

1 Defining the Problem

Curve and surface modeling play a major role in Reverse Engineering during object reconstruction through CAD software [9, 18] or for CFD simulations involving moving interfaces such as flame propagation problems [11]. Usually, one has access only to a discrete (finite) data set and a smooth curve (surface) is needed in order to perform operations such as numerical integration or differentiation. Although not unique, the solution to this problem is generally sought as a spline curve (surface) that either interpolates or provides an approximation which is *closest* to the data. An approximation *fit* is desirable for experimental measurements or computational data which may be slightly inaccurate or noisy whereas an interpolating approach [15, 10] is more suitable when the data (or part of it) is known to be exact. This paper focuses on approximation techniques since the algorithm presented here was developed for curve reconstruction on numerical solutions on aircraft icing accretion [1].

Defining a spline implies fixing an approximation degree, finding its control points (net) and knot sequence. The ideal spline *fit* is a function that minimizes the error (under certain metric) employing a minimum number of knots (control points). Unfortunately, this represents a great challenge since both the “optimal” number of control points and knot location are unknown and in general, it is not possible to derive explicit formulas [6]. Thus, the solution is found by casting a suitable optimization problem, usually based on the least-squares minimization [14, 2, 19]. The methodology proposed here attempts to solve the *minimal* spline problem using a heuristic approach where knots are strategically introduced as part of an iterative scheme that drives the solution towards a desired accuracy.

The simplest solution to the spline least-squares optimization problem is to fix the number of control points, prescribe a knot sequence alongside a suitable curve parametrization and solve a linear system for the control points [14, Ch. 9.4], [3]. The usual parametrization choices are based on the chord-length or centripetal models which have proven to improve the quality of the approximation, especially for detection of sharp features such as “cusps” [5, 21, 7, 10]. On the other hand, a fixed knot sequence may be too oversimplified and shape manipulation (knot insertion/removal) [13, 23, 4] can be introduced as part of an error controlled iterative scheme. For example, one can continuously insert new knots and apply least-squares minimization (*i.e.* find new control points) until a specified tolerance is attained [16, 12]. Alternatively, one can start with a large number of control points and remove knots successively as long as the quality of the approximation is not destroyed [8, 22]. Ultimately, the problem can be formulated as a

non-linear optimization, allowing “free knots” as part of the least-squares solution [20, 1]. The free-knots problem generally yields to better results but this occurs at the expense of computational efficiency [17] and in addition, it can result in many stationary points [6].

This paper presents a new algorithm from the *knot insertion* category; starting with a minimum number of knots, we look for the span with greatest weighted error, insert a new knot and find the control points using a linear least-squares solver. If the new approximation is above the tolerance step, the knot sequence is updated. Otherwise, the step is and a new knot is inserted at the next “high-error” span. When no new knots overcome the step control, the scheme proceeds with the “best so far”, tags the knot location and when a valid step is found, attempts to remove any potential redundant knots. The algorithm stops if either the global error hits certain tolerance (1), it has reached the maximum number of iterations (2) or no new knots can improve the current approximation (3). Although we cannot prove that this strategy is optimal, our results suggest that this knot distribution leads to accurate results remaining at relatively low computational costs, thus becoming suitable for engineering applications.

In the following sections we discuss the different aspects of our method. Section ?? defines the algorithm building blocks: B-spline functions, least squares approximation and the “best knot” sequence problem. In section ?? implementation details are given and section ?? shows the results of applying this tool on several data types including curve fitting on noisy data as well as ice-layer reconstruction. Finally, in section ?? we conclude and highlight the areas for improvement as well as future applications.

2 B-splines and Fitting

Here we only provide basic details on B-spline functions and refer the reader to [14, 2, 19] for further discussion. Given a knot sequence $U = \{u_0, \dots, u_m\}$ such that $u_i \leq u_{i+1}$, $i = 0, \dots, m-1$, the i^{th} B-spline of degree p can be defined through the recurrence formula:

$$N_{i,0} = \begin{cases} 1 & \text{if } u_i \leq t \leq u_{i+1} \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

$$N_{i,p} = \frac{t - u_i}{u_{i+p} - u_i} N_{i,p-1}(t) + \frac{u_{i+p+1} - t}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1}(t). \quad (2)$$

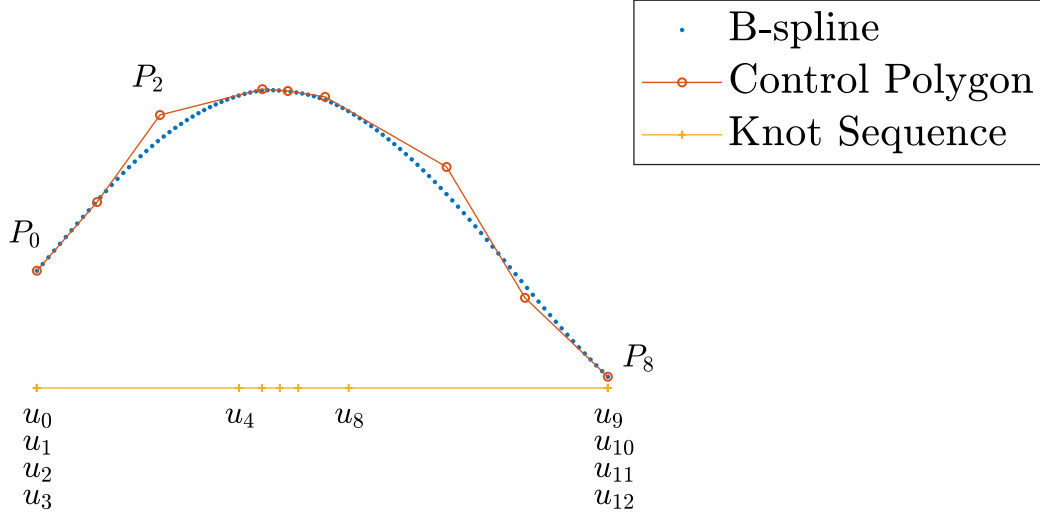


Figure 1: A cubic B-spline curve approximation to a piece of a sine wave employing nine control points and the knot sequence $U = \{u_0, \dots, u_{12}\}$.

A B-spline curve is defined as a combination of B-splines of the form:

$$C(t) = \sum_{i=0}^n N_{i,p}(t) \mathbf{P}_i, \quad \mathbf{P}_i = \text{control point.} \quad (3)$$

From this definition and as shown in Figure 1, it is clear that the approximation degree, knot sequence and control polygon fully determine a B-spline curve.

2.1 Linear Least Squares Approximation

Given a data set $\{X_i\}_{i=1}^m$ and $n > m$ control points, we look for the B-spline curve

$$C(t) = \sum_{i=0}^n N_{i,p}(t) \mathbf{P}_i, \quad t \in [0, 1] \quad (4)$$

that approximates the data in the least squares sense:

1. $C(0) = X_0, \quad C(1) = X_m$ and
2. $\sum_{k=1}^{m-1} |X_k - C(t_k)|^2$ is a minimum with respect to \mathbf{P}_i .

Here the t_k are associated to each data point (X_k) by a previously chosen parametrization.

The solution to this problem can be found as follows [14, Ch. 9.4.1]: let

$$R_k = Q_k - N_{0,p}(t_k) \cdot Q_0 - N_{m,p}(t_k) \cdot Q_m, \quad k = 1, \dots, m-1.$$

Since the B-spline knots are fixed, minimizing equation (4) gives:

$$\sum_{i=1}^{n-1} (N_{\ell,p}(t_k) N_{i,p}(t_k)) P_i = \sum_{k=1}^{m-1} N_{\ell,p}(t_k) R_k, \quad \ell = 1, \dots, n-1 \quad (5)$$

which produces a determined system of equations that can be solved applying any linear matrix solver. The only restrictions on the knot-sequence is the Whitney-condition:

The downside of a solver of this kind is that it assumes prior knowledge of the “correct” number of control points as well as the knot location. In order to overcome this, we propose an algorithm which strategically inserts new knots as part of an error controlled iterative scheme. We wish to remark that the optimization function remains linear and as a function of the control points only. The knot sequence increases within each iteration and is based on the error bisection approach presented next.

2.1.1 Knot Placement

?? A suitable knot sequence must ensure that each span contains at least one t_k value so that (5) is well posed. We start with the minimum number of control points ($n = p$) and the following *clamped* knot sequence $U = \{0, \underbrace{\dots}_{p-1}, 0, 1, \underbrace{\dots}_{p-1}, 1\}$.

Then, at each iteration, we find the span with greatest squared error:

$$e_k = \frac{1}{N_k} \sum_{i=1}^{N_k} (C(t_i) - X_i)^2, \quad N_k = \#t_i \in [u_k, u_{k+1}), \quad i = 1, \dots, m-1,$$

where the distance function is computed as follows:

$$C(t_i) - X_i = \min_{t \in [0,1]} |C(t) - X_i|, \quad (6)$$

i.e., the *actual* distance to the data. The new knot position is determined by the first t_i such that $t_i \geq \frac{e_k}{2}$ and taking the value $\tilde{u} = 0.5(t_i + t_{i+1})$. This knot choice ensures that the spans $[u_i, \tilde{u})$, $[\tilde{u}, u_{i+1})$ are non empty. Alternatively, one could find all the spans that are above the targeted tolerance and bisect each of them [14, Ch.]. However, this may lead to an excessive number of control points since it does not account for the fact that each span has

impact in more than one control point. Furthermore, a span bisection does not account for the error distribution within the span nor the fact that it can produce empty spans. Our results suggest that a strategic knot insertion consistently improves the error as shown in Figure ??.

It is well known that as the number of control points reach the number of data points ($n = m$), the spline curve might develop wiggles. Hence, the knot insertion process needs to be controlled so that “locally”, the approximation process does not attempt interpolate each data point. This is accomplished by introducing the following constraints: the new knot has to be at a *minimum* distance from its neighboring knots and initially, the minimum t_k ’s per span should be set greater than one. We say initially because for sharp features with few data support, this condition may need to be relaxed in order to for example, detect a cusp. However, such artifacts may not be visible to the Least-Squares solver since curve has is smooth along the chosen parametrization . Figure 2 illustrates this phenomena: the curve values that are fed to the solver correspond to the centripetal parametrization (smooth) and the uniform values correspond to sampling the curve uniformly around the resulting curve. Observe how the Bspline wiggles in between the centripetal knots. This is because in that region, almost all the knot spans contain only one $t - value$ resulting in a zig-zagging control polygon.

the resulting approximating B-spline curve may develop undesirable wiggles or artificial loops (self-intersections).

2.1.2 The Algorithm

Now that the least squares approach and knot placement have been discussed, we introduce the main algorithm and discuss implementation details. Once the desired tolerance is selected, we use the centripetal method in order to obtain a curve parametrization:

$$d = \sum_{k=1}^n \sqrt{|X_k - X_{k-1}|}, \quad (7)$$

$$t_0 = 0, \quad t_m = 1, \quad t_k = t_{k-1} + \frac{\sqrt{|X_k - X_{k-1}|}}{d}, \quad k = 1, \dots, m-1. \quad (8)$$

This method has proven to be suitable for detecting curve features such as “cusps” [].

Note 2.1 *If the initial number of control points is set $n > p$, we define the*

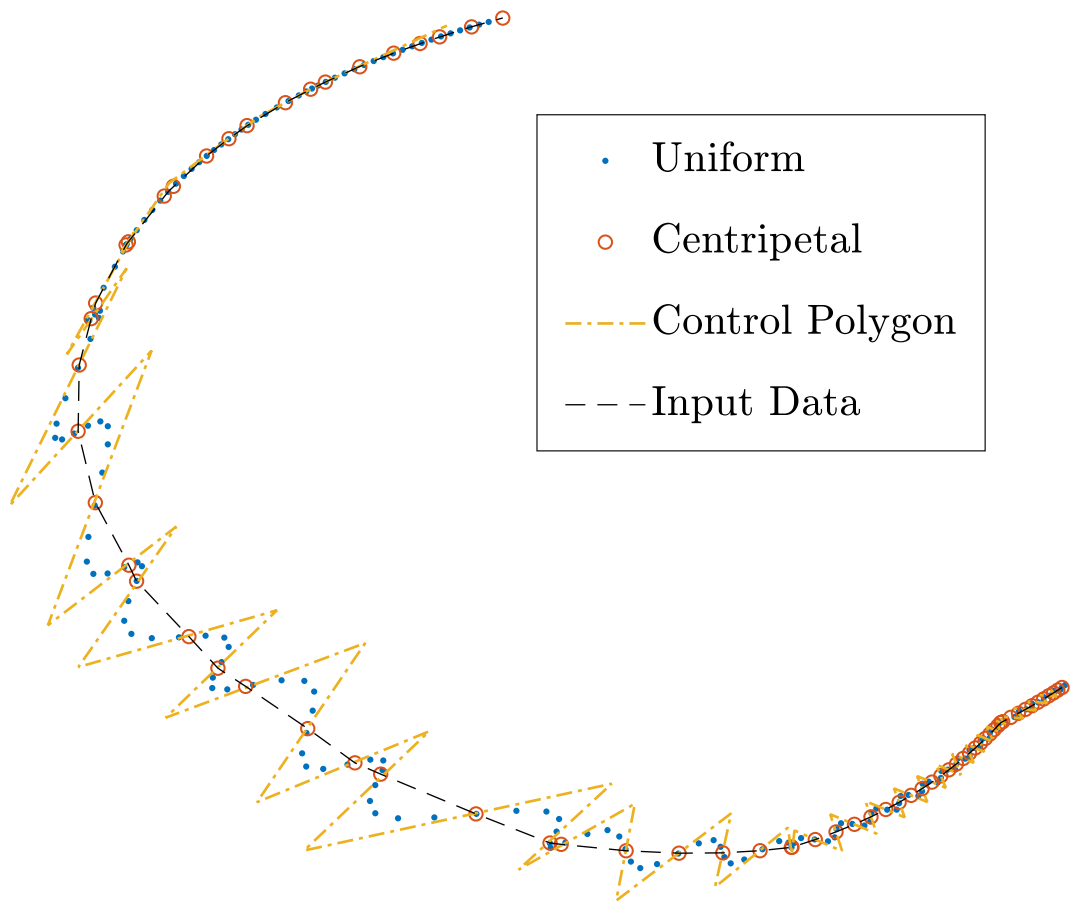


Figure 2: B-spline approximation sampled at two different parametrizations (centripetal and uniform) revealing the “hidden” wiggles as a result excessive number of control points.

internal knots sequence proposed by proposed by []:

$$d = \frac{m+1}{n-p+1} \Rightarrow i = \text{int}(j \cdot d), \quad \alpha = jd - i \quad (9)$$

$$u_{p+j} = (1 - \alpha)u_{i-1} + \alpha u_i, \quad j = 1, \dots, n - p. \quad (10)$$

At each iteration, we solve (??) and compute the global error. If we haven't reached the desired tolerance, we insert a knot as discussed in section ?? and find the new control points. If the new configuration has not significantly improved the previous iteration, we reject the knot and find the next worst error. If all knots were found to be "unsuitable" we proceed with the best configuration, insert a new knot and tag its value. Once a significant improvement has been found, we try to remove all the "redundant" points computed between the tagged knot and the knot at valid iteration and when possible, the knots in between are removed. The algorithm stops when either the tolerance has reached, the number of iterations have exceeded the maximum or when the approximation has stagnated: no new knot produces a better approximation. The implementation steps are shown in Algorithm ??.

3 The Algorithm

The work presented here provides a solution to data approximation in the least-squares sense through an iterative scheme that aims to attain a user specified accuracy. In order to avoid solving the non-linear "free knots" problem, we propose the following approach: start with few (or minimal) number of control points and perform a curve fit solving the linear least-squares problem in order to find the points location. If the error is unacceptable, locate the knot span with highest squared error and insert a new knot such that the error is split in halves. The process iterates until the desired tolerance is reached or it is not possible to introduce new suitable knots. We will discuss later what unsuitable knots mean.

are not visible to the Hence, we believe that a "knot increasing" iterative scheme is more stable than approximating the curve starting with a high number of control points and iterative remove the redundant ones. The usual approach for solving the least squares problem consists of a pre-stage that finds a suitable curve parametrization (*e.g.*, centripetal or chord-length) $t_{j=1}^m$ that targets the m data points and then compute the knots location followed by finding the control points location that minimizes the error in the least squares sense. This curve discretization does observe the curve behavior between the evaluations $B(t_j)$ and $B(t_{j+1})$, where a "parasitic" loop may

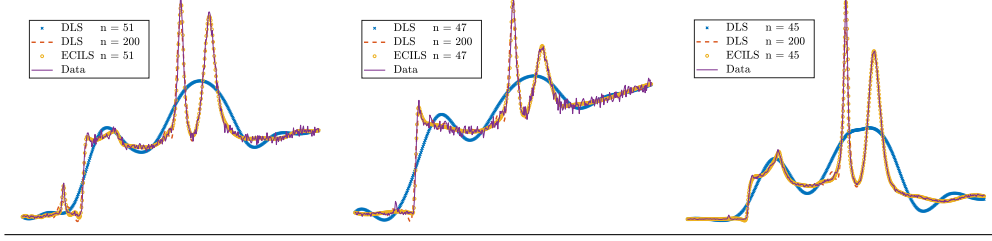


Figure 3: Three data samples from Raman Spectroscopies (*) and the resulting approximation using the new algorithm versus a direct fit for different number of control points.

develop. If the local error is smaller than the tolerance, that curve zone will not be altered (*i.e.*, candidate for knot-removal in a knot decreasing iterative scheme) thus, producing undesirable results. Furthermore, knot spans that contain few t -values can end up having a dense control point area producing wiggles. Again, from the least-squares perspective, such wiggles do not exist as illustrated in Figure ??.

3.0.1 Computational Costs

The methodology presented here attempts to avoid the “free knot problem” which involves solving a non-linear least squares solver. By sequentially increasing the number of control points together with a well chosen knot position, we can obtain a suitable approximation which relies on solving a linear system thus, avoiding excessive computational costs. Although we cannot prove that our knot position is optimal, it is designed to increase the accuracy at where the maximum error in the least squares sense occur, thus it is in accordance with the minimization solver. Furthermore as shown in the next section, our results reveal that this routine is suitable for noisy data and does not “destroy” sharp features.

4 Results

We begin by studying the performance of the solver on “basic shapes” as well as smooth data. Then we apply this routine on a level-set solution employed for studying ice formation and finally, we study its behavior on nosy data.

Figure ?? shows a helix obtained from

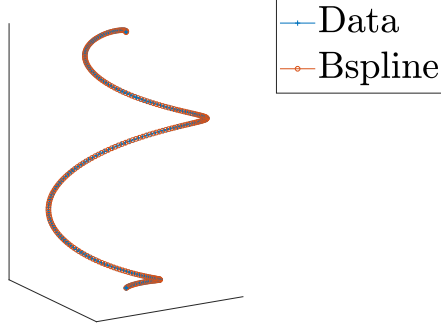
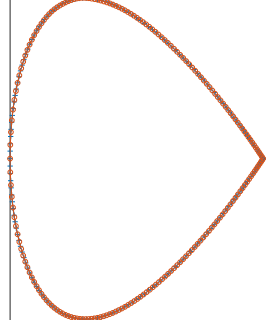
| R.M.S. | Control Points | | | |
|---------|---|-----|--|-----|
| | Sphere Curve | | NACA | |
| TOL | ECILS | DLS | ECILS | DLS |
| 1.0e-05 | 30 | 22 | 10 | 19 |
| 1.0e-06 | 52 | 36 | 19 | 35 |
| 1.0e-07 | 84 | 60 | 42 | 54 |
| |  | |  | |

Table 1: Results when approximating a curve moving along a sphere (see Figure ??) using 200 points.

4.1 Basic Shapes Approximation

Consider

References

- [1] G. Beliakov. Least Squares Splines with Free Knots: Global Optimization Approach. *Applied Mathematics and Computation*, 149(3):783 – 798, 2004.
- [2] de Boor, C. *A Practical Guide to Splines*. Applied Mathematical Sciences. Springer New York, 2001.
- [3] de Boor, C. and Rice, J. R. Least Squares Cubic Spline Approximation I–Fixed Knots. 1968.
- [4] Goldman, R.N. and Lyche, T. *Knot Insertion and Deletion Algorithms for B-Spline Curves and Surfaces*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1992.
- [5] Hoschek, J. Intrinsic Parametrization for Approximation. *Computer Aided Geometric Design*, 5(1):27–31, 1988.

- [6] Jupp, D. LB. Approximation to Data by Splines with Free Knots. *SIAM Journal on Numerical Analysis*, 15(2):328–343, 1978.
- [7] Lee, E.TY. Choosing Nodes in Parametric Curve Interpolation. *Computer-Aided Design*, 21(6):363 – 370, 1989.
- [8] Lyche, T. and Mørken, K. Knot Removal for Parametric B-Spline Curves and Surfaces. *Computer Aided Geometric Design*, 4(3):217 – 230, 1987.
- [9] Ma, W. and Kruth, J-P. NURBS Curve and Surface Fitting for Reverse Engineering. *The International Journal of Advanced Manufacturing Technology*, 14(12):918–927, Dec 1998.
- [10] Ma, W. Kruth, J-P. Parameterization of Randomly Measured Points for Least Squares Fitting of B-Spline Curves and Surfaces. *Computer-Aided Design*, 27(9):663 – 675, 1995.
- [11] Malladi, R. and Sethian, J. A. and Vemuri, B. C. Shape Modeling with Front Propagation: a Level Set Approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(2):158–175, Feb 1995.
- [12] Park, H. and Lee, J-H. B-Spline Curve Fitting Based on Adaptive Curve Refinement Using Dominant Points. *Computer-Aided Design*, 39(6):439 – 451, 2007.
- [13] Piegl, L. Modifying the Shape of Rational B-Splines. Part 1: Curves. *Computer-Aided Design*, 21(8):509 – 518, 1989.
- [14] Piegl, L. and Tiller, W. *The NURBS Book*. Monographs in Visual Communication. Springer Berlin Heidelberg, 2012.
- [15] Piegl, L.A. and Tiller, W. Computing Offsets of NURBS Curves and Surfaces. *Computer-Aided Design*, 31(2):147 – 156, 1999.
- [16] Piegl, L.A. and Tiller, W. Surface Approximation to Scanned Data. *The Visual Computer*, 16(7):386–395, Nov 2000.
- [17] Randrianarivoy, M. and Brunnett, G. Approximation by NURBS Curves with Free Knots. In Greiner, G, editor, *Vision, Modeling and Visualization*. IOS Press, November 2002.
- [18] Sarkar, B. and Mengh, C-H. Smooth-Surface Approximation and Reverse Engineering. *Computer-Aided Design*, 23(9):623 – 628, 1991.

- [19] Schumaker, L.L. *Spline Functions: Computational Methods*. Other titles in applied mathematics. Society for Industrial and Applied Mathematics (SIAM), 2015.
- [20] Schwetlick, H. and Schütze, T. Least Squares Approximation by Splines with Free Knots. *BIT Numerical Mathematics*, 35(3):361–384, 1995.
- [21] T. Speer, M. Kuppe, and J. Hoschek. Global Reparametrization for Curve Approximation. *Computer Aided Geometric Design*, 15(9):869 – 877, 1998.
- [22] Tiller, W. Knot-Removal Algorithms for NURBS Curves and Surfaces. *Computer-Aided Design*, 24(8):445 – 453, 1992.
- [23] Wolfgang, B. Inserting New Knots into B-Spline Curves. *Computer-Aided Design*, 12(4):199 – 201, 1980.