

Article

Spline Approximation, Part 2: From Polynomials in the Monomial Basis to B-splines—A Derivation

Nikolaj Ezhov ^{1,*}, Frank Neitzel ¹ and Svetozar Petrovic ^{1,2}

¹ Institute of Geodesy and Geoinformation Science, Technische Universität Berlin, 10623 Berlin, Germany; frank.neitzel@tu-berlin.de (F.N.); svetozar.petrovic@campus.tu-berlin.de (S.P.)

² Section 1.2: Global Geomonitoring and Gravity Field, GFZ German Research Centre for Geosciences, 14473 Potsdam, Germany

* Correspondence: nikolaj.ezhov@alumni.tu-berlin.de

Abstract: In a series of three articles, spline approximation is presented from a geodetic point of view. In part 1, an introduction to spline approximation of 2D curves was given and the basic methodology of spline approximation was demonstrated using splines constructed from ordinary polynomials. In this article (part 2), the notion of B-spline is explained by means of the transition from a representation of a polynomial in the monomial basis (ordinary polynomial) to the Lagrangian form, and from it to the Bernstein form, which finally yields the B-spline representation. Moreover, the direct relation between the B-spline parameters and the parameters of a polynomial in the monomial basis is derived. The numerical stability of the spline approximation approaches discussed in part 1 and in this paper, as well as the potential of splines in deformation detection, will be investigated on numerical examples in the forthcoming part 3.

Keywords: spline; B-spline; polynomial; monomial; basis change; Lagrange; Bernstein; interpolation; approximation; least squares adjustment

Citation: Ezhov, N.; Neitzel, F.; Petrovic, S. Spline Approximation, Part 2: From Polynomials in the Monomial Basis to B-splines—A Derivation. *Mathematics* **2021**, *9*, 2198. <https://doi.org/10.3390/math9182198>

Academic Editor: Clemente Cesarano

Received: 13 July 2021
Accepted: 25 August 2021
Published: 8 September 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).

1. Introduction

In engineering geodesy, the use of point clouds derived from areal measurement methods, such as terrestrial laser scanning or photogrammetry, results in the necessity to approximate them by a curve or surface that can be described using a continuous mathematical function, often by means of splines. In part 1 of a three-part series of articles, presented by Ezhov et al. [1], the basic methodology of spline approximation was shown by means of ordinary cubic polynomials concatenated under constraints for continuity, smoothness, and continuous curvature. The resulting linear adjustment problem can be solved within the Gauss-Markov model with constraints for the unknowns.

However, a starting point for advanced considerations in engineering geodesy are almost always the formulas for B-spline curves and B-spline surfaces given in the textbook by Piegler and Tiller [2] (pp. 81 and 100), where the functional values of the B-spline basis functions are recursively computed according to the formulas by de Boor [3] and Cox [4]. As these formulas have a very complex mathematical derivation, but are still very easy to use, they are mostly used like a given recipe. Attempts to explain B-splines in an illustrative way often only include explanations of the application of the de Boor's algorithm, as, for example, in the contribution by Lowther and Shene [5]. A derivation of the B-spline with the help of quadratic Bézier spline curves was presented by Berkahn [6] (p. 73 ff.).

In this paper, we take the spline representation presented by Ezhov et al. [1] as a starting point to derive the B-spline form by means of basis changes. This is to show the user that the de Boor's algorithm can be interpreted as another, very elegant and numerically stable, representation of the simple and intuitive representation based on or-

dinary polynomials.

A brief overview of the historical development of the B-spline can be found, for example, in the textbook by Farin [7] (p. 119). He explains that the forerunner of the B-spline first appeared in a publication on approximation theory by the mathematician Lobatschewsky [8]. It was constructed as a convolution of certain probability distributions. However, at that time the term “spline” has not been introduced yet. Subsequently, in 1946, Schoenberg used B-splines in his seminal publications [9,10] for data approximation. It is commonly accepted that, with these articles, the modern representation of spline approximation began and the term “spline” was used for the first time, see the historical notes in the textbook by Schumaker [11] (p. 10). Further on, many authors dealt with the utilization and the development of the B-splines. Farin [7] (p. 119) points out that one of the most important developments in this theory was the introduction of the recurrence relations, discovered by de Boor [3], Cox [4], and Mansfield. The contribution of Mansfield to this development is mentioned by de Boor in [3]. However, the recurrence relation had already appeared in contributions by Popoviciu and Chakalov in the 1930s, see the article by de Boor and Pinkus [12] and the literature cited therein.

B-splines were later fully exploited in computer graphics. Consequently, most papers about B-splines were published in this field, see for example, the contributions by the authors Piegler and Tiller [2], Szilvasi-Nagy [13], Farin [7], and Zheng et al. [14]. However, there are several publications where B-splines were used for statistical data analysis, e.g., the contributions by Wold [15] or Anderson and Turner [16].

Applications of (B)-splines in geodesy can be traced back to 1975, when Sünkel began using them for various problems. In [17], bicubic spline functions were used for the reconstruction of functions from discrete data. In [18], the representation of geodetic integral formulas by bicubic spline functions was shown using splines constructed from ordinary cubic polynomials, see [18] (p. 10 ff.), similar to the representation used later by Ezhov et al. [1]. In [19], the local interpolation of gravity data by spline functions was derived, and in [20] (p. 18 ff.) B-splines were used for smooth surface representation. As references for the B-spline functions, the publications of Schoenberg [21], Schumaker [22], and Späth [23] are listed by Sünkel [20].

On the basis of B-splines on the line, Jekeli [24] (p. 7 ff.) showed the use of spherical B-splines for representations of functions on a sphere for geopotential modeling. Mautz et al. [25] used B-splines to develop a representation of global ionosphere maps based on B-spline wavelets. The local improvement of the International Reference Ionosphere (IRI) by means of an N-dimensional B-spline surface was developed by Koch and Schmidt [26].

The application of (B)-splines in engineering geodesy can be traced back to 1985, when Dzapo et al. [27] used cubic splines for the determination of the lengths of railroad tracks. A recent application of B-spline curves and B-spline surfaces is the approximation of 3D point cloud data, as shown e.g. by Bureick et al. [28]. Kermarrec et al. [29] applied hierarchical B-splines for the approximation of 3D point cloud data obtained from measurements with a terrestrial laser scanner. Furthermore, B-spline models play an important role in spatio-temporal deformation analysis as shown by Harmening [30].

The main goal of this paper is to present an alternative derivation of the B-spline function to the purely mathematical derivation presented by Schoenberg [9] and the recursion formulas for the B-spline function developed by de Boor [3]. This derivation is based on the transition from the representation of a polynomial in the monomial basis (ordinary polynomial) to the Lagrangian form and, from there, to the Bernstein form, which finally results in the B-spline representation. In all investigations, univariate splines are used in form of a spline function $y = f(x)$. For the derivation of the formulas, we first consider the case of interpolation. The developed formulas are then used for spline approximation using least squares adjustment.

The general interpolation problem is, to find for $n+1$ given data points $(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)$, with $x_0 < x_1 < \dots < x_n$ a function $f: \mathbb{R} \rightarrow \mathbb{R}$ in such a way that the interpolation condition

$$f(x_i) = y_i, \quad i = 0, 1, \dots, n \quad (1)$$

is fulfilled. The choice of a suitable function f depends on the properties of the data, e.g., its smoothness or periodicity. Furthermore, the choice of f is influenced by computational aspects, i.e., the computational effort for the determination of the coefficients and numerical stability of the resulting equation system. Functions that are commonly used for interpolation are:

- Polynomials;
- Trigonometric functions;
- Exponential functions;
- Logarithmic functions;
- Rational functions.

Ezhov et al. [1] elaborated a spline approximation using piecewise ordinary cubic polynomials. In the following, we will take a closer look at the interpolation using polynomials in different representations to bridge the gap to the B-spline representation.

In general, the set of functions for interpolating given data points consists of coefficients a_i and so-called basis functions $\phi_0(x), \dots, \phi_n(x)$ and the interpolating function is defined as a linear combination

$$f(x) = \sum_{i=0}^n a_i \phi_i(x) \quad (2)$$

of these basis functions. Considering the interpolation condition (1), we obtain

$$f(x_i) = \sum_{i=0}^n a_i \phi_i(x_i) = y_i, \quad i = 0, 1, \dots, n. \quad (3)$$

The simplest and most common type of interpolation uses polynomials. These polynomials can be represented in different ways, as shown by Gander [31], but all of them will give the same result for the interpolation.

To avoid excessive formula derivations and to illustrate the geometric relationships in the formula derivations, *quadratic splines* that consist of piecewise parabola segments are considered in the following. The generalization to splines of a higher degree is relatively straightforward.

Using the example of a parabola, we start with the representation of a polynomial in the monomial basis (ordinary polynomial) and perform the following transitions:

- (i) From ordinary polynomial to Lagrangian form (Section 2);
- (ii) From Lagrangian form to Bernstein form (Section 3);
- (iii) From Bernstein form to B-spline representation (Section 4).

The reason for the representation of a polynomial in a different basis than the well-known monomial basis applied by Ezhov et al. [1] often lies in the fact that for the computation of the polynomial coefficients equation systems result, whose solution requires less computational effort and is numerically more stable. However, it is important to understand that the change of the basis functions still results in the *same* interpolating polynomial for the given data, and only the *representation* of the polynomial changes. After the derivation of the B-spline representation, this form is used for spline approximation using least squares adjustment (Section 5). The interesting fact that the determined spline parameters can be used for a transition “backwards” from B-spline to ordinary polynomial is shown in Appendix A. The transition “forwards” from ordinary polynomial to B-spline is shown in Appendix B.

2. Transition from Ordinary Polynomial to Lagrangian Form

With the monomial basis functions

$$\phi_i(x) = x^i, \quad i = 0, 1, \dots, n, \tag{4}$$

we obtain for the parabola with $n = 2$ from (3)

$$P_2(x) = a_0 + a_1x + a_2x^2. \tag{5}$$

A polynomial in the monomial basis is often referred to as standard form or ordinary polynomial. Arranging the monomials in a vector $\mathbf{m}(x) = [1, x, x^2]^T$ and the coefficients in a vector $\mathbf{a} = [a_0, a_1, a_2]^T$, we obtain

$$P_2(x) = \mathbf{a}^T \mathbf{m}(x). \tag{6}$$

A visualization of the monomial functions for the parabola is shown in Figure 1.

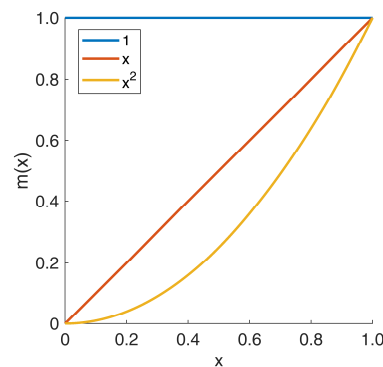


Figure 1. Monomial basis functions for a parabola in the interval $[0, 1]$.

Considering the interpolation condition (1), the coefficients can be determined from the solution of the linear equation system

$$\mathbf{y} = \mathbf{V} \mathbf{a}, \tag{7}$$

with $\mathbf{y} = [y_0, y_1, y_2]^T$ and the Vandermonde matrix

$$\mathbf{V} = \begin{bmatrix} 1 & x_0 & x_0^2 \\ 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \end{bmatrix}, \tag{8}$$

cf. the textbook by Farin [7] (p. 100). Using a numerical example with three points that define a parabola from the handbook by Bronshtein et al. [32] (p. 918), see Table 1, the determination of the coefficients will be demonstrated.

Table 1. Numerical example for interpolating a parabola.

i	x_i	y_i
0	0	1
1	1	3
2	3	2

From Equation (7), we obtain for this numerical example

$$\begin{bmatrix} 1 \\ 3 \\ 2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 3 & 9 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix}. \tag{9}$$

This system of equations can be solved with, for example, the Gaussian elimination method and the numerical result is $a_0 = 1, a_1 = \frac{17}{6}, a_2 = -\frac{5}{6}$. Thus, the equation of the parabola reads

$$P_2(x) = 1 + \frac{17}{6}x - \frac{5}{6}x^2. \tag{10}$$

It can be stated that, when using the monomial basis, the numerical effort to determine the coefficients requires roughly n^3 operations using direct methods for solving systems of linear equations, as stated e.g., by Farin [7] (p. 102), and is therefore comparably high. It can be reduced by employing iterative methods. In addition, the matrix of the equation system is increasingly ill-conditioned as the degree of the polynomial increases. Both the conditioning of the resulting linear equation system and the computational effort for determining the coefficients can be improved by using other basis functions. The result is the same interpolating polynomial, but in a different *representation*.

In the following subsections we will illustrate two ways how to perform a transition from the monomial basis to a representation with Lagrange polynomials.

2.1. Transition by Means of Basis Transformation

We want to solve the problem of basis transformation between two different sets of coefficients a_i and b_i fulfilling

$$P_2(x) = \sum_{i=0}^2 a_i \phi_i(x) = \sum_{i=0}^2 b_i \psi_i(x), \tag{11}$$

where $\phi_i(x)$ and $\psi_i(x)$ are two different sets of basis functions. Using the monomial basis function (4) in the first summation formula, we obtain

$$P_2(x) = \sum_{i=0}^2 a_i \phi_i(x) = a_0 + a_1x + a_2x^2 = \begin{bmatrix} a_0 & a_1 & a_2 \end{bmatrix} \begin{bmatrix} 1 \\ x \\ x^2 \end{bmatrix}, \tag{12}$$

as already shown in (5) and (6). Multiplying out the second summation formula in (11) yields

$$P_2(x) = \sum_{i=0}^2 b_i \psi_i(x) = b_0\psi_0(x) + b_1\psi_1(x) + b_2\psi_2(x) = \begin{bmatrix} b_0 & b_1 & b_2 \end{bmatrix} \begin{bmatrix} \psi_0(x) \\ \psi_1(x) \\ \psi_2(x) \end{bmatrix} \tag{13}$$

and, according to (11), we can write

$$\begin{bmatrix} a_0 & a_1 & a_2 \end{bmatrix} \begin{bmatrix} 1 \\ x \\ x^2 \end{bmatrix} = \begin{bmatrix} b_0 & b_1 & b_2 \end{bmatrix} \begin{bmatrix} \psi_0(x) \\ \psi_1(x) \\ \psi_2(x) \end{bmatrix}. \tag{14}$$

In the case of monomial basis, we selected basis functions and obtained the coefficients, see (7), while here we do the opposite. We select $b_0 = y_1, b_1 = y_2, b_2 = y_3$ for the coefficients to obtain corresponding basis functions for which we introduce the new notation $l_i(x)$ with $i = 0, 1, 2$. Furthermore, for a_0, a_1, a_2 we introduce the solution according to (7) and we obtain

$$[y_0 \ y_1 \ y_2] \begin{bmatrix} 1 & 1 & 1 \\ x_0 & x_1 & x_2 \\ x_0^2 & x_1^2 & x_2^2 \end{bmatrix}^{-1} \begin{bmatrix} 1 \\ x \\ x^2 \end{bmatrix} = [y_0 \ y_1 \ y_2] \begin{bmatrix} l_0(x) \\ l_1(x) \\ l_2(x) \end{bmatrix}. \tag{15}$$

By comparing the coefficients of the vectors $[y_0 \ y_1 \ y_2]$, we see that the basis functions $l_i(x)$ can be computed from

$$\begin{bmatrix} l_0(x) \\ l_1(x) \\ l_2(x) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ x_0 & x_1 & x_2 \\ x_0^2 & x_1^2 & x_2^2 \end{bmatrix}^{-1} \begin{bmatrix} 1 \\ x \\ x^2 \end{bmatrix}. \tag{16}$$

Introducing the vector $\mathbf{l}(x) = [l_0(x) \ l_1(x) \ l_2(x)]^T$ yields

$$\mathbf{l}(x) = (\mathbf{V}^T)^{-1} \mathbf{m}(x). \tag{17}$$

The same formula can also be obtained taking the basis transformation from Lagrange to monomials $\mathbf{V}^T \mathbf{l}(x) = \mathbf{m}(x)$, as presented by Gander [31], and solving it for $\mathbf{l}(x)$. From (17), we obtain the result

$$\mathbf{l}(x) = \begin{bmatrix} \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} & \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} & \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)} \end{bmatrix}^T, \tag{18}$$

which is called the Lagrange basis. A visualization of the Lagrange basis functions for the parabola is shown in Figure 2.

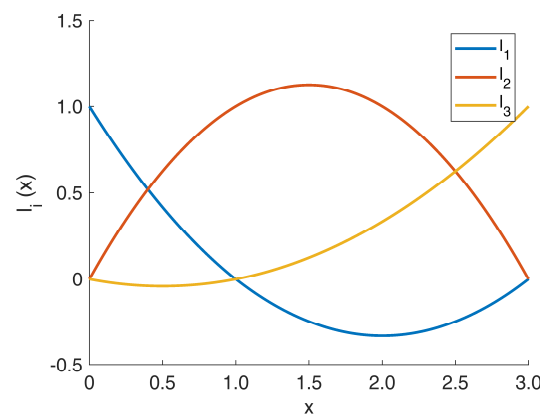


Figure 2. Lagrange basis functions for a parabola in the interval $[0, 3]$ for the example $x_0 = 0$, $x_1 = 1$ and $x_2 = 3$.

Finally, the resulting polynomial reads

$$P_2(x) = \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} y_0 + \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} y_1 + \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)} y_2. \tag{19}$$

An alternative derivation of the Lagrange representation of a polynomial is presented in the following subsection.

2.2. Transition by Linear Combinations

In this subsection, we present an alternative derivation of the polynomial in the Lagrange representation. We use the fact that each parabola can be represented as a combination of two lines. In fact, if we take a look at (5), we find that it can be rearranged into the form $f(x) = a_0 + (a_1 + a_2 x)x$, which represents a combination (product) of two lines

plus some constant value a_0 . By using this idea, the Lagrange representation of a quadratic polynomial can be derived, which results in a linear combination of three quadratic functions $p(x)$ and the respective y_i coordinates.

We consider the points $(x_0, y_0), (x_1, y_1), (x_2, y_2)$, where $x_0 < x_1 < x_2$. Now two lines

$$f_{0,1}(x) = a_0 + a_1x \tag{20}$$

within the interval $[x_0, x_1)$ and

$$f_{1,1}(x) = b_0 + b_1x \tag{21}$$

within the interval $[x_1, x_2]$ are defined, see Figure 3. The subscripts i and d in $f_{i,d}$ represent the number of the polynomial i for a given degree d .

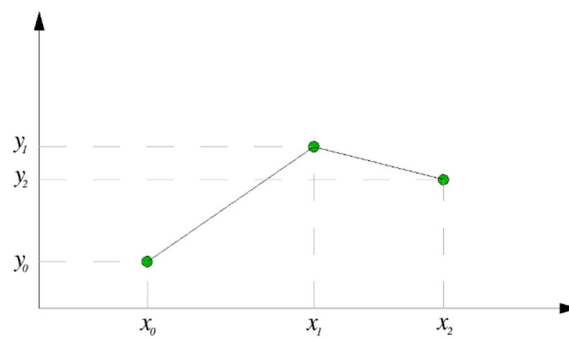


Figure 3. Line interpolation.

From two given points, the slope of the line (20) can be written as

$$a_1 = \frac{y_1 - y_0}{x_1 - x_0}. \tag{22}$$

The y -intercept can be computed from

$$a_0 = y_0 - x_0 \frac{y_1 - y_0}{x_1 - x_0} \quad \text{or} \quad a_0 = y_1 - x_1 \frac{y_1 - y_0}{x_1 - x_0}. \tag{23}$$

Inserting (22) and (23) into (20) yields, after some rearrangement, for the first line

$$f_{0,1}(x) = \frac{x_1 - x}{x_1 - x_0} y_0 + \frac{x - x_0}{x_1 - x_0} y_1. \tag{24}$$

Analogously, the equation for the second line can be written as

$$f_{1,1}(x) = \frac{x_2 - x}{x_2 - x_1} y_1 + \frac{x - x_1}{x_2 - x_1} y_2. \tag{25}$$

By taking a combination of these two line equations, where x varies within the interval $[x_0, x_2]$, the expression

$$f_{0,2}(x) = \frac{x_2 - x}{x_2 - x_0} f_{0,1}(x) + \frac{x - x_0}{x_2 - x_0} f_{1,1}(x) \tag{26}$$

for the parabola is derived, which represents a linear combination of three quadratic polynomial functions and the coordinates y_0 , y_1 and y_2 .

To explain how the factors in front of the terms $f_{0,1}(x)$ and $f_{1,1}(x)$ in (26) are derived, the idea behind the Lagrange polynomials has to be explained. Lagrange was

looking for an interpolation polynomial that could be constructed without solving a system of equations, e.g., with Vandermonde matrix as design matrix, see (7). There is a widely accepted assumption that his idea for the solution was to find a function that, at each given data point $(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)$, gives 1 and, at the rest of the other given points gives 0. For a mathematician, this type of polynomials was, presumably, relatively easy to find. We will take the case of a parabola into consideration. Therefore, for each given x_i , the following polynomials

$$p_{x_0}(x) = \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)}, \tag{27}$$

$$p_{x_1}(x) = \frac{(x - x_0)(x - x_2)}{(x_1 - x_0)(x_1 - x_2)}, \tag{28}$$

$$p_{x_2}(x) = \frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)} \tag{29}$$

can be created.

As previously mentioned, these functions $p_{x_i}(x)$ are equal to 1 when the variable x is equal to the corresponding coordinate x_i of the given point and 0 at all other given points. Therefore, by multiplying each of these functions $p_{x_i}(x)$ with their corresponding values y_i ensures that, at the particular given point x_i , the result will be y_i and 0 at all other given points. The summation of all these functions $p_{x_i}(x)$, multiplied by their corresponding values y_i , results in a Lagrangian interpolation polynomial

$$f(x) = \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)}y_0 + \frac{(x - x_0)(x - x_2)}{(x_1 - x_0)(x_1 - x_2)}y_1 + \frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)}y_2. \tag{30}$$

The quadratic polynomial (30) represents a linear combination of the Lagrangian basis polynomials with the constants y_0, y_1 and y_2 as coefficients. A general formula for the definition of the Lagrange interpolation polynomials can be found e.g., in the handbook by Bronshtein et al. [32] (p. 918).

As previously explained, a quadratic polynomial can be represented as a combination of two lines. Using the Lagrangian form for a polynomial of first order, the following two line equations

$$f_{0,1}(x) = \frac{x - x_1}{x_0 - x_1}y_0 + \frac{x - x_0}{x_1 - x_0}y_1, \tag{31}$$

$$f_{1,1}(x) = \frac{x - x_2}{x_1 - x_2}y_1 + \frac{x - x_1}{x_2 - x_1}y_2 \tag{32}$$

are derived. If we follow the already described Lagrange’s idea to obtain a quadratic polynomial, the following multiplications

$$f_{0,2}(x) = \frac{x - x_2}{x_0 - x_2}f_{0,1}(x) + \frac{x - x_0}{x_2 - x_0}f_{1,1}(x) \tag{33}$$

must be performed.

Consequently, Equation (33) results in a Lagrangian polynomial of second order. Looking at (33), the factors $(x - x_2)/(x_0 - x_2)$ and $(x - x_0)/(x_2 - x_0)$ are those that appear in front of the terms $f_{0,1}(x)$ and $f_{1,1}(x)$ in (26). However, if we compare the Lagrange polynomials in (31), (32), (33) with the expressions in (24), (25), (26), the only difference is that (31), (32) and (33) have negative denominators and numerators at some points in the

Lagrange polynomials, e.g. $(x - x_2) / (x_0 - x_2)$. In (24), (25) and (26) there are no negative denominators or numerators, although the equations are the same as in (31), (32) and (33). This is for reasons of better analogy with the B-spline form.

After the origin of the factors in front of $f_{0,1}(x)$ and $f_{1,1}(x)$ in (26) is explained, inserting (24) and (25) into (26) finally yields

$$f_{0,2}(x) = \frac{(x_1 - x)(x_2 - x)}{(x_1 - x_0)(x_2 - x_0)} y_0 + \frac{(x - x_0)(x_2 - x)}{(x_1 - x_0)(x_2 - x_1)} y_1 + \frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)} y_2. \tag{34}$$

This form coincides with (19) and represents a second degree Lagrange interpolating polynomial through three points, see the handbook by Bronshtein et al. [32] (p. 918). The plot of the parabola is depicted in Figure 4.

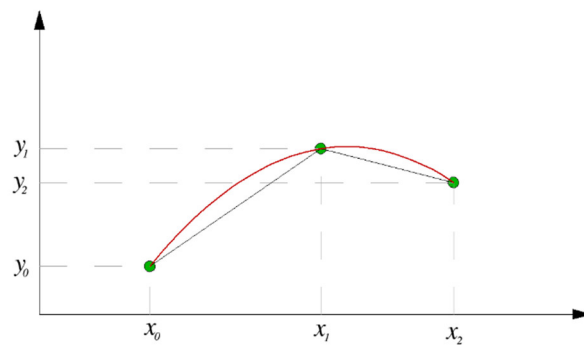


Figure 4. Lagrange interpolation by lines (black) and by a parabola (red).

Inserting the values of the numerical example from Table 1 into (34) yields the Lagrangian form

$$f_{0,2}(x) = \frac{(1-x)(3-x)}{(1-0)(3-0)} \cdot 1 + \frac{(x-0)(3-x)}{(1-0)(3-1)} \cdot 3 + \frac{(x-0)(x-1)}{(3-0)(3-1)} \cdot 2, \tag{35}$$

which can be “simplified” to the monomial basis form

$$f_{0,2}(x) = 1 + \frac{17}{6}x - \frac{5}{6}x^2. \tag{36}$$

The result is the same as in (10), but the values of the parameters a_0, a_1, a_2 are now derived with the advantage of not explicitly solving a linear equation system. A disadvantage of Lagrange interpolation in practical application is directly visible from (19), resp. (34), namely that each time a value x changes, the Lagrange basis polynomials must be recalculated. Further general limits of Lagrange interpolation, e.g., that polynomial interpolants may oscillate, are explained by Farin [7] (p. 101 ff.).

3. Transition from Lagrangian Form to Bernstein Form

The Bernstein form, named after S. N. Bernstein, was used to prove the Weierstrass approximation theorem; see the explanation by Farin [7] (p. 90 ff.). The coefficients of a Bernstein polynomial over the interval $[0, 1]$ are all non-negative and sum up to 1, as Farin [7] (p. 57 ff.) showed, which is called a *convex combination*; see the textbook by Rockafellar [33] (p. 11). Therefore, by using the Bernstein form, numerical instabilities are avoided.

Looking at Figure 3, the interpolating parts of the line segments can be extended to both ends of the interval $[x_0, x_2]$ in which the polynomial is defined. At the end of the extended line segments, two additional y -coordinates for the values x_0 and x_2 are

computed, as seen in Figure 5. To distinguish these two y -coordinates from the given values, a different notation, y'_0 and y'_2 , is used (not to be confused with the derivation of a function). In a general case, all five points have different coordinates which implies that the coordinate x_1 is not necessarily in the middle of the interval $[x_0, x_2]$. Therefore, in general $x_1 \neq (x_0 + x_2) / 2$.

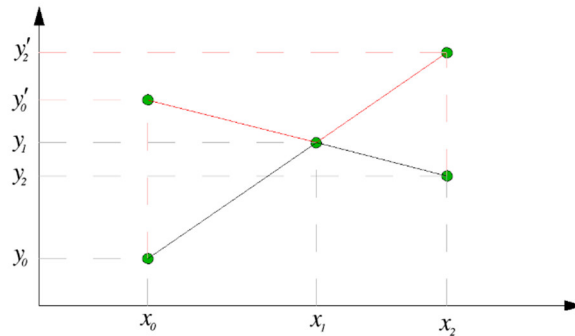


Figure 5. Extended line segments from Lagrangian form.

From Figure 5, as explained previously, the line equations

$$f_{0,1}(x) = \frac{x_2 - x}{x_2 - x_0} y_0 + \frac{x - x_0}{x_2 - x_0} y'_2, \quad f_{1,1}(x) = \frac{x_2 - x}{x_2 - x_0} y'_0 + \frac{x - x_0}{x_2 - x_0} y_2 \quad (37)$$

are derived. By taking combinations of these two lines, it follows

$$f_{0,2}(x) = \left(\frac{x_2 - x}{x_2 - x_0} \right)^2 y_0 + \frac{x_2 - x}{x_2 - x_0} \frac{x - x_0}{x_2 - x_0} y'_2 + \frac{x - x_0}{x_2 - x_0} \frac{x_2 - x}{x_2 - x_0} y'_0 + \left(\frac{x - x_0}{x_2 - x_0} \right)^2 y_2, \quad (38)$$

which can be brought into the form

$$f_{0,2}(x) = \left(\frac{x_2 - x}{x_2 - x_0} \right)^2 y_0 + 2 \frac{x_2 - x}{x_2 - x_0} \frac{x - x_0}{x_2 - x_0} \left(\frac{y'_2 + y'_0}{2} \right) + \left(\frac{x - x_0}{x_2 - x_0} \right)^2 y_2. \quad (39)$$

A geometrical interpretation of the term $(y'_2 + y'_0) / 2$ is depicted in Figure 6.

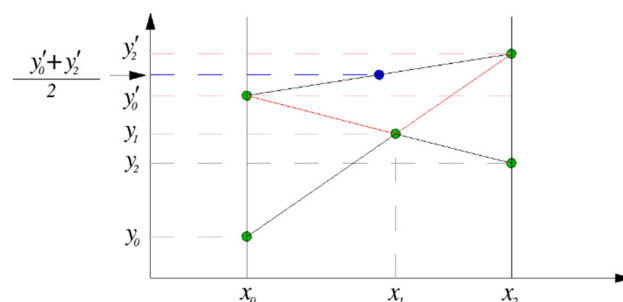


Figure 6. Geometrical interpretation of the term $(y'_2 + y'_0) / 2$.

Equation (39) is a more general representation of the Bernstein form, independent of the limits of the interval $[x_0, x_2]$, which means that x can vary between any two real values. For proving the Weierstrass approximation theorem, Bernstein used a special interval of length 1, i.e., $[x_0, x_2] = [0, 1]$. Using this interval simplifies the equations, which were later used by Bézier and de Casteljau. Therefore, when using $[x_0, x_2] = [0, 1]$, Equation (37) results in

$$f_{0,1}(x) = (1-x)y_0 + xy'_2, \quad f_{1,1}(x) = (1-x)y'_0 + xy_2 \tag{40}$$

and (39) obtains the form

$$f_{0,2}(x) = (1-x)^2 y_0 + 2x(1-x) \left(\frac{y'_2 + y'_0}{2} \right) + x^2 y_2, \tag{41}$$

in which $(1-x)^2$, $2x(1-x)$ and x^2 are the Bernstein basis polynomials and y_0 , $(y'_2 + y'_0)/2$ and y_2 are the Bernstein coefficients. In general, a polynomial in Bernstein form of degree n can be written as

$$P_n(x) = \sum_{i=0}^n \beta_i B_{i,n}(x), \tag{42}$$

where β_i are the Bernstein coefficients and

$$B_{i,n}(x) = \binom{n}{i} x^i (1-x)^{n-i}, \quad i = 0, 1, \dots, n \tag{43}$$

are the Bernstein basis polynomials; see e.g., the handbook by Bronshtein et al. [32] (p. 935). For a parabola, we obtain $B_{0,2} = (1-x)^2$, $B_{1,2} = 2x(1-x)$ and $B_{2,2} = x^2$, cf. (41). These basis polynomials for a parabola are depicted in Figure 7.

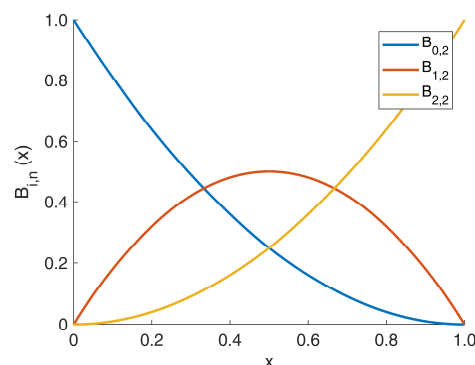


Figure 7. Bernstein basis polynomials for a parabola in the interval [0, 1].

In (39), the term $(y'_2 + y'_0)/2$, see Figure 6, reveals one fundamental property of the quadratic polynomial that, to the best of our knowledge, is not described in the literature so far. It can be stated as:

All pairs of line segments that have one end fixed at the beginning or at the end of the parabola and the other end at the opposite sides of the interval, and intersect themselves on the parabola, have an equal average of the coordinates at the ends that are not fixed. Moreover, the extremum of the parabola lies on the intersection of those line segments that have equal values at the opposite ends of the interval.

The proof of this statement is straightforward. A parabola is uniquely defined by three points. If we retain end points (x_0, y_0) and (x_2, y_2) , replacing the point (x_1, y_1) by any other point on the same parabola, we will again obtain the equation of the form (39). The equation of the parabola remains unchanged although y'_0 and y'_2 have changed, since the only term depending on the new point is $(y'_0 + y'_2)/2$. A parabola is described by a differentiable function. By computing its derivative, equating it with zero and solving the resulting equation for x , we obtain the coordinate of the extremum x_e . Consequently, the resulting equation for the coordinate of the extremum y_e can be

solved. By substituting these two values for x_e and y_e in a line equation constructed for y'_0 and y'_2 , where $x = x_0$ and $x = x_2$ respectively, it can be shown that $y'_0 = y'_2 = C_y$. Therefore, $y'_0 \neq y'_2$ implies $C_y = (y'_0 + y'_2)/2$. This proves the above statement, which is illustrated in Figure 8. M and N represent the fixed points at the beginning and at the end of the parabola, with m_1, \dots, m_6 and n_1, \dots, n_6 as end points at the opposite sides of the interval. Every pair of line segments is defined as $(\overline{Mn_0}, \overline{Nm_6})$, $(\overline{Mn_1}, \overline{Nm_5})$, $(\overline{Mn_2}, \overline{Nm_4})$, ..., $(\overline{Mn_6}, \overline{Nm_0})$ and depicted in light blue color. Figure 8 illustrates that all these pairs of line segments intersect on the parabola.

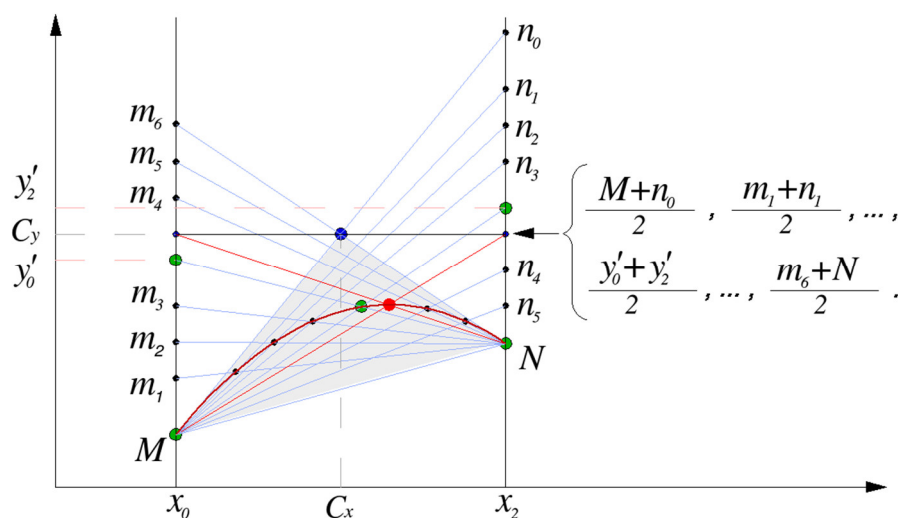


Figure 8. Construction of a Bernstein polynomial. The extremum of the parabola is marked with a red dot and the control point with a blue dot. Green dots indicate points on the parabola, resp. points resulting from extension of straight lines between these points to the boundaries of the interval $[x_0, x_2]$. The convex hull formed by the control points (x_0, y_0) , (C_x, C_y) and (x_2, y_2) is highlighted in grey.

Considering the above, there are an infinite number of combinations defining the term $(y'_2 + y'_0)/2$ depending on the position of the point (x_1, y_1) , between M and N , that defines the parabola, cf. Figure 6. All of these pairs of line segments, as previously explained, have an equal average of the y -coordinates at the end points. Therefore, the term that uniquely represents all these averages is written more generally in the form

$$C_y = \frac{y'_m + y'_n}{2}, \tag{44}$$

where y'_m and y'_n represent any pair of y' -coordinates from the mentioned line segments that intersect on the parabola. Correspondingly, the coordinate C_x is defined as

$$C_x = \frac{x_i + x_{i+1}}{2}, \tag{45}$$

where x_i and x_{i+1} define the ends of the interval in which the parabola varies. Hence, C_x is always in the middle of the interval.

There is an additional property of the line segments $\overline{Mn_0}$ and $\overline{Nm_6}$. They intersect at the point (C_x, C_y) and are tangential to the curve. This can be easily proven. We can compute the first derivative and solve for y'_m and y'_n from (44), then construct two line equations for $\overline{(x_0, y_0)(x_2, y_n)}$ and $\overline{(x_0, y_m)(x_2, y_2)}$ and solve for their intersection.

In the context of geometric modelling, the point defined by (C_x, C_y) is called *control point* and, concerning basis splines (B-splines), it is named *de-Boor-point*. In Figure 8, this point is marked with a blue dot. Moreover, the first and the last point of the parabola are also referred to as control points, see e.g., the textbook by Piegl and Tiller [2] (pp. 389–390).

With (44), the general form of (39) can be written as

$$f_{0,2}(x) = \left(\frac{x_2 - x}{x_2 - x_0}\right)^2 y_0 + 2 \frac{x_2 - x}{x_2 - x_0} \frac{x - x_0}{x_2 - x_0} C_y + \left(\frac{x - x_0}{x_2 - x_0}\right)^2 y_2. \tag{46}$$

As a more general representation of the Bernstein polynomial, (46) is a convex combination as well. The proof is obvious. Since $x_0 < x_2$, all functions in front of y_0 , C_y and y_2 are non-negative and have a sum of 1. The control points (x_0, y_0) , (C_x, C_y) and (x_2, y_2) form a convex hull, highlighted in grey in Figure 8, that encloses all realizations of x in (46), which means that all points of the parabola lie within the convex hull. A convex hull is the smallest convex set that contains a given set, with a convex set being a set wherein the straight line segment, connecting any two points of the set is completely contained within the set; see e.g., the textbook by Farin [7] (p. 439). The advantage of the general representation of the Bernstein polynomial is that it can be applied directly without scaling the data set to the interval $[x_0, x_2] = [0, 1]$.

If we now consider the example from Table 1 once again, we get, with $y'_0 = 3.5$ and $y'_2 = 7.0$, from (44) the result $C_y = 5.25$ and therefore

$$f_{0,2}(x) = \left(\frac{3-x}{3-0}\right)^2 \cdot 1 + 2 \cdot \frac{3-x}{3-0} \cdot \frac{x-0}{3-0} \cdot 5.25 + \left(\frac{x-0}{3-0}\right)^2 \cdot 2, \tag{47}$$

which can be “simplified” to the monomial basis form

$$f_{0,2}(x) = 1 + \frac{17}{6}x - \frac{5}{6}x^2. \tag{48}$$

This is of course the same as (10) and (36).

For the case of the interpolation by a parabola, it can be concluded that both the Bernstein and the Lagrangian form yield the same result. However, the Bernstein form has the advantage that all points of the parabola lie in the convex hull defined by its parameters.

4. Transition from Bernstein Form to B-spline

In the previous section, the Bernstein form was derived with a more general approach, instead of the standard representation within the interval $[0, 1]$. This form can be easily used for derivation of the B-spline.

In this section, Equation (46) is used for further derivations. It is assumed that the line segments that define the quadratic polynomial intersect at the extremum of the parabola and have an identical coordinate C_{y_i} . Since the notion of spline is to be explained, only the endpoints of two smoothly connected parabolas are considered, which are better known as knots (x_0, y_0) , (x_1, y_1) and (x_2, y_2) . The problem of the constraints and how they are imposed on the spline, for spline interpolation, is not taken into consideration. It is presumed that the first parabola of the spline is already defined, e.g., by imposing a constraint for the direction of the tangent in point x_0 as boundary condition, and in this case, is identical with the one from the previous section, see Figure 8.

The initial situation is depicted in Figure 9, where:

- One parabola already exists between the knots (x_0, y_0) and (x_1, y_1) ; and

- One knot (x_2, y_2) is given that is to be interpolated.

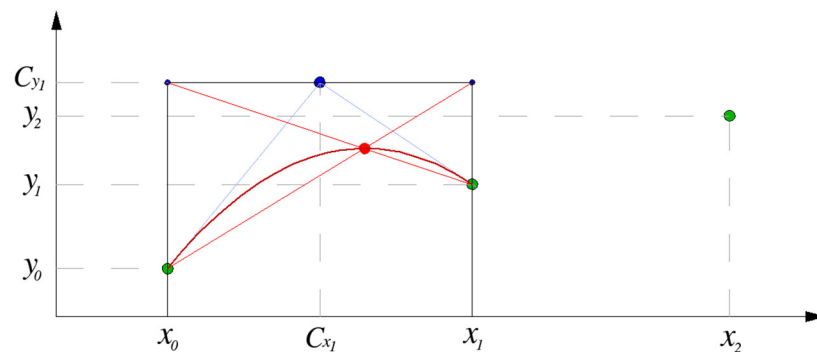


Figure 9. Initial situation before the construction of the second part of the spline. The extremum of the first parabola is marked with a red dot, the control point with a blue dot, and the green dots indicate points to be interpolated.

The task is now to construct a second parabola between the knots (x_1, y_1) and (x_2, y_2) which is tangential to the first one.

The second parabola is constructed as follows:

- The line segment $\overline{(x_0, C_{y_1})(x_1, y_1)}$ is extended from the end point (x_1, y_1) of the parabola to the end of the interval $[x_1, x_2]$, yielding the additional coordinates (x_2, C_{y_2}) and therefore the new line segment $\overline{(x_1, y_1)(x_2, C_{y_2})}$, see Figure 10;
- Between points (x_1, C_{y_2}) and (x_2, y_2) , the line segment $\overline{(x_1, C_{y_2})(x_2, y_2)}$ can be constructed, see Figure 11;
- By taking combinations of these two line segments, within the interval $[x_1, x_2]$, another parabola will be constructed, smoothly connected to the previous one. The whole function constructed from these two smoothly connected parabolas represents a spline. Both parabolas end tangentially at the point (x_1, y_1) w.r.t the line segment, defined as $\overline{(C_{x_1}, C_{y_1})(C_{x_2}, C_{y_2})}$, see Figure 12.

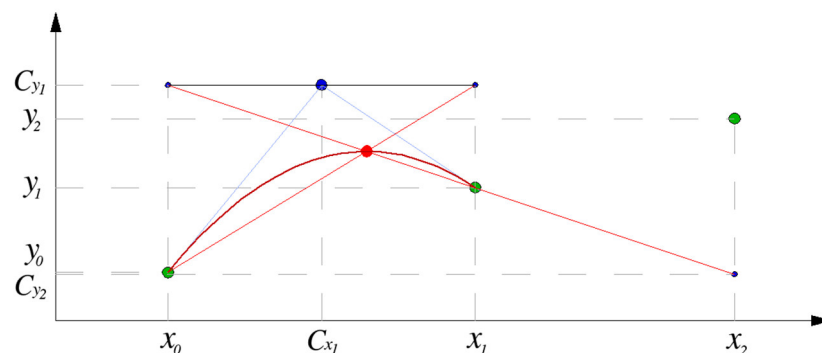


Figure 10. Line segment $\overline{(x_1, y_1)(x_2, C_{y_2})}$. The extremum of the first parabola is marked with a red dot, the control point with a blue dot, and the green dots indicate points to be interpolated.

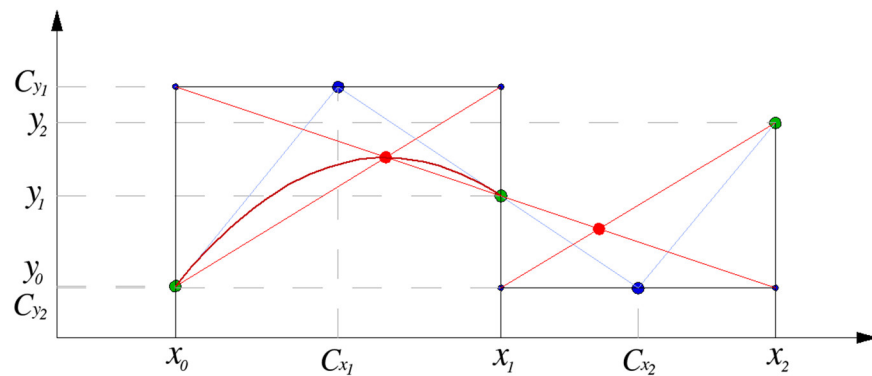


Figure 11. Line segment $(x_1, C_{y_2})(x_2, y_2)$. The extrema of the parabolas are marked with a red dot, the control points with a blue dot, and the green dots indicate points to be interpolated.

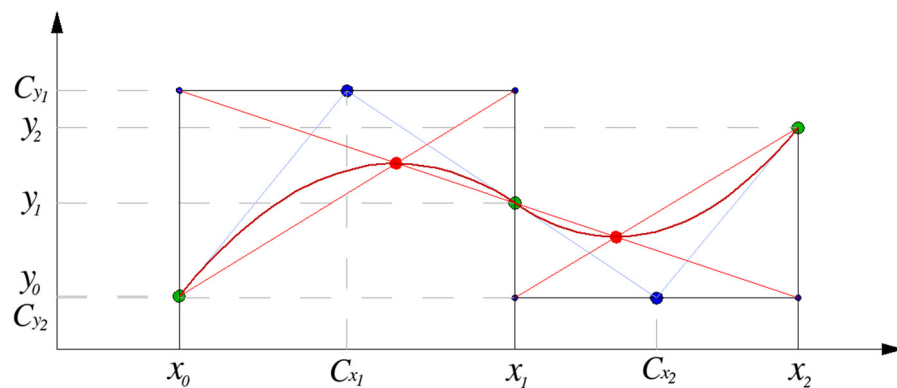


Figure 12. Second parabola within the interval $[x_1, x_2]$, smoothly connected to the previous one. The extrema of the parabolas are marked with a red dot, the control points with a blue dot, and the green dots indicate points to be interpolated.

Finally, this is a type of spline that is constructed by using convex combinations and it is exactly the same as the B-spline. This is shown by the equations derived further in the text. Algebraically, this derivation is similar to the one in the previous section. The first pair of line equations is written as

$$f_{0,1}(x) = \frac{x_1 - x}{x_1 - x_0} y_0 + \frac{x - x_0}{x_1 - x_0} C_{y_1}, \quad f_{1,1}(x) = \frac{x_2 - x}{x_2 - x_0} C_{y_1} + \frac{x - x_0}{x_2 - x_0} C_{y_2}. \quad (49)$$

By taking a combination of these two lines within the interval $[x_0, x_1]$, it follows

$$f_{0,2}(x) = \left(\frac{x_1 - x}{x_1 - x_0} \right)^2 y_0 + \left(\frac{x_1 - x}{x_1 - x_0} \frac{x - x_0}{x_1 - x_0} + \frac{x - x_0}{x_1 - x_0} \frac{x_2 - x}{x_2 - x_0} \right) C_{y_1} + \frac{(x - x_0)^2}{(x_2 - x_0)(x_1 - x_0)} C_{y_2}. \quad (50)$$

For the second parabola the same approach is applied. The second pair of line equations is defined in the same way as the previous one, yielding

$$f_{2,1}(x) = \frac{x_2 - x}{x_2 - x_0} C_{y_1} + \frac{x - x_0}{x_2 - x_0} C_{y_2}, \quad f_{3,1}(x) = \frac{x_2 - x}{x_2 - x_1} C_{y_2} + \frac{x - x_1}{x_2 - x_1} y_2. \quad (51)$$

Analogously to (50), but in this case within the interval $[x_1, x_2]$, we obtain

$$f_{1,2}(x) = \frac{x_2 - x}{x_2 - x_1} \frac{x_2 - x}{x_2 - x_0} C_{y_1} + \left(\frac{x_2 - x}{x_2 - x_1} \frac{x - x_0}{x_2 - x_0} + \frac{x - x_1}{x_2 - x_1} \frac{x_2 - x}{x_2 - x_1} \right) C_{y_2} + \left(\frac{x - x_1}{x_2 - x_1} \right)^2 y_2. \tag{52}$$

If the above derived expressions are compared to an explicit expression of a quadratic B-spline it becomes clear that they are the same. However, for the sake of clarity we have used a particular notation that does not correspond to the usual B-spline notation.

Looking at (50) and (52), the only unknowns are C_{y_1} and C_{y_2} , all other values are given. It appears that a quadratic spline, constructed from two linear polynomials, can be estimated with only two equations. This type of reasoning would lead to a paradox. Therefore, for this type of a spline, since all constraints (smoothness and continuity) are hidden inside the equations, at least four observations are necessary for a solution. For comparison, a spline constructed from two polynomials of the form $a_0 + a_1x + a_2x^2$ requires at least three observations and three constraints.

The idea presented in this section can be used for constructing splines of higher degrees as well. In those cases, depending on the degree of the spline, one would need more knots and more control points for creating a spline segment, while the number of combinations increases. Moreover, the expressions became more complicated, which makes this approach unsuitable for practical applications. However, de Boor [3] developed an algorithm for computing the basis functions that, in our case, constructs the expressions in front of the coordinates of the control points in (50) and (52).

De Boor’s algorithm is based on the Cox-de Boor recursion formula that we will use for explaining the computation of a B-Spline. Recursion itself is non-intuitive, which makes this formula difficult to be explained analytically. The higher the degree of the spline, the more cumbersome and harder to follow the equations become. Therefore, in order to make a comparison with (50) and (52), we will use an explicit expression of a quadratic B-spline that is derived from the Cox-de Boor recursion formula. For this purpose, we use the knots and the naming convention from the previous example (49)–(52). For this formula to work, we have to rename the parameters (control points) y_0 to C_{y_0} and y_2 to C_{y_3} . The Cox-de Boor recursion requires additional knots at the beginning and at the end of the spline based on the degree of the spline. If the spline degree is $d = 2$, two additional knots are presumed to be both at the beginning and at the end of the spline. In our case, they are

$$x_{-2}, x_{-1} \text{ and } x_3, x_4, \tag{53}$$

which is called knot multiplication. Additional knots with e.g.,

$$x_{-2} = x_{-1} = x_0 \text{ and } x_2 = x_3 = x_4 \tag{54}$$

are especially used to gain endpoint interpolation. The additional knots do not necessarily have to be equal to x_0 and x_2 , they are just required to follow the order $x_{-2} \leq x_{-1} \leq x_0$ and $x_2 \leq x_3 \leq x_4$. However, in our example we follow the convention of the knot multiplication, and because of the knot multiplication, formula (56) has zero divisions, and the solution for this problem is to declare that “anything divided by zero is equal to zero”.

The base case of the recursion formula is

$$B_{i,0}(x) = \begin{cases} 1, & x_i \leq x < x_{i+1}, \\ 0, & \text{otherwise,} \end{cases} \tag{55}$$

with a recursive step to compute the *basis functions*

$$B_{i,d}(x) = \frac{x - x_i}{x_{i+d} - x_i} B_{i,d-1}(x) + \frac{x_{i+d+1} - x}{x_{i+d+1} - x_{i+1}} B_{i+1,d-1}(x), \tag{56}$$

which is summed over the k -th interval of the spline $f_{k,d}(x)$, such as

$$f_{k,d}(x) = \sum_{i=k-d}^k C_{y_{i+d}} B_{i,d}(x). \tag{57}$$

At the beginning of the recursion (56), index d denotes the degree of the spline and i is a computed index $i = k - d, \dots, k$. In each recursive step, d is lowered down for 1 until it reaches 0. At $d = 0$, the recursion formula terminates and the base case (55) is evaluated.

The index k is the index of the interval $[x_k, x_{k+1}]$ with $k = 0, 1, \dots, l - 1$, where l is the number of knots. Therefore, for $k = 0$, the recursion formula (56) is computed for every i of the sum (57) and evaluated in (55). Afterwards, for $k = 1$, the same computation is performed, and so on, until the last interval. The parameters C_{y_i} in (57) correspond to each basis function terminated by $B_{i,0}(x)$ in (55) at the end of the recursion. An example of quadratic basis functions using the knots $x_{k_0} = 0$, $x_{k_1} = 1$ and $x_{k_2} = 3$, resulting in two intervals $k = 0$ and $k = 1$, and the knot sequence $[0, 0, 0, 1, 3, 3, 3]$, is shown in Figure 13.

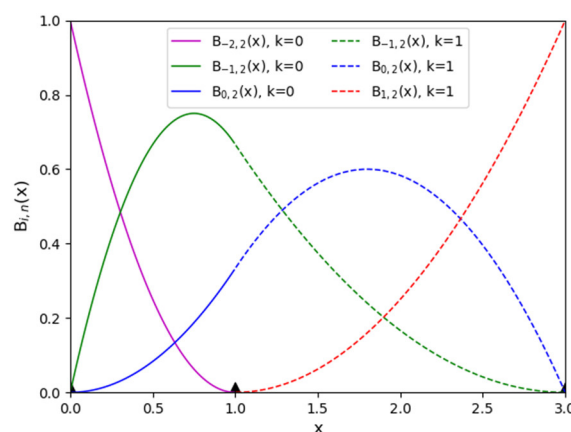


Figure 13. Quadratic B-Spline basis functions using the knots $x_{k_0} = 0$, $x_{k_1} = 1$ and $x_{k_2} = 3$, resulting in two intervals $k = 0$ and $k = 1$, and the knot sequence $[0, 0, 0, 1, 3, 3, 3]$. The knots are marked with black triangles on the x -axis.

To compare this approach with (50) and (52), we use an explicit expression of a generic quadratic B-spline formula. For deriving it, we consider $d = 2$ in (56), then we follow all recursive steps until $d = 0$ for every $B_{i,d}(x)$, which yields the explicit expression of a generic quadratic B-spline formula

$$B_{i,2}(x) = \frac{(x - x_i)^2}{(x_{i+2} - x_i)(x_{i+1} - x_i)} B_{i,0}(x) + \frac{(x_{i+3} - x)^2}{(x_{i+3} - x_{i+1})(x_{i+3} - x_{i+2})} B_{i+2,0}(x) + \left(\frac{(x - x_i)(x_{i+2} - x)}{(x_{i+2} - x_i)(x_{i+2} - x_{i+1})} + \frac{(x_{i+3} - x)(x - x_{i+1})}{(x_{i+3} - x_{i+1})(x_{i+2} - x_{i+1})} \right) B_{i+1,0}(x). \tag{58}$$

For clearer description, we treat each part of (57) separately and, for the computation, we use the knot sequence (53). At the interval $k = 0$, with index $i = -2$, the first element of the sum is

$$C_{y_0} B_{-2,2}(x) = C_{y_0} \left[\frac{(x-x_{-2})^2}{(x_0-x_{-2})(x_{-1}-x_{-2})} B_{-2,0}(x) + \frac{(x_1-x)^2}{(x_1-x_{-1})(x_1-x_0)} B_{0,0}(x) + \left(\frac{(x-x_{-2})(x_0-x)}{(x_0-x_{-2})(x_0-x_{-1})} + \frac{(x_1-x)(x-x_{-1})}{(x_1-x_{-1})(x_0-x_{-1})} \right) B_{-1,0}(x) \right]. \tag{59}$$

With $i = -1$, the second element of the sum is

$$C_{y_1} B_{-1,2}(x) = C_{y_1} \left[\frac{(x-x_{-1})^2}{(x_1-x_{-1})(x_0-x_{-1})} B_{-1,0}(x) + \frac{(x_2-x)^2}{(x_2-x_0)(x_2-x_1)} B_{1,0}(x) + \left(\frac{(x-x_{-1})(x_1-x)}{(x_1-x_{-1})(x_1-x_0)} + \frac{(x_2-x)(x-x_0)}{(x_2-x_0)(x_1-x_0)} \right) B_{0,0}(x) \right] \tag{60}$$

and, with $i = 0$, the last element of the sum is

$$C_{y_2} B_{0,2}(x) = C_{y_2} \left[\frac{(x-x_0)^2}{(x_2-x_0)(x_1-x_0)} B_{0,0}(x) + \frac{(x_3-x)^2}{(x_3-x_1)(x_3-x_2)} B_{2,0}(x) + \left(\frac{(x-x_0)(x_2-x)}{(x_2-x_0)(x_2-x_1)} + \frac{(x_3-x)(x-x_1)}{(x_3-x_1)(x_2-x_1)} \right) B_{1,0}(x) \right]. \tag{61}$$

Taking the knot multiplications (53) and (54) into account for all elements of the sums (59)–(61) yields

$$C_{y_0} B_{-2,2}(x) = C_{y_0} \left[\frac{(x-x_0)^2}{(x_0-x_0)(x_0-x_0)} B_{-2,0}(x) + \frac{(x_1-x)^2}{(x_1-x_0)(x_1-x_0)} B_{0,0}(x) + \left(\frac{(x-x_0)(x_0-x)}{(x_0-x_0)(x_0-x_0)} + \frac{(x_1-x)(x-x_0)}{(x_1-x_0)(x_0-x_0)} \right) B_{-1,0}(x) \right], \tag{62}$$

$$C_{y_1} B_{-1,2}(x) = C_{y_1} \left[\frac{(x-x_0)^2}{(x_1-x_0)(x_0-x_0)} B_{-1,0}(x) + \frac{(x_2-x)^2}{(x_2-x_0)(x_2-x_1)} B_{1,0}(x) + \left(\frac{(x-x_0)(x_1-x)}{(x_1-x_0)(x_1-x_0)} + \frac{(x_2-x)(x-x_0)}{(x_2-x_0)(x_1-x_0)} \right) B_{0,0}(x) \right], \tag{63}$$

$$\begin{aligned}
 C_{y_2} B_{0,2}(x) = C_{y_2} & \left[\frac{(x-x_0)^2}{(x_2-x_0)(x_1-x_0)} B_{0,0}(x) \right. \\
 & + \frac{(x_2-x)^2}{(x_2-x_1)(x_2-x_2)} B_{2,0}(x) \\
 & \left. + \left(\frac{(x-x_0)(x_2-x)}{(x_2-x_0)(x_2-x_1)} + \frac{(x_2-x)(x-x_1)}{(x_2-x_1)(x_2-x_1)} \right) B_{1,0}(x) \right].
 \end{aligned}
 \tag{64}$$

The base case (55) evaluates only the functions in front of $B_{0,0}(x)$, all other base functions are not taken into consideration nor are the fractions where division by zero occurred. Finally, by summing up all results (62), (63) and (64) for the first interval, as stated in (57), and evaluating by (55), we obtain

$$\begin{aligned}
 f_{0,2}(x) &= \sum_{i=2}^0 C_{y_{i+2}} B_{i,2}(x) \\
 &= \frac{(x_1-x)^2}{(x_1-x_0)(x_1-x_0)} C_{y_0} \\
 &+ \left(\frac{(x-x_0)(x_1-x)}{(x_1-x_0)(x_1-x_0)} + \frac{(x_2-x)(x-x_0)}{(x_2-x_0)(x_1-x_0)} \right) C_{y_1} \\
 &+ \frac{(x-x_0)^2}{(x_2-x_0)(x_1-x_0)} C_{y_2}.
 \end{aligned}
 \tag{65}$$

If Equation (65) is compared to (50), it can be concluded that they are identical. It must just be taken into consideration, as stated previously, that y_0 is renamed to C_{y_0} . If the same procedure is applied to $k = 1$, the result is identical to (52), considering that y_2 is renamed to C_{y_3} .

Since spline functions $y = f(x)$ are considered, only the component C_{y_i} of a control point (C_x, C_{y_i}) appears as a factor in front of the basis functions $B_{i,d}(x)$. Using a parametric spline curve, both components of a control point appear in front of the basis functions as e.g., applied by Bureick et al. [34].

The Cox-de Boor recursion Formulas (55) and (56) are the basis of the de Boor’s algorithm. The presented functions in front of the parameters are called basis functions and the curve is called B-spline curve.

5. B-spline Approximation

As pointed out by Ezhov et al. [1], in engineering geodesy it is not appropriate to apply a spline interpolation due to the high point density and the fact that measurements are affected by random measurement errors. Observation errors and other abrupt changes in the data points would be modeled, resulting in a strongly oscillating spline. The solution to this is to divide the sequence of points into not so many intervals, determined by predefined knots. The distribution of the knots is a fundamental problem in spline approximation and different knot placement strategies can be applied to solve it, as explained by Ezhov et al. [1]. Within the resulting intervals, we consider an overdetermined configuration and, hence, the parameters of a B-spline can be computed by least squares adjustment.

Since the Cox-de Boor Formulas (55)–(57) manage the whole process of computing the coefficients of the unknowns, the approximation can be performed with a simple linear functional model. Here, we consider quadratic splines that consist of piecewise parabolic segments. A generalization to splines of higher degree is straightforward.

5.1. Definition of the Problem

Let:

- $y_0, y_1, \dots, y_j, \dots, y_m$ be a sequence of observed values;
- $\Sigma_{LL} = \sigma_0^2 \mathbf{Q}_{LL}$ be the variance-covariance matrix of the observations y_j ;
- $x_0 < x_1 < \dots < x_j < \dots < x_m$ be a sequence of non-decreasing error-free values referring to the observations;
- $x_{k_0} < \dots < x_{k_{n-1}}$ be a non-decreasing sequence of user-defined values for the knots on the x -axis which are regarded as error-free.

To determine:

- $C_{y_0}, C_{y_1}, \dots, C_{y_n}$, which represent the y -component of the control “points” of the quadratic B-spline.

After defining these quantities, we can create a spline approximation from an overdetermined configuration using the Gauss-Markov model.

5.2. Formulation of the Adjustment Problem

The observation equations are of the same form for each quadratic polynomial within a spline. This is a consequence of the Cox-de Boor recursion formula. This approach (55), (56) and (57) determines to which interval a value x_j belongs. One has to consider that the knot sequence must be given in advance.

Considering x_j as fixed values, yields also $B_{i,d}(x)$, computed from (55), (56), and (57), as fixed values. Here, we assume that $B_{i,2}(x)$ corresponds to the given parameter C_{y_i} . Explaining the observation equations explicitly by means of the Cox-de Boor’s formula is too cumbersome. Hence, with y_j as observations, the observation equations

$$\begin{aligned}
 y_0 + v_0 &= \sum_{i=1}^n \hat{C}_{y_i} B_{i,2}(x_0), \\
 y_1 + v_1 &= \sum_{i=1}^n \hat{C}_{y_i} B_{i,2}(x_1), \\
 &\vdots \\
 y_m + v_m &= \sum_{i=1}^n \hat{C}_{y_i} B_{i,2}(x_m)
 \end{aligned}
 \tag{66}$$

can be set up. The observation vector

$$\mathbf{L} = [y_0 \quad y_1 \quad y_2 \quad \dots \quad y_m]^T
 \tag{67}$$

contains all observations, while the vector of unknown parameters can be written as

$$\hat{\mathbf{X}} = [\hat{C}_{y_0} \quad \hat{C}_{y_1} \quad \dots \quad \hat{C}_{y_n}]^T.
 \tag{68}$$

From the stochastic model

$$\Sigma_{LL} = \sigma_0^2 \mathbf{Q}_{LL},
 \tag{69}$$

with σ_0^2 as theoretical variance factor, the corresponding weight matrix is obtained from

$$\mathbf{P} = \mathbf{Q}_{LL}^{-1},
 \tag{70}$$

supposing the cofactor matrix to be non-singular. By looking at the observation equations (66), it is obvious that this spline approximation problem is linear and, hence, can easily be written in matrix notation

$$\mathbf{L} + \mathbf{v} = \mathbf{A}\hat{\mathbf{X}}, \quad (71)$$

where \mathbf{A} is the design matrix that contains the coefficients of the unknowns. Equation (71), together with (69), is denoted as Gauss-Markov model; see e.g., the textbook by Niemeier [35] (p. 137). Considering the sequence of the unknowns in (68), the design matrix reads

$$\mathbf{A} = \begin{bmatrix} B_{0,2}(x_0) & B_{1,2}(x_0) & \cdots & B_{n,2}(x_0) \\ B_{0,2}(x_1) & B_{1,2}(x_1) & \cdots & B_{n,2}(x_1) \\ \vdots & \vdots & \ddots & \vdots \\ B_{0,2}(x_m) & B_{1,2}(x_m) & \cdots & B_{n,2}(x_m) \end{bmatrix}. \quad (72)$$

Even though the construction of this matrix is quite trivial, the presence of the basis functions makes it rather unusual. However, with the derivations in Section 4, it is at least clear how these basis functions can be obtained. Looking at the notation in (72), one might be misled that \mathbf{A} is a full matrix, which is not the case. To put it simply, the Cox-de Boor formula evaluates only those x_j that belong to a particular interval and everything else is equal to zero.

5.3. Least Squares Adjustment

One feature of the B-splines is that constraints for continuity, smoothness and continuity of curvature are accounted for implicitly in the functional model. Therefore, no additional constraints have to be introduced and the least squares solution for the unknowns can be obtained from

$$\hat{\mathbf{X}} = (\mathbf{A}^T \mathbf{P} \mathbf{A})^{-1} \mathbf{A}^T \mathbf{P} \mathbf{L}. \quad (73)$$

With the residuals

$$\mathbf{v} = \mathbf{A}\hat{\mathbf{X}} - \mathbf{L} \quad (74)$$

from (71), the a posteriori estimate of the reference standard deviation

$$s_0 = \sqrt{\frac{\mathbf{v}^T \mathbf{P} \mathbf{v}}{r}} \quad (75)$$

can be computed, where r is the redundancy of the adjustment problem. Further details on the a posteriori analysis of adjustment results and on knot placement strategies can be found in the article by Ezhov et al. [1]. The fact that the solution for the unknowns (68) can be used for a transition “backwards” from B-spline to ordinary polynomial is shown in Appendix A.

6. Conclusions and Outlook

In engineering geodesy, point clouds derived from areal measurement methods, such as terrestrial laser scanning or photogrammetry, are often approximated by a continuous mathematical function for further analysis, such as deformation monitoring. In many cases, the formulas for B-spline curves and B-spline surfaces, given in the textbook by Piegl and Tiller [2] (pp. 81 and 100), are applied, where the functional values of the B-spline basis functions are recursively computed according to the formulas by de Boor [3] and Cox [4]. This approach is very easy to handle and results in a numerically stable solution for the unknowns to be determined. As these formulas have a very complex mathematical derivation, but are still very easy to use, they are mostly used like a given recipe without a deeper understanding of their derivation.

In part 1 of a series of three articles, Ezhov et al. [1] explained the basic methodology of spline approximation using splines constructed from ordinary polynomials. In this

paper (part 2) the goal was to develop an alternative derivation of the B-spline. To avoid excessive formula derivations and to illustrate the geometric relationships, quadratic splines were considered that consist of piecewise parabolic segments. Starting with the representation of a parabola in the monomial basis (ordinary polynomial) a transition to the Langrangian form was performed and, from there, to the Bernstein form, which finally resulted in the B-spline representation. In all investigations univariate splines were used, in the form of a spline function $y = f(x)$.

For the derivation of the formulas, we first considered the case of interpolation. The developed formulas were then used for spline approximation by means of least squares adjustment. With the values y_i as observations and x_i as error-free values, the resulting linear adjustment problem could be solved within the Gauss-Markov model. Finally, in Appendix A it was shown that the determined spline parameters can be used for a transition “backwards” from B-spline to ordinary polynomial. The transition “forwards” from ordinary polynomial to B-spline was shown in Appendix B.

The more general case, where both values y_i and x_i are introduced as observations into an approximation with a spline function, was already elaborated by Neitzel et al. [36]. Investigations of the numerical stability of the spline approximation approaches, based on ordinary polynomials and on truncated polynomials, discussed in part 1 and the B-splines explained in this article, as well as the potential of splines in deformation detection, will be presented in a forthcoming part 3 of this series of three articles.

Author Contributions: Conceptualization, F.N. and S.P.; methodology, N.E.; writing—original draft preparation, F.N. and N.E.; writing—review and editing, F.N. and S.P. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Acknowledgments: The authors acknowledge support from the German Research Foundation and the Open Access Publication Fund of TU Berlin.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Transition from B-spline to Ordinary Polynomial

Through Sections 2–4, it was explained how a constituent part of the B-spline can be derived from an ordinary second-degree polynomial. In Appendix A, the parameters of an ordinary second-degree spline are to be expressed as functions of the B-spline parameters. We consider the following case, cf. Figure 12:

- Three given points (x_0, y_0) , (x_1, y_1) , (x_2, y_2) are to be interpolated by a quadratic spline;
- using the given B-spline parameters C_{y_1} , C_{y_2} , i.e., the y -component of the control points.

Since the parameters of each polynomial of an ordinary second-degree spline are derived in the same way, in the text that follows, only the derivation of the first polynomial that varies within the interval $[x_0, x_1]$ is described.

From the line segments $\overline{(x_0, y_0)(x_1, C_{y_1})}$ and $\overline{(x_0, C_{y_1})(x_2, C_{y_2})}$, see Figure 12, two line equations of the form $f_{0,1}(x) = a_0 + a_1x$ resp. $f_{1,1}(x) = b_0 + b_1x$ can be derived. From (49), the slope for the first line equation is

$$a_1 = \frac{C_{y_1} - y_0}{x_1 - x_0} \quad (\text{A1})$$

and the y -intercept is

$$a_0 = y_0 - x_0 \frac{C_{y_1} - y_0}{x_1 - x_0}. \tag{A2}$$

Analogously, the parameters for the second line equation are

$$b_1 = \frac{C_{y_2} - C_{y_1}}{x_2 - x_0} \tag{A3}$$

and

$$b_0 = C_{y_1} - x_0 \frac{C_{y_2} - C_{y_1}}{x_2 - x_0}. \tag{A4}$$

The spline segment of the form of a second degree polynomial is derived by taking a combination

$$f_{0,2}(x) = \frac{x_1 - x}{x_1 - x_0} (a_0 + a_1 x) + \frac{x - x_0}{x_1 - x_0} (b_0 + b_1 x) \tag{A5}$$

of these two lines in the same way as in (50). Since x is the variable, after some rearrangement, the coefficients in front of the variable can be expressed as

$$f_{0,2}(x) = \frac{a_0 x_1 - b_0 x_0}{x_1 - x_0} + \left(\frac{b_0 - a_0 + a_1 x_1 - b_1 x_0}{x_1 - x_0} \right) x + \left(\frac{b_1 - a_1}{x_1 - x_0} \right) x^2. \tag{A6}$$

The derived coefficients of the polynomial that varies within the interval $[x_0, x_1]$ read

$$A_0 = \frac{a_0 x_1 - b_0 x_0}{x_1 - x_0}, \quad A_1 = \frac{b_0 - a_0 + a_1 x_1 - b_1 x_0}{x_1 - x_0}, \quad A_2 = \frac{b_1 - a_1}{x_1 - x_0}. \tag{A7}$$

By inserting the terms from (A7) into (A6), we obtain the ordinary second-degree polynomial

$$f_{0,2}(x) = A_0 + A_1 x + A_2 x^2. \tag{A8}$$

Only the interval $[x_0, x_1]$ is considered as the domain of the definition of the spline. The parameters of the ordinary second-degree polynomial, which varies within the interval $[x_1, x_2]$, can be derived analogously to the first one and would take the form

$$f_{1,2}(x) = B_0 + B_1 x + B_2 x^2. \tag{A9}$$

The spline function expressed by these two polynomials is identical with the one expressed by a B-spline.

A more general derivation of the B-spline representation of polynomials was developed by Lyche et al. [37] (pp. 15–18), and it was shown that polynomials can be represented in terms of B-splines of at least the same degree.

Appendix B. Transition from Ordinary Polynomial to B-spline

In Appendix A, we showed how second-degree B-spline parameters can be used to derive the parameters for an equivalent representation in the form of ordinary second-degree polynomials. In Appendix B, we explain the reverse procedure: how the ordinary quadratic spline parameters A_0, A_1, A_2 , respectively B_0, B_1, B_2 , can be used for derivation of the B-spline parameters y_0, C_1, C_2 and y_2 .

With Equation (A8), we can express each parameter of the ordinary cubic polynomial as follows:

$$A_0(x_1 - x_0) = y_0 \left(\frac{x_1^2}{x_1 - x_0} \right) + C_1 \left(-\frac{x_0 x_1}{x_1 - x_0} - \frac{x_0 x_2}{x_2 - x_0} \right) + C_2 \left(\frac{x_0^2}{x_2 - x_0} \right), \quad (\text{A10})$$

$$A_1(x_1 - x_0) = y_0 \left(-\frac{2x_1}{x_1 - x_0} \right) + C_1 \left(\frac{2x_0}{x_2 - x_0} + \frac{2x_1}{x_1 - x_0} \right) + C_2 \left(-\frac{2x_0}{x_2 - x_0} \right), \quad (\text{A11})$$

$$A_2(x_1 - x_0) = y_0 \left(\frac{1}{x_1 - x_0} \right) + C_1 \left(-\frac{1}{x_2 - x_0} - \frac{1}{x_1 - x_0} \right) + C_2 \left(\frac{1}{x_2 - x_0} \right). \quad (\text{A12})$$

According to this approach, the parameters in (A9) can be expressed as:

$$B_0(x_2 - x_1) = C_1 \left(\frac{x_2^2}{x_2 - x_0} \right) + C_2 \left(\frac{x_0 x_2}{x_2 - x_0} - \frac{x_1 x_2}{x_2 - x_1} \right) + y_2 \left(\frac{x_1^2}{x_2 - x_1} \right), \quad (\text{A13})$$

$$B_1(x_2 - x_1) = C_1 \left(-\frac{2x_2}{x_2 - x_0} \right) + C_2 \left(\frac{2x_1}{x_2 - x_1} + \frac{2x_2}{x_2 - x_0} \right) + y_2 \left(-\frac{2x_1}{x_2 - x_1} \right), \quad (\text{A14})$$

$$B_2(x_2 - x_1) = C_1 \left(\frac{1}{x_2 - x_0} \right) + C_2 \left(-\frac{1}{x_2 - x_1} - \frac{1}{x_2 - x_0} \right) + y_2 \left(\frac{1}{x_2 - x_1} \right). \quad (\text{A15})$$

From Equations (A10)–(A15), two separate equation systems can be formed to solve for the B-spline parameters. Since the equations are linear, they can be solved by applying some of the methods from linear algebra, such as Gaussian elimination.

Once the systems are solved, the results for the parameters of the system (A10)–(A12) are

$$y_0 = A_0 + A_1 x_0 + A_2 x_0^2, \quad (\text{A16})$$

$$C_1 = A_0 + A_1 \frac{x_0 + x_1}{2} + A_2 x_0 x_1, \quad (\text{A17})$$

$$C_2 = A_0 + A_1 \frac{x_1 + x_2}{2} + A_2 x_1 x_2 \quad (\text{A18})$$

and for the system (A13)–(A15), the results are

$$C_1 = B_0 + B_1 \frac{x_0 + x_1}{2} + B_2 x_0 x_1, \quad (\text{A19})$$

$$C_2 = B_0 + B_1 \frac{x_1 + x_2}{2} + B_2 x_1 x_2, \quad (\text{A20})$$

$$y_2 = B_0 + B_1 x_2 + B_2 x_2^2. \quad (\text{A21})$$

In the terms of B-spline, y_0 and y_2 are parameters and, in this derivation, they are treated as such. If y_0 and y_2 are considered as known values, we can also solve only for C_1 and C_2 as a system of two equations. However, in such case, the solutions for C_1 and C_2 are not as elegant as those from the equations above.

References

1. Ezhov, N.; Neitzel, F.; Petrovic, S. Spline Approximation—Part 1: Basic Methodology. *J. Appl. Geod.* **2018**, *12*, 139–155.
2. Piegl, L.; Tiller, W. *The NURBS Book*, 2nd ed.; Springer: Berlin/Heidelberg, Germany, 1997.
3. de Boor, C. On calculating with B-splines. *J. Approx. Theory* **1972**, *6*, 50–62.
4. Cox, M. The numerical evaluation of B-splines. *IMA J. Appl. Math.* **1972**, *10*, 134–149.
5. Lowther, J.; Shene, C.-K. Teaching B-splines is not difficult! *ACM SIGCSE Bull.* **2003**, *35*, 381–385.
6. Berkhahn, V. Geometric Modelling in Civil Engineering Informatics. Habilitation Thesis, University of Hannover, Hanover, Germany, Shaker Verlag: Aachen, Germany, 2005. (In German)

7. Farin, G. *Curves and Surfaces for CAD: A Practical Guide*, 5th ed.; Morgan Kaufmann Publishers: San Francisco, CA, USA, 2002.
8. Lobatschewsky, N. Probabilité des résultats moyens tirés d'observations répétées. *J. Die Reine Angew. Math.* **1842**, *24*, 164–170.
9. Schoenberg, I.J. Contributions to the problem of approximation of equidistant data by analytic functions, Part A: On the problem of smoothing of graduation, a first class of analytic approximation formulae. *Quart. Appl. Math.* **1946**, *4*, 45–99.
10. Schoenberg, I.J. Contributions to the problem of approximation of equidistant data by analytic functions, Part B: On the problem of osculatory interpolation, a second class of analytic approximation formulae. *Quart. Appl. Math.* **1946**, *4*, 112–141.
11. Schumaker, L. *Spline Functions: Basic Theory*, 3rd ed.; Cambridge University Press: Cambridge, UK, 2007.
12. de Boor, C.; Pinkus, A. The B-spline recurrence relations of Chakalov and of Popoviciu. *J. Approx. Theory* **2003**, *124*, 115–123.
13. Szilvasi-Nagy, M. Almost curvature continuous fitting of B-spline surfaces. *J. Geom. Graph.* **1998**, *2*, 33–43.
14. Zheng, W.; Bo, P.; Liu, Y.; Wang, W. Fast B-spline curve fitting by L-BFGS. *Comput. Aided Geom. Des.* **2012**, *29*, 448–462.
15. Wold, S. Spline functions in data analysis. *Technometrics* **1974**, *16*, 1–11.
16. Anderson, I.; Turner, D. Discrete B-spline approximation in a variety of norms. In *Advanced Mathematical and Computational Tools in Metrology V*; Ciarlini, P., Cox, M.G., Filipe, E., Pavese, F., Richter, D., Eds.; World Scientific Publishing Company: Singapore, 2001, pp. 1–15.
17. Sünkel, H. Reconstruction of functions from discrete mean values using bicubic spline functions. In *Mitteilungen der Geodätischen Institute der Technischen Universität Graz*; Contributions of the Graz Group to the XVI General Assembly of IUGG/IAG in Grenoble 1975, Folge 20; Meissl, P., Moritz, H., Rinner, K., Eds.; Graz University of Technology: Graz, Austria, 1975; pp. 267–304.
18. Sünkel, H. The representation of geodetic integral formulae by bicubic spline functions. In *Mitteilungen der Geodätischen Institute der Technischen Universität Graz*; Folge 28; Graz University of Technology: Graz, Austria, 1977. (In German)
19. Sünkel, H. Local interpolation of gravity data using spline functions. In *Tagungsbericht über das 1. Alpgravimetrie-Kolloquium*; Steinhäuser, P., Ed.; Zentralanstalt für Meteorologie und Geodynamik: Vienna, Austria, 1977; pp. 83–101. (In German)
20. Sünkel, H. *A General Surface Representation Module Designed for Geodesy (GSPP)*; Report No. 292; Department of Geodetic Science, The Ohio State University: Columbus, OH, USA, 1980.
21. Schoenberg, I.J. *Cardinal Spline Interpolation*; Regional Conference Series in Applied Mathematics, No. 12; SIAM: Philadelphia, PA, USA, 1973.
22. Schumaker, L.L. Fitting surfaces to scattered data. In *Approximation Theory II*; Lorentz, G.G., Chui, C.K., Schumaker, L.L., Eds.; Academic Press: New York, NY, USA, 1976; pp. 203–268.
23. Späth, H. *Spline Algorithms for Construction of Smooth Curves and Surfaces*; Oldenbourg: München-Vienna, Austria, 1973. (In German)
24. Jekeli, C. *Spline Representations of Functions on a Sphere for Geopotential Modeling*; Report No. 475; Department of Civil and Environmental Engineering and Geodetic Science, The Ohio State University: Columbus, OH, USA, 2005.
25. Mautz, R.; Ping, J.; Heki, K.; Schaffrin, B.; Shum, C.; Potts, L. Efficient spatial and temporal representations of global ionosphere maps over Japan using B-spline wavelets. *J. Geod.* **2005**, *78*, 660–667.
26. Koch, K.R.; Schmidt, M. N-dimensional B-spline surface estimated by lofting for locally improving IRI. *J. Geod. Sci.* **2011**, *1*, 41–51.
27. Dzapov, M.; Junasevic, M.; Lapaine, M.; Petrovic, S. Use of splines for determination of lengths from starting point along railway lines. *Geod. List.* **1985**, *39*, 99–105. (In Croatian)
28. Bureick, J.; Neuner, H.; Harmening, C.; Neumann, I. Curve and surface approximation of 3D point clouds. *Avn Allg. Vermess. Nachr.* **2016**, *123*, 315–327.
29. Kermarrec, G.; Hartmann, J.; Faust, H.; Hartmann, K.; Besharat, R.; Samuel, G.; Lixian, C.; Alkhatib, H. Understanding hierarchical B-splines with a case study: approximation of point clouds from TLS observations. *Z. Geodäsie Geoinf. Landmanagement* **2020**, *145*, 224–235.
30. Harmening, C. *Spatio-Temporal Deformation Analysis Using Enhanced B-Spline Models of Laser Scanning Point Clouds*. Ph.D. Thesis, Technische Universität Wien, Faculty of Mathematics and Geoinformation, Vienna, Austria, 2020.
31. Gander, W. Change of basis in polynomial interpolation. *Numer. Linear Algebra Appl.* **2005**, *12*, 769–778.
32. Bronshtein, I.N.; Semendiyayev, K.A.; Musiol, G.; Muehlig, H. *Handbook of Mathematics*, 5th ed.; Springer: Berlin/Heidelberg, Germany, 2007.
33. Rockafellar, R.T. *Convex Analysis*; Princeton Mathematical Series 28; Princeton University Press: Princeton, NJ, USA, 1970.
34. Bureick, J.; Alkhatib, H.; Neumann, I. Robust spatial approximation of laser scanner point clouds by means of free-form curve approaches in deformation analysis. *J. Appl. Geod.* **2016**, *10*, 27–35.
35. Niemeier, W. *Adjustment Computation, Statistical Evaluation Methods*, 2nd ed.; De Gruyter: Boston, MA, USA, 2008. (In German)
36. Neitzel, F.; Ezhov, N.; Petrovic, S. Total least squares spline approximation. *Mathematics* **2019**, *7*, 462.
37. Lyche, T.; Manni, C.; Speleers, H. B-Splines and Spline Approximation. 2017. Available online: https://www.mat.uniroma2.it/~speleers/cime2017/material/notes_lyche.pdf (accessed on 9 July 2021).