Fairness Criteria for Algebraic Curves

Pavel Chalmovianský, Bert Jüttler

March 7, 2003

Abstract.
We develop methods for the variational design of algebraic curves. Our approach is based on truly geometric fairness criteria, such as the elastic bending energy. In addition, we take certain feasibility criteria for the algebraic curve segment into account. We describe a computational technique for the variational design of algebraic curves, using an SQP (sequential quadratic programming) – type method for constrained optimization. As demonstrated in this paper, the powerful techniques of variational design can be used not only for parametric representations, but also for curves in implicit form.

Keywords. Variational design, algebraic curves, implicit representation.

1 Introduction

For parametric curves and surfaces, the various techniques of *variational design* are now widely being used in geometric modeling [2, 5, 6]. These techniques have been highly successful in various applications, including scattered data fitting (e.g. for reverse engineering), where they help to generate high–quality curves and surfaces from measurement data (point clouds).

Implicitly defined curves and surfaces have several potential advantages. For instance, surface–surface intersections can easily be traced if one surface is given in implicit form, and the other surface by a parametric representation. Also, the algorithms for curve and surface fitting do not assume the existence of a parameterization of the data, which is often difficult to generate, in particular for more complex data sets (see [8, 9] and the references cited therein).

Variational design of curves and surfaces in *implicit* representation, however, has not attracted much attention so far. Similar to the parametric case, such techniques are needed in order to improve the quality of the results. For instance, low degree algebraic spline curves can be generated with the help of techniques for bivariate Hermite interpolation, using Clough–Tocher or Powell–Sabin elements. According to our numerical experiences, this process often introduces wiggles and oscillations, and fairing techniques are therefore needed.

Note, that level set methods often use geometric criteria for controlling the evolution of a curve (or surface) [12]. Typically, these techniques are based on the local behavior of a curve (e.g. depending on curvature). In addition it is often difficult to maintain the desired topology.

Based on globally defined, truly geometric fairness criteria, such as the elastic bending energy (cf. [3]), we discuss a method for the variational design of algebraic curves. In addition to the fairness measures, our objective function also takes certain feasibility criteria for the algebraic curve segment into account. We develop a computational technique for the variational design of algebraic curves, using an SQP–type method for constrained optimization.

The paper is organized as follows. After introducing some basic notations in the next section, we describe our computational approach to the minimization of our objective function (also taking feasibility criteria into account) in section 3. An application to algebraic curve design is then described in the section 4. Finally, we conclude this paper.

2 Fairness Functionals for Planar Curves

We recall the Bernstein-Bézier representation of algebraic curves, and discuss the bending energy of implicitly defined curve segments.

2.1 Preliminaries

Following the notations introduced in [11], we use the Bernstein-Bézier representation of the algebraic curve segment with respect to a fixed basis triangle $\triangle \mathbf{v}_0\mathbf{v}_1\mathbf{v}_2 \subset \mathbb{R}^2$. Let $\mathbf{u} = (u_0, u_1, u_2)$ with $|\mathbf{u}| = u_0 + u_1 + u_2 = 1$ be the barycentric coordinates of an arbitrary point $\mathbf{p} = (p_1, p_2) \in \mathbb{R}^2$ with respect to this triangle, i.e., $\mathbf{p} = u_0\mathbf{v}_0 + u_1\mathbf{v}_1 + u_2\mathbf{v}_2$ and

$$u_0 = u_0(\mathbf{p}) = \frac{[\mathbf{p}\,\mathbf{v}_1\mathbf{v}_2]}{[\mathbf{v}_0\mathbf{v}_1\mathbf{v}_2]},\ u_1 = u_1(\mathbf{p}) = \frac{[\mathbf{v}_0\mathbf{p}\,\mathbf{v}_2]}{[\mathbf{v}_0\mathbf{v}_1\mathbf{v}_2]},\ \text{and } u_2 = u_2(\mathbf{p}) = \frac{[\mathbf{v}_0\mathbf{v}_1\mathbf{p}]}{[\mathbf{v}_0\mathbf{v}_1\mathbf{v}_2]}. \tag{1}$$

where

$$[\mathbf{a}\,\mathbf{b}\,\mathbf{c}] = \det\begin{pmatrix} 1 & 1 & 1 \\ a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \end{pmatrix}. \tag{2}$$

Geometrically, these coordinates can be interpreted as the oriented areas of certain triangles, normalized by the oriented area of the domain triangle, see Figure 1. We use the triangle

$$\mathbf{v}_0 = (0,0)^\top,\ \mathbf{v}_1 = (1,0)^\top \quad \text{and} \quad \mathbf{v}_2 = (v_{21}, v_{22})^\top, \tag{3}$$

with $v_{21} \in \mathbb{R}$ and $v_{22} > 0$ for the examples generated in the following sections. Note that any triangle can be mapped to it by a similarity transformation. The solutions of the problem discussed below is invariant with respect to similarities of the plane.

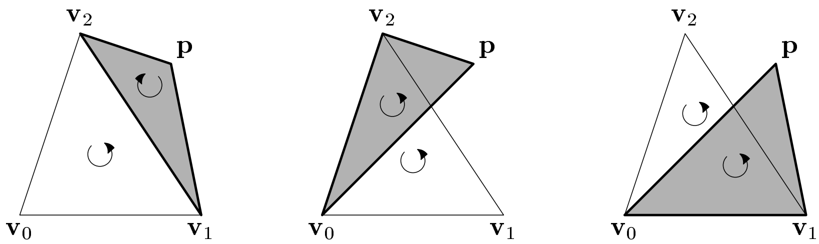

Figure 1: Barycentric coordinates of a point $\mathbf{p}$ with respect to the triangle $\triangle \mathbf{v}_0\mathbf{v}_1\mathbf{v}_2$.

The bivariate Bernstein-Bézier polynomials of degree n with respect to the basis triangle are defined as

$$B^n_{ijk}(u_0, u_1, u_2) = \frac{n!}{i!j!k!} u_0^i u_1^j u_2^k \quad \text{or, equivalently,} \quad B^n_{\mathbf{i}}(\mathbf{u}) = \binom{n}{\mathbf{i}} \mathbf{u}^{\mathbf{i}} \tag{4}$$

with the multi-indices $\mathbf{i} = (i, j, k) \in \mathbb{Z}^3_+$, $|\mathbf{i}| = i + j + k = n$.

Any bivariate polynomial $G = G(\mathbf{p})$ of degree n has a unique Bernstein-Bézier representation

$$G(\mathbf{p}) = G(u_0\mathbf{v}_0 + u_1\mathbf{v}_1 + u_2\mathbf{v}_2) = g(\mathbf{u}) = \sum_{\mathbf{i}\in\mathbb{Z}^3_+, |\mathbf{i}|=n} B^n_{\mathbf{i}}(\mathbf{u})\, b_{\mathbf{i}}, \tag{5}$$

with certain real coefficients $b_{\mathbf{i}} = b_{i,j,k}$, where the barycentric parameters $u_i = u_i(\mathbf{p})$ depend on $\mathbf{p}$, see (1).

Throughout this paper, we consider a segment of an algebraic curve of order n. This segment is a part of the zero contour of a polynomial G of degree n,

$$\tilde{\mathcal{C}} = \{\mathbf{p} \mid G(\mathbf{p}) = 0 \text{ and } \mathbf{p} \in D\} \tag{6}$$

where the domain D is the strip bounded by two parallels through $\mathbf{v}_0, \mathbf{v}_1$ with the direction of a suitable unit vector $\vec{\mathbf{r}} = (r_1, r_2)$. The end points of the curve segment are the vertices $\mathbf{v}_0$ and $\mathbf{v}_1$, and the lines $\overline{\mathbf{v}_0\mathbf{v}_2}$, $\overline{\mathbf{v}_1\mathbf{v}_2}$ of the domain triangle are assumed to be the tangents at these two points. As a necessary condition for the domain D and thus the vector $\vec{\mathbf{r}}$ is

$$\triangle \mathbf{v}_0\mathbf{v}_1\mathbf{v}_2 \subset D. \tag{7}$$

In order to obtain curves which are useful for geometric modelling, we assume that there exists a curve segment which has the shape of a single arc $\mathcal{C}$ connecting $\mathbf{v}_0$ and $\mathbf{v}_1$, without singularities along this arc, such that

$$\mathcal{C} \subseteq \tilde{\mathcal{C}}. \tag{8}$$

Suitable feasibility criteria, which are needed in order to achieve and to maintain this shape during the optimization, will be discussed in Section 3.

As a necessary condition for the end point and the end tangent property, the coefficients $b_{\mathbf{i}}$ satisfy

$$b_{n,0,0} = b_{n-1,0,1} = b_{0,n,0} = b_{0,n-1,1} = 0, \tag{9}$$

(see Figure 2, left). These conditions are also sufficient, provided that both segment end points are non-singular.

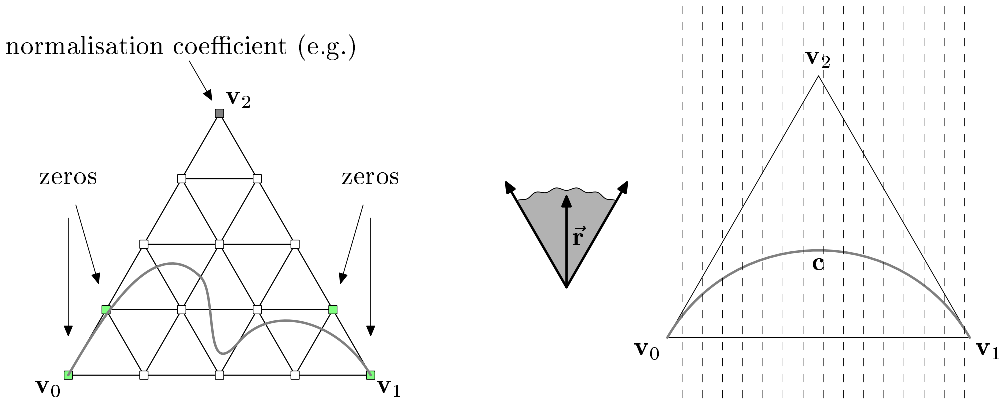

Figure 2: Left: boundary values of Bernstein-Bézier coefficients; right: parameterization of the segment as a graph of a function.

2.2 Parameterization

In order to evaluate the various quantities needed during the optimization process (bending energy and its derivatives), we need a parameterization of the curve segment $\mathcal{C}$. This parametric representation has the form $\mathbf{c}(t) = (x_1(t), x_2(t))$ for $t \in I$, where $I = [0, 1]$ is the parameter interval, and

$$G(\mathbf{c}(t)) = 0 \quad \text{for} \quad t \in I. \tag{10}$$

Clearly, such a parameterization is generally not available in closed form; only a numerical approximation can be given.

According to a well–known result from the theory of elastic curves, these curves can turn at most by angle of π (see e.g. [3]). Motivated by this fact, we assume that the curve segment $\mathcal{C}$ is a graph of a function, where the "y–axis" has an appropriately chosen direction $\vec{\mathbf{r}}$, such that it is contained in the wedge spanned by the vectors $\mathbf{v}_2 - \mathbf{v}_0$ and $\mathbf{v}_2 - \mathbf{v}_1$.[1] Under this assumption, the parameterization of the algebraic curve can be obtained by projecting it parallel to $\vec{\mathbf{r}}$ onto the edge $\overline{\mathbf{v}_0\mathbf{v}_1}$ of the basis triangle. Consequently, the parametric representation has the form

$$\mathbf{c}(t) = \begin{pmatrix} x_1(t) \\ x_2(t) \end{pmatrix} = \begin{pmatrix} t \\ 0 \end{pmatrix} + y(t)\vec{\mathbf{r}}, \quad t \in [0, 1], \tag{11}$$

where $y(t)$ is one of the n roots of the bivariate polynomial $G(\mathbf{p})$ along the line $(t, 0) + \lambda\vec{\mathbf{r}}$. Clearly, one has to pick the "right" root among the possibly n different ones; this issue will be addressed later. For the sake of further computation, we need the function $y(t)$. Since G is a polynomial and since we suppose that no singular points along the curve segment $\mathcal{C}$ exist, the function $y(t)$ is C^2 in $[0, 1]$.

[1] In our implementation, we choose this direction as $\vec{\mathbf{r}} = (0, 1)^\top$, see Figure 2, right. Of course, the calculation becomes slightly more complicated in the case where the coordinate system is not orthogonal.

The ordinate y of a point $\mathbf{c}(t)$ depends both on the abscissa t and on the Bernstein-Bézier coefficients

$$\boldsymbol{b} = (b_{\mathbf{i}})_{|\mathbf{i}|=n,\, \mathbf{i}\in\mathbb{Z}_+^3} \in \mathbb{R}^{d_n} \quad \text{where} \quad d_n = \binom{n+2}{2}. \tag{12}$$

During the optimization we will need the derivatives of $y = y(t, \boldsymbol{b})$ with respect to both quantities. The result of the optimization itself depends only on the control coefficients vector $\boldsymbol{b}$. In order to keep the notations simple, let

$$F(x_1(t,\boldsymbol{b}), x_2(t,\boldsymbol{b}), \boldsymbol{b}) = G(\mathbf{c}(t)) = 0, \tag{13}$$

see (5), (10), (11) and (12). Similarly,

$$F_i = \frac{\partial F}{\partial x_i} \text{ and } F_{ij} = \frac{\partial^2 F}{\partial x_i \partial x_j} \text{ for } i,j \in \{1,2\} \tag{14}$$

Lemma 2.1 *The derivatives of the abscissa $y = y(t, \boldsymbol{b})$ with respect to the parameter t and with respect to the coefficients in $\boldsymbol{b} = (b_{\mathbf{i}})_{|\mathbf{i}|=n,\, \mathbf{i}\in\mathbb{Z}_+^3}$ can be computed by differentiating* (13)*. All these derivatives exist, provided that F is sufficiently often differentiable, and $r_1 F_1 + F_2 \neq 0$.*

Proof: The proof follows from the implicit function theorem. The details can be seen in the following example. □

Example: Let $\mathbf{v}_0 = (0,0)^\top$, $\mathbf{v}_1 = (1,0)^\top$ and $\mathbf{v}_2 = (0,1)^\top$, and $\mathbf{c}(t) = (t, y(t))$, hence $u_0 = 1 - t - y$, $u_1 = t$ and $u_2 = y$. Consider a quadratic polynomial G satisfying the boundary conditions (9). Only two of the six Bernstein-Bézier coefficients are non–zero, and we get

$$F = B^2_{1,1,0}(u_0,u_1,u_2) b_{1,1,0} + B^2_{0,0,2}(u_0,u_1,u_2) b_{0,0,2} = 2(1-t-y)\, t\, b_{1,1,0} + y^2\, b_{0,0,2} \tag{15}$$

We are interested in the zero contour $F = 0$. Using the implicit function theorem, we compute the first derivatives of y with respect to the parameter t and with respect to one of the coefficients $b_{\mathbf{i}}$. Differentiating (13) leads to the equations

$$F_1 + F_2 \dot{y} = 0, \quad \text{and} \quad \frac{\partial F}{\partial b_{\mathbf{i}}} + F_2 \frac{\partial y}{\partial b_{\mathbf{i}}} = 0 \tag{16}$$

which can then easily be solved for $\dot{y} = (dy/dt)$ and $(\partial y / \partial b_{\mathbf{i}})$, provided that $F_2 \neq 0$.

The derivatives of higher order can be computed inductively in a similar way. Note that each differentiation with respect to one of the coefficients $b_{\mathbf{i}}$ or the curve parameter t increases the order of the required derivatives of the defining function F (resp. f) by one. Fortunately, all the second order partial derivatives of F with respect to $b_{\mathbf{i}}$ vanish.

2.3 Bending energy and objective function

We consider the bending energy

$$f_{\text{bend}}(\mathcal{C}) = \int_{\mathcal{C}} \kappa^2 \, \mathrm{d}s, \tag{17}$$

of a planar curve segment, where κ and s are the curvature and the arc length parameter. Clearly, this is a geometrically invariant fairness measure, independent of the parameterization. In the parametric case, it is very popular to use the linearized version instead [5, 6],

$$f_{\text{bend}}(\mathcal{C}) = \int_{\mathcal{C}} |\ddot{\mathbf{c}}|^2 \, \mathrm{d}t, \tag{18}$$

since it leads to simplified calculations. However, this functional is not geometrically invariant; it depends on the choice of the parameterization.

Since no generic parameterization is known in the algebraic case, and the use of the parameterization (11) does not simplify the problem, it is more appropriate to use the exact functional.

The curves minimizing (17) are called *elastica*. According to the rich literature on these curves (whose origins can be traced back to Leonhard Euler), there exists no unique minimizer to G^1 Hermite boundary data (two points with associated tangents), as curves with larger and larger loops can achieve arbitrarily small (non–negative) values of the functional. Instead, there exists a sequence of local minima with different lengths. See e.g. [7] for additional information.

The objective function takes the form

$$f(\boldsymbol{b}) = f_{\text{bend}}(\mathcal{C}) = \int_{\mathcal{C}} \kappa^2 \mathrm{d}s = \int_0^1 \frac{[\dot{\mathbf{c}}, \ddot{\mathbf{c}}]^2}{(\dot{\mathbf{c}})^{\frac{5}{2}}} \mathrm{d}t \tag{19}$$

where for $\vec{\mathbf{a}}_1 = (a_{11}, a_{12})^\top$, $\vec{\mathbf{a}}_2 = (a_{21}, a_{22})^\top$ we denote

$$[\vec{\mathbf{a}}_1, \vec{\mathbf{a}}_2] = \begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} \text{ and } (\vec{\mathbf{a}}_1) = \langle \vec{\mathbf{a}}_1, \vec{\mathbf{a}}_1 \rangle \tag{20}$$

We will need the first and second derivatives of f during the optimization process. They can be evaluated by applying the differentiation to the function under the integral in (19). Using (11) and

$$\frac{\partial \dot{\mathbf{c}}}{\partial b_{\mathbf{i}}} = \frac{\partial \dot{y}}{\partial b_{\mathbf{i}}} \vec{\mathbf{r}} \quad \text{and} \quad \frac{\partial \ddot{\mathbf{c}}}{\partial b_{\mathbf{i}}} = \frac{\partial \ddot{y}}{\partial b_{\mathbf{i}}} \vec{\mathbf{r}} \tag{21}$$

we get for the first derivative

$$\frac{\partial f}{\partial b_{\mathbf{i}}} = \int_0^1 \frac{r_2 \ddot{y}(t)}{(1+\dot{y}(t)^2)^{\frac{7}{2}}} \left(2 r_2 \ddot{y}(t)(1+\dot{y}(t)^2) - 5 r_2 \ddot{y}(t)(r_1 \frac{\partial \dot{y}(t)}{\partial b_{\mathbf{i}}} + \dot{y}(t) \frac{\partial \dot{y}(t)}{\partial b_{\mathbf{i}}} \|\vec{\mathbf{r}}\|) \right) \mathrm{d}t. \tag{22}$$

The second derivative of the objective function f can be computed similarly. For the sake of brevity, we omit the details.

In the case of a Cartesian coordinate system, $\vec{\mathbf{r}} = (0,1)^\top$, where $\mathbf{c}(t) = (t, y(t))$, $\dot{\mathbf{c}}(t) = (1, \dot{y}(t))$ and $\ddot{\mathbf{c}}(t) = (0, \ddot{y}(t))$, the formula (22) simplifies to

$$\frac{\partial f}{\partial b_{\mathbf{i}}} = \int_0^1 \frac{\ddot{y}(t)}{(1+\dot{y}(t)^2)^{\frac{7}{2}}} \left(2 \frac{\partial \ddot{y}(t)}{\partial b_{\mathbf{i}}} (1+\dot{y}(t)^2) - 5 \ddot{y}(t) \dot{y}(t) \frac{\partial \dot{y}(t)}{\partial b_{\mathbf{i}}} \right) \mathrm{d}t, \tag{23}$$

The second derivative of f with respect to coefficients is then equal to

$$\frac{\partial^2 f}{\partial b_{\mathbf{i}} \partial b_{\mathbf{j}}} = \int_0^1 \frac{1}{(1+\dot{y}(t)^2)^{\frac{9}{2}}} (z_1(t) + \ddot{y}(t)(1+\dot{y}(t)) z_2(t)) \mathrm{d}t \tag{24}$$

where

$$z_1(t) = \left(\frac{\partial \ddot{y}(t)}{\partial b_{\mathbf{i}}} (1+\dot{y}(t)^2) - 7\ddot{y}(t)\dot{y}(t) \frac{\partial \dot{y}(t)}{\partial b_{\mathbf{i}}} \right) \left(2\frac{\partial \ddot{y}(t)}{\partial b_{\mathbf{i}}} (1+\dot{y}(t)^2) - 5\ddot{y}(t)\dot{y}(t) \frac{\partial \dot{y}(t)}{\partial b_{\mathbf{i}}} \right) \tag{25}$$

and

$$\begin{aligned} z_2(t) \quad = \quad & \left(\frac{\partial^2 \ddot{y}(t)}{\partial b_{\mathbf{i}} \partial b_{\mathbf{j}}} (1+\dot{y}(t)^2) + 2\frac{\partial \ddot{y}(t)}{\partial b_{\mathbf{i}}} \dot{y}(t) \frac{\partial \dot{y}(t)}{\partial b_{\mathbf{j}}} \right) - \\ & -5 \left(\frac{\partial \ddot{y}(t)}{\partial b_{\mathbf{j}}} \dot{y}(t) \frac{\partial \dot{y}(t)}{\partial b_{\mathbf{i}}} + \ddot{y}(t) \frac{\partial \dot{y}(t)}{\partial b_{\mathbf{j}}} \frac{\partial \dot{y}(t)}{\partial b_{\mathbf{i}}} + \ddot{y}(t)\dot{y}(t) \frac{\partial^2 \dot{y}(t)}{\partial b_{\mathbf{i}} \partial b_{\mathbf{j}}} \right) \end{aligned} \tag{26}$$

For the sake of optimization, we map any polynomial to the coefficient space. Any polynomial (5) of degree n corresponds to a point $\boldsymbol{b} \in \mathbb{R}^{d_n}$ (see also (12)). Conversely, each point of $\boldsymbol{b} \in \mathbb{R}^{d_n} - \{\vec{\mathbf{0}}\}$ represents some homogeneous polynomial of degree n.

Further properties of the objective function f can be deduced from the boundary conditions for segment $\mathcal{C}$ and the fact that the function $G(\mathbf{p})$ is polynomial. Since four coefficients vanish (see Figure 2, left), it suffices to optimize just in the subspace of $\mathbb{R}^{d_n}$ given by (9). Moreover, the objective function is a homogeneous function of order 0, i.e.

$$f(k\boldsymbol{b}) = f(\boldsymbol{b}) \tag{27}$$

for all $k \neq 0$. More precisely, the objective function is geometrically invariant, i.e. it is not changed by multiplying the equation of the algebraic curve by non-zero k, since the zero contour does not change.

Summing up, we get

Lemma 2.2 *The objective function $f(\boldsymbol{b})$ has the following properties:*

1. *It is a homogeneous function of order 0 on the linear subspace of $\mathbb{R}^{d_n}$ characterized by (9), therefore its domain is isomorphic to the real projective space $\mathbb{RP}^{d_n-5}$.*

2. *The Taylor expansion of the objective function $f(\boldsymbol{b})$ with respect to the coefficients $b_{\mathbf{i}}$ can be computed by differentiating the right-hand side of (19) and using the results of the Lemma 2.1.*

Due to the homogeneity of the objective function, the coordinates of the point $\boldsymbol{b}$ (representing the algebraic curve) can be normalized by choosing some of its non-zero coordinates (see Figure 2, left).

3 Minimization of fairness functionals

Clearly, the minimum of a highly non-linear function cannot be calculated directly. Hence, we approximate the original function with a sequence of simpler functions, for which the minimum can be calculated. This leads to the sequential quadratic programming (SQP)-type method which is based on a local quadratic approximation.

We introduce the notion of the feasibility for the algebraic curve.

Definition 3.1 *An algebraic curve given by* (6) *is* feasible, *if the following three conditions are met*

(i) ***boundary conditions:*** *It satisfies the prescribed G^1 boundary conditions at the endpoints*

(ii) ***functionality:*** *The part of the curve between endpoints can be parameterized as in* (11) *using the chosen direction* $\vec{\mathbf{r}}$

(iii) ***regularity:*** *The parameterized part of the curve contains no singular point (in the sense of algebraic geometry).*

We assume that the initial algebraic curve for the optimization meets the feasibility requirements. Consequently, we have a segment $\mathbf{c}^0(t)$ (described by the Bernstein-Bézier coefficient vector $\boldsymbol{b}^0$ over the triangle), which is a graph of a function with respect to the chosen direction $\vec{\mathbf{r}}$. We are looking for the minimum of $f(\boldsymbol{b})$ with respect to the control points, such that the feasibility conditions are still satisfied.

The algorithm calculates a sequence of curves represented by coefficients

$$\boldsymbol{b}^0, \boldsymbol{b}^1, \boldsymbol{b}^2, \ldots, \boldsymbol{b}^h, \ldots \text{ for } h \in \mathbb{Z}_+, \tag{28}$$

such that the conditions of the Definition 3.1 are always fulfilled. Each curve segment can be parameterized by $\mathbf{c}^h(t)$, such that it is feasible in the above sense. Moreover, the values of the function f evaluated on the curves converges monotonically to a local minimum.

Algorithm 3.1 (Iteration step) The sequence (28) is generated by the following iteration:

1. Approximate the objective function $f(\boldsymbol{b})$ by a quadratic function $q^h(\boldsymbol{b})$ in the neighborhood of point $\boldsymbol{b}^h$.

2. Find the extremum $\boldsymbol{b}^*$ of the quadratic function $q^h(\boldsymbol{b})$.

3. Choose the step size $\tau^{h+1} \in \mathbb{R}$ and find the next point from

$$\boldsymbol{b}^{h+1} = (1 - \tau^{h+1})\boldsymbol{b}^h + \tau^{h+1}\boldsymbol{b}^* \tag{29}$$

   such that the feasibility is maintained.

The details of the steps in the minimization algorithm are described in the remainder of this section.

3.1 Approximation by quadratic functional and its minimization

We approximate the objective function f by the second order Taylor expansion with respect to the free coefficients (see Lemma 2.2) in a point $\boldsymbol{b}^h$ in order to compute the point $\boldsymbol{b}^{h+1}$,

$$q^h(\mathbf{x}) = f(\boldsymbol{b}^h) + \nabla f(\boldsymbol{b}^h)^\top \mathbf{x} + \frac{1}{2}\mathbf{x}^\top \nabla^2 f(\boldsymbol{b}^h)\mathbf{x}. \tag{30}$$

The extremum point $\boldsymbol{b}^*$ of the function q^h can be found by the solving of the following system of linear equations

$$\nabla f(\boldsymbol{b}^h) + \nabla^2 f(\boldsymbol{b}^h)\mathbf{x} = \vec{0} \tag{31}$$

Clearly, the solution of the system (31) depends on the regularity of the symmetric matrix $\nabla^2 f(\boldsymbol{b}^h)$, and the type of the extremum depends on the signs of the (real) eigenvalues of the matrix. If the extremum exists, we use it as an approximation $\boldsymbol{b}^{h+1}$ to the minimum

$$\boldsymbol{b}^{h+1} = \mathbf{x}. \tag{32}$$

The quadratic approximant q^h is positive definite in a sufficiently small neighborhood of a (local) minimum of the function f. Hence, the existence of a local minimum for the quadratic approximant q^h and its convergence toward the minimum of the function f is guaranteed within this neighborhood. In order to find $\boldsymbol{b}^{h+1}$, we need the first and the second derivative of the objective function, which can be evaluated using Lemma 2.2 (see also (23) and (24) for special $\vec{\mathbf{r}}$). Note, that this involves derivative of order four of $y(t, \boldsymbol{b})$. However, at most derivatives of order two with respect to t are needed!

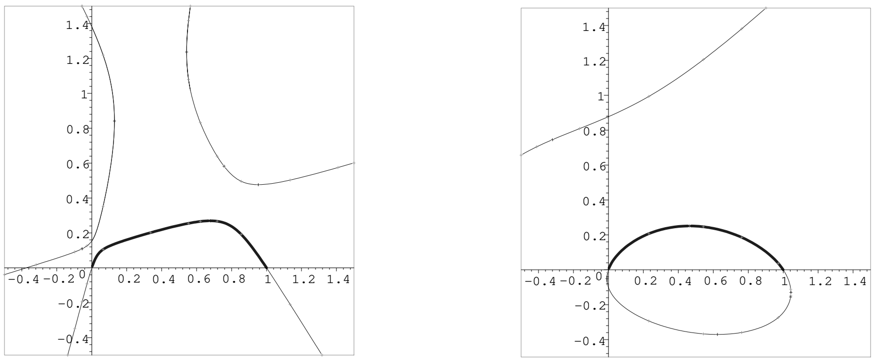

Figure 3: An optimization of a cubic curve. Bending energy: $f_{\text{bend}} = 9.18$ (left), $f_{\text{bend}} = 4.96$ (right)

As a first example, this fairing has been applied to the cubic algebraic curve which is shown in Figure 3. The value of the bending energy for the initial curve is equal to 9.18. After several iterations we arrived at the curve with the energy value 4.96. The values of the bending energy for the curves generated during the fairing can be seen in Figure 4.

We need to address several problems, which may occur during the optimization. First, the point $\boldsymbol{b}^*$ of the extremum calculated in (31) could be a local maximum or a saddle

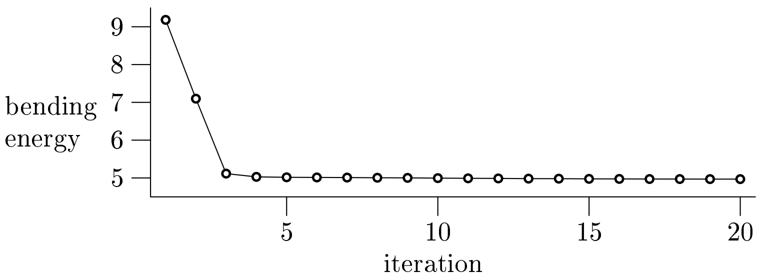

Figure 4: Convergence of the bending energy to minimum for example in Figure 3

point of the quadratic function q^h. Hence, the point $\boldsymbol{b}^h$ is "closer" to a local maximum (or a saddle point) of f than to a local minimum. We choose for the point $\boldsymbol{b}^{h+1}$ on the line

$$L_h = (1-\tau)\boldsymbol{b}^h + \tau\boldsymbol{b}^*, \tag{33}$$

as follows:

Algorithm 3.2

1. If the $f(\boldsymbol{b}^*) - f(\boldsymbol{b}) < 0$, we choose $\boldsymbol{b}^{h+1} = \boldsymbol{b}^*$ (most common case).
2. If $\langle \nabla f(b^h), \boldsymbol{b}^* - \boldsymbol{b}^h \rangle < 0$, find the point $\boldsymbol{b}^{h+1}$ on the segment $\boldsymbol{b}^h\boldsymbol{b}^*$ by bisection towards $\boldsymbol{b}^h$ (overshooting).
3. If the $\langle \nabla f(b^h), \boldsymbol{b}^* - \boldsymbol{b}^h \rangle > 0$, find the point $\boldsymbol{b}^{h+1}$ on the segment $\boldsymbol{b}^h\boldsymbol{b}_*$ by bisection, where $\boldsymbol{b}_* = 2\boldsymbol{b}^h - \boldsymbol{b}^*$ (change of the direction).
4. If $\langle \nabla f(b^h), \boldsymbol{b}^* - \boldsymbol{b}^h \rangle = 0$, we are either in minimum or in saddle point. Stop.

The situation of saddle point in case 4 is a pathological case and is usually difficult to deal with numerically. A more precise distinction can be done via calculation of the eigenvalues of the Hessian matrix $\nabla^2 f(\boldsymbol{b}^h)$. The convergence of the method may slow down in a neighborhood of a saddle point.

3.2 Penalty function removing singularities

Unfortunately, the solution of (31) often produces infeasible curves. The curve may not be a function in the direction $\vec{\mathbf{r}}$ (Figure 5, left) or it may have singularities (Figure 5, right).

For the sake of completeness we recall the definition of a singular point. A singular point on a planar algebraic curve $G(x_1, x_2) = 0$ is a point of the curve whose coordinates satisfy the following three equations

$$G(x_1, x_2) = 0, \quad G_1(x_1, x_2) = 0, \quad G_2(x_1, x_2) = 0 \tag{34}$$

where $G_i = (\partial/\partial x_i)G$. In a neighborhood of such a point one cannot generally guarantee that the curve can be written locally as a function[2] and we consider such curve segments

[2] There are examples of both cases, the curve is locally a function and the curve is not locally a function.

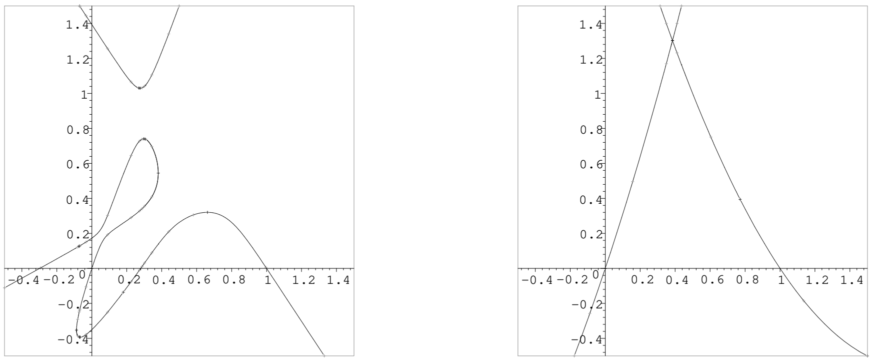

Figure 5: Infeasible algebraic curves fulfilling the endpoint constraints (left: functional assumption violated; right: singular point on the curve segment)

$\mathcal{C}$ as infeasible. Hence, we need a method for determining, whether the curve segment contains a singular points. There are available both symbolic and numerical approaches.

Since the exact symbolic methods (such as resultant methods, Gröbner basis methods, see for more details e.g. [4]) are expensive to compute, we use a penalty function. It keeps the solutions (28) away from the points in $\mathbb{RP}^{d_n-5}$, which represent curves with singularities along the considered arc $\mathcal{C}$. Clearly, the additional term in the objective function will cause an increase of the bending energy for the final curve.

To avoid points where both partial derivatives, G_1 and G_2 vanish, we have chosen the penalty function

$$f_{\text{penalty}}(\mathcal{C}) = \int_{\mathcal{C}} \frac{1}{P(c(s))^2} + (P(c(s)) - 1)^2 \mathrm{d}s, \tag{35}$$

where

$$P(x_1, x_2) = D_{\vec{\mathbf{n}}} G(x_1, x_2) \text{ and } \vec{\mathbf{n}} = \frac{\nabla G(x_1, x_2)}{\|\nabla G(x_1, x_2)\|}, \tag{36}$$

and $D_{\vec{\mathbf{n}}}$ is the directional derivative operator in the direction $\vec{\mathbf{n}}$. That is, we would like to keep the directional derivative of $G(\mathbf{p})$ in each point of the segment $\mathcal{C}$ in the normalized gradient direction far away from zero and close to 1.

A short calculation using (13) leads to

$$f_{\text{penalty}}(\mathcal{C}) = \int_{\mathcal{C}} \frac{1}{\|\nabla F(c(s))\|^2} + (\|\nabla F(c(s))\| - 1)^2 \mathrm{d}s \tag{37}$$

where

$$\nabla F(x_1, x_2, \boldsymbol{b}) = (F_1(x_1, x_2, \boldsymbol{b}), F_2(x_1, x_2, \boldsymbol{b}))^\top. \tag{38}$$

The evaluation of the (37) and its derivatives with respect to the coefficients $b_{\mathbf{i}}$ is similar to the evaluation of (19).

Lemma 3.1 *If the point $\boldsymbol{b}_0 \in \mathbb{RP}^{d_n-5}$ represents an algebraic curve $\mathcal{C}$ which satisfies the conditions* (i) *and* (ii) *of the Definition 3.1 with a singularity on $\mathcal{C}$, then the function $f_{\text{penalty}}(\mathcal{C})$ defined by* (37) *has a singularity in the point $\boldsymbol{b}_0$ and*

$$\lim_{\boldsymbol{b} \to \boldsymbol{b}_0} f_{\text{penalty}}(\boldsymbol{b}) = +\infty \tag{39}$$

Proof: Let $\{\mathbf{b}_i\}_{i=1}^{\infty} \subset \mathbb{RP}^{d_n-5}$ be a sequence of points representing regular algebraic curves as in (13), such that

$$\lim_{i\to\infty} \boldsymbol{b} = \boldsymbol{b}_0. \tag{40}$$

Let $\mathbf{c}_i(s)$ be the natural parameterizations of the curves in the neighborhood of $\mathbf{c}_i(0)$ and $\mathbf{c}_i(0) \to \mathbf{p}$, where $\mathbf{p}$ is a singularity on $\mathcal{C}$. Since the convergence of the curves $\mathbf{c}_i$ towards the singular limit curve $\mathcal{C}$ is uniform and ∇F is continuous, for some $\varepsilon > 0$

$$\lim_{i\to\infty} \int_0^{\varepsilon} \frac{1}{\|\nabla F(\mathbf{c}_i(s))\|^2} \mathrm{d}s = +\infty \tag{41}$$

and (39) follows easily. □

Consequently, the modification of the objective function leads to

$$f(\boldsymbol{b}) = f_{\text{bend}}(\boldsymbol{b}) + \alpha_{\text{penalty}} f_{\text{penalty}}(\boldsymbol{b}) \tag{42}$$

with a non-negative weight α_{penalty}. Clearly, the influence of the penalty function can be controlled by the weight. This is demonstrated by Figure 6. We started the optimization of the curve segment in top row. The three curve segments in the bottom row have been obtained after two steps of the iteration with weights $\alpha_{\text{penalty}} \in \{1.5, 5.0, 50.0\}$. Note the different shape of the level curves (algebraic offsets).

Notes for the implementation. We use Gaussian quadrature to evaluate the integrals during the minimization. We implemented quadrature rule which is exact for polynomials of degree 37. The method with such high order is being used since we get considerably good results via lower number of the evaluations of the functions under integral in (19), (23) and (24) when compared to some other integration methods.

The Gaussian quadrature is originally designed for highly differentiable functions. This is not the case if a singularity along the curve segment $\mathcal{C}$ occurs. Since the numerical value of the integral is computed just from a finitely many evaluations of f, a contribution of a singularity can be overlooked.

This problem can be addressed by the refining of the integration rule (usage of the higher order Gaussian quadrature or splitting the interval to more segments on which the rule is applied). Clearly, the change of the integration rule requires a higher number of evaluations. A cheaper solution which often improves the result (according to our numerical experience), is an increase of the weight α_{penalty}. Consequently, the bending energy becomes less important.

Singularities at the ends of the segment $\mathcal{C}$ may also cause problems with numerical integration, as the integration points (the roots of the Legendre polynomials) are in (a, b). We have therefore modified the rule for the integration of the penalty function with two additional terms evaluated at the endpoints, with very small weights. Their role is just to prevent the occurrence of singularities there. Hence, they must not influence the numerical value of the integral too much in a regular case. Typical additional weights in such terms, used in the examples (see e.g. Figure 9) were approximately in ratio 10^{-3} with the boundary weights of the Gaussian quadrature.

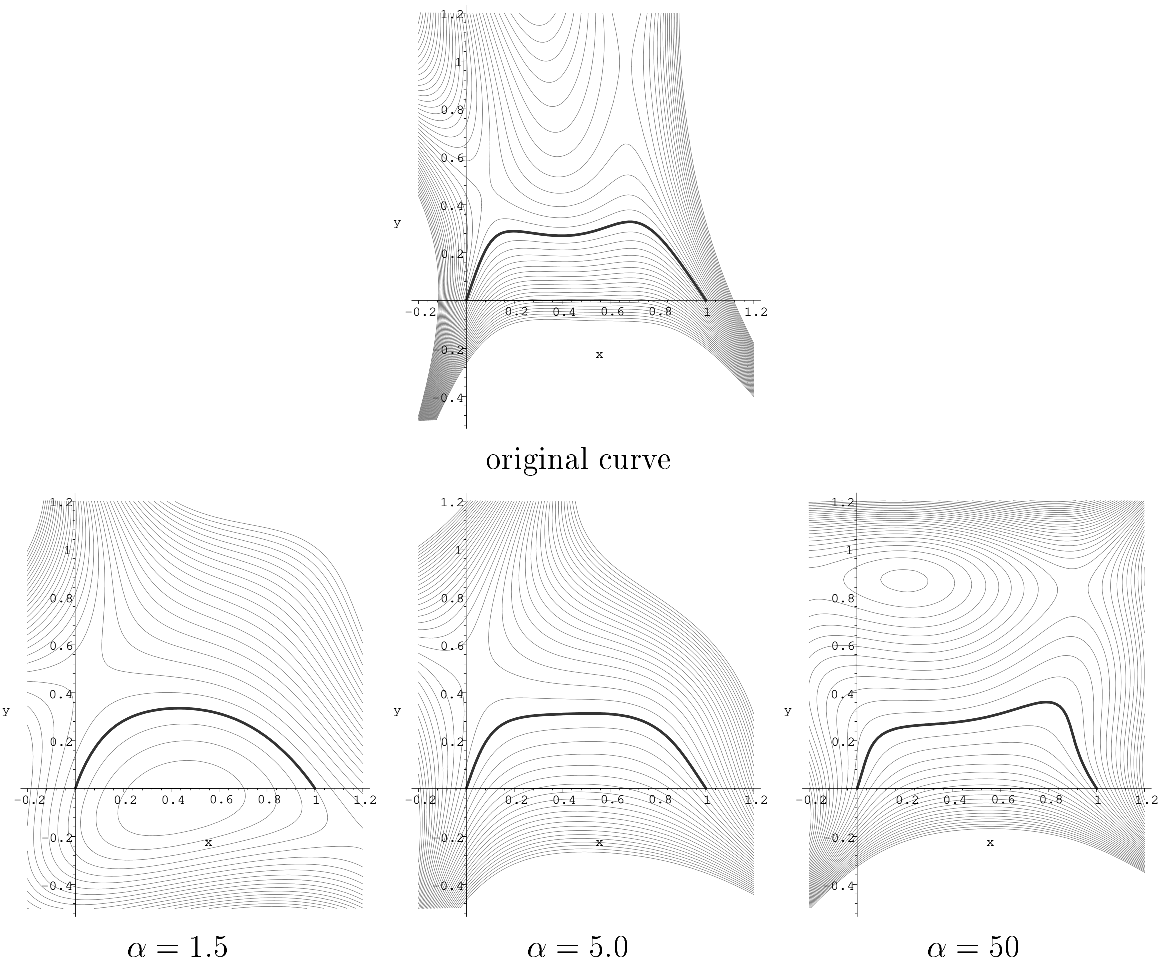

Figure 6: Influence of the weight $\alpha = \alpha_{\text{penalty}}$. Note that range and distance of the level curves (algebraic offsets) are identical.

3.3 Functional shape of the curve

As an another difficulty, the calculated curve might not fulfill the condition (ii) in the Definition 3.1 (see Figure 5, left). Hence, it is not a function with respect to the fixed direction $\vec{\mathbf{r}}$. Clearly, the set of all feasible curves (feasible domain) is subset in the $\mathbb{RP}^{d_n}$. Since the initial curve of every iteration step represented in (29) by the point $\boldsymbol{b}^h$ is supposed to be feasible, we "jumped" during the step 2 of Algorithm 3.1 out of the feasible domain.

We choose the next feasible curve $\boldsymbol{b}^{h+1}$ on the segment between $\boldsymbol{b}^h$ and the new estimated solution $\boldsymbol{b}^*$ (or $\boldsymbol{b}_*$) (see Section 3.1). Moreover, we require its objective function value

$$f(\boldsymbol{b}^{h+1}) < f(\boldsymbol{b}^h). \tag{43}$$

Since the initial curve does not have a tangent in the direction of $\vec{\mathbf{r}}$, there is a whole neighborhood of feasible curves around it. We look for a point representing such a curve, which is both feasible and has a lower value of the objective function (see Figure 7).

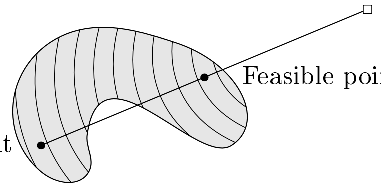

Figure 7: Seeking of improvements just in the feasible area (gray) of the objective function during the optimization

If we do not start in a local minimum of $f(\boldsymbol{b})$, there is a direction $\vec{\mathbf{d}}$ and a point $\boldsymbol{b}^h + \epsilon\vec{\mathbf{r}}$ for some $\epsilon > 0$ in a sufficiently small neighborhood of $\boldsymbol{b}^h$ in the direction $\vec{\mathbf{d}}$, such that

$$f(\boldsymbol{b}^{h+1} + \epsilon\vec{\mathbf{d}}) < f(\boldsymbol{b}^h). \tag{44}$$

and the point $\boldsymbol{b}^{h+1} + \epsilon\vec{\mathbf{d}}$ is feasible.

We find a feasible point on the line by halving the stepsize until a a feasible point is found. We need an efficient method for detecting the feasibility of a curve to finish the task.

Our method, which traces the curve, can be summarized as follows:

Algorithm 3.3 (Functionality Testing) Let $\bar{\mathbf{t}} = \{0 = t_0, t_1, \ldots, t_{q-1}, t_q = 1\} \subset I$, $q \in \mathbb{Z}_+$ be an increasing sequence of parameters. We would like to find the points of the curve segment $\mathcal{C}$ so that $\mathbf{p}_i = \mathbf{c}(t_i)$ for $i = 0, \ldots, q$. Let

$$l_r = (1 - t_r)\mathbf{v}_0 + t_r\mathbf{v}_1 + s\vec{\mathbf{r}} \tag{45}$$

be the line with the parameter $s \in \mathbb{R}$ and $0 \leq r \leq q$. The algorithm proceeds in several steps:

1. Start in the vertex $\mathbf{p}_0 = \mathbf{v}_0$.

2. Suppose the vertices $\mathbf{p}_0, \ldots, \mathbf{p}_r$ are found for some $r < q$. Using Taylor expansion of the curve $\mathcal{C}$ in the point $\mathbf{p}_r$, we determine the location of the point $\hat{\mathbf{p}}_{r+1}$, which is the point lying on the line l_{r+1} and Taylor expansion of certain order for the curve in point $\mathbf{p}_r$. We used the first and the second order approximation

$$\hat{\mathbf{c}}(h) = \mathbf{c}(t_r) + h\dot{\mathbf{c}}(t_r) \ \ \{+\frac{h^2}{2}\ddot{\mathbf{c}}(t_r)\} \tag{46}$$

(see Figure 8). Alternatively, one can also use more advanced Runge-Kutta-type methods.

3. Calculate the point $\mathbf{p}_{r+1}$ on the $\mathcal{C}$ lying on the line l_{r+1}, using Newton's method with the initial value $\hat{\mathbf{p}}_{r+1}$.

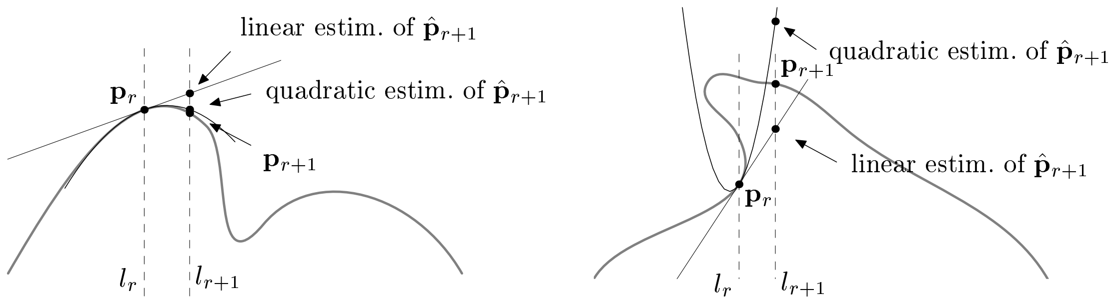

Figure 8: The tracing of the curve during the functionality detection. Left: the curve is a function in given direction. Right: the curve is not a function in the given direction.

4. Accept the point $\mathbf{p}_{r+1}$ in case it exists and is close to the estimation $\hat{\mathbf{p}}_{r+1}$ and continue with the step 2. Stop if the point $\mathbf{p}_{r+1}$ does not exist or was not accepted or if $r = q - 1$.

Since we use a penalty term in the objective function (42), we have no problems with singular points along the considered segment $\mathcal{C}$. Hence, we can use the Taylor expansion of the curve $\mathbf{c}(t)$ in the Step 2.

Clearly, the correct result of the algorithm depends on the suitable choice of the sequence $\bar{\mathbf{t}}$. We used those of the Gaussian quadrature integration for checking the feasibility, since the roots of the Legendre polynomial are quite regularly distributed along the interval I and their number was sufficient for our examples. If more points are needed, the sequence can be refined.

The example in the Figure 9 illustrates the fairing of a quartic algebraic curve. Initially, it was not possible to handle the curve by the method described in Section 3.1 (compare with Figure 5). We start to fair the curve at the Figure 9, left with bending energy $f_{\text{bend}} = 67.92$. One can see a result of fairing with weight $\alpha_{\text{penalty}} = 0.5$. The bending energy decreased to $f_{\text{bend}} = 4.1004$. The continuation with fairing after the change to $\alpha_{\text{penalty}} = 0.0$ makes just a small difference in shape (see Figure 9, right) and the value of bending energy $f_{\text{bend}} = 4.1000$.

4 Algebraic Curve Shaping

Variational modeling comprises the shape optimization of objects subject to geometric constraints. We use the numerical technique to solve the following problem: Find an algebraic curve segment, such that

1. it interpolates given points and tangents at them,

2. the curve is as fair as possible (cf. Section 2.3), without singularities,

3. the curve is pulled toward chosen fixed point $\mathbf{p}$.

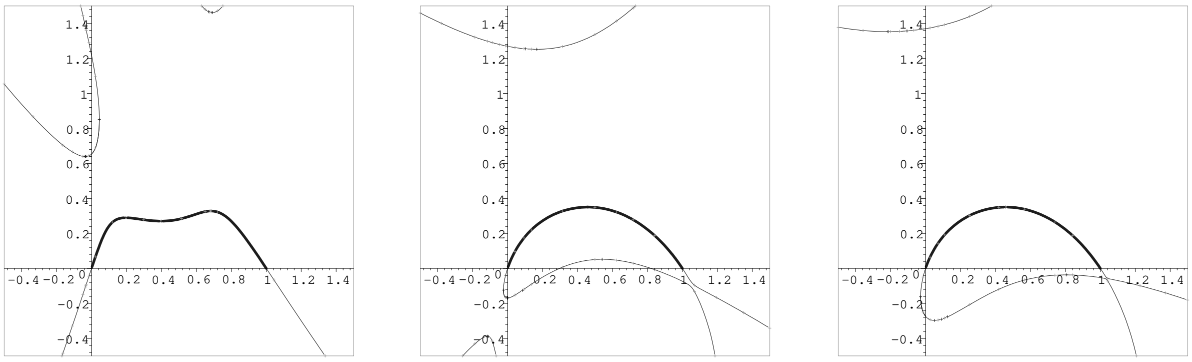

Figure 9: The fairing of the algebraic segment using penalty function (left: initial curve, $f_{\text{bend}} = 67.92$; middle: using penalty function with weight $\alpha_{\text{penalty}} = 0.5$, $f_{\text{bend}} = 4.1004$; right: further optimization of curve without penalty function with $\alpha_{\text{penalty}} = 0.0$ gives $f_{\text{bend}} = 4.1000$).

This leads to the optimization problem presented in Section 3, with an additional term responsible for the condition 3.

Let $\mathbf{p} = [\bar{x}_1, \bar{x}_2]^\top$. As a naive idea, the additional term of the objective function could be chosen an expression $G(\bar{x}_1, \bar{x}_2)^2$. This is called the algebraic distance of a point $\mathbf{p}$ from the curve $G = 0$. Unfortunately, this approach does not work in those cases, where another branch of the algebraic curve is present in a neighborhood of the point $\mathbf{p}$ (see e.g. Figure 12, right). Clearly, in this case the algebraic distance might be measured with respect to the wrong branch. Even though we start at the beginning with a point, which is close to the curve segment $\mathcal{C}$ and far from the other branches, this can change during the optimization. Summing up, the algebraic distance is not useful for our purpose.

As a more geometrically oriented approach we use, is the squared distance of $\mathbf{p}$ from the segment $\mathcal{C}$,

$$f_{\text{pull}}(\mathcal{C}) = \|(\mathbf{c}(s_{\mathbf{p}}) - \mathbf{p})\|^2 \tag{47}$$

where the point $\mathbf{c}(s_{\mathbf{p}})$ is the footpoint from the point $\mathbf{p}$ with the minimum distance property

$$\|\mathbf{c}(s_{\mathbf{p}}) - \mathbf{p}\| = \min_{\substack{s \in I \\ \mathbf{c}(s) \text{ is footpoint from } \mathbf{p}}} \|\mathbf{c}(s) - \mathbf{p}\|. \tag{48}$$

We assume, that the footpoint exists, otherwise we take either the closest point on the segment $\mathcal{C}$ or the point which forms a minimal angle of vectors $\mathbf{p} - \mathbf{c}(s)$ and the $\nabla G(c(s))$.

The footpoints from the point $\mathbf{p}$ to the implicit form of the curve $\mathcal{C}$ can be calculated by solving the following equations

$$G(x_1(s), x_2(s)) = 0 \tag{49}$$

$$\begin{vmatrix} G_1(x_1(s), x_2(s)) & G_2(x_1(s), x_2(s)) \\ x_1(s) - \bar{x}_1 & x_2(s) - \bar{x}_2 \end{vmatrix} = 0, \tag{50}$$

for $s \in I$. Clearly, it says the vectors $\mathbf{c}(s) - \mathbf{p}$ and $\nabla G(\mathbf{c}(s))$ are linearly dependent (see Figure 10). The point $\mathbf{c}(s)$ in such shortest vector $\mathbf{c}(s) - \mathbf{p}$ is taken as the point $\mathbf{c}(s_{\mathbf{p}})$.

The uniqueness of such a parameter value $s_{\mathbf{p}}$ is guaranteed only for the points $\mathbf{p}$ lying in a sufficiently small neighborhood of the segment $\mathcal{C}$.

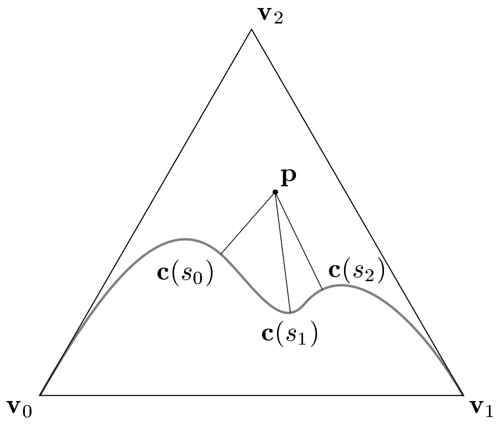

Figure 10: The footpoints of the point $\mathbf{p}$ on the curve segment $\mathcal{C}$

We can now proceed further in the spirit of the Section 3 and extend the objective function by a new term $f_{\text{pull}}(\mathcal{C})$ weighted with a non-negative weight α_{pull}. Its final form is

$$f(\boldsymbol{b}) = f_{\text{bend}}(\boldsymbol{b}) + \alpha_{\text{penalty}} f_{\text{penalty}}(\boldsymbol{b}) + \alpha_{\text{pull}} f_{\text{pull}}(\boldsymbol{b}) \tag{51}$$

with non-negative weights $\alpha_{\text{penalty}}, \alpha_{\text{pull}}$.

The calculation of the Taylor expansion for the term $f_{\text{pull}}(\boldsymbol{b})$ with respect to the coefficients $(b_{\mathbf{i}})_{|\mathbf{i}|=n,\, \mathbf{i}\in\mathbb{Z}_+^3}$ differs a little from those in Section 3, because the parameter $s_{\mathbf{p}}$ must be calculated in each iteration separately. Hence, the approximation of the $f_{\text{pull}}(\boldsymbol{b})$ for the SQP method consists of the following steps for each new curve in the sequence (28):

1. Calculate the closest footpoint of $\mathcal{C}$ from $\mathbf{p}$.

2. Compute the Taylor expansion of the objective function $f_{\text{pull}}(\boldsymbol{b})$ with respect to the coefficients.

The calculation of $\mathbf{c}(s_{\mathbf{p}})$ starts with the approximation of $\mathcal{C}$ granted by the Algorithm 3.3 for parameters set $\bar{\mathbf{t}}$. As an initial point for the computation we take the point $\mathbf{c}(t_m)$ for $m \in \{0, \ldots, q\}$, such that minimizes

$$\beta_{\text{dist}} \|\mathbf{p} - \mathbf{c}(t_r)\| + \beta_{\text{angle}} \left\langle \frac{\nabla G(\mathbf{c}(t_r))}{\|\nabla G(\mathbf{c}(t_r))\|}, \frac{\mathbf{p} - \mathbf{c}(t_r)}{\|\mathbf{p} - \mathbf{c}(t_r)\|} \right\rangle \tag{52}$$

where the non-negative real constants β_{dist} and β_{angle} must be chosen appropriately. The geometric meaning of the first term in (52) is the weighted distance of $\mathbf{p}$ from the vertices of the approximating polygon. The second term represents the weighted cosine of the angle between vectors $\mathbf{p} - \mathbf{c}(t_r)$ and $\nabla G(c(t_r))$ (see Figure 11). Combining both terms makes the choice of the starting point more stable.

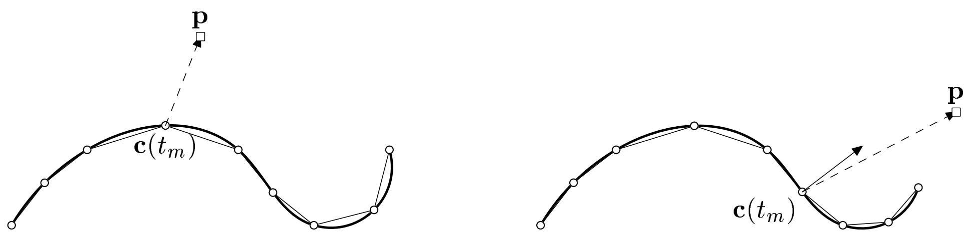

Figure 11: The discrete approximation of the footpoint (left: the smallest distance is the best rule in this case; right: the smallest angle between the normal direction of the curve point and the difference direction $\mathbf{p} - \mathbf{c}(t_r)$ determines better the initial value for footpoint location)

For the brevity, we use the following notation during the computation

$$\frac{\partial}{\partial b_{\mathbf{i}}} x_1 = x_{1|\mathbf{i}}, \quad \frac{\partial}{\partial b_{\mathbf{i}}} x_2 = x_{2|\mathbf{i}}, \quad \frac{\partial^2}{\partial b_{\mathbf{i}} \partial b_{\mathbf{j}}} x_1 = x_{1|\mathbf{ij}} \quad \text{and} \quad \frac{\partial^2}{\partial b_{\mathbf{i}} \partial b_{\mathbf{j}}} x_2 = x_{2|\mathbf{ij}} \tag{53}$$

for all $b_{\mathbf{i}} \in (b_{\mathbf{i}})_{|\mathbf{i}|=n, \mathbf{i} \in \mathbb{Z}_+^3}$. The coefficients in the second order Taylor expansion for the $f_{\text{pull}}(\boldsymbol{b})$ part of the objective function can be computed using

$$\begin{aligned} \Delta_1 &= x_1 - \bar{x}_1 & (54) \\ \Delta_2 &= x_2 - \bar{x}_2 & (55) \end{aligned}$$

as follows

$$\begin{aligned} (\nabla f_{\text{pull}})_{\mathbf{i}} &= \frac{\partial}{\partial b_{\mathbf{i}}} \|\mathbf{c}(s_{\mathbf{p}}) - \mathbf{p}\|^2 = 2(x_{1|\mathbf{i}} \Delta_1 + x_{2|\mathbf{i}} \Delta_2)(s_{\mathbf{p}}) & (56) \\ (\nabla^2 f_{\text{pull}})_{\mathbf{ij}} &= \frac{\partial^2}{\partial b_{\mathbf{i}} \partial b_{\mathbf{j}}} \|\mathbf{c}(s_{\mathbf{p}}) - \mathbf{p}\|^2 \\ &= 2(x_{1|\mathbf{ij}} \Delta_1 + x_{1|\mathbf{i}} x_{1|\mathbf{j}} + x_{2|\mathbf{ij}} \Delta_2 + x_{2|\mathbf{i}} x_{2|\mathbf{j}}) & (57) \end{aligned}$$

where all values are evaluated at the point $\mathbf{c}(s_{\mathbf{p}}) = (x_1(s_{\mathbf{p}}), x_2(s_{\mathbf{p}}))$. The first and the second derivatives of $\mathbf{c}(s_{\mathbf{p}})$ can be computed from the equations (49) and (50) using the differential operator $(\partial/\partial b_i)$ and solving obtained equations similarly as in the Lemma 2.1. Summarizing, we obtain the following formulas in matrix form

$$\begin{pmatrix} x_{1|\mathbf{i}} \\ x_{2|\mathbf{i}} \end{pmatrix} = \begin{pmatrix} F_1 & F_2 \\ \langle \nabla F_1, \Delta^\perp \rangle - F_2 & \langle \nabla F_2, \Delta^\perp \rangle + F_1 \end{pmatrix}^{-1} \begin{pmatrix} -F_{\mathbf{i}} \\ -F_{1|\mathbf{i}} \Delta_2 + F_{2|\mathbf{i}} \Delta_1 \end{pmatrix}, \tag{58}$$

where $\Delta^\perp = (\Delta_2, -\Delta_1)^\top$. Similarly, the second order derivatives in (57) can be computed.

Lemma 4.1 *The second order Taylor expansion of the term* (47) *at the footpoint* $\mathbf{c}(s_{\mathbf{p}})$ *can be calculated by differentiating* (49) *and* (50). *The closest footpoint either does not exist or is uniquely determined for* $\mathbf{p}$ *in a sufficiently small neighborhood of* $\mathcal{C}$.

The examples of such optimization for various points $\mathbf{p}$ can be seen in the Figure 12. The initial curve segment (see Figure 9,left) was pulled toward three different points $[0, 0.8]$ (left),$[\frac{1}{3}, 0.8]$ (middle) and $[0.6, 0.8]$ (right). The final bending energy resp. squared distance of the point $\mathbf{p}$ to the curve segment rich the values $f_{\text{bend}} = 10.04$, $f_{\text{pull}} = 0.06$ (left), $f_{\text{pull}} = 0.001$, $f_{\text{bend}} = 12.44$ (middle) and $f_{\text{bend}} = 8.44$, $f_{\text{pull}} = 0.18$ (right). The corresponding weights in the objective function (48) are $(\alpha_{\text{penalty}}, \alpha_{\text{pull}}) = (0.00001, 350)$ (left), $(\alpha_{\text{penalty}}, \alpha_{\text{pull}}) = (0.0001, 300)$ (middle) and $(\alpha_{\text{penalty}}, \alpha_{\text{pull}}) = (0.03, 80)$ (right).

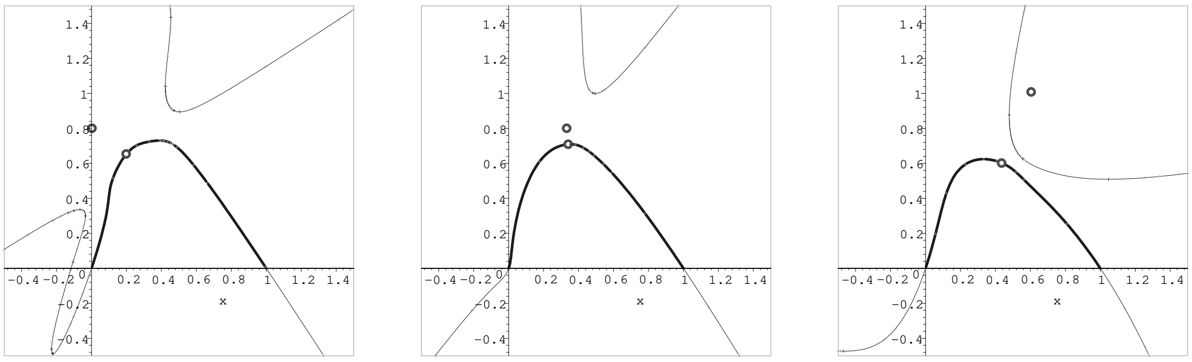

Figure 12: The pulling of the algebraic curve segment towards a point

Clearly, the results of the fairing process can be further controlled by the ratio of the parameters $1 : \alpha_{\text{penalty}} : \alpha_{\text{pull}}$ to balance the curvature and the closeness of the curve to the chosen point $\mathbf{p}$. We can try to pull the curve as close to the point $\mathbf{p}$ as possible with raising the weight α_{pull}. The bending energy usually raises considerably as the curve gets closer to the point $\mathbf{p}$.

Experiments with different weights in the objective function are shown in Figure 13. The same initial segment as in Figure 12 was used in all three cases. For the left figure, $(\alpha_{\text{penalty}}, \alpha_{\text{pull}}) = (0.05, 100)$, the square of the final distance increased to $f_{\text{pull}} = 0.06$ and the bending energy decreased to $f_{\text{bend}} = 9.73$. For the middle figure, $(\alpha_{\text{penalty}}, \alpha_{\text{pull}}) = (0.0001, 750)$, the final bending energy is $f_{\text{bend}} = 12.44$ and $f_{\text{pull}} = 0.001$. For the right figure, $(\alpha_{\text{penalty}}, \alpha_{\text{pull}}) = (0.0001, 1000)$, the final distance is $f_{\text{pull}} = 0.0006$ and $f_{\text{bend}} = 11.97$. As one may observe, the bending energy is lower than in the middle case, but the curvature changes the sign more often. According to our experience, the curvature might oscillate more due to the big weight of the pulling term.

5 Conclusions and Future Work

We have presented a method for variational design at algebraic curves. By applying our method to an algebraic curve, it is possible to get a fair algebraic curve segment without singular points and minimal bending energy. Moreover, the algorithm can be extended by additional tools, such as pulling, for designing algebraic segments. All tools are configurable by parameters with intuitive meaning. We showed, that feasibility can be guaranteed by a suitable objective function. The alternative approach would be to use discriminating families (see [1, 13, 14]).

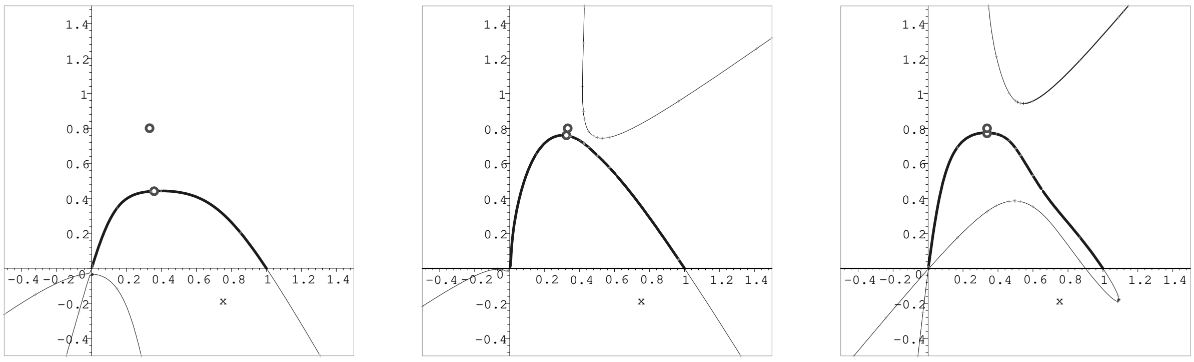

Figure 13: Adjusting the pulling weight in objective function

There are several possible extensions. Firstly, the method could be applied to algebraic splines (i.e. piecewise algebraic curves). In addition to applying fairing process to the polynomial pieces, one needs to find an appropriate way of minimizing curvature discontinuities, which occur – in the G^1 case – at the boundaries of the polynomial segments. This seems to be a non-trivial task.

Another challenging problem is the extension to the case of algebraic surfaces, where the minimization itself is less well understood. The existence of the minimum for Willmore-type energies for certain classes of surfaces is known (see e.g. [10]). In the surface case one has to deal with more complicated domains, which causes many technical difficulties.

References

[1] C.L. Bajaj and G. Xu. Regular algebraic curve segments (III) – applications in interactive design and data fitting. *Comput. Aided Geom. Des.*, 18(3):149–173, 2001.

[2] Guido Brunnett, Hans Hagen, and Paolo Santarelli. Variational design of curves and surfaces. *Surveys Math. Indust.*, 3(1):1–27, 1993.

[3] Guido H. Brunnett. Properties of minimal-energy splines. In *Hagen, Hans (ed.), Curve and surface design, SIAM, Philadelphia*, pages 3–22. 1992.

[4] David Cox, John Little, and Donal O'Shea. *Using algebraic geometry.* Springer New York, 1998.

[5] Günther Greiner. Surface construction based on variational principles. In *Laurent, Pierre-Jean (ed.) et al., Wavelets, images, and surface fitting. A K Peters Wellesley*, pages 277–286. 1994.

[6] H. Hagen. Variational principles in curve and surface design. In *Fisher, R. B. (ed.), Design and application of curves and surfaces. (Mathematics of surfaces V.) Clarendon Press Oxford* , pages 169–190. 1994.

[7] Emery Jou and Weimin Han. Minimal-energy splines with various end constraints. In *Hagen, Hans (ed.), Curve and surface design. SIAM Philadelphia*, pages 23–40. 1992.

[8] B. Jüttler. Least–squares fitting of algebraic spline curves via normal vector estimation. In R. Cipolla and R.R. Martin, editors, *The Mathematics of surfaces IX.*, pages 263–280. Clarendon Press, Oxford, 2000.

[9] Bert Jüttler and Alf Felis. Least-squares fitting of algebraic spline surfaces. *Adv. Comput. Math.*, 17(1-2):135–152, 2002.

[10] Andreas Kipp and Ulrich Reif. On the existence of solutions to non-parametric fairing problems. *J. Math. Anal. Appl.*, 238(2):540–550, 1999.

[11] Thomas W. Sederberg. Planar piecewise algebraic curves. *Comput. Aided Geom. Des.*, 1:241–255, 1984.

[12] Osher Stanley and Fedkiw Ronald. *Level Set Methods and Dynamic Implicit Surfaces.* Springer, Heidelberg, 2003.

[13] G. Xu, C.L. Bajaj, and C.I. Chu. Regular algebraic curve segments. II: Interpolation and approximation. *Comput. Aided Geom. Des.*, 17(6):503–519, 2000.

[14] G. Xu, C.L. Bajaj, and W. Xue. Regular algebraic curve segments. I: Definitions and characteristics. *Comput. Aided Geom. Des.*, 17(6):485–501, 2000.