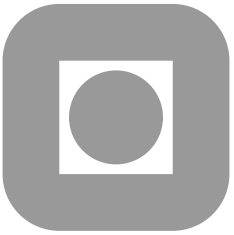
NORGES TEKNISK-NATURVITENSKAPELIGE
UNIVERSITET

High order interpolation of curves in the plane

by

Tormod Bjøntegaard, Einar M. Rønquist and Øystein Tråsdahl

PREPRINT
NUMERICS NO. 11/2009



NORWEGIAN UNIVERSITY OF
SCIENCE AND TECHNOLOGY
TRONDHEIM, NORWAY

This report has URL
http://www.math.ntnu.no/preprint/numerics/2009/N11-2009.pdf
Address: Department of Mathematical Sciences, Norwegian University of Science and Technology, N-7491 Trondheim, Norway.

High order interpolation of curves in the plane

Tormod Bjøntegaard, Einar M. Rønquist and Øystein Tråsdahl

December 18, 2009

We consider high order representations of curves in the plane. Through a series of complementary numerical examples, we show that classical interpolation is sometimes far from optimal in the sense of minimizing the interpolation error using a fixed number of degrees-of-freedom (the Kolmogorov n-width problem). We propose a new way of constructing a high order interpolant of a curve in the plane. The main ingredients are: a parametric representation of the curve, an implicit reparametrization of the curve, and choosing the internal interpolation points in such a way that the tangent vectors of the exact curve and the tangent vectors of the interpolant coincide at these points. Numerical results indicate that the proposed interpolant is close to optimal in the sense of minimizing the L^2-error between the exact curve and its interpolant. We also show that the error may decrease exponentially fast even for curves which are not analytic when regarded as a function in the classical setting. Finally, we use the proposed method to construct an improved representation of the boundary of a deformed quadrilateral domain. We show that the discretization error associated with solving the Poisson problem in the deformed domain may be significantly smaller than the error resulting from a standard approach.

Keywords Polynomial interpolation, reparametrization, high order methods, approximation of curves

1 Introduction

The motivation behind the work presented in this paper has been to solve partial differential equations in complex domains using high order methods. As a simple example, consider the numerical solution of the Poisson problem in a deformed quadrilateral domain Ω. A numerical solution based on high order polynomials necessitates an accurate representation of the geometry. This is typically achieved by first constructing an accurate representation of the boundary of the domain, $\partial\Omega$, and then constructing a mapping between the reference domain $\hat{\Omega} = (-1, 1)^2$ and Ω. Assuming that the physical domain is not too distorted, the latter can readily be achieved via a Gordon-Hall transfinite interpolation procedure [7].

Despite the fact that spectral (element) methods have been used to solve PDEs in complex geometries for a long time [4, 5, 8, 9], few results exist in the literature for how

Corresponding author: Einar M. Rønquist. Tel.: +47 73 59 35 47; fax: +47 73 59 35 24.
E-mail address: ronquist@math.ntnu.no

to *best* construct a high order representation of a single curve in the plane. In the case of a deformed quadrilateral, we need to approximate four curves in the plane (the four edges of Ω) before we are able to construct the mapping between $\hat{\Omega}$ and Ω.

In this paper we focus on different ways of approximating a curve using high order polynomials. We compare previously used methods with a new method. The numerical results indicate that there is a potential to do better than currently used methods, sometimes much better. In addition, the insight provided by the numerical results also suggests new ways of constructing high order interpolants of a general function $y(x)$.

The outline of the paper is a follows. In Section 2 we discuss high order representations of one-dimensional functions. In Section 3 we apply the results and insight from Section 2 to approximate the edges of a deformed quadrilateral domain. We then solve a Laplace problem in this domain and discuss the impact of the geometry representation on the discretization error. Finally, in Section 4 we summarize our main findings.

2 High-order representation of a one-dimensional function

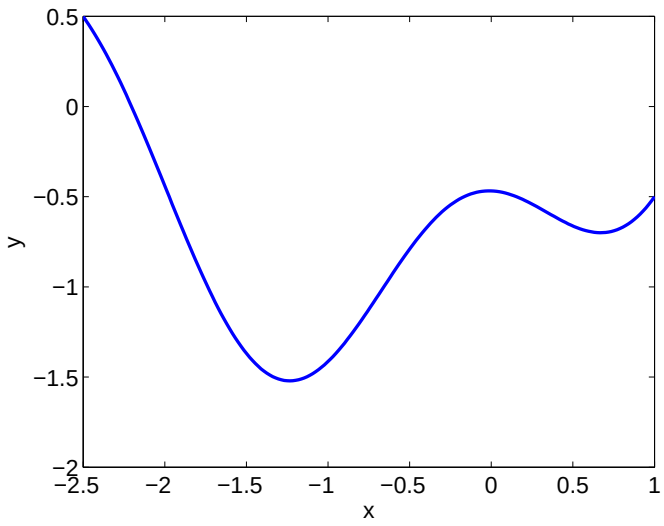


Figure 1: A function $y(x)$, $x \in [a, b]$; here $a = -2.5$ and $b = 1$.

Consider a one-dimensional function $y(x)$, $x \in [a, b]$, as depicted in Figure 1. High order approximation of such a function is a classical topic, and there is a vast literature covering both theoretical and numerical aspects, e.g., see [1, 2, 3]. For example, a high order interpolant can be constructed by choosing points x_j, $j = 0, \ldots, N$, and evaluating the function at these points, yielding $y_j = y(x_j)$, $j = 0, \ldots, N$. Based on these values, an interpolating polynomial $y_N(x)$ of degree N can be constructed. For example, the points x_j can be chosen as $x_j = a + \frac{b-a}{2}(\xi_j + 1)$, where ξ_j, $j = 0, \ldots, N$, are the Gauss-Lobatto Legendre (GLL) points. We have here tacitly assumed an *affine* mapping $x(\xi)$ given explicitly as $x(\xi) = a + \frac{b-a}{2}(\xi + 1)$ for $\xi \in [-1, 1]$, with $y(\xi) = y(x(\xi))$. Note that, in order to keep the notation as simple as possible, we will use the same symbol for the function y both when expressed in terms of the physical coordinate x and in terms of the reference coordinate ξ. It should be clear from the context what is meant. If not, we will provide explicit reference to the independent variable.

If the function $y(x)$ is regular, the interpolation error $\|y - y_N\|$ will decay rapidly as N

increases. In addition, the interpolation error will be close to the best approximation error (or projection error).

Since the mapping $x(\xi)$ is affine, both the function $y_N(x)$ and $y_N(\xi) = y_N(x(\xi))$ are polynomials of degree N. Specifically, the function $y_N(\xi)$ is given as

$$y_N(\xi) = \sum_{j=0}^{N} y_j \ell_j(\xi), \tag{1}$$

where $\ell_j(\xi)$ is the Nth order Lagrangian interpolant through the GLL points, i.e., $\ell_j(\xi) \in \mathbf{P}_N(-1,1)$ and $\ell_j(\xi_i) = \delta_{ij}$, $0 \le i,j \le N$, and where $y_j = y(x(\xi_j))$.

In the case of approximating a curve in the plane, a common approach is to view the curve as a given function $y(x)$. However, the specific function $y(x)$ will depend on the orientation of our coordinate system and the representation of the curve is therefore not unique. Hence, a high order polynomial approximation of the curve will also depend on our choice of coordinate system.

We may also describe a curve through a parametric representation. For example, a curve in the plane can be given by two functions $g_1(\xi)$ and $g_2(\xi)$,

$$\begin{aligned} x(\xi) &= g_1(\xi), \\ y(\xi) &= g_2(\xi). \end{aligned} \tag{2}$$

For any value value $\xi \in [-1,1]$, there exists a unique point $(x(\xi), y(\xi))$ on the curve. For example, if $g_1(\xi) = a + \frac{b-a}{2}(\xi+1)$, i.e., the affine mapping we discussed above, we may eliminate ξ by first expressing ξ as a function of x, and then recover the original function $y(x)$ from $y(x) = g_2(\xi(x))$. However, the parametric representation (2) allows for additional flexibility, also when it comes to numerical approximation.

In the following we consider high order polynomial approximations $x_N(\xi)$ and $y_N(\xi)$ of $x(\xi)$ and $y(\xi)$. Both $x_N(\xi)$ and $y_N(\xi)$ are elements of $\mathbf{P}_N(-1,1)$. These approximations can be expressed explicitly as

$$\begin{aligned} x_N(\xi) &= \sum_{j=0}^{N} x_j \ell_j(\xi), \\ y_N(\xi) &= \sum_{j=0}^{N} y_j \ell_j(\xi), \end{aligned} \tag{3}$$

where x_j and y_j are the nodal values for each approximation. We will here only consider interpolants to the exact curve $(x(\xi), y(\xi))$, $\xi \in [-1,1]$, i.e., we require that the nodal points (x_j, y_j), $j = 0, \ldots, N$, are located on the exact curve. We also assume that the end points of the exact curve and the numerical curve always coincide. Hence, our problem is to determine the "best" set of (internal) values ξ_j^*, $j = 1, \ldots, N-1$, $-1 < \xi_j^* < 1$, where $x_j = g_1(\xi_j^*)$ and $y_j = g_2(\xi_j^*)$, i.e., this is a problem with $N-1$ degrees-of-freedom.

Let us now briefly consider a few choices which can be used to represent a curve in the plane, e.g., an edge of a deformed quadrilateral domain.

2.1 The standard method

The first method is precisely the very first approach we discussed, where $x(\xi) = g_1(\xi)$ is a simple affine mapping. The numerical approximation is given by (3), with nodal values $x_j = a + \frac{b-a}{2}(\xi_j + 1)$ and $y_j = y(x_j)$; see Figure 2. By construction, $y_N(\xi)$ is a polynomial

of degree N. However, we also note that $y_N(x)$ will also be a polynomial of degree N, i.e., we consider classical interpolation. In the following, we refer to this method as the "standard" method.

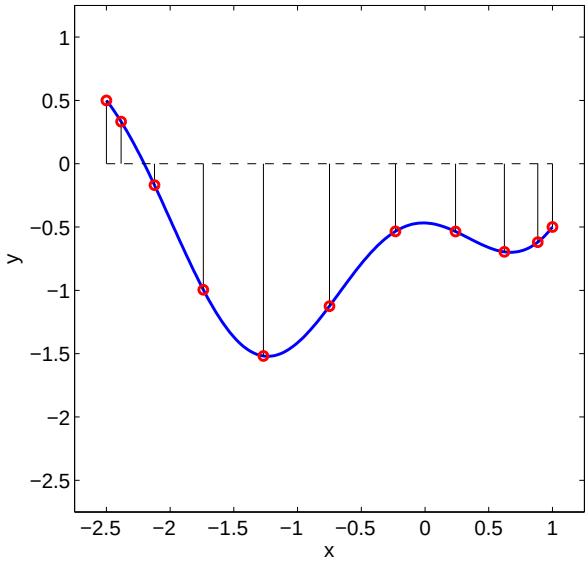


Figure 2: The function $y(x)$ evaluated at the (affinely mapped) GLL-nodes.

2.2 The chord method

Even if the original curve is represented in a coordinate system as depicted in Figure 1, we can always transform this representation to a coordinate system where the x-axis is aligned with the chord connecting the two end points of the curve; see Figure 3. This can be achieved through a simple rotation and translation. After the transformation to this new coordinate system (with coordinates x' and y'), we can again apply the standard method as described above.

Note that the transformed function $y'(x')$ is different from the original function $y(x)$ even though both functions describe the same curve. This will, of course, also be true for the associated numerical approximations (x'_N, y'_N) and (x_N, y_N).

Also note that the parametric representation of the exact curve $(x'(\eta), y'(\eta))$, $\eta \in [-1, 1]$, can be viewed as a reparametrization [11] of the original parametric representation $(x(\xi), y(\xi))$, $\xi \in [-1, 1]$. To this end, there exists a mapping $\xi(\eta)$ from $[-1, 1]$ to $[-1, 1]$ such that $x'(\eta) = x(\xi(\eta))$ and $y'(\eta) = y(\xi(\eta))$. In essence, the two parametric representations differ in the sense that we are moving along the same curve at different "speeds".

Using the standard method in the transformed coordinate system, we end up with a set of nodal values (x'_j, y'_j), $j = 0, \ldots, N$. These coordinates are then transformed back to the corresponding coordinates (x_j, y_j), $j = 0, \ldots, N$, in the original coordinate system. The reason for transforming back to the original coordinate system is motivated by the fact that we ultimately want to use these representations in the context of solving partial differential equations. In this case, all the curve segments associated with the boundary of the computational domain are part of a common description. For example, these representations may be the input to a Gordon-Hall algorithm as discussed in the Introduction.

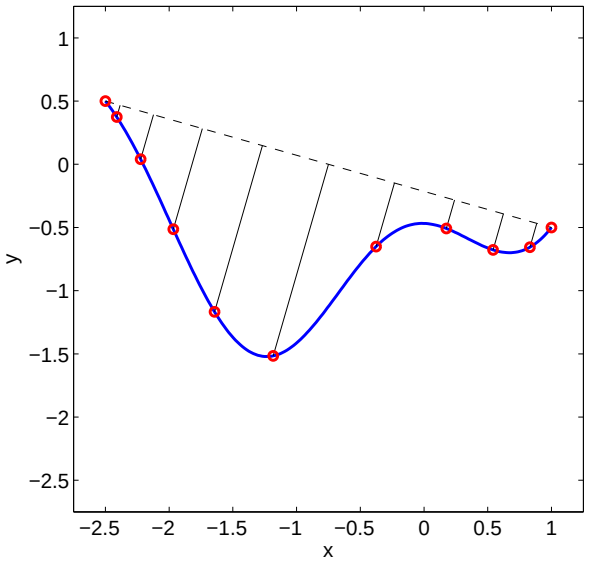


Figure 3: The original function $y(x)$ can be considered in a transformed coordinate system in which the transformed x-axis is aligned with the chord between the two end points of the curve. The function is then evaluated at the (affinely mapped) GLL-nodes relative to this coordinate system.

The numerical representation of the curve will therefore be exactly the same as before and given by (3). However, the *numerical* values of (x_j, y_j), $j = 0, \ldots, N$, will be different than the standard method applied in the original coordinate system. In the following, we will refer to this method as the "chord" method.

Note that, even though $x_N(\xi)$ and $y_N(\xi)$ both belong to $\mathbf{P}_N(-1, 1)$, the implicitly given function $y_N(x_N)$ is not, in general, a polynomial. In the particular case when the chord is parallel to the x-axis, the chord method coincides with the standard method.

2.3 The arc length method

Let s be the arc length as we move along the curve $y(x)$, with $s = 0$ at $x = a$ and $s = L$ at $x = b$, i.e., the length of the curve is L. Construct the affine mapping $s(\xi) = \frac{L}{2}(\xi + 1)$, $\xi \in [-1, 1]$, and define the values $s_j = s(\xi_j)$, $j = 0, \ldots, N$. Each value s_j corresponds to a unique point along the curve with coordinates (x_j, y_j); see Figure 4. The arc length method uses these points as the nodal values in the numerical representation (3). In other words, the nodal points are distributed along the curve as a GLL-distribution according to arc length. As for the chord method we note that, even though $x_N(\xi)$ and $y_N(\xi)$ both belong to $\mathbf{P}_N(-1, 1)$, the implicitly given function $y_N(x_N)$ is not, in general, a polynomial.

2.4 The L^2-method

The three previous representations all rely in one way or another on a point distribution which corresponds to an affinely mapped GLL-distribution: for the standard method, the points along the x-axis correspond to a GLL-distribution; for the chord method, the points along the chord correspond to a GLL-distribution; for the arc length method, the points along the arc of the curve correspond to a GLL-distribution.

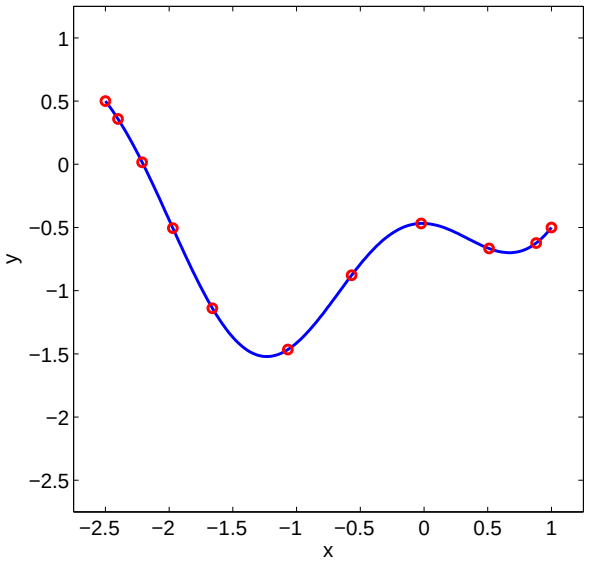


Figure 4: The original function $y(x)$ is here evaluated at points corresponding to a GLL-distribution according to arc length.

We can, of course, also propose a method where there are no such restrictions on the nodal points. Instead, we search for $N-1$ internal points along the curve which will ensure a good numerical approximation to the exact curve.

As a way to measure how good the approximation is, we consider the L^2-error

$$\mathcal{J} = ||y - y_N||_{L^2} = \left(\int_a^b (y(x) - y_N(x))^2 \mathrm{d}x \right)^{1/2}. \tag{4}$$

Another way of measuring the error is by comparing the arc length of the exact curve with the arc length of the numerical approximation. The arc length of a curve parametrized as $(x(\xi), y(\xi))$ is given as

$$L = \int_{-1}^{1} \sqrt{x'(\xi)^2 + y'(\xi)^2}\, \mathrm{d}\xi, \tag{5}$$

and the arc length L_N of the numerical approximation is given analogously.

The exact curve is given by the parametric representation $(x(\xi), y(\xi))$, $\xi \in [-1, 1]$, while the numerical approximation is given by (3). Again, we assume that the end points of the exact curve coincide with the end points of the numerical curve, and the internal points (x_j, y_j), $j = 1, \ldots, N-1$, in (3) are all unique and located somewhere along the exact curve. As stated earlier, our problem is to determine the "best" set of (internal) values ξ_j^*, $j = 1, \ldots, N-1$, $-1 < \xi_j^* < 1$, where $x_j = g_1(\xi_j^*)$ and $y_j = g_2(\xi_j^*)$, i.e., this is a problem with $N-1$ degrees-of-freedom. As before, the resulting numerical approximation will be an interpolant, however, the implicitly defined interpolating function $y_N(x_N)$ will not, in general, be a polynomial even though $x_N(\xi)$ and $y_N(\xi)$ are both in $\mathbf{P}_N(-1, 1)$.

In principle, the $N-1$ values ξ_j^*, $j = 1, \ldots, N-1$, $-1 < \xi_j^* < 1$, can be determined by defining $N-1$ independent conditions. A natural idea is to derive these conditions by minimizing the functional $\mathcal{J}$, i.e., by requiring that the L^2-error be minimized. We will refer to such a method as the "L^2-method". The resulting minimizer can be viewed as the solution to the Kolmogorov n-width problem applied to the interpolation of curves [12].

In principle, a Newton method can be used to solve the system of nonlinear equations. Unfortunately, we will later see that it is often very difficult to find the global minimum of $\mathcal{J}$ and we therefore propose an alternative method.

2.5 The equal-tangent method

As mentioned in the previous section, the $N-1$ values ξ_j^*, $j = 1, \ldots, N-1$, $-1 < \xi_j^* < 1$, can be determined by defining $N-1$ independent conditions. As an alternative to the L^2-method we propose the following conditions:

$$\frac{\mathrm{d}x_N}{\mathrm{d}\xi}(\xi_j)\frac{\mathrm{d}y}{\mathrm{d}\xi}(\xi_j^*) - \frac{\mathrm{d}y_N}{\mathrm{d}\xi}(\xi_j)\frac{\mathrm{d}x}{\mathrm{d}\xi}(\xi_j^*) = 0, \qquad j = 1, \ldots, N-1. \tag{6}$$

One way to better understand these equations is by considering the exact curve subject to a reparametrization $\xi(\eta)$, $\eta \in [-1, 1]$, $\xi \in [-1, 1]$. That is, we consider the representation $(\tilde{x}(\eta), \tilde{y}(\eta)) = (x(\xi(\eta)), y(\xi(\eta)))$ which, of course, represents the same curve. The only difference between the representation $(\tilde{x}(\eta), \tilde{y}(\eta))$, $-1 \leq \eta \leq 1$, and $(x(\xi), y(\xi))$, $-1 \leq \xi \leq 1$, is that we move along the exact curve at different "speeds". With this approach, it is possible to find a reparametrization such that the conditions (6) can be expressed as

$$\frac{\mathrm{d}x_N}{\mathrm{d}\xi}(\xi_j)\frac{\mathrm{d}\tilde{y}}{\mathrm{d}\eta}(\xi_j) - \frac{\mathrm{d}y_N}{\mathrm{d}\xi}(\xi_j)\frac{\mathrm{d}\tilde{x}}{\mathrm{d}\eta}(\xi_j) = 0, \qquad j = 1, \ldots, N-1. \tag{7}$$

Hence, by solving the system (6), we are implicitly finding a reparametrization of the exact curve such that the exact curve $(\tilde{x}(\eta), \tilde{y}(\eta))$ and the numerical approximation $(x_N(\xi), y_N(\xi))$ have equal tangent vectors at the internal GLL points ξ_j, $j = 1, \ldots, N-1$; see Figure 5. We will refer to this method as the equal-tangent method. A Newton method will be used to solve the system of nonlinear equations (6).

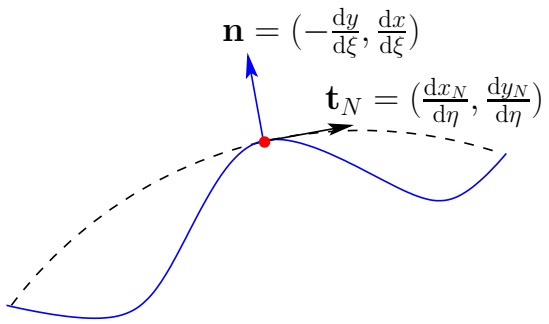


Figure 5: The solid (blue) analytical curve and the dashed (black) numerical curve with associated normal and tangent vectors. Here, $N = 2$. At the internal interpolation points (only one in this example), we require that the analytical normal vector, $\mathbf{n}$, and the numerically computed tangent vector, $\mathbf{t}_N$, are orthogonal.

2.6 Numerical results: curves in the plane

We now present several examples illustrating the different alternative high order representations of curves in the plane. The exact curves have been selected in order to illustrate the behavior of the various methods in different situations. Some additional remarks regarding the implementation are given at the end of Section 2.

2.6.1 Case 1

The first example is $2\pi/3$ radians of a circle with radius $r = 1$ as described by the function $y(x) = \sqrt{1 - x^2}$, $x \in [-\sin(\frac{\pi}{3}), \sin(\frac{\pi}{3})]$; see Figure 6. The two most common parametrizations of this function are

$$x(\xi) = -\frac{\sqrt{3}}{2} + \frac{\sqrt{3}}{2}(\xi + 1),$$
$$y(\xi) = \sqrt{1 - (x(\xi))^2},$$

and

$$x(\xi) = \sin(\theta(\xi)),$$
$$y(\xi) = \cos(\theta(\xi)),$$

with $\theta(\xi) = -\frac{\pi}{3} + \frac{\pi}{3}(\xi + 1)$. For both parametrizations, $-1 \leq \xi \leq 1$.

If we define the interpolation points to be $(x(\xi_j), y(\xi_j))$, $j = 0, \ldots, N$, using the parametrizations above, then the first parametrization yields the standard method and the second parametrization yields the arc length method. As the trigonometric functions are very smooth and can be very well approximated by polynomials (over short intervals), the arc length method is the better of the two alternatives.

We compute the interpolants for all the methods for polynomial degrees from $N = 2$ to $N = 30$ and compare the interpolation error (4) in Figure 7. All the interpolation methods yield spectral convergence, as expected. However, it is surprising to observe how much potential accuracy is lost by choosing the standard method. Approximately three times the polynomial degree is needed to achieve the same level of accuracy as the arc length method. In the context of solving partial differential equations in deformed domains, this can have a tremendous impact on the computational cost to reach a desired accuracy, particularly in higher space dimensions.

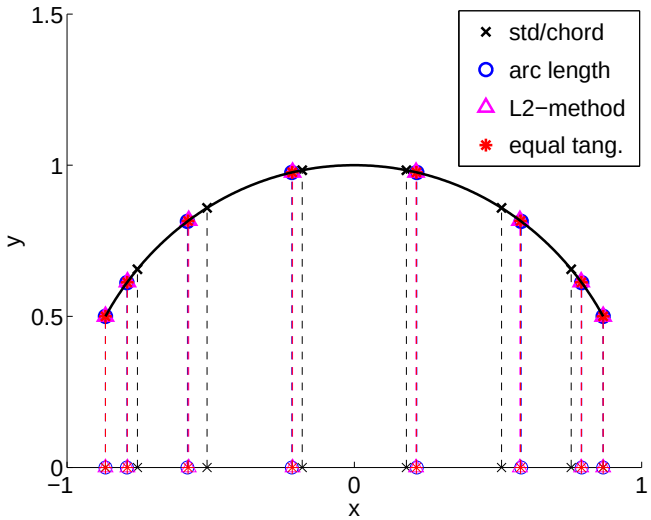


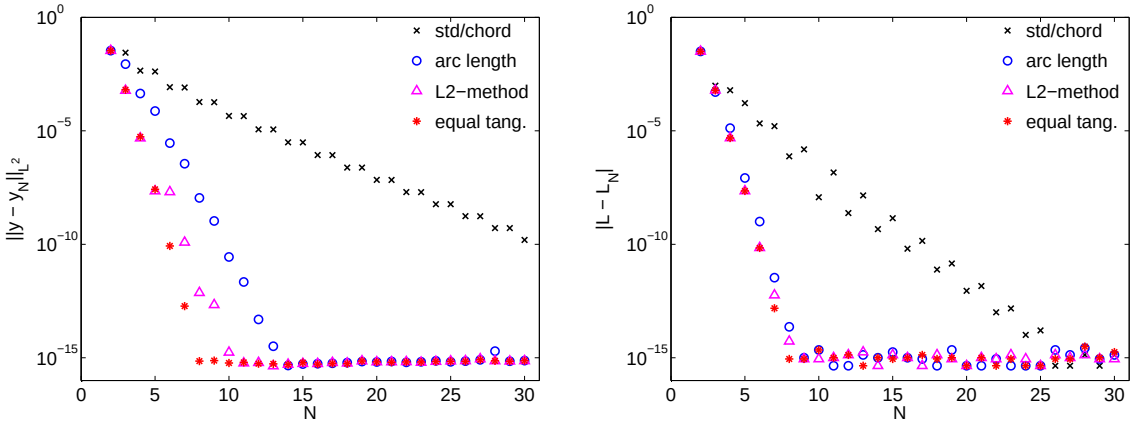
Figure 6: Interpolation points for four different methods ($N = 7$): GLL distribution along the x-axis, GLL distribution according to arc length, and distributions found iteratively by minimizing the L^2-error and by requiring equal tangent vectors.

Another interesting observation is that we are actually able to do better than the arc length method. The equal-tangent method performs visibly better than the arc length method when considering the L^2-norm of the error, as does the L^2-optimal method for small N. When measuring the error in arc length, all the methods except the standard

method converge so fast that the difference between them is hardly noticeable. Note that, even though the difference between the point distribution at $N = 7$ was hardly visible in Figure 6, the L^2-error of the equal-tangent interpolant is smaller by several orders of magnitude than the interpolant based on the arc length method.

It is important to realize that we are not assuming any particular parametrization when we use the equal-tangent method, e.g., we can use either one of the two parametrizations indicated in the beginning of this section. The equal-tangent method will automatically and implicitly construct a reparametrization of the given curve, from which a very good interpolant can be constructed.

The L^2-optimal interpolant does not live up to its name as it is not the solution with the smallest L^2-error for all N. For $N \leq 5$ it is the best method, but then it seems to "lose track" of the optimal solution. This is a reflection of the inherent problem with the L^2-optimization strategy: the functional $\mathcal{J}$ that we minimize becomes increasingly more complex as N increases, with several local minima. At a certain point, a sophisticated global minimization algorithm is needed to find the minimum, and the cost of the process may render it unfeasible. Our implementation uses a local search, and is not guaranteed to find the global minimum, at least not for large N. The use of repeated Newton iterations with different initial values ensures that the error does not evolve worse than the other distributions, and this helps the method converge.



(a) L^2-error (b) Arc length error

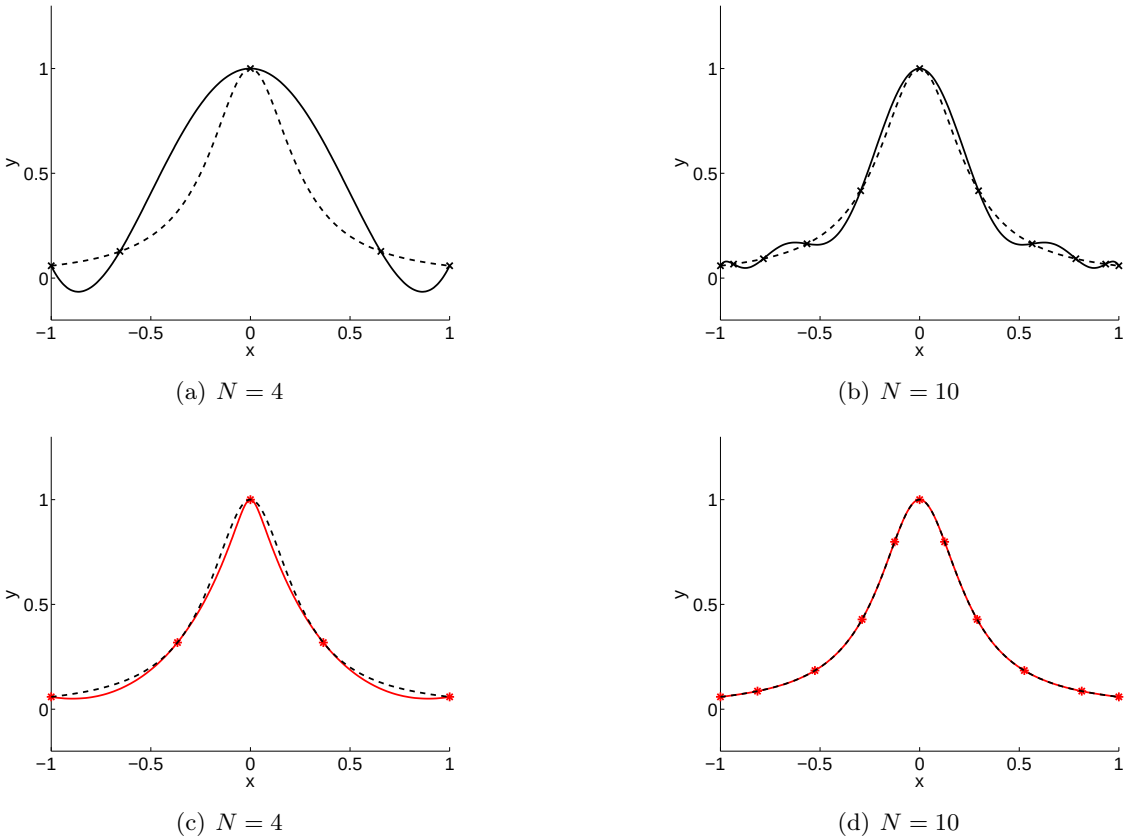
Figure 7: Interpolation error for Case 1 for the four different strategies, measured both in the L^2-norm (left plot) and in arc length (right plot). We see a huge difference between the standard method and the arc length method. Also note that we are actually able to do slightly better than the arc length method using the new strategy with equal tangents at the (reparametrized) interpolation points. The L^2-optimal method finds the best solution for $N \leq 5$, but for higher values of N the global minimizer of the L^2-error becomes too difficult to find in practice.

2.6.2 Case 2

The second example is the function famous for illustrating the Runge phenomenon [6]. The function we consider can easily be parametrized by

$$x(\xi) = \xi,$$
$$y(\xi) = \frac{1}{1 + 16\xi^2}.$$

The standard method yields a polynomial $y_N(x)$, $x \in [-1, 1]$, and a slight (and expected) oscillatory behavior can be observed in Figure 8. No oscillations are observed for the equal-tangent method.



(a) $N = 4$ (b) $N = 10$

(c) $N = 4$ (d) $N = 10$

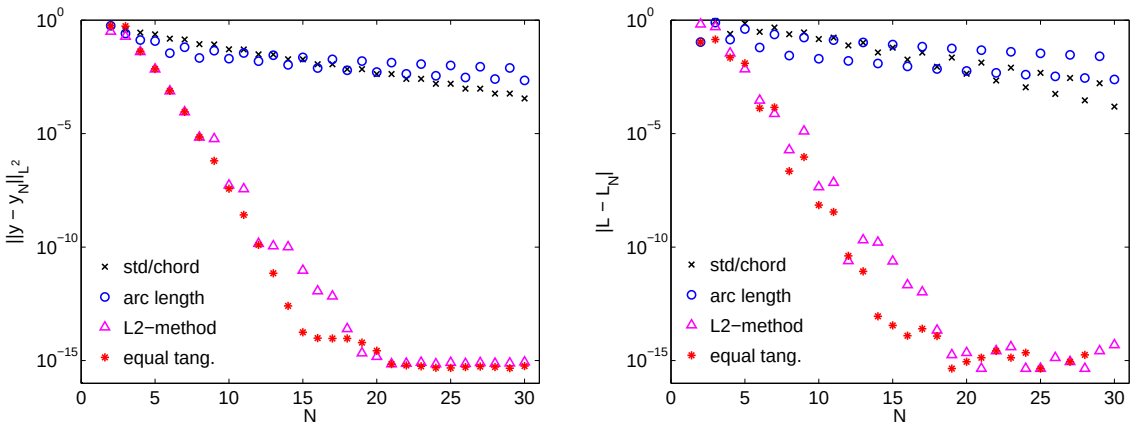
Figure 8: The exact curve (dashed) and the interpolant in Case 2 for $N = 4$ and $N = 10$, using the standard method (top) and equal-tangent method (bottom). We see that the standard method results in oscillations which decrease as N increases, while no oscillations are observed with the new method.

Figure 9 shows the error, both in the L^2-norm and in the arc length. The standard method gives exponential convergence, but with very low convergence rate. The arc length method converges even slower. There is a huge difference in performance between the various methods. The equal-tangent method converges fast from the very beginning, reaching machine precision already at $N \approx 15$. The L^2-method converges at exactly the same rate up to a certain N and then again experiences problems with finding the global minimum. The fact that the two converge with the same speed serves as an indication that the equal-tangent method yields close to the optimal solution for this case.

2.6.3 Case 3

The third example is designed to highlight the importance of viewing the exact solution as a parametrized curve instead of a function $y(x)$. The curve is parametrized by

$$x(\xi) = \xi + \frac{1}{10},$$
$$y(\xi) = \sqrt{(\xi + 21/10)^{1/3} - 1},$$



(a) L^2-error (b) Arc length error

Figure 9: Interpolation error for Case 2 measured in the discrete L^2-norm (left) and in arc length (right). The standard method performs a bit better than the arc length method for this example; however, both methods converge quite slowly. In contrast, the equal-tangent method converges very fast, and the error is at machine precision level for all $N \geq 15$.

i.e., the exact solution is given by $y = \sqrt{(x+2)^{1/3} - 1}$, where $x \in (-\frac{9}{10}, \frac{11}{10})$. We have chosen the domain carefully to avoid the singularity at $x = -1$, and hence the standard method can be expected to yield spectral convergence. The chord and arc length methods should also work well for this example. This is verified in Figure 10, where we see the interpolants at $N = 6$; the differences between the exact and the numerical curves are not visible at all.

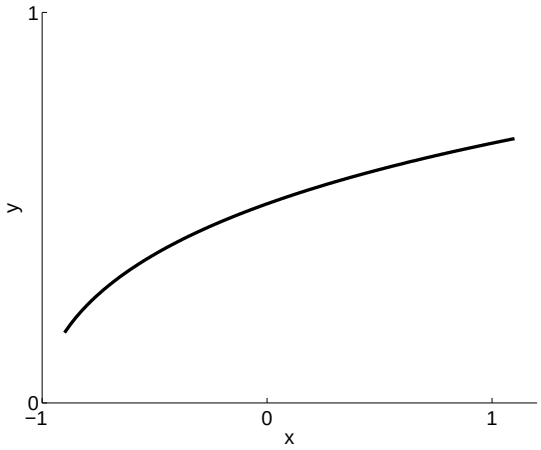
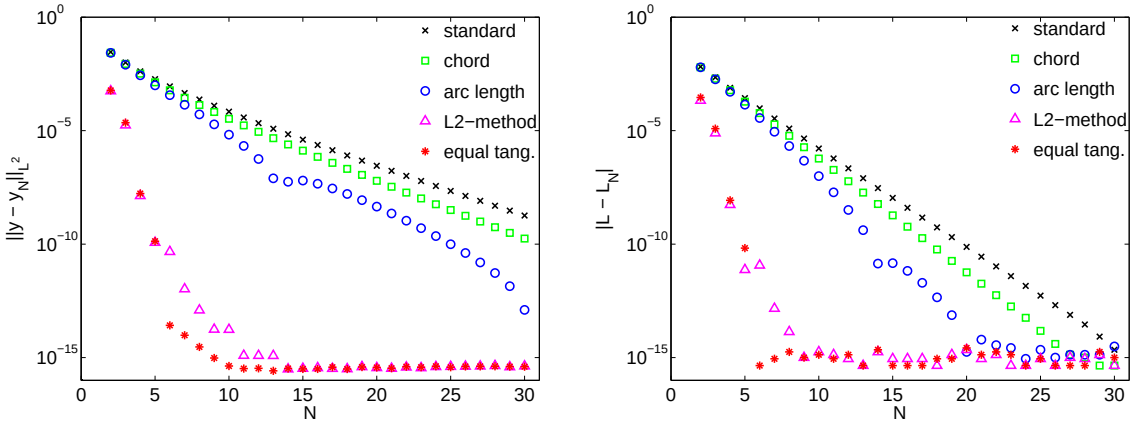


Figure 10: The function in Case 3 given by $y(x) = \sqrt{(x+2)^{1/3} - 1}$. The interpolants based on the standard method, the chord method and the arc length method are also plotted here for $N = 6$ with no visible difference.

However, none of the methods mentioned above are optimal. The inverse function $x(y)$ is actually a polynomial: $x(y) = (y^2 + 1)^3 - 2$. Hence, applying classical interpolation (the standard method) to the function $x(y)$ instead of to the function $y(x)$ should enable us to represent the curve with *no interpolation errors* for $N \geq 6$. This fact is automatically detected by our new strategy of equal tangents at the interpolation points. The new method is superior to the traditional ones right from the beginning. The equal-tangent

method also maintains the optimal point distribution for $N > 6$, although this relies on a good initial guess in the Newton iteration; see the comments in Section 2.7.2. Hence, the interpolation error is at machine precision level for all $N \geq 6$, as can be seen in Figure 11. Although the three other methods result in relatively good performance, they are far from optimal. The L^2-method again finds almost the same solution as the equal tangent method up to a certain N, where it suffers a slight deterioration in convergence rate, due to difficulties finding a global minimum.



(a) L^2-error (b) Arc length error

Figure 11: Interpolation error for Case 3 for five different strategies of choosing the point distribution. The standard method, the chord method and the arc length method yield roughly the same performance for this example. The strategy with equal tangents, however, is able to represent the curve exactly from $N = 6$, and the error is at machine precision level for $N \geq 6$.

2.6.4 Case 4

Let us now consider a function of finite (and low) regularity. In contrast to classical interpolation results [1, 3], we will demonstrate that it is possible to construct a high order interpolant with an associated interpolation error which decays exponentially fast as the order of the interpolant is increased.

The exact curve is represented by the function $y(x) = 1 - |x|^3$, $x \in [-1, 1]$, and can readily be parametrized as $x(\xi) = \xi$ and $y(\xi) = y(x(\xi))$, $\xi \in [-1, 1]$; see Figure 12. The curve looks smooth, however, the function is not analytic. The standard method (which coincides with the chord method in this case) will yield algebraic convergence.

Figure 13 depicts the distribution of the interpolation points for the standard method and the equal-tangent method when $N = 15$. The difference between the two point distributions is clearly visible: the equal-tangent method seems to cluster the interpolation points around $x = 0$, the region of low regularity of the exact function. In contrast, the standard method implies the usual clustering of the Gauss points near the two end points.

In order to better understand the implication of the different point distributions, we show in Figure 14 the functions $x_N(\xi)$ and $y_N(\xi)$. The standard method yields the affine mapping (in fact, linear mapping) $x_N(\xi) = \xi$, while $y_N(\xi)$ is the standard high order (GLL) interpolant of $y(x)$. Hence, $y_N(\xi)$ is a polynomial approximation of a function of low regularity, and we obtain the expected algebraic convergence. However, if we allow the mapping $x(\xi)$ to be nonlinear, the function $y(\xi) = y(x(\xi))$ also changes, and it may have higher regularity than the corresponding function for the standard method.

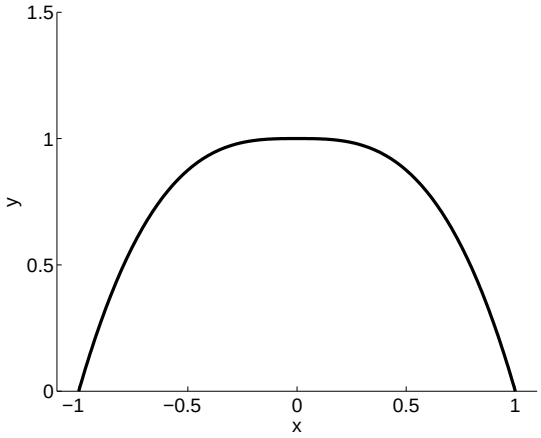


Figure 12: The curve in Case 4, given by $y(x) = 1 - |x|^3$.

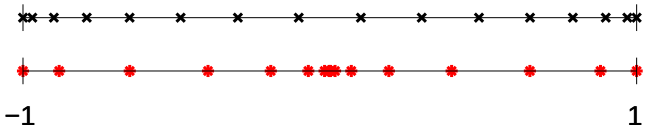
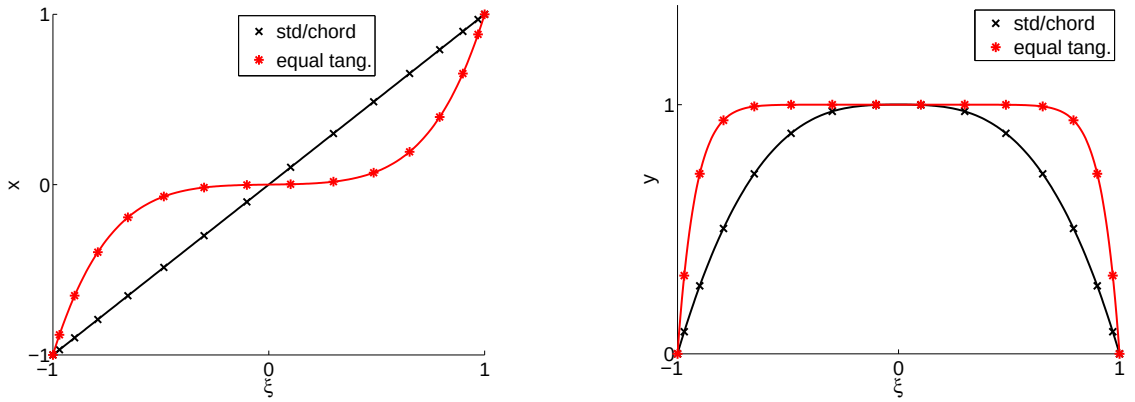


Figure 13: Interpolation points on the curve $y(x) = 1 - |x|^3$ for $N = 15$, projected down to the x-axis. Top: the point distribution for the standard method, which coincides with the GLL distribution. Bottom: point distribution for the equal-tangent method, which results in a clustering of points around $x = 0$, the region of low regularity for the exact function $y(x)$.

In Figure 15 we plot the interpolation error, measured in the L^2-norm. The equal-tangent method gives a dramatic improvement compared to the chord and arc length methods. Not only do we achieve exponential convergence; the convergence is also rapid, with machine precision level achieved at $N = 18$. The L^2-method again coincides with the equal-tangent method when N is small, but convergence now seriously degrades as N increases, resulting in a convergence rate only slightly better than the traditional methods.



(a) $x_N(\xi)$ (b) $y_N(\xi)$

Figure 14: A plot of x_N and y_N as functions of the reference coordinate ξ for the two different interpolation methods. The equal-tangent method results in a nonlinear mapping $x_N(\xi)$, something which will improve the regularity of $y_N(\xi)$ compared to the standard method.

The numerical results for this test problem suggest that there must exist a parametrization $(x(\xi), y(\xi))$ of the original function $y(x)$ such that $x(\xi)$ and $y(\xi)$ are analytic and can be (individually) well interpolated by high order polynomials $x_N(\xi)$ and $y_N(\xi)$. We do not know the analytic expressions of the optimal parametrization, however, the polynomial approximations $x_N(\xi)$ and $y_N(\xi)$ in Figure 14 for the equal-tangent method give a good indication of the optimal parametrization.

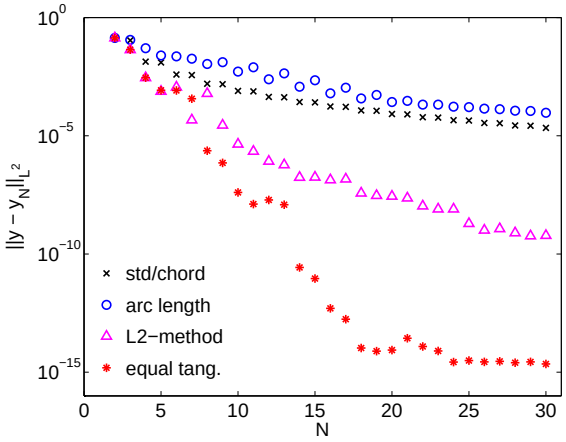


Figure 15: Interpolation error for Case 4 measured in the discrete L^2 norm. The standard method and the arc length method yield similar performance, and the interpolation error decays algebraically with N since the exact $y(x)$ function is not analytic. However, we are able to achieve exponential convergence with the equal-tangent method, since we are no longer trying to approximate a non-analytic function $y(\xi)$.

2.6.5 Case 5

The last example features a smooth, non-symmetric function $y(x) = \frac{1}{2}e^{-x}\cos(2\pi x)$ defined on $[0, 2]$; see Figure 16. The natural parametrization is

$$x(\xi) = \xi + 1,$$
$$y(\xi) = \frac{1}{2}e^{-(\xi+1)}\cos(2\pi(\xi + 1)).$$

Since this functional description of the curve involves two very regular functions, we expect the standard method to work well. Figure 17 confirms this, showing that the standard method converges to machine precision at $N \approx 25$.

A striking feature of Figure 17 is the poor performance of the chord method. The y-values at the end points of the curve do not differ very much, and hence the standard method and the chord method should not give very large differences in point distribution. Still, the chord method converges extremely slowly compared to the standard method. The reason for this is the inability of the chord distribution to resolve the change in curvature around the minimum at $x \approx 0.5$.

The equal-tangent method gives a very good convergence rate. The method converges to machine precision for a polynomial degree which is only half the polynomial degree needed for the standard method.

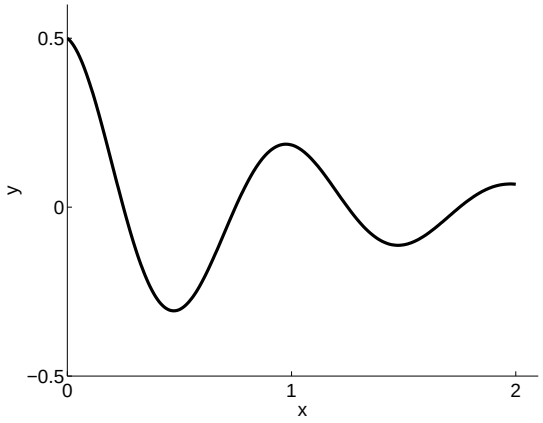


Figure 16: The curve in Case 5 given by $y(x) = \frac{1}{2}e^{-x}\cos(2\pi x)$.

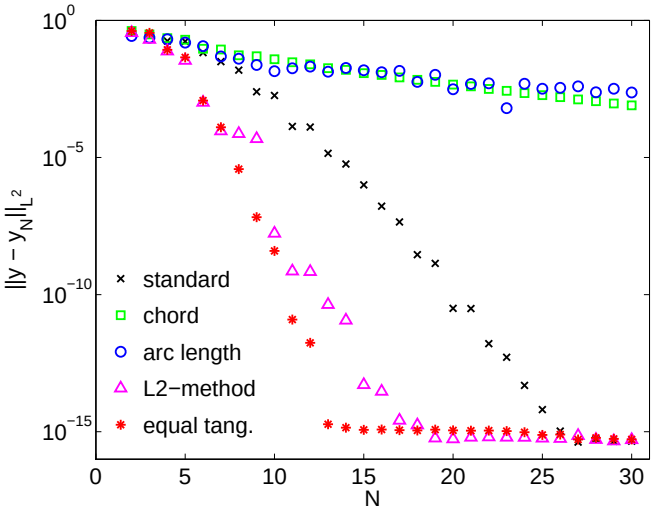


Figure 17: Interpolation error for Case 5 measured in the discrete L^2 norm. Both the chord method and the arc length method converge very slowly, while the standard method converges rapidly as expected. Still, there is a significant potential for improvement in performance when we go from the standard method to the equal-tangent method.

2.7 Remarks to the numerical results

We close this section with a few remarks related to the numerical results. These are included in order to better understand the details of the implementation, and in order to be successful in reproducing these results.

2.7.1 Error computation

We first comment on how the interpolation error (4) is computed numerically in these tests. First, the integrand needs to be transformed to an integral over the reference domain so that we can use Gauss quadrature. To this end, we use the standard affine mapping

$$x^{\mathrm{STD}}(\xi) = a + \frac{b-a}{2}(\xi + 1).$$

This mapping is convenient to use here since the Jacobian of the mapping is trivial and known explicitly. We then use GLL quadrature with M points, i.e.,

$$\begin{aligned}\|y - y_N\|_{L^2}^2 &= \int_a^b (y(x) - y_N(x))^2 \,\mathrm{d}x \\ &= \frac{b-a}{2} \int_{-1}^{1} \left(y(x^{\mathrm{STD}}(\xi)) - y_N(x^{\mathrm{STD}}(\xi)) \right)^2 \mathrm{d}\xi \\ &\approx \frac{b-a}{2} \sum_{\alpha=0}^{M} \rho_\alpha \left(y(x^{\mathrm{STD}}(\xi_\alpha)) - y_N(x^{\mathrm{STD}}(\xi_\alpha)) \right)^2,\end{aligned}$$

where ρ_α and ξ_α, $\alpha = 0, \ldots, M$, are the GLL quadrature weights and quadrature points, respectively. We have used $M = 3N$ in order to ensure that the quadrature error is always completely subdominant the interpolation error.

In the last expression for the L^2-error we only have access to $y_N(x^{\mathrm{STD}}(\xi_\alpha))$ for the standard method. For the other interpolation methods, the mapping $x_N(\xi)$ is generally non-affine. Hence, we need to use an iterative procedure to find the reference coordinate η_α such that

$$x^{\mathrm{STD}}(\xi_\alpha) = x_N(\eta_\alpha), \qquad \alpha = 0, \ldots, M.$$

Once we have found η_α, we can evaluate $y_N(x^{\mathrm{STD}}(\xi_\alpha))$ using the high order polynomial interpolant (3) as

$$y_N(x^{\mathrm{STD}}(\xi_\alpha)) = \sum_{j=0}^{N} y_j \ell_j(\eta_\alpha), \qquad \alpha = 0, \ldots, M.$$

2.7.2 Solving the nonlinear system of equations

The implementation of the L^2-method and the equal-tangent method necessitates the solution of nonlinear systems of equations. For both methods, solving these systems is challenging. In the present work, we have been using a standard Newton iteration. The main difficulty has been to construct a sufficiently good initial guess for the Newton iteration. In order to improve the robustness, we have also limited the change in the solution per iteration by introducing a relaxation parameter.

The system of nonlinear equations resulting from minimizing the L^2-error is particularly hard to solve. The Newton iteration can easily get trapped into finding a local minimum, for which the corresponding point distribution is non-optimal. The sensitivity to the initial guess seems to be very high and seems to increase as the polynomial degree N increases.

The equal-tangent method does not correspond to solving a minimization problem. Instead the method finds a point distribution which (i) interpolates the exact curve and (ii) results in equal tangent vectors at the interpolation points. However, this method is also sensitive to the initial guess. In particular, the system (6) may have multiple solutions.

The way we have implemented both the L^2-method and the equal-tangent method is by first trying to solve the problem for a low polynomial degree, and then successively increase the value of N using the solution achieved for $N - 1$ to produce an initial guess. Hence, this corresponds to a bootstrapping approach. This may not always be sufficient and we have therefore also used the point distribution corresponding to the chord method as an initial guess. Each of these initial guesses will result in a set of interpolation points corresponding to a polynomial degree N. In order to proceed to the next value $N + 1$, we

start from whichever point distribution produces the smallest L^2-error for the polynomial degree N.

Occasionally, the preset maximum number of Newton iterations is reached without reaching the preset tolerance for the Newton iteration. In such cases, we have used the solution corresponding to the smallest L^2-error over all the iterations.

3 Application to the numerical solution of PDEs

In this section we consider the numerical solution of the two-dimensional Poisson problem in a deformed quadrilateral domain Ω. In order to avoid any errors due to the representation of the right hand side, we limit our attention to the following simple Laplace problem:

$$\begin{aligned} \nabla^2 u &= 0 && \text{in } \Omega, \\ u &= e^x \sin y && \text{on } \partial\Omega. \end{aligned} \tag{8}$$

This problem has the exact solution $u(x, y) = e^x \sin y$ which is analytic.

We discretize the Laplace problem (8) based on the equivalent weak form. To this end, we use a pure spectral discretization based on high order polynomials [10, 1, 5, 4]. The difference between the exact solution u and the numerical solution u_N is measured in the energy norm,

$$|||u - u_N|||^2 = \int_\Omega \nabla(u - u_N) \cdot \nabla(u - u_N) \, \mathrm{d}A. \tag{9}$$

As usual, the integral on the right hand side is transformed to an integral over the associated reference domain $\hat{\Omega} = (-1, 1)^2$, and exact integration is replaced by Gauss-Lobatto Legendre (GLL) quadrature.

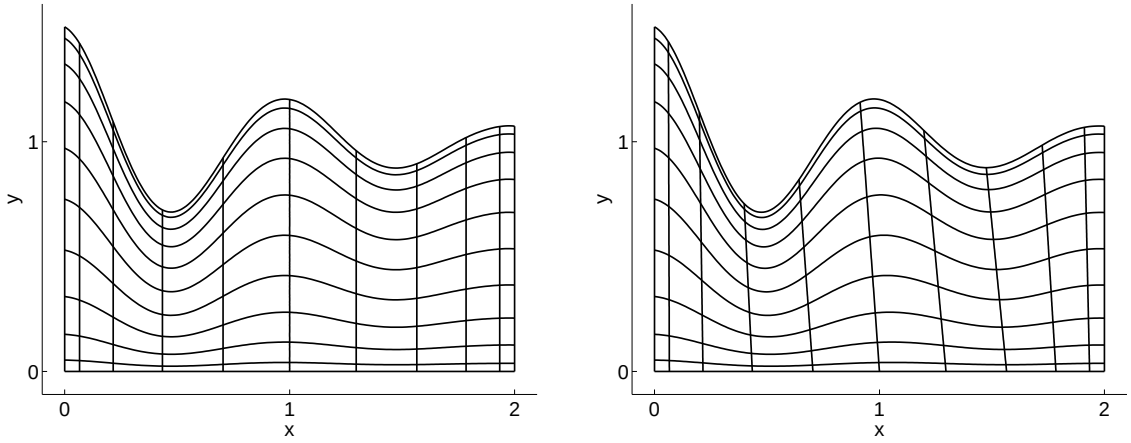
The mapping between $\hat{\Omega}$ and Ω is constructed based on transfinite interpolation [7], also referred to as the Gordon-Hall algorithm; this is a very common way to construct this mapping. In the following, we consider an isoparametric mapping, i.e., the geometry is also approximated using high order polynomials of degree N. In order to use the Gordon-Hall algorithm for this case, we need to have available a high order approximation of all the four sides of Ω.

In the numerical tests below, we will approximate the boundary of Ω using the various high order interpolation methods discussed in Section 2. In particular, we will compare the methods based on two different error measures: (i) the error $|A - A_N|$ where A is the exact area of Ω and A_N is the area based on an isoparametric representation of Ω; and (ii) the discretization error (9). The former measures only the quality of the geometry approximation, whereas the latter also takes into account the suitability of the numerical grid for a spectral discretization of (8). In order to eliminate quadrature errors in the error computations, we will use overintegration based on a polynomial degree $3N$.

3.1 Case 1

The first domain consists of three straight edges and one deformed edge; see Figure 18. The top edge is the same curve as in Case 5 in the previous section, shifted upwards and given by $y(x) = 1 + \frac{1}{2}e^{-x}\cos(2\pi x)$. This curve was very well approximated using the equal-tangent method; see Figure 17. We apply the same methods as in Section 2 for distributing the points along each edge of the boundary, and then use the Gordon-Hall algorithm for interpolating to the interior.

Figure 19 shows the difference between the exact area and the area of the computational



(a) The standard method

(b) The equal-tangent method

Figure 18: Computational grid for Case 1 using a polynomial degree $N = 10$. The left grid is generated using the standard method for distribution of interpolation points along the four edges. The right grid is constructed using the equal-tangent method along the top edge. Transfinite interpolation (the Gordon-Hall algorithm) is used to compute the internal grid points in both grids.

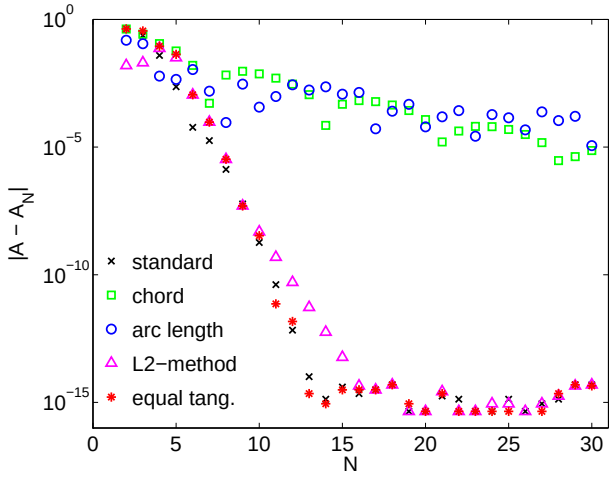


Figure 19: The difference between the area of the exact domain and the computational domain in Case 1 as a function of the polynomial degree N. Note that the area is computed with negligible numerical errors.

domain for the different interpolation methods, or equivalently, for the different point distributions. All methods result in the standard GLL point distribution along the straight edges; only the point distribution along the top edge differs between the different strategies. The chord method and the arc length method show the same poor performance as they did in the one-dimensional case; these results can be attributed to the problem approximating the top edge for these methods. The standard method, the equal-tangent method and the L^2-method are all orders of magnitude better starting from $N = 6$. However, there is no longer a significant difference between these three methods.

In Figure 20 we also show the corresponding discretization error (9). We observe that the relative performance of the various interpolation methods is similar to the results obtained

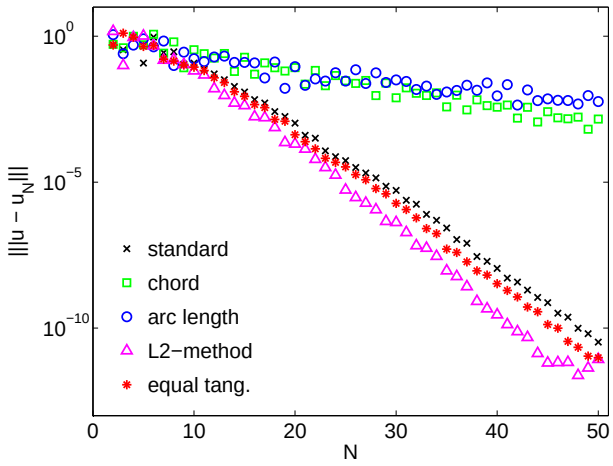
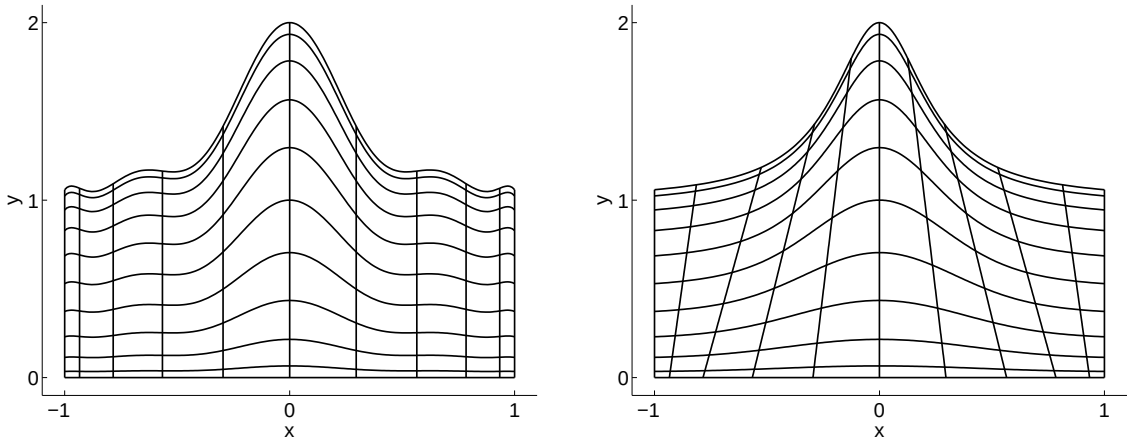


Figure 20: The discretization error (9) when solving the Laplace problem (8) over the domain in Case 1, measured in the energy norm. Even though the magnitude of the error is larger than for the computed area of the domain, the different interpolation methods show the same relative performance.

for the error in the area of the domain. The standard method seems to give close to optimal results for this problem.

3.2 Case 2

Again, we consider a quadrilateral domain with three straight edges and a single deformed edge. In this case, the top edge is the Runge function considered in the previous section. The numerical results for Case 2 in Section 2 showed that this function is poorly approximated using the standard method and the arc length method. However, the numerical results also showed that the equal-tangent method does an excellent job, converging fast and reaching machine precision level at $N = 15$.



(a) The standard method

(b) The equal-tangent method

Figure 21: Computational grid for Case 2 using a polynomial degree $N = 10$.

Figure 21 shows the computational grids using the standard method and the equal-tangent method. For $N = 10$ we can still see the ripples along the top edge when using

the standard method; these ripples represent a typical feature of a standard polynomial approximation of the Runge function. In contrast, the top edge of the domain looks very smooth using the equal-tangent method.

Figure 22 shows the error in area of the domain, indicating the same convergence behavior as we observed for the interpolation error of the pure Runge function.

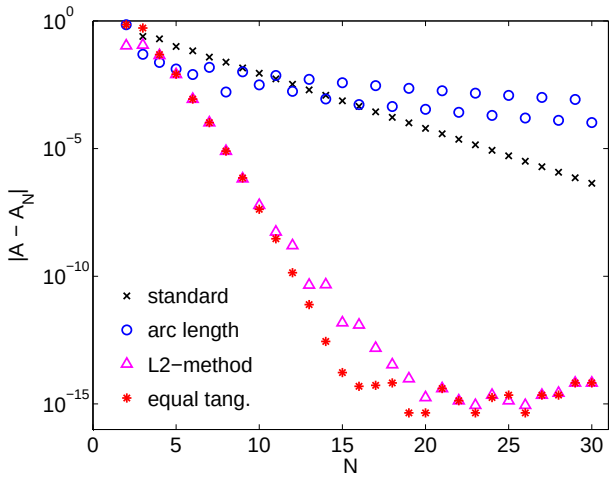


Figure 22: The difference between exact area and the area of the computational domain in Case 2. The quality of the representation of the top edge seems to determine the convergence rate.

Figure 23 shows the discretization error (9) when solving the Laplace problem in this domain. Even though the grid points using the equal-tangent method are slanted towards the middle of the domain, the representation of the field variables does not degrade. The discretization error converges similarly as the error in the area, with the equal-tangent method and the L^2-method outperforming the other methods.

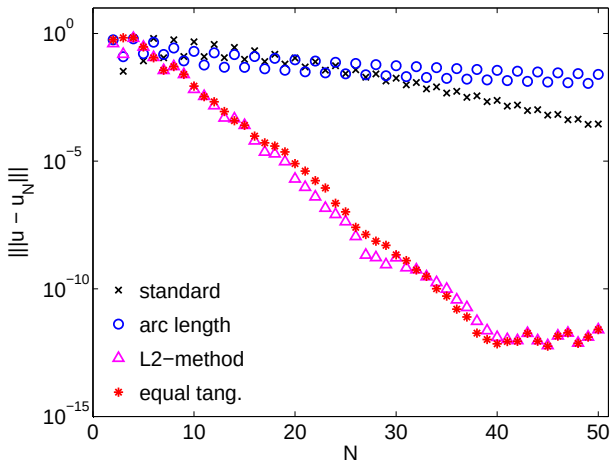
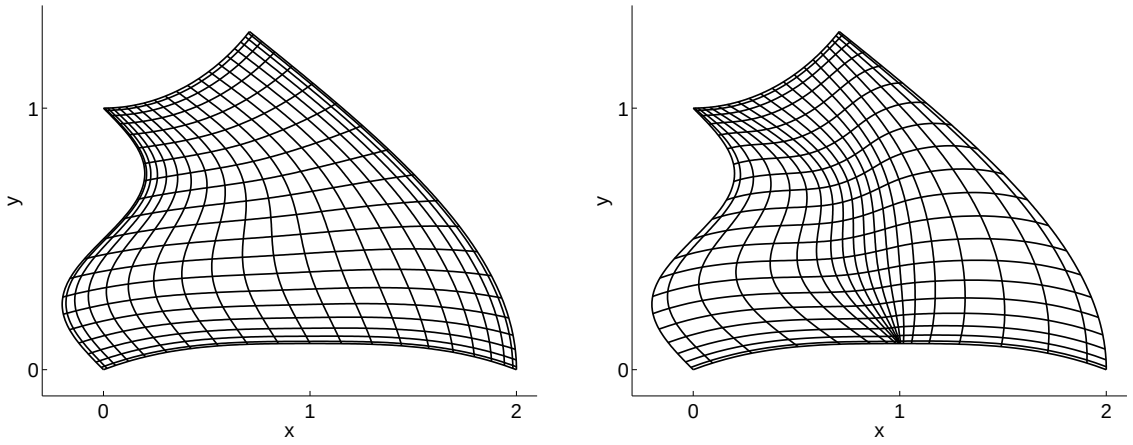


Figure 23: The discretization error in the solution of the Laplace problem (8) over the domain in Case 2, measured in the energy norm. The standard method and the arc length method do not represent the top edge well, and the convergence rate is correspondingly slow. The difference between the L^2-method and the equal-tangent method is not as large as it was when we were only interpolating the Runge function (see Figure 9); however, notice the big difference in the absolute error level here compared to Figure 9 for the same value of N.

3.3 Case 3

We now consider a domain where all the four edges are deformed; see Figure 24. The top edge is part of a circle of radius one, the left edge and the right edge are trigonometric functions of y, and the bottom edge is a scaled version of the non-analytic curve from Case 4 in the previous section. The finite regularity of the bottom edge will prevent the chord method and the arc length method from representing the computational domain in the best possible way.



(a) The chord method / arc length method

(b) The equal-tangent method

Figure 24: Computational grids in Case 3, using a polynomial degree $N = 20$.

The numerical results for Case 4 in Section 2 showed that the equal-tangent method was able to represent the bottom curve surprisingly well. Figure 24 shows the computational grid using two different methods for interpolating the boundary of the domain. The left grid is usually considered as an "optimal" grid, with the bottom edge and the left and the right edges approximated using the chord method, and the top edge approximated using the arc length method. The right grid is fully based on using the equal-tangent method for the approximation of the domain boundary. The most notable feature is the clustering of grid points towards the center of the bottom curve.

Figure 25 shows the same behavior as we observed for Case 4 in the one-dimensional case. Using the equal-tangent method, we are able to represent the bottom edge so well that the error in the area of the computational domain converges exponentially fast.

In Figure 26 we compare the discretization error as a function of N for the various methods. We see that the convergence behavior is a bit different from the convergence behavior in Figure 25; the equal-tangent method is not optimal for small N. This is because the clustering of grid points along the bottom edge strongly affects the mapping from the two-dimensional reference domain to the physical domain. This means that even if the exact solution is very regular on the physical domain, we may be facing steep gradients and boundary layers on the reference domain. Hence, there is a trade-off between a good representation of the geometry and a smooth mapping from the reference domain to the physical domain.

In this particular case, the good geometry representation starts to pay off for larger N; in fact, it is the only geometry representation that results in spectral convergence for this problem.

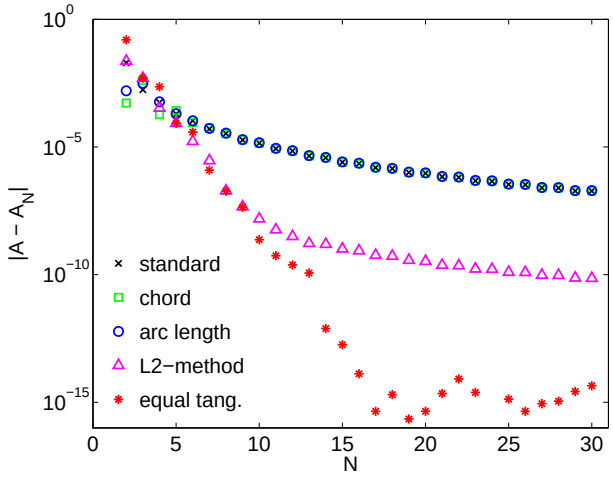


Figure 25: Difference between exact area and the area of the numerical domain in Case 3. The non-analytic bottom edge causes algebraic convergence for both the standard method, the chord method and the arc length method. The equal-tangent method, however, manages to represent this curve properly, and consequently achieves exponential convergence. This example also highlights the difference in robustness between the equal-tangent method and the L^2 method, as the latter only achieves algebraic convergence.

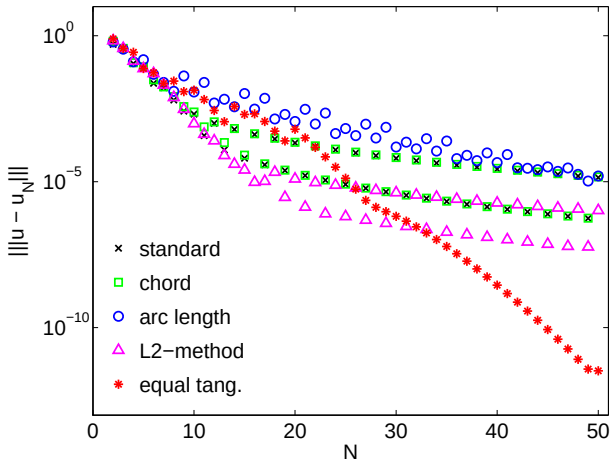


Figure 26: The error in the solution of the Laplace problem (8) in the geometry of Case 3, measured in the energy norm. The algebraic convergence of the standard method, the chord method and the arc length method is due to their inability to represent the bottom edge (a function of low regularity) accurately. The equal-tangent method is able to represent the geometry to spectral convergence, at the cost of a heavily distorted tensor-product GLL grid. This is not ideal for representing the field variables in this case, and the method is clearly not optimal for small N. However, for large N the error in the geometry representation becomes dominant, and at a certain point the equal-tangent becomes the best, being the only one to achieve exponential convergence.

4 Conclusions

In this paper we have compared different ways of constructing high order interpolants of a curve in the plane. We have compared standard methods with a new method which can also be regarded as a new way of constructing high order polynomial interpolants of a general function $y(x)$. The new method is proposed as an alternative to computing an L^2-optimal interpolant since the latter is very hard to find in practice.

The numerical results show that the new method can give significantly faster convergence compared to more standard methods. The most extreme case is exponential convergence obtained for a function $y(x)$ with a finite regularity when the polynomial degree N is increased.

The numerical results also show that the new method give consistently good results. This is in contrast to commonly used methods which seem to give small interpolation errors for some curves, but not for others.

In this study we have also investigated the impact of the geometry representation in the context of solving partial differential equations. We have shown that the overall discretization error for the Poisson problem can sometimes be significantly reduced by a more accurate representation of the boundary of the domain (in our case, the four edges of a deformed quadrilateral domain).

The new method involves solving a nonlinear system of equations. This is a nontrivial issue where the availability of a good initial guess plays an important role. However, assuming that we can solve this problem, the potential payoff may be considerable in the context of numerical solution of partial differential equations using high order methods.

Future work will focus on constructing high order interpolants of curves in three-dimensions, the representation of curved surfaces in three dimensions, and solving partial differential equations defined in three-dimensional domains. However, the approximation of curves in the plane should be investigated further, both with respect to finding an even more robust way of computing the optimal interpolant, but also with respect to a theoretical analysis.

Acknowledgment

The work has been supported by the Research Council of Norway under contract 185336/V30. The support is gratefully acknowledged.

References

[1] C. Bernardi and Y. Maday. *Spectral methods.* Handbook of Numerical Analysis. Volume V: Techniques of Scientific Computing (Part 2), edited by P.G. Ciarlet and J.L. Lions. Elsevier, 1997.

[2] J.-P. Berrut and L. N. Trefethen. Barycentric Lagrange interpolation. *SIAM Review*, 45(3):501–517, 2004.

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

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

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

[6] B. Fornberg. *A Practical Guide to Pseudospectral Methods.* Cambridge, 1998.

[7] 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.

[8] J. Hesthaven and T. Warburton. *Nodal Discontinuous Galerkin Methods.* Springer, 2008.

[9] G. Karniadakis and S. Sherwin. *Spectral/hp Element Methods for CFD.* Oxford, 2005.

[10] Y. Maday and A.T. Patera. Spectral element methods for the Navier-Stokes equations. *in: A.K. Noor, J. T. Oden (Eds.), State of the Art Surveys in Computational Mechanics, ASME, New York*, pages 71–143, 1989.

[11] B. O'Neill. *Elementary Differential Geometry.* Academic Press, 2006.

[12] A. T. Patera Y. Maday, N. C. Nguyen and G. S. H. George. A general multipurpose interpolation procedure: the magic points. *Commun. Pure Appl. Anal.*, 8(1):383–404, 2009.