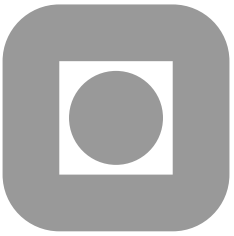
NORGES TEKNISK-NATURVITENSKAPELIGE
UNIVERSITET

High order interpolation of parametric curves and surfaces in $\mathbb{R}^3$

by

Øystein Tråsdahl

PREPRINT
NUMERICS NO. 5/2011



NORWEGIAN UNIVERSITY OF
SCIENCE AND TECHNOLOGY
TRONDHEIM, NORWAY

This report has URL
http://www.math.ntnu.no/preprint/numerics/2011/N5-2011.pdf
Address: Department of Mathematical Sciences, Norwegian University of Science and Technology, N-7491 Trondheim, Norway.

High order interpolation of parametric curves and surfaces in $\mathbb{R}^3$

Øystein Tråsdahl

September 13, 2011

In this paper, high order interpolation of parametric curves and surfaces in $\mathbb{R}^3$ is studied. The topic differs from classical interpolation of functions since any reparametrization of the given curve or surface can be interpolated. This leads to the question whether there exists an optimal reparametrization that results in the lowest possible interpolation error. This can also be viewed as a Kolmogorov n-width problem in terms of polynomial interpolation: how to best use the available degrees-of-freedom in order to minimize the interpolation error. Here, this problem is studied numerically, and different interpolation methods are presented and compared. The methods are introduced in the context of parametric curves and then extended to parametric surfaces when possible. The results are relevant for numerical solution of PDEs using high order methods.

Keywords: Geometric Hermite interpolation, reparametrization, high order polynomials, geometric continuity

1 Introduction

Polynomial interpolation of parametric curves and surfaces is a central part of Computer Aided Geometric Design (CAGD). The traditional way to interpolate a given parametric curve $\boldsymbol{f}$ in $\mathbb{R}^d$ is to view it as a vector-valued *function* and interpolate each of the d components separately. Polynomials of degree N can be made to interpolate the curve in $N+1$ points in this way. If $\boldsymbol{f}$ is in C^k (i.e., each component is a C^k function), one can also choose to interpolate fewer points and instead match both function values and derivatives in the interpolation points. A polynomial of degree N can interpolate a function and its k first derivatives at n points if $N = n(k+1) - 1$. Common for both approaches is that the interpolant can be constructed by solving systems of *linear* equations, and that the approximation order (as defined for approximation of functions) is $N+1$. As an example, consider cubic spline curves in $\mathbb{R}^2$ which can be constructed to interpolate function values and derivatives at the end points of each curve segment, giving the approximation order four.

The parametrization of a curve can be thought of as the position vector for a particle traversing the curve. This implies that the first derivative of $\boldsymbol{f}$ describes its velocity, the second derivative the acceleration and so forth. If our goal is to approximate the curve

Correspondence to `trasdahl@math.ntnu.no`.

as a geometric object, these quantities are of little interest. Instead we are interested in geometric properties of the curve, such as tangent directions, curvature and torsion. It is possible to construct interpolants based on such quantities, and it can yield a higher approximation order than classical interpolation. In [5] it was shown that under certain conditions, cubic polynomial curves in $\mathbb{R}^2$ can interpolate both function values, tangent directions *and* curvature at the end points, resulting in approximation order six. The price to pay for the increased accuracy is a system of non-linear equations that must be solved. The interpolation method was viewed as a generalization of Hermite interpolation based on geometric quantities and was therefore called *geometric Hermite interpolation*.

In recent years we have seen a lot of work on geometric Hermite interpolation in the CAGD community; e.g., see [6, 7, 8, 10, 12, 16, 21, 23, 27]. The work has led to a conjecture [14] about the highest possible approximation order that can be attained when interpolating parametric curves in $\mathbb{R}^d$ by polynomials of degree N. The conjecture has been confirmed in some special cases, but it remains unproven. Most of the authors focus on planar curves, but there has also been some work done on curves in $\mathbb{R}^3$: cubic interpolation was studied in [11, 14], quartic in [4, 26], and quintic in [24].

The concept of geometric Hermite interpolation can also be applied in the context of parametric surfaces, but the problem is much harder due to the increased number of unknowns. Mørken [18] gives a detailed discussion of the optimal approximation order and constructs a quadratic Taylor approximant with approximation order four. Lagrange interpolation of surfaces with quadratic polynomials is considered in [15].

There has been surprisingly little work done on *high order* interpolation of parametric curves and surfaces. In the field of CAGD polynomials of degree $N > 5$ are not so common in applications. However, the subject is relevant in the context of solving PDEs in deformed geometries using high order methods [3, 9]. Here, the accuracy of the numerical solution is directly influenced by the accuracy of the geometry representation [17]. It is common to use an *isoparametric approach*, representing the geometry with the same polynomial degree as the other field variables. For example, in a Legendre spectral element method, deformed quadrilaterals or hexahedrons are approximated by tensor-product polynomials, constructed by interpolating the exact geometry. In this context a reparametrization may yield a better representation of not only the geometry, but also the primary field variables [25]. Still, the topic has not been given much attention in the literature.

In the context of high order interpolation the concept of approximation order is not commonly used since the approximation approach is global and we only use one polynomial curve segment for the entire curve (as opposed to for example a spline approach). Convergence is rather defined in terms of how the interpolation error, measured in some appropriate norm, decreases as the polynomial degree N increases. It is well known from classical interpolation theory that smooth functions can be interpolated by polynomials to exponential convergence, i.e., the interpolation error decreases faster than any algebraic power of N [2]. An optimal interpolation method may thus be defined as one that yields exponential convergence with the highest possible rate, i.e., with the largest possible negative exponent. Functions of finite regularity yield algebraic convergence in classical interpolation, but as we will see, the choice of interpolation points *implicitly defines* a reparametrization, which is the function that is actually being interpolated. A good interpolation method may give us exponential convergence, even if the given parametrization is a function of low regularity.

The outline of the paper is a follows. In Section 2 we first present the framework for polynomial interpolation of parametric curves and discuss how the option of reparametrization makes the subject different from classical interpolation. We then present two interpolation

methods that are commonly used in the high order methods community, and we introduce three new methods: one optimization method aimed at directly minimizing the interpolation error, and two methods in the family of geometric Hermite interpolation. In Section 3 we compare the different methods through several numerical examples. In Section 4 we discuss how to extend the methods to interpolation of parametric surfaces, and some numerical examples are presented in Section 5. The conclusions of this study are summarized in Section 6.

2 Interpolation of parametric curves

Consider a curve $\mathscr{C}$ in $\mathbb{R}^3$, defined by a given parametrization

$$\boldsymbol{f}(\eta) = (f_1(\eta), f_2(\eta), f_3(\eta)), \qquad \eta \in [-1, 1]. \tag{1}$$

The curve is C^k-continuous if each of the parametric functions f_i, $i = 1, 2, 3$, are in C^k. The problem we set out to solve is how to best interpolate this curve by polynomials, i.e., a parametric curve

$$\boldsymbol{p}(\xi) = (p_1(\xi), p_2(\xi), p_3(\xi)), \qquad \xi \in [-1, 1], \tag{2}$$

where p_i, $i = 1, 2, 3$ are functions in $\mathbb{P}_N([-1, 1])$, the (discrete) space of polynomials of degree less than or equal to N. This problem is different from classical polynomial approximation of functions since $\mathscr{C}$ can be *reparametrized*. Specifically, for all $\varphi \in W = \{\psi \in C^\infty([-1, 1]) \mid \psi(\pm 1) = \pm 1, \text{ and } \psi' > 0\}$ the function

$$\boldsymbol{g}(\xi) = \boldsymbol{f}(\varphi(\xi)) \tag{3}$$

describes the same curve, so interpolation of $\boldsymbol{g}$ instead of $\boldsymbol{f}$ gives an approximation of the same geometric object. Intuitively, a reparametrization means traversing the curve at a different speed. There exist infinitely many reparametrizations of any given curve, and some may be better suited for polynomial interpolation than others. Hence, finding the best interpolant involves finding the best parametrization, a problem which is very difficult.

From classical interpolation theory we know that for a well chosen set of interpolation points (e.g., the Gauss points), a scalar function $u \in H^\sigma([-1, 1])$ can be interpolated by polynomials $I_N u$ with an interpolation error [2]

$$||u - I_N u||_{L^2} \leq cN^{-\sigma}||u||_{H^\sigma(\Omega)}, \tag{4}$$

where c is a constant. If u is analytic, the error will decrease faster than any algebraic power of N, and we obtain exponential convergence. This also translates to vector-valued functions.

Let us illustrate the importance of reparametrization with an example. Consider a curve defined by the parametrization

$$\begin{aligned} f_1(\eta) &= \eta + 1, \\ f_2(\eta) &= \sqrt{(\eta + c)^{1/3} - 1}, \\ f_3(\eta) &= (\eta + c)^{2/3} - 1, \end{aligned} \tag{5}$$

where $\eta \in [-1, 1]$ and c is a constant. For $c > 2$ the parametric functions are smooth, and $\boldsymbol{f}$ can be interpolated by polynomials to exponential convergence. However, applying the particular change of variable

$$\eta = \varphi(\xi) = \left((a\xi + b)^2 + 1\right)^3 - c$$

to $\boldsymbol{f}$, we get the reparametrization

$$\begin{aligned} g_1(\xi) &= ((a\xi + b)^2 + 1)^3 - c + 1, \\ g_2(\xi) &= a\xi + b, \\ g_3(\xi) &= ((a\xi + b)^2 + 1)^2 - 1, \end{aligned} \tag{6}$$

where all the parametric functions are polynomials of degree less than or equal to six. Interpolating this parametrization will result in exact representation of the curve for $N \geq 6$, which is obviously a great improvement.

When using Legendre spectral element methods in deformed hexahedra, the edges are approximated by parametric curves and the faces are approximated by parametric surfaces [9]. The end points of a curve are interpolation points and the interpolation points are mapped from the Gauss-Lobatto-Legendre (GLL) points ξ_j, $j = 0, \ldots, N$ by the interpolant. We represent such an interpolant using Lagrange interpolation polynomials through the GLL points,

$$p_i(\xi) = \sum_{j=0}^{N} \alpha_j^i \ell_j(\xi), \qquad i = 1, 2, 3, \tag{7}$$

where the coefficients α_j^i are determined by the interpolation conditions. In classical interpolation, this means simply evaluating the given parametrization $\boldsymbol{f}$ in the GLL points, i.e., $\alpha_j^i = f_i(\xi_j)$. Reparametrizing the curve before interpolating yields $\alpha_j^i = g_i(\xi_j)$, in which case the interpolation points are no longer mapped from the GLL points by $\boldsymbol{f}$, but rather from the points

$$\eta_j = \varphi(\xi_j), \qquad j = 0, \ldots, N. \tag{8}$$

Note that due to the nodal representation (7), the interpolant always maps the GLL points to the interpolation points.

In the current context, the mapping $\varphi(\xi)$ is unknown, since we do not know *a priori* which reparametrization is best suited for polynomial interpolation. The η_j, $j = 0, \ldots, N$, in (8) can thus be viewed as *free variables* which can be manipulated subject to certain restrictions, imposed by W. Specifically, we must require all $\eta_j \in [-1, 1]$, and that they appear in consecutive order, i.e.,

$$-1 \leq \eta_0 < \eta_1 < \ldots < \eta_N \leq 1. \tag{9}$$

Interpolation of the end points implies setting $\eta_0 = -1$ and $\eta_N = 1$, and we are left with $N - 1$ *degrees-of-freedom* which can be used to improve the approximation properties of the interpolant.

The interpolant is uniquely defined by the choice of η_j through the definition

$$\alpha_j^i = f_i(\eta_j) \tag{10}$$

of the expansion coefficients in (7). The change of variable φ, on the other hand, is only partially determined by (8). To turn the statement around, one can say that there are (infinitely) many reparametrizations that, when interpolated in the classical sense, yield the same interpolant. It will be convenient to choose φ to be the polynomial of lowest degree that satisfies (8). This can be done if the polynomial interpolating $\eta_0, \ldots, \eta_N$ is monotonic. It is then a (uniquely determined) function in $W \cap \mathbb{P}_N([-1, 1])$, and we will refer to it as φ_N.

2.1 Measuring the interpolation error

It is not trivial to define a norm for the interpolation error in the context of parametric curves. The norm should measure the distance between the geometric objects represented by $\boldsymbol{f}$ and $\boldsymbol{p}$, regardless of the particular parametrizations chosen. One metric satisfying this requirement is the *Hausdorff metric* [20]. Unfortunately, this norm is not very well suited for numerical calculations. Other possibilities are the *normal distance* proposed by Degen [6], or the metric proposed by Mørken and Scherer [19].

We will use none of these metrics, but rather the L^2-like norm

$$||\boldsymbol{f} - \boldsymbol{p}|| = \Big(\int_a^b \sum_{i=2}^{3} \big(f_i(f_1^{-1}(x)) - p_i(p_1^{-1}(x)) \big)^2 \, \mathrm{d}x \Big)^{1/2}, \tag{11}$$

where $a = f_1(-1)$ and $b = f_1(1)$. It will be implemented using GLL quadrature with over-integration to ensure that the quadrature error is sub-dominant. The reason for choosing this norm is that it makes the interpolation error an explicit function of the free variables η_j, $i = 1, \ldots, N-1$. This will enable us to define an interpolation method based on direct minimization of the interpolation error.

It might not be immediately clear why a norm like

$$||\boldsymbol{f} - \boldsymbol{p}|| = \Big(\int_{-1}^{1} \sum_{i=1}^{3} \big(f_i(\varphi_N(\xi)) - p_i(\xi) \big)^2 \, \mathrm{d}\xi \Big)^{1/2} \tag{12}$$

is not acceptable. The problem is that basing the norm on an integral over the parametric domain makes the norm parametrization-dependent. Even though $\boldsymbol{f}(\varphi_N(\xi_j)) = \boldsymbol{p}(\xi_j), j = 0, \ldots, N$ (i.e., the reparametrization and the interpolant reach the interpolation points "at the same time"), it is not given that $\boldsymbol{p}(\xi)$ is the best approximation of $\boldsymbol{f}(\varphi_N(\xi))$ for any other given ξ. After all, φ_N was *chosen* among all the changes of variable that yield the interpolant $\boldsymbol{p}$.

It should be noted that this is mainly a theoretical problem. If $\boldsymbol{f}$ is smooth, then so is $\boldsymbol{f} \circ \varphi_N$, and hence $\boldsymbol{p}$ will approximate it to exponential convergence. This implies that the curves are being traversed with approximately the same velocity. Moreover, if $f_1(\eta) = \eta$, then (11) and (12) coincide.

The definition of the norm (11) puts a restriction on the curves that can be studied, since it is only defined when the first parametric function $f_1(\eta)$ is monotonic, i.e., when the curve can be uniquely determined by specifying its x-coordinate. This restriction on the curves is not a limitation on the interpolation methods studied here. However, the numerical results will only include curves from this subset in order to be able to quantitatively compare the different methods.

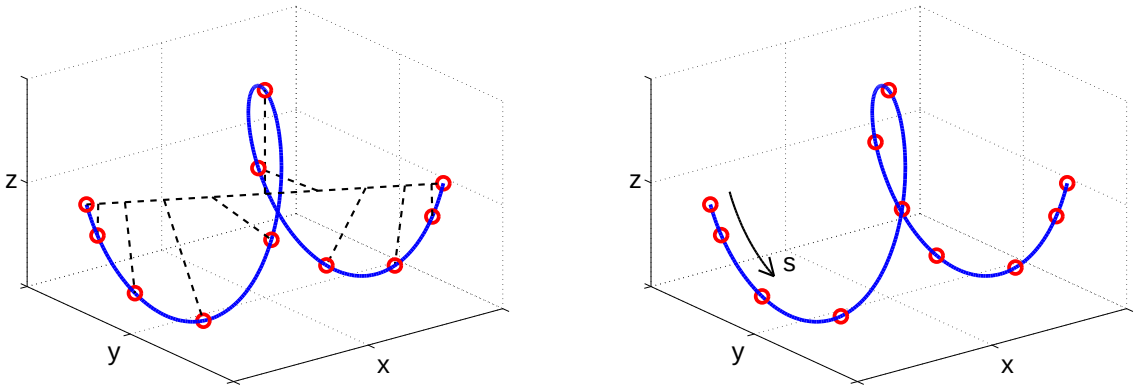
2.2 Interpolation methods

As already mentioned, the simplest way to interpolate a parametric curve $\boldsymbol{f}$ is to view it as a vector-valued function and let $\alpha_j^i = f_i(\xi_j)$, $j = 0, \ldots, N$. This may be satisfactory if we know f_i, $i = 1, 2, 3$ to be smooth functions, but in other cases it may be far from optimal.

In the high order methods community there are two common interpolation methods that are independent of the parametrization [9]. Both rely in some way on an affine mapping of the GLL points from the parametric variable to the physical domain.

The first interpolation method considered here, referred to as the *chord method*, is defined by first mapping the GLL points affinely to the chord between the two end points of the curve, and then letting the interpolation points be the intersection between the exact curve

and the normal planes to the chord at these affinely mapped points (Figure 1a). Finding these intersection points requires an iterative procedure like Newton's method. The chord method obviously does not work for closed curves, and it also fails in cases where the the curve intersects a normal plane to the chord in more than one point. However, this is not a significant limitation in the context of high order methods for solving PDEs. The chord method will yield (rapid) exponential convergence if the curve can be described by a smooth function in a rotated coordinate system where the abscissa is parallel to the chord.



(a) Chord method

(b) Arc-length method

Figure 1: Two common methods for choosing interpolation points. Both methods involve an affine mapping of the GLL points: the chord method along the chord between the end points, the arc-length method in the arc-length variable s.

The second method is based on a GLL distribution in the arc-length variable s, and is called the *arc-length method*. On a curve of length L, construct the affine mapping $s(\xi) = \frac{L}{2}(\xi + 1)$, $\xi \in [-1, 1]$, and define the values $s_j = s(\xi_j)$, $j = 0, \ldots, N$, associated with the GLL points ξ_j. Each value s_j corresponds to a unique point along the curve with coordinates (x_j, y_j, z_j), which is then defined as an interpolation point (Figure 1b). Again, an iterative procedure such as Newton's method is required.

The arc-length method is equivalent to interpolating a reparametrization $\boldsymbol{g}$ with constant Jacobian

$$J(\xi) = \left(\sum_{i=1}^{3} g_i'(\xi)^2 \right)^{1/2}.$$

This, of course, does not guarantee that the parametric functions are smooth.

In the example with the curve parametrized by (5) and (6), none of these two interpolation methods correspond to classical interpolation of $\boldsymbol{f}$ or $\boldsymbol{g}$.

2.3 The L^2-method

A good interpolant should yield a small interpolation error, measured in the norm (11). The norm was chosen because it enables us to explicitly evaluate the measured interpolation error as a function of the free variables $\eta_1, \ldots, \eta_{N-1}$ (when approximated with GLL quadrature). This makes it possible to define an interpolation method based on direct minimization of the measured interpolation error, using a (global) optimization algorithm. We will use the objective function

$$\Lambda(\eta_1, \ldots, \eta_{N-1}) = ||\boldsymbol{f} - \boldsymbol{p}||^2, \tag{13}$$

and the method will be referred to as the L^2-*method*.

In order to avoid the evaluation of the inverses of f_1 and p_1, we only consider parametrizations $\boldsymbol{f}$ where the first component $f_1(\eta)$ is linear. All curves that can be measured by the norm (11) can be reparametrized this way. We then have

$$f_1(\varphi_N(\xi)) = p_1(\xi), \tag{14}$$

which allows a change of variable $x = f_1(\varphi_N(\xi))$, and the norm can be rewritten as

$$||\boldsymbol{f} - \boldsymbol{p}|| = \Big(\int_{-1}^{1} \sum_{i=2}^{3} \left(f_i(\varphi_N(\xi)) - p_i(\xi)\right)^2 p_1'(\xi)\,\mathrm{d}\xi \Big)^{1/2}, \tag{15}$$

Here, all terms can be evaluated explicitly; GLL quadrature will be used to evaluate the integrals.

The minimization is implemented using Newton's method, which requires the first and second order partial derivatives of Λ w.r.t. $\eta_1, \ldots, \eta_{N-1}$. These can be found explicitly from the formulation (15) when the change of variable φ_N is viewed as a function of both ξ and $\eta_1, \ldots, \eta_{N-1}$. It is represented using a standard linear combination of N-th order Lagrangian interpolants

$$\varphi_N(\xi; \eta_1, \ldots, \eta_{N-1}) = \sum_{j=0}^{N} \eta_j \ell_j(\xi). \tag{16}$$

Similarly, the components of the polynomial interpolant are defined as

$$p_i(\xi; \eta_1, \ldots, \eta_{N-1}) = \sum_{j=0}^{N} f_i(\eta_j) \ell_j(\xi). \tag{17}$$

The first partial derivatives of (13) with respect to the η_j are then given by

$$\begin{aligned}\frac{\partial \Lambda}{\partial \eta_j} = \int_{-1}^{1} \sum_{i=2}^{3} 2 \Big(f_i(\varphi_N(\xi)) - p_i(\xi)\Big) \Big(f_i'(\varphi_N(\xi)) \frac{\partial \varphi_N}{\partial \eta_j}(\xi) - \frac{\partial p_i}{\partial \eta_j}(\xi)\Big)\, p_1'(\xi) \\ + \Big(f_i(\varphi_N(\xi)) - p_i(\xi)\Big)^2 \frac{\partial p_1'}{\partial \eta_j}(\xi)\,\mathrm{d}\xi,\end{aligned}$$

where the simplified notation $p_1'(\xi)$ is used to remind the reader that p_1 is originally a function of ξ, even though it also depends on the *parameters* $\eta_1, \ldots, \eta_{N-1}$. The second order partial derivatives $\frac{\partial^2 \Lambda}{\partial \eta_j \partial \eta_k}$ are easily derived by repeated partial differentiation, and we do not write them out here. They include terms with the first and second derivatives of f_2 and f_3, so these need to be known explicitly. The remaining functions can be differentiated numerically without error by means of differentiation matrices, since all of the functions are polynomials.

The L^2-method should, by construction, give the best interpolant in terms of the measured interpolation error. However, it is based on a very hard global minimization problem. The objective function Λ is almost never globally convex, and its complexity increases as the polynomial degree N increases. This is connected to the global interpolation approach: moving just one interpolation point (locally) changes the entire interpolant (globally). Newton's method is a local minimization algorithm and can not be expected to find the global minimum. Some measures will be taken to make the method more robust (see Section 2.7), but the increasing complexity will be reflected in the numerical results; see Section 3.

2.4 Geometric interpolation

From classical interpolation theory we know that interpolation of a smooth function in $N+1$ points yields approximation order $N+1$. This is an incentive for using the available degrees-of-freedom in curve interpolation to *increase* the number of interpolation points.

By construction, the points $\boldsymbol{p}(\xi_j)$, $j = 0, \ldots, N$ are always interpolation points. To achieve interpolation in one *additional* point, we need $\boldsymbol{p}$ to satisfy the interpolation condition

$$\boldsymbol{f}(\varphi_N(\xi^*)) = \boldsymbol{p}(\xi^*) \tag{18}$$

for a ξ^* that is *not* a GLL point. Equation (18) represents a system of three (non-linear) equations, and ξ^* is a free variable. This means that we need two more degrees-of-freedom to find a solution in the general case. Since we have $N-1$ free variables $\eta_1, \ldots, \eta_{N-1}$, it is in principle possible to achieve a total of $N + 1 + \lfloor (N-1)/2 \rfloor$ interpolation points.

This argumentation can also be applied to interpolation of curves in $\mathbb{R}^d$. In this case we still have $N-1$ free variables, but now an additional interpolation point requires $d-1$ degrees-of-freedom. Assuming that the system of non-linear equations always has a solution leads to the following conjecture [19]:

Conjecture 2.1. *Let $\mathscr{C}$ be a curve in $\mathbb{R}^d$. A polynomial curve of degree N can be made to interpolate $\mathscr{C}$ at*

$$m = N + 1 + \left\lfloor \frac{N-1}{d-1} \right\rfloor \tag{19}$$

points.

The conjecture also applies to Hermite interpolation if one defines interpolation of k coalescing interpolation points as interpolation of a (yet unknown) reparametrization $\boldsymbol{g}$ and its $k-1$ first derivatives. For the first derivative, this means that we do not have to require $\boldsymbol{f}'(\varphi(\xi^*)) = \boldsymbol{p}'(\xi^*)$, only that they point in the same direction. This kind of requirement can be expressed in terms of *geometric continuity*. A curve $\mathscr{C}$ is said to be G^k-*continuous* if its arc-length parametrization is C^k-continuous [1]. An equivalent definition can be found in [8]. In terms of interpolation we say that two curves have *contact order* k if the left segment of the interpolant meets the right segment of the exact curve with G^k-continuity, and vice-versa. First order contact means a common tangent direction, while second order contact additionally requires common curvature *and* coinciding osculating planes.

With this definition, coalescing interpolation points means increased contact order. Interpolation in the conjectured maximum number of interpolation points, but with some points coalescing, is exactly the same as *geometric Hermite interpolation* that was described in the introduction. Consider for example cubic polynomial curves in $\mathbb{R}^2$, which can interpolate a given curve in six points, according to the conjecture. If three interpolation points coalesce at each end point, the contact order is raised to two, and we have the same interpolation conditions as in [5].

In the extreme case where all interpolation points coalesce to *one* point we get a Taylor approximation of $\boldsymbol{f}$. For a planar parametric curve $\boldsymbol{f}(\eta) = (\eta, y(\eta))$ one can easily show [21] that a one-point G^k-interpolant coincides with the $k+1$ first terms of the Taylor expansion of y. Hence, one-point G^k-interpolation in $\mathbb{R}^2$ yields *approximation order* $k+1$. The argument can be extended to general curves in $\mathbb{R}^d$; see [19] for definitions of norms and approximation order. Then, according to the conjecture, (19) is the highest attainable approximation order for curve interpolation in $\mathbb{R}^d$ at a given N. Some authors [14, 21]

have actually formulated the conjecture in terms of approximation order, stating that the approximation order (19) can be attained for any curve $\mathscr{C}$ in $\mathbb{R}^d$.

The conjecture has turned out to be very difficult to prove. Since the subject has been studied mostly within the CAGD community, most authors are concerned with low polynomial degrees ($N \leq 5$). Of more general results, we mention Rababah [21, 22] who showed that curves in $\mathbb{R}^d$ can be interpolated by one-point interpolation to approximation order $4N/3$ for arbitrary N, and Floater [12] who showed optimal approximation order $2N$ for conic sections.

2.5 The extra-points method

We propose an interpolation method based on Conjecture 2.1, which we will refer to as the *extra-points method*.

Assuming that f_1 is invertible, the interpolation condition (18) can be reduced to a system of two equations by *choosing* a specific ξ^* and *defining* $x^* = f_1(\varphi_N(\xi^*)) = p_1(\xi^*)$. Interpolation at x^* then requires the two equations

$$f_i(f_1^{-1}(x^*)) = p_i(p_1^{-1}(x^*)), \qquad i = 2, 3$$

to be solved. Furthermore, when f_1 is linear then (14) holds, and the inverses can be eliminated. The extra-points method can then be defined as finding a root of the vector-valued function $\mathbf{\Psi}$ with components

$$\Psi_k^i(\eta_1, \ldots, \eta_{N-1}) = f_i(\varphi_N(\xi_k^*)) - p_i(\xi_k^*), \qquad i = 2, 3, \tag{20}$$

where ξ_k^*, $k = 1, \ldots, \lfloor (N-1)/2 \rfloor$ are pre-defined values in $[-1, 1]$. The dependency of Ψ_k^i on η_j, $j = 1, \ldots, N-1$ is explicit in the representations (16) and (17) of φ_N and p_i, respectively. Note that for odd N there is one more degree-of-freedom than the number of equations to be solved, so we are left with one "unused" degree-of-freedom.

Applying Newton's method to solve (20) requires the partial derivatives of Ψ_k^i, which are given by

$$\frac{\partial \Psi_k^i}{\partial \eta_j} = f_i'(\varphi_N(\xi_k^*)) \frac{\partial \varphi_N}{\partial \eta_j}(\xi_k^*) - \frac{\partial p_i}{\partial \eta_j}(\xi_k^*).$$

The derivatives of the parametric functions $f_i(\eta)$ must be known; the rest are computed numerically with differentiation matrices.

The solution obtained depends on the choice of the values ξ_k^*. Numerical experiments show that the best results are usually obtained when the ξ_k^* are close to a subset of the GLL points. In the limit when the points coalesce, the proposed method becomes useless because (20) is always satisfied. According to the previous discussion, one should instead raise the contact order at the coalescing points. This leads to the next proposed method.

2.6 The equal-tangents method

Two coalescing interpolation points should yield first order contact (common tangent directions) between the exact curve and the interpolant. Common tangent directions implies that a tangent vector $\boldsymbol{t}_N$ to the interpolant is orthogonal to all vectors in the normal plane of the exact curve, i.e.,

$$\boldsymbol{t}_N \cdot \boldsymbol{n} = 0 \qquad \forall \boldsymbol{n} \text{ such that } \boldsymbol{t} \cdot \boldsymbol{n} = 0,$$

where $\boldsymbol{t}$ is a tangent vector to the exact curve. The space of normal vectors to a curve in $\mathbb{R}^3$ is two-dimensional, so we must use two linearly independent normal vectors and

make both dot products zero. The tangent vector is easily found by differentiating the parametrization,

$$\boldsymbol{t}_N = \begin{bmatrix} p_1'(\eta) \\ p_2'(\eta) \\ p_3'(\eta) \end{bmatrix}.$$

It does not have to be normalized for our application, which is an advantage, since it would have resulted in more complicated non-linear equations. We then *choose* the two normal vectors

$$\boldsymbol{n}_1 = \begin{bmatrix} f_2'(\eta) \\ -f_1'(\eta) \\ 0 \end{bmatrix} \quad \text{and} \quad \boldsymbol{n}_2 = \begin{bmatrix} f_3'(\eta) \\ 0 \\ -f_1'(\eta) \end{bmatrix}.$$

Again, this is a choice to ease the implementation. It clearly would not work for curves where $f_1'(\eta)$ can be zero, since the two vectors then become linearly dependent, but we have already excluded such curves from the current study for the definition of the norm (11).

The number of degrees-of-freedom allows G^1-interpolation in $\lfloor (N-1)/2 \rfloor$ (unique) points. By construction we have interpolation in $N+1$ points, so a subset of the interpolation points must be chosen; we choose the set of internal points ξ_k with odd indices k. The interpolation conditions can then be expressed as a system of equations

$$\Theta_k^i(\eta_1, \ldots, \eta_{N-1}) = f_1'(\varphi_N(\xi_k))\, p_i'(\xi_k) - f_i'(\varphi_N(\xi_k))\, p_1'(\xi_k) = 0, \qquad i = 2, 3. \tag{21}$$

Again Newton's method is used for finding a solution of the non-linear system. This requires us to differentiate (21) with respect to the independent variables $\eta_1, \ldots, \eta_{N-1}$. The partial derivatives can be written out explicitly as

$$\begin{aligned} \frac{\partial \Theta_k^i}{\partial \eta_j} = {} & f_1''(\varphi_N(\xi_k)) \frac{\partial \varphi_N}{\partial \eta_j}(\xi_k)\, p_i'(\xi_k) + f_1'(\varphi_N(\xi_k)) \frac{\partial p_i'}{\partial \eta_j}(\xi_k) \\ & - f_i''(\varphi_N(\xi_k)) \frac{\partial \varphi_N}{\partial \eta_j}(\xi_k)\, p_1'(\xi_k) - f_i'(\varphi_N(\xi_k)) \frac{\partial p_1'}{\partial \eta_j}(\xi_k). \end{aligned}$$

Note again that for odd N we have one more degree-of-freedom than the number of equations.

2.7 Implementation

The three interpolation methods proposed here (the L^2-method, the extra-points method and the equal-tangents method) are based on quite simple criteria, but their implementations are challenging. We mention a few aspects here that are important in order for the methods to work well in practice.

As mentioned previously, the extra-points method depends on the choice of ξ_k^*; letting these parameters be close to some of the GLL points is often a good choice. In all the numerical experiments here $\xi_k^* = \xi_{2k-1} + \varepsilon$ for $k = 1, \ldots, \lfloor (N-1)/2 \rfloor$ and $\varepsilon = 10^{-2}$.

All the proposed methods are highly dependent on a good set of initial values η_j for the Newton iterations. For the L^2-method, this is connected to the fact that we are trying to solve a *global* minimization problem with a *local* minimization algorithm. For the extra-points and equal-tangents methods, it is due to the fact that non-linear functions may have several roots. The conjecture says nothing about the uniqueness of the solution, and numerical experiments have confirmed the existence of several solutions in many cases.

We want the solution with the smallest interpolation error, which we will refer to as the optimal solution for the given interpolation method.

We have made the observation that when we are able to find the optimal solution (or something close to optimal), the reparametrization seems to converge to a particular function as the polynomial degree increases. In other words, for a given (high) N, the functions $p_i(\xi)$, $i = 1, 2, 3$ are very similar to the corresponding functions at $N - 1$. This leads us toward the idea of a bootstrapping algorithm, in which we use the solution from $N - 1$ as the initial guess by evaluating $\varphi_{N-1}(\xi)$ in the current $N + 1$ GLL points. Starting all the way from a polynomial degree of one, such a bootstrapping approach implies an added computational cost. However, the improved robustness has been more important in the current study.

The bootstrapping approach often yields good initial guesses, but not always. Newton's method may not succeed, or it may find a non-optimal solution (we can usually recognize non-optimal solutions from a sudden change in the convergence rate as N increases). In such cases it may help to use other initial guesses, e.g., perturbations of the bootstrapping solution, or the solutions found with the chord or arc-length methods. Here, we use such additional initial values to increase the robustness of the methods. When different initial values yield different solutions after the Newton iterations, the interpolation error is compared and the solution with the smallest interpolation error is chosen.

One can also add to the robustness by making sure that the solution never violates the restriction (9) imposed by W during the Newton iterations. Experience shows that the extra-points and equal-tangents methods sometimes find "illegal solutions". To avoid this, we limit the step sizes in the Newton iterations, and we also explicitly check the condition (9).

3 Numerical Results

We now present a series of numerical tests to illuminate the challenges of reparametrization and to illustrate the performance of the various methods in different situations. All the parametrizations are defined on the interval $[-1, 1]$.

Case 1

Consider one and a half rotations of a helix, which is most naturally parametrized by

$$\begin{aligned} f_1(\eta) &= \frac{3}{2}\pi\eta, \\ f_2(\eta) &= \sin(\frac{3}{2}\pi\eta), \\ f_3(\eta) &= \cos(\frac{3}{2}\pi\eta). \end{aligned}$$

This particular parametrization yields a constant Jacobian $J = 3\pi/\sqrt{2}$, so the arc-length method corresponds to interpolating $\boldsymbol{f}$ in the GLL points. Due to the regularity of the given parametric functions, it gives rapid exponential convergence; see Figure 2. The chord method also yields exponential convergence, but much slower than the arc-length method. To reach the same level of interpolation error, approximately three times the polynomial degree N is needed with the chord method. In the context of solving PDEs using high order methods, this has a huge impact on the computation time.

The extra-points and equal-tangents methods give almost exactly the same interpolation error; in fact, the solutions are almost exactly the same. This is due to our choice of ξ_k^*

in the extra-points method. Both methods converge quite a bit faster than the arc-length method, reaching machine precision at $N = 15$.

By construction, the L^2-method should give the optimal solution. However, experience shows that the L^2-method usually finds the optimal solution (or something very close) for small N when the objective function is easier to minimize globally. If an interpolation method is able follow and maintain this convergence rate for higher N, it is a strong indication that it is able to yield a solution which is close to optimal. This is what we observe in this case, strongly suggesting that the extra-points and equal-tangents methods yield close to the optimal solution.

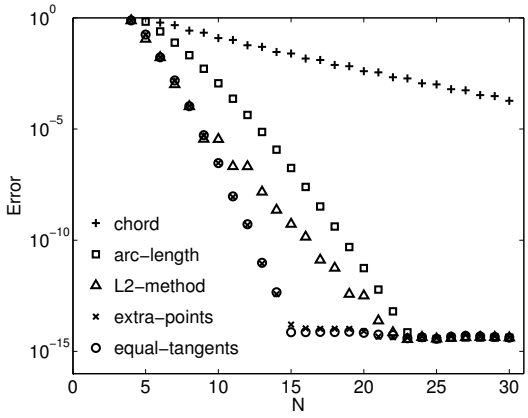


Figure 2: Interpolation error for Case 1, the helix, measured in a discrete version of the norm (11). Note the large difference in convergence rate between the chord method and the arc-length method. The latter is usually considered optimal for this particular case, but the new methods show that it is possible to do better.

Case 2

The curve defined by the parametrization (5) is designed in such a way that we know that there exists a reparametrization (6) by low order polynomials. However, none of the interpolation methods get this reparametrization as an initial value for their iterative procedures.

The chord method and the arc-length method both converge exponentially; see Figure 3. Note that this could not be foreseen from the given parametrizations, as these methods do not correspond to interpolating any of them. We also see that the new methods give considerably better results, although none of them give exact representation of the curve at $N = 6$. The extra-points and equal-tangents methods are very close, with error on machine level precision from $N = 7$. The L^2-method also yields good results, reaching machine level precision at $N = 14$. It does, however, display the weakness that is characteristic for this method: the functional to be minimized becomes increasingly complicated as N increases, with many local minima, and our simple minimization algorithm has difficulty finding a global minimum. The result is a convergence rate that decreases as N increases.

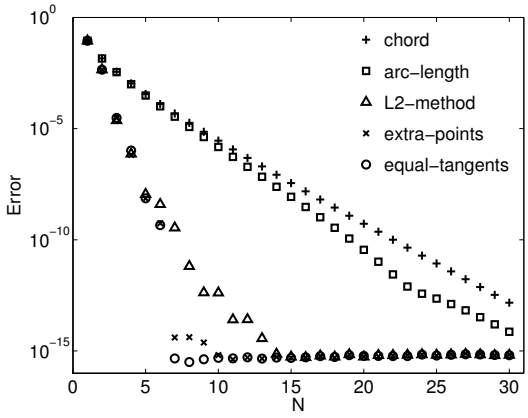


Figure 3: Interpolation error for Case 2, measured in the same norm as before. The reparametrization (6) shows that exact representation of the curve is possible at $N = 6$. None of the methods achieve this, but the extra-points and equal-tangents methods yield rapid convergence, representing the curve to machine precision at $N = 7$. The chord method and the arc-length method both converge exponentially, but much slower than the new methods.

Case 3

The parametric curve

$$\begin{aligned} f_1(\eta) &= -\frac{5}{2} + \frac{7}{4}(\eta + 1), \\ f_2(\eta) &= \frac{1}{2}\sin\left(\frac{3\pi\eta}{2}\right) + \frac{9}{8}(\eta + 1)^2 - \frac{9}{4}(\eta + 1), \\ f_3(\eta) &= \frac{1}{2}\cos(\pi\eta), \end{aligned} \tag{22}$$

has only analytic components, so a classical approach, simply interpolating the given functions in the GLL points, will give (rapid) exponential convergence. However, the chord method and the arc-length method both yield very slow (although still exponential) convergence; see Figure 4. Hence, neither of them correspond to classical interpolation.

The new methods all give vastly better performance (and better than classical interpolation of (22)). They converge at approximately the same rate and reach machine precision between $N = 15$ and $N = 20$. Compared to the two traditional methods, only a small fraction of the polynomial degree is needed to reach the same level of accuracy.

Case 4

The parametric curve

$$\begin{aligned} f_1(\eta) &= \eta + 1, \\ f_2(\eta) &= \sin(\pi(\eta + 1)), \\ f_3(\eta) &= \sin(2\pi\eta), \end{aligned}$$

bears some resemblance to the parametrization of the helix, except that the trigonometric functions are phase-shifted and have different periods. Interestingly, comparing the Figures 2 and 5, we see that the relative performance of the chord method and the arc-length method are opposite. In the current case, the chord method is vastly better than the arc-length method, which converges extremely slowly and is rather useless.

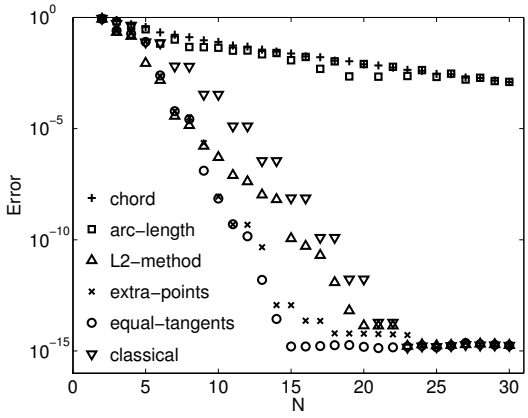


Figure 4: Interpolation error for Case 3. The given parametrization (22) consists of smooth functions, so classical interpolation results in rapid exponential convergence. Neither the chord method nor the arc-length method corresponds to classical interpolation of (22), and both methods yield very low convergence rate.

Again, the new methods outperform the traditional ones. The extra-points and equal-tangents methods give very similar results and converge very fast, and the L^2-method is almost as good.

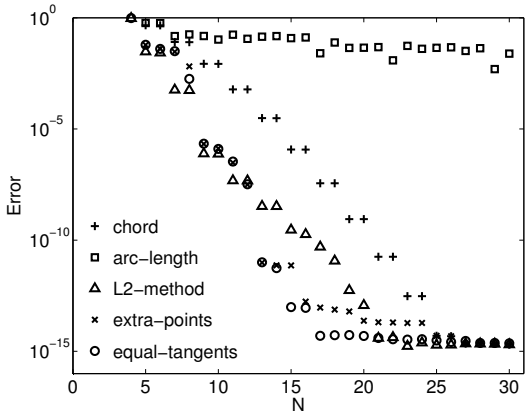


Figure 5: Interpolation error for Case 4. The results are similar to Case 3, except that the chord method is now efficient, while the arc-length method is useless.

Case 5

Consider the parametric curve

$$\begin{aligned} f_1(\eta) &= \eta, \\ f_2(\eta) &= |\eta - \frac{1}{2}|, \\ f_3(\eta) &= |\eta + \frac{1}{2}|. \end{aligned}$$

Again the Jacobian $J = \sqrt{3}$ is constant, so classical interpolation of $\boldsymbol{f}$ corresponds to the arc length method. The curve is G^0, i.e., it has break points, and $\boldsymbol{f}$ has two C^0 components. Such a curve is normally considered unsuited for classical high order interpolation, and indeed the arc-length method gives low order algebraic convergence. So does the chord method, as Figure 6 shows.

The new methods, on the other hand, give exponential convergence. The interpolation points are clustered close to the break points, so the implicitly defined reparametrization is almost stationary at these points.

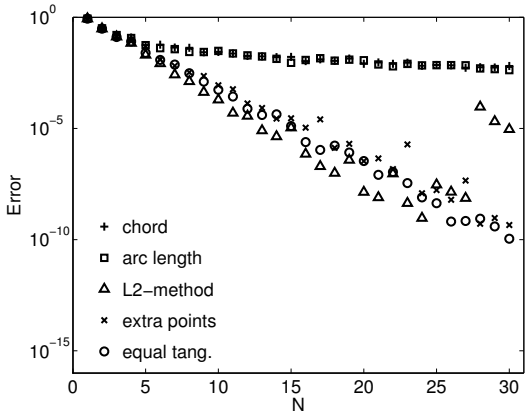


Figure 6: Interpolation error for Case 5. The chord and arc-length methods yield low order algebraic convergence, whereas the new methods converge exponentially, except for some instability for large N.

Case 6

The parametric curve

$$\begin{aligned} f_1(\eta) &= \eta, \\ f_2(\eta) &= \frac{1}{1+16\eta^2}, \\ f_3(\eta) &= \frac{1}{1+16(\eta+1)^2}, \end{aligned}$$

is difficult to interpolate since two of its components are Runge functions. It was shown in [25] that the Runge function can be very well approximated (without oscillations) when viewed as a planar parametric curve. Here, one of the functions is shifted along the x-axis so that the curve is not equivalent to the standard Runge function.

The chord method and the arc-length method both result in very low convergence rates; see Figure 7. The chord method yields unwanted oscillations in the solution, whereas the arc-length method method results in a poor approximation of $f_2(\eta)$ around $\eta = 0$. The three new methods converge fast in comparison, reaching machine precision around $N = 30$. These methods yield practically no unwanted oscillations.

4 Interpolation of parametric surfaces

Let $\boldsymbol{f}$ be a parametric surface in $\mathbb{R}^3$, described in a Cartesian coordinate system by

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} f_1(\eta_1,\eta_2) \\ f_2(\eta_1,\eta_2) \\ f_3(\eta_1,\eta_2) \end{bmatrix} = \boldsymbol{f}(\eta_1,\eta_2), \qquad \eta_1,\eta_2 \in [-1,1]. \tag{23}$$

The parametrization can be viewed as mapping $\boldsymbol{f}$ from a *reference domain* $\widehat{\Omega} = [-1,1] \times [-1,1] \subset \mathbb{R}^2$ to a physical domain $\Omega \in \mathbb{R}^3$. A reparametrization of the surface can be

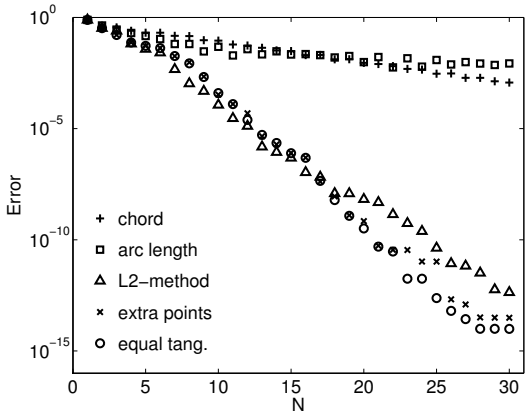


Figure 7: Interpolation error for Case 6. The chord and arc-length methods yield very slow convergence. Again there is a huge gap in performance between these methods and the three new methods.

found by a change of variables: for all bijective maps φ from $\widehat{\Omega}$ onto itself, the function

$$\boldsymbol{g}(\xi_1, \xi_2) = \boldsymbol{f}(\varphi(\xi_1, \xi_2)) = \boldsymbol{f}(\eta_1, \eta_2)$$

describes the same surface. Note that ξ_1 and ξ_2 here represent two independent variables and not two GLL points; it should be clear from the context what is meant. Different parametrizations can consist of functions of different regularity, and this will affect the convergence rate in polynomial interpolation.

The interpolant is a parametric surface $\boldsymbol{p}$ described by

$$\boldsymbol{p}(\xi_1, \xi_2) = \begin{bmatrix} p_1(\xi_1, \xi_2) \\ p_2(\xi_1, \xi_2) \\ p_3(\xi_1, \xi_2) \end{bmatrix}, \qquad \xi_1, \xi_2 \in [-1, 1],$$

where each component p_i is a polynomial of degree less than or equal to N in each reference variable. It is conveniently represented by sums of Lagrangian interpolants in the tensor-product GLL points, i.e.,

$$p_i(\xi_1, \xi_2) = \sum_{m=0}^{N} \sum_{n=0}^{N} \alpha_{mn}^i \ell_m(\xi_1) \ell_n(\xi_2), \qquad i = 1, 2, 3. \tag{24}$$

We note that, for a given ξ_1, $\boldsymbol{p}$ is a parametric curve when viewed as a function of ξ_2, and vice versa. Hence, the rectilinear mesh that is made up by the interpolation points in $\widehat{\Omega}$ is mapped to a curvilinear mesh on Ω; see Figure 8.

The expansion coefficients are determined by letting them be points somewhere on the exact surface, i.e., $\alpha_{mn}^i = f_i(\eta_{1,mn}, \eta_{2,mn})$. The coordinates $\eta_{1,mn}$ and $\eta_{2,mn}$ in $\widehat{\Omega}$ can be viewed as *free parameters* that implicitly determine the change of variable φ_N, a polynomial of degree N in ξ_1 and ξ_2 such that $(\eta_{1,mn}, \eta_{2,mn}) = \varphi_N(\xi_m, \xi_n)$, $0 \leq m, n \leq N$, i.e., $(\eta_{1,mn}, \eta_{2,mn})$ are the images of the tensor-product GLL points. There are $2(N+1)^2$ values to be interpolated; hence the dimension of the discrete space is $2(N+1)^2$.

We want the interpolation methods described in this paper to be applicable in the context of high order methods for solving PDEs in deformed hexahedra. The construction of a numerical approximation of the hexahedron often starts with an interpolation of the six faces, followed by a transfinite interpolation method to patch them together. For the

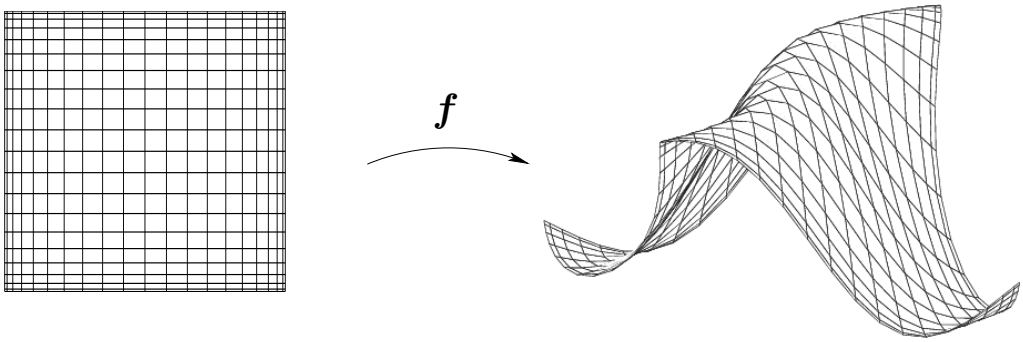


Figure 8: The surface is mapped from a reference domain $\widehat{\Omega} = [-1,1]\times[-1,1]$ by the parametrization $\boldsymbol{f}$. The interpolation points are mapped from the tensor-product GLL points.

latter to be possible, a consistent representation of the shared edges is necessary. This puts a few restrictions on the choice of $\eta_{1,mn}$ and $\eta_{2,mn}$. In particular, we require that the interpolation points that are mapped from the boundary of $\widehat{\Omega}$ interpolate the boundary of Ω, and that the corner points map to the corner points. This leaves $N-1$ free parameters on each of the four boundary curves of an individual face. Together with the $2(N-1)^2$ free parameters in the interior of Ω, we have a total of

$$2(N-1)^2 + 4(N-1) = 2N^2 - 2$$

degrees-of-freedom.

When each boundary curve is considered separately, these requirements are in essence the same as the requirements that we made for the curve interpolants in Section 2. This enables us to use the interpolation methods from Section 2 on each of the four boundary curves of Ω.

4.1 Interpolation error and approximation order

In order to be able to measure the interpolation error in a parametrization-independent norm, we will consider surfaces that can be represented as functions

$$z = h(x,y),$$

in a Cartesian coordinate system. This allows the use of the L^2-norm

$$||\boldsymbol{f} - \boldsymbol{p}|| = \Big(\iint_{\Pi_\Omega} \big(h(x,y) - h_N(x,y)\big)^2 \,\mathrm{d}x\,\mathrm{d}y \Big)^{1/2}, \tag{25}$$

where Π_Ω is the projection of Ω to the xy-plane and $h_N(x,y)$ is a parametrization-independent representation of the interpolant. The latter is not always readily available from the parametric description of the interpolant. However, in the case where f_1 and f_2 are affine in both variables the norm can (without error) be transformed to an integral over the reference domain $\widehat{\Omega}$,

$$||\boldsymbol{f} - \boldsymbol{p}|| = \Big(\int_{-1}^{1}\int_{-1}^{1} \big(f_3(\varphi_N(\xi_1,\xi_2)) - p_3(\xi_1,\xi_2)\big)^2 \, J \,\mathrm{d}\xi_1\,\mathrm{d}\xi_2 \Big)^{1/2}, \tag{26}$$

where

$$J = \frac{\partial f_1}{\partial \eta_1}\frac{\partial f_2}{\partial \eta_2} - \frac{\partial f_2}{\partial \eta_1}\frac{\partial f_1}{\partial \eta_2}$$

is the Jacobian of the mapping $\boldsymbol{f}$. To keep the evaluation of the norm simple, we restrict our investigation to parametric surfaces that fit this requirement. Note again that this is not a restriction for the interpolation methods. The error norm is implemented as a discrete version of (26), based on GLL quadrature with overintegration.

In the context of interpolation using a fixed polynomial degree N (and possibly using several patches to represent the entire surface), the concept of approximation order is relevant. With traditional interpolation methods, interpolation by polynomials of degree N gives approximation order $N+1$. For example, bicubic Bézier patches give fourth order convergence as the size of the patch decreases.

The conjectured optimal approximation order in curve interpolation was based on counting the number of equations that must be solved and comparing it with the number of free parameters. The same can be done in the context of interpolation of surfaces. Mørken [18] did it by counting the number of (non-linear) equations that must be solved in order to reduce the degree of a classical bivariate Taylor approximant without reducing the approximation order. He showed that the approximation order k is bounded by

$$k \leq \sqrt{3N^2 + 9N - 23/4} - \frac{1}{2},$$

which means that one theoretically can achieve approximation order $k = 2N$ for $N < 7$. However, the asymptotically optimal approximation order as N increases is approximately $k = \sqrt{3}N$.

As before, in the context of high order interpolation, convergence is more conveniently evaluated in terms of how the interpolation error decreases as a function of N. Interpolating a parametric surface $\boldsymbol{f}$ where all the components f_i are analytic will give exponential convergence, while low order components will give algebraic convergence. Again, the goal is exponential convergence with the highest possible rate for all surfaces. The note on approximation order serves only as indication of the possible improvement when the free parameters are chosen in a clever way.

4.2 Interpolation methods

Classical interpolation of the parametric surface $\boldsymbol{f}$ means discarding the possibility of reparametrization and setting $\alpha^i_{mn} = f_i(\xi_m, \xi_n)$. This corresponds to interpolation of the vector-valued *function* $\boldsymbol{f}$ in the tensor-product GLL points.

A simple, parametrization-independent alternative is to apply one of the curve interpolation method from Section 2 to the boundary curves of the surface, and then to find the internal points by a method for transfinite interpolation, e.g., the Gordon-Hall algorithm [13]. However, relying only on a transfinite interpolation method can yield a very crude approximation of the interior of the surface. If all three components of the interpolant are determined by the Gordon-Hall algorithm, the interior points will in general not be interpolation points. We therefore add a third step to make sure that all the coefficients in (24) are interpolation points. This will be our basic interpolation procedure:

1. Interpolate the boundary as four separate space curves.

2. Use Gordon-Hall transfinite interpolation for x and y.

3. Find z by function evaluation $h(x, y)$ at the internal interpolation points.

If the function $h(x, y)$ is not known, the last step would require an iterative procedure to find η_1 and η_2 from the system

$$x = f_1(\eta_1, \eta_2)$$
$$y = f_2(\eta_1, \eta_2)$$

in each internal interpolation point. The z-coordinate could then be found by evaluating $z = f_3(\eta_1, \eta_2)$.

This algorithm will be the basis for our extension of the chord method and the arc-length method to interpolation of surfaces. Hence, they are only applied to the boundary curves; the interpolation points in the interior are determined by steps 2 and 3. Since there is no natural way to define the chord or the arc-length across a surface, this is in fact the most natural way to extend the methods to surface interpolation.

Since the Gordon-Hall algorithm in principle represents the interior as a weighted sum of the boundaries, it is clear that given a smooth boundary representation, we get a smooth representation of the interior. However, we have no way of knowing if this will be an *optimal* representation of the interior. Hence, we add another step in the algorithm:

4. Apply a surface interpolation algorithm to improve the distribution of interpolation points in the interior.

There are two main reasons why we do not skip the first three steps and go directly to a surface interpolation algorithm for the entire surface. First, experience from numerical experiments have shown that a good representation of the boundary is sometimes the single most important factor in achieving a good representation of the surface. Secondly, the restriction we have made on the interpolation points on the boundary means that there is a difference between the boundary and the interior in the number of degrees-of-freedom associated with each interpolation point. Hence, a boundary point cannot be treated exactly like an interior point. This does not prohibit us from interpolating the entire surface simultaneously, but it makes it more natural to treat them separately.

The L^2-method can be defined as the optimization procedure to find the interpolant that minimizes the functional

$$\mathcal{J} = ||\boldsymbol{f} - \boldsymbol{p}||^2, \tag{27}$$

where $||\cdot||$ is the norm (25). For parametric surfaces $\boldsymbol{f}$ where f_1 and f_2 are affine, the method can be implemented based on the simpler form (26). Viewing the functional (27) as a function of the $2N^2 - 2$ independent variables $\eta_{1,mn}$ and $\eta_{2,mn}$, it is in principle possible to minimize it with Newton's method in the same way as was done for curves in Section 2.3. One can do this for the entire surface simultaneously, using all the free parameters, or one can apply it as step 4 in the algorithm, using only the free parameters in the interior.

Based on our experience with interpolation of parametrized curves, it should not come as a surprise that (27) is very hard to minimize. The rapidly increasing number of free parameters makes the method infeasible, and it will not be implemented here.

The extra-points method can also be extended to interpolation of surfaces. Due to the similarity between this method and the equal-tangents method, we choose to focus on only one of these methods in the context of surface interpolation. To avoid the dependence of the method on a choice of extra interpolation points, the equal-tangents method is chosen.

4.3 The equal-tangents method

When it comes to tangent and normal vectors, the situation for surfaces in $\mathbb{R}^3$ is in a sense opposite to the curve case: there is one unique surface normal and a two-dimensional

tangent plane. The equal-tangents method must therefore be based on requiring equal *tangent spaces* or, equivalently, equal normal vectors. This can be achieved by making a normal vector to the exact surface orthogonal to two linearly independent tangent vectors to the interpolant at the chosen interpolation points. The normal vector is given by

$$\boldsymbol{n} = \big(\frac{\partial f_2}{\partial \eta_1}\frac{\partial f_3}{\partial \eta_2} - \frac{\partial f_3}{\partial \eta_1}\frac{\partial f_2}{\partial \eta_2}, \frac{\partial f_3}{\partial \eta_1}\frac{\partial f_1}{\partial \eta_2} - \frac{\partial f_1}{\partial \eta_1}\frac{\partial f_3}{\partial \eta_2}, \frac{\partial f_1}{\partial \eta_1}\frac{\partial f_2}{\partial \eta_2} - \frac{\partial f_2}{\partial \eta_1}\frac{\partial f_1}{\partial \eta_2}\big)^T, \tag{28}$$

and the natural choice of tangent vectors is

$$\boldsymbol{t}_1^N = \big(\frac{\partial p_1}{\partial \xi_1}, \frac{\partial p_2}{\partial \xi_1}, \frac{\partial p_3}{\partial \xi_1}\big)^T \qquad \text{and} \qquad \boldsymbol{t}_2^N = \big(\frac{\partial p_1}{\partial \xi_2}, \frac{\partial p_2}{\partial \xi_2}, \frac{\partial p_3}{\partial \xi_2}\big)^T. \tag{29}$$

Equal tangent spaces is achieved when

$$\boldsymbol{n} \cdot \boldsymbol{t}_i^N = 0, \qquad i = 1, 2. \tag{30}$$

Just as in the case of curve interpolation, this means two equations have to be solved for equal tangents in one interpolation point. However, each interpolation point in the interior of Ω is associated with *two* degrees-of-freedom, as opposed to points on curves that only yield *one* degree-of-freedom. Hence, if we are able to solve the resulting system of non-linear equations, equal tangents should be possible in *all* the internal interpolation points. The boundary curves, on the other hand, can be interpolated with equal tangents in only $\lfloor (N-1)/2 \rfloor$ the points.

The equal-tangents method can be implemented either according to the four-step algorithm or as a method for the entire surface. In order to study the importance of a good representation of the boundary, we implement the method according to the four-step algorithm, and consider the solution before and after the last step. We will refer to the first as *equal-tangents boundary* and the second as *equal-tangents surface*.

All the systems of non-linear equations are solved with Newton's method, and the remarks from Section 2.7 still apply. Most importantly, the dependency on good initial guesses is important to achieve the best solution in the interior of Ω, and a bootstrapping method will be applied.

5 Numerical Results

Case 1

For surfaces that can be described by functions on the form

$$h(x, y) = f(x) + g(y), \qquad a \leq x \leq b, \quad c \leq y \leq d,$$

the Gordon-Hall method should be sufficient for an optimal representation of the entire surface, given that we are able to find an optimal representation of the boundary. The reason for this is that for a fixed x^*, the curve described by $h(x^*, y)$ is the same as the boundary curves $h(a, y)$ and $h(b, y)$, only shifted vertically. Hence, the optimal set of interpolation points is the same. For example, consider the surface described by the function

$$h(x, y) = \sqrt{x + 2} + \frac{1}{2}\arctan(y), \qquad -1 \leq x, y \leq 1. \tag{31}$$

The interpolation methods considered here are all based on a parametric description of the surface. This is trivial to find from the function description, using affine mappings f_1 and f_2.

Figure 9 shows that applying step 4 in the interpolation process, i.e., enforcing equal tangents also in the interior, makes no difference in the convergence rate. It is the representation of the boundary that separates the two equal-tangents methods from the other two methods.

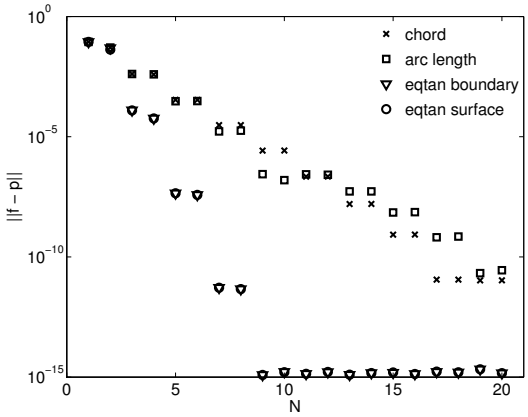


Figure 9: The interpolation error in Case 1, measured in the discrete L^2-norm. The surface can be described by a function of the form $h(x, y) = f(x) + g(y)$. With a good representation of the boundary, the Gordon-Hall algorithm is sufficient for a good representation of the entire surface.

Case 2

Consider the surface given by

$$h(x,y) = \frac{3}{2} + \frac{3}{10}\left(\frac{3}{2} - y\right)\,\sin\left(\frac{\pi}{3}x\right) + \frac{3}{10}\,y\,\cos\left(\frac{4}{3}\pi x\right), \qquad 0 \le x, y \le \frac{3}{2}. \tag{32}$$

Figure 10 shows that there is a vast difference between the chord method and the arc-length method in the convergence rate. Applying the equal-tangents method on the boundary gives only a slight improvement compared to the chord method, whereas the equal-tangents surface method gives a significant improvement. However, at $N = 9$ the latter seems to lose track of the optimal solution, and the convergence rate decreases dramatically. This is most likely due to a failure of Newton's method to find a solution to the equal-tangents problem in the interior. After all, at $N = 9$ there are $2(N-1)^2 = 128$ free parameters in the interior, far more than what we ever encountered in curve interpolation.

Case 3

Consider now a surface described by a function of low regularity,

$$h(x,y) = (x^2 + y^2)^{3/2}, \qquad -1 \le x, y \le 1. \tag{33}$$

Affine mappings f_1 and f_2 will make f_3 a function of low regularity in both η_1 and η_2, and classical interpolation will give low order algebraic convergence. In fact, the chord method corresponds to the classical interpolant, and Figure 11 confirms the poor convergence rate. The arc-length method works even worse for this surface.

The reason for the low regularity of f_3 is a singularity in the third partial derivatives at the origin. The boundary curves, on the other hand, are smooth functions that can be interpolated by high order polynomials to exponential convergence. This is a problem for

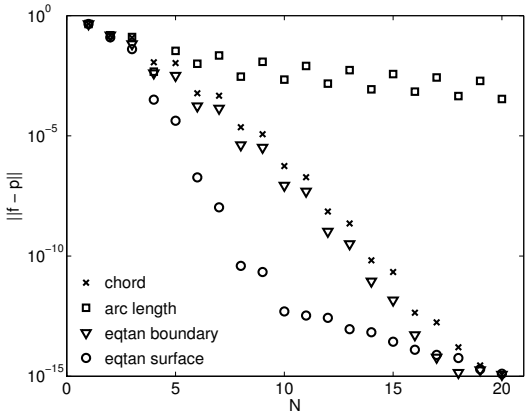


Figure 10: The interpolation error in Case 2. The surface is described by a smooth function, but the arc-length method results in a very low convergence rate. The equal-tangents surface method shows that very rapid convergence is possible, but we are not able to maintain the convergence rate until we reach machine precision, most likely due to the difficulty of finding solutions of the non-linear system of equations that arises from the equal-tangents conditions.

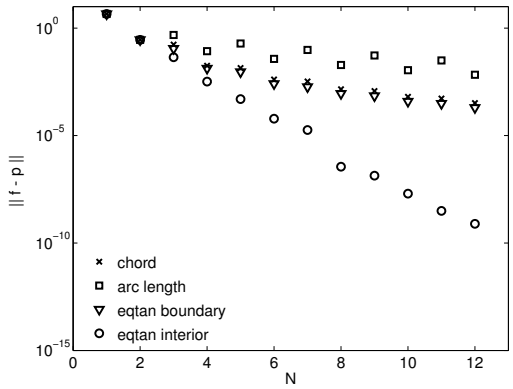


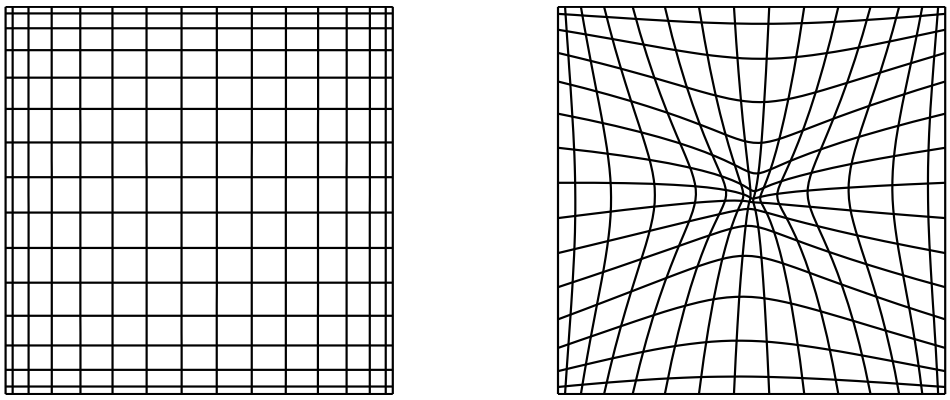
Figure 11: The interpolation error in Case 3. The surface is described by a function of low regularity, but the equal-tangents surface method finds a set of interpolation points that corresponds to interpolating a smooth reparametrization. This results in exponential convergence. The plot only extends to $N = 12$ because we are not able to maintain the same convergence rate for higher N.

the equal-tangents boundary method; it may give a good representation of the boundary, but the Gordon-Hall algorithm does not take the singularity at the origin into account. The result is low order algebraic convergence.

However, with the addition of the fourth step in the algorithm, we are indeed able to get exponential convergence. Figure 12 shows the mesh of interpolation points projected onto the xy-plane, with polynomial degree $N = 15$ and using the equal-tangents boundary and equal-tangents surface methods. The latter results in all internal interpolation points moving toward the origin, the position of the singularity. This corresponds to interpolating a reparametrization of the surface of higher regularity than $\boldsymbol{f}$ – hence the increased convergence rate.

The exact surface is rotationally symmetric around the z-axis, and one may therefore expect the optimal mesh of interpolation points to be rotationally symmetric as well. The mesh in Figure 12b is not entirely symmetric, but one should be careful with con-

cluding that a more symmetric mesh will give better approximation properties. It is in general impossible to predict the convergence rate from the mesh unless one knows which parametrization it interpolates.



(a) Equal-tangents boundary (b) Equal-tangents surface

Figure 12: The interpolant at $N = 15$, projected onto the xy-plane. Left: solution obtained using the equal-tangents boundary method. Right: solution obtained using the equal-tangents surface method. The latter yields a clustering of interpolation points around the origin, since this is the position of the singularity in the exact surface. The non-linear reparametrization enables us to achieve exponential convergence.

Case 4

The last surface is

$$h(x,y) = \frac{\arctan(2x)\,\sin(2x+(y+1)^2)}{1+x^2+y^2}, \qquad -1 \le x,y \le 1, \tag{34}$$

which is a little more complicated than the other surfaces. Both the chord method and the arc-length method give low exponential convergence rates. By applying the equal-tangents method on the boundary, we achieve a better convergence rate. When we apply equal-tangents in the interior as well, the convergence rate is improved even more. However, we see that the convergence rate decreases as N increases, most likely due to the failure of Newton's method to find the optimal solution.

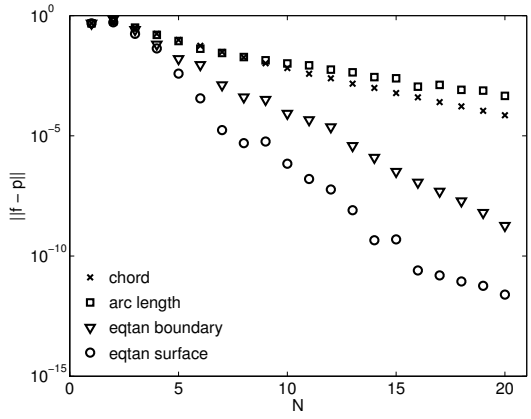


Figure 13: The interpolation error in Case 4.

6 Conclusion

High order interpolation of parametric curves and surfaces is an important part of the geometry representation in high order methods for solving PDEs in deformed rectangles and hexahedra, and the interpolation method chosen may have a big influence on the error in the numerical solution [25]. Still, the topic has received very little attention in the literature. On the other hand, a lot of work has been done on interpolation of parametric curves (and some on parametric surfaces) in the CAGD environment, but almost all of it concerns only *low* order polynomial interpolation.

Any parametric curve or surface can be reparametrized before being interpolated, and some reparametrizations will result in a smaller interpolation error than others. Finding the optimal reparametrization is in general a very difficult (and unsolved) problem. In the context of high order methods for solving PDEs, most authors settle with relatively simple and computationally inexpensive interpolation methods. The two most common methods of this kind is the chord method and the arc-length method, both of which are studied in this paper. These methods are based on heuristic arguments and rarely yield significantly better results than classical interpolation (i.e., interpolation without reparametrization).

In order to construct better interpolation methods for parametric curves, a new interpolation method (the L^2-method) is introduced. It is based on doing a direct minimization of the interpolation error using Newton's method. The method works quite well for low polynomial degrees N (although it is expensive), but for high N it often degrades, since Newton's method is not sufficient for finding the *global* minimum of the objective function.

In the CAGD community, interpolation methods based on parametrization-independent quantities such as tangents, curvature and torsion have been suggested, and they are referred to as *geometric Hermite interpolation* methods. These methods are conjectured to be optimal in terms of approximation order (as defined for interpolation using a *fixed* N), but they are costly and difficult to implement, since they require systems of non-linear equations to be solved. Two methods in the family of geometric Hermite interpolation are proposed, and they yield very good results. Some of the results are assumed to be very close to optimal, since they are approximately equal to the solution found by the L^2-method for low N, and they often maintain a constant convergence rate until machine precision.

Some of the interpolation methods are extended to interpolation of parametric surfaces, and the relative performance of the different methods is often similar to the curve interpolation results. However, in the context of surfaces one can choose to apply costly interpolation methods only on the boundary, or one can do it over the entire surface. In some cases the former is enough to achieve a vast improvement from classical interpolation, but sometimes one needs to consider the entire surface to achieve any significant improvement.

Some important commonly known limitations of high order interpolation are challenged when considering interpolation of parametric curves and surfaces, because of the option of reparametrization. For example, curves and surfaces of low regularity can be interpolated to exponential convergence (as a function of N), as shown by examples in this paper. Whether this is possible for *all* curves and surfaces of low regularity is a topic for future work.

Another such limitation is the Runge phenomenon, which describes the unwanted oscillations displayed by the interpolant for certain functions and point distributions. Examples from this paper and from [25] show that this can be avoided by reparametrization. In fact, in all the numerical experiments considered, we have seen no examples of curves that

could not be interpolated without oscillations. This includes the Runge function (viewed as a parametric curve), C^0 curves and functions with boundary layers (again, viewed as a parametric curve). Verifying (or disproving) the claim that any parametric curve can be interpolated by high order polynomials without oscillations is a topic for future work.

Acknowledgments

The work has been supported by the Research Council of Norway under contract 185336/V30. The author would like to thank Prof. E.M. Rønquist for many helpful comments and suggestions throughout the work on this paper.

References

[1] B.A. Barsky and T.D. DeRose. Geometric continuity of parametric curves: three equivalent characterizations. *Computer Graphics and Applications, IEEE*, 9(6):60–69, 1989.

[2] C. Canuto, M. Y. Hussaini, A. Quarteroni, and T. A. Zang. *Spectral Methods, Fundamentals in Single Domains*. Springer, 2006.

[3] C. Canuto, M. Y. Hussaini, A. Quarteroni, and T. A. Zang. *Spectral Methods, Evolution to Complex Geometries and Applications to Fluid Dynamics*. Springer, 2007.

[4] X.D. Chen, W. Ma, and J. Zheng. Geometric interpolation method in R^3 space with optimal approximation order. *Computer-Aided Design & Applications*, 7(6):919–928, 2010.

[5] C. de Boor, K. Höllig, and M. Sabin. High accuracy geometric Hermite interpolation. *Computer Aided Geometric Design*, 4(4):269–278, 1987.

[6] W.L.F. Degen. Best approximations of parametric curves by splines. In *Geometric modelling*, pages 59–73. Springer-Verlag, 1993.

[7] W.L.F. Degen. High accurate rational approximation of parametric curves. *Computer Aided Geometric Design*, 10(3-4):293–313, 1993.

[8] W.L.F. Degen. Geometric Hermite interpolation – in memoriam Josef Hoschek. *Computer Aided Geometric Design*, 22(7):573–592, 2005.

[9] M.O. Deville, P.F. Fischer, and E.H. Mund. *High-Order Methods for Incompressible Fluid Flow*. Cambridge University Press, 2002.

[10] Y.Y. Feng and J. Kozak. On G2 continuous cubic spline interpolation. *BIT Numerical Mathematics*, 37(2):312–332, 1997.

[11] Y.Y. Feng and J. Kozak. On spline interpolation of space data. *Mathematical Methods for Curves and Surfaces II, M. Dœhlen, T. Lyche, and LL Schumaker (eds.), Vanderbilt University Press, Nashville*, pages 167–174, 1998.

[12] M.S. Floater. An $O(h^{2n})$ Hermite approximation for conic sections. *Computer Aided Geometric Design*, 14(2):135–151, 1997.

[13] W. Gordon and C. Hall. Construction of curvilinear co-ordinate systems and applications to mesh generation. *International Journal for Numerical Methods in Engineering*, 7:461–477, 1973.

[14] K. Hollig and J. Koch. Geometric Hermite interpolation. *Computer Aided Geometric Design*, 12(6):567–580, 1995.

[15] G. Jaklič, J. Kozak, M. Krajnc, V. Vitrih, and E. Žagar. On geometric Lagrange interpolation by quadratic parametric patches. *Computer Aided Geometric Design*, 25(6):373–384, 2008.

[16] G. Jaklič, J. Kozak, M. Krajnc, and E. Žagar. On geometric interpolation by planar parametric polynomial curves. *Mathematics of Computation*, 76(260):1981, 2007.

[17] Y. Maday and E.M. Rønquist. Optimal error analysis of spectral methods with emphasis on non-constant coefficients and deformed geometries. *Computer Methods in Applied Mechanics and Engineering*, 80(1-3):91–115, 1990.

[18] K. Mørken. On geometric interpolation of parametric surfaces. *Computer Aided Geometric Design*, 22(9):838–848, 2005.

[19] K. Mørken and K. Scherer. A general framework for high-accuracy parametric interpolation. *Mathematics of Computation*, 66(217):237–260, 1997.

[20] C.W. Patty. *Foundations of Topology*. Jones & Bartlett Publishers, Inc., 2nd edition, 2009.

[21] A. Rababah. High order approximation method for curves. *Computer Aided Geometric Design*, 12(1):89–102, 1995.

[22] A. Rababah. High accuracy Hermite approximation for space curves in $\mathbb{R}^d$. *Journal of mathematical analysis and applications*, 325(2):920–931, 2007.

[23] R. Schaback. Interpolation with piecewise quadratic visually C2 Bézier polynomials. *Computer Aided Geometric Design*, 6(3):219–233, 1989.

[24] K. Scherer. Parametric polynomial curves of local approximation of order 8. *Curve and Surface Fitting: Saint-Malo 99*, pages 375–384, 2000.

[25] E.M. Rønquist T. Bjøntegaard and Ø. Tråsdahl. High order interpolation of curves in the plane. Technical report, Norwegian University of Science and Technology, http://www.math.ntnu.no/preprint/numerics/2009/N11-2009.pdf, 2009.

[26] L. Xu and J. Shi. Geometric Hermite interpolation for space curves. *Computer aided geometric design*, 18(9):817–829, 2001.

[27] J.H. Yong and F.F. Cheng. Geometric Hermite curves with minimum strain energy. *Computer Aided Geometric Design*, 21(3):281–301, 2004.