MATHEMATICS OF COMPUTATION
Volume 76, Number 258, April 2007, Pages 845–865
S 0025-5718(07)01923-0
Article electronically published on January 9, 2007

AN UNCONDITIONALLY CONVERGENT METHOD FOR COMPUTING ZEROS OF SPLINES AND POLYNOMIALS

KNUT MØRKEN AND MARTIN REIMERS

Abstract. We present a simple and efficient method for computing zeros of spline functions. The method exploits the close relationship between a spline and its control polygon and is based on repeated knot insertion. Like Newton's method it is quadratically convergent, but the new method overcomes the principal problem with Newton's method in that it always converges and no starting value needs to be supplied by the user.

1. Introduction

B-splines is a classical format for representing univariate functions and parametric curves in many applications, and the toolbox for manipulating such functions is rich, both from a theoretical and practical point of view. A commonly occurring operation is to find the zeros of a spline, and a number of methods for accomplishing this have been devised. One possibility is to use a classical method like Newton's method or the secant method [2], both of which leave us with the question of how to choose the initial guess(es). For this we can exploit a very nice feature of spline functions, namely that every spline comes with a simple, piecewise linear approximation, the *control polygon*. It is easy to show that a spline whose control polygon is everywhere of one sign cannot have any zeros. Likewise, a good starting point for an iterative procedure is a point in the neighbourhood of a zero of the control polygon. More refined methods exploit another important feature of splines, namely that the control polygon converges to the spline as the spacing of the knots (the joins between adjacent polynomial pieces) goes to zero. One can then start by inserting knots to obtain a control polygon where the zeros are clearly isolated and then apply a suitable iterative method to determine the zeros accurately. Although hybrid methods of this type can be tuned to perform well, there are important unanswered problems. Where should the knots be inserted in the initial phase? How many knots should be inserted in the initial phase? How should the starting value for the iterations be chosen? Will the method always converge?

In this paper we propose a simple method that provides answers to all the above questions. The method is very simple: Iteratively insert zeros of the control polygon as new knots of the spline. It turns out that all accumulation points of this procedure will be zeros of the spline function, and we prove below that the method is unconditionally convergent. In addition it is essentially as efficient as Newton's method, and asymptotically it behaves like this method. A similar strategy for

Received by the editor April 1, 2005 and, in revised form, November 27, 2005.
2000 *Mathematics Subject Classification.* Primary 41A15, 65D07, 65H05.

Bézier curves was used in [6]; however, no convergence analysis was given. Another related method is "Bézier clipping"; see [5]. The "Interval Newton" method proposed in [3] is, as our method, unconditionally quadratically convergent and avoids the problem of choosing an initial guess. It is based on iteratively dividing the domain into segments that may contain a zero, using estimates for the derivatives of the spline. For an overview of a number of other methods for polynomials; see [8].

2. Background spline material

The method itself is very simple, but the analysis of convergence and convergence rate requires knowledge of a number of spline topics which we summarise here.

Let $f = \sum_{i=1}^{n} c_i B_{i,d,\boldsymbol{t}}$ be a spline in the n-dimensional spline space $\mathbb{S}_{d,\boldsymbol{t}}$ spanned by the n B-splines $\{B_{i,d,\boldsymbol{t}}\}_{i=1}^{n}$. Here d denotes the polynomial degree, and the nondecreasing sequence $\boldsymbol{t} = (t_1, \ldots, t_{n+d+1})$ denotes the knots (joins between neighbouring polynomial pieces) of the spline. We assume that $t_i < t_{i+d+1}$ for $i = 1, \ldots, n$ which ensures that the B-splines are linearly independent. We also make the common assumption that the first and last $d+1$ knots are identical. This causes no loss of generality as any spline can be converted to this form by inserting the appropriate number of knots at the two ends; see page 847 below. Without this assumption the ends of the spline will have to be treated specially in parts of our algorithm.

The *control polygon* $\Gamma = \Gamma_{\boldsymbol{t}}(f)$ of f relative to $\mathbb{S}_{d,\boldsymbol{t}}$ is defined as the piecewise linear function with vertices at $(\bar{t}_i, c_i)_{i=1}^{n}$, where $\bar{t}_i = (t_{i+1} + \ldots + t_{i+d})/d$ is called the ith *knot average*. This piecewise linear function is known to approximate f itself and has many useful properties that can be exploited in analysis and development of algorithms for splines. One such property which is particularly useful when it comes to finding zeros is a spline version of Descartes' rule of signs for polynomials: A spline has at most as many zeros as its control polygon (this requires the spline to be connected; see below). More formally,

$$Z(f) \le S^-(\Gamma) = S^-(\boldsymbol{c}) \le n - 1, \tag{1}$$

where $Z(f)$ is the number of zeros, counting multiplicities, and $S^-(\Gamma)$ and $S^-(\boldsymbol{c})$ are the number of strict sign changes (ignoring zeros) in Γ and $\boldsymbol{c}$ respectively; see [4]. We say that z is a zero of f of multiplicity m if $f^{(j)}(z) = 0$ for $j = 0, \ldots, m-1$ and $f^{(m)}(z) \ne 0$. A simple zero is a zero of multiplicity $m = 1$. The inequality (1) holds for splines that are *connected*, meaning that for each $x \in [t_1, t_{n+d+1})$ there is at least one i such that $c_i B_{i,d,\boldsymbol{t}}(x) \ne 0$. This requirement ensures that f is nowhere identically zero.

The derivative of f is a spline in $\mathbb{S}_{d-1,\boldsymbol{t}}$ which can be written in the form $f' = \sum_{j=1}^{n+1} \Delta c_j B_{j,d-1,\boldsymbol{t}}$ where

$$\Delta c_j = \frac{c_j - c_{j-1}}{\bar{t}_j - \bar{t}_{j-1}} = d\,\frac{c_j - c_{j-1}}{t_{j+d} - t_j} \tag{2}$$

if we use the conventions that $c_0 = c_{n+1} = 0$ and $\Delta c_j = 0$ whenever $\bar{t}_j - \bar{t}_{j-1} = 0$. This is justified since $\bar{t}_j - \bar{t}_{j-1} = 0$ implies $t_j = t_{j+d}$ and hence $B_{j,d-1,\boldsymbol{t}} \equiv 0$. Note that the jth coefficient of f' equals the slope of segment number $j-1$ of the control-polygon of f.

The second derivative of a spline is obtained by differentiating f' which results in a spline f'' of degree $d-2$ with coefficients

$$\Delta^2 c_j = (d-1)\frac{\Delta c_j - \Delta c_{j-1}}{t_{j+d-1} - t_j}. \tag{3}$$

We will need the following well-known stability property for B-splines. For all splines $f = \sum c_i B_{i,d,\boldsymbol{t}}$ in $\mathbb{S}_{d,\boldsymbol{t}}$ the two inequalities

$$K_d^{-1}\|\boldsymbol{c}\|_\infty \le \|f\|_\infty \le \|\boldsymbol{c}\|_\infty \tag{4}$$

hold for some constant K_d that depends only on the degree d; see e.g. [4]. Here the norm of f is taken over the interval $[t_1, t_{n+d+1}]$.

Suppose that we insert a new knot x in $\boldsymbol{t}$ and form the new knot vector $\boldsymbol{t}^1 = \boldsymbol{t} \cup \{x\}$. Since $\mathbb{S}_{d,\boldsymbol{t}} \subseteq \mathbb{S}_{d,\boldsymbol{t}^1}$, we know that $f = \sum_{i=1}^n c_i B_{i,d,\boldsymbol{t}} = \sum_{i=1}^{n+1} c_i^1 B_{i,d,\boldsymbol{t}^1}$. More specifically, if $x \in [t_p, t_{p+1})$ we have $c_i^1 = c_i$ for $i = 1, \ldots, p-d$;

$$c_i^1 = (1-\mu_i)c_{i-1} + \mu_i c_i \quad \text{with} \quad \mu_i = \frac{x - t_i}{t_{i+d} - t_i} \tag{5}$$

for $i = p-d+1, \ldots, p$; and $c_i^1 = c_{i-1}$ for $i = p+1, \ldots, n+1$; see [1]. (If the new knot does not lie in the interval $[t_{d+1}, t_{n+1})$, the indices must be restricted accordingly. This will not happen when the first and last knots occur $d+1$ times as we have assumed here.) It is not hard to verify that the same relation holds for the knot averages,

$$\bar{t}_i^1 = (1-\mu_i)\bar{t}_{i-1} + \mu_i \bar{t}_i \tag{6}$$

for $i = p-d+1, \ldots, p$. This means that the corners of Γ^1, the refined control polygon, lie on Γ. This property is useful when studying the effect of knot insertion on the number of zeros of Γ.

We count the number of zeros of Γ as the number of strict sign changes in the coefficient sequence $(c_i)_{i=1}^n$. The position of a zero of the control polygon is obvious when the two ends of a line segment have opposite signs. However, the control polygon can also be identically zero on an interval in which case we associate the zero with the left end point of the zero interval. More formally, if $c_{k-1}c_{k+\ell} < 0$ and $c_{k+i} = 0$ for $i = 0, \ldots, \ell-1$, we say that k is the *index* of the zero z, which is given by

$$z = \min\bigl\{x \in [\bar{t}_{k-1}, \bar{t}_{k+\ell}] \mid \Gamma(x) = 0\bigr\}.$$

Note that in this case $c_{k-1} \neq 0$ and $c_{k-1}c_k \le 0$.

We let $S_{i,j}^-(\Gamma)$ denote the number of zeros of Γ in the half-open interval $(\bar{t}_i, \bar{t}_j]$. It is clear that $S_{1,n}^-(\Gamma) = S^-(\Gamma)$ and that $S_{i,k}^-(\Gamma) = S_{i,j}^-(\Gamma) + S_{j,k}^-(\Gamma)$ for i, j, k such that $1 \le i < j < k \le n$. We note that if Γ^1 is the control polygon of f after inserting one knot, then for any $k = 2, \ldots, n$

$$S^-(\Gamma^1) \le S^-(\Gamma) - S_{k-1,k}^-(\Gamma) + S_{k-1,k+1}^-(\Gamma^1). \tag{7}$$

To prove this we first observe that the two inequalities

$$S_{1,k-1}^-(\Gamma^1) \le S_{1,k-1}^-(\Gamma),$$
$$S_{k+1,n+1}^-(\Gamma^1) \le S_{k,n}^-(\Gamma),$$

are true since the corners of Γ^1 lie on Γ; see (5). The inequality (7) follows from the identity

$$S^-(\Gamma^1) = S_{1,k-1}^-(\Gamma^1) + S_{k-1,k+1}^-(\Gamma^1) + S_{k+1,n+1}^-(\Gamma^1),$$

and the two inequalities above.

The rest of this section is devoted to blossoming which is needed in later sections to prove convergence and establish the convergence rate of our zero finding algorithm. The ith B-spline coefficient of a spline is a continuous function of the d interior knots $t_{i+1}, \dots, t_{i+d}$ of the ith B-spline,

$$c_i = F(t_{i+1}, \dots, t_{i+d}), \tag{8}$$

and the function F is completely characterised by three properties:

- it is affine (a polynomial of degree 1) in each of its arguments,
- it is a symmetric function of its arguments,
- it satisfies the diagonal property $F(x, \dots, x) = f(x)$.

The function F is referred to as the *blossom* of the spline and is often written as $\mathcal{B}[f](y_1, \dots, y_d)$.

The affinity of the blossom means that $F(x, \dots) = ax + b$ where a and b depend on all arguments but x. From this it follows that

$$F(x, \dots) = \frac{z-x}{z-y} F(y, \dots) + \frac{x-y}{z-y} F(z, \dots). \tag{9}$$

Strictly speaking, blossoms are only defined for polynomials, so when we refer to the blossom of a spline we really mean the blossom of one of its polynomial pieces. However, the relation $c_i = \mathcal{B}[f_j](t_{i+1}, \dots, t_{i+d})$ remains true as long as f_j is the restriction of f to one of the (nonempty) intervals $[t_j, t_{j+1})$ for $j = i, i+1, \dots, i+d$.

We will use the notation $F' = \mathcal{B}[f']$ and $F'' = \mathcal{B}[f'']$ for blossoms of the derivatives of a spline, i.e.

$$\Delta c_i = F'(t_{i+1}, \dots, t_{i+d-1}) \tag{10}$$

and

$$\Delta^2 c_i = F''(t_{i+1}, \dots, t_{i+d-2}). \tag{11}$$

By differentiating (9) with respect to x we obtain

$$DF(x, \dots) = \frac{F(z, \dots) - F(y, \dots)}{z-y}. \tag{12}$$

More generally, we denote the derivative of $F(x_1, \dots, x_d)$ with respect to x_i by $D_i F$. The derivative of the blossom is related to the derivative of f by

$$d\, D_i F(y_1, \dots, y_d) = F'(y_1, \dots, y_{i-1}, y_{i+1}, \dots, y_d). \tag{13}$$

This relation follows since both sides are symmetric, affine in each of the arguments except y_i and agree on the diagonal. The latter property follows from the relation (shown here in the quadratic case)

$$\begin{aligned}
\frac{f(x+h) - f(x)}{h} &= \frac{F(x+h, x+h) - F(x,x)}{h} \\
&= \frac{F(x+h, x+h) - F(x, x+h)}{h} + \frac{F(x, x+h) - F(x,x)}{h} \\
&= D_1 F(x, x+h) + D_2 F(x,x)
\end{aligned}$$

which tends to $f'(x) = 2DF(x,x)$ when $h \to 0$.

Higher order derivatives will be indicated by additional subscripts as in $D_{i,j}F(y_1, \ldots, y_d)$, but note that the derivative is zero if differentiation with respect to the same variable is performed more than once.

We will need a nonstandard, multivariate version of Taylor's formula. This is obtained from the fundamental theorem of calculus which states that

$$g(x) = g(a) + \int_0^1 D_t g(a + t(x-a))\, dt = g(a) + (x-a)g'(a) \tag{14}$$

when g is an affine function. Suppose now that F is a function that is affine in each of its d arguments. Repeated application of (14) then leads to

$$\begin{aligned} F(y_1, \ldots, y_d) &= F(a_1, \ldots, a_d) + \sum_{i=1}^{d} (y_i - a_i) D_i F(a_1, \ldots, a_d) \\ &+ \sum_{i=2}^{d} \sum_{j=1}^{i-1} (y_i - a_i)(y_j - a_j) D_{i,j} F(a_1, \ldots, a_i, y_{i+1}, \ldots, y_d). \end{aligned} \tag{15}$$

Finally, we will need bounds for the second derivatives of the blossom in terms of the spline. Results in this direction are well-known folklore, but a proof of a specific result may be difficult to locate. We have therefore included a short proof here.

Lemma 1. *There is a constant K_{d-2} such that for all $y_1, \ldots, y_d$ and $1 \le i, j \le d$,*

$$|D_{i,j}F(y_1, \ldots, y_d)| \le \frac{K_{d-2}}{d(d-1)} \|D^2 f\|_\infty. \tag{16}$$

Proof. Two applications of (13) yield

$$D_{i,j}F(y_1, \ldots, y_d) = \frac{1}{d(d-1)} F''\big((y_1, \ldots, y_d) \setminus \{y_i, y_j\}\big).$$

By (4) we have that $\big|F''\big((y_1, \ldots, y_d) \setminus \{y_i, y_j\}\big)\big| \le K_{d-2}\|D^2 f\|_\infty$, and (16) follows. □

3. Root finding algorithm

The basic idea of the root finding algorithm is to exploit the close relationship between the control polygon and the spline, and we do this by using the zeros of the control polygon as an initial guess for the zeros of the spline. In the next step we refine the control polygon by inserting these zeros as knots. We can then find the zeros of the new control polygon, insert these zeros as knots and so on. The method can be formulated in a particularly simple way if we focus on determining the left-most zero. There is no loss in this since once the left-most zero has been found, we can split the spline at this point by inserting a knot of multiplicity d into $\boldsymbol{t}$ and then proceed with the other zeros.

Note that the case where f has a zero at the first knot t_1 can easily be detected a priori; the spline is then disconnected at t_1; see page 846 for a definition of disconnectedness. In fact, disconnected splines are degenerate, and this degeneracy is easy to detect. We therefore assume that the spline under consideration is connected.

We give a more refined version of the method in Algorithm 2 which focuses on determining an arbitrary zero of f.

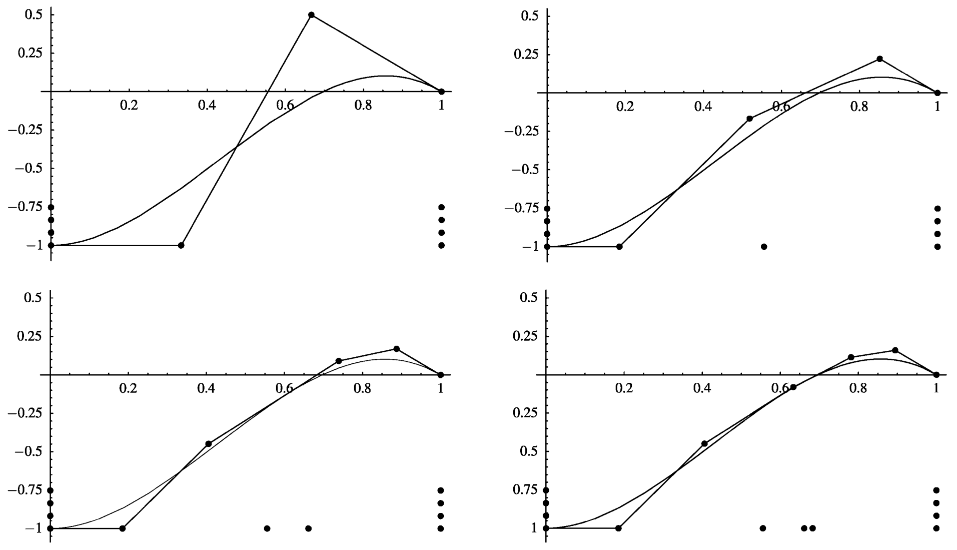

FIGURE 1. Our algorithm applied to a cubic spline with knot vector $\boldsymbol{t} = (0,0,0,0,1,1,1,1)$ and B-spline coefficients $\boldsymbol{c} = (-1,-1,1/2,0)$.

Algorithm 1. *Let f be a connected spline in $\mathbb{S}_{d,\boldsymbol{t}}$ and set $\boldsymbol{t}^0 = \boldsymbol{t}$. Repeat the following steps for $j = 0, 1, \ldots,$ until either the sequence (x_j) is found to converge or no zero of the control polygon can be found.*

1. *Determine the first zero x_{j+1} of the control polygon Γ^j of f relative to the space $\mathbb{S}_{d,\boldsymbol{t}^j}$.*
2. *Form the knot vector $\boldsymbol{t}^{j+1} = \boldsymbol{t}^j \cup \{x_{j+1}\}$.*

We will show below that if f has zeros, this procedure will converge to the first zero of f, otherwise it will terminate after a finite number of steps. A typical example of how the algorithm behaves is shown in Figure 1.

In the following, we only discuss the first iteration through Algorithm 1 and therefore omit the superscripts. In case $d = 1$ the control polygon and the spline are identical, and so the zero is found in the first iteration. We will therefore assume $d > 1$ in the rest of the paper. The first zero of the control polygon is the zero of the linear segment connecting the two points $(\bar{t}_{k-1}, c_{k-1})$ and $(\bar{t}_k, c_k)$ where k is the smallest zero index, i.e. k is the smallest integer such that $c_{k-1}c_k \leq 0$ and $c_{k-1} \neq 0$; see page 847 for the definition of zero index. The zero is characterised by the equation

$$(1-\lambda)c_{k-1} + \lambda c_k = 0$$

which has the solution

$$\lambda = \frac{-c_{k-1}}{c_k - c_{k-1}}.$$

The control polygon therefore has a zero at

$$\begin{aligned} x_1 = (1-\lambda)\bar{t}_{k-1} + \lambda\bar{t}_k &= \bar{t}_{k-1} - \frac{c_{k-1}(t_{k+d}-t_k)}{d(c_k-c_{k-1})} \\ &= \bar{t}_k - \frac{c_k(t_{k+d}-t_k)}{d(c_k-c_{k-1})}. \end{aligned} \tag{17}$$

Using the notation (2) we can write this in the simple form

$$x_1 = \bar{t}_k - \frac{c_k}{\Delta c_k} = \bar{t}_{k-1} - \frac{c_{k-1}}{\Delta c_k}, \tag{18}$$

from which it is apparent that the method described by Algorithm 1 resembles a discrete version of Newton's method.

When x_1 is inserted in $\boldsymbol{t}$, we can express f on the resulting knot vector via a new coefficient vector $\boldsymbol{c}^1$ as in (5). The new control points lie on the old control polygon, and hence this process is variation diminishing in the sense that the number of zeros of the control polygon is nonincreasing. In fact, the knot insertion step in Algorithm 1 either results in a refined control polygon that has at least one zero in the interval $(\bar{t}^1_{k-1}, \bar{t}^1_{k+1}]$ or the number of zeros in the refined control polygon has been reduced by at least 2 compared with the original control polygon.

Lemma 2. *If k is the index of a zero of Γ and $S^-_{k-1,k+1}(\Gamma^1) = 0$, then $S^-(\Gamma^1) \le S^-(\Gamma) - 2$.*

Proof. From (7) we see that the number of sign changes in Γ^1 is at least one less than in Γ, and since the number of sign changes has to decrease in even numbers the result follows. $\square$

This means that if Γ^1 has no zero in $(\bar{t}^1_{k-1}, \bar{t}^1_{k+1}]$, the zero in $(\bar{t}_{k-1}, \bar{t}_k]$ was a false warning; there is no corresponding zero in f. In fact, we have accomplished more than this since we have also removed a second false zero from the control polygon. If we still wish to find the first zero of f we can restart the algorithm from the left-most zero of the refined control polygon. However, it is useful to be able to detect that zeros in the control polygon have disappeared so we reformulate our algorithm with this ingredient. In addition, we need this slightly more elaborate algorithm to carry out a detailed convergence analysis.

Algorithm 2. *Let f be a connected spline in $\mathbb{S}_{d,\boldsymbol{t}}$, and set $\boldsymbol{t}^0 = \boldsymbol{t}$ and $\boldsymbol{c}^0 = \boldsymbol{c}$. Let $k_0 = k$ be a zero index for Γ. Repeat the following steps for $j = 0, 1, \ldots,$ or until the process is halted in step 3:*

1. *Compute $x_{j+1} = \bar{t}^j_{k_j} - c^j_{k_j}/\Delta c^j_{k_j}$.*
2. *Form the knot vector $\boldsymbol{t}^{j+1} = \boldsymbol{t}^j \cup \{x_{j+1}\}$ and compute the B-spline coefficients $\boldsymbol{c}^{j+1}$ of f in $\mathbb{S}_{d,\boldsymbol{t}^{j+1}}$.*
3. *Choose k_{j+1} to be the smallest of the two integers k_j and k_j+1, but such that k_{j+1} is the index of a zero of Γ^{j+1}. Stop if no such number can be found or if f is disconnected at x_{j+1}.*

Before turning to the analysis of convergence, we establish a few basic facts about the algorithm. We shall call an infinite sequence (x_j) generated by this algorithm a *zero sequence.* We also introduce the notation $\hat{\boldsymbol{t}}^j = (t^j_{k_j}, \ldots, t^j_{k_j+d})$ to denote what we naturally term the *active knots* at level j. In addition we denote by $a^j = \sum_{i=1}^{d-1} t^j_{k_j+i}/(d-1)$ the average of the interior, active knots.

Lemma 3. *The zero x_{j+1} computed in Algorithm 2 satisfies the relations $x_{j+1} \in (\bar{t}^j_{k_j-1}, \bar{t}^j_{k_j}] \subseteq (t^j_{k_j}, t^j_{k_j+d}]$, and if $x_{j+1} = t^j_{k_j+d}$, then f is disconnected at x_{j+1} with $f(x_{j+1}) = 0$.*

Proof. Since $c^j_{k-1} \neq 0$ we must have $x_{j+1} \in (\bar{t}^j_{k-1}, \bar{t}^j_k]$. Since we also have $(\bar{t}^j_{k-1}, \bar{t}^j_k] \subseteq (t^j_k, t^j_{k+d}]$, the first assertion follows.

For the second assertion, we observe that we always have $x_{j+1} \leq \bar{t}^j_k \leq t^j_{k+d}$. This means that if $x_{j+1} = t^j_{k+d}$ we must have $x_{j+1} = \bar{t}^j_k$ and $c^j_k = 0$. But then $x_{j+1} = t^j_{k+1} = \cdots = t^j_{k+d}$ so $f(x_{j+1}) = c^j_k = 0$. □

Our next result shows how the active knots at one level are derived from the active knots on the previous level.

Lemma 4. *If k_j, then $\hat{\boldsymbol{t}}^{j+1} = \hat{\boldsymbol{t}}^j \cup \{x_{j+1}\} \setminus \{t^j_{k_j+d}\}$. Otherwise, if $k_{j+1} = k_j + 1$, then $\hat{\boldsymbol{t}}^{j+1} = \hat{\boldsymbol{t}}^j \cup \{x_{j+1}\} \setminus \{t^j_{k_j}\}$.*

Proof. We know that $x_{j+1} \in (t^j_{k_j}, t^j_{k_j+d}]$. Therefore, if $k_{j+1} = k_j$ the latest zero x_{j+1} becomes a new active knot while $t^j_{k_j+d}$ is lost. The other case is similar. □

4. Convergence

We now have the necessary tools to prove that a zero sequence (x_j) converges; afterwards we will then prove that the limit is a zero of f.

We first show convergence of the first and last active knots.

Lemma 5. *Let (x_j) be a zero sequence. The corresponding sequence of initial active knots $(t^j_{k_j})_j$ is an increasing sequence that converges to some real number t_- from below, and the sequence of last active knots $(t^j_{k_j+d})$ is a decreasing sequence that converges to some real number t_+ from above with $t_- \leq t_+$.*

Proof. From Lemma 3 we have $x_{j+1} \in (t^j_{k_j}, t^j_{k_j+d}]$, and due to Lemma 4 we have $t^{j+1}_{k_{j+1}} \geq t^j_{k_j}$ and $t^{j+1}_{k_{j+1}+d} \leq t^j_{k_j+d}$ for each j. Since $t^j_{k_j} \leq t^j_{k_j+d}$ the result follows. □

This lemma implies that $x_j \in (t^\ell_{k_\ell}, t^\ell_{k_\ell+d}]$ for all j and ℓ such that $j > \ell$. Also, the set of intervals $\{[t^j_{k_j}, t^j_{k_j+d}]\}^\infty_{j=0}$ in which we insert the new knots is nested and these intervals tend to a limit,

$$[t_{k_0}, t_{k_0+d}] \supseteq [t^1_{k_1}, t^1_{k_1+d}] \supseteq [t^2_{k_2}, t^2_{k_2+d}] \supseteq \cdots \supseteq [t_-, t_+].$$

Proposition 6. *A zero sequence converges to either t_- or t_+.*

The proof of convergence goes via several lemmas; however in one situation the result is quite obvious: From Lemma 5 we deduce that if $t_- = t_+$, the active knots and hence the zero sequence must all converge to this number, so there is nothing to prove. We therefore focus on the case $t_- < t_+$.

Lemma 7. *None of the knot vectors $(\boldsymbol{t}^j)^\infty_{j=0}$ have knots in (t_-, t_+).*

Proof. Suppose that there is at least one knot in (t_-, t_+); by the definition of t_- and t_+ this must be an active knot for all j. Then, for all j sufficiently large, the knot $t^j_{k_j}$ will be so close to t_- and $t^j_{k_j+d}$ so close to t_+ that the two averages

$\bar{t}_{k_j-1}$ and $\bar{t}^j_{k_j}$ will both lie in (t_-, t_+). Since $x_{j+1} \in (\bar{t}^j_{k_j-1}, \bar{t}^j_{k_j}]$, this means that $x_{j+1} \in (t_-, t_+)$. As a consequence, there are infinitely many knots in (t_-, t_+). But this is impossible since for any given j only the knots $(t^j_{k_j+i})_{i=1}^{d-1}$ can possibly lie in this interval. □

Lemma 8. *Suppose $t_- < t_+$. Then there is an integer ℓ such that for all $j \geq \ell$ either $t^j_{k_j}, \dots, t^j_{k_j+d-1} \leq t_-$ and $t^j_{k_j+d} = t_+$ or $t^j_{k_j+1}, \dots, t^j_{k_j+d} \geq t_+$ and $t^j_{k_j} = t_-$.*

Proof. Let K denote the constant $K = (t_+ - t_-)/(d-1) > 0$. From Lemma 5 we see that there is an ℓ such that $t^j_{k_j} > t_- - K$ and $t^j_{k_j+d} < t_+ + K$ for $j \geq \ell$. If the lemma was not true, it would be easy to check that $\bar{t}^j_{k_j-1}$ and $\bar{t}^j_{k_j}$ would have to lie in (t_-, t_+) and hence x_{j+1} would lie in (t_-, t_+) which contradicts the previous lemma. □

Lemma 9. *Suppose that $t_- < t_+$. Then the zero sequence (x_j) and the sequences of interior active knots $(t^j_{k_j+1})$, ..., $(t^j_{k_j+d-1})$ all converge and one of the following is true: Either all the sequences converge to t_- and $x_j \leq t_-$ for j larger than some ℓ, or all the sequences converge to t_+ and $x_j \geq t_+$ for all j larger than some ℓ.*

Proof. We consider the two situations described in Lemma 8 in turn. Suppose that $t^j_{k_j}$, ..., $t^j_{k_j+d-1} \leq t_-$ for $j \geq \ell$. This means that $\bar{t}^j_{k_j} < t_+$ and since x_{j+1} cannot lie in (t_-, t_+), we must have $x_{j+1} \leq t_-$ for $j \geq \ell$. Since no new knots can appear to the right of t_+ we must have $t^j_{k_j+d} = t_+$ for $j \geq \ell$. Moreover, since $t^j_{k_j} < x_{j+1} \leq t_-$, we conclude that (x_j) and all the sequences of interior active knots converge to t_-. The proof for the case $t^j_{k_j+1}, \dots, t^j_{k_j+d} \geq t_+$ is similar. □

Lemma 9 completes the proof of Proposition 6. It remains to show that the limit of a zero sequence is a zero of f.

Lemma 10. *Any accumulation point of a zero sequence is a zero of f.*

Proof. Let z be an accumulation point for a zero sequence (x_j), and let ϵ be any positive, real number. Then there must be positive integers ℓ and k such that $t^\ell_{k+1}, \dots, t^\ell_{k+d}$ and $x_{\ell+1}$ all lie in the interval $(z - \epsilon/2, z + \epsilon/2)$. Let $\bar{t} = \bar{t}^\ell_k$ and let $\Gamma = \Gamma_{\boldsymbol{t}^\ell}(f)$ be the control polygon of f in $\mathbb{S}_{d,\boldsymbol{t}^\ell}$. We know that the derivative $f' = \sum_i \Delta c_i B_{i,d-1,\boldsymbol{t}^\ell}$ is a spline in $\mathbb{S}_{d-1,\boldsymbol{t}^\ell}$, and from (4) it follows that $\|(\Delta c_i)\|_\infty \leq K_{d-1}\|f'\|_\infty$ for some constant K_{d-1} depending only on d. From this we note that for any real numbers x and y we have the inequalities

$$\big|\Gamma(x) - \Gamma(y)\big| \leq \int_x^y \big|\Gamma'(t)\big|\, dt \leq \big\|(\Delta c_i)\big\|_\infty |y - x| \leq K_{d-1}\|f'\|_\infty |y - x|.$$

In particular, since $\Gamma(x_{\ell+1}) = 0$ it follows that

$$\big|\Gamma(\bar{t})\big| = \big|\Gamma(\bar{t}) - \Gamma(x_{\ell+1})\big| \leq K_{d-1}\|f'\|_\infty\, \epsilon.$$

In addition it follows from Theorem 4.2 in [4] that

$$\big|f(\bar{t}) - \Gamma(\bar{t})\big| \leq C(t^\ell_{k+d} - t^\ell_{k+1})^2 \leq C\epsilon^2,$$

where C is another constant depending on f and d, but not on $\boldsymbol{t}^\ell$. Combining these estimates we obtain

$$\begin{aligned}\left|f(z)\right| &\leq \left|f(z)-f(\overline{t})\right| + \left|f(\overline{t})-\Gamma(\overline{t})\right| + \left|\Gamma(\overline{t})\right| \\ &\leq \|f'\|\epsilon + C\epsilon^2 + K_{d-1}\|f'\|\epsilon.\end{aligned}$$

Since this is valid for any positive value of ϵ we must have $f(z)=0$. □

Lemmas 5, 9 and 10 lead to our main result.

Theorem 11. *A zero sequence converges to a zero of f.*

Recall that the zero finding algorithm does not need a starting value and there are no conditions in Theorem 11. On the other hand all control polygons of a spline with a zero must have at least one zero. For such splines the algorithm is therefore unconditionally convergent (for splines without zeros the algorithm will detect that the spline is of one sign in a finite number of steps).

5. Some further properties of the algorithm

Before turning to an analysis of the convergence rate of the zero finding algorithm, we need to study the limiting behavior of the algorithm more closely. Convergence implies that the coefficients of the spline converge to function values. A consequence of this is that the algorithm behaves like Newton's method in the limit.

Proposition 12. *Let (x_j) be a zero sequence converging to a zero z of f. Then at least one of c_{k_j-1} and c_{k_j} converges to $f(z)=0$ when j tends to infinity. The divided differences also converge in that $\Delta c^j_{k_j} \to f'(z)$, $\Delta^2 c^j_{k_j} \to f''(z)$ and $\Delta^2 c^j_{k_j+1} \to f''(z)$.*

Proof. We have seen that all the interior active knots $t^j_{k_j+1}, \ldots, t^j_{k_j+d-1}$ and at least one of $t^j_{k_j}$ and $t^j_{k_j+d}$ all tend to z when j tends to infinity. The first statement therefore follows from the diagonal property and the continuity of the blossom. By also applying Lemma 9 we obtain convergence of some of the divided differences. In particular we have $\Delta c^j_{k_j} = F'(t^j_{k_j+1},\ldots,t^j_{k_j+d-1}) \to f'(z)$ and similarly for $\Delta^2 c^j_{k_j}$ and $\Delta^2 c^j_{k_j+1}$. □

Our next lemma gives some more insight into the method.

Lemma 13. *Let f be a spline in $\mathbb{S}_{d,\boldsymbol{t}}$ and let $G:\mathbb{R}^d \mapsto \mathbb{R}$ denote the function*

$$G(y_1,\ldots,y_d) = \overline{y} - \frac{F(y_1,\ldots,y_d)}{F'(y_1,\ldots,y_{d-1})} \tag{19}$$

where $\overline{y}=(y_1+\cdots+y_d)/d$ and F and F' are the blossoms of f and f' respectively. Let z be a simple zero of f and let $\boldsymbol{z}_d$ be the $d-$vector $(z,\ldots,z)$. The function G has the following properties:

1. *G is continuous at all points where F' is nonzero.*
2. *$G(y,\ldots,y)=y$ if and only if $f(y)=0$.*
3. *The gradient satisfies $\nabla G(\boldsymbol{z}_d)=\boldsymbol{0}$.*
4. *$G(y_1,\ldots,y_d)$ is independent of y_d.*

Proof. Recall from equation (8) that the B-spline coefficient c_i of f in $\mathbb{S}_{d,\boldsymbol{t}}$ agrees with the blossom F of f, that is $c_i = F(t_{i+1}, \ldots, t_{i+d})$. Recall also that $c_i - c_{i-1} = D_dF(t_{i+1}, \ldots, t_{i+d})(t_{i+d} - t_i) = F'(t_{i+1}, \ldots, t_{i+d-1})/d$; see (12) and (13). This means that the computation of the new estimate for the zero in (18) can be written as

$$x_1 = \bar{t}_k - \frac{c_k}{\Delta c_k} = \bar{t}_k - \frac{F(t_{i+1}, \ldots, t_{i+d})}{d\,D_dF(t_{i+1}, \ldots, t_{i+d})} = \bar{t}_k - \frac{F(t_{i+1}, \ldots, t_{i+d})}{F'(t_{i+1}, \ldots, t_{i+d-1})}.$$

The continuity of G follows from the continuity of F. The second property of G is immendiate. To prove the third property, we omit the arguments to F and G, and as before we use the notation D_iF to denote the derivative of $F(y_1, \ldots, y_d)$ with respect to y_i. The basic iteration can then be written $G = \bar{y} - F/(d\,D_dF)$ while the derivative with respect to y_i is

$$D_iG = \frac{1}{d}\Big(1 - \frac{D_iFD_dF - FD_{i,d}F}{D_dF^2}\Big).$$

Evaluating this at $(y_1, \ldots, y_d) = \boldsymbol{z}_d$ and observing that the derivative satisfies $D_iF(\boldsymbol{z}_d) = d\,F'(\boldsymbol{z}_{d-1}) = D_dF(\boldsymbol{z}_d)$ while $F(\boldsymbol{z}_d) = 0$, we see that $D_iG(\boldsymbol{z}_d) = 0$.

To prove the last claim we observe that $D_{d,d}F(y_1, \ldots, y_d) = 0$ since F is an affine function of y_d. From this it follows that $D_dG(y_1, \ldots, y_d) = 0$. □

Since $G(y_1, \ldots, y_d)$ is independent of y_d, it is, strictly speaking, not necessary to list it as a variable. On the other hand, since y_d is required to compute the value of $G(y_1, \ldots, y_d)$ it is convenient to include it in the list.

Lemma 13 shows that a zero of f is a fixed point of G, and it is in fact possible to show that G is a kind of contractive mapping. However, we will not make use of this here. Our main use of Lemma 13 is the following immediate consequence of Property 2 which is needed later.

Corollary 14. *If $t^\ell_{k_\ell+1} = \cdots = t^\ell_{k_\ell+d-1} = z$, then $x_j = z$ for all $j > \ell$.*

We say that a sequence (y_j) is asymptotically increasing (decreasing) if there is an ℓ such that $y_j \leq y_{j+1}$ ($y_j \geq y_{j+1}$) for all $j \geq \ell$. A sequence that is asymptotically increasing or decreasing is said to be asymptotically monotone. If the inequalities are strict for all $j \geq \ell$ we say that the sequence is asymptotically strictly increasing, decreasing or monotone.

Lemma 15. *If a zero sequence is asymptotically decreasing there is some integer ℓ such that $t^j_{k_j} = t_-$ for all $j > \ell$. Similarly, if a zero sequence is asymptotically increasing there is some ℓ such that $t^j_{k_j+d} = t_+$ for all $j > \ell$.*

Proof. Suppose that (x_j) is decreasing and that $t_{k_j} < t_-$ for all j. For each i there must then be some ν such that $x_\nu \in (t^j_{k_j}, t_-)$. But this is impossible when (x_j) is decreasing and converges to either t_- or t_+. The other case is similar. □

Lemma 16. *Let (x_j) be an asymptotically increasing or decreasing zero sequence converging to a simple zero z. Then $t_- < t_+$.*

Proof. Suppose (x_j) is asymptotically decreasing, i.e., that $x_j \geq x_{j+1}$ for all j greater than some ℓ, and that $t_- = t_+ = z$. Then, according to Lemma 15, we have $t^j_{k_j} = t_-$ for large j. This implies that the active knots $t^j_{k_j}, \ldots, t^j_{k_j+d}$ all tend to z, and satisfy $z = t^j_{k_j} \leq \cdots \leq t^j_{k_j+d}$ for large j. Now consider the Taylor expansion

(15) in the special case where $a_i = z$ for all i and $y_i = t^j_{k_j+i}$ for $i = 1, \ldots, d$. Then $F(y_1, \ldots, y_d) = c^j_{k_j}$ and $f(z) = 0$ so

$$c^j_{k_j} = f'(z)(\bar{t}^j_{k_j} - z) + \sum_{i=2}^{d}\sum_{j=1}^{i-1}(y_i - z)(y_j - z)D_{i,j}F(z, \ldots, z, y_{i+1}, \ldots, y_d). \tag{20}$$

The second derivatives of F can be bounded independently of the knots in terms of the second derivative of f; see Lemma 1. This means that for sufficiently large values of j, the first term on the right in (20) will dominate. The same argument can be applied to $c^j_{k_j-1}$, and hence both $c^j_{k_j-1}$ and $c^j_{k_j}$ will have the same nonzero sign as $f'(z)$ for sufficiently large j. But this contradicts the general assumption that $c^j_{k_j-1}c^j_{k_j} \le 0$.

The case that (x_j) is asymptotically increasing is similar. $\square$

Before we continue, we need a small technical lemma.

Lemma 17. *Let $\boldsymbol{t}^1$ be a knot vector obtained by inserting a knot z into $\boldsymbol{t}$, and let $f = \sum_{j=1}^n c_j B_{j,d,\boldsymbol{t}} = \sum_{j=1}^{n+1} c^1_j B_{j,d,\boldsymbol{t}^1}$ be a spline in $\mathbb{S}_{d,\boldsymbol{t}}$ such that* sign $c_{k-1}c_k < 0$ *for some k. If $c^1_k = 0$, then*

$$z = \frac{t_{k+1} + \cdots + t_{k+d-1}}{d-1}. \tag{21}$$

Proof. If c^1_k is going to be zero after knot insertion it must obviously be a strict convex combination of the two coefficients c_{k-1} and c_k which have opposite signs. From (5) we know that the formula for computing c^1_k is

$$c^1_k = \frac{t_{k+d} - z}{t_{k+d} - t_k}c_{k-1} + \frac{z - t_k}{t_{k+d} - t_k}c_k. \tag{22}$$

But we also have the relation $(1-\lambda)(\bar{t}_{k-1}, c_{k-1}) + \lambda(\bar{t}_k, c_k) = (z, 0)$ which means that $\lambda = (z - \bar{t}_{k-1})/(\bar{t}_k - \bar{t}_{k-1})$ and

$$0 = \frac{\bar{t}_k - z}{\bar{t}_k - \bar{t}_{k-1}}c_{k-1} + \frac{z - \bar{t}_{k-1}}{\bar{t}_k - \bar{t}_{k-1}}c_k. \tag{23}$$

If $c^1_k = 0$ the weights used in the convex combinations (22) and (23) must be the same,

$$\frac{z - t_k}{t_{k+d} - t_k} = \frac{z - \bar{t}_{k-1}}{\bar{t}_k - \bar{t}_{k-1}}.$$

Solving this equation for z gives the required result. $\square$

In the normal situation where $f'(z)f''(z) \neq 0$ a zero sequence behaves very nicely.

Lemma 18. *Let (x_j) be a zero sequence converging to a zero z and suppose that f' and f'' are both continuous and nonzero at z. Then either there is an ℓ such that $x_j = z$ for all $j \ge \ell$ or (x_j) is strictly asymptotically monotone.*

Proof. We will show the result in the case where both $f'(z)$ and $f''(z)$ are positive; the other cases are similar. To simplify the notation we set $k = k_j$ in this proof. We know from the above results that the sequences $(t^j_{k+1})_j, \ldots, (t^j_{k+d-1})_j$ all converge to z. Because blossoms are continuous functions of the knots, there must be an ℓ such that for all $j \geq \ell$ we have

$$\begin{aligned}\Delta c_k^j &= F'(t_{k+1}^j, \dots, t_{k+d-1}^j) > 0,\\ \Delta^2 c_k^j &= F''(t_{k+1}^j, \dots, t_{k+d-2}^j) > 0,\\ \Delta^2 c_{k+1}^j &= F''(t_{k+2}^j, \dots, t_{k+d-1}^j) > 0.\end{aligned}$$

This implies that the control polygons $\{\Gamma^j\}$ are convex on $[\bar{t}_{k-2}^j, \bar{t}_{k+1}^j]$ and increasing on $[\bar{t}_{k-1}^j, \bar{t}_{k+1}^j]$ for $j \geq \ell$. Recall that c_k^{j+1} is a convex combination of $c_{k-1}^j < 0$ and $c_k^j \geq 0$. There are two cases to consider. If $c_k^{j+1} \geq 0$ we have $c_{k-1}^{j+1} < 0$. In other words Γ^{j+1} must be increasing and pass through zero on the interval $I = [\bar{t}_{k-1}^{j+1}, \bar{t}_k^{j+1}]$ which is a subset of $[\bar{t}_{k-2}^j, \bar{t}_k^j]$ where Γ^j is convex. But then $\Gamma^{j+1}(x) \geq \Gamma^j(x)$ on I so $x_{j+2} \leq x_{j+1}$. If on the other hand $c_k^{j+1} < 0$, then Γ^{j+1} is increasing and passes through zero on $[\bar{t}_k^{j+1}, \bar{t}_{k+1}^{j+1}]$ which is a subset of $[\bar{t}_{k-1}^j, \bar{t}_{k+1}^j]$ where Γ^j is also convex. We can therefore conclude that $x_{j+2} \leq x_{j+1}$ in this case as well.

It remains to show that either $x_{j+2} < x_{j+1}$ for all $j \geq \ell$ or $x_p = z$ from some p onwards. If for some j we have $x_{j+2} = x_{j+1}$, then this point must be common to Γ^{j+1} and Γ^j. It turns out that there are three different ways in which this may happen.

(i) The coefficient c_{k-1}^j is the last of the initial coefficients of Γ^j that is not affected by insertion of x_{j+1}. Then we must have $c_{k-1}^j = c_{k-1}^{j+1}$ and $x_{j+1} \leq \bar{t}_k^{j+1}$ which means that $x_{j+1} \in [t_{k+d-1}^j, t_{k+d}^j)$. In addition we must have $c_k^{j+1} \geq 0$ for otherwise $x_{j+2} < x_{j+1}$. But then $k_{j+1} = k$ and therefore $t_{k_{j+1}+d}^{j+1} = x_{j+1} = x_{j+2}$. From Lemma 3 we can now conclude that $x_{j+p} = z$ for all $p \geq 1$.

(ii) The coefficient $c_{k+1}^{j+1} = c_k^j$ is the first one of the later coefficients that is not affected by insertion of x_{j+1} and $x_{j+1} > \bar{t}_k^{j+1}$. This is similar to the first case.

(iii) The final possibility is that x_{j+1} lies in a part of Γ^{j+1} where all vertices are strict convex combinations of vertices of Γ^j and the convexity and monotonicity assumptions of the first part of the proof are valid. This means that the zero x_{j+1} must be a vertex of Γ^{j+1} since the interior of the line segments of Γ^{j+1} in question lie strictly above Γ^j. From Lemma 17 we see that this is only possible if $x_{j+1} = a^j = (t_{k+1}^j + \cdots + t_{k+d-1})/(d-1)$. Without loss we may assume that all the interior active knots are old x_j's, and since we know that (x_j) is asymptotically decreasing we must have $x_{j+1} \leq t_{k+1}^j \leq t_{k+d-1}^j$ for sufficiently large j. Then $x_{j+1} = a^j$ implies that $x_{j+1} = t_{k+1}^j = \ldots = t_{k+d-1}^j$ and so x_{j+1} is a fixed point of G by properties 2 and 4 in Lemma 13 . Therefore $x_i = z$ for all $i \geq j$.

Thus, if for sufficiently large j we have $x_{j+1} = x_{j+2}$, then we will also have $x_{j+p} = x_{j+1}$ for all $p > 1$. This completes the proof. □

Lemmas 16 and 18 are summed up in the following theorem.

Theorem 19. *Let (x_j) be a zero sequence converging to a zero z of f, and suppose that f' and f'' are both nonzero at z. Then there is an ℓ such that either $x_j = z$ for all $j \geq \ell$, or one of the following two statements are true:*

1. *if $f'(z)f''(z) > 0$, then $t_- < t_+ = z$ and $x_j > x_{j+1}$ for all $j \geq \ell$,*
2. *if $f'(z)f''(z) < 0$, then $z = t_- < t_+$ and $x_j < x_{j+1}$ for all $j \geq \ell$.*

Proof. We first note that if f' or f'' are not continuous at z, then there must be a knot of multiplicity at least $d-1$ at z; it is then easy to see that $x_j = z$ for all $j \geq 1$. If f' and f'' are both continuous at z we can apply Lemma 18 and Lemma 16 which proves the theorem except that we do not know the location of z. But it is impossible to have $z = t_-$ in the first case as this would require $x_j \in (t_-, t_+)$ for large j, hence $z = t_+$. The second case follows similarly. □

When $f'(z)$ and $f''(z)$ are nonzero, which accounts for the most common types of zeros, Theorem 19 gives a fairly accurate description of the behavior of the zero sequence. If $f'(z)$ is nonzero, but $f''(z)$ changes sign at z, we have observed a zero sequence to alternate on both sides of z, just like Newton's method usually does in this situation.

The main use of Theorem 19 is in the next section where we consider the convergence rate of our method; the theorem will help us to establish a strong form of quadratic convergence.

6. Convergence rate

The next task is to analyse the convergence rate of the zero finding method. Our aim is to prove that it converges quadratically, just like Newton's method. As we shall see, this is true when f' and f'' are nonzero at the zero. The development follows the same idea as is usually employed to prove quadratic convergence of Newton's method, but we work with the blossom instead of the spline itself.

We start by making use of (15) to express the error $x_1 - z$ in terms of the knots and B-spline coefficients.

Lemma 20. *Let f be spline in $\mathbb{S}_{d,\boldsymbol{t}}$ with blossom F that has a zero at z, and let x_1 be given as in* (17)*. Then*

$$x_1 - z = \frac{1}{d\,D_d F(t_{k+1}, \dots, t_{k+d})} \Big(\sum_{i=1}^{d-1} (t_{k+i} - z)^2 D_{i,d} F(t_{k+1}, \dots, t_{k+d}) + \sum_{i=2}^{d-1} \sum_{j=1}^{i-1} (t_{k+i} - z)(t_{k+j} - z) D_{i,j} F(t_{k+1}, \dots, t_{k+i}, z, \dots, z) \Big).$$

Proof. In this proof we use the shorthand notation $F_k = F(t_{k+1}, \dots, t_{k+d})$. We start by subtracting the exact zero z from both sides of (18),

$$x_1 - z = \frac{1}{d\,D_d F_k} \Big(\sum_{i=1}^{d} (t_{k+i} - z) D_d F_k - F_k \Big). \tag{24}$$

We add and subtract the linear term of the Taylor expansion (15) with $y_i = z$ and $a_i = t_{k+i}$ for $i = 1, \dots, d$, inside the brackets; this part of the right-hand side then becomes

$$-\sum_{i=1}^{d}(z-t_{k+i})D_iF_k - F_k + \sum_{i=1}^{d}(z-t_{k+i})D_iF_k - \sum_{i=1}^{d}(z-t_{k+i})D_dF_k.$$

The last two sums in this expression can be simplified so that the total becomes

$$-\sum_{i=1}^{d}(z-t_{k+i})D_iF_k - F_k + \sum_{i=1}^{d-1}(z-t_{k+i})(t_{k+d}-t_{k+i})D_{i,d}F_k. \tag{25}$$

We now make use of the Taylor expansion (15),

$$\begin{aligned} f(x) = F(x,\dots,x) = F_k &+ \sum_{i=1}^{d}(x-t_{k+i})D_iF_k \\ &+ \sum_{i=2}^{d}\sum_{j=1}^{i-1}(x-t_{k+i})(x-t_{k+j})D_{i,j}F(t_{k+1},\dots,t_{k+i},x,\dots,x), \end{aligned}$$

and set x equal to the zero z so that the left-hand side of this equation is zero. We can then replace the first two terms in (25) with the quadratic error term in this Taylor expansion. The total expression in (25) then becomes

$$\sum_{i=2}^{d}\sum_{j=1}^{i-1}(z-t_{k+i})(z-t_{k+j})D_{i,j}F(t_{k+1},\dots,t_{k+i},z,\dots,z) \\ + \sum_{i=1}^{d-1}(z-t_{k+i})(t_{k+d}-t_{k+i})D_{i,d}F_k.$$

The terms in the double sum corresponding to $i=d$ can be combined with the sum in the second line, and this yields

$$\sum_{i=2}^{d-1}\sum_{j=1}^{i-1}(z-t_{k+i})(z-t_{k+j})D_{i,j}F(t_{k+1},\dots,t_{k+i},z,\dots,z) \\ + \sum_{i=1}^{d-1}(z-t_{k+i})^2D_{i,d}F_k.$$

Replacing the terms inside the bracket in (24) with this expression gives the required result. $\square$

We are now ready to show quadratic convergence to a simple zero.

Theorem 21. *Let (x_j) be a zero sequence converging to a zero z and suppose that $f'(z) \neq 0$. Then there is a constant C depending on f and d but not on $\boldsymbol{t}$, such that for sufficiently large j we have*

$$|x_{j+1}-z| \le C \max_{i=1,\dots,d-1} |t^j_{k_j+i}-z|^2.$$

Proof. From Lemma 9 we have that the sequences of interior active knots $(t^j_{k_j+1})_j$, $\dots$, $(t^j_{k_j+d-1})_j$ all converge to z. Therefore, there is an ℓ such that for $j \ge \ell$, the denominator in Lemma 20 satisfies $\left|d\,D_dF(t^j_{k_j+1},\dots,t^j_{k_j+d})\right| > M$ for some $M > 0$ independent of $\boldsymbol{t}$. Let j be an integer greater than ℓ and let $\delta_j =$

$\max_{i=1,\dots,d-1} |t^j_{k_j+i} - z|$. Using (16) we can estimate the error from Lemma 20 with $x_1 = x_{j+1}$ as

$$|x_{j+1} - z| \le \frac{\delta_j^2 d(d-1) \max_{p,q} |D_{p,q}F|}{2M} \le C\delta_j^2,$$

where $C = K_{d-2}\|D^2 f\|_\infty/(2M)$. □

This result can be strengthened in case a zero sequence (x_j) converges monotonically to z.

Theorem 22. *Let (x_j) be a zero sequence converging to a zero z, suppose that $f'(z)$ and $f''(z)$ are both nonzero, and set $e_n = |x_n - z|$. Then there is a constant C depending on f and d, but not on $\boldsymbol{t}$, such that for sufficiently large j we have*

$$e_{j+1} \le Ce^2_{j-d+2}.$$

Proof. From Theorem 19 we know that (x_j) is asymptotically monotone. There is therefore an ℓ such that for $j \ge \ell$ we have $\max_{i=1,\dots,d-1} |t_{k_j+i} - z| = e_{j-d+2}$. The result then follows from Theorem 21. □

This result implies that if we sample the error in Algorithm 2 after every $d-1$ knot insertions, the resulting sequence converges quadratically to zero.

7. Stability

In this section we briefly discuss the stability properties of Algorithm 2. It is well known that a situation where large rounding errors may occur is when a small value is computed from relatively large values. Computing zeros of functions fall in this category as we need to compute values of the function near the zero, while the function is usually described by reasonably large parameters. For example, spline functions are usually given by reasonably large values of the knots and B-spline coefficients, but near a zero these numbers combine such that the result is small. It is therefore particularly important to keep an eye on rounding errors when computing zeros.

Our method consists of two parts where rounding errors potentially may cause problems, namely the computation of x_{j+1} by the first step in Algorithm 2 and the computation of the new B-spline coefficients in step 2. Let us consider each of these steps in turn.

The new estimate for the zero is given by the formula

$$x_{j+1} = \overline{t}^j_{k_j} - \frac{c^j_{k_j}(t^j_{k_j+d} - t^j_{k_j})}{(c^j_{k_j} - c^j_{k_j-1})d},$$

which is in fact a convex combination of the two numbers $\overline{t}^j_{k_j-1}$ and $\overline{t}^j_{k_j}$; see (17). Recall that $c^j_{k_j-1}$ and $c^j_{k_j}$ have opposite signs while $t^j_{k_j}$ and $t^j_{k_j+d}$ are usually well separated so the second term on the right can usually be computed without much cancellation. This estimate x_{j+1} is then inserted as a new knot, and new coefficients are computed via (5) as a series of convex combinations. Convex combinations are generally well suited to floating-point computations except when combining two numbers of opposite signs to obtain a number near zero. This can potentially happen when computing the new coefficient

$$c^{j+1}_k = (1-\mu_k)c^j_{k-1} + \mu_k c^j_k,$$

since we know that c^j_{k-1} and c^j_k have opposite signs. However, it turns out that in most cases, the magnitude of one of the two coefficients tends to zero with j, whereas the other one remains bounded away from zero.

Proposition 23. *Let z be the first zero of f and suppose that $f'(z) \neq 0$ and $f''(z) \neq 0$. Then one of $c^j_{k_j}$ and $c^j_{k_j-1}$ will tend to zero while the other will tend to $f'(z)(t_+ - t_-)/d \neq 0$.*

Proof. When the first two derivatives are nonzero we know from Theorem 19 that $t_- < t_+$, and $t^j_{k_j+1}, \ldots, t^j_{k_j+d-1}$ will all tend to either t_- or t_+. For this proof we assume that they tend to t_-; the other case is similar. Then $\lim_{j\to\infty} c^j_{k_j-1} = F(t^j_{k_j}, \ldots, t^j_{k_j+d-1}) = f(z) = 0$, while

$$\lim_{j\to\infty} \frac{c^j_{k_j} - c^j_{k_j-1}}{t^j_{k_j+d} - t^j_{k_j}} = \lim_{j\to\infty} F'(t^j_{k_j+1}, \ldots, t^j_{k_j+d-1})/d = f'(z)/d.$$

Since $t^j_{k_j+d} - t^j_{k_j} \to t_+ - t_-$ and $c^j_{k_j-1} \to 0$, we must have that $c^j_{k_j} \to f'(z)(t_+ - t_-)/d$ which is nonzero. □

This result ensures that the most critical convex combination usually behaves nicely, so in most cases there should not be problems with numerical stability. This corresponds well with our practical experience. However, as with Newton's method and many others, we must expect the numerical performance to deteriorate when $f'(z)$ becomes small.

8. Implementation and numerical examples

Our algorithm is very simple to implement and does not require any elaborate spline software. To illustrate this fact we provide pseudo code for an algorithm to compute the smallest zero of a spline, returned in the variable `x`. The knots `t` and the coefficients `c` are stored in vectors (indexed from 1). For efficiency the algorithm overwrites the old coefficients with the new ones during knot insertion.

Pseudo code for Algorithm 1.

```
// Connected spline of degree d
// with knots t and coefficients c given
if (c(1)==0) return t(1);
k=2;
for  (it = 1; it<=max_iterations; it++) {

  // Compute the index of the smallest zero
  // of the control polygon
  n = size(c);
  while  (k<=n AND (c(k-1)*c(k)>0 )) k++;
  if (k>n) return NO_ZERO;

  // Find zero of control polygon and check convergence
  x = knotAverage(t,d,k)
                     - c(k) * (t(k+d)-t(k))/(c(k)-c(k-1))/d;
  xlist.append(x);
  if ( converged(t,d,xlist) ) return x;

  // Refine spline by Boehms algorithm
  mu = k;
  while  (x>=t(mu+1)) mu++;
  c.append(c(n)); // Length of c increased by one
  for (i=n; i>=mu+1; i--) c(i) = c(i-1);
  for (i=mu ; i>=mu-d+1; i--) {
    alpha = (x-t(i))/(t(i+d)-t(i));
    c(i)  = (1-alpha)*c(i-1) + alpha*c(i);
  }
  t.insert(mu+1,x);
}
// Max_iterations too small for convergence
```

This code will return an approximation to the leftmost root of the spline unless the total number of allowed iterations `max_iterations` is too low (or the tolerance is too small; see below). Note that it is assumed that the spline is connected. In particular, this means that the first coefficient must be nonzero.

The function `converged` returns true when the last inserted knot x equals t_{k+d} (in which case the spline has become disconnected at x; see Lemma 3); or when the sequence of computed zeros of the control polygons are deemed to converge in a traditional sense. Our specific criterion for convergence is (after at least d knots have been inserted),

$$\frac{\max_{i,j} |x_i - x_j|}{\max(|t_k|, |t_{k+d}|)} < \epsilon,$$

where the maximium is taken over the d last inserted knots, and $\epsilon > 0$ is a small user-defined constant. This expression measures the relative difference of the last d knots, and $\epsilon = 10^{-15}$ is a good choice when the computations are performed in double precision arithmetic.

In principle, our method should always converge, so there should be no need for a bound on the number of iterations. However, this is always a good safety net, as long as the maximum number of iterations is chosen as a sufficiently big integer.

There is of course a similar algorithm for computing the largest zero. If one needs to compute all zeros of a spline, this can be done sequentially by first computing the smallest zero, split the spline at that point, compute the second smallest zero and so on. Alternatively, the computations can be done in parallel by inserting all the zeros of the control polygon in each iteration. We leave the details to the interested reader.

A spline with $d + 1$ knots at both ends and without interior knots is usually referred to as a polynomial in Bernstein form or a Bézier polynomial. In this way, the algorithm can obviously be used for computing real zeros of polynomials.

Before considering some examples and comparing our method with other methods, we need to have a rough idea of the complexity of the method. To determine the correct segment of the control polygon requires a search the first time, thereafter choosing the right segment only involves one comparison. Computing the new estimate for the zero is also very quick as it only involves one statement. What takes time is computing the new B-spline coefficients after the new knot has been inserted. This usually requires d convex combinations. As we saw above, we often

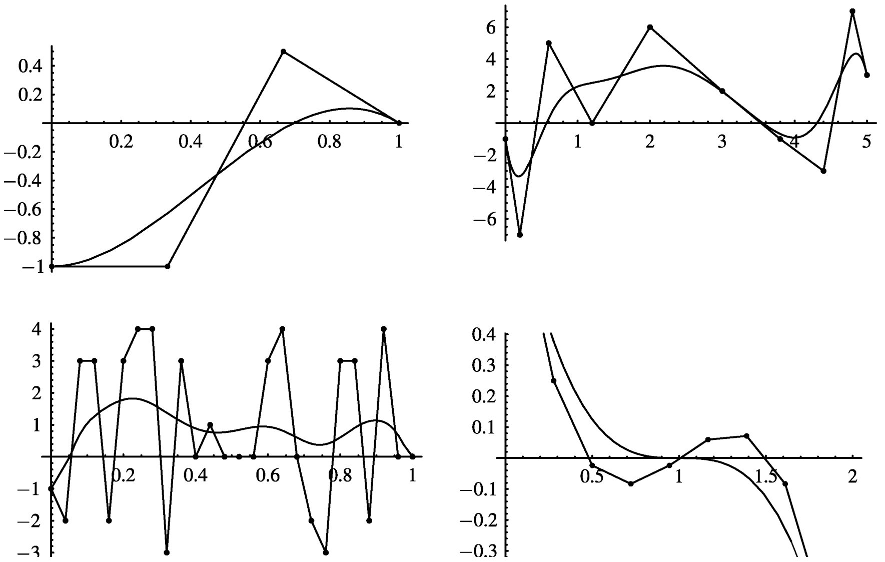

FIGURE 2. Our test examples in reading order: cubic Bézier function, quintic spline, degree 25 spline, cubic polynomial with a double root.

have quadratic convergence if we sample the error every $d-1$ iterations, and the work involved in $d-1$ knot insertions is $d(d-1)$ convex combinations.

We have estimated the errors $e_j = |x_j - z|$ of our algorithm for the four examples shown in Figure 2. The first three have simple roots, while the last has a double root. In Table 1 we have compared the errors produced by our method with those produced by Newton's method and the natural generalisation of the method in [6] (see below) to B-splines. We have compared every $d-1$th step in our method with every iteration of the other methods. Quadratic convergence is confirmed for all the methods for the first three examples, whereas all methods only have linear convergence for the double zero (as for Newton's method, we have observed higher than second order convergence in cases with $f'(z) \neq 0$ and $f''(z) = 0$.)

We used the smallest zero of the control polygon as a starting value for Newton's method. Note that it is not hard to find examples where this starting point will make Newton's method diverge. When using Newton's method to compute zeros of spline functions we have to evaluate $f(x)$ and $f'(x)$ in every step. With careful coding, this requires essentially the same number of operations as inserting d knots at a single point when f is a spline, or roughly $d(d-1)/2$ convex combinations. On the other hand, the method in [6] is based on inserting d knots at the zeros of the control polygon, in effect splitting the curve into two pieces by Bézier subdivision. The complexity of this is the same as for Newton's method. Although no convergence rate was given in that paper, it is reasonable to expect a quadratic convergence rate (as is indicated by the numbers in Table 1). In fact, it should be quite straightforward to prove this with the technique developed in this paper. Bézier clipping [5], which is a method often used for ray-tracing, is very similar to Rockwood's method as the curve is split by inserting d knots where the convex hull of the control polygon of the curve intersects the x-axis.

Example	Method	E_0	E_1	E_2	E_3	E_4	E_5
Cubic Bézier Simple root	Our	1.41e-1	1.31e-2	1.46e-4	1.70e-8	2.30e-16	4.22e-32
	Rockwood	1.41e-1	2.05e-2	8.71e-4	1.80e-6	7.72e-12	1.42e-22
	Newton	1.41e-1	2.05e-2	8.71e-4	1.80e-6	7.72e-12	1.42e-22
Quintic Simple root	Our	1.34e-1	3.06e-3	1.24e-6	1.53e-13	2.21e-27	4.54e-55
	Rockwood	1.34e-1	5.47e-3	2.72e-5	6.87e-10	4.39e-19	1.80e-37
	Newton	1.34e-1	5.06e-3	2.33e-5	5.04e-10	2.36e-19	5.21e-38
Degree 25 spline Simple root	Our	3.79e-3	2.64e-5	7.72e-11	2.60e-22	1.48e-45	5.10e-71
	Rockwood	3.79e-3	3.50e-5	5.49e-9	1.35e-16	8.22e-32	8.22e-32
	Newton	3.79e-3	7.64e-5	2.62e-8	3.08e-15	4.25e-29	8.13e-57
Cubic polynomial Double root	Our	5.19e-1	2.68e-1	1.37e-1	6.95e-2	3.52e-2	1.78e-2
	Rockwood	5.19e-1	3.34e-1	2.23e-1	1.49e-1	9.90e-2	6.60e-2
	Newton	5.19e-1	3.46e-1	2.31e-1	1.54e-1	1.03e-1	6.84e-2

Table 1. The absolute errors $|x_j - z|$ for our method (inserting $d-1$ knots), Rockwoods method and Newtons method for the three examples. The computations have been performed in extended arithmetic in order to include more iterations.

9. Conclusion

We have proposed and analysed a simple zero finding algorithm that exploits the close relationship between a spline and its control polygon. The main advantage is that it overcomes the primary disadvantage of Newton's method in that it is unconditionally convergent at no extra cost and with the same convergence order. Though the theoretical rate of convergence is comparable with other methods, it appears that our method usually converges more quickly in practice.

Quadratic convergence as guaranteed by Theorems 21 and 22 only hold for simple zeros; for multiple zeros we cannot expect more than first order convergence. However, it should be possible to adjust our method to yield second order convergence even for higher order zeros, just like for Newton's method; see [2]. If it is known that a zero z has multiplicity m, another possibility is to apply Algorithm 2 to $f^{(m-1)}$; this yields a quadratically convergent algorithm for computing a zero of multiplicity m. As with the adjustments to Newton's method, this requires that the multiplicity is known. One possibility is to device a method that detects the multiplicity and then apply Algorithm 2 to the appropriate derivative. The ultimate would of course be a method that is quadratically convergent irrespective of the multiplicity. This is a topic for future research.

We believe that methods similar to the one presented here can be applied to other systems of functions that have a control polygon, that can be subdivided and that exhibit a variation diminishing property. Examples of such systems are subdivision curves, Chebychev splines and L-splines; see e.g. [7].

Another topic for future research is to extend our algorithm to a method for computing zero sets of tensor product spline surfaces. This is by no means obvious as the idea of inserting a zero of the control polygon as a knot does not carry over to the tensor product setting.

References

[1] W. Boehm. Inserting new knots into B-spline curves. *Computer Aided Design*, 12:199–201, 1980.

[2] G. Dahlquist and A. Björck. *Numerical Methods.* Wiley-Interscience, 1980.

[3] T. A. Grandine. Computing zeroes of spline functions. *Computer Aided Geometric Design*, 6:129–136, 1989. MR992731 (90a:65115)

[4] T. Lyche and K. Mørken. Spline Methods, draft. Deparment of Informatics, University of Oslo, http://heim.ifi.uio.no/~knutm/komp04.pdf, 2004.

[5] T. Nishita, T. W. Sederberg, and M. Kakimoto. Ray tracing trimmed rational surface patches. In *Proceedings of the 17th annual conference on Computer graphics and interactive techniques*, pages 337–345. ACM Press, 1990.

[6] A. Rockwood, K. Heaton, and T. Davis. Real-time Rendering of Trimmed Surfaces. *Computer Graphics*, 23(3), 1989.

[7] L. L. Schumaker. *Spline functions: Basic theory.* Prentice Hall, 1974.

[8] M. R. Spencer. *Polynomial real root finding in Bernstein form.* Ph.D. thesis, Brigham Young University, 1994.

Department of Informatics and Center of Mathematics for Applications, P.O. Box 1053, Blindern, 0316 Oslo, Norway
E-mail address: `knutm@ifi.uio.no`

Center of Mathematics for Applications, P.O. Box 1053, Blindern, 0316 Oslo, Norway
E-mail address: `martinre@ifi.uio.no`