Rend. Sem. Mat. Univ. Pol. Torino
Vol. 61, 3 (2003)
Splines and Radial Functions

M.L. Sampoli

CLOSED SPLINE CURVES BOUNDING MAXIMAL AREA

Abstract. In this paper we study the problem of constructing a closed spline curve in $\mathbb{R}^2$, which interpolates a given set of data points, is shape-preserving and which, in addition, bounds the maximal area. The construction is done by using the so-called *Abstract Schemes* (AS). The resulting spline curve, expressed in its piecewise Beźier representation, has degree 3 and continuity C^1 and can be extended to a curve of degree 6 and continuity C^2, with similar properties.

1. Introduction

In the definition and development of mathematical models which could describe real objects or real phenomena a great deal of research has been done in the field of *constrained interpolation* and approximation. Constrained interpolation indeed arises in various applications. In industry, for instance, when we are dealing with the problem of designing the network curves constituting the tail of an aircraft we should avoid any oscillations which could affect the aerodynamic properties of the resulting surface. In these cases we give additional constraints such as *smoothing* constraints or *shape-preserving* constraints. In this context, an interesting problem, important for its applications in naval engineering and ship industry, is that of constructing a closed curve in $\mathbb{R}^2$, interpolating a given set of points, shape-preserving and bounding maximal area.

Aim of the paper is indeed to present a method to solve this problem. The proposed method is based on the application of Abstract Schemes. These schemes have been developed to solve general constrained interpolation problems (see for instance [4], [17], [7]).

The basic idea behind AS is given by observing that when we interpolate some data points by a spline, and want to fulfill other requirements, we usually dispose of several free parameters $\mathbf{d}_0, \mathbf{d}_1, \ldots, \mathbf{d}_N$ ($\mathbf{d}_i \in \mathbb{R}^q$), which are associated with the knots. If we now express the constraints as conditions relative to each interval between two knots, they can be rewritten as a sequence of inclusion conditions: $(\mathbf{d}_i, \mathbf{d}_{i+1}) \in D_i \subset \mathbb{R}^{2q}$, where the sets D_i are the corresponding feasible domains. In this setting the problems of existence, construction and selection of an optimal solution can be studied with the help of Set Theory in a general way.

The remainder of the paper is organized as follows. In the next section the fundamental ideas of AS will be recalled. In order to make the paper self-contained it is more convenient to present here the basic ideas and the main results on AS, although they can be found in other papers as well. In Section 3 we shall present the application of AS for the construction of C^1 cubic curves with maximal area. The C^2 spline curves will be constructed in Section 4 and the examples and the final conclusions will be reported in Section 5.

2. Basic ideas on abstract schemes

The main idea which gave rise to abstract schemes was the observation that most methods used in constrained interpolation have a common structure, even if, at a first sight, they seem quite different from each other. This structure can be sketched as follows: first a suitable set of piecewise functions is chosen and a set of parameters $(\mathbf{d}_0, \mathbf{d}_1, \ldots, \mathbf{d}_N)$ is selected. Then, each function piece is expressed using these parameters. A further step consists in rewriting the constraints in terms of these parameters and deriving a set of *admissible domains* D_i. This common procedure is thus exploited to build up a general theory, in order to check the feasibility of the problem, and then to provide a general purpose algorithm for computing a solution (given by an *optimal* sequence).

The first abstract schemes were developed some years ago by Schmidt and independently by Costantini, see for instance [16] and [3] (and the survey papers [5], [17]). In those papers an abstract algorithm to construct univariate functions subjected to separable constraints was presented.

Let us see now a rigorous formulation. Supposing we are given the sequences of sets $D_i \subset \mathbb{R}^q \times \mathbb{R}^q$, with $D_i \neq \emptyset$, for every $i = 0, 1, ..., N-1$, which define the constraint domains, we may define the *global* set

$$\mathbf{D} := \{(\mathbf{d}_0, \mathbf{d}_1, \ldots, \mathbf{d}_N) \in \mathbb{R}^{q\ (N+1)} \quad s.\,t.\ (\mathbf{d}_i, \mathbf{d}_{i+1}) \in D_i,\ i = 0, 1, \ldots, N-1\}.$$

This is the solution set. Therefore a problem of constrained interpolation can be suitably reduced to the study of set $\mathbf{D}$. Indeed we shall consider the following problems

P1 *Is* $\mathbf{D}$ *non empty? In other words do there exist sequences* $(\mathbf{d}_0, \mathbf{d}_1, ..., \mathbf{d}_N)$ *such that*

$$(\mathbf{d}_i, \mathbf{d}_{i+1}) \in D_i,\ i = 0, 1, \ldots, N-1. \tag{1}$$

P2 *If there exist sequences fulfilling (1), is it possible to build up an algorithm which computes one among them efficiently?*

Obviously, if the solution is not unique we will select *the best one*, that is the sequence which will minimize or maximize some objective functional (we will see that in our case the functional is related with the area).

We assume the problem is *well posed*, in the sense that the sets D_i are supposed non empty for every i; we suppose also that D_i are closed sets, for every i (not necessarily compact). Moreover in general, as the solution is always made up piecewisely, we shall adopt the same notation used for piecewise curves: we call *i-th segment* the portion of the solution from the i-th to the $i+1$-st breakpoint.

A solution to these problems can be obtained using a two-sweep strategy ([4]), processing the data first in one *direction*, for instance *from left to right*, through algorithm A1 (forward sweep), and then in the opposite direction, through algorithm A2 (backward sweep). In more detail, let us denote with $\Pi_1,\ \Pi_2 : \mathbb{R}^q \times \mathbb{R}^q \to \mathbb{R}^q$ the projection maps from the "x_1x_2-plane" onto the "x_1-axis" and "x_2-axis" respectively, and let us define the sets

$$B_i := \Pi_1(D_i)\ ;\quad i = 0, 1, \ldots, N-1\ ;\qquad B_N := \mathbb{R}^q\ . \tag{2}$$

Now, in the forward sweep, as we may observe that for every parameter $\mathbf{d}_i$ either the constraint domain coming from the segment $(i-1, i)$, that is D_{i-1}, or the one coming from the segment $(i, i+1)$, that is D_i, have to be taken into account, we determine, for every parameter, the *true* admissible domain A_i. This is indeed done by algorithm A1.

Algorithm A1.

1. *Set* $A_0 := B_0$, $J := N$

2. *For* $i = 1, \ldots, N$

  2.1 *Set* $A_i := \Pi_2(D_{i-1} \cap \{A_{i-1} \times B_i\})$.

  2.2 *If* $A_i = \emptyset$ *set* $J := i$ *and stop.*

3. *Stop.*

In this connection, we have the following result, [4].

THEOREM 1. *P1 has a solution if, and only if,* $J = N$ *that is* $A_i \neq \emptyset$, $i = 0, 1, \ldots, N$. *If* $(\mathbf{d}_0, \mathbf{d}_1, \ldots, \mathbf{d}_N)$ *is a solution then*

$$\mathbf{d}_i \in A_i\ ;\ i = 0, 1, \ldots, N. \tag{3}$$

We remark that, in general, a solution of P1 is not unique and that the necessary condition (3) is not sufficient. Thus, if the sequence of non empty sets $A_0, \ldots, A_N$ has been defined by algorithm A1, a first simple scheme for computing a sequence $(\mathbf{d}_0, \mathbf{d}_1, \ldots, \mathbf{d}_N)$ is provided by the following algorithm (backward sweep) whose effectiveness is guaranteed by Theorem 2 (we refer again to [4] for the proof).

Algorithm A2.

1. *Choose any* $\mathbf{d}_N \in A_N$.

2. *For* $i = N-1, N-2, \ldots, 0$

  2.1 *set* $C_i(\mathbf{d}_{i+1}) := \Pi_1(D_i \cap \{A_i \times \{\mathbf{d}_{i+1}\}\})$

  2.2 *Choose any* $\mathbf{d}_i \in C_i(\mathbf{d}_{i+1})$

3. *Stop.*

THEOREM 2. *Let the sequence* $A_0, A_1, \ldots, A_N$ *be given by algorithm A1, with* $A_i \neq \emptyset$; $i = 0, 1, \ldots, N$. *Then algorithm A2 can be completed (that is the sets* $C(\mathbf{d}_{i+1})$ *are not empty) and any sequence* $(\mathbf{d}_0, \mathbf{d}_1, \ldots, \mathbf{d}_N)$ *computed by algorithm A2 is a solution for problem P2.*

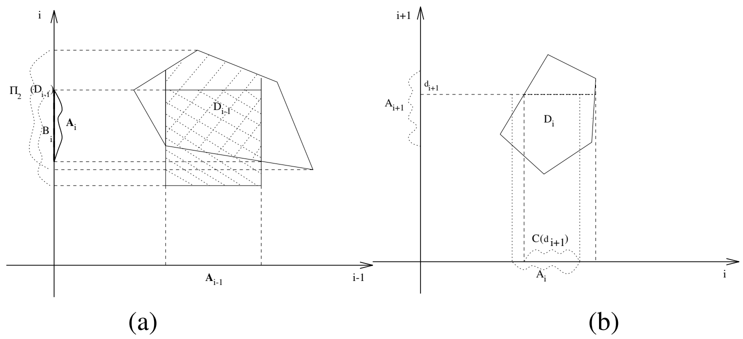


Figure 1: Algorithm 1: construction of the admissible domains A_i,(a). Algorithm 2: graphical sketch of step 2.1.(b).

2.1. Boundary conditions

When we are constructing closed curves the end points should be handled as the other inner points so that the solution $\mathbf{s}(t)$ has to satisfy the condition $\mathbf{s}^{(k)}(x_0) = \mathbf{s}^{(k)}(x_N)$ for $k = 0, 1, \ldots$ up to the continuity order considered. This kind of conditions are called *non separable* boundary conditions. In terms of abstract scheme formalization the above conditions reduce to find a sequence $(\mathbf{d}_0, \mathbf{d}_1, \ldots, \mathbf{d}_N)$ such that $\mathbf{d}_N = \beta(\mathbf{d}_0)$, where β is any continuous function with continuous inverse.

These conditions, giving a direct relationship between the first and last element of the sequence $(\mathbf{d}_0, \mathbf{d}_1, \ldots, \mathbf{d}_N)$, destroy the sequential structure of our scheme. A possible strategy would consist in considering, among all the sequences $(\mathbf{d}_0, \mathbf{d}_1, \ldots, \mathbf{d}_N)$ which belong to $\mathbf{D}$, the ones where starting with an element $\mathbf{d}_0 \in A_0$, end up with $\mathbf{d}_N = \beta(\mathbf{d}_0)$. In other words we first check whether there are in the admissible domain A_0 some $\mathbf{d}_0$ such that $\beta(\mathbf{d}_0) \in A_N$ or, equivalently ,whether $\beta(A_0) \cap A_N \neq \emptyset$. Then, in the backward sweep we pick up one of these $\mathbf{d}_N$ and go back ending in $\mathbf{d}_0 = \beta^{-1}(\mathbf{d}_N)$.

A full development of this procedure, along with the conditions under which such sequences exist and the related constructive algorithms to determine them, would require some theory of set-valued maps whose details are beyond the scope of this paper. We refer to [4] for theoretical aspects and to [6], [8] for an idea of some practical applications.

2.2. Selecting an optimal solution

It is clear from algorithms A1 and A2 that it is possible to find infinite sequences $(\mathbf{d}_0, \mathbf{d}_1, \ldots, \mathbf{d}_N)$ satisfying the constraints, as in algorithm A2 the admissible sets $C_i(\mathbf{d}_{i+1})$, defined in step 2.1, do not reduce, in general, to a single point. It is therefore a natural idea to look for an *optimal* sequence, where the optimality criterion can be given as the maximum or the minimum of a suitable functional F that is

$$\max_{(\mathbf{d}_0,\mathbf{d}_1,\ldots,\mathbf{d}_N)\in\mathbf{D}} F(\mathbf{d}_0, \mathbf{d}_1, \ldots, \mathbf{d}_N)\ , \tag{4}$$

and we specify the functional F according to our requirements. Although several forms of functionals could be considered, for the sake of simplicity, we shall limit ourselves to

$$F(\mathbf{d}_0, \mathbf{d}_1, \ldots, \mathbf{d}_N)\ :=\ \sum_{i=0}^{N-1} g_i(\mathbf{d}_i, \mathbf{d}_{i+1}), \tag{5}$$

where g_i gives the local contribution to the objective function and, in shape-preserving problems, can be connected with the shape of the resulting function (we shall consider the area bounded by a closed curve).

To solve the optimization problem we present here an approach based on dynamic programming (DP) [2]. As we will see later, this approach is well suited to deal with discrete problems. Moreover DP is extremely flexible, as many functionals and any kind of separable constraints (i.e. constraints which can be related to only one curve segment and then whic can be expressed *separately* from segment to segment) can be processed using the same algorithmic structure and, unlike other optimization methods, constraints play here a positive role, limiting the size of the *decision space*. In this regard, we may observe that the functional recurrence relations of dynamic programming can be very efficiently linked with the constraints in Algorithm A2. We refer to [9] for full details on how implement dynamic programming in Algorithm A2.

Below is reported a sketch of the algorithm where we have stored in Φ_i the cost associated with the i-th stage and in T_i is stored the optimal policy (therefore $\max_{(\mathbf{d}_0,\mathbf{d}_1,\ldots,\mathbf{d}_N)\in\mathbf{D}} F(\mathbf{d}_0, \mathbf{d}_1, \ldots, \mathbf{d}_N) = \max_{\mathbf{d}_N} \Phi_N(\mathbf{d}_N)$). As a consequence, starting with the optimal $\mathbf{d}_N$, we obtain the optimal $\mathbf{d}_{N-1} := T_{N-1}(\mathbf{d}_N)$ and so on.

Algorithm A2DP.

1. *For any* $\delta_0 \in A_0$ *set* $\Phi_0(\delta_0) := 0$

2. *For* $i = 1, 2, \ldots, N$

2.1 *For any* $\delta_i \in A_i$ *compute* $C_{i-1}(\delta_i) := \Pi_1(D_{i-1} \cap \{A_{i-1} \times \{\delta_i\}\})$

2.2 *For any* $\delta_i \in A_i$ *compute* $\Phi_i(\delta_i) := \max_{\delta_{i-1}\in C_{i-1}(\delta_i)} (g(\delta_{i-1}, \delta_i) + \Phi_{i-1}(\delta_{i-1})) = g(T_{i-1}(\delta_i), \delta_i) + \Phi_{i-1}(T_{i-1}(\delta_i))$ *and the corresponding optimizing value* $T_{i-1}(\delta_i)$

3. *Compute* $\mathbf{d}_N$ *such that* $\Phi_N(\mathbf{d}_N) = \max_{\delta_N \in A_N} \Phi_N(\delta_N)$

4. *For* $i = N-1, \ldots, 0$

4.1. $\mathbf{d}_i := T_i(\mathbf{d}_{i+1})$

5. *Stop.*

2.3. Multivariate case

The two-sweep scheme, given by algorithms A1 and A2 (or A2DP), has turned out to be an effective method to solve several problems and its main attraction relies in the fact that it is *general*, being applicable to a wide range of problems.

However an its closer inspection shows us that, although there is no assumption on the subsets D_i of $\mathbb{R}^q \times \mathbb{R}^q$, which are the basic elements of the imposed constraints, the practical usage of this method has been so far confined to the case $D_i \in \mathbb{R} \times \mathbb{R}$, that means $q = 1$. This is due to the fact that in algorithms, either A1 or A2 we have to compute the projection of intersections of subsets in a product space. More precisely, we may recall, for instance, that step 2.1 of A1, which is the kernel of all the modifications and improvements later developed, requires the computation of the following set:

$$A_i := \Pi_2(D_{i-1} \cap \{A_{i-1} \times B_i\}),$$

and this leads, even in the simplest higher dimension case, that is $q = 2$, to intersections and projections of arbitrary subsets of $\mathbb{R}^2 \times \mathbb{R}^2$. Even in the case of linear inequalities for the constraints (D_i would be a polytope of $\mathbb{R}^4$), the corresponding algorithm is extremely difficult to implement and has an unaffordable computational cost. Indeed, in $\mathbb{R}^q$, the computational cost of set intersections and their projections is given by $O(n^{q-1}\ log\ n)$, where n is the number of polytope vertices, see [15] for full details.

Thus, the practical application of abstract schemes has been for many years restricted to *univariate problems*, where we have only one parameter associated with every knot (two for every segment). This limitation is rather restrictive as univariate problems suffice in general to model interpolation of *functions*, while are not suitable for interpolation of *parametric curves*, which can represent closed curves. We may see that usually parametric planar curve interpolation gives rise already to constraint domains in $\mathbb{R}^2 \times \mathbb{R}^2$.

Recent research has been therefore devoted to develop a new theory and construct new methods suitable and applicable to *multivariate* constraint problems (see for instance [7], [14]). It is worthwhile to repeat that the dimension q of the parameter space is not related to the dimension of the point space we are working in.

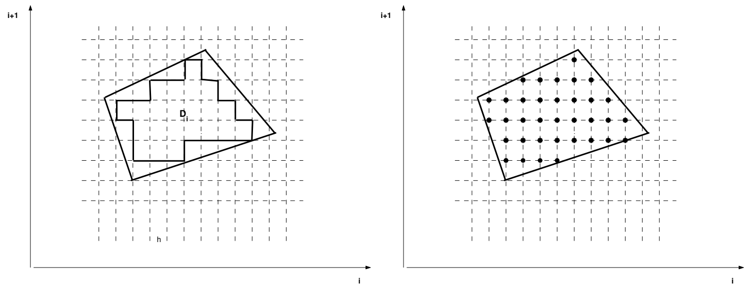
Recently a new approach has been proposed (see [9], [8]). It is based on the observation that if we consider the union of $2q$-*boxes* (i.e. rectangular parallelepipeds with facets parallel to the coordinate hyperplanes) the computational cost of their intersections and projections is reduced to $O(n\ log^{q-1}\ n)$, [15]. The basic simple idea of the new method is that of approximating the constraint domains D_i with a union of $2q$-boxes $\widetilde{D}_i$

We refer to [9] for full details on this new approach. For the sake of completeness we report here the main ideas on how this approximation is performed.

For every domain D_i, we suppose we are able to give an estimate of a lower and/or upper bound for each dimension. Then we may choose a step size $\mathbf{h} = (h_1, h_2, \ldots, h_q)$ and, starting from the lower (upper) bound, construct a multidimensional grid in $\mathbb{R}^q \times \mathbb{R}^q$ whose dimension is assigned. We thus approximate every domain D_i with the union of those boxes whose vertices are contained in D_i. By construction we have $\widetilde{D}_i \subseteq D_i$ and we may easily see that, given $\mathbf{h}$, for $\mathbf{h} \to \mathbf{0}$, we have $meas\ (D_i \backslash \widetilde{D}_i) \to 0$.

The next step consists in making a further approximation. Once we have obtained the domains $\widetilde{D}_i$, we consider only the discrete values for the parameters $(\mathbf{d}_i, \mathbf{d}_{i+1})$, corresponding to the vertices of the considered boxes. This is equivalent to working with discrete domains, which we denote by $\overline{D}_i$. We then select the points of the grid which are vertices of a $2q$-box contained in D_i. At the end of this process we obtain a sequence of domains $\overline{D}_i$ such that again approximate D_i and $\overline{D}_i \subseteq D_i$.

As in the continuous case, we may select an optimal solution by optimizing a suitable functional, using dynamic programming. The fact that the parameters $\mathbf{d}_i$ vary in discrete domains is well suited for applying the dynamic programming in the minimization process. Regarding the convergence analysis, the following result holds (we refer to [9] for the proof).



(a) (b)

Figure 2: (a):Every domain D_i is replaced by a union of $2q$-boxes.(b) only the values at the vertices of the considered boxes are taken.

THEOREM 3. *Let the domains $D_0, D_1, \ldots, D_{N-1}$, with $D_i \subset \mathbb{R}^q \times \mathbb{R}^q$, be given. Let $\overline{D}_0, \overline{D}_1, \ldots, \overline{D}_{N-1}$ be the corresponding discrete domains obtained with a grid of step size* $\mathbf{h}$. *Let us denote now with $(\mathbf{d}_0^*, \mathbf{d}_1^*, \ldots, \mathbf{d}_N^*)$ a solution in* $\mathbf{D}$ *which maximizes also a continuous functional F, with a unique absolute maximum, and let $(\bar{\mathbf{d}}_0^*, \bar{\mathbf{d}}_1^*, \ldots, \bar{\mathbf{d}}_N^*)$ be a discrete counterpart. Then*

$$\lim_{h_{max} \to 0} (\bar{\mathbf{d}}_0^*, \bar{\mathbf{d}}_1^*, \ldots, \bar{\mathbf{d}}_N^*) = (\mathbf{d}_0^*, \mathbf{d}_1^*, \ldots, \mathbf{d}_N^*)\ , \quad h_{max} := \max(h_1, h_2, \ldots, h_q)\ .$$

We remark that as the parameters $(\mathbf{d}_i, \mathbf{d}_{i+1})$ can assume only the discrete values corresponding to non zero elements of the i-th logical matrix, the operations of intersection, projection, cartesian product, etc. are easily performed on the matrix by the logical operators AND, OR, taking only some planes, putting together more planes and so on. This way of proceeding has revealed to be very effective from the computational point of view and it can be extended straightforwardly to domains in $\mathbb{R}^q \times \mathbb{R}^q$ (the number of planes is in general given by $2q$).

3. Interpolating spline curves maximizing the bounded area

Let us suppose we are given a set of points $\mathbf{I}_i \in \mathbb{R}^2$, $i = 0, 1, \ldots, N$, along with a parameterization t_i $i = 0, 1, \ldots, N$, (for instance chord length). Let us indicate with $\mathbf{L}_i := \mathbf{I}_{i+1} - \mathbf{I}_i$ the polygonal data and set the quantities $k_i := t_{i+1} - t_i$, $\widetilde{\mathbf{L}}_i := (\mathbf{I}_{i+1} - \mathbf{I}_i)/k_i$.

Our goal is to construct an interpolating C^1 spline curve, $\mathbf{s}(t)$, which bounds maximal area. In order to obtain a curve useful for the applications we consider the additional constraint of shape preservation, i.e. we require that the resulting curve preserves the convexity and inflections as prescribed by the given data.

To express the spline curve we use the piecewise Bézier representation so that each spline segment is a Bézier curve

$$\mathbf{s}_i(t) := \sum_{j=0}^{n} \mathbf{b}_{j,i} B_{n,j}(t) , \qquad t \in [t_i, t_{i+1}] ,$$

where $\mathbf{b}_{j,i}$ are called *control points* and form the *control polygon* and $B_{n,j}(t)$ are the Bernstein polynomials of degree n, defined by $B_{n,j}(u) = \left(\begin{array}{c} n \\ j \end{array} \right) (1-u)^{n-j} u^j$, and $u = \frac{t-t_i}{t_{i+1}-t_i}$.

From the properties of Bézier splines we have that the resulting curve will be uniquely determined once the control polygon for every segment is constructed, see for instance [11]. By construction we have that $\mathbf{b}_{0,i} = \mathbf{I}_i$ and $\mathbf{b}_{n,i} = \mathbf{I}_{i+1}$, in this way, as Bézier curves pass through the first and the last control points, we are guaranteed the resulting curve interpolates the given data.

Let us consider now curves of degree three. In this case, for each segment i, the control points to be determined are $\mathbf{b}_{0,i}$, $\mathbf{b}_{1,i}$, $\mathbf{b}_{2,i}$, and $\mathbf{b}_{3,i}$. The first and the last points are set equal to two interpolation data points. So we have to determine the two inner control points. From cubic Bézier polynomial properties we have that the global curve is C^1 if and only if, for every segment i, the inner control points $\mathbf{b}_{1,i}$ and $\mathbf{b}_{2,i}$ are taken as follows

$$\mathbf{b}_{1,i} := \mathbf{I}_i + \frac{k_i}{3}\mathbf{T}_i, \quad \mathbf{b}_{2,i} := \mathbf{I}_{i+1} - \frac{k_i}{3}\mathbf{T}_{i+1}, \tag{6}$$

where the vectors $\mathbf{T}_i$ and $\mathbf{T}_{i+1}$, are respectively the curve tangent vectors at interpolation points $\mathbf{I}_i$ and $\mathbf{I}_{i+1}$.

If we express the tangent vectors as

$$\mathbf{T}_i \ := \ u_i \widetilde{\mathbf{L}}_{i-1} + v_i \widetilde{\mathbf{L}}_i \ ; \qquad i = 0, \ldots, N. \tag{7}$$

where we set $\mathbf{I}_{-1} = \mathbf{I}_{N-1}$, $t_{-1} = t_{N-1}$, and $\mathbf{I}_{N+1} = \mathbf{I}_1$, $t_{N+1} = t_1$, the free parameters to be determined in order to uniquely construct the resulting curve are the values of u_i and v_i, for every segment $i = 0, \ldots, N-1$. These being two dimensional vectors, the problem is called *bivariate*.

It is a standard practice to express the shape-preserving conditions through the control points. Indeed, due to the *variation diminishing* property of cubic Bézier polynomials, a sufficient condition to locally reproduce the convexity of the polygonal data is that the corresponding control polygon is convex as well. This condition can be given by the following relationships

$$(8) \qquad \begin{aligned} &(\widetilde{\mathbf{L}}_{i-1} \wedge \widetilde{\mathbf{L}}_i) \cdot ((\mathbf{b}_{1,i} - \mathbf{I}_i) \wedge (\mathbf{b}_{2,i} - \mathbf{b}_{1,i})) \ \geq 0 \\ &(\widetilde{\mathbf{L}}_i \wedge \widetilde{\mathbf{L}}_{i+1}) \cdot ((\mathbf{b}_{2,i} - \mathbf{b}_{1,i}) \wedge (\mathbf{I}_{i+1} - \mathbf{b}_{2,i})) \geq 0 \ . \end{aligned}$$

We have to reformulate the shape-preserving conditions in terms of parameters u_i and v_i. The resulting curve should belong to the portion plane between $\mathbf{L}_i$ and the prolongations of $\mathbf{L}_{i-1}$ and $\mathbf{L}_{i+1}$; this can be assured imposing this condition to the tangent vectors giving rise to the condition $u_i, v_i \ \geq \ 0$. Moreover, a necessary condition for (8) (which can be easily deduced from the limit case of collinear data points) is given by

$$(9) \qquad 0 \leq u_i \leq 3, \quad 0 \leq v_i \leq 3 \ .$$

Let us then define the quantities $\rho_i \ := \|(\widetilde{\mathbf{L}}_{i-1} \wedge \widetilde{\mathbf{L}}_i)\|$ and $\sigma_i \ := \|(\widetilde{\mathbf{L}}_{i-1} \wedge \widetilde{\mathbf{L}}_{i+1})\|$, for $i = 0, \ldots, N$. Using (7) the conditions (8) can be rewritten, after straightforward computations, as

$$(10) \qquad \begin{cases} u_i(3 - u_{i+1})\rho_i^2 - u_i v_{i+1} \ \rho_i \sigma_i - v_i v_{i+1} \rho_i \rho_{i+1} \geq 0 \\ v_{i+1}(3 - v_i)\rho_{i+1}^2 - u_i v_{i+1} \ \rho_{i+1} \sigma_i - u_i u_{i+1} \rho_i \rho_{i+1} \geq 0 \ \ . \end{cases}$$

From the above expressions we get immediately the form of the constraints domains,

$$(11) \qquad \begin{aligned} D_i \ := \{ \quad & (u_i, v_i, u_{i+1}, v_{i+1}) \in \mathbb{R}^2 \times \mathbb{R}^2 \ \text{ such that} \\ & u_i(3 - u_{i+1})\rho_i^2 - u_i v_{i+1} \ \rho_i \sigma_i - v_i v_{i+1} \rho_i \rho_{i+1} \geq 0 \ ; \\ & v_{i+1}(3 - v_i)\rho_{i+1}^2 - u_i v_{i+1} \ \rho_{i+1} \sigma_i - u_i u_{i+1} \rho_i \rho_{i+1} \geq 0 \ ; \\ & u_i, v_i, u_{i+1}, v_{i+1} \ \geq \ 0\}. \end{aligned}$$

3.1. Optimization process

Our goal is to select as optimal solution the one which maximize a suitable functional related to the area bounded by the curve (applying Algorithm A2DP).

Therefore we have to compute the area of the region bounded by a parametric closed curve. We observe that such a region can be divided into two parts: the region enclosed in the polygonal line connecting the data points and the region between the polygonal line and the spline curve. As we consider *interpolating* spline curves the region bounded by the polygonal line is fixed, therefore we may restrict our attention to maximizing the area (which can be also negative) between the polygonal line and the curve.

The construction of the curve is done segment by segment (piecewisely), we may then maximize for each segment the area S_i bounded by the curve $s(t)$ for $t \in [t_i, t_{i+1}]$ and the line connecting $\mathbf{I}_i$ and $\mathbf{I}_{i+1}$. Using polar coordinates we may see that the area S_i can be expressed in the following way

$$S_i := \int_{t_i}^{t_{i+1}} (x(t)\frac{dy}{dt} - y(t)\frac{dx}{dt})dt \tag{12}$$

where $x(t)$ and $y(t)$ are respectively the x and y components of the parametric curve $s(t)$. We should remark, that in this case the data points have to be ordered anticlockwise. The global functional we maximize is then given by

$$F(\mathbf{d}_0, \mathbf{d}_1, \ldots, \mathbf{d}_N) := \sum_{i=0}^{N-1} S_i \ . \tag{13}$$

Regarding the error we make in the discretisation process, it can be proven the following result (for the proof we refer again to [8]).

THEOREM 4. *For every segment* $[i, i+1]$*, let us define the discrete and the continuous optimal solutions respectively*

$$\bar{s}_i^* := \bar{s}^*_{|[t_i, t_{i+1}]} \ ; \ \ s_i^* := s^*_{|[t_i, t_{i+1}]} \ ,$$

where, the first one is computed using the discrete domains whose grid step size is $\mathbf{h}$*, then, setting* $h_{max} = \max(h_1, h_2, \ldots, h_q)$ *and* $k_i = t_{i+1} - t_i$ *we have*

$$\|\bar{s}_i^*(t) - s_i^*(t)\|_\infty \le h_{max} \ \frac{k_i}{4} \ . \tag{14}$$

4. C^2 Continuity

We consider now the problem of constructing spline curves with C^2 continuity. The idea is to start from a C^1 piecewise curve and modify it such that it still satisfies the imposed constraints, has maximal area, and it is also of the required continuity order along the pieces.

Thus, once we have obtained the cubic spline curve, interpolating the data set, convexity preserving and bounding maximal area, with C^1 continuity along the segments, we raise the degree of the resulting spline up to six and work on its Bézier control polygon in order to construct a shape-preserving C^2 interpolating spline with again bounds the maximal area. Following [8] this construction can be performed with the help of abstract schemes.

Let us consider the generic i-th interval. Taking the control polygon of the cubic Bézier curve $\mathbf{I}_i, \mathbf{b}_{1,i}, \mathbf{b}_{2,i}, \mathbf{I}_{i+1}$ and inserting a collinear point in every polygonal segment we obtain a control polygon with seven vertices, namely $\mathbf{I}_i, \mathbf{b}^*_{1,i}, \mathbf{b}^*_{2,i} = \mathbf{b}_{1,i}, \mathbf{b}^*_{3,i}, \mathbf{b}^*_{4,i} = \mathbf{b}_{2,i}, \mathbf{b}^*_{5,i}, \mathbf{I}_{i+1}$, corresponding to a Bézier curve of degree six. By construction, at each end point the curve has second derivative equal to zero, giving rise to a global C^2 spline curve. Moreover, as the shape of the control polygon is not changed the shape-preserving properties are maintained.

For the sake of simplicity the three additional points, $\mathbf{b}^*_{1,i}, \mathbf{b}^*_{3,i}, \mathbf{b}^*_{5,i}$, can be taken as midpoints of each polygonal segment.

The C^1 continuity is guaranteed if we keep the points $\mathbf{b}^*_{1,i}$ and $\mathbf{b}^*_{5,i}$ fixed. On the other hand the requirement of C^2 continuity $\ddot{s}_i(t_{i+1}) = \ddot{s}_{i+1}(t_{i+1})$ can be rewritten in terms of control points, taking into account that for curve of degree six we have (see for instance [11])

$$\text{(15)} \qquad \ddot{s}_i(t_{i+1}) = \frac{30}{k_i^2}(\mathbf{b}^*_{4,i} + \mathbf{I}_{i+1} - 2\,\mathbf{b}^*_{5,i}) ,$$

$$\text{(16)} \qquad \ddot{s}_{i+1}(t_{i+1}) = \frac{30}{k_{i+1}^2}(\mathbf{b}^*_{2,i+1} + \mathbf{I}_{i+1} - 2\,\mathbf{b}^*_{1,i+1}) ,$$

where, as usual, $k_i = t_{i+1} - t_i$. We recall that the C^2 curve so far constructed corresponds to the limit position $\mathbf{b}^*_{2,i} = \mathbf{b}_{1,i}$ and $\mathbf{b}^*_{4,i} = \mathbf{b}_{2,i}$, with $\mathbf{b}_{1,i}$, $\mathbf{b}_{2,i}$ control points of the cubic curve.

In general the C^2 continuity gives a linear relationship between $\mathbf{b}^*_{2,i+1}$'s and $\mathbf{b}^*_{4,i}$'s. If we keep fixed the points $\mathbf{b}^*_{3,i}$'s, we may choose the points $\mathbf{b}^*_{2,i}$'s as free parameters for $i = 1, 2, \ldots, N-1$ and express the $\mathbf{b}^*_{4,i}$'s according to continuity relations.

$$\text{(17)} \qquad \mathbf{b}^*_{4,i} := 2\,\mathbf{b}_{5,i} - \mathbf{I}_{i+1} + \frac{k_i^2}{k_{i+1}^2}\left(\mathbf{b}^*_{2,i+1} + \mathbf{I}_{i+1} - 2\,\mathbf{b}^*_{1,i+1}\right)$$

In order to maintain shape-preserving requirements the resulting control polygon should have the same convexity of the initial control polygon. More precisely the point $\mathbf{b}^*_{2,i}$ should belong to the triangle whose vertices are $\mathbf{b}^*_{1,i}$, $\mathbf{b}_{1,i}$ and $\mathbf{b}^*_{3,i}$, and equivalently that $\mathbf{b}^*_{4,i}$ should belong to the triangle given by $\mathbf{b}^*_{3,i}$, $\mathbf{b}_{2,i}$ and $\mathbf{b}^*_{5,i}$.

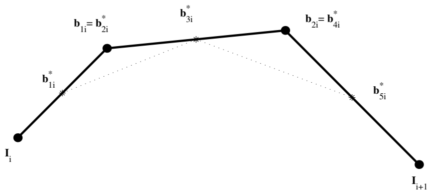


Figure 3: Starting control polygon for C^2 continuity.

Using AS formalization, these requirements lead to constraint domains belonging to $\mathbb{R}^2 \times \mathbb{R}^2$, therefore analogously to what we have done in the previous section, we approximate them with a union of hyperrectangles and take only the discrete values corresponding to the vertices.

Analogously to what we have done in the cubic case, we can now compute, for every segment i, the area bounded by the polygonal line and the curve and maximize it. This procedure lead to an *optimal* C^2 spline curve.

5. Numerical results

In this section we present the performance of the proposed scheme on three examples. In every case, in order to better comment the results obtained the curves will be depicted along with the curvature at the normal direction (*porcupine representation*),

The first example is about a symmetric set of data proposed the first time by Kaklis and Sapidis in [12]. The resulting curves are shown in Figures 4-5, where it is also displayed the polygonal line connecting the data. More precisely in Figure 4 it is shown the resulting C^1 shape-preserving curve which interpolates the given data, and maximize the bounded area.

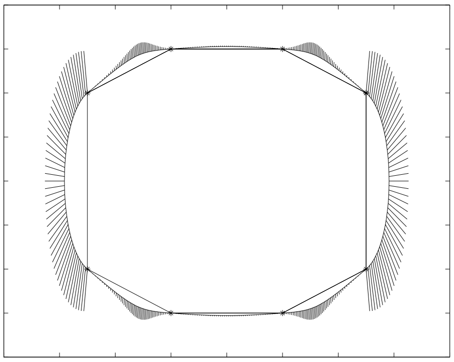


Figure 4: C^1 cubic spline curve.

In Figure 5 the spline curve with C^2 continuity is shown where, again the result is obtained though the maximization of the bounded area. We may note numerically that in this case, having performed a second optimization process the total area is increased.

The second example concerns a set of data taken from a ship hull. The results are displayed in Figure 6. Again we observe experimentally that the spline curve with C^2 continuity has a bounded area which is increased with respect to the area of the corresponding C^1 curve.

As a last example we consider a set of data with one inflection point (which is reproduced in the resulting spline curve). The two resulting curves are shown in Figure 7.

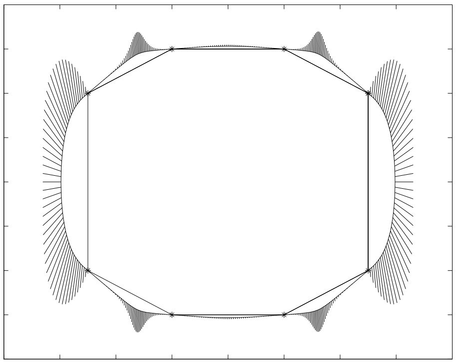
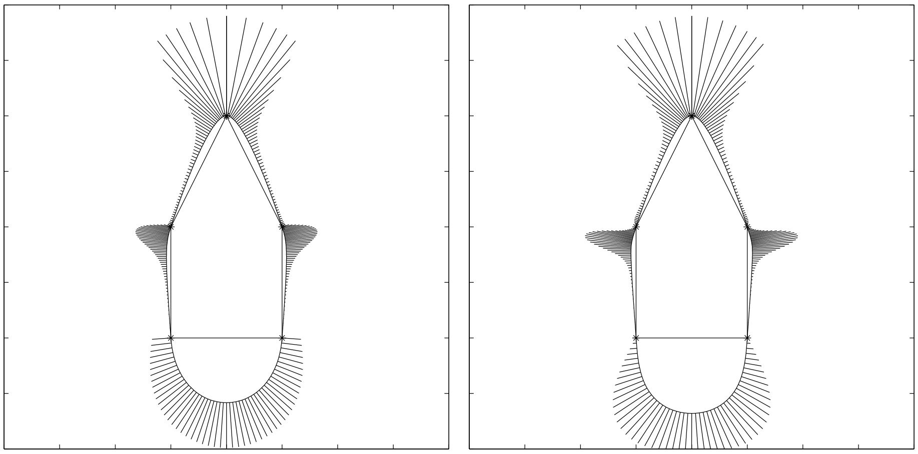
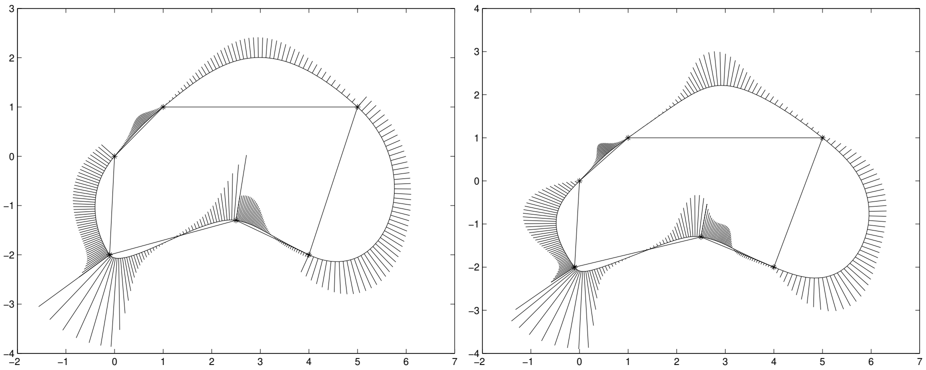


Figure 5: C^2 spline curve of degree 6.



(a) (b)

Figure 6: Example 2: (a) resulting cubic spline curve with C^1 continuity.(b) resulting spline of degree six with C^2 continuity.



(a) (b)

Figure 7: Example 3:(a) resulting cubic spline curve with C^1 continuity.(b) resulting spline of degree six with C^2 continuity.

References

[1] ASATURYAN S., COSTANTINI P. AND MANNI C., *G^2 Shape-preserving parametric planar curve interpolation*, in: "Designing and Creating Shape-Preserving Curves and Surfaces", (Eds. Nowacki H. and Kaklis P.), B.G. Teubner, Stuttgart 1998, 89–98.

[2] BELLMAN R. AND DREYFUS S., *Applied dynamic programming*, Princeton University Press, New York 1962.

[3] COSTANTINI P., *An algorithm for computing shape-preserving splines of arbitrary degree*, Journal of Computational and Applied Mathematics, **22** (1988).

[4] COSTANTINI P., *A general method for constrained curves with boundary conditions*, in: "Multivariate Approximation from CAGD to Wavelets", (Eds. Jetter K. and Utreras F.I.), World Scientific Publishing Co., Singapore 1993.

[5] COSTANTINI P., *Abstract schemes for functional shape-preserving interpolation*, in: "Advanced Course on FAIRSHAPE", (Eds. Nowacki H. and Kaklis P.), B.G. Teubner, Stuttgart 1996, 185–199.

[6] COSTANTINI P., *Boundary-valued shape-preserving interpolating splines*, ACM Transactions on Mathematical Software **23** 2 (1997), 229–251.

[7] COSTANTINI P. AND SAMPOLI M. L., *Abstract schemes and constrained curve interpolation*, in: "Designing and Creating Shape-Preserving Curves and Surfaces", (Eds. Nowacki H. and Kaklis P.), B.G. Teubner, Stuttgart 1998, 121–130.

[8] COSTANTINI P. AND SAMPOLI M. L., *A general scheme for shape preserving planar interpolating curves*, BIT **40** 4 (2003)

[9] COSTANTINI P. AND SAMPOLI M. L., *Constrained interpolation in $\mathbb{R}^3$ by abstract schemes*, in: "Curve and surface design: Saint-Malo 2002", (Eds. Lyche T., Mazure M.L. and Schumaker L.L.), Nashboro Press, Nashville 2003, 93–102.

[10] GOODMAN T. N. T. AND UNSWORTH K., *Shape preserving interpolation by parametrically defined curves*, SIAM J. Numer. Anal. **25** (1988), 1451–1465.

[11] HOSCHEK J. AND LASSER D., *Fundamentals of computer aided geometric design*, AK Peters Ltd., Wellesley 1993.

[12] KAKLIS P. D. AND SAPIDIS N. S., *Convexity preserving polynomial splines of non uniform degree*, Computer Aided Geometric Design **12** (1995), 1–26.

[13] MULANSKY B. AND SCHMIDT J. W., *Convex interval interpolation using three-term staircase algorithm*, Numerische Mathematik **82** (1999), 313–337.

[14] MULANSKY B. AND SCHMIDT J. W., *Composition based staircase algorithm and constrained interpolation with boundary conditions*, Numerische Mathematik **85** (2000), 387–408.

[15] PREPARATA F. P. AND SHAMOS M. I., *Computational geometry*, Springer-Verlag, Berlin, New York 1985.

[16] SCHMIDT J. W., *On shape-preserving spline interpolation: existence theorems and determination of optimal splines*, Approximation and Function Spaces **22** PWN-Polish Scientific Publishers, Warsaw 1989.

[17] SCHMIDT J. W., *Staircase algorithm and construction of convex spline interpolants up to the continuity C^3*, Computer Math. Appl. **31** (1996), 67–79.

AMS Subject Classification: 65D05, 65D17.

Maria Lucia SAMPOLI
Dipartimento di Scienze Matematiche ed Informatiche
Università di Siena
Via del Capitano 15
53100 Siena, ITALIA
e-mail: `sampoli@unisi.it`