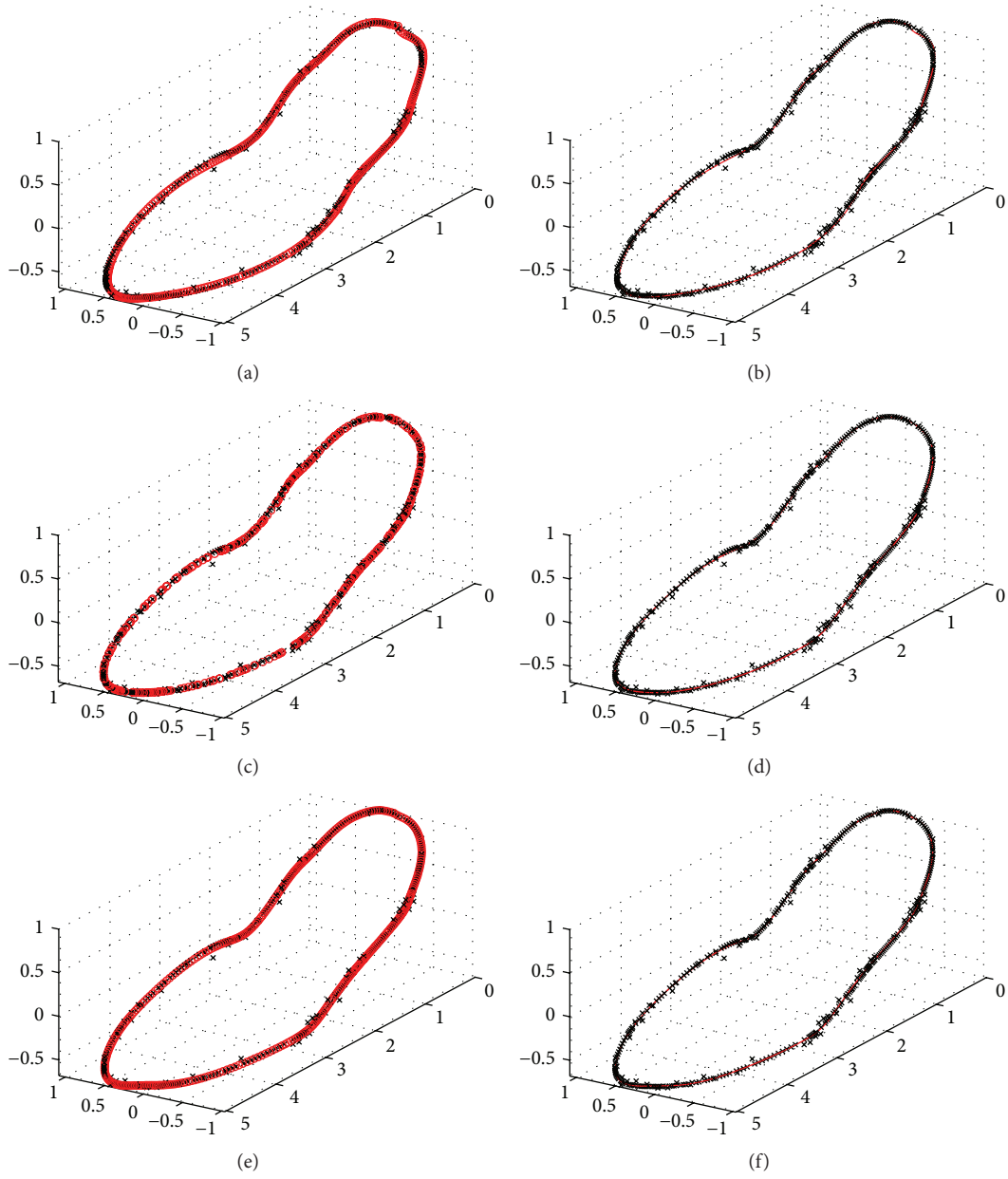

Figure 3: Experimental results for Example 2: left: reconstructed points (red circles); right: best fitting curve (red solid line); top: nonrational case; middle: AIO rational case; bottom: sequential rational case. In all cases, $n_{\text{pol}} = 24$.

This result is a clear indication of the effectiveness of our proposal to use BIC to keep the dimension of the problem as low as possible and to prevent overfitting.

This third example has also been used to illustrate the good performance of our neighborhood function, described in Section 5.2.1. Figure 7 shows two graphical examples of the evolution of the BIC for the rational all-in-one schema *versus* the number of evaluations of the fitness function, given by the parameter n_{feval}. The pictures display the examples *torus* and *torus50* from Table 1, corresponding, respectively, to the noiseless case (top) and the noisy case with SNR = 50 (bottom). Both pictures show the evolution of the maximum, mean, and minimum BIC in a color-coded representation (in blue, green, and red, resp.). These BIC values have been obtained with our method from 20 executions out of 26 executions after removing the three best and three worst results for each case. As the reader can see, our neighborhood function allows the method to escape from local minima, a situation that happens particularly at earlier stages of the evolution, associated with an intensive exploration of the search space. Two temporal windows have also been included in the pictures to enlarge these initial stages by zooming for better visualization. After this initial period, the BIC decreases slower and the fitting error reaches a plateau where the exploitation phase becomes dominant. Finally, convergence to the optimal values (marked by the vertical

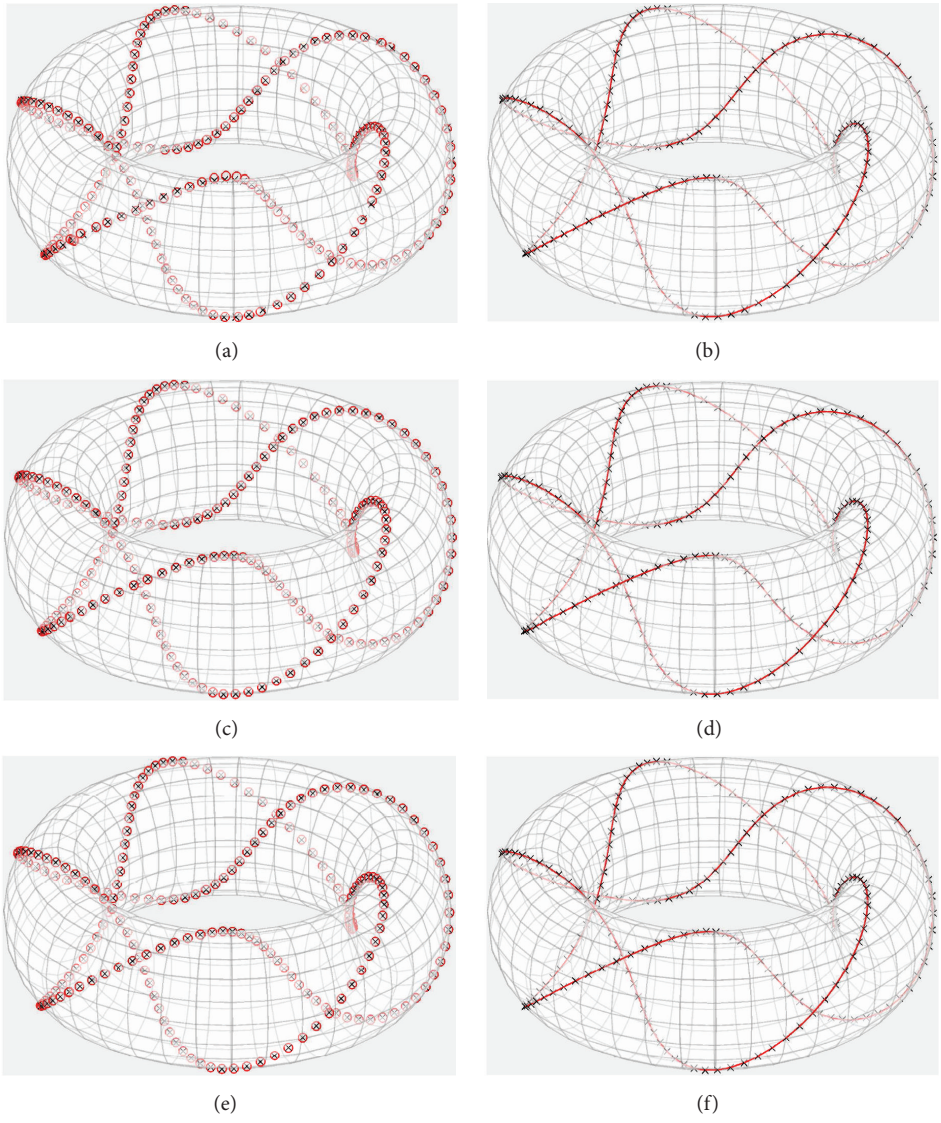

Figure 4: Experimental results for Example 3 (without noise): left: reconstructed points (red circles); right: best fitting curve (red solid line); top: nonrational case; middle: AIO rational case; bottom: sequential rational case.

magenta line) is achieved and the fitting error does no longer improve. These pictures clearly show that the method is well suited for multimodal problems, being able to escape from local minima, thus preventing premature convergence to happen.

7. Conclusions and Future Work

In this paper we introduce two new simulated annealing schemas for continuous optimization. They are applied to obtain the rational Bézier curve that fits better a given set of noisy data points in the least-squares sense. This is a very difficult problem that requires computing four different sets of unknowns: data parameters, poles, weights, and the curve degree. Besides, these free variables are strongly related to each other in a highly nonlinear way. This leads to a difficult continuous nonlinear optimization problem that cannot be decomposed into several independent subproblems. To address this challenging issue, we propose an optimization method combining classical methods (least-squares minimization), modern stochastic methods (simulated annealing), and information science metrics (Bayesian Information Criterion (BIC)). The simulated annealing algorithm is applied to perform data parameterization and to determine the weights of the poles of the fitting curve. This is done by using two different schemas: the all-in-one schema, which computes both sets of unknowns together simultaneously, and the sequential schema, which computes each set of unknowns in sequence, using the previous set of computed variables as the new input. The least-squares minimization is used to calculate the poles of the fitting curve. Finally, we apply the BIC to determine the optimal degree of the best rational Bézier fitting curve. This methodology has been applied to a benchmark of three illustrative examples of 2D

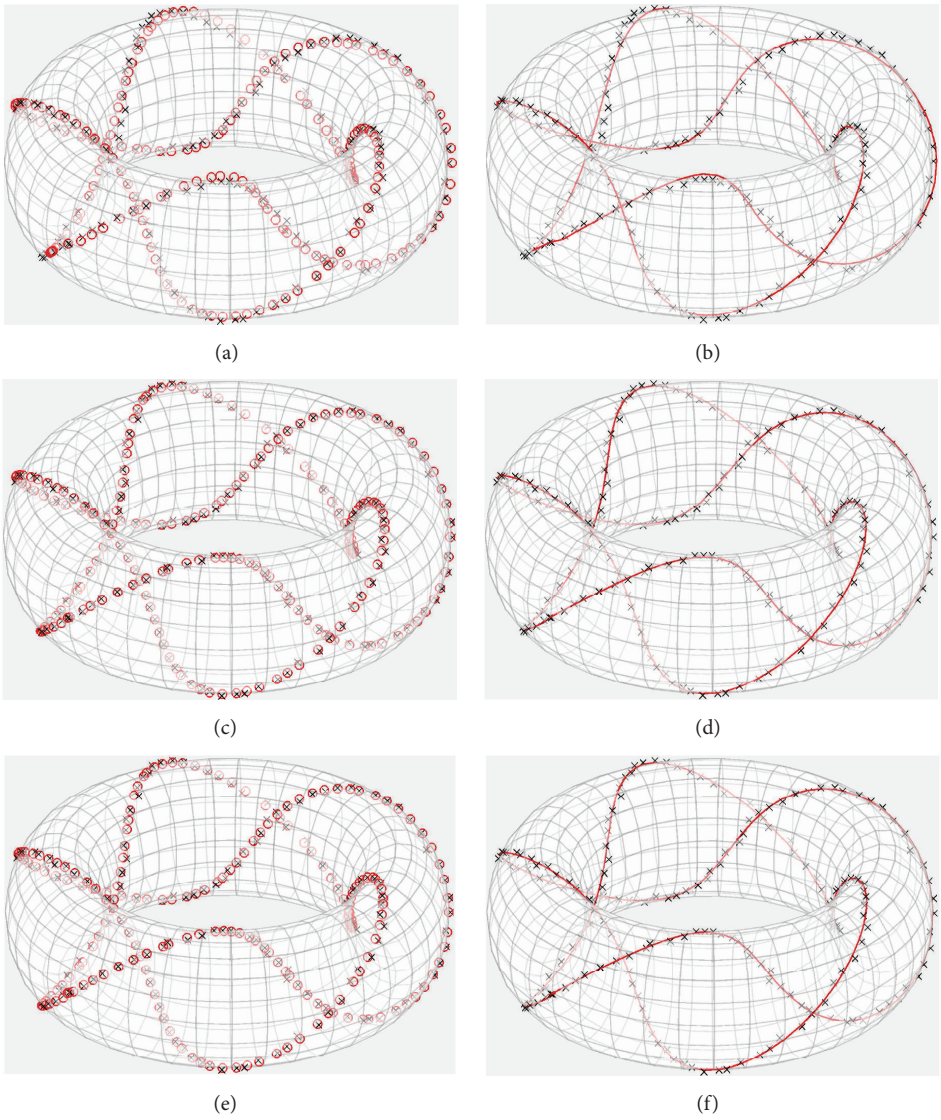

Figure 5: Experimental results for Example 3 (with SNR = 50): left: reconstructed points (red circles); right: best fitting curve (red solid line); top: nonrational case; middle: AIO rational case; bottom: sequential rational case.

and 3D noisy data points. These examples have been carefully chosen so that they exhibit challenging features such as self-intersections or strong changes of slopes and curvatures. The first two examples correspond to real-world instances that replicate the usual conditions of real-world applications, including the presence of noise of low-medium intensity. The last one is an academic example designed to analyze the effect of different levels of noise on our method.

Our computational experiments on the proposed benchmark show that our method is very suitable for the given data fitting problem. From the two schemas proposed, the sequential one performed particularly well in all instances, although the AIO schema also provides very good fitting results for our examples. These good numerical results are not obtained at the expense of a large number of variables. On the contrary, the application of the BIC allows to obtain models with a remarkable low number of free variables, even for difficult shapes with complicated features and noise. We also compared our results with our previous method with nonrational curves in [5]. The new rational method clearly outperforms our previous approach for all instances of our benchmark. Our experiments with different levels of noise on the last example also show that the method is robust against noise of low to medium intensity.

The main limitations of our approach concern its performance in situations of high-intensity noise. Figure 6 shows our results for the third example with noise of SNR = 10. In general, our method is able to capture the tendency of the data even under these strongly adverse conditions, but some problems may arise in the neighborhood of the initial and last poles of closed curves. In particular, the continuity of such curves at that point cannot be assured. This situation is not critical at all; it can readily be avoided by introducing additional constraints in our problem. However, as expected,

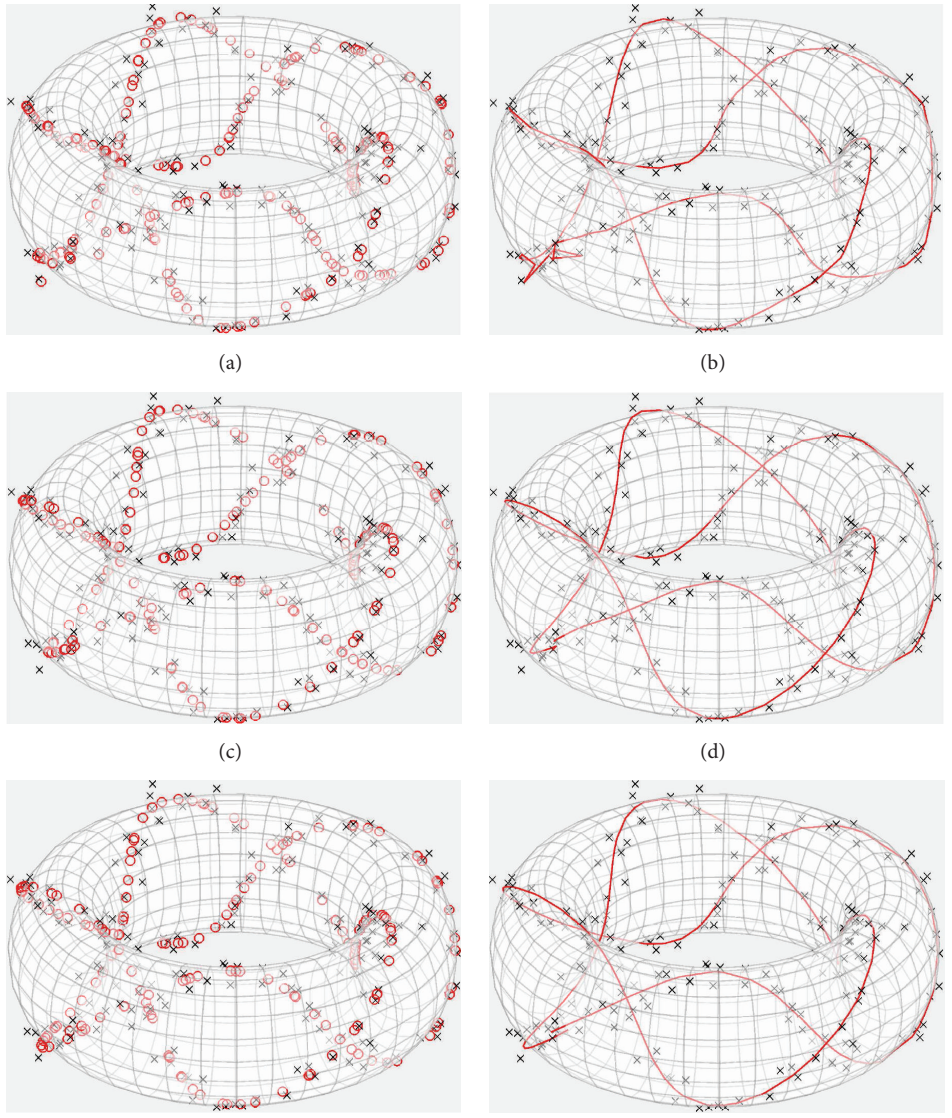

Figure 6: Experimental results for Example 3 (with SNR = 10): left: reconstructed points (red circles); right: best fitting curve (red solid line); top: nonrational case; middle: AIO rational case; bottom: sequential rational case.

the performance of the method is affected by the noise intensity, meaning that some kind of preprocessing (such as filtering) might be advisable in highly noisy environments for real-world applications.

Future work can be divided into three main directions. Firstly, we want to extend the general methodology to other families of curves well suited for data fitting. Also, the extension of this method to both nonrational and rational surfaces is part of our future work. On the other hand, we wish to apply this approach to other related engineering areas, such as robot path planning. Finally, we want to apply the methodology to different *multiobjective* engineering problems and improve its performance by taking advantage of the massive parallelism capabilities of general-purpose computing on graphics processing units (GPGPU).

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper. Any commercial identity mentioned in this paper is cited solely for scientific purposes.

Acknowledgments

This research has been kindly supported by the Computer Science National Program of the Spanish Ministry of Economy and Competitiveness, Project Ref. #TIN2012-30768, Toho University (Funabashi, Japan), and the University of Cantabria (Santander, Spain). The authors are particularly grateful to the Department of Information Science of Toho

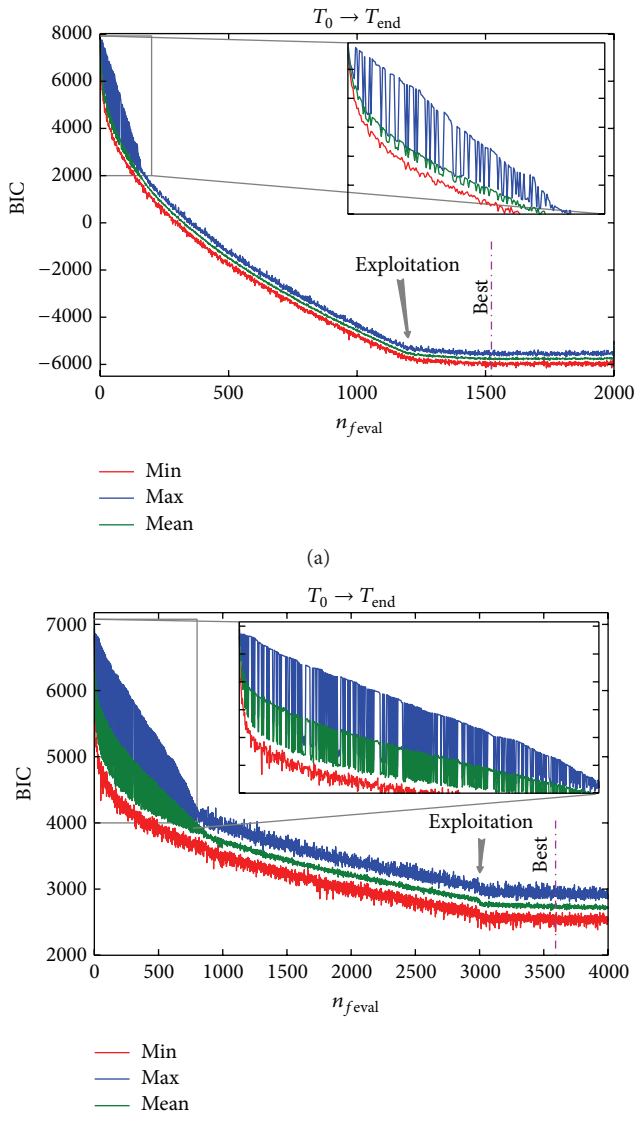

FIGURE 7: Evolution of the maximum, mean, and minimum value of BIC (in blue, green, and red color, resp.) versus the number of function evaluations for the curve on a torus example: top: noiseless case; bottom: noisy case (with SNR = 50). The inner boxed pictures also show a zoom of the initial exploration stages in both cases for better visualization.

University for all the facilities given to carry out this work. Special recognition is also due to the anonymous reviewers for several valuable comments and suggestions that helped us to improve this paper significantly.

References

[1] N. M. Patrikalakis and T. Maekawa, *Shape Interrogation for Computer Aided Design and Manufacturing*, Springer, Heidelberg, Germany, 2009.

[2] H. N. Fitter, A. B. Pandey, D. D. Patel, and J. M. Mistry, "A review on approaches for handling Bézier curves in CAD for manufacturing," *Procedia Engineering*, vol. 97, pp. 1155–1166, 2014.

[3] R. S. Tavares, T. C. Martins, and M. S. G. Tsuzuki, "Simulated annealing with adaptive neighborhood: a case study in off-line robot path planning," *Expert Systems with Applications*, vol. 38, no. 4, pp. 2951–2965, 2011.

[4] A. Kharal and A. Saleem, "Neural networks based airfoil generation for a given C_p using Bezier-PARSEC parameterization," *Aerospace Science and Technology*, vol. 23, no. 1, pp. 330–344, 2012.

[5] C. Loucera, A. Gálvez, and A. Iglesias, "Simulated annealing algorithm for Bézier curve approximation," in *Proceedings of the International Conference on Cyberworlds (CW '14)*, pp. 182–189, IEEE Computer Society Press, Santander, UK, October 2014.

[6] D. L. B. Jupp, "Approximation to data by splines with free knots," *SIAM Journal on Numerical Analysis*, vol. 15, no. 2, pp. 328–343, 1978.

[7] M. J. D. Powell, "Curve fitting by splines in one variable," in *Numerical Approximation to Functions and Data*, J. G. Hayes, Ed., Athlone Press, London, UK, 1970.

[8] J. R. Rice, *The Approximation of Functions*, vol. 2, Addison-Wesley, Reading, Mass, USA, 1969.

[9] P. Dierckx, *Curve and Surface Fitting with Splines*, Oxford University Press, Oxford, UK, 1993.

[10] L. Piegl and W. Tiller, *The NURBS Book*, Springer, Berlin, Germany, 1997.

[11] R. E. Barnhill, *Geometric Processing for Design and Manufacturing*, SIAM, Philadelphia, Pa, USA, 1992.

[12] H. Park, "An error-bounded approximate method for representing planar curves in B-splines," *Computer Aided Geometric Design*, vol. 21, no. 5, pp. 479–497, 2004.

[13] W. P. Wang, H. Pottmann, and Y. Liu, "Fitting B-spline curves to point clouds by curvature-based squared distance minimization," *ACM Transactions on Graphics*, vol. 25, no. 2, pp. 214–238, 2006.

[14] H. Park and J.-H. Lee, "B-spline curve fitting based on adaptive curve refinement using dominant points," *Computer-Aided Design*, vol. 39, no. 6, pp. 439–451, 2007.

[15] P. Gu and X. Yan, "Neural network approach to the reconstruction of free-form surfaces for reverse engineering," *Computer-Aided Design*, vol. 27, no. 1, pp. 59–64, 1995.

[16] G. K. Knopf and J. Kofman, "Adaptive reconstruction of free-form surfaces using Bernstein basis function networks," *Engineering Applications of Artificial Intelligence*, vol. 14, no. 5, pp. 577–588, 2001.

[17] M. Hoffmann, "Numerical control of Kohonen neural network for scattered data approximation," *Numerical Algorithms*, vol. 39, no. 1–3, pp. 175–186, 2005.

[18] J. Barhak and A. Fischer, "Parameterization and reconstruction from 3D scattered points based on neural network and PDE techniques," *IEEE Transactions on Visualization and Computer Graphics*, vol. 7, no. 1, pp. 1–16, 2001.

[19] G. Echevarría, A. Iglesias, and A. Gálvez, "Extending neural networks for B-spline surface reconstruction," in *Computational Science—ICCS 2002*, vol. 2330 of *Lectures Notes in Computer Science*, pp. 305–314, Springer, Berlin, Germany, 2002.

[20] A. Iglesias, G. Echevarría, and A. Gálvez, "Functional networks for B-spline surface reconstruction," *Future Generation Computer Systems*, vol. 20, no. 8, pp. 1337–1353, 2004.

[21] A. Gálvez and A. Iglesias, "Efficient particle swarm optimization approach for data fitting with free knot B-splines," *Computer-Aided Design*, vol. 43, no. 12, pp. 1683–1692, 2011.

[22] A. Gálvez, A. Iglesias, A. Avila, C. Otero, R. Arias, and C. Manchado, "Elitist clonal selection algorithm for optimal choice of free knots in B-spline data fitting," *Applied Soft Computing Journal*, vol. 26, pp. 90–106, 2015.

[23] F. Yoshimoto, T. Harada, and Y. Yoshimoto, "Data fitting with a spline using a real-coded genetic algorithm," *Computer-Aided Design*, vol. 35, no. 8, pp. 751–760, 2003.

[24] L. Zhao, J. Jiang, C. Song, L. Bao, and J. Gao, "Parameter optimization for Bezier curve fitting based on genetic algorithm," in *Advances in Swarm Intelligence*, vol. 7928 of *Lecture Notes in Computer Science*, pp. 451–458, Springer, 2013.

[25] A. Gálvez, A. Iglesias, and L. Cabellos, "Cuckoo search with lévy flights for weighted bayesian energy functional optimization in global-support curve data fitting," *The Scientific World Journal*, vol. 2014, Article ID 138760, 11 pages, 2014.

[26] L. Jing and L. Sun, "Fitting B-spline curves by least squares support vector machines," in *Proceedings of the 2nd International Conference on Neural Networks and Brain (ICNN&B '05)*, vol. 2, pp. 905–909, IEEE Press, Beijing, China, October 2005.

[27] A. Gálvez and A. Iglesias, "Firefly algorithm for explicit B-spline curve fitting to data points," *Mathematical Problems in Engineering*, vol. 2013, Article ID 528215, 12 pages, 2013.

[28] M. Sarfraz and S. A. Raza, "Capturing outline of fonts using genetic algorithm and splines," in *Proceedings of the 5th International Conference on Information Visualisation (IV '01)*, pp. 738–743, IEEE Computer Society Press, London, UK, July 2001.

[29] A. GáLvez and A. Iglesias, "A new iterative mutually coupled hybrid GA-PSO approach for curve fitting in manufacturing," *Applied Soft Computing Journal*, vol. 13, no. 3, pp. 1491–1504, 2013.

[30] G. Farin, "Algorithms for rational Bézier curves," *Computer-Aided Design*, vol. 15, no. 2, pp. 73–77, 1983.

[31] S. Kirkpatrick, J. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.

[32] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of state calculations by fast computing machines," *The Journal of Chemical Physics*, vol. 21, no. 6, pp. 1087–1092, 1953.

[33] M. Malek, M. Guruswamy, M. Pandya, and H. Owens, "Serial and parallel simulated annealing and tabu search algorithms for the traveling salesman problem," *Annals of Operations Research*, vol. 21, no. 1, pp. 59–84, 1981.

[34] D. Vanderbilt and S. G. Louie, "A Monte Carlo simulated annealing approach to optimization over continuous variables," *Journal of Computational Physics*, vol. 56, no. 2, pp. 259–271, 1984.

[35] B. Suman and P. Kumar, "A survey of simulated annealing as a tool for single and multiobjective optimization," *Journal of the Operational Research Society*, vol. 57, no. 10, pp. 1143–1160, 2006.

[36] A. Basu and L. N. Frazer, "Rapid determination of the critical temperature in simulated annealing inversion," *Science*, vol. 249, no. 4975, pp. 1409–1412, 1990.

[37] W. Ben-Ameur, "Computing the initial temperature of simulated annealing," *Computational Optimization and Applications*, vol. 29, no. 3, pp. 369–385, 2004.

[38] E. H. Aarts and P. J. Van Laarhoven, "Statistical cooling: a general approach to combinatorial optimization problems," *Philips Journal of Research*, vol. 40, no. 4, pp. 193–226, 1985.

[39] G. Schwarz, "Estimating the dimension of a model," *The Annals of Statistics*, vol. 6, no. 2, pp. 461–464, 1978.