Fitting Uncertain Data with NURBS

Wolfgang Heidrich, Richard Bartels, George Labahn

Abstract. Fitting of uncertain data, that is, fitting of data points that are subject to some error, has important applications for example in statistics and for the evaluation of results from physical experiments. Fitting in these problem domains is usually achieved with polynomial approximation, which involves the minimization of an error at discrete data points. Norms typically used for this minimization include the l_1, l_2 and l_∞ norms, which are chosen depending on the problem domain and the expected type of error on the data points.

In this paper we describe how the l_1 and l_∞ norms can be applied to integral and rational B-spline fitting as a linear programming problem. This allows for the use of B-splines and NURBS for the fitting of uncertain data.

§1. Introduction

An important field of applications for approximation techniques is the fitting of uncertain data that occurs, for example, in statistics or as a result of measurements in experiments. Approximation techniques in this context usually involve fitting a polynomial through a set of data points that are subject to some error. The type of error depends on the specific application domain, and could for example be uniformly distributed over the data points, or concentrated in a few outliers.

Depending on the application domain and the expected type of error, different norms are selected for the minimization process. In particular, the l_2 norm is typically used for data sets in which every data point is subject to a small, normally distributed error. The l_1 norm is well-suited for removing a small set of outliers from a set of data points with otherwise high precision. Finally, the l_∞ norm is appropriate if every single data point is very precise, and therefore the approximation error should be evenly distributed amongst the data points.

In the context of B-spline approximation, the l_2 norm has traditionally been used almost exclusively. Consequently, the form of B-spline approximation described in most of the literature is not well suited for handling, for example, uncertain data with outliers.

Proceedings of Chamonix 1996
A. Le Méhauté, C. Rabut, and L. L. Schumaker (eds.), pp. 1–8.

ISBN 1-xxxxx-xxx-x.

In the following we will describe how the l_1 and l_∞ norms can be applied to integral and rational spline approximation problems using a linear programming approach. This allows for the efficient combination of B-splines and NURBS with standard techniques for fitting uncertain data.

§2. Preliminaries

The task of B-spline approximation is fitting a B-spline curve with N control points $\mathbf{c} = [c_1, \ldots, c_N]^T$ through a set of $M > N$ (typically $M \gg N$) data points $\mathbf{d} = [d_1, \ldots, d_M]^T$ at given parameter values $u_1, \ldots, u_M$. Such an approximation problem leads to an over-determined system of a linear equations

$$\underbrace{\begin{bmatrix} B_1(u_1) & \cdots & B_N(u_1) \\ B_1(u_2) & \cdots & B_N(u_2) \\ \vdots & \ddots & \vdots \\ B_1(u_M) & \cdots & B_N(u_M) \end{bmatrix}}_{\mathbf{B}} \cdot \begin{bmatrix} c_1 \\ \vdots \\ c_N \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_M \end{bmatrix}, \tag{1}$$

where $B_i(u)$ is the i^{th} B-spline basis function.

Since this system of equations is over-determined, in general an exact solution does not exist. Therefore, instead of directly solving Equation 1, it is necessary to find an approximate solution by minimizing the error $\|\mathbf{B} \cdot \mathbf{c} - \mathbf{d}\|$ with respect to some norm $\|.\|$.

As we have stated above, the typical choice for this norm in the B-spline literature is the least-squares (l_2) norm. It can be shown that a least-squares solution of (1) can be obtained as the solution of the linear equation system

$$\mathbf{B}^T\mathbf{B} \cdot \mathbf{c} = \mathbf{B}^T \cdot \mathbf{d}. \tag{2}$$

This square system of size $N \times N$ can be solved efficiently, which is one reason for the success of this approach.

In contrast to the least-squares norm, the l_1 and l_∞ minimizations cannot be obtained as the solutions of a linear equation system. Fortunately, it is possible to formulate the minimization using these two norms as a linear programming problem, which also allows for an efficient implementation.

A *linear programming problem* is a problem of the form

Minimize

$$\mathbf{q}^T \cdot \mathbf{x} - q_0$$

subject to

$$\mathbf{A} \cdot \mathbf{x} = \mathbf{b}.$$

In addition, some of the variables x_i may be restricted to non-negative values: $x_i \geq 0$.

The expression $\mathbf{q}^T \cdot \mathbf{x} - q_0$, which is to be minimized, is called *objective function*, while $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$ is a set of *constraints* that are to be fulfilled. The

name "linear programming problem" reflects the fact that both the objective function and the constraints are linear in the unknowns $\mathbf{x}$.

Problems of this general form are usually solved using the *simplex method* [3], or one of its descendants. It is important to note that optimized versions of the simplex method exist for various problems of more specific forms.

§3. Integral B-spline Fitting with the l_1 and l_∞ Norms

The l_1 B-spline approximation $\|\mathbf{B}\cdot\mathbf{c}-\mathbf{d}\|_1$ of a data set can be reduced to a linear programming problem. Introducing $\mathbf{b}_i$ as a shorthand notation for the i^{th} row-vector of the B-spline matrix $\mathbf{B}$, the approximation using the l_1 norm can be written as

$$\min_{\mathbf{c}} \sum_{i=1}^{M} |\mathbf{b}_i^T\mathbf{c} - d_i|.$$

The first step of transforming this to a linear programming problem is to make the function linear by removing the absolute value function. This can be achieved by introducing two vectors $\mathbf{p}$ and $\mathbf{n}$ of *slack variables* p_i and n_i. Using these two vectors of variables, we can write

$$\mathbf{b}_i^T\mathbf{c} - d_i = p_i - n_i \quad ; \quad p_i \geq 0,\ n_i \geq 0. \tag{3}$$

The intention is to have

$$p_i = \begin{cases} \mathbf{b}_i^T\mathbf{c} - d_i & ;\mathbf{b}_i^T\mathbf{c} - d_i \geq 0 \\ 0 & ;\text{otherwise} \end{cases} \tag{4}$$

and

$$n_i = \begin{cases} 0 & ;\mathbf{b}_i^T\mathbf{c} - d_i \geq 0 \\ -\mathbf{b}_i^T\mathbf{c} + d_i & ;\text{otherwise} \end{cases}. \tag{5}$$

Using these equations, the expression

$$\sum_{i=1}^{M} p_i + n_i$$

becomes the objective function and Equation 3 becomes the constraint. We define $\mathbf{0}_N$ to be a vector of N zeroes, and $\mathbf{1}_N$ to be a vector of N ones. The linear programming problem for the l_1 approximation can now be written as

Minimize

$$[\,\mathbf{0}_N^T \quad \mathbf{1}_M^T \quad \mathbf{1}_M^T\,]\cdot\begin{bmatrix}\mathbf{c}\\ \mathbf{p}\\ \mathbf{n}\end{bmatrix} \tag{6}$$

subject to

$$[\,\mathbf{B} \quad -\mathbf{Id}_M \quad \mathbf{Id}_M\,]\cdot\begin{bmatrix}\mathbf{c}\\ \mathbf{p}\\ \mathbf{n}\end{bmatrix} = \mathbf{d} \quad ; \quad p_i \geq 0,\ n_i \geq 0$$

The constraints take care that Equation 3 actually holds, while the minimization of the objective function guarantees Equations 4 and 5.

This linear programming problem can be solved using the normal simplex method. However, it is also possible to use an optimized version of the simplex method that makes use of the special structure of this problem [3].

The minimization with respect to the l_∞ norm can be expressed as a linear programming problem in a similar fashion. The l_∞ fit to some data points $\mathbf{d}$ is given as

$$\min_{\mathbf{c}} \left(\max_{i=1\ldots M} \left| \mathbf{b}_i^T \cdot \mathbf{c} - d_i \right| \right).$$

By defining $c_0 := \max_{i=0\ldots M} \left| \mathbf{b}_i^T \cdot \mathbf{c} - d_i \right|$, the approximation process can be rewritten as a linear programming problem. The expression c_0 becomes the objective function, while the constraints are given as

$$\begin{aligned} c_0 &\geq 0 \\ c_0 &\geq -\mathbf{b}_i^T \cdot \mathbf{c} + d_i \Leftrightarrow \quad \mathbf{b}_i^T \cdot \mathbf{c} + c_0 \geq d_i \\ c_0 &\geq \quad \mathbf{b}_i^T \cdot \mathbf{c} - d_i \Leftrightarrow -\mathbf{b}_i^T \cdot \mathbf{c} + c_0 \geq -d_i \end{aligned} \tag{7}$$

In matrix form this results to

Minimize

$$[\, \mathbf{0}_N^T \quad 1 \,] \cdot \begin{bmatrix} \mathbf{c} \\ c_0 \end{bmatrix} \tag{8}$$

subject to

$$\begin{bmatrix} \mathbf{B} & 1 \\ -\mathbf{B} & 1 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{c} \\ c_0 \end{bmatrix} \geq \begin{bmatrix} \mathbf{d} \\ -\mathbf{d} \end{bmatrix} \quad ; \quad c_0 \geq 0$$

It should be noted that this is a more general type of linear programming problem than stated above, since the constraints in this case are inequalities. By the introduction of additional slack variables, however, this problem can be transformed so that only equalities are used.

The resulting linear programming problem can again be solved either using the standard simplex algorithm or a version that is especially optimized for the structure of this specific problem [3].

§4. Rational B-spline Fitting

While approximations with integral B-splines are often sufficient, it is sometimes desirable to use rational B-splines (NURBS). In order for the solution to be useful, it is then essential to ensure that the weights of the rational B-spline curve remain positive. A clever scheme for rational B-spline fitting with positive weights has recently been proposed by Ma and Kruth [2].

The basic idea of this approach is to separate the calculation of the weights from the actual fitting process. The weights are obtained as the solution of a homogeneous linear equation system. In order to ensure that the weights remain positive, Ma and Kruth use quadratic programming with constraints to compute this solution. Once the weights have been obtained, they

are substituted into the linear equation system for rational approximation, and then the usual integral approximation process is applied to the system.

In the following we shall briefly review this approach before we describe how quadratic programming can be replaced by more efficient linear programming.

The approximation of M data points d_i with a NURBS curve with N control points c_i and weights w_i yields the following set of equations:

$$d_j = \frac{\sum_{i=1}^{N} w_i c_i B_i(u_j)}{\sum_{i=1}^{N} w_i B_i(u_j)} \quad ; \quad j = 1 \dots M. \tag{9}$$

Please recall that each of these expressions actually consists of e equations, where e is the dimension of the space. Following the setting in [2], we will in the following assume that the space is three-dimensional, that is, $e = 3$. This is only to make the notation more readable and comprehensible, and does not constitute a limitation of the algorithm itself.

Rewriting these linear equations in matrix form, and then performing some matrix manipulations (see [1] and [2] for details), yields

$$\begin{bmatrix} \mathbf{B}^T\mathbf{B} & & & -\mathbf{B}^T\mathbf{D}_x\mathbf{B} \\ & \mathbf{B}^T\mathbf{B} & & -\mathbf{B}^T\mathbf{D}_y\mathbf{B} \\ & & \mathbf{B}^T\mathbf{B} & -\mathbf{B}^T\mathbf{D}_z\mathbf{B} \\ & & & \mathbf{M} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{c}_x \\ \mathbf{c}_y \\ \mathbf{c}_z \\ \mathbf{w} \end{bmatrix} = \begin{bmatrix} \mathbf{0}_N \\ \mathbf{0}_N \\ \mathbf{0}_N \\ \mathbf{0}_N \end{bmatrix}, \tag{10}$$

where

$$\begin{aligned} \mathbf{M} = \quad & \mathbf{B}^T\mathbf{D}_x^2\mathbf{B} + \mathbf{B}^T\mathbf{D}_y^2\mathbf{B} + \mathbf{B}^T\mathbf{D}_z^2\mathbf{B} \\ & -(\mathbf{B}^T\mathbf{D}_x\mathbf{B})(\mathbf{B}^T\mathbf{B})^{-1}(\mathbf{B}^T\mathbf{D}_x\mathbf{B}) \\ & -(\mathbf{B}^T\mathbf{D}_y\mathbf{B})(\mathbf{B}^T\mathbf{B})^{-1}(\mathbf{B}^T\mathbf{D}_y\mathbf{B}) \\ & -(\mathbf{B}^T\mathbf{D}_z\mathbf{B})(\mathbf{B}^T\mathbf{B})^{-1}(\mathbf{B}^T\mathbf{D}_z\mathbf{B}) \end{aligned} \tag{11}$$

In this equation, $\mathbf{D}_x$, $\mathbf{D}_y$ and $\mathbf{D}_z$ are diagonal matrices holding the components of the data points. $\mathbf{c}_x = [c_1^x w_1, \dots, c_N^x w_N]^T$ is the vector of the x-components of all control points in homogeneous form. Similarly, $\mathbf{c}_y$ and $\mathbf{c}_z$ are the vectors of the y- and z-components of the control points, respectively, and $\mathbf{w}$ is the vector of weights. $\mathbf{B}$ is the B-spline matrix from Sections 2 and 3.

Thus, the weights have been separated from the control points, as they can be determined from the homogeneous linear equation system

$$\mathbf{M} \cdot \mathbf{w} = \mathbf{0}_N. \tag{12}$$

These weights are then substituted into Equation (9), and finally Ma and Kruth propose to apply a standard least-squares approximation, as described in Section 2, to each component of the control points separately.

Note that the null space of $\mathbf{M}$ might have a dimension larger than 1, in which case more than one solution of (12) exists, and a specific one has to be chosen. On the other hand, the matrix $\mathbf{M}$ may also have full rank, in which case an exact solution of Equation 11 does not exist, so that a minimization process becomes necessary in order to obtain an approximation.

The authors of [2] suggest the use of the least-squares norm for determining the weights. This minimization process yields an approximation in cases where $\mathbf{M}$ is non-singular, and selects a specific set of weights in cases where multiple solutions exist. A least-squares approximation of (12) can be obtained from the singular value decomposition (SVD, see [1] for details).

However, it is usually desirable to have only positive weights, because otherwise singularities in the the resulting curve may arise. Since it is not clear how a set of positive weights can be obtained from the SVD in an automated fashion, Ma and Kruth and also propose another approach, which is based on solving the quadratic programming problem

Minimize

$$\mathbf{w}^T \cdot \mathbf{M}^T\mathbf{M} \cdot \mathbf{w}$$

subject to

$$w_i > 0.$$

While this approach works and yields the desired results, it relies on quadratic programming, which is relatively inefficient. On the other hand, since $\mathbf{M}$ is not geometrically meaningful, it is not clear, why the least-squares norm should be preferred over other norms for this minimization.

We therefore propose the use of the l_1 or l_∞ norm, which can be implemented more efficiently as a linear programming problem, similar to the implementation of the integral fitting process. A modified version of (6) for determining a set of positive weights with the l_1 norm is given by

Minimize

$$[\,\mathbf{0}_N^T \quad \mathbf{1}_N^T \quad \mathbf{1}_N^T \quad \mathbf{1}_N^T\,] \cdot \begin{bmatrix} \mathbf{w} \\ \mathbf{p} \\ \mathbf{n} \\ \mathbf{s} \end{bmatrix} \tag{13}$$

subject to

$$\begin{bmatrix} \mathbf{M} & -\mathbf{Id}_N & \mathbf{Id}_N & \\ \mathbf{Id}_N & & & -\mathbf{Id}_N \end{bmatrix} \cdot \begin{bmatrix} \mathbf{w} \\ \mathbf{p} \\ \mathbf{n} \\ \mathbf{s} \end{bmatrix} = \begin{bmatrix} \mathbf{0}_N \\ \mathbf{1}_N \end{bmatrix} \quad ; \quad p_i \geq 0,\ n_i \geq 0,\ s_i \geq 0$$

The additional vector $\mathbf{s}$ of slack variables is used to ensure that the resulting weights obey the condition $\mathbf{w} = \mathbf{1}_N + \mathbf{s}$, with $s_i \geq 0$. Therefore the smallest resulting weight will be 1. Since all weights can be scaled by a constant factor without changing the resulting curve, this is not a restriction for the algorithm.

The linear programming Problem (8) for the l_∞ norm can be modified in a similar fashion, and also allows for the computation of positive weights.

Having computed the weights with one of these norms, it is then possible to determine the control points of the approximating NURBS curve by applying one of the l_1, l_2 or l_∞ norms to the Equation 9, similarly to the integral case.

§5. Results

There are several differences between the scenario described in this paper and traditional fitting of uncertain data. First of all, splines as described in this paper define *parametric* instead of *functional* curves. However, since the B-spline approximation treats every component of the vector space separately, the parametric approximation can be seen as a composition of several independent functional approximations.

Another difference is that splines have different approximation properties than polynomials. Because of the local control property of B-splines, it is to be expected that a spline approximation locally changes to meet outliers better, and that the removal of outliers is therefore not as good as in the polynomial case. On the other hand, with splines it is possible to create good approximations to data sets with a more complex shape. The negative effects of the local control property can partly be compensated by using the error-specific norms.

Figure 1 shows two examples for integral quadratic B-spline approximation using different norms. From top to bottom the figure contains the original function, the l_1, l_2 and l_∞ fit.

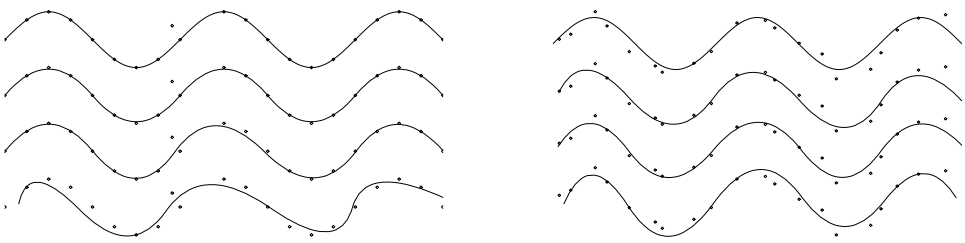

Fig. 1. Fitting of uncertain data with integral B-splines..

Both data sets consists of 21 points that have been originally sampled from a sine curve. The image on the left contains a single outlier that is being removed nicely by the l_1 fit. The data set on the right has been generated by jittering the original points with a normally distributed error. In this situation, the least-squares fit produces the best results.

The left side os Figure 2 shows another example for an integral quadratic fit with different norms (from top to bottom: l_1, l_2 and l_∞).

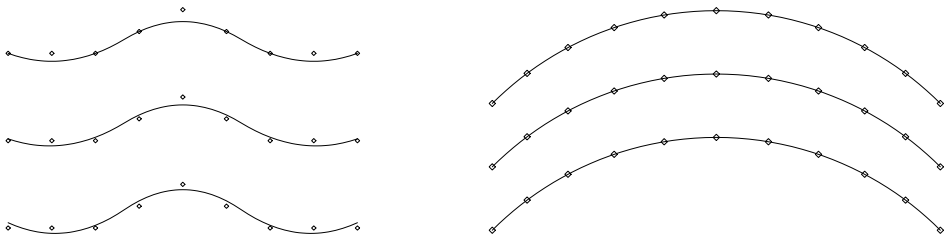

Fig. 2. fitting of precise data (left) and reproduction of a quarter circle (right).

Under the assumption that the positions of the data points are *exact*, the l_∞ fit produces the best result, since it manages to distribute the error evenly across the parameter domain, while the l_1 norm treats the second, fifth and eighth point as outliers, and is therefore not able to correctly represent the shape of the data set. The quality of the l_2 norm is somewhere inbetween these two results.

We have also implemented rational fitting using the l_1 and l_∞ norms to determine the weights. In our experiments, we were not able to find any differences in the quality of the weights obtained using these two norms compared to the least-squares approach proposed by the authors of [1].

In particular, experiments show that both the l_1 and l_∞ norms are capable of reproducing conic sections and other sampled NURBS curves, as long as enough data points are provided together with the exact parameterization. An example for such an approximation is shown on the right side of Figure 2.

References

1. Wolfgang Heidrich, Spline Extensions for the MAPLE Plot System. Master Thesis, Department of Computer Science, University of Waterloo, 1995.
2. Weiyin Ma and Jean-Pierre Kruth, Mathematical Modeling of Free-Form Curves and Surfaces from Discrete Points with NURBS. In P. J. Laurent, A. Le Méhaute, L. L. Schumaker, editors, *Curves and Surfaces in Geometric Design*, pages 319-326, A K Peters, 1994.
3. G. A. Watson, Approximation Theory and Numerical Methods. John Wiley & Sons, 1980.

Wolfgang Heidrich
Universität Erlangen
Graphische Datenverarbeitung
Am Weichselgarten 9
D-91058 Erlangen, Germany
Email: Heidrich@informatik.uni-erlangen.de

Richard Bartels, George Labahn
Department of Computer Science
University of Waterloo
Waterloo, Ontario, Canada
N2L 3G1
Email: rhbartel@cgl.uwaterloo.ca, glabahn@daisy.uwaterloo.ca