Article

Data Interpolation by Near-Optimal Splines with Free Knots Using Linear Programming

Lakshman S. Thakur¹ and Mikhail A. Bragin^{2,*} ¹ Department of Operations and Information Management, School of Business, University of Connecticut, Storrs, CT 06269, USA; thakur@uconn.edu² Department of Electrical and Computer Engineering, School of Engineering, University of Connecticut, Storrs, CT 06269, USA

* Correspondence: mikhael.bragin@uconn.edu

Abstract: The problem of obtaining an optimal spline with free knots is tantamount to minimizing derivatives of a nonlinear differentiable function over a Banach space on a compact set. While the problem of data interpolation by quadratic splines has been accomplished, interpolation by splines of higher orders is far more challenging. In this paper, to overcome difficulties associated with the complexity of the interpolation problem, the interval over which data points are defined is discretized and continuous derivatives are replaced by their discrete counterparts. The l_∞ -norm used for maximum r th order curvature (a derivative of order r) is then linearized, and the problem to obtain a near-optimal spline becomes a linear programming (LP) problem, which is solved in polynomial time by using LP methods, e.g., by using the Simplex method implemented in modern software such as CPLEX. It is shown that, as the mesh of the discretization approaches zero, a resulting near-optimal spline approaches an optimal spline. Splines with the desired accuracy can be obtained by choosing an appropriately fine mesh of the discretization. By using cubic splines as an example, numerical results demonstrate that the linear programming (LP) formulation, resulting from the discretization of the interpolation problem, can be solved by linear solvers with high computational efficiency and the resulting spline provides a good approximation to the sought-for optimal spline.

Keywords: optimal splines; linear programming; data interpolation; splines with free knots

Citation: Thakur, L.S.; Bragin, M.A. Data Interpolation by Near-Optimal Splines with Free Knots Using Linear Programming. *Mathematics* **2021**, *9*, 1099. <https://doi.org/10.3390/math9101099>

Academic Editor: Aleksandr Rakhmangulov

Received: 19 April 2021

Accepted: 8 May 2021

Published: 13 May 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 (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

When an exact functional form in a model is unknown, it is often described by measurements represented by available data points and some expected assumptions about the function. Traditionally, data points were interpolated by polynomials more frequently than by spline functions. In a basic way, spline theory deals with a simple but important question of how one uses a *lower degree piecewise* polynomial function rather than a *single larger degree* polynomial to represent given data and attendant properties. Spline theory attempts to obtain appropriate problem formulations to obtain the spline, its existence and characterization of solutions, and properties and computation of such functions. While polynomials have many desirable properties, polynomial interpolation suffers from Runge's Phenomena [1]—a polynomial oscillation at the edges of an interval. With large data, the derivatives of a single polynomial at each of the data points tend to grow, thereby resulting in multiple local extrema. In particular, it has been shown that polynomials, extensively used, for example, in thermometry [2], are not capable of recovering local behavior with good quantifiable accuracy. Spline interpolation, on the other hand, tends to greatly reduce oscillation between data points due to lower degree of polynomials. Cubic splines, with a multitude of applications, in particular, have been shown to be able to reproduce local characteristics well [2]. In addition, computationally, the problem of finding a polynomial interpolating a large number of data points is equivalent to inverting a Vandermonde matrix with a large condition number, thereby rendering the

interpolation problem intractable, inefficient, and unstable in computer implementations [3]. On the other hand, the great flexibility of splines, resulting from the use of several lower degree polynomials smoothly connected at optimally chosen points (usually referred to as “knots”), resolves the major problems mentioned above effectively. In spline applications, the following features are desired: 1. representation of the function with lower degree polynomials, which leads to efficient computations as well as a good approximation of the original function, and 2. continuous or essentially bounded derivatives (often second and third derivatives in most applications, for quadratic and cubic splines, respectively). The boundedness of derivatives in the interpolation by splines is natural since one can always have a function interpolating the given point with arbitrarily high derivatives, for example, the second derivative in the context of quadratic splines. In function approximation, therefore, to estimate the most favorable possible error in function approximation, one would like to determine the smallest curvature (the second derivative) possible of a function that interpolates the points by finding the optimal quadratic spline [4].

Many important applications require interpolation by splines. The recent and ongoing COVID-19 pandemic has spurred advances in spline-related research; specifically, splines have been shown to be useful for modeling and forecasting the spread of COVID-19 [5–8], and COVID-19 trends [9]. In [10], splines were also used to analyze the relationship between the ambient temperature and the transmission of COVID-19. Moreover, splines were shown to be useful for capturing the nonlinear relationship between fasting blood glucose levels and risk of intensive care unit admission for COVID-19 patients [11] in comparison to other conventional linear, dichotomous, or categorical methods. Optimal spline with free knots as well as more recent application areas of splines will be briefly discussed in Section 2. Section 3 provides our linear programming formulation to obtain near-optimal splines with free knots interpolating a given set of data points efficiently and with the desired level of accuracy. In Section 4, numerical results will be provided to demonstrate computational efficiency of the method developed in terms of the CPU time and accuracy, including Example 1 that shows each step and all computations involved in a five-point cubic spline problem for full clarity of each step. In Section 5, concluding remarks are given.

2. Literature Review

Spline Applications. Splines have come in common use in multiple and diverse disciplines such as robotics and statistics. Cubic splines are used in the problem of coordination and planning of the robotic motion [12]. The trajectories for robot motion are frequently described by smooth cubic splines [13]. Cubic splines ($r = 3$) are used to characterize the dependency of the Markov model statistical parameters on the conditional parameters [14]. Many filtering techniques require the use of splines. Splines are used to extract curves to model approximation to the shapes of segmented images that result from smoothing the lines of an object in the presence of its shape uncertainty [15]. In digital signal processing, splines are used to generate new sequences with a higher sampling rate (upsampling) [16]. Other recent spline applications include (1) operations research [17–19], (2) dynamic programming [20,21], (3) mathematical programming [22–24], (4) statistics [25–31], (5) control theory [31–36], (6) computer graphics [37,38], (7) path planning [31], and (8) portfolio selection under fluctuating interest rates [20,21].

General Overview of Optimal Splines with Free Knots. Spline functions were introduced as early as in the nineteenth century by Lobachevsky ([39], p. 165) and later by Schoenberg [40]. Since then, the subject has grown vigorously and many excellent texts are available on the subject: [25,40–45].

Studies have shown that the approximations by splines can be significantly improved if knots are allowed to be free rather than at a priori fixed locations (e.g., data points) [46]. When the knots of a spline are fixed a priori (for example, knots coincide with data points), it is called a spline with fixed knots [47,48]. When the knots are not fixed but are determined optimally, it is called a spline with free or variable knots. Generally, as one would expect,

the analysis for fixed knots tends to be simpler than for free knots [25] (p. 190). A spline that is required to pass through the data points exactly when data are fairly accurate is called an *interpolatory* spline; otherwise, it is an *approximating* spline. The methods given in this section focus on optimal interpolatory splines, with free knots with cubic splines being a special case. A comparative analysis showing advantages of free-knot splines over data-smoothing techniques has been given in [49]. Typically, the degree of smoothness is controlled by both the number and the position of the knots.

It has been shown in [26] that there always exists an optimal spline with $\|f^{(r)}\|_\infty = k^*$ for each polynomial comprising the spline and $(n - r - 1)$ knots, where n is the number of data points and r is the degree of the spline that interpolates them. It is noted that, for a given value k^* , an optimal spline with $\|f^{(r)}\|_\infty = k^*$ is not necessarily unique (since some of the polynomials comprising the spline (but not all) that interpolate given data points may have $\|f^{(r)}\|_\infty$ that is less than the optimal value k^*). The apparatus of the proof of the above result is complex, but a simple proof for the quadratic case $r = 2$ is given in [50].

3. Problem Formulation

3.1. General Problem Formulation

For a given set of data points $\{x_i, y_i\}_{i=1}^n$ such that $A = x_1 < \dots < x_n = B$, a spline function $S(x)$ of degree r with the knots at $x_{knot_1}, \dots, x_{knot_K}$ is a real valued function defined on the interval $[A, B]$ such that

- (i) $S(x)$ passes through all the data points, i.e., $S(x_i) = y_i$;
- (ii) In each interval $(A, x_{knot_1}), (x_{knot_1}, x_{knot_2}), \dots, (x_{knot_K}, B)$, where K is the number of knots, $S(x)$ is given by a polynomial of degree r or less;
- (iii) $S(x)$ and its derivatives of orders $1, 2, \dots, r-1$ are continuous in the interval (A, B) and
- (iv) The r th derivative of $S(x)$ can be discontinuous at a finite number of points.

Thus, a spline function of degree r is a piecewise polynomial function with $(r - 1)$ continuous derivatives. We could also say that it is a member of C^{r-1} for which the r th derivative is a step function. In other words, it is an r th order indefinite integral of a step function. Polynomial pieces of a spline are “spliced” appropriately at the knots where the r th derivative jumps from one value to another (often with opposite signs), enabling it to turn or change its shape, while all derivatives up to $(r - 1)^{th}$ order remain continuous, imparting smoothness to the spline. It should be noted that x -coordinates of given data points $\{x_i, y_i\}_{i=1}^n$ are generally not the knot points.

The problem of finding the optimal spline of r th degree is

$$\min_{f \in C^{r-1}} \|f^{(r)}\|_\infty = \min_{f \in C^{r-1}} \max |f^{(r)}| \quad (1)$$

$$\text{s.t. } f(x_i) = y_i, \quad (2)$$

where $\{x_i, y_i\}_{i=1}^n$ are given data points.

The formulation (1) and (2), though simple and elegant, is unfortunately not easy to solve. This formulation amounts to the optimization over a Banach space [51] of continuous functions on a compact set $[A, B]$, which is well-known to be notoriously difficult ([31], p. 3). Within (1) and (2), the decision variable is a function f that minimizes $\|f^{(r)}\|_\infty$. While the problem of data interpolation by quadratic splines has been accomplished [4], interpolation by splines of higher orders is far more challenging. Our key contributions to resolve this difficulty are to consider obtaining a near-optimal solution in a computationally efficient manner, and to prove that the near-optimal splines approach optimal splines as discretization mesh is refined. This will be achieved by discretizing the entire space into segments over which the optimization is performed by replacing continuous function with their discrete counterparts and by converting the resulting problem into a linear programming problem that can be solved in polynomial time by using Simplex method [52–54], as will be explained next.

3.2. Linear Optimization Formulation for Near-Optimal Solution to (1) and (2)

This section presents a numerical approach for finding near-optimal splines in maximum norm space (typically referred to as L_∞ space) since we minimize $\|f^{(r)}\|_\infty = \max |f^{(r)}|$. The main idea of the approach is to formulate a linear optimization problem, the solution of which approximates $\|f^{(r)}\|_\infty$, with certain tolerance, of the optimal spline interpolating given n data points $\{x_i, y_i\}_{i=1}^n$.

The idea is presented by using n equidistant data points (note that in the following development of the formulation, there are no requirements of the point equidistance and the data points are not restricted to be equidistant) such that $A = x_1$, $B = x_n$ and $x_i - x_{i-1} = H$, $H > 0$, $i = 2, \dots, n$, and $[A, B] = \bigcup_{i=1}^{n-1} [x_i, x_{i+1}]$. In order to discretize the problem, we chose to divide each interval $[x_i, x_{i+1}]$, $i = 1, \dots, n-1$ in m subintervals of equal width. Thus, we partition interval $[A, B]$ into l evenly spaced subintervals, where $l = (n-1)m$. Note that the number of points defining the l subintervals is $l+1$ and that the points are denoted by $\{\hat{x}_j\}_{j=1}^{(n-1)m+1=l+1}$, such that the relation $\hat{x}_{(i-1)m+1} = x_i$, $i = 1, \dots, n-1$ holds between original data points x_i and data points $\hat{x}_j$ created by subdivisions.

For example, $\hat{x}_1 = x_1 = A$, $\hat{x}_{11} = x_2$, $\hat{x}_{21} = x_3, \dots, \hat{x}_{(n-1)m+1=l+1} = x_n = B$. It should also be pointed out that the general formula for $\hat{x}_j$ is $\hat{x}_j = B \frac{(j-1)}{l} + A \frac{(1-j+l)}{l}$, so that $\hat{x}_1 = A$, $\hat{x}_2 = \frac{B}{l} + A \frac{(1+l)}{l}$, $\hat{x}_3 = \frac{2B}{l} + A \frac{(2+l)}{l}, \dots, \hat{x}_{l+1} = B$. Additionally, while H denotes the width of each interval $[x_i, x_{i+1}]$ specified in the data, h denotes the width of each smaller interval $[\hat{x}_j, \hat{x}_{j+1}]$ created after m subdivisions of each interval. In other words, $H = mh$.

After the interval $[A, B]$ is discretized, the derivatives are replaced by their discrete counterparts. Derivatives of a function are often approximated by finite differences. A discrete analog of the r th derivative of a continuous function is the r th order difference divided by h^r . A general formula of the r th order forward difference is based on r consecutive (sub)intervals and is represented as follows [55–57]

$$\Delta_h^r[f](x) = \sum_{i=0}^r (-1)^i \binom{r}{i} f(x + (r-i)h).$$

Central differences $\delta_h^r[f](x) = \sum_{i=0}^r (-1)^i \binom{r}{i} f(x + (\frac{r}{2} - i)h)$ provide more accurate approximations as compared to forward (or backward) differences: $\frac{\Delta_h^r[f](x)}{h^r} = f^{(r)}(x) + O(h)$ and $\frac{\delta_h^r[f](x)}{h^r} = f^{(r)}(x) + O(h^2)$; thus, errors in forward differences are of the order h , while in central differences, they are of the order h^2 . Essentially, this is due to considering the values of the function at $h/2$ distance away both to the left and to the right of the function in central differences rather than the entire h distance only on one side of the function in forward differences, resulting in a better approximation of the function derivative by central differences.

Briefly, the first-order difference $\frac{f(x+\frac{h}{2}) - f(x-\frac{h}{2})}{h}$ divided by h can be expanded into Taylor series in the following way:

$$\begin{aligned} \frac{f(x+\frac{h}{2}) - f(x-\frac{h}{2})}{h} &\approx \\ \left(\frac{f(x)}{h} + \frac{f'(x)}{2} + \frac{f''(x)}{2}h + \frac{1}{48}f^{(3)}(x)h^2 \right) - \left(\frac{f(x)}{h} - \frac{f'(x)}{2} + \frac{f''(x)}{2}h - \frac{1}{48}f^{(3)}(x)h^2 \right) &= \\ f'(x) + \frac{1}{24}f^{(3)}(x)h^2 \end{aligned}$$

Additionally,

$$\frac{f(x+h) - f(x)}{h} \approx \frac{f(x)}{h} + f'(x) + \frac{1}{2}f''(x)h - \frac{f(x)}{h} = f'(x) + \frac{1}{2}f''(x)h,$$

from which one can observe that $\frac{f(x+\frac{h}{2})-f(x-\frac{h}{2})}{h} - f'(x) = O(h^2)$ while $\frac{f(x+h)-f(x)}{h} - f'(x) = O(h)$. Similar arguments hold for higher order finite differences.

Even though the forward difference approximates the r th derivative at $\hat{x}_j$ with the error of the order of $O(h)$, it approximates the r th derivative at midpoint $\frac{\hat{x}_{j+1}+\hat{x}_{j+2}}{2}$ with the error of the order of $O(h^2)$, as in the discussion of central differences above. Indeed, if we denote $\hat{z}_j = \frac{\hat{x}_{j+1}+\hat{x}_{j+2}}{2}$, then

$$\frac{f(\hat{x}_j) - rf(\hat{x}_{j+1}) + \frac{r(r-1)}{2}f(\hat{x}_{j+2}) + \cdots + (-1)^r f(\hat{x}_{j+3})}{h^r} = \frac{f(\hat{z}_j - \frac{3h}{2}) - rf(\hat{z}_j - \frac{h}{2}) + \frac{r(r-1)}{2}f(\hat{z}_j + \frac{h}{2}) + \cdots + (-1)^r f(\hat{z}_j + \frac{3h}{2})}{h^r},$$

which is the r th-order central difference, and therefore, the overall accuracy of the approximation is of the order of $O(h^2)$.

Our approach to find a near-optimal solution of (1) and (2) is to minimize the maximum of the r th-order differences directly:

$$\left. \begin{array}{l} \min_{f(\hat{x}_1), f(\hat{x}_2), \dots, f(\hat{x}_{(m-1)n+1})} \max \left| \frac{\Delta_h^r[f](\hat{x}_j)}{h^r} \right| \\ \text{s.t., } f(\hat{x}_{(m-1)i+1}) = y_i, \quad i = 1, \dots, n \end{array} \right\} \quad (3)$$

Note that in this discretized version (and for the rest of the paper), points denoted by $f(\hat{x}_j)$ are decision variables, where $\hat{x}_j, j = 1, \dots, (n-1)m+1$ are endpoints of subintervals.

Since the objective function in formulation (3) is clearly nonlinear due to $\max$ and absolute value operators, in order to deal with this nonlinearity, the problem is converted into an equivalent linear counterpart (similar to ([58], p. 28)):

$$\left. \begin{array}{l} \min_{f(\hat{x}_1), f(\hat{x}_2), \dots, f(\hat{x}_{(m-1)n+1})} k \\ \text{s.t., } \frac{\Delta_h^r[f](\hat{x}_j)}{h^r} \leq k, \quad j = 1, \dots, l-r \\ \frac{\Delta_h^r[f](\hat{x}_j)}{h^r} \geq -k, \quad j = 1, \dots, l-r \\ f(\hat{x}_{(m-1)i+1}) = y_i, \quad i = 1, \dots, n \end{array} \right\} \quad (4)$$

Since $f(\hat{x}_{(m-1)i+1}) = y_i$ are given as data, the actual decision variables to determine are $f(\hat{x}_j)$, with the exception of $f(\hat{x}_{(m-1)i+1}), i = 1, \dots, n$, although for more convenient implementation, $f(\hat{x}_{(m-1)i+1})$ can still be treated as decision variables. Additionally, due to linearity, (4) can be solved with much higher efficiency than (3); regardless of the problem instance, formulation (4) remains the LP problem and the LP optimization methods converge within the polynomial time, as discussed above. The reduction of complexity is generally the goal of all linearization processes used to deal with nonlinearity from a computational perspective.

Theorem 1. The optimal value of formulation (4) approaches k^* , which is the optimal value of the formulation (1) and (2), as the number of subintervals approaches infinity ($m \rightarrow \infty$).

Proof. As established above, formulation (4)

$$\left. \begin{array}{l} \min_{f(\hat{x}_1), f(\hat{x}_2), \dots, f(\hat{x}_{(m-1)n+1})} k \\ \text{s.t., } \frac{\Delta_h^r[f](\hat{x}_j)}{h^r} \leq k, \quad j = 1, \dots, l-r \\ \frac{\Delta_h^r[f](\hat{x}_j)}{h^r} \geq -k, \quad j = 1, \dots, l-r \\ f(\hat{x}_{(m-1)i+1}) = y_i, \quad i = 1, \dots, n \end{array} \right\}$$

is equivalent to formulation (3):

$$\left. \begin{array}{l} \min_{f(\hat{x}_1), f(\hat{x}_2), \dots, f(\hat{x}_{(m-1)n+1})} \max \frac{|\Delta_h^r[f](\hat{x}_j)|}{h^r} \\ \text{s.t.}, f(\hat{x}_{(m-1)i+1}) = y_i, \quad i = 1, \dots, n \end{array} \right\}$$

As the number of subintervals approaches infinity ($m \rightarrow \infty$), the third-order difference approaches the third derivative, so formulation (4) becomes

$$\left. \begin{array}{l} \min_{f(\hat{x}_1), f(\hat{x}_2), \dots} \max |f^{(3)}(\hat{x}_j)| \\ \text{s.t.}, f(x_i) = y_i, \quad i = 1, \dots, n \end{array} \right\} \quad (5)$$

Despite having the same objective function, formulation (5) is theoretically not equivalent to the formulation of the original problem (1) and (2). Any countable (even infinitely countable) set of decision variables $\{f(\hat{x}_j)\}$ is a set of measure zero (a set of points that can be enclosed by intervals with arbitrarily small width), whereas the space over which problem (1) and (2) is optimized is continuous, namely, the entire interval $[A, B]$. In other words, formulations (1) and (2), and (5), though they have the same objective function, are optimized over different decision variable spaces, namely, discrete and continuous, respectively.

Notwithstanding the differences between formulations (1) and (2), and (5) noted above, their respective optimal values differ by an infinitesimally small value as the number of subintervals approaches infinity. Indeed, for an arbitrarily small $\varepsilon > 0$, we can always find a large enough m , the number of subintervals (of each interval), such that $|\hat{x}_j - \hat{x}_{j+1}| < \varepsilon$. Since the third derivative of f exists almost everywhere for a cubic spline by definition, we can define central differences at the midpoints of each subinterval

$(\hat{x}_j, \hat{x}_{j+1})$ as $\frac{\delta_h^r[f](\frac{\hat{x}_j + \hat{x}_{j+1}}{2})}{h^r} = f^{(3)}(x) + O(h^2)$. Now, as explained before, note that

$$\frac{\delta_h^r[f](\frac{\hat{x}_j + \hat{x}_{j+1}}{2})}{h^r} = \frac{\Delta_h^r[f](\hat{x}_{j-1})}{h^r}$$

Therefore, the optimal solution to (5) differs from the optimal solutions by a quantity of the order of $O(h^2)$. In other words, for any two arbitrarily close decision variables $f(\hat{x}_j)$ and $f(\hat{x}_{j+1})$, the respective third derivative of f evaluated at any point between them differs from the third derivative evaluated at either of these points by a quantity that is arbitrary small except possibly at the points of discontinuity of $f^{(r)}$, which constitute a set of measure zero since $f^{(r)}$ exists almost everywhere. Therefore, respective optimal values of formulations (5)

$$\left. \begin{array}{l} \min_{f(\hat{x}_1), f(\hat{x}_2), \dots} \max |f^{(r)}(\hat{x}_j)| \\ \text{s.t.}, f(x_i) = y_i, \quad i = 1, \dots, n \end{array} \right\}$$

and (1) and (2)

$$\left. \begin{array}{l} \min_{f \in C^2} \|f^{(r)}\|_{\infty} \\ \text{s.t.}, f(x_i) = y_i \end{array} \right\}$$

differ by a quantity of the order of $O(h^2)$. $\square$

4. Numerical Testing

The algorithms described above were implemented in CPLEX on an Intel Core i7 CPU Q 720 at 1.60 GHz and 4.00 GB RAM (see the CPLEX code in Appendix A).

4.1. Numerical Examples

In this section, five examples are presented. Example 1 provides a step by step walk-through of the algorithm. Examples 2–4 illustrate the performance of the algorithm in

terms of the CPU time for different test functions. Scalability as well as accuracy results are presented in Example 5. In addition, Example 5 gives computational evidence that the optimal value of formulation (4) approaches the optimal value of (1) and (2) as $O(h^2)$.

Example 1. *Walk-Through Example.* Consider a problem with five data points: $(x_1 = 1, y_1 = 3)$, $(x_2 = 2, y_2 = 5)$, $(x_3 = 3, y_3 = 4.5)$, $(x_4 = 4, y_4 = 6.6)$, and $(x_5 = 5, y_5 = 2.6)$ and let $r = 3$. We want to find a near-optimal cubic spline while illustrating all steps of the formulation very clearly and providing all basic details.

We divide each interval into two subintervals, so $n = 5, m = 2, l = (n - 1)m = 8$ and $h = 0.5$ with the resulting subintervals $[\hat{x}_1, \hat{x}_2] = [1, 1.5]$, $[\hat{x}_2, \hat{x}_3] = [1.5, 2]$, $[\hat{x}_3, \hat{x}_4] = [2, 2.5]$, $[\hat{x}_4, \hat{x}_5] = [2.5, 3]$, $[\hat{x}_5, \hat{x}_6] = [3, 3.5]$, $[\hat{x}_6, \hat{x}_7] = [3.5, 4]$, $[\hat{x}_7, \hat{x}_8] = [4, 4.5]$, and $[\hat{x}_8, \hat{x}_9] = [4.5, 5]$. Note that some of the $\hat{x}_j, j = 1, \dots, 9$ are problem data values $x_i, i = 1, \dots, 5$: $\hat{x}_1 = x_1$, $\hat{x}_3 = x_2$, $\hat{x}_5 = x_3$, $\hat{x}_7 = x_4$, and $\hat{x}_9 = x_5$. Formulation (4) takes the following form:

$$\begin{aligned} & \min_{f(\hat{x}_1), f(\hat{x}_2), f(\hat{x}_3), f(\hat{x}_4), f(\hat{x}_5), f(\hat{x}_6), f(\hat{x}_7), f(\hat{x}_8), f(\hat{x}_9)} k \\ \text{s.t. } & -k \leq \frac{f(\hat{x}_1) - 3f(\hat{x}_2) + 3f(\hat{x}_3) - f(\hat{x}_4)}{0.5^3} \leq k; \quad -k \leq \frac{f(\hat{x}_2) - 3f(\hat{x}_3) + 3f(\hat{x}_4) - f(\hat{x}_5)}{0.5^3} \leq k; \\ & -k \leq \frac{f(\hat{x}_3) - 3f(\hat{x}_4) + 3f(\hat{x}_5) - f(\hat{x}_6)}{0.5^3} \leq k; \quad -k \leq \frac{f(\hat{x}_4) - 3f(\hat{x}_5) + 3f(\hat{x}_6) - f(\hat{x}_7)}{0.5^3} \leq k; \\ & -k \leq \frac{f(\hat{x}_5) - 3f(\hat{x}_6) + 3f(\hat{x}_7) - f(\hat{x}_8)}{0.5^3} \leq k; \quad -k \leq \frac{f(\hat{x}_6) - 3f(\hat{x}_7) + 3f(\hat{x}_8) - f(\hat{x}_9)}{0.5^3} \leq k; \\ & f(\hat{x}_1) = y_1 = 3; f(\hat{x}_3) = y_2 = 5; f(\hat{x}_5) = y_3 = 4.5; f(\hat{x}_7) = y_4 = 6.6; f(\hat{x}_9) = y_5 = 2.6; \end{aligned}$$

where $\hat{x}_1 = x_1, \hat{x}_2 = \frac{x_1 + x_2}{2}, \hat{x}_3 = x_2, \hat{x}_4 = \frac{x_2 + x_3}{2}, \hat{x}_5 = x_3, \hat{x}_6 = \frac{x_3 + x_4}{2}, \hat{x}_7 = x_4, \hat{x}_8 = \frac{x_4 + x_5}{2}$, and $\hat{x}_9 = x_5$. The decision variables are: $f(\hat{x}_1), f(\hat{x}_2), f(\hat{x}_3), f(\hat{x}_4), f(\hat{x}_5), f(\hat{x}_6), f(\hat{x}_7), f(\hat{x}_8)$, and $f(\hat{x}_9)$, but $f(\hat{x}_1), f(\hat{x}_3), f(\hat{x}_5), f(\hat{x}_7)$ and $f(\hat{x}_9)$ are known to be y_1, y_2, y_3, y_4 and y_5 , respectively, so the actual decision variables are only $f(\hat{x}_2), f(\hat{x}_4), f(\hat{x}_6)$, and $f(\hat{x}_8)$. The solution is $f(\hat{x}_2) = 4.9625, f(\hat{x}_4) = 4.4125, f(\hat{x}_6) = 5.6625, f(\hat{x}_8) = 6.0125$, and $k = 10.4$.

Figures 1 and 2 present a near-optimal cubic spline, and the cubic polynomials comprising the spline. The splines are represented by discrete points, interpolating the above data points by using $m = 10$.

Figure 1. Near-optimal cubic spline for Example 1 with $m = 10$.

Example 2. *Exponential Function.* Consider a problem with $n = 20$ and $m = 50$ and a “test” function $f(x) = e^{x/10}$.

Example 3. *Trigonometric Function.* Consider a problem with $n = 20$ and $m = 50$ and a “test” function $f(x) = \sin(x)$.

Example 4. *Polynomial Function.* Consider a problem with $n = 20$ and $m = 50$ and a “test” function:

$$f(x) = \frac{98442}{34295} + \frac{2725}{157757}x + \frac{97012}{788785}x^2 - \frac{8448}{788785}x^3, x \leq 8.125$$

$$f(x) = -\frac{6798459}{34295} + \frac{671965}{157757}x - \frac{314828}{788785}x^2 + \frac{8448}{788785}x^3, x > 8.125$$

Figure 2. Polynomials comprising the near-optimal cubic spline for Example 1 with $m = 10$.

After being discretized, functions f within Examples 2–4 can be represented as 20 points $\{x_i, y_i\}_{i=1}^{20}$, which are shown in Table 1. Note that in most practical cases, the exact functional form in which the cubic spline is used to approximate is not necessarily polynomial and is unknown, so the optimal k^* of optimal splines interpolating given points may differ from the exact k of given test functions (see Table 2). When the test function is polynomial, the optimal k^* is very close to the exact k (see Example 5).

4.2. Scalability Results

To test scalability of our approach, formulation (1) and (2) is used to test problems with varying number of subintervals. The computation times are summarized in Table 3.

Example 5. *Scalability and Accuracy Testing.* Consider five data points $(1, 4)$; $(2, 15)$; $(3, 39.75)$; $(4, 78.25)$; and $(5, 124.75)$ chosen in such a way that the optimal spline has the known value of a third derivative equal to 6.

As one can see in Table 3, the algorithm scales well since increase in the computation time is almost linear for the number of subintervals. Moreover, as the number of subintervals doubles, the accuracy increases (the gap, the relative difference between $\hat{k}$ and k^* , decreases) by roughly four times.

4.3. Comparison with Other Methods

Our method is compared with the methods implemented in MATLAB R2021a by using data for Example 1. With a mesh of discretization of 0.01, the default spline interpolation [59] computes splines within 0.0156 seconds and obtains the l_∞ -norm (max norm) of the third derivative of 12.15, while our method computed splines within 0.071 seconds and obtains the third derivative of 10.767, which is 12.84% lower than the value obtained by MATLAB. The spline interpolation with “optimal” knots [60] obtains the knot location at $x = 3$, which is one of the data points, and the l_∞ -norm of the third derivative is the same as that obtained within [59]. The difference is explained by the fact that results in [60] are motivated by works as in [47,48]; therefore, optimality is with respect to knots selected from

an *a priori* given points, while within our method, the position of knots is not restricted as being fixed, thereby leading to a lower objective function.

Table 1. Data for Examples 2–4.

x	Example 2	Example 3	Example 4
$x_1 = 1$	$y_1 = 1.1051$	$y_1 = 0.8415$	$y_1 = 3$
$x_2 = 2$	$y_2 = 1.2214$	$y_2 = 0.9093$	$y_2 = 3.3113$
$x_3 = 3$	$y_3 = 1.3498$	$y_3 = 0.1411$	$y_3 = 3.74$
$x_4 = 4$	$y_4 = 1.4918$	$y_4 = -0.757$	$y_4 = 4.2219$
$x_5 = 5$	$y_5 = 1.6487$	$y_5 = -0.959$	$y_5 = 4.6928$
$x_6 = 6$	$y_6 = 1.8221$	$y_6 = -0.279$	$y_6 = 5.0883$
$x_7 = 7$	$y_7 = 2.0137$	$y_7 = 0.6570$	$y_7 = 5.3443$
$x_8 = 8$	$y_8 = 2.2255$	$y_8 = 0.9894$	$y_8 = 5.3963$
$x_9 = 9$	$y_9 = 2.4596$	$y_9 = 0.4121$	$y_9 = 5.1947$
$x_{10} = 10$	$y_{10} = 2.7183$	$y_{10} = -0.5440$	$y_{10} = 4.7732$
$x_{11} = 11$	$y_{11} = 3.0042$	$y_{11} = -1$	$y_{11} = 4.196$
$x_{12} = 12$	$y_{12} = 3.3201$	$y_{12} = -0.537$	$y_{12} = 3.5274$
$x_{13} = 13$	$y_{13} = 3.6693$	$y_{13} = 0.4202$	$y_{13} = 2.8317$
$x_{14} = 14$	$y_{14} = 4.0552$	$y_{14} = 0.9906$	$y_{14} = 2.1731$
$x_{15} = 15$	$y_{15} = 4.4817$	$y_{15} = 0.6503$	$y_{15} = 1.6159$
$x_{16} = 16$	$y_{16} = 4.9530$	$y_{16} = -0.288$	$y_{16} = 1.2244$
$x_{17} = 17$	$y_{17} = 5.4739$	$y_{17} = -0.961$	$y_{17} = 1.0628$
$x_{18} = 18$	$y_{18} = 6.0496$	$y_{18} = -0.751$	$y_{18} = 1.1953$
$x_{19} = 19$	$y_{19} = 6.6859$	$y_{19} = 0.1499$	$y_{19} = 1.6863$
$x_{20} = 20$	$y_{20} = 7.3890$	$y_{20} = 0.9129$	$y_{20} = 2.6$

Table 2. Computed optimal values and CPU time for Examples 2–4.

	CPU Time (s)	$\hat{k}$	Actual k
Example 2 (Exp.)	0.25	0.0062	0.0073891
Example 3 (Sine)	0.24	0.8947	1
Example 4 (Polyn.)	0.2	0.0645	0.0642609

Table 3. Scalability results for Example 5.

No. of Sub-Intervals (m)	$\hat{k}$	$\frac{k^* - \hat{k}}{k^*}$	CPU Time (s)
2	5.75	4.17%	0.05
4	5.93548	1.08%	0.05
8	5.98373	0.27%	0.03
16	5.99592	0.07%	0.03
32	5.99898	0.02%	0.04
64	5.99974	0.00%	0.08
128	5.99993	0.00%	0.22
256	5.99988	0.00%	0.38
512	5.99999	0.00%	0.27

5. Conclusions

Used in very diverse areas of applications, classical data interpolation by a general spline with free knots is formulated as a linear programming problem to minimize spline l_∞ -norm (max norm) of the derivative of order r , for reduced complexity, and the problem is efficiently solved by using linear programming solvers. Theoretical convergence to the true optimal splines is proven, indicating that the interpolatory data points obtained by solving the LP problem are arbitrarily close to the points on the true optimal splines. Testing results based on cubic interpolatory splines provide good approximations to optimal splines within a fast computation time. The main benefit of the new method is the significant reduction in complexity to obtain near-optimal splines with near-optimal curvature. Another advantage is that the method has a potential to be generalizable for 2D dimensional (and higher) cases, whereby “interpolatory surfaces” can be obtained by using 2D finite differences. Moreover, as indicated in the Introduction section, since the splines are expected to recover local behavior with good quantifiable accuracy, the proposed method is suitable for finding local as well as global minima of the implied underlying function quickly. The complexity involved in finding a global optimum of the resulting spline based on “merge sort” is “subquadratic” $O(n \cdot \log(n))$. A limitation of this study is that the near-optimal position of knots cannot be efficiently obtained in online implementations since the method only outputs points belonging to the near-optimal splines, although near-optimal positions of knots can be estimated offline based on the points where the third derivative (for cubic splines) changes its sign. Future research will be to obtain an exact functional representation of the optimal splines as well as the exact positions of the knots.

Author Contributions: Conceptualization, L.S.T.; methodology, M.A.B.; validation, M.A.B.; investigation, L.S.T. and M.A.B.; writing—original draft preparation, L.S.T. and M.A.B.; software, M.A.B.; writing—review and editing, L.S.T. and M.A.B. All authors have read and agreed to the published version of the manuscript.

Funding: School of Business Summer Research grant, University of Connecticut.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

Appendix A. CPLEX Code

For illustrative purposes, the CPLEX code is provided for Example 3:

```
//Data File For Example 3
```

```
nbData = 20;
```

```
nbPoints = 30;
```

```
xmin = -100;
```

```
xmax = 100;
```

```
kmin = 0;
```

```
kmax = 100;
```

```
DataPoints = [0.841470985, 0.909297427, 0.141120008, -0.756802495, -0.958924275, -0.279415498,  
0.656986599, 0.989358247, 0.412118485, -0.544021111, -0.999990207, -0.536572918, 0.420167037,  
0.990607356, 0.65028784, -0.287903317, -0.961397492, -0.750987247, 0.14987721, 0.912945251];
```

```
//Main Model
```

```
int nbPoints = ...;
```

```
int nbData = ...;
```

```
int xmin = ...;
```

```
int xmax = ...;
```

```
int kmin = ...;
```

```
int kmax = ...;
```

```

range points = 1..nbData;
range t = 1..(nbData - 1)*nbPoints+1;
dvar float x[t] in xmin..xmax;
dvar float k in kmin..kmax;
float DataPoints[points] = ...;
minimize k;
subject to {
forall (p in points) x[nbPoints*(p-1)+1] == DataPoints[p];
forall (t in 1..(nbData - 1)*nbPoints-2)
(x[t]-3*x[t+1]+3*x[t+2]-x[t+3])*nbPoints*nbPoints*nbPoints<=k;
forall (t in 1..(nbData - 1)*nbPoints-2)
(x[t]-3*x[t+1]+3*x[t+2]-x[t+3])*nbPoints*nbPoints*nbPoints>=-k;}

```

References

- Runge, C. Über empirische Funktionen und die Interpolation zwischen äquidistanten Ordinaten. *Z. Math. Phys.* **1901**, *46*, 224–243.
- Ciarlini, P.; Ichim, D. Free-knot cubic spline modelling in cryogenic thermometer calibration. *Measurement* **2006**, *39*, 815–820. [\[CrossRef\]](#)
- Lundengård, K. Generalized Vandermonde matrices and determinants in electromagnetic compatibility. Ph.D. Thesis, Mälardalen University, Vasteras, Sweden, 2018.
- Thakur, L.S. A direct algorithm for optimal quadratic splines. *Numer. Math.* **1990**, *57*, 313–332. [\[CrossRef\]](#)
- Demertzis, K.; Tsiotas, D.; Magafas, L. Modeling and forecasting the COVID-19 temporal spread in Greece: An exploratory approach based on complex network defined splines. *Int. J. Environ. Res. Public Health* **2020**, *17*, 4693. [\[CrossRef\]](#)
- Kounchev, O.; Simeonov, G.; Kuncheva, Z. The tvbg-seir spline model for analysis of covid-19 spread, and a tool for prediction scenarios. *arXiv* **2020**, arXiv:2004.11338.
- Roy, A.; Karmakar, S. Bayesian semiparametric time varying model for count data to study the spread of the COVID-19 cases. *arXiv* **2020**, arXiv:2004.02281.
- Appadu, A.R.; Kelil, A.S.; Tijani, Y.O. Comparison of some forecasting methods for COVID-19. *Alex. Eng. J.* **2021**, *60*, 1565–1589. [\[CrossRef\]](#)
- Agiwal, V.; Kumar, J.; Yip, Y.C. Study the trend pattern in COVID-19 using spline-based time series model: A bayesian paradigm. *Preprints* **2020**. [\[CrossRef\]](#)
- Wang, M.; Jiang, A.; Gong, L.; Luo, L.; Guo, W.; Li, C.; Zheng, J.; Li, C.; Yang, B.; Zeng, J.; et al. Temperature significant change COVID-19 Transmission in 429 cities. *medRxiv* **2020**. [\[CrossRef\]](#)
- Alahmad, B.; Al-Shammari, A.A.; Bennakhi, A.; Al-Mulla, F.; Ali, H. Fasting blood glucose and COVID-19 severity: Nonlinearity matters. *Diabetes Care* **2020**, *43*, 3113–3116. [\[CrossRef\]](#)
- Kolter, J.Z.; Ng, A.Y. Task-space trajectories via cubic spline optimization. In Proceedings of the 2009 IEEE International Conference on Robotics and Automation, Kobe, Japan, 12–17 May 2009; pp. 1675–1682.
- Das, A.; Naroditsky, O.; Zhiwei, Z.; Samarasekera, S.; Kumar, R. Robust visual path following for heterogeneous mobile platforms. In Proceedings of the 2010 IEEE International Conference on Robotics and Automation, Anchorage, AK, USA, 3–7 May 2010; pp. 2431–2437.
- Yu, D.; Deng, L.; Gong, Y.; Acero, A. A novel framework and training algorithm for variable-parameter hidden Markov models. *IEEE Trans. Audio Speech Lang. Process.* **2009**, *17*, 1348–1360.
- Christopoulos, V.; Schrater, P. Handling shape and contact location uncertainty in grasping two-dimensional planar objects. In Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems, San Diego, CA, USA, 29 October–2 November 2007; pp. 1557–1563.
- Mitra, S.K. *Digital Signal Processing-A Computer Based Approach*, 3rd ed.; McGraw-Hill: New York, NY, USA, 2005.
- Chen, V.C.; Ruppert, D.; Shoemaker, C.A. Applying experimental design and regression splines to high-dimensional continuous-state stochastic dynamic programming. *Oper. Res.* **1999**, *47*, 38–53. [\[CrossRef\]](#)
- De Farias, D.P.; Van Roy, B. The linear programming approach to approximate dynamic programming. *Oper. Res.* **2003**, *51*, 850–865. [\[CrossRef\]](#)
- Alizadeh, F.; Eckstein, J.; Noyan, N.; Rudolf, G. Arrival rate approximation by nonnegative cubic splines. *Oper. Res.* **2008**, *56*, 140–156. [\[CrossRef\]](#)
- Pflüger, D.; Schober, P.; Valentin, J. Solving high-dimensional dynamic portfolio choice models with hierarchical B-splines on sparse grids. *SSRN* **2019**. [\[CrossRef\]](#)
- Kirkby, L.; Deng, S. Swing option pricing by dynamic programming with b-spline density projection. *Int. J. Theor. Appl. Financ.* **2020**, *22*, 1950038. [\[CrossRef\]](#)
- Séguin, S.; Côté, P.; Audet, C. Self-scheduling short-term unit commitment and loading problem. *IEEE Trans. Power Syst.* **2016**, *31*, 133–142. [\[CrossRef\]](#)

23. Rodriguez Sarasty, J.A. Mixed-Integer Programming Approaches for Hydropower Generator Maintenance Scheduling. Ph.D. Thesis, École Polytechnique de Montréal, Montreal, QC, Canada, 2018.
24. Venter, J.V. Variable Selection in Logistic Regression Using Exact Optimisation Approaches. Ph.D. Thesis, North-West University, Potchefstroom, South Africa, 2020.
25. Nurnberger, G. *Approximation by Spline Functions*; Springer: Berlin/Heidelberg, Germany, 1989.
26. Karlin, S. Interpolation properties of generalized perfect splines and the solutions of certain Extremal problems. *Trans. Am. Math. Soc.* **1975**, *206*, 25–66. [\[CrossRef\]](#)
27. Gauthier, J.; Wu, Q.V.; Gooley, T.A. Cubic splines to model relationships between continuous variables and outcomes: A guide for clinicians. *Bone Marrow Transplant.* **2020**, *55*, 675–680. [\[CrossRef\]](#)
28. Shepherd, B.E.; Rebeiro, P.F. Assessing and interpreting the association between continuous covariates and outcomes in observational studies of HIV using splines. *J. Acquir. Immune Defic. Syndr.* **2017**, *74*, e60. [\[CrossRef\]](#) [\[PubMed\]](#)
29. Minniakhmetov, I.; Dimitrakopoulos, D.; Godoy, M. High-order spatial simulation using Legendre-like orthogonal splines. *Math. Geosci.* **2018**, *50*, 753–780. [\[CrossRef\]](#)
30. Tian, Y.; Baro, E.; Zhang, R. Performance evaluation of regression splines for propensity score adjustment in post-market safety analysis with multiple treatments. *J. Biopharm. Stat.* **2019**, *29*, 810–821. [\[CrossRef\]](#)
31. Egerstedt, M.B.; Martin, C.F. *Control Theoretic Splines: Optimal Control, Statistics, and Path Planning*; Princeton University Press: Princeton, NJ, USA, 2009.
32. Kano, H.; Egerstedt, M.; Nakata, H.; Martin, C.F. B-splines and control theory. *Appl. Math. Comput.* **2003**, *145*, 263–288. [\[CrossRef\]](#)
33. Zhang, Z.; Tomlinson, J.; Martin, C. Splines and linear control theory. *Acta Appl. Math.* **1997**, *49*, 1–34. [\[CrossRef\]](#)
34. Sun, S.; Egerstedt, M.B.; Martin, C.F. Control theoretic smoothing splines. *IEEE Trans. Autom. Control.* **2000**, *45*, 2271–2279. [\[CrossRef\]](#)
35. Balseiro, P.; Cabrera, A.; Stuchi, T.J.; Koiller, J. About simple variational splines from the Hamiltonian viewpoint. *arXiv* **2017**, arXiv:1711.02773.
36. Lai, S.; Lan, M.; Gong, K.; Chen, B.M. Axis-coupled trajectory generation for chains of integrators through smoothing splines. *Control. Theory Technol.* **2019**, *17*, 48–61. [\[CrossRef\]](#)
37. Samreen, S.; Sarfraz, M.; Jabeen, N.; Hussain, M.Z. Computer aided design using a rational quadratic trigonometric spline with interval shape control. In Proceedings of the 2017 International Conference on Computational Science and Computational Intelligence (CSCI), Las Vegas, NV, USA, 14–16 December 2017; pp. 246–251.
38. Ibraheem, F.; Hussain, M.Z. Visualization of Constrained Data Using Trigonometric Splines. In Proceedings of the 2017 21st International Conference Information Visualisation (IV), London, UK, 11–14 July 2017; pp. 400–404.
39. Renyi, A. *Wahrscheinlichkeitsrechnung*; VEB Deutscher Verlag der Wissenschaften: Berlin, Germany, 1962.
40. Schoenberg, I.J. *Approximations with Special Emphasis on Spline Functions*; Academic Press: New York, NY, USA, 1969.
41. Ahlberg, J.H.; Nilson, E.N.; Walsh, J.L. *The Theory of Splines and Their Applications*; Mathematics in Science and Engineering; Academic Press: New York, NY, USA, 1967; Volume 38.
42. Greville, T.N.E. Introduction to Spline Functions. In *Theory and Applications of Spline Functions*; University of Wisconsin: Madison, WI, USA; Academic Press: New York, NY, USA, 1969; pp. 1–35.
43. De Boor, C. *A Practical Guide to Splines*; Springer: New York, NY, USA, 1978.
44. Schumaker, L.L. *Spline Functions: Basic Theory*; Wiley: New York, NY, USA, 1981.
45. Dierckx, P. *Curve and Surface Fitting with Splines*; Clarendon Press: Oxford, UK, 1995.
46. Suchomski, P. Method of optimal variable-knot spline interpolation in the l_2 discrete norm. *Int. J. Syst. Sci.* **1991**, *22*, 2263–2274. [\[CrossRef\]](#)
47. Gaffney, P.W.; Powell, M.J.D. Optimal interpolation. In *Numerical Analysis*, Watson, G.A., Ed.; Lecture Notes in Mathematics; No. 506; Springer: Berlin/Heidelberg, Germany, 1976; pp. 90–99.
48. Micchelli, C.A.; Rivlin, T.J.; Winograd, S. The optimal recovery of smooth functions. *Numer. Math.* **1974**, *80*, 903–906. [\[CrossRef\]](#)
49. Gervini, D. Free-knot spline smoothing for functional data. *J. R. Stat. Soc. Ser. B Stat. Methodol.* **2006**, *68*, 671–687. [\[CrossRef\]](#)
50. Thakur, L.S. Uniformly Extremal Solutions in Sobolev Function Spaces for the Quadratic Case: Characterization and Applications. *SIAM J. Optim.* **1993**, *3*, 236–247. [\[CrossRef\]](#)
51. Banach, S. *Théorie des Opérations Linéaires [Theory of Linear Operations]*; Funduszu Kultury Narodowej: Warszawa, Poland, 1932.
52. Khachiyan, L.G. A polynomial algorithm in linear programming. *Dokl. Akad. Nauk SSSR* **1979**, *20*, 191–194. (In Russian) [\[CrossRef\]](#)
53. Khachiyan, L.G. Polynomial algorithms in linear programming (Zhurnal Vychisitel'noi Matematiki i Matematicheskoi Fiziki). *USSR Comput. Math. Math. Phys.* **1980**, *20*, 53–72. (In Russian) [\[CrossRef\]](#)
54. Spielman, D.A.; Teng, S.H. Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time. *J. ACM (JACM)* **2004**, *51*, 385–463. [\[CrossRef\]](#)
55. Boole, G.A. *A Treatise on The Calculus of Finite Differences*, 2nd ed.; Macmillan and Company: New York, NY, USA, 1872.
56. Levy, H.; Lessman, F. *Finite Difference Equations*; Dover Publications: Mineola, NY, USA, 1992.
57. Hildebrand, F.B. *Finite-Difference Equations and Simulations*; Prentice-Hall: Englewood Cliffs, NJ, USA, 1968.
58. Luenberger, D.G. *Linear and Nonlinear Programming*; Addison-Wesley: Reading, MA, USA, 1984.
59. Available online: <https://www.mathworks.com/help/matlab/ref/spline.html> (accessed on 29 April 2021).
60. Available online: <https://www.mathworks.com/help/curvefit/how-to-choose-knots.html> (accessed on 29 April 2021).