Abstract

We have devised a new method for constructing discrete approximations to fair curves and surfaces by directly minimizing an arbitrarily selected fairness functional subject to geometric constraints. The nonlinear optimization problem is solved efficiently by a Sobolev gradient method. We first describe the method in general terms and then present results which demonstrate its effectiveness for constructing minimum variation curves which interpolate specified control points, tangent vectors, and/or curvature vectors.

Keywords: Discrete curve fairing; Surface fairing; Interpolation; Sobolev gradient

1. Introduction

The problem of constructing mathematical representations of fair curves and surfaces is central to the modeling, design, and manufacturing of automobile bodies, aircraft, ship hulls, industrial parts, and consumer products. It is also of critical importance in computer graphics, scientific visualization, medical imaging, and virtual environments.

A fair curve or surface is constructed by minimizing a functional, representing the fairness measure, subject to designer-specified geometric constraints on position, tangents, and/or curvatures. The representation of the curve or surface is chosen to have more than enough degrees of freedom to satisfy the constraints with the remaining freedom used to achieve the fairness objective. While fairness is subjective, and there are many choices for the functional (Roulier and Rando, 1994), excellent results are obtained by minimizing variation of curvature (Moreton and Séquin, 1994). This produces smooth shapes free of extraneous inflection, but the price is a high computational cost associated with solving a system of nonlinear equations (the Euler equations corresponding to the functional). In practice, therefore, shape...
quality is compromised by using quadratic functionals for which the Euler equations are linear. In order to achieve acceptable shapes, an expert designer must manipulate the curve or surface indirectly through time-consuming operations on a large number of control points.

We seek to develop efficient methods for minimizing complex fairness measures, such as those involving variation of curvature, and thus to produce a far more effective and less tedious design process. More precisely, we apply the theory of Sobolev gradients (Neuberger, 1997) to the minimization problem. Rather than storing and solving, or even formulating, a system of nonlinear equations, we treat the minimization problem directly by a gradient descent method, but in place of the standard gradient we use a Sobolev gradient that depends on the current approximation to the solution (thus resulting in a variable metric method). The Sobolev gradient method has been applied successfully to several difficult problems (Renka and Neuberger, 1995; Neuberger, 1997; Neuberger and Renka, 1998). Despite its proven success, however, the method is not widely known or understood, and we therefore provide an introduction to this powerful method in the next section.

Section 2 describes the Sobolev gradient method in general terms. In Section 3 we apply the method to the construction of minimum variation curves, and in Section 4 we present test results for the computational procedures described in Section 3. Section 5 concludes the paper with discussion and plans for future research.

2. Sobolev gradient method

We present a very informal and general description of the Sobolev gradient method. The problem is to minimize an energy functional $\phi$ such as the integral over some domain $\Omega$ (an interval, rectangle, etc.) of an expression involving a function $f$ from $\Omega$ to $\mathbb{R}^N$ and its derivatives up to order $k$ for some integers $N \geq 1$ and $k \geq 1$. The minimizing function $f$ is an element of some Sobolev space such as $H^{k,2}(\Omega, \mathbb{R}^N)$—the linear space of functions whose derivatives up to order $k$ are ‘square integrable’ (elements of $L^2(\Omega, \mathbb{R}^N)$). Note that $H^{k,2}(\Omega, \mathbb{R}^N)$ is a subspace of $L^2(\Omega, \mathbb{R}^N)$. Now consider a gradient descent method such as the method of steepest descent. At each step the current approximation $f$ to the minimizing function is updated by adding a multiple of the descent direction. The negative gradient of $\phi$ at $f$ is the direction which maximizes the decrease in $\phi$ per unit change in $f$. The key idea behind our method is that the gradient depends strongly on how ‘unit change in $f$’ is measured. The $L^2$ gradient results from the choice of $L^2(\Omega, \mathbb{R}^N)$ as the domain of $\phi$. This corresponds to the standard gradient (vector of partial derivatives) in the finite dimensional problem obtained by a discretization. However, the $L^2$ norm, involving only function values, is insensitive to rapid changes in the function, and allows a very rough gradient. A typical example is depicted in Fig. 3. As a result of this roughness, the standard method of steepest descent converges slowly if at all for functionals that involve derivatives. A nonlinear conjugate gradient method is not much better. On the other hand, using the Sobolev norm to measure the change in $f$ produces what Neuberger has termed the ‘Sobolev gradient’ which is quite smooth as shown in Fig. 4. The descent method with the discretized Sobolev gradient requires relatively few iterations even on problems for which the standard method fails completely.

The method by which the discretized Sobolev gradient is computed depends on the method of discretization. For finite differencing, it goes as follows. The Sobolev inner product is $\langle g, h \rangle_S = \langle Dg, Dh \rangle_{L^2(\Omega)}$ for some differential operator $D$ which maps elements of the Sobolev space $S$ to $L^2(\Omega)$ (or, more generally, $L^2(\Omega)^N$ for some $N \geq 1$). Typically we seek a function $f$ which satisfies some linear
constraints such as specified boundary conditions. (We can also treat nonlinear constraints.) We therefore start with an initial approximation which satisfies the constraints and restrict the gradient to the subspace \( S_0 \) of functions that satisfy homogeneous conditions, so that the updated approximation also satisfies the constraints. At the solution \( f \), the directional derivative (Fréchet derivative) of \( \phi \) in the direction \( h \) vanishes for all \( h \in S_0 : \phi'(f)h = 0 \ \forall \ h \in S_0 \). Now for all \( f \in S \), \( \phi'(f) \) is a bounded linear functional on \( S_0 \), and so by the Riesz Representation Theorem, there is a unique element \( \nabla_S \phi(f) \in S_0 \), termed the Sobolev gradient of \( \phi \) at \( f \), such that

\[
\phi'(f)h = \langle \nabla_S \phi(f), h \rangle \quad \forall \ h \in S_0 .
\]

Also, for \( f \) in a dense subspace of \( S \), \( \phi'(f) \) is bounded in the \( L^2 \) metric and hence uniquely represented by the \( L_2 \) gradient \( \nabla \phi(f) \in S_0 \):

\[
\phi'(f)h = \langle \nabla \phi(f), h \rangle_{L^2(\Omega)} \quad \forall \ h \in S_0 .
\]

We therefore have \( \langle \nabla \phi(f), h \rangle_{L^2(\Omega)} = \langle \nabla_S \phi(f), h \rangle_{S} = \langle (D^*D)\nabla_S \phi(f), h \rangle_{L^2(\Omega)} \quad \forall \ h \in S_0 \) and thus

\[
\nabla_S \phi(f) = (D^*D)^{-1}\nabla \phi(f) .
\]

While \( \nabla \phi \) is only densely defined, the above equation can be extended to all of \( S \) by continuity of \( \nabla_S \phi \).

At each descent step in the discretized problem a Sobolev gradient is computed from the standard gradient (list of partial derivatives) by solving a linear system in which the matrix corresponding to \( D^*D \) (the inverse of the smoothing operator) is symmetric, positive definite, and sparse. The system may be solved by an iterative method such as multigrid, so it is not necessary to explicitly store the matrix.

Note that the poor performance of standard descent methods for functionals involving derivatives reflects the lack of integrity in the function space setting. The \( L_2 \) gradient, being only densely defined, is everywhere discontinuous. Thus the discretized gradient not only lacks smoothness, but is sensitive to small changes in \( f \), and the sensitivity increases with the fineness of the mesh.

In (Renka and Neuberger, 1995), we applied the Sobolev gradient method to the problem of constructing minimal surfaces. The functional in the analogous minimum curve-length problem is

\[
\int s'(t) \, dt = \int \| f'(t) \| \, dt
\]

for a parametric function \( f \) with arc-length \( s \). We showed that the method produced the solution in a single iteration!

3. Discrete minimum variation curves

Given a sequence of control points \( p_0, p_1, \ldots, p_n \) in \( \mathbb{E}^3 \), along with unit tangent vectors \( t_j \) and normal curvature vectors \( K_j \), each specified at an arbitrary subset of the control points, we seek to construct a discrete approximation to a smooth (\( C^3 \) continuous) curve \( f \) which interpolates the control point data and minimizes the variation of curvature.

The free parameters defining \( f \) are computed by minimizing a discretization of the functional

\[
\phi(f) = \int \| K'(s) \|^2 \, ds = \int \| f''(s) \|^2 \, ds ,
\]

where \( s \) denotes arc length and \( K(s) = f''(s) \) is the curvature of \( f \) in the normal direction. Thus we minimize the variation of the normal curvature vector rather than that of the curvature \( \kappa = \| K \| \) which is not differentiable at points where it has zero value. In the case of an open curve, the natural end conditions are \( K(s) = K'(s) = 0 \) at \( s = 0 \) and \( s = L \) for curve length \( L \).
3.1. Discretization

The approximation to \( \mathbf{f} \) is taken to be the polygonal curve defined by a sequence of \( m+1 \) vertices \( f_i, \ i = 0, \ldots, m \). (Note, however, that the representation of \( \mathbf{f} \) also includes derivative approximations defined below.) Throughout the remainder of this paper, \( \mathbf{f} \) will denote the \((m+1)\)-vector of vertices as well as the underlying smooth curve that it approximates. The distinction will be clear from the context. We define an integer array of indexes \( \text{index} \) such that \( \text{index}(0) = 0 \), \( \text{index}(j) - \text{index}(j - 1) \geq 3 \) for \( j = 1, \ldots, n \), and \( \text{index}(n) = m \). Then the control point interpolation conditions are \( f_i = p_j \) for \( i = \text{index}(j) \) and \( j = 0, \ldots, n \). This may be thought of as a parameterization in which the knots (parameter values associated with the control points) are the integers \( t_j = \text{index}(j) \), and the vertices are function values \( f_i = f(i) \). The array \( \text{index} \) is chosen so that the knot interval length (number of vertices between each pair of adjacent control points \( p_{j-1} \) and \( p_j \)) is approximately proportional to the chord length \( \| p_j - p_{j-1} \| \) (chord length parameterization) or to its square root (centripetal parameterization), but is bounded below by 3.

We denote segment lengths by \( \Delta s_i \), midpoint unit tangent vectors by \( \Delta \mathbf{f}_i \), vertex normal curvature vectors by \( \Delta^2 \mathbf{f}_i \), and midpoint curvature derivative vectors by \( \Delta^3 \mathbf{f}_i \), where ‘midpoint’ vectors are derivative approximations at \( t = i - 0.5 \):

\[
\begin{align*}
\Delta s_i &= \| f_i - f_{i-1} \| \quad (i = 1, \ldots, m), \\
\Delta \mathbf{f}_i &= \frac{f_i - f_{i-1}}{\Delta s_i} \quad (i = 1, \ldots, m), \\
\Delta^2 \mathbf{f}_i &= \frac{\Delta \mathbf{f}_{i+1} - \Delta \mathbf{f}_i}{(\Delta s_i + \Delta s_{i+1})/2} \quad (i = 1, \ldots, m-1), \\
\Delta^3 \mathbf{f}_i &= \frac{\Delta^2 \mathbf{f}_i - \Delta^2 \mathbf{f}_{i-1}}{\Delta s_i} \quad (i = 1, \ldots, m).
\end{align*}
\]

Note that \( \Delta^3 \mathbf{f}_1 \) and \( \Delta^3 \mathbf{f}_m \) require values for \( \Delta^2 \mathbf{f}_0 \) and \( \Delta^2 \mathbf{f}_{m-1} \). In the case of an open curve, these are taken to be \( \Delta^2 \mathbf{f}_1 \) and \( \Delta^2 \mathbf{f}_{m-1} \), respectively, resulting in the natural end conditions \( \Delta^3 \mathbf{f}_1 = \Delta^3 \mathbf{f}_{m-1} = 0 \). In the case of a closed curve with \( p_0 = p_n \), we have \( \mathbf{f}_0 = \mathbf{f}_n \), and we extend the above definitions with \( \Delta s_0 = \Delta s_m \), \( \Delta \mathbf{f}_0 = \Delta \mathbf{f}_n \), and \( \Delta^2 \mathbf{f}_0 = \Delta^2 \mathbf{f}_n = \frac{\Delta \mathbf{f}_1 - \Delta \mathbf{f}_m}{(\Delta s_0 + \Delta s_1)/2} \).

The integral defining variation of curvature is approximated by the composite rectangle (midpoint) rule giving the discretized functional

\[
\phi(\mathbf{f}) = \sum_{i=1}^{m} \| \Delta^3 \mathbf{f}_i \|^2 \Delta s_i.
\]

Note that an alternative representation of the curve in terms of control points and basis functions would result in a far more complicated expression for \( \phi \) as an integral over some parameter range, and the integral would have to be discretized by a quadrature rule. The choice of curve representation is further discussed in Section 5.

3.2. Curve length penalty

A disadvantage of minimizing the variation of curvature, rather than a parameterization-dependent quadratic functional, is the lack of a unique solution and solutions with unbounded curve length. We
found that for some data sets, the computed solution exhibited inappropriate loops or circular arcs with radii which increased at each iteration. To avoid this problem we penalize total curve length by modifying the functional as follows:

\[ \phi(f) = \sum_{i=1}^{m} \frac{\|\Delta^3 f_i\|^2}{\Delta s_i} + w \sum_{i=1}^{m} (\Delta s_i)^2, \]  

where \( w \) is a nonnegative weight. Note that, for parameter \( t \), the penalty term approximates \( \int \| f'(t) \|^2 \, dt \), while the total curve length is actually \( \int \| f'(t) \| \, dt \). The Euler equations are \( f''(t) = 0 \) and \( \frac{d}{dr} f = 0 \), respectively, both corresponding to minimum length but with uniform parameterization only in the former case. We found that using this alternative to the actual curve length for the penalty term was necessary to avoid very poorly distributed vertices with some data sets. We employed an analogous technique to compute uniformly parameterized discrete minimal surfaces in (Renka and Neuberger, 1995).

We solve a sequence of minimization problems associated with decreasing weights, each solution serving as initial estimate for the next problem, and finally obtain a stable solution with \( w = w_0 \geq 0 \).

In addition to solving the problem of avoiding large loops, the penalty term has the added benefit that a nonzero value of \( w_0 \) serves as a tension factor which may be used to control the shape of the curve. As \( w_0 \) increases, the curve approaches the piecewise linear interpolant of the control points.

### 3.3. Method

The functional \( \phi \) is minimized subject to the interpolation constraints by a gradient descent method using the Sobolev gradient defined below. Even the disreputable method of steepest descent is effective with this gradient, but a nonlinear conjugate gradient method is faster, with a typical speedup factor of about three. Our code and test results are based on the Polak–Ribiere variant of the Fletcher–Reeves conjugate gradient method (Polak, 1971). However, we have recently begun experimenting with the Barzilai and Borwein method (Raydan, 1997).

The steepest descent iteration would be as follows:

\[ f_{n+1} = f_n - \alpha_n \nabla_S \phi(f_n), \]

where the initial estimate \( f_0 \) is computed by the method described in Section 3.3.1, the discretized Sobolev gradient \( \nabla_S \phi(f_n) \) is computed from the standard gradient \( \nabla \phi(f_n) \) by solving the linear system described in Section 3.3.2, and the step-size \( \alpha_n \) is chosen to minimize \( \phi(f_{n+1}) \)—a line search. The iteration is terminated when the relative change in \( \phi \) falls below a tolerance.

### 3.3.1. Initial estimate

An initial estimate for \( f \) which satisfies the interpolation conditions is constructed by a method similar to that of Moreton and Séquin (1994). First, the unspecified unit tangent vectors and normal curvature vectors are selected by heuristic methods. Our code treats numerous special cases including a curve with only two control points, a closed curve, and a user-specified curvature vector with value 0. We do not enumerate all the cases, but the basic ideas are as follows. If only a (nonzero) curvature vector is user-specified, the tangent direction is taken to be orthogonal to that in the plane of the curvature vector and an incident chord direction of the control polygon. If neither the tangent nor the curvature vector is specified at an interior knot, the tangent direction is taken to be the inverse-distance weighted average of the incident chord directions, and the curvature vector is taken to be orthogonal to the tangent vector with
magnitude equal to the reciprocal of the radius of the circumcircle of the corresponding control point and its two neighbors. In the case of an endpoint of an open curve (with more than two control points), the tangent direction is obtained by reflecting the tangent vector at the other end of the chord in the line defined by the chord.

The second step is to compute the vertices \( f_i \), in each knot interval as values of the Hermite quintic polynomial interpolant of the endpoint values (control points) and first and second derivative vectors obtained from the unit tangent vectors and normal curvature vectors, respectively, by assuming \( s'(t) \) is constant over the interval for arc length \( s \) and parameter \( t \). The final step is to adjust the computed vertices, if necessary, to exactly satisfy the tangent and curvature constraints. This requires projecting two neighboring vertices of the corresponding control point onto the nearest pair of points that satisfy the constraint.

The crucial property of the initial estimate is that it satisfy the interpolation constraints. Unlike other fairing methods based on minimizing nonlinear functionals, the initial estimate need not be close to a critical point in order to ensure convergence. For minimization problems with a unique solution, our experience with the Sobolev gradient method is that, even with a very poor initial estimate, a small number of descent steps results in a fairly good approximation. The problem treated here, however, has multiple critical points, and we found it necessary to start with a reasonable initial estimate.

3.3.2. Linear spaces, inner products, and gradients

Denote by \( S_0 \) the set of perturbations for \( f \) which preserve the interpolation conditions. This is a subspace of the \((m+1)\)-vectors of vertices \( h \in (\mathbb{R}^3)^{m+1} \) with the following properties: for \( j = 0, \ldots, n \) and \( i = \text{index}(j) \), \( h_i = 0; -h_{i-1} = h_{i+1} = \alpha t_j \) for scalar \( \alpha \) if only a tangent vector \( t_j \) is specified at control point \( p_j \); and \( h_{i-1} = h_{i+1} = 0 \) if a curvature vector is specified at \( p_j \), where \( h_{i-1} = h_{m-1} \) and \( h_{m+1} = h_1 \) in the case of a closed curve. An endpoint of an open curve has tangent defined by the first or last two vertices and curvature vector defined by the first or last three vertices.

Denote the orthogonal projection onto \( S_0 \) by \( P_0 : (\mathbb{R}^3)^{m+1} \rightarrow S_0 \). \( P_0 \) is applied by simply zeroing the appropriate elements in the case of control points and curvature vectors. In the case of a control point \( p_j \) with specified unit tangent vector \( t_j \) only, \( P_0 \) is applied to \( h \) by setting \( h_{i-1} \) and \( h_{i+1} \) to the nearest pair of vectors which sum to 0 and have direction \( t_j \), where \( i = \text{index}(j) \).

The Sobolev inner product associated with a curve (the current approximation to the solution) \( f \) is

\[
\langle g, h \rangle_f = \int (g''(s), h''(s)) \, ds + w \int (g'(s), h'(s)) \, ds,
\]

where \( s \) is the arc length associated with \( f \). Note that this is positive (defines an inner product) on functions with three or more zeros (and square integrable third derivatives), and is intrinsic to the curve (independent of the parameterization). The discretized inner product on \( S_0 \) is

\[
\langle g, h \rangle_f = \sum_{i=1}^{m} \left( \Delta^3 g_i, \Delta^3 h_i \right) + w \left( \Delta g_i, \Delta h_i \right) \Delta s_i = \langle Dg, Dh \rangle_{2m},
\]

where

\[
D = \begin{pmatrix} D_3 \\ D_1 \end{pmatrix} : (\mathbb{R}^3)^{m+1} \rightarrow (\mathbb{R}^3)^{2m}
\]

is the discrete differential operator defined by

\[
D_3 g_i = \sqrt{\Delta s_i} \Delta^3 g_i, \quad D_1 g_i = \sqrt{w \Delta s_i} \Delta g_i,
\]
and, for \( r, s \in (\mathbb{R}^3)^m \), the discretized \( L_2 \) inner product is

\[
\langle r, s \rangle_m = \sum_{i=1}^m \langle r_i, s_i \rangle.
\]

Note that the Sobolev inner product depends on the current value of \( f \), resulting in a variable metric method. Also (with actual curve length as the penalty term) we have, formally, \( \phi(f) = \langle f, f \rangle_f \). This has the misleading appearance of a quadratic form but, of course, \( D \) depends nonlinearly on \( f \). The equation reflects the fact that the inner product \( \langle \cdot, \cdot \rangle_f \) is defined by the same differential operator \( D \) used to define \( \phi(f) \). This need not be the case. It is sufficient that the operators involve the same order of differentiation.

Now, for \( g, h \in S_0 \),

\[
\langle g, h \rangle_f = \langle Dg, Dh \rangle_{2m} = \langle D^T Dg, h \rangle_{m+1} = \langle D^T Dg, P_0 h \rangle_{m+1} = \langle P_0 D^T Dg, h \rangle_{S_0}
\]

since \( P_0 h = h \) for \( h \in S_0 \) and \( P_0^T = P_0 \). Note that the adjoint and transpose of \( D \) are related by \( D^* = P_0 D^T \). Now let \( g \) be the Sobolev gradient of \( \phi \) at \( f \). Then by the Rietz Representation Theorem,

\[
\phi'(f) h = \langle \nabla \phi(f), h \rangle_{S_0} = \langle g, h \rangle_f = \langle P_0 D^T Dg, h \rangle_{S_0}, \quad \forall h \in S_0,
\]

so that the Sobolev gradient \( g \) is defined by

\[
g = (P_0 D^T D|_{S_0})^{-1} \nabla \phi(f),
\]

where \( |_{S_0} \) denotes the restriction to elements of \( S_0 \). An expression for the discretized \( L_2 \) gradient \( \nabla \phi(f) \) is obtained by simply differentiating the expression for \( \phi \).

In constructing the 6th order operator \( D^T D \), \( D \) is slightly modified by excluding the first and last rows of \( D_3 \) (which would produce a non-banded zero structure in the case of a closed curve). Thus, \( D \) is represented by a \( 2m - 2 \) by \( m + 1 \) matrix which is applied to each of the three components of an element of \( S_0 \). Note also that \( D \) has at most four nonzeros per row, and \( D^T D \) is a 7-diagonal symmetric positive semi-definite matrix of order \( m + 1 \), but its restriction to \( S_0 \) is a positive definite matrix with order \( k \leq m - n \). A straightforward approach to treating the order-\( k \) system would involve additional arrays with the zero entries omitted. We avoid this additional storage and complexity as follows. The rows and columns of \( D^T D \) associated with interpolation constraints are replaced by those of the identity matrix scaled by the original diagonal entry (in order to preserve the scaling of \( D^T D \)). The first and last rows and columns are omitted, and the resulting order-\( (m - 1) \) system is solved by a direct method (using an \( RR^T \) factorization, with upper triangular matrix \( R \), and a pair of triangular solve steps for each of the three solution components). It is only necessary to store the nonzero entries in the upper triangle of \( D^T D \) resulting in a storage requirement of less than \( 4m \) floating-point numbers. An alternative, using even less storage, would be a conjugate gradient method requiring only matrix-vector products: at each iteration \( D^T D \) and \( P_0 \) are applied to the current approximation. Note that it is necessary to apply the projection \( P_0 \) at each iteration.

4. Test results

We present results for four data sets. In all cases we use about 200 vertices with chord-length parameterization for the knot distribution. Convergence is defined by an upper bound of \( 10^{-4} \) on the
Table 1

<table>
<thead>
<tr>
<th>Data set</th>
<th>$n$</th>
<th>$m$</th>
<th>NS</th>
<th>$\phi$</th>
<th>$L$</th>
<th>$\int \kappa^2$</th>
<th>$|\kappa|_{\infty}$</th>
<th>$\int \tau^2$</th>
<th>$|\tau|_{\infty}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>6</td>
<td>198</td>
<td>7</td>
<td>3.12</td>
<td>8.365</td>
<td>2.747</td>
<td>0.9927</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>200</td>
<td>9</td>
<td>6.28</td>
<td>6.284</td>
<td>6.282</td>
<td>1.004</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>11</td>
<td>199</td>
<td>7</td>
<td>2.07</td>
<td>17.77</td>
<td>4.383</td>
<td>0.5148</td>
<td>3.949</td>
<td>0.6372</td>
</tr>
<tr>
<td>4</td>
<td>12</td>
<td>200</td>
<td>0</td>
<td>25.18</td>
<td>42.62</td>
<td>10.28</td>
<td>1.396</td>
<td>34.02</td>
<td>4.683</td>
</tr>
</tbody>
</table>

Our first test data set, borrowed from (Woodford, 1969), represents an open planar curve with no constraints on tangents or curvatures. The initial estimate and minimum variation curve are depicted in Figs. 1 and 2 with the control points marked by X’s.

Figs. 3 and 4 display the $L^2$ gradient and Sobolev gradient, respectively, after the first descent step. Actually, these are plots with the second component of the gradient on the vertical axis and the first component of $f$ on the horizontal axis. Plots of the first component of the gradient are similar but with smaller values. Note that these curves interpolate zeros at the data abscissae. The smoothness of the Sobolev gradient and lack of smoothness of the $L^2$ gradient is similar with additional iterations and with relative change in $\phi$. The penalty term is not used: $w = 0$. Table 1 displays, for each data set, the number of chords $n$ in the control polygon, the number of edges $m$ in the polygonal curve $f$, the number of descent steps NS required for convergence, the discretized energy functional $\phi$, and discrete approximations to the total curve length $L$ and some measures of fairness involving curvature $\kappa$ and, for the space curve, torsion $\tau$. In the case of the fourth data set, we have included the measures associated with the initial estimate (NS = 0) for comparison with those of the solution.
other data sets. Not surprisingly, the standard descent method, using the $L_2$ gradient, failed to converge. Plots of signed curvature and its derivative are depicted in Figs. 5 and 6. The X’s in these plots merely indicate values at the control point abscissae—not interpolated values.

The second data set is taken to be the four control points $(1, 0, 0)$, $(0, 1, 0)$, $(-1, 0, 0)$, and $(0, -1, 0)$, along with specified unit tangent vectors and curvature vectors from the unit circle. Plots of $\mathbf{f}$, $\Delta \mathbf{f}$, and $\Delta^2 \mathbf{f}$
are visually indistinguishable from the circle. We therefore display only the curvature and its derivative (in Figs. 7 and 8).

Data set 3 is taken from the helix curve \((\sin(t), \cos(t), t)\) (Goodman et al., 1998) with the following sequence of 12 evaluation points:

\[
\left(0, \frac{1}{3}\pi, \frac{2}{5}\pi, \pi, \frac{9}{5}\pi, \frac{39}{20}\pi, \frac{41}{20}\pi, \frac{11}{5}\pi, \frac{10}{3}\pi, \frac{11}{3}\pi, \frac{4}{3}\pi, 4\pi\right).
\]

Fig. 9 displays the projection onto the \(xy\) plane of a porcupine plot of the interpolant \(\mathbf{f}\) with normal curvature vectors \(\Delta^3\mathbf{f}\). The curvature and torsion of \(\mathbf{f}\) are depicted in Figs. 10 and 11. Note that the true helix curve has constant curvature \(\kappa = 0.5\) and torsion \(\tau = -0.5\). Its length is \(L = 4\sqrt{2}\pi \cong 17.772\), corresponding to \(\int \kappa^2 = \int \tau^2 \cong 4.443\). The error in the computed torsion is primarily due to forcing \(\Delta^3\mathbf{f} = 0\) and hence \(\tau = 0\) at the endpoints (see Fig. 11).
The final data set, borrowed from (Kaklis and Karavelas, 1997), is a closed space curve defined by control points \((0, 0, 6), (1.2, 0, 0), (2.5, 0.5, 0), (3.75, 2.5, 0), (3.5, 6, 0), (2.5, 8, -3), (0, 8, -3), (-2.5, 8, -3), (-3.5, 6, 0), (-3.75, 2.5, 0), (-2.5, 0.5, 0), (-1.2, 0, 0)\). The projection onto the \(xy\) plane of the minimum variation curve is depicted in Fig. 12. By varying descent parameters, we found additional critical points, some with smaller variation of curvature (and larger curve length) than that
of the curve computed with default parameter values. The smaller variation of curvature is associated with larger loops between the first and second and the first and last control points (shown at the bottom of Fig. 12).

5. Discussion and further research

We have described an effective method for constructing a discrete approximation to a curve which minimizes an arbitrary functional, representing a measure of fairness, subject to equality constraints. The method has the high computational cost associated with all global methods, but it is fast enough to be used in an interactive environment, perhaps not on currently available workstations, but certainly on future generations of workstations. For the small data sets used in our testing, execution times were under one second on a Pentium III.

The method requires a finite difference discretization in which the curve is represented by discrete vertices and derivative vectors. This provides more flexibility than piecewise polynomials. If a standard representation (B-splines, NURBS, etc.) is required, it can be obtained by fitting the discrete data. For example, we can easily construct the $C^3$ Hermite degree-7 polynomial interpolant of the knot values (control points) and first three derivative vectors (thus preserving the constraints) using the natural parameterization by arc length. The Hermite form can then be converted to a B-spline.

Our current implementation of the method allows only linear equality constraints, but it could be extended to treat inequality constraints by employing an active set method or an interior point method. This would enable us to construct curves that satisfy precise shape-preservation criteria such as those listed in (Goodman and Ong, 1997) and (Asaturyan et al., 2001).

We plan to apply our method to free-form surface design with triangulated surface approximations as discussed, for example, in (Welch and Witkin, 1994) and (Schneider and Kobbelt, 2000).

A well-documented Fortran-77 implementation of the method may be obtained by sending an email request to the author.
Acknowledgement

The author gratefully acknowledges the contribution of his long-time friend and frequent collaborator, John Neuberger, upon whose theory of Sobolev gradients this work is based, but retains full responsibility for any misrepresentation of that theory.

References