Gaussian quadrature rules for C^1 quintic splines with uniform knot vectors

Michael Bartoň[a,*], Rachid Ait-Haddou[b], Victor Manuel Calo[c]

[a]*BCAM – Basque Center for Applied Mathematics, Alameda de Mazarredo 14, 48009 Bilbao, Basque Country, Spain*
[b]*A-803 Mihogaoka 19, Ibaraki City 567-0047 Osaka, Japan*
[c]*CSIRO Professorial Chair in Computational Geoscience, Western Australian School of Mines, Faculty of Science and Engineering, Curtin University, Kent Street, Bentley, Perth, Western Australia, 6102, Australia*

Abstract

We provide explicit quadrature rules for spaces of C^1 quintic splines with uniform knot sequences over finite domains. The quadrature nodes and weights are derived via an explicit recursion that avoids numerical solvers. Each rule is optimal, that is, requires the minimal number of nodes, for a given function space. For each of n subintervals, generically, only two nodes are required which reduces the evaluation cost by 2/3 when compared to the classical Gaussian quadrature for polynomials over each knot span. Numerical experiments show fast convergence, as n grows, to the "two-third" quadrature rule of Hughes et al. [23] for infinite domains.

Keywords: Gaussian quadrature, quintic splines, Peano kernel, B-splines, C^1 continuity, quadrature for isogeometric analysis

1. Introduction

Numerical quadrature has been of interest for decades due to its wide applicability in many fields spanning collocation methods [37], integral equations [4], finite elements methods [38] and most recently, isogeometric analysis [13]. Numerical quadrature is an important tool for high-speed solution frameworks [12, 16] as it is computationally cheap and robust when compared to analytic integration methods [20].

A quadrature rule, or in short a quadrature, is said to be an *m-point rule*, if m evaluations of a function f are needed to approximate its weighted integral

---

*Corresponding author

Email addresses: `Michael.Barton@kaust.edu.sa` (Michael Bartoň), `rachid.aithaddou70@gmail.com` (Rachid Ait-Haddou), `Victor.Calo@Curtin.edu.au` (Victor Manuel Calo)

over an interval $[a, b]$

$$\int_a^b \omega(x)f(x)\,\mathrm{d}x = \sum_{i=1}^m \omega_i f(\tau_i) + R_m(f), \qquad (1)$$

where ω is a fixed non-negative *weight function* defined over $[a, b]$. Typically, the rule is required to be *exact*, that is, $R_m(f) \equiv 0$ for each element of a predefined linear function space $\mathcal{L}$. Moreover, the rule is said to be *optimal* if m is the *minimal* number of *nodes* τ_i at which f has to be evaluated.

In the literature, the term *optimality* may also refer to the approximation error that the quadrature rule produces. That is, the number of nodes is given and their layout is sought such that they minimize the error for a given class of functions. Köhler and Nikolov [25, 26] showed that the Gauss-type quadrature formulae associated with spaces of spline functions with equidistant knots are asymptotically optimal in non-periodic Sobolev classes. This is a motivation for studying Gauss-type quadrature formulae for spaces of spline functions, in particular, with equidistant knots. In this paper, by *optimal* (or Gaussian) we exclusively mean quadrature rules with the minimal number of nodes.

In the case when $\mathcal{L}$ is the linear space of polynomials of degree at most $2m - 1$, then the m-point Gaussian quadrature rule [20] is both exact and optimal. The Gaussian nodes are the roots of the orthogonal polynomials p_m where $(p_0, p_1, \ldots, p_m, \ldots)$ is the sequence of orthogonal polynomials with respect to the measure $\mu(x) = \omega(x)dx$. Typically, the nodes of the Gaussian quadrature rule have to be computed numerically which becomes expensive, especially for high-degree polynomials.

Regarding the quadrature rules for splines, Micchelli and Pinkus [30] derived the optimal number of quadrature nodes for a given knot sequence and specified the range of intervals, the knot sequence subintervals, that contain at least one node. There are two main difficulties compared to the polynomial case: firstly, the optimal quadrature rule is not in general guaranteed to be unique, e.g., when the boundary constraints are involved, and, secondly, [30] determines only a *range* of intervals, i.e., each node has several potential subintervals to lie in. The latter issue is crucial as one cannot apply even expensive numerical solvers, because the *algebraic* system to solve is not explicitly known. For each assumed layout of nodes, one would have to solve a particular algebraic system using e.g., [3, 18, 36]. However, the number of eventual systems grows exponentially in the number of subintervals and therefore such an approach is not feasible. Instead, theorems that derive exact layouts of the nodes are essential. Our work in this paper contributes such a theoretical result for a particular family of splines, namely C^1 quintics with uniform knot sequences.

The quadrature rules for splines differ depending on the particular space of interest $S_{d,c}$, where d is the *degree* and c refers to *continuity*. For cases with lower continuity, the "interaction" between polynomial pieces is lower and hence, a higher number of nodes is required for the optimal quadrature rule. The choice of the domain over which to define the quadrature can bring a significant simplifications. Whilst there are few rules that are exact and optimal over a

finite domain, their counterparts are known when the integration domain is the whole *real line* [23]. These *half-point rules* of Hughes et al. are independent of the polynomial degree and the "half" indicates that the number of quadrature points is roughly half the number of basis functions.

The half-point rules can be altered even for spaces with lower continuity, e.g., for $S_{4,1}$ an optimal rule was also derived in [23]. The rule is called a "two-third rule" as it requires only two evaluations per subinterval whilst the classical Gaussian rule for polynomials needs three nodes. However, these rules are exact only over the real line. Because in most applications a *finite domain* is needed, additional nodes have to be added to satisfy the boundary constraints and a numerical solver has to be employed [5]. We focus only on optimal rules as there are many schemes that introduce redundant nodes in order to overcome the problem with a finite interval, see [5] and the references therein. The optimal rules we introduce are unique, and require the minimum number of nodes for $S_{5,1}$.

Regarding optimal rules over finite domains, Nikolov [32] proved the unique layout of nodes of the quadrature rule for $S_{3,1}$ with uniform knot sequences and derived a recursive algorithm that computes the nodes and weights in a closed form. Recently [2], we generalized this result for $S_{3,1}$ over a special class of *non-uniform* knot sequences, called symmetrically-stretched. The rules possess the three fundamental properties to make these rules useful in practice. These are, the rules are exact, optimal, and yield a closed form algorithm, without intervention of any numerical solver.

Finding Gaussian rules for splines in a closed form as was done in [32] is rather a rare result. To compute Gaussian rules numerically, we have recently used the fact that a Gaussian quadrature is a solution (root) of a well-constrained piece-wise polynomial system of equations. Since the problem is highly non-linear, numerical optimization typically fails to find the global minimizer from a mediocre initial guess. We showed that the Gaussian quadrature rules from close-enough spline spaces offer good initial guesses for the optimization stage to succeed [7]. Starting with a known Gaussian rule, e.g., a union of classical polynomial Gaussian rules, the underlying spline space is continuously transformed to the desired configuration and the root is numerically traced. Using this *homotopy continuation* concept, we derived numerically Gaussian rules for spline spaces of various degrees and continuities [6, 8].

From the point of view of applications, certain types of quadratures (in terms of the relation between degrees and continuities of the underlying spline spaces) are very important in isogeometric analysis as these spaces appear in Galerkin discretizations when building the mass and stiffness matrices, see [6, 8]. The quadrature rule for the spline spaces considered in this work, $S_{5,1}$, can be used, e.g., to exactly integrate the products in the stiffness matrix when the original splines are C^2 cubics ($S_{3,2}$). The rule is exact, however, it is sub-optimal as the products belong to $S_{4,1}$. Nevertheless, this rule offers optimal convergence for tensor product spaces for second order partial differential equations [38]. Another relevant application where $S_{5,1}$ appears is when computing areas of planar curved domains. When these domains are parametrized by C^2 cubics

(splines used in geometric modeling the most frequently [19]), the unit normals appearing in the area formula belong exactly to this space.

In the context of isogeometric analysis, quadrature rules for splines are important tools [1, 21, 22, 24, 28, 34] because they are cheap and elegant alternatives to symbolic integration [20]. Recently, alternative methods of building mass and stiffness matrices have been proposed [27, 28]. They exploit the observation that, under certain conditions, the optimal convergence rate of the linear system can be achieved despite the fact that the integration rule is not exact. In this work, however, we focus on quadrature rules that are *exact*, that is, the rules reproduce the integrals under affine mappings exactly up to machine precision. Another recent alternative for efficient mass and stiffness matrix assembly is a weighted quadrature that is generated for each row of the mass/stiffness matrix separately [10].

The computation of the nodes and weights of an Gaussian spline quadrature, is rather challenging as one has to, first, derive the *correct layout* of the nodes and then, typically, to solve non-linear systems of algebraic equations. For polynomials of degree higher than three, the use of a numerical solver seems unavoidable. Nevertheless, in this work we derive an optimal, explicit (recursive) quadrature rule, despite the fact that we deal with degree five polynomials. We prove that there exists an algebraic factorization in every recursion step which makes the rule explicit. This is a rather surprising result because the degree five basis functions yield in our case a curve-curve intersection problem consisting of two implicit bivariate cubics. Such a scenario might in general have nine real intersection points. However, we prove that the resultant is of degree five and only its quadratic component contributes to the real solutions. Consequently, our rule is explicit and therefore no numerical solver is needed. We also show numerically, when the number of subintervals grows, that the rule rapidly converges to the "two-third" rule of Hughes [23].

The rest of the paper is organized as follows. In Section 2, we recall some basic properties of $S_{5,1}$ and derive their Gaussian quadrature rules. In Section 3, the error estimates are given and Section 4 shows some numerical experiments that validate the theory proposed in this work. Finally, possible extensions of our method are discussed in Section 5.

2. Gaussian quadrature formulae for C^1 quintic splines

In this section we state a few basic properties of $S_{5,1}$ splines and derive explicit formulae for computing quadrature nodes and weights for spline spaces with uniform knot sequences over a finite domain. Throughout the paper, π_d denotes the linear space of polynomials of degree at most d and $[a, b]$ is a non-trivial real *compact* interval.

2.1. C^1 quintic splines with uniform knot sequences

We detail several properties of spline basis functions. We consider a uniform partition $\mathcal{X}_n = (a = x_0, x_1, ..., x_{n-1}, x_n = b)$ of the interval $[a, b]$ with n

subintervals and define $h := \frac{1}{n} = x_k - x_{k-1}$ for all $k = 1, \ldots, n$. We denote by $S_{5,1}^n$ the linear space of C^1 quintic splines over a uniform knot sequence $\mathcal{X}_n = (a = x_0, x_1, \ldots, x_n = b)$

$$S_{5,1}^n = \{f \in C^1[a,b] : f|_{(x_{k-1},x_k)} \in \pi_5, k = 1, \ldots, n\}. \tag{2}$$

The dimension of the space $S_{5,1}^n$ is $4n + 2$.

Remark 1. *In the B-spline literature [11, 17, 19], the knot sequence is usually written with knots' multiplicities. However, in the isogeometric analysis literature, see e.g., [9, 33], the knot vector is usually split into a vector carrying the partition of the interval and a vector containing continuity information (knot multiplicity). As in this paper the multiplicity is always four at every knot, we follow the latter notation and, throughout the paper, write $\mathcal{X}_n$ without multiplicity, i.e., $x_k < x_{k+1}$, $k = 0, \ldots, n-1$.*

Similarly to [2, 32], we find it convenient to work with the non-normalized B-spline basis. To define the basis, we extend our knot sequence $\mathcal{X}_n$ with two extra knots outside the interval $[a, b]$ in a uniform fashion

$$x_{-1} = x_0 - h \quad \text{and} \quad x_{n+1} = x_n + h. \tag{3}$$

The choice of x_{-1} and x_{n+1} allows us to simplify expressions in Section 2.2, but this setting does not affect the quadrature rule derived later in Theorem 2.1. We follow [15] and denote by $\mathbf{D} = \{D_i\}_{i=1}^{4n+2}$ the basis of $S_{5,1}^n$ where

$$\begin{aligned}
D_{4k-3}(t) &= [x_{k-2}, x_{k-2}, x_{k-1}, x_{k-1}, x_{k-1}, x_{k-1}, x_k](. - t)_+^5, \\
D_{4k-2}(t) &= [x_{k-2}, x_{k-1}, x_{k-1}, x_{k-1}, x_{k-1}, x_k, x_k](. - t)_+^5, \\
D_{4k-1}(t) &= [x_{k-1}, x_{k-1}, x_{k-1}, x_{k-1}, x_k, x_k, x_k](. - t)_+^5, \\
D_{4k}(t) &= [x_{k-1}, x_{k-1}, x_{k-1}, x_k, x_k, x_k, x_k](. - t)_+^5.
\end{aligned} \tag{4}$$

where $[.]f$ stands for the divided difference and $u_+ = \max(u, 0)$ is the truncated power function. The direct computation of the divided differences gives the following explicit expressions for $t \in [x_{k-2}, x_{k-1}]$

$$\begin{aligned}
D_{4k-3}(t) &= \frac{(t-x_{k-2})^4(x_k+8x_{k-1}-9t)}{4h^6}, \\
D_{4k-2}(t) &= \frac{(t-x_{k-2})^5}{4h^6},
\end{aligned} \tag{5}$$

and for $t \in [x_{k-1}, x_k]$

$$\begin{aligned}
D_{4k-3}(t) &= \frac{(x_k-t)^5}{4h^6}, \\
D_{4k-2}(t) &= \frac{(x_k-t)^4(x_{k-2}+8x_{k-1}-9t)}{4h^6}, \\
D_{4k-1}(t) &= \frac{10(t-x_{k-1})^2(x_k-t)^3}{h^6}, \\
D_{4k}(t) &= \frac{10(t-x_{k-1})^3(x_k-t)^2}{h^6}.
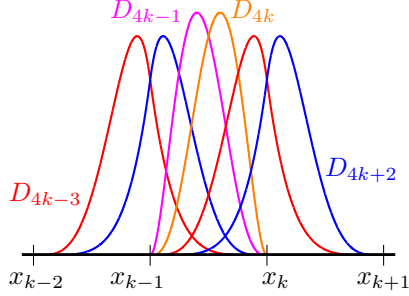\end{aligned} \tag{6}$$

5

Figure 1: Four consecutive knots $x_{k-2}, \ldots, x_{k+1}$ of a uniform knot sequence, each of multiplicity four. Six non-normalized spline basis functions $D_{4k-3}, \ldots, D_{4k+2}$ with non-zero support on $[x_{k-1}, x_k]$ are displayed.

The functions have the following pattern: six basis functions $D_{4k-3}, \ldots, D_{4k+2}$ have non-zero support on $[x_{k-1}, x_k]$, moreover, two of them, D_{4k-1} and D_{4k}, act only $[x_{k-1}, x_k]$ and are scaled Bernstein basis functions, see (6) and Fig. 1.

Among the basic properties of the basis $\mathbf{D}$, we need to recall the fact that $D_{4k+2}(t) \leq D_{4k+1}(t)$ for $t \in [x_{k-1}, x_k]$ for $k = 1, \ldots, n$ and are equal at the knot x_k, that is, $D_{4k+1}(x_k) = D_{4k+2}(x_k) = \frac{1}{4h}$. Moreover, we have that $D_{4k-1}(\frac{x_{k-1}+x_k}{2}) = D_{4k}(\frac{x_{k-1}+x_k}{2}) = \frac{5}{16h}$. From (5) and (6), the integrals of the basis functions are computed

$$I[D_k] = \frac{1}{6} \quad \text{for } k = 3, 4, \ldots, 4n, \tag{7}$$

where $I[f]$ stands for the integral of f over the interval $[a, b]$. The first and the last two integrals are

$$I[D_1] = I[D_{4n+2}] = \frac{1}{24} \quad \text{and} \quad I[D_2] = I[D_{4n+1}] = \frac{1}{8}. \tag{8}$$

A less obvious property that binds together four consecutive basis functions, which is used later for our quadrature rule in Section 2.2, is formalized as follows.

Lemma 2.1. *Let $\mathcal{X}_n = (a = x_0, x_1, \ldots, x_n = b)$, be a uniform knot sequence and for any $k = 1, \ldots, n$ define*

$$P_k(t) = 2D_{4k+1}(t) - 2D_{4k+2}(t) + \frac{1}{2}D_{4k-1}(t) - D_{4k}(t). \tag{9}$$

Then $P_k(t) \geq 0$ for any $t \in (x_{k-1}, x_k)$ and $P_k(t) = 0$ if and only if $t = \frac{x_{k-1}+x_k}{2}$.

Proof. Over an interval (x_{k-1}, x_k), the function P_k is a single quintic polynomial. Therefore, it can be expressed in terms of Bernstein basis and can be viewed as a Bézier curve on a particular domain, see Fig. 2. Looking at its
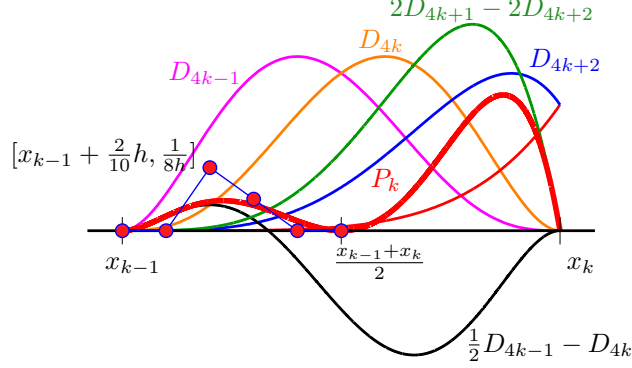
6

Figure 2: The linear blend of basis functions (9), P_k, is expressed as a Bézier curve on $[x_{k-1}, \frac{x_{k-1}+x_k}{2}]$ with a control point sequence (red dots) with non-negative y-coordinates (11). Consequently, P_k is non-negative on $[x_{k-1}, x_k]$ and has a single root (of multiplicity two) at $\frac{x_{k-1}+x_k}{2}$.

shape, one cannot conclude non-negativity from the control polygon, when considered on the whole interval (x_{k-1}, x_k). Hence we define

$$\begin{aligned} P_k^1(t) &= P_k(t) \quad \text{on} \; [x_{k-1}, \tfrac{x_{k-1}+x_k}{2}], \\ P_k^2(t) &= P_k(t) \quad \text{on} \; [\tfrac{x_{k-1}+x_k}{2}, x_k], \end{aligned} \tag{10}$$

and using $h = x_k - x_{k-1}$ we further write

$$P_k^1(t) = \sum_{i=0}^{5} q_i^1 B_i^5(t), \; \text{where} \; B_i^5(t) = \binom{5}{i} \left(\frac{2t - 2x_{k-1}}{h} \right)^i \left(\frac{2x_{k-1} + h - 2t}{h} \right)^{5-i}$$

and analogously for P_k^2. The conversion from monomial to Bernstein basis gives the control points $(p_0^1, \dots, p_5^1)$ of P_k^1 over the interval $[x_{k-1}, x_{k-1} + h/2]$ as

$$(p_0^1, p_1^1, p_2^1, p_3^1, p_4^1, p_5^1) = \left(0, 0, \frac{1}{8h}, \frac{1}{16h}, 0, 0 \right) \tag{11}$$

and similarly for P_k^2 we obtain

$$(p_0^2, p_1^2, p_2^2, p_3^2, p_4^2, p_5^2) = \left(0, 0, \frac{1}{16h}, \frac{1}{4h}, \frac{1}{2h}, 0 \right). \tag{12}$$

Therefore, P_k^1 and P_k^2 are non-negative on open intervals $(x_{k-1}, x_{k-1}+h/2)$ and $(x_{k-1} + h/2, x_k)$, respectively. Due to the fact that $(p_4^1, p_5^1) = (p_0^2, p_1^2) = (0,0)$, the only root (with multiplicity two) of P_k on (x_{k-1}, x_k) is $x_{k-1} + h/2$. $\qquad \square$
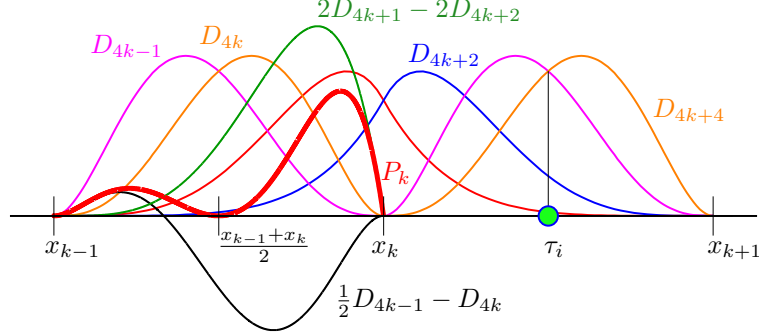
Figure 3: The assumption of existence of a single node τ_i inside $[x_k, x_{k+1}]$ implies $\tau_i = \frac{x_{k+1}+x_k}{2}$. Consequently, the rule (13) must return zero for P_k on $[x_{k-1}, x_k]$, i.e. $\mathcal{Q}_{x_{k-1}}^{x_k}(P_k) = 0$. As P_k is non-negative on (x_{k-1}, x_k) with one double root $\frac{x_{k-1}+x_k}{2}$, this fact violates the assumption of a single node in (x_k, x_{k+1}).

Remark 2. $D_{4k+1} - D_{4k+2}$, D_{4k-1}, D_{4k} are all positive polynomials on (x_{k-1}, x_k) and therefore there exist infinitely many non-negative blends. However, the existence of a non-negative blend when the coefficients have to satisfy a certain constraint is not obvious and the full impact of this particular blend with coefficients $2, \frac{1}{2}, -1$ will be seen later in Lemma 2.2.

2.2. Gaussian quadrature formulae

In this section, we derive a quadrature rule for the family $S_{5,1}^n$, see (2). Similarly to [2], we derive a quadrature rule that is optimal, exact and explicit.

With respect to exactness and optimality, according to [29, 30] there exists a quadrature rule

$$\mathcal{Q}_a^b(f) := \sum_{i=1}^{2n+1} \omega_i f(\tau_i) \cong \int_a^b f(t)\mathrm{d}t \tag{13}$$

that is exact for every function f from the space $S_{5,1}^n$. The explicitness follows from the construction.

Lemma 2.2. *Let $\mathcal{X}_n = (a = x_0, x_1, ..., x_n = b)$ be a uniform knot sequence. Each of the intervals $J_k = (x_{k-1}, x_k)$ $(k = 1, \ldots, n)$ contains at least two nodes of the Gaussian quadrature rule (13).*

Proof. We proceed by induction on the index of the segment J_k. There must be at least two nodes of the Gaussian quadrature rule inside the interval J_1. If there were no node inside J_1, the exactness of the rule would be violated for D_1. If there was only one node, using the exactness of the quadrature rule for D_3 and D_4, it must have been the midpoint $\tau_1 = \frac{(x_0+x_1)}{2}$ with the weight $\omega_1 = \frac{8}{15}h$. However, this contradicts exactness of D_1 and D_2 as $D_1 < D_2$ on (x_0, x_1).

Now, let us assume that every segment J_k, $k < n$, contains–two or more– Gaussian nodes and prove that J_{k+1} contains at least two nodes too. By contradiction, if there is no node inside (x_k, x_{k+1}), the exactness of the quadrature

rule (13) for D_{4k+3} is violated. If there is a single node in (x_k, x_{k+1}), due to the exactness of the quadrature rule (13) for D_{4k+3} and D_{4k+4}, it must be the midpoint $\tau_i = \frac{(x_k + x_{k+1})}{2}$ as it is their only intersection point, see Fig. 3, and their integrals are equal, see (7), $I[D_{4k+3}] = I[D_{4k+4}] = \frac{1}{6}$. The corresponding weight must be $\omega_i = \frac{8}{15}h$. Moreover $D_{4k+1}(\tau_i) - D_{4k+2}(\tau_i) = -\frac{5}{64h}$ and combining with ω_i, we have

$$2\omega_i(D_{4k+1}(\tau_i) - D_{4k+2}(\tau_i)) = -\frac{1}{12}. \tag{14}$$

Consider a blend $P_k(t) = 2D_{4k+1}(t) - 2D_{4k+2}(t) + \frac{1}{2}D_{4k-1}(t) - D_{4k}(t)$, see Fig. 3. As P_k is a blend of basis functions, the rule (13) must integrate it exactly on $[x_{k-1}, x_{k+1}]$, that is $\mathcal{Q}_{x_{k-1}}^{x_{k+1}}(P_k) = I(P_k) = -\frac{1}{12}$. However, combining this fact with (14), the rule must return zero when applied to P_k on $[x_{k-1}, x_k]$, i.e. $\mathcal{Q}_{x_{k-1}}^{x_k}(P_k) = 0$. But due to Lemma 2.1, P_k is non-negative on (x_{k-1}, x_k) with the only root at $\frac{x_{k-1} + x_k}{2}$, which contradicts the assumption of a single quadrature node in (x_k, x_{k+1}) and completes the proof. $\square$

Corollary 1. *If n is an* even *integer, then each of the intervals $J_k = (x_{k-1}, x_k)$ ($k = 1, 2, \ldots, n$) contains exactly two Gaussian nodes and the middle $x_{n/2} = (a + b)/2$ of the interval $[a, b]$ is also a Gaussian node. If n is* odd *then each of the intervals $J_k = (x_{k-1}, x_k)$ ($k = 1, 2, \ldots, n; k \neq (n+1)/2$) contains exactly two Gaussian nodes, while the interval $J_{(n+1)/2}$ contains three Gaussian nodes: the middle $(a + b)/2$ and the other two positioned symmetrically with respect to $(a + b)/2$.*

Proof. From [30], the optimal quadrature rule (13) is known to require $2n + 1$ Gaussian nodes. From Lemma 2.2, we know the location of $2n$ of them as each of the intervals J_k contains at least two nodes. The last node must be the midpoint $(a + b)/2$. We prove this by contradiction, distinguishing two cases depending on the parity of n. For n even, if one of the intervals J_k has more than two nodes then, by symmetry, J_{n-k} has to contain the same number of nodes and we exceed $2n + 1$, contradicting our quadrature rule (13). For n odd, let us assume that the middle interval $J_{(n+1)/2}$ contains exactly two nodes. Then, by symmetry, at least two of the remaining intervals contain three nodes, contradicting our quadrature rule (13). Therefore, the middle interval $J_{(n+1)/2}$ contains exactly three nodes, where the middle one is, again by symmetry, forced to be the midpoint $(a + b)/2$. $\square$

With the knowledge of the exact layout of the optimal quadrature nodes, we now construct a scheme that starts at the boundary of the interval and parses to its middle, recursively computing the nodes and weights. This process requires to solve only for the roots of a quadratic polynomial.

Let us denote

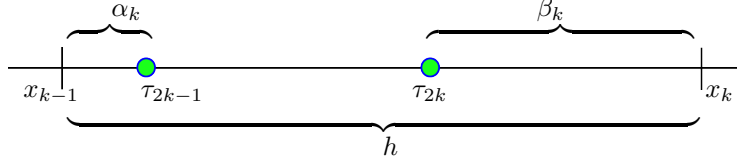$$\alpha_k = \tau_{2k-1} - x_{k-1}, \quad \beta_k = x_k - \tau_{2k}, \tag{15}$$

Figure 4: Notation on $[x_{k-1}, x_k]$.

where τ_{2k-1} and τ_{2k}, $\tau_{2k-1} < \tau_{2k}$, are the two quadrature nodes on (x_{k-1}, x_k), $k = 1, \ldots, [n/2] + 1$, see Fig. 4. Keeping in mind $h = x_k - x_{k-1}$, we have

$$x_k - \tau_{2k-1} = h - \alpha_k, \quad \tau_{2k} - x_{k-1} = h - \beta_k. \tag{16}$$

Let ω_{2k-1} and ω_{2k} be the corresponding weights of the Gaussian quadrature rule over the interval (x_{k-1}, x_k). The exactness requirement of the rule when applied to D_{4k-1} and D_{4k}, see (6) and (7), gives the following algebraic constraints

$$
\begin{aligned}
\omega_{2k-1}\alpha_k^2(h - \alpha_k)^3 + \omega_{2k}(h - \beta_k)^2\beta_k^3 &= \frac{h^6}{60}, \\
\omega_{2k-1}\alpha_k^3(h - \alpha_k)^2 + \omega_{2k}(h - \beta_k)^3\beta_k^2 &= \frac{h^6}{60}.
\end{aligned}
\tag{17}
$$

The exactness of the rule when applied on D_{4k-3} and D_{4k-2}, respectively, gives

$$\omega_{2k-1}(h - \alpha_k)^5 + \omega_{2k}\beta_k^5 = 4h^6 r_{4k-3},$$

$$\omega_{2k-1}\left(\frac{5(h - \alpha_k)^4}{2h^5} - \frac{9(h - \alpha_k)^5}{4h^6}\right) + \omega_{2k}\left(\frac{5\beta_k^4}{2h^5} - \frac{9\beta_k^5}{4h^6}\right) = r_{4k-2}, \tag{18}$$

where r_{4k-3} and r_{4k-2} are the *residues* between the exact integrals, see (7) and (8), and the result of the rule when applied to D_{4k-3} and D_{4k-2} on the previous interval $[x_{k-2}, x_{k-1}]$, respectively. That is

$$
\begin{aligned}
r_{4k-3} &= I[D_{4k-3}] - \mathcal{Q}_{x_{k-2}}^{x_{k-1}}(D_{4k-3}), \\
r_{4k-2} &= I[D_{4k-2}] - \mathcal{Q}_{x_{k-2}}^{x_{k-1}}(D_{4k-2}).
\end{aligned}
\tag{19}
$$

Due to the fact that both (17) and (18) are linear in ω_{2k-1} and ω_{2k}, their elimination from (17) gives

$$
\begin{aligned}
\omega_{2k-1} &= \frac{h^5(h - 2\beta_k)}{60\alpha_k^2(h - \alpha_k)^2(h - \alpha_k - \beta_k)}, \\
\omega_{2k} &= \frac{h^5(h - 2\alpha_k)}{60\beta_k^2(h - \beta_k)^2(h - \alpha_k - \beta_k)},
\end{aligned}
\tag{20}
$$

and from (18) we obtain

$$
\begin{aligned}
\omega_{2k-1} &= \frac{-2h^5(9\beta_k r_{4k-3} - 10h r_{4k-3} + \beta_k r_{4k-2})}{5(h - \alpha_k)^4(h - \alpha_k - \beta_k)}, \\
\omega_{2k} &= \frac{-2h^5(h r_{4k-3} + \alpha_k r_{4k-2} + 9 r_{4k-3}\alpha_k - h r_{4k-2})}{5\beta_k^4(h - \alpha_k - \beta_k)}.
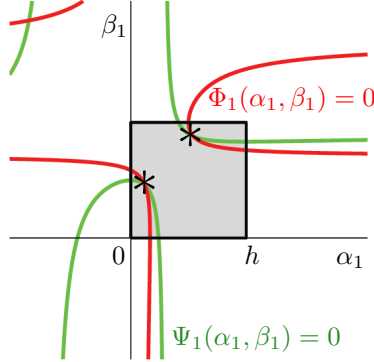\end{aligned}
\tag{21}
$$

10

Figure 5: The algebraic system (22) over the domain $(0, h) \times (0, h)$ (grey) for the first $(k = 1)$ subinterval $[x_0, x_1]$ is shown. The two intersection points correspond to the two Gaussian nodes on $[x_0, x_1]$ and are computed by projection onto α_1-axis using the resultant. The coordinates of the intersection points with respect to α_1-axis are the roots of the quadratic polynomial (24).

Equating ω_{2k-1} and ω_{2k} from (20) and (21) we obtain

$$
\begin{aligned}
\Phi_k(\alpha_k, \beta_k) &= 0, \\
\Psi_k(\alpha_k, \beta_k) &= 0,
\end{aligned}
\tag{22}
$$

an algebraic system of degree three with the unknowns α_k and β_k. Solving this non-linear system of two equations with two unknowns can be interpreted as the intersection problem of two algebraic curves, see Fig. 5. The domain of interest is $(0, h) \times (0, h)$ as both quadrature points lie inside (x_{k-1}, x_k).

Using the resultant, see e.g. [14], of these two algebraic curves in the direction of β_k, one obtains a univariate polynomial, in general, of degree nine. Interestingly, our system (22) produces–for all admissible residues r_{4k-3} and r_{4k-2}–only a quintic polynomial $E_k(\alpha_k)$ that gets factorized over $\mathbb{R}$ as

$$
Res_{\beta_k}(\Phi_k(\alpha_k, \beta_k), \Psi_k(\alpha_k, \beta_k)) = E_k(\alpha_k) = Q_k(\alpha_k)C_k(\alpha_k),
\tag{23}
$$

where Q_k is a quadratic factor and the vector of its coefficients with respect to the *monomial* basis, $\mathbf{q}_{mo}^k = (q_0^k, q_1^k, q_2^k)$, is–in terms of the residues–expressed as

$$
\begin{aligned}
q_2^k &= 1 - 480r_{4k-3} + 576r_{4k-3}^2 + 576r_{4k-2}^2 - 1152r_{4k-2}r_{4k-3}, \\
q_1^k &= 2h(12r_{4k-2} + 108r_{4k-3} - 1), \\
q_0^k &= h^2(1 - 24r_{4k-2} + 24r_{4k-3}),
\end{aligned}
\tag{24}
$$

and for the vector of monomial coefficients $\mathbf{c}_{mo}^k = (c_0^k, c_1^k, c_2^k, c_3^k)$ of the cubic factor C_k we obtain

$$
\begin{aligned}
c_3^k &= -216r_{4k-3} - 24r_{4k-2} + 2, \\
c_2^k &= h(24r_{4k-2} - 24r_{4k-3} - 5), \\
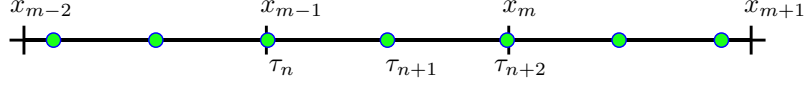c_1^k &= 4h^2, \\
c_0^k &= -h^3.
\end{aligned}
\tag{25}
$$

11

Figure 6: The situation for odd n on the middle interval $J_m = [x_{m-1}, x_m]$. The node τ_{n+1} is the middle of the interval, τ_n and τ_{n+2} are computed from (32).

We recall that two roots of E_k defined in (23) determine the two quadrature nodes that lie inside $[x_{k-1}, x_k]$. Interestingly, the cubic factor does not contribute to the computation of the nodes which is formalized as follows.

Lemma 2.3. *The cubic polynomial C_k defined in (25) has no roots inside $[0, h]$.*

A proof can be found in Appendix.

We now proceed to the main contribution of the paper: a recursive algorithm that computes the nodes and the weights of the Gaussian quadrature for uniform C^1 quintic splines. Due to Lemma 2.3, the recursion operates in a closed form fashion by solving for the roots of quadratic polynomials (24). Before we state the theorem, we need to establish some notation.

Let us denote by A_k and B_k, $k = 2, \ldots, [n/2] + 1$, the actual values of residues (19) when being evaluated at the nodes τ_{2k-3} and τ_{2k-2} on the interval $[x_{k-2}, x_{k-1}]$, i.e.,

$$
\begin{aligned}
A_k &= I[D_{4k-3}] - \omega_{2k-3} D_{4k-3}(\tau_{2k-3}) - \omega_{2k-2} D_{4k-3}(\tau_{2k-2}), \\
B_k &= I[D_{4k-2}] - \omega_{2k-3} D_{4k-2}(\tau_{2k-3}) - \omega_{2k-2} D_{4k-2}(\tau_{2k-2}),
\end{aligned}
\tag{26}
$$

and the coefficients of the quadratic polynomial (24) become

$$
\begin{aligned}
a_k &= 1 - 480 A_k + 576 A_k^2 + 576 B_k^2 - 1152 B_k A_k, \\
b_k &= 2h(12 B_k + 108 A_k - 1), \\
c_k &= h^2(1 - 24 B_k + 24 A_k).
\end{aligned}
\tag{27}
$$

In the case when n is odd, see Fig. 6, the middle subinterval contains three nodes and the algebraic system that needs to be solved results in a quadratic polynomial with the following coefficients

$$
\begin{aligned}
\widetilde{a}_m &= -2(108 A_m + 12 B_m + 1), \\
\widetilde{b}_m &= 2h(108 A_m + 12 B_m - 1), \\
\widetilde{c}_m &= h^2(12 A_m - 12 B_m + 1).
\end{aligned}
\tag{28}
$$

We are now ready to formalize the main theorem.

Theorem 2.1. *The sequences of nodes and weights of the Gaussian quadrature rule (13) are given explicitly by the initial values $A_1 = \frac{1}{24}$ and $B_1 = \frac{1}{8}$ and the recurrence relations ($k = 1, \ldots, [n/2]$) for the nodes*

$$
\tau_{2k-1} = x_{k-1} + \frac{-b_k - \sqrt{b_k^2 - 4 a_k c_k}}{2 a_k} \quad and \quad \tau_{2k} = x_k - \frac{-b_k + \sqrt{b_k^2 - 4 a_k c_k}}{2 a_k}
\tag{29}
$$

and for the weights

$$\omega_{2k-1} = \frac{h^5(2\tau_{2k} - 2x_{k-1} - h)}{60(\tau_{2k-1} - x_k + h)^2(x_k - \tau_{2k-1})^2(\tau_{2k} - \tau_{2k-1})},$$

$$\omega_{2k} = \frac{h^5(2x_k - 2\tau_{2k-1} - h)}{60(x_{k-1} - \tau_{2k} + h)^2(\tau_{2k} - x_{k-1})^2(\tau_{2k} - \tau_{2k-1})}. \tag{30}$$

If n is even (n = 2m), then $\tau_{n+1} = x_m = (a+b)/2$ and

$$\omega_{n+1} = 4h(\frac{1}{6} - A_{m+1}). \tag{31}$$

If n is odd (n = 2m − 1), then $\tau_{n+1} = (a+b)/2$,

$$\tau_n = x_{m-1} + \frac{-\widetilde{b}_m - \sqrt{\widetilde{b}_m^2 - 4\widetilde{a}_m\widetilde{c}_m}}{2\widetilde{a}_m} \quad and \quad \tau_{n+2} = x_m - \frac{-\widetilde{b}_m + \sqrt{\widetilde{b}_m^2 - 4\widetilde{a}_m\widetilde{c}_m}}{2\widetilde{a}_m} \tag{32}$$

and the corresponding weights are

$$\omega_n = \omega_{n+2} = \frac{h(108A_m + 12B_m - 1)^2}{30(156A_m - 36B_m + 1)},$$

$$\omega_{n+1} = \frac{4h(1152A_mB_m + 264A_m - 576A_m^2 - 576B_m^2 - 24B_m + 1)}{15(156A_m - 36B_m + 1)}. \tag{33}$$

Proof. We proceed by induction. Assume the quadrature nodes (τ_{2l-1}, τ_{2l}) and weights $(\omega_{2l-1}, \omega_{2l})$ are known for $l = 1, \ldots, k-1$ ($k \leq [n/2]$) and compute the new ones on (x_{k-1}, x_k). For $k = 1$, as there are no nodes on (x_{-1}, x_0), (26) gives $A_1 = I[D_1] = \frac{1}{24}$ and $B_1 = I[D_2] = \frac{1}{8}$. By Corollary 1, there are exactly two nodes inside (x_{k-1}, x_k). Due to Lemma 2.3, only the roots of the quadratic factor in (23) contribute to the computation of the nodes and hence solving $Q_k(\alpha_k) = 0$ with coefficients from (27) gives α_k and β_k. Combining these with (15) results in (29). The weights are computed from (20) using the identities (15) and (16). By Corollary 1, we have $\tau_{n+1} = (a+b)/2$. If n is even, using the exactness of the quadrature for D_{2n+1}, the associated weight is computed from

$$\frac{1}{6} = I[D_{2n+1}] = A_{m+1} + \omega_{n+1}D_{2n+1}(\tau_{n+1}). \tag{34}$$

Evaluating $D_{2n+1}((a+b)/2) = \frac{1}{4h}$ gives (31). If n is odd, due Corollary 1, there are three nodes inside (x_{m-1}, x_m); one is the middle point $(a+b)/2$ and the other two are symmetric with respect to it, see Fig. 6. Using the notation of (15) for the middle interval, i.e., $\alpha_m = \tau_{2m-1} - x_{m-1}$, the rule must integrate exactly D_{4m-3}, D_{4m-2} and D_{4m-1} which gives the following 3×3 algebraic system

$$\begin{aligned}
\frac{(h-\alpha_m)^5 + \alpha_m^5}{4h^6}\omega_n &+ \frac{1}{128h}\omega_{n+1} &= A_m, \\
\frac{(h-\alpha_m)^4(9\alpha_m + h) + \alpha_m^4(10h - 9\alpha_m)}{4h^6}\omega_n &+ \frac{11}{128h}\omega_{n+1} &= B_m, \\
\frac{10\alpha_m^2(h-\alpha_m)^2}{h^5}\omega_n &+ \frac{5}{16h}\omega_{n+1} &= \frac{1}{6},
\end{aligned} \tag{35}$$

13

Algorithm 1	GaussianQuadrature($[a,b], n$)

1: **INPUT**: compact interval $[a, b]$ and number of uniform segments n

2: $A_1 = \frac{1}{24}$; $B_1 = \frac{1}{8}$;
3: **for** $k = 1$ to $[n/2]$ **do**
4:     compute τ_{2k-1}, τ_{2k} from (29), and ω_{2k-1}, ω_{2k} from (30);
5: **end for**
6: $\tau_{n+1} = (a+b)/2$;   `/* middle node */`
7: **if** n is even **then**
8:     compute ω_{n+1} from (31);
9: **else**
10:    compute τ_n and τ_{n+2} from (32) and ω_n, ω_{n+1} and ω_{n+2} from (33);
11: **end if**
12: **for** $k = 1$ to $[n/2]$ **do**
13:    $\tau_{2n-2k+3} = \tau_{2k-1}$; $\tau_{2n-2k+2} = \tau_{2k}$;   `/*symmetry */`
14:    $\omega_{2n-2k+3} = \omega_{2k-1}$; $\omega_{2n-2k+2} = \omega_{2k}$;
15: **end for**
16: **OUTPUT**: $\{\tau_i, \omega_i\}_{i=1}^{2n+1}$, set of nodes and weights of the Gaussian quadrature for $S_{5,1}^n$ on interval $[a, b]$;

with unknowns α_m, ω_n and ω_{n+1}. Eliminating ω_{n+1} from the first two and second two equations, respectively, and solving for ω_n we obtain

$$\omega_n = \frac{2h^5(11A_m - B_m)}{5(h^2 - 2h\alpha_m + 2\alpha_m^2)(h - 2\alpha_m)^2} = \frac{h^5(240B_m - 11)}{60(h^2 + 9h\alpha_m - 9\alpha_m^2)(h - 2\alpha_m)^2}$$

and the problem reduces to solving for the roots of a univariate (rational) function in α_m. The numerator is a quadratic polynomial with coefficients (28) which proves (32). Inserting (28) into (35) and solving for ω_n and ω_{n+1} then gives (33) and completes the proof. $\qquad\square$

For the convenience, we summarize the recursion in Algorithm 1.

3. Error estimation for the C^1 quintic splines quadrature rule

In the previous section, we have derived a quadrature rule that exactly integrates functions from $S_{5,1}^n$ with uniform knot sequences. If f is not an element of $S_{5,1}^n$, the rule produces a certain error, also known as *remainder*. The analysis of this error is the objective of this section.

Let $W_1^r = \{f \in C^{r-1}[a, b]; \ f^{(r-1)}\text{abs. cont.}, \ ||f^{(r)}||_{L_1} < \infty\}$. As the quadrature rule (13) is exact for polynomials of degree at most five, for any element $f \in W_1^d$, $d \geq 6$, we have

$$R_{2n+1}[f] := I[f] - \mathcal{Q}_a^b[f] = \int_a^b K_6(R_{2n+1}; t)f^{(6)}(t)dt,$$

where the Peano kernel [20] is given by

$$K_6(R_{2n+1}; t) = R_{2n+1}\left[\frac{(t - .)_+^5}{5!}\right].$$

An explicit representation for the Peano kernel over the interval $[a, b]$ in terms of the weights and nodes of the quadrature rule (13) is given by

$$K_6(R_{2n+1}; t) = \frac{(t - a)^6}{720} - \frac{1}{120}\sum_{k=1}^{2n+1} \omega_k(t - \tau_k)_+^5. \tag{36}$$

Moreover, according to a general result for monosplines and quadrature rules [30], the only zeros of the Peano kernel over (a, b) are the knots of multiplicity four of the quintic spline. Therefore, for any $t \in (a, b)$, $K_6(R_{2n+1}; t) \geq 0$ and, by the mean value theorem for integration, there exists a real number $\xi \in [a, b]$ such that for $f \in C^6[a, b]$

$$R_{2n+1}(f) = c_{2n+1,6}f^{(6)}(\xi) \quad \text{with} \quad c_{2n+1,6} = \int_a^b K_6(R_{2n+1}; t)dt. \tag{37}$$

Hence, the constant $c_{2n+1,6}$ of the remainder R_{2n+1} is always positive and our quadrature rule belongs to the family of positive definite quadratures of order 6, e.g., see [31, 32, 35]. Integration of (36) gives

Theorem 3.1. *The error constant $c_{2n+1,6}$ in (37) of the quadrature rule (13) is given by*

$$c_{2n+1,6} = \frac{(b - a)^7}{5040} - \frac{1}{720}\sum_{k=1}^{2n+1} \omega_k(\tau_k - a)^6. \tag{38}$$

As a direct consequence of Theorem 3.1 and (37), one can upper bound the error of the quadrature rule, (29) and (30), when applied to a function f in W_1^r that does not belong to $S_{5,1}^n$. This requires only to have a bound of $f^{(6)}$ on $[a, b]$.

4. Numerical Experiments

In this section, we show some examples of quadrature nodes and weights for particular numbers of subintervals and discuss the asymptotic behavior of the rule for $n \to \infty$.

In the case when the domain is the whole real line, the exact and optimal rule is easy to compute. Similarly to [23], Eq.(29), where the rule was derived for $S_{4,1}$, for $S_{5,1}$ case one obtains

$$\int_{\mathbb{R}} f(t)\,\mathrm{d}t = \sum_{i \in \mathbb{Z}} h\left(\frac{7}{15}f(ih) + \frac{8}{15}f\left(\frac{2i + 1}{2}h\right)\right), \tag{39}$$

that is, the nodes are the knots and the middles of the subintervals. Similarly to [23], only two evaluations per subinterval are needed which gives 2/3 cost
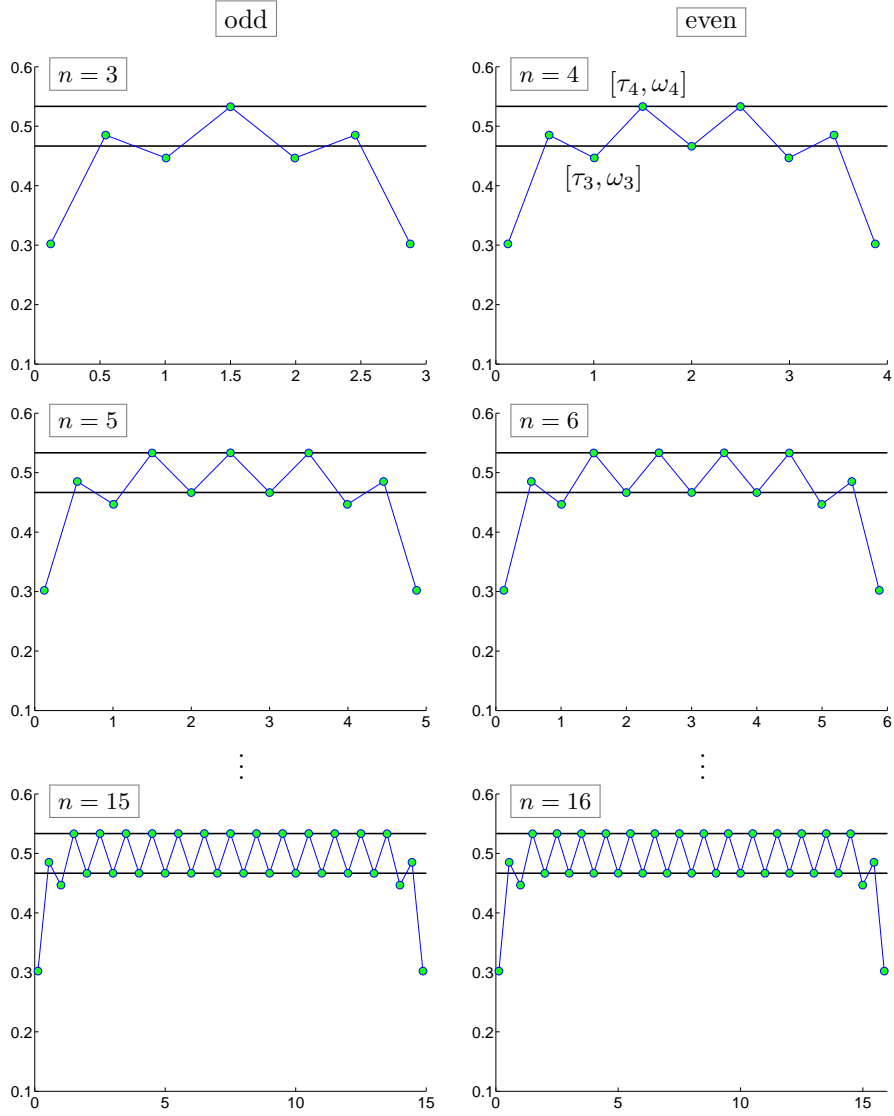
Figure 7: The quadrature rules (13) for various n are shown. The interval is set $[a, b] = [0, n]$, i.e., the distance between the neighboring knots is normalized to $h = 1$. The green dots visualize the quadrature; the x-coordinates are the nodes and the y-coordinates the corresponding weights. As $n \to \infty$, the nodes converge to the knots and the midpoints of the subintervals, the weights converge to $0.4\overline{6}$ and $0.5\overline{3}$ (black lines), cf. Table 1 and (39).

reduction ratio when compared to the classical Gaussian quadrature for polynomials. Observe the convergence of our general uniform rule to its limit, (39), when $n \to \infty$. The weights and nodes are shown in Table 1, see also Fig. 7. Only few initial nodes and weights differ from the limit values as $\frac{7}{15} = 0.4\overline{6}$ and $\frac{8}{15} = 0.5\overline{3}$. From Table 1 we conclude that for large values of n, one needs to compute only the first nine nodes and weights that differ from the limit values

Table 1: Nodes and weights for Gaussian quadrature (13) with double-precision for various n are shown. To observe the convergence to (39), the interval was set as $[a, b] = [0, n]$. Due to the symmetry, only the first $n + 1$ nodes and weights are displayed.

i	$n = 5$		$n = 6$	
	τ_i	ω_i	τ_i	ω_i
1	0.1225148226554413	0.3020174288145723	0.1225148226554413	0.3020174288145723
2	0.5441518440112252	0.4850196082224646	0.5441518440112252	0.4850196082224646
3	1.0064654716056596	0.4467177201362911	1.0064654716056596	0.4467177201362911
4	1.5002730728687338	0.5330387209380418	1.5002730728687338	0.5330387209380418
5	2.0000387957905171	0.4665398664562177	2.0000387972956304	0.4665398713719121
6	2.5	0.5333333108648244	2.5000000105321137	0.5333333220982075
7	–	–	3	0.4666666568370204
	$n = 7$		$n = 8$	
1	0.1225148226554413	0.3020174288145723	0.1225148226554413	0.3020174288145723
2	0.5441518440112252	0.4850196082224646	0.5441518440112252	0.4850196082224646
3	1.0064654716056596	0.4467177201362911	1.0064654716056596	0.4467177201362911
4	1.5002730728687338	0.5330387209380418	1.5002730728687338	0.5330387209380418
5	2.0000387972956304	0.4665398713719121	2.0000387972956304	0.4665398713719121
6	2.5000000105321137	0.5333333220982075	2.5000000105321137	0.5333333220982075
7	3.0000000015045261	0.4666666617518435	3.0000000015045293	0.4666666617518435
8	3.5	0.5333333333333333	3.5000000000000000	0.5333333333333333
9	–	–	4	0.4666666666666665
	$n = 9$		$n = 10$	
1	0.1225148226554413	0.3020174288145723	0.1225148226554413	0.3020174288145723
2	0.5441518440112252	0.4850196082224646	0.5441518440112252	0.4850196082224646
3	1.0064654716056596	0.4467177201362911	1.0064654716056596	0.4467177201362911
4	1.5002730728687338	0.5330387209380418	1.5002730728687338	0.5330387209380418
5	2.0000387972956304	0.4665398713719121	2.0000387972956304	0.4665398713719121
6	2.5000000105321137	0.5333333220982075	2.5000000105321137	0.5333333220982075
7	3.0000000015045293	0.4666666617518435	3.0000000015045293	0.4666666617518435
8	3.5000000000000000	0.5333333333333333	3.5000000000000000	0.5333333333333333
9	4.0000000000000000	0.4666666666666666	4.0000000000000000	0.4666666666666665
10	4.5	0.5333333333333333	4.5000000000000000	0.5333333333333333
11	–	–	5.0000000000000000	0.4666666666666666

by more than $\varepsilon = 10^{-16}$.

5. Conclusion

We have presented a recursive algorithm that computes quadrature nodes and weights for spaces of quintic splines with uniform knot sequences over finite domains. The presented quadrature is explicit, that is, in every step of the recursion the new nodes and weights are computed in closed form, without using

a numerical solver. The number of nodes per subinterval is two and hence the cost reduction compared to the classical Gaussian quadrature for polynomials is 2/3. For 3D problems where tensor product rules are used, this integration cost reduction ratio is only $(\frac{2}{3})^3 \doteq 27\%$.

We have also shown numerically that in the limit, when the length of the interval goes to infinity, our rule converges to the "two-third rule" of Hughes et al. [23] that is known to be exact and optimal over the real line.

For C^1 splines, our quadrature rule is optimal (Gaussian), that is, it requires minimal number of evaluations. However, it is still exact for any quintic splines with higher continuity than C^1 and therefore can be safely applied in such a space.

Acknowledgements

References

[1] C. Adam, T.J.R. Hughes, S. Bouabdallah, M. Zarroug, and H. Maitournam. Selective and reduced numerical integrations for NURBS-based isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering*, 284:732–761, 2015.

[2] R. Ait-Haddou, M. Bartoň, and V.M. Calo. Explicit Gaussian quadrature rules for C^1 cubic splines with symmetrically stretched knot sequences. *Journal of Computational and Applied Mathematics*, 290:543–552, 2015.

[3] M. Aizenshtein, M. Bartoň, and G. Elber. Global solutions of well-constrained transcendental systems using expression trees and a single solution test. *Computer Aided Geometric Design*, 29(5):265 – 279, 2012.

[4] K. E. Atkinson. *A Survey of Numerical Methods for the Solution of Fredholm Integral Equations of the Second Kind*. SIAM, Philadelphia, 1976.

[5] F. Auricchio, F. Calabrò, T. J. R. Hughes, A. Reali, and G. Sangalli. A simple algorithm for obtaining nearly optimal quadrature rules for NURBS-based isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering*, 249-252(1):15–27, 2012.

[6] M. Bartoň and V.M. Calo. Gauss–Galerkin quadrature rules for quadratic and cubic spline spaces and their application to isogeometric analysis. *Computer-Aided Design*, 82:57–67, 2017.

[7] M. Bartoň and V.M. Calo. Gaussian quadrature for splines via homotopy continuation: rules for C^2 cubic splines. *Journal of Computational and Applied Mathematics*, 296:709–723, 2016.

[8] M. Bartoň and V.M. Calo. Optimal quadrature rules for odd-degree spline spaces and their application to tensor-product-based isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering*, 305:217–240, 2016.

[9] A. Buffa, J. Rivas, G. Sangalli, and R. Vazquez. Isogeometric discrete differential forms in three dimensions. *SIAM J. Numer. Anal.*, 49:818–844, 2011.

[10] F. Calabrò, G. Sangalli, and M. Tani. Fast formation of isogeometric Galerkin matrices by weighted quadrature. *Computer Methods in Applied Mechanics and Engineering*, 2016.

[11] E. Cohen, R. F. Riesenfeld, and G. Elber. *Geometric Modeling with Splines: An Introduction*. A. K. Peters, 2001.

[12] N.O. Collier, L. Dalcin, and V. M. Calo. PetIGA: High-performance isogeometric analysis. `http://arxiv.org/abs/1305.4452`, 2013.

[13] J. A. Cottrell, T.J.R. Hughes, and Y. Bazilevs. *Isogeometric Analysis: Toward Integration of CAD and FEA*. John Wiley & Sons, 2009.

[14] D. Cox, J. Little, and D. O'Shea. *Using Algebraic Geometry*. Springer, 2005.

[15] H. B. Curry and I.J. Schoenberg. On Pólya frequency functions IV: the fundamental spline functions and their limits. *J. Analyse Math.*, 17:71–107, 1966.

[16] L. Dalcin and N. Collier. Petiga: High performance isogeometric analysis. https://bitbucket.org/dalcinl/petiga, 2012.

[17] C. de Boor. On calculating with B-splines. *Journal of Approximation Theory*, 6(1):50–62, 1972.

[18] G. Elber and M.S. Kim. Geometric constraint solver using multivariate rational spline functions. *SMA '01 Proceedings of the sixth ACM symposium on Solid modeling and applications*, pages 1 – 10, 2001.

[19] G. Farin, J. Hoschek, and M. S. Kim. *Handbook of Computer Aided Geometric Design*. Elsevier, Amsterdam, 2002.

[20] W. Gautschi. *Numerical Analysis*. Springer, 1997.

[21] R. Hiemstra, F. Calabrò, and T.J.R. Schillinger, D.and Hughes. Optimal and reduced quadrature rules for tensor product and hierarchically refined splines in isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering*, 2016.

[22] M. Hillman, J.S. Chen, and Y. Bazilevs. Variationally consistent domain integration for isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering*, 284:521–540, 2015.

[23] T.J.R. Hughes, A. Reali, and G. Sangalli. Efficient quadrature for NURBS-based isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering*, 199(58):301 – 313, 2010.

[24] K.A. Johannessen. Optimal quadrature for univariate and tensor product splines. *Computer Methods in Applied Mechanics and Engineering*, 2016.

[25] P. Köhler and G. Nikolov. Error bounds for Gauss type quadrature formulae related to spaces of splines with equidistant knots. *Journal of Approximation Theory*, 81:368–388, 1995.

[26] P. Köhler and G. Nikolov. Error bounds for optimal definite quadrature formulae. *Journal of Approximation Theory*, 81:397–405, 1995.

[27] A. Mantzaflaris and B. Jüttler. Exploring matrix generation strategies in isogeometric analysis. In *International Conference on Mathematical Methods for Curves and Surfaces*, pages 364–382. Springer, 2012.

[28] A. Mantzaflaris and B. Jüttler. Integration by interpolation and look-up for galerkin-based isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering*, 284:373–400, 2015.

[29] C. Micchelli. The fundamental theorem of algebra for monosplines with multiplicities. *Lineare Operatoren und Approximation*, pages 419–430, 1972.

[30] C.A. Micchelli and A. Pinkus. Moment theory for weak Chebyshev systems with applications to monosplines, quadrature formulae and best one-sided l^1 approximation by spline functions with fixed knots. *SIAM J. Math. Anal.*, 8:206 – 230, 1977.

[31] G. Nikolov. Asymptotically optimal definite quadrature formulae. *ZAMM SII*, 75:653 – 654, 1995.

[32] G. Nikolov. On certain definite quadrature formulae. *Journal of Computational and Applied Mathematics*, 75(2):329 – 343, 1996.

[33] A. Sarmiento, D. Garcia, L. Dalcin, N. Collier, and V. M. Calo. Micropolar fluids using B-spline divergence conforming spaces. In *Procedia Computer Science*, volume 29, pages 991–1001, 2014.

[34] D. Schillinger, S. Hossain, and T.J.R. Hughes. Reduced bézier element quadrature rules for quadratic and cubic splines in isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering*, 277:1–45, 2014.

[35] G. Schmeisser. Optimale Quadraturformeln mit semidefiniten Kernen. *Numer. Math.*, 20:32 – 53, 1972.

[36] E.C. Sherbrooke and N.M. Patrikalasis. Computation of the solutions of nonlinear polynomial systems. *Computer Aided Geometric Design*, 10(5):379 – 405, 1993.

[37] I. H. Sloan. A quadrature-based approach to improving the collocation method. *Numerische Mathematik*, 54(1):41 – 56, 1988.

[38] G. Strang and G.J. Fix. *An analysis of the finite element method*, volume 212. Prentice-hall Englewood Cliffs, NJ, 1973.

Appendix

Proof of Lemma 2.3. Without loss of generality, we may assume $h = 1$ as the roots of C_k change with scaling factor h, cf. (25). The cubic C_k can be split into two summands f_k and g_k, the first independent and the latter dependent on r_{4k-3} and r_{4k-2}, i.e.

$$f_k = 2t^3 - 5t^2 + 4t - 1 = (t-1)^2(t - \frac{1}{2}),$$

$$g_k = (-216r_{4k-3} - 24r_{4k-2})t^3 + (24r_{4k-2} - 24r_{4k-3})t^2 \tag{40}$$

$$= -24t^2(t - \frac{r_{4k-2} - r_{4k-3}}{9r_{4k-3} + r_{4k-2}})$$

We show that $C_k(t) < 0$ on $[0,1]$. We denote by ξ_g the non-zero root of g_k,

$$\xi_g = \frac{r_{4k-2} - r_{4k-3}}{9r_{4k-3} + r_{4k-2}},$$

and consider the particular subdivision of $[0,1]$ into 1) $[0, \xi_g]$, 2) $[\xi_g, \frac{1}{2}]$, and 3) $[\frac{1}{2}, 1]$. We investigate C_k on each of these three subintervals separately:

Case 3) We express C_k in Bernstein (B) basis and show all its coefficients are negative. Let $T_{[\frac{1}{2},1]}$ be the transformation matrix [19] from monomial to Bernstein basis on $[\frac{1}{2}, 1]$

$$T_{[\frac{1}{2},1]} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ \frac{1}{2} & \frac{2}{3} & \frac{5}{6} & 1 \\ \frac{1}{4} & \frac{5}{12} & \frac{2}{3} & 1 \\ \frac{1}{8} & \frac{1}{4} & \frac{1}{2} & 1 \end{pmatrix}, \tag{41}$$

and let $\mathbf{c}_B^k = (c_0^B, c_0^B, c_0^B, c_0^B)$ be the vector of Bernstein coefficients. Then the conversion is given by $\mathbf{c}_B^k = \mathbf{c}_{mo}^k T_{[\frac{1}{2},1]}$ and we obtain

$$\begin{aligned} c_0^B &= -33r_{4k-3} + 3r_{4k-2}, \\ c_1^B &= \frac{1}{12} - 64r_{4k-3} + 4r_{4k-2}, \\ c_2^B &= -124r_{4k-3} + 4r_{4k-2}, \\ c_3^B &= -240r_{4k-3}. \end{aligned} \tag{42}$$

Looking at the c_i^B coefficients, we start with the second one and prove that $64r_{4k-3} - 4r_{4k-2} > \frac{1}{12}$. We know that r_{4k-3} and r_{4k-2} are by definition both positive and also $r_{4k-2} > r_{4k-3}$, which is a direct consequence of $D_{4k+1}(t) > D_{4k+2}(t)$ on (x_{k-1}, x_k), and also $r_{4k-3}, r_{4k-2} < \frac{1}{6}$. Consider the blend $P_k(t)$ from Lemma 2.1. $P_k(t) \geq 0$ gives polynomial inequality $2D_{4k+1}(t) - 2D_{4k+2}(t) + \frac{1}{2}D_{4k-1}(t) \geq D_{4k}(t)$. By evaluating both sides at the two nodes and by multiplying with corresponding weights, i.e., by applying the quadrature rule $\mathcal{Q}$ on $[x_{k-1}, x_k]$, we obtain

$$2(4r_{4k-3} - r_{4k-2}) + \frac{1}{12} \geq \frac{1}{6} \tag{43}$$

because D_{4k-1} and D_{4k} act only on this interval and hence the rule reproduces their integrals exactly. Combining (43) with $r_{4k-3} > 0$ proves the desired inequality. Moreover, combining (43) with $\frac{1}{6} > r_{4k-2}$ gives

$$16r_{4k-3} > 5r_{4k-2}, \tag{44}$$

and the other three inequalities follow directly from (44) and the fact $r_{4k-3} > 0$.

Case 1) Similarly to case 3), we compute the c_i^B coefficients and show that all are negative. The conversion is given by $\mathbf{c}_B^k = \mathbf{c}_{mo}^k T_{[0,\xi_g]}$, where

$$T_{[0,\xi_g]} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & \frac{1}{3}\xi_g & \frac{1}{2}\xi_g & \xi_g \\ 0 & 0 & \frac{1}{3}\xi_g^2 & \xi_g^2 \\ 0 & 0 & 0 & \xi_g^3 \end{pmatrix} \tag{45}$$

is the corresponding transformation matrix. We obtain $\mathbf{c}_B^k = (c_0^B, c_1^B, c_2^B, c_3^B)$

$$
\begin{aligned}
c_0^B &= -1, \\
c_1^B &= \frac{-31r_{4k-3} + r_{4k-2}}{3(9r_{4k-3} + r_{4k-2})}, \\
c_2^B &= -\frac{4}{3}\frac{80r_{4k-3}^2 - 5r_{4k-3}r_{4k-2} + 6(r_{4k-3} - r_{4k-2})^3}{(9r_{4k-3} + r_{4k-2})^2}, \\
c_3^B &= \frac{-100r_{4k-3}^2(11r_{4k-3} - r_{4k-2})}{(9r_{4k-3} + r_{4k-2})^3}.
\end{aligned}
\tag{46}
$$

Negativity of c_1^B and c_3^B follows directly from (44). It remains to prove

$$80r_{4k-3}^2 - 5r_{4k-3}r_{4k-2} + 6(r_{4k-3} - r_{4k-2})^3 > 0,$$

which using (44) and $r_{4k-2} > r_{4k-3}$ simplifies to

$$64r_{4k-3}^2 > 6r_{4k-2}^3,$$

which again follows from (44) using $r_{4k-2} < \frac{1}{6}$.

Case 2) f_k has a double root at $t = 1$ and g_k has a double root at $t = 0$. For the third root, it holds

$$\frac{r_{4k-2} - r_{4k-3}}{9r_{4k-3} + r_{4k-2}} < \frac{1}{2} \tag{47}$$

which follows from (44). From case 3), we know $g_k(\frac{1}{2}) = -33r_{4k-3} + 3r_{4k-2}$ and from case 1) $f_k(\xi_g) = c_3^B$ that are both negative. Moreover, we know that f_k is monotonically increasing while g_k is monotonically decreasing on $(\xi_g, \frac{1}{2})$. Therefore $f_k(t) < 0$ on $[\xi_g, \frac{1}{2})$ and $g_k < 0$ on $(\xi_g, \frac{1}{2}]$ which completes the proof. $\qquad\square$