Gauss-Green cubature and moment computation over arbitrary geometries ⋆

Alvise Sommariva [a], Marco Vianello [a,*]

[a] *Department of Pure and Applied Mathematics, University of Padua (Italy)*

Abstract

We have implemented in Matlab a Gauss-like cubature formula over arbitrary bivariate domains with a piecewise regular boundary, which is tracked by splines of maximum degree p (spline curvilinear polygons). The formula is exact for polynomials of degree at most $2n-1$ using $N \sim cmn^2$ nodes, $1 \le c \le p$, m being the total number of points given on the boundary. It does not need any decomposition of the domain, but relies directly on univariate Gauss-Legendre quadrature via Green's integral formula. Several numerical tests are presented, including computation of standard as well as orthogonal moments over a nonstandard planar region.

Key words: Gauss-like cubature, splines, curvilinear polygons, Green's formula, moment computation

1 Introduction.

We consider the problem of constructing a cubature formula over bivariate domains with piecewise regular boundary (curvilinear polygons)

$$\sum_{\lambda \in \Lambda_{2n-1}} \omega_\lambda \, f(x_\lambda, y_\lambda) \approx I_\Omega(f) = \iint_\Omega f(x,y)\, dxdy \ , \ \ \Omega \subset \mathbb{R}^2 \qquad (1)$$

which is exact for all bivariate polynomials of degree at most $2n-1$, stable (i.e. such that $\sum_{\lambda \in \Lambda_{2n-1}} |\omega_\lambda|$ is bounded with n), and simple to implement

⋆ Work supported by the ex-60% funds of the University of Padova and by the GNCS-INdAM.

* Corresponding author. Address: via Trieste 63, 35121 Padova (Italy). Fax: +39 049 8271444. Telephone: +39 049 8271370.

Email address: marcov@math.unipd.it (Marco Vianello).

by one of the most popular computing tools, Matlab (cf. [16]). We make the further assumption that the boundary is discretized by a suitable sequence of points, split into subsequences (one for each regular boundary piece), and that it is tracked by a spline interpolating curve of suitable degree $\leq p$ on each subsequence (spline curvilinear polygon); cf. [2]. Thus the only information we have on the domain are the split discrete boundary and the spline type on each subsequence.

Our work starts from the observation that numerical cubature codes for such general domains do not seem to be available (in particular, Matlab codes). Typically, the domain has to be split by the user into "simpler" parts (triangles, generalized rectangles, generalized sectors) where then suitable adaptive methods are used; see, e.g., the popular `CubPack` package [4]. Alternatively, a brute-force approach is suggested to manage nonstandard domains (like e.g. by the Matlab `dblquad` automatic integrator, cf. [16]) that is integrating the product of the given integrand by the characteristic function of the domain on some enclosing rectangle. Such a technique can work but is often unreliable and clearly inefficient since an artificial discontinuity at the boundary is introduced.

In a recent paper [25], we have introduced a completely different approach, in the special case of cubature over polygons, obtaining a product-like Gauss formula. The key idea is that of resorting to Green's integral formula (the divergence theorem in dimension 2, cf. [1])

$$I_\Omega(f) = \oint_{\partial\Omega} \mathcal{F}(x,y)\, dy \ , \quad \mathcal{F}(x,y) = \int f(x,y)\, dx \ , \tag{2}$$

(f being continuous on a domain Ω with piecewise regular boundary described counterclockwise), via Gauss-Legendre discretization (cf. e.g. [9]) of the x-primitive $\mathcal{F}(x,y)$. Such a discretized primitive is then integrated along the sides still by Gauss-Legendre quadrature (we have recently used Green's formula also in the context of cubature from scattered data by radial basis functions, cf. [25,26]). When the integrand is a bivariate polynomial of degree at most $2n-1$, the first quadrature is exact with n nodes and gives a polynomial of degree at most $2n$, which restricted to a side is a polynomial of degree at most $2n$ in the side parametrization. This can be integrated exactly by $n+1$ Gauss-Legendre nodes on the side. No triangulation is required, since Green's formula needs only the boundary as a counterclockwise sequence of vertices.

Here we extend the method to spline curvilinear polygons, which allow to approximate very general domains. When the integrand is a bivariate polynomial of degree at most $2n-1$, integration of the discretized x-primitive between any couple of consecutive points on the boundary entails integration of a polynomial of degree at most $2np+p-1$ in the underlying parametrization (recall that we use splines of degree $q \leq p$ on each subsequence of points on the boundary).

This can be accomplished by at most $nq + (q+1)/2$ Gauss-Legendre nodes (q odd). Hence, we get algebraic degree of exactness $2n-1$ using a total number of cubature nodes N, with $mn(n+1) \leq N \leq mn(np + (p+1)/2)$, where m is the total number of points on the boundary. Again, no decomposition of the bivariate domain is needed, since Green's formula via spline tracking of the boundary needs only a counterclockwise sequence of points (organized in suitable subsequences).

It is worth recalling that an approach based on tracking boundaries by spline curves and on Green's formula, has been recently adopted to compute exact moments of planar figures; cf., e.g., [13,21] and references therein. Indeed, Green's formula (also in its discrete version) is widely used in pattern recognition and image analysis (cf., e.g., [18,6,22,23] and references therein), whereas it seems much less popular in the framework of numerical cubature.

The paper is organized as follows. The cubature formula with stability and error estimates and a discussion on some of its features are given in section 2. In section 3, we show the behavior of the formula by integrating some test functions over a nonconvex domain (a lune), where for comparison the boundary is tracked by splines of different degree. Moreover, we give an application which exploits polynomial exactness of the cubature formula over general planar regions, namely the computation of standard as well as orthogonal area moments over a nonstandard hand-shape region.

We like to conclude this introduction recalling that Carl Friedrich Gauss (1777-1855) was the father of two of the three key tools used in the cubature formula: the divergence theorem (known also as Gauss-Green theorem, cf. [7,12]), and interpolatory quadrature at the zeros of Legendre polynomials (cf. [8]). Since the third key tool is spline interpolation, we have decided to term "`SplineGauss`" the Matlab cubature code, cf. [27].

2 Gauss-Green cubature via spline boundaries.

We begin by stating the main result of the paper (construction of Gauss-like cubature formulas over spline curvilinear polygons) as a theorem.

Theorem 2.1 *Let $\Omega \subset \mathbb{R}^2$ be the closure of a bounded and simply connected domain with piecewise regular boundary, which is described counterclockwise by a sequence of "vertices"*

$$V_i = (\alpha_i, \beta_i) \; , \;\; i = 1, \ldots, \nu \; ,$$

$$\partial\Omega = (V_1 \frown V_2) \cup (V_2 \frown V_3) \cup \cdots \cup (V_\nu \frown V_{\nu+1}) \; , \;\; V_{\nu+1} = V_1 \; , \qquad (3)$$

where each curvilinear side $(V_i \frown V_{i+1})$ *is tracked by a spline curve* $S_i(t)$ *of degree* p_i*, interpolating an ordered subsequence of* m_i *boundary control points* $P_{i1} = V_i, P_{i2}, \ldots, P_{im_i} = V_{i+1}$*, with a suitable parametrization*

$$S_i(t) = (S_{i1}(t), S_{i2}(t)) \ , \ \ t \in [\alpha_i, \beta_i] \ , \ \ S_i(t_{ij}) = P_{ij} \ , \ j = 1, \ldots, m_i \ . \tag{4}$$

Let $f \in C(\mathcal{R})$ *and let* ξ *be a fixed abscissa, where*

$$\Omega \subseteq \mathcal{R} = [a, b] \times [c, d] \ , \ \ \xi \in [a, b] \ . \tag{5}$$

Let $\{\tau_k^s\}$ *and* $\{w_k^s\}$*,* $1 \le k \le s$*, be the nodes and weights of the Gauss-Legendre quadrature formula of degree of exactness* $2s - 1$ *on* $[-1, 1]$*, cf.* [10].

Then, the following cubature formula is exact over Ω *for all bivariate polynomials of degree at most* $2n - 1$

$$I_{2n-1}(f) = \sum_{\lambda \in \Lambda_{2n-1}} \omega_\lambda f(x_\lambda, y_\lambda) \ , \tag{6}$$

where λ *is a 4-index*

$$\Lambda_{2n-1} = \{\lambda = (i, j, k, h) : \ 1 \le i \le L, \ 1 \le j \le m_i - 1 \ , 1 \le k \le n_i, \ 1 \le h \le n\} \ , \tag{7}$$

m_i *is the number of points* $\{P_{ij}\}$ *on the side* $(V_i \frown V_{i+1})$*, and* n_i *depends on the type of splines used on such a side*

$$n_i = \begin{cases} np_i + p_i/2 \ , & p_i \ \textit{even} \\ np_i + (p_i + 1)/2 \ , & p_i \ \textit{odd} \end{cases} \tag{8}$$

The nodes and weights in (6) *are given by*

$$x_\lambda = \frac{1 + \tau_h^n}{2} \, S_{i1}(q_{ijk}) \ + \ \frac{1 - \tau_h^n}{2} \ , \ \ y_\lambda = S_{i2}(q_{ijk}) \ , \tag{9}$$

$$\omega_\lambda = \frac{\Delta t_{ij}}{4} \, w_k^{n_i} \, w_h^n \, (S_{i1}(q_{ijk}) \ - \xi) \ S'_{i2}(q_{ijk}) \ , \tag{10}$$

where we have defined

$$q_{ijk} = \frac{\Delta t_{ij}}{2} \, \tau_k^{n_i} \ + \ \frac{t_{ij+1} + t_{ij}}{2} \ , \ \ \Delta t_{ij} = t_{ij+1} - t_{ij} \ . \tag{11}$$

The overall number of cubature nodes is

$$N = n \sum_{i=1}^{\nu} (m_i - 1) n_i \ , \tag{12}$$

with

$$mn(n + 1) \le N \le mn \left(np + \frac{p + 1}{2} \right) \ , \ \ p = \max_{1 \le i \le \nu} \{p_i\} \ , \tag{13}$$

m being the overall number of control points given on the boundary. Moreover, we have the stability estimate

$$\sum_{\lambda \in \Lambda_{2n-1}} |\omega_\lambda| \leq (b-a)\, \ell_n \ , \quad \lim_{n \to \infty} \ell_n = \ell(\partial\Omega) \ , \tag{14}$$

and the error estimate

$$|I_\Omega(f) - I_{2n-1}(f)| \leq (\mu(\Omega) + (b-a)\, \ell_n)\, E_{2n-1}(f; \mathcal{R}) \ ,$$

$$E_{2n-1}(f; \mathcal{R}) = \min_{p \in \mathbb{P}^2_{2n-1}} \|f - p\|_{\infty, \mathcal{R}} \ , \tag{15}$$

where $\ell(\partial\Omega)$ denotes the length of the boundary and $\mu(\Omega)$ the Lebesgue measure of the integration domain.

Proof. By Green's formula (2) and (3)-(4) we can write

$$I_\Omega(f) = \sum_i \int_{(V_i \frown V_{i+1})} \mathcal{F}(x,y)\, dy = \sum_{i,j} \int_{P_{ij} \frown P_{ij+1}} \mathcal{F}(x,y)\, dy \ , \tag{16}$$

where we have taken the x-primitive

$$\mathcal{F}(x,y) = \int_\xi^x f(v,y)\, dv \ . \tag{17}$$

Notice that it is important to have a fixed abscissa ξ here, in order to fix the primitive in Green's formula. Then by using the parametrization

$$\sum_{i,j} \int_{P_{ij} \frown P_{ij+1}} \mathcal{F}(x,y)\, dy = \sum_{i,j} \int_{t_{ij}}^{t_{ij+1}} \mathcal{F}(S_i(t))\, S'_{i2}(t)\, dt$$

$$= \sum_{i,j} \int_{t_{ij}}^{t_{ij+1}} \left(\int_\xi^{S_{i1}(t)} f(v, S_{i2}(t))\, dv \right) S'_{i2}(t)\, dt$$

$$= \sum_{i,j} \frac{\Delta t_{ij}}{4} \iint_{[-1,1]^2} (S_{i1}(q_{ij}(u)) - \xi)\, S'_{i2}(q_{ij}(u))\, f \left(\frac{S_{i1}(q_{ij}(u)) - \xi}{2} \tau + \right.$$

$$\left. + \frac{S_{i1}(q_{ij}(u)) + \xi}{2}, S_{i2}(q_{ij}(u)) \right) d\tau\, du \ , \tag{18}$$

where we have used the standard affine change of variables to rewrite the univariate integrals in $[-1,1]$, and we have defined

$$q_{ij}(u) = \frac{\Delta t_{ij}}{2}\, u + \frac{t_{ij+1} + t_{ij}}{2} \ . \tag{19}$$

Now observe that, when $f(x,y)$ is a polynomial of degree at most $2n-1$, then the integrand in (18) is again a bivariate polynomial of degree at most $2n-1$ in τ and $2np_i + p_i - 1$ in u for every j, and thus it is integrated exactly in $[-1,1]^2$ by the product Gauss-Legendre formula with $n \times n_i$ nodes and weights (n_i being defined in (8)). The application of such a formula gives (6)-(11): in particular, the $\{q_{ijk}\}_k$ are n_i Gauss-Legendre quadrature nodes in the parameter interval $[t_{ij}, t_{ij+1}]$, since $q_{ijk} = q_{ij}(\tau_k^{n_i})$. Moreover, (12)-(13) are immediately obtained by counting the overall number of values $\{q_{ijk}\}_k$ pertaining the union of spline subarcs $P_{ij} \frown P_{ij+1}$, recalling that for each of such values there are exactly n cubature nodes as in (9).

As for estimate (14), first recall that the sum of the Gauss-Legendre weights is 2, and observe that $|S_{i1}(t) - \xi| \leq b - a$ for every i and $t \in [\alpha_i, \beta_i]$. Then, by (10) we can write

$$\sum_{\lambda} |\omega_\lambda| \leq \frac{b-a}{2} \sum_h w_h^n \sum_{i,j,k} \frac{\Delta t_{ij}}{2} w_k^{n_i} |S'_{i2}(q_{ijk})| \leq (b-a)\ell_n \ ,$$

$$\ell_n = \sum_i \ell_{in} \to \sum_i \ell(V_i \frown V_{i+1}) = \ell(\partial\Omega) \ , \quad n \to \infty \ , \tag{20}$$

since the bound

$$\ell_{in} = \sum_{j,k} \frac{\Delta t_{ij}}{2} w_k^{n_i} \sqrt{(S'_{i1}(q_{ijk}))^2 + (S'_{i2}(q_{ijk}))^2} \geq \sum_{j,k} \frac{\Delta t_{ij}}{2} w_k^{n_i} |S'_{i2}(q_{ijk})| \tag{21}$$

is a Gauss-Legendre quadrature formula applied to the integral defining the length of the spline arc $V_i \frown V_{i+1}$.

Finally, (15) is the extension to the cubature framework of the well-known error estimate for Polya-Steklov-like quadrature formulas (cf. e.g. [14,28]). In fact, denoting by p^*_{2n-1} the best uniform polynomial approximation to f on Ω with degree $2n-1$, by polynomial exactness we have

$$|I_\Omega(f) - I_{2n-1}(f)| \leq |I_\Omega(f) - I_\Omega(p^*_{2n-1})| + |I_\Omega(p^*_{2n-1}) - I_{2n-1}(p^*_{2n-1})|$$

$$+|I_{2n-1}(p^*_{2n-1}) - I_{2n-1}(f)| \leq \left(\mu(\Omega) + \sum_{\lambda \in \Lambda_{2n-1}} |\omega_\lambda|\right) E_{2n-1}(f;\mathcal{R}) \ . \qquad \textbf{q.e.d.}$$

We make now some remarks, in order to deepen some important features of Gauss-Green cubature over spline curvilinear polygons: convergence rate, boundary approximation, location of the cubature nodes, shape of the domains.

Remark 1 (*convergence rate*)

Concerning the convergence rate of (6) as $n \to \infty$, by the multivariate extension of Jackson theorem (cf. e.g. [19]) and (15), we get immediately

$$I_\Omega(f) = I_{2n-1}(f) + \mathcal{O}\left((2n-1)^{-(p+\theta)}\right) \ , \quad f \in C^{p+\theta}(\mathcal{R}) \ , \tag{22}$$

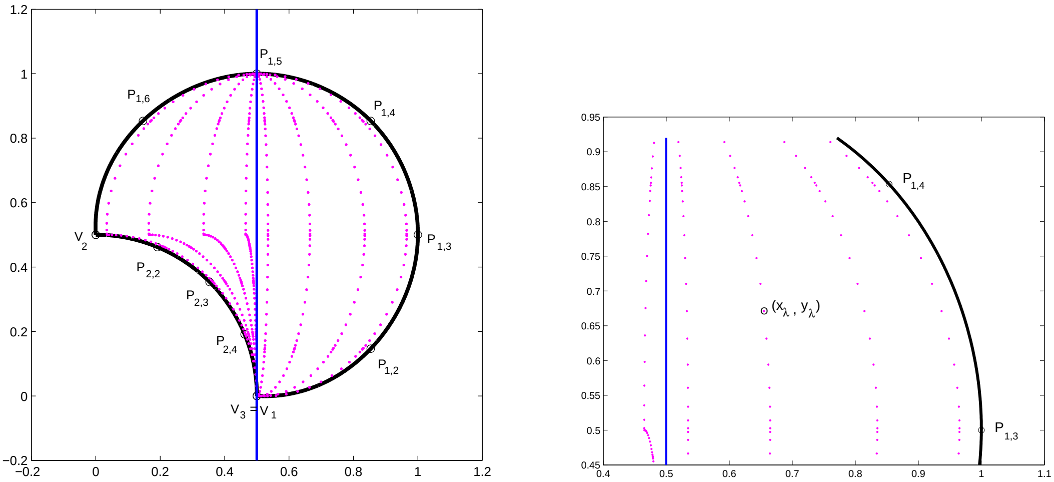

Fig. 1. Example of distribution of boundary control points and cubature points on a lune (top), with a suitable zoom (bottom); the reference abscissa in (5) is $\xi = 0.5$.

for every function f with Hölder continuous p-th partial derivatives, i.e. $p \geq 0$ and $\theta \in (0, 1]$.

Remark 2 (*boundary approximation*)

One of the goals of constructing an algebraic cubature formula like (6), is that of approximating the integral of a given function over some domain Ω_0, whose boundary is approximated by a spline interpolating curve, that is $\partial\Omega \approx \partial\Omega_0$, where $\Omega \approx \Omega_0$ is the corresponding spline curvilinear polygon as in the statement of Theorem 1.

This approach allows to treat quite general domains, with two distinct practical situations. One is when the boundary is not known analytically, but is given by a possibly nonrefineable sample (think e.g. to a geographical region, a digital image, ...) and then is tracked by spline curves. The other situation arises when the boundary is known analytically, and the spline interpolating curve is chosen taking into account the boundary regularity. For example, if the domain is the union or difference of overlapping domains with piecewise regular boundaries, it may be convenient to add the relevant boundary intersections among the vertices $\{V_i\}$ of the spline curvilinear polygon, in order to avoid a rough approximation in their neighborhood. Moreover, an appropriate choice of vertices can allow to track possible linear sides by only two points. See, e.g., Figure 1 where the domain is the difference of two overlapping disks (a lune), the sides being two circular arcs tracked by cubic splines with 5 and 7 equispaced control points, respectively (thus we have two vertices V_1, V_2, and 10 overall control points).

Clearly, the effect of the approximation of the boundary of Ω_0 has be taken into account in estimating the overall integration error. The quality of the approximation depends on various features of the spline construction, like the spline degrees (via the boundary regularity), but also the choice of the parametrization at a given degree. We recall two widely adopted parametrizations (the symbols refer to (4)), that are the "equal increment" one

$$[\alpha_i, \beta_i] = [1, m_i] \;,\;\; t_{ij} = j \;,\;\; j = 1, \dots, m_i \;, \tag{23}$$

and the "cumulative chordal" one

$$[\alpha_i, \beta_i] = \left[0, \sum_{j=1}^{m_i-1} |\Delta t_{ij}|\right] \;,\;\; \Delta t_{ij} = |P_{ij+1} - P_{ij}| \;,\;\; j = 1, \dots, m_i - 1 \;, \tag{24}$$

cf., e.g., [2,5,29].

Assume for simplicity that the boundary has a global C^4 parametrization, and that it is tracked by a cubic spline curve via the cumulative chordal parametrization (i.e., $i = \nu = 1$ in (3)-(4) and (24)). Then, setting $H = \max \Delta t_{1j}$, cf. (24), and using $\Omega_0 \triangle \Omega = (\Omega_0 \setminus \Omega) \cup (\Omega \setminus \Omega_0)$, by the results in [20] we can write the following error estimate

$$\begin{aligned} |I_{\Omega_0}(f) - I_{2n-1}(f)| &\leq I_{\Omega_0 \triangle \Omega}(|f|) + |I_\Omega(f) - I_{2n-1}(f)| \\ &\leq \|f\|_{\infty, \Omega_0 \triangle \Omega}\, \mu(\Omega_0 \triangle \Omega) + (\mu(\Omega) + (b-a)\, \ell_n)\, E_{2n-1}(f; \mathcal{R}) \\ &\approx \|f\|_{\infty, \Omega_0 \triangle \Omega}\, \ell(\partial \Omega_0)\, \mathcal{O}(H^4) + \; (\mu(\Omega_0) + (b-a)\, \ell(\partial \Omega_0))\, E_{2n-1}(f; \mathcal{R}) \;, \end{aligned} \tag{25}$$

for H sufficiently small and n sufficiently large.

Remark 3 (*location of the cubature nodes*)

It is worth stressing that, in general, the cubature nodes fall also outside the spline curvilinear polygon (in the enclosing rectangle $\mathcal{R} \supseteq \Omega$). This is the reason why f is assumed to be continuous and computable also in $\mathcal{R}$, and the error estimate (15) involves the best uniform polynomial approximation on $\mathcal{R}$.

With a certain class of geometric figures, however, a change of coordinates which we describe below, can ensure that the cubature nodes are all inside the domain (i.e., the spline curvilinear polygon Ω), and in these cases it is not required that f has an extension outside Ω preserving its regularity. This class is characterized geometrically by the existence of a "base-line" (say ℓ), whose intersection with the domain is connected, and such that in addition each line orthogonal to it (say q) has a connected intersection (if any) with the domain, containing the point $\ell \cap q$. Such class contains for example all convex spline curvilinear polygons, by choosing the line connecting a pair of boundary points with maximal distance (a not easy problem to solve). But it contains also nonconvex domains, see Fig. 1 and Fig. 2 bottom-left and top-right. In practice, it could be important to use splines with "shape-preserving" properties (see, e.g., [15] and references therein).

The change of variables in the integration consists then simply in a rotation of the co-ordinate system such that the base-line becomes parallel to the (new) y-axis, choosing as ξ the abscissa of its intersection with the (new) x-axis. The resulting cubature nodes fall then necessarily inside the domain. Moreover, it is also easy to realize that the resulting cubature weights are all positive (a fact that is not true in general). This latter property was not noticed in [26], where we treated the simpler case of ordinary polygons, corresponding here to linear splines. Indeed, in (10) when the boundary point $(S_{i1}(q_{ijk}), S_{i2}(q_{ijk}))$ is on the right (resp. left) of the base-line, i.e., $S_{i1}(q_{ijk}) - \xi > 0$ (resp. $S_{i1}(q_{ijk}) - \xi < 0$), then $S'_{i2}(q_{ijk})$ is positive (resp. negative). If the spline curvilinear polygon does not fall in the class above, but approximates a domain Ω_0 in the class, then choosing a corresponding base-line we can expect that the cubature nodes lie in the neighborhood of Ω_0 (see Remark 2).

Our implementation of Gauss-Green cubature over spline curvilinear polygons (cf. [27]) accepts a pair of points, say $A = (x_A, y_A)$ and $B = (x_B, y_B)$, which define the base-line if the user can provide them, otherwise takes by default a pair of maximal distance control points. In practice, this entails only that in the construction of the nodes and weights the boundary control points $P_{ij} = (x_{ij}, y_{ij})$ and ξ have to be substituted by

$$\hat{x}_{ij} = x_{ij}\ \cos\phi + y_{ij}\ \sin\phi\ ,\ \ \hat{y}_{ij} = -x_{ij}\ \sin\phi + y_{ij}\ \cos\phi\ , \tag{26}$$

$$\hat{\xi} = x_A\ \cos\phi + y_A\ \sin\phi\ , \tag{27}$$

where $\phi = \arccos\left(|y_B - y_A|/\|B - A\|_2\right)$, $0 \le \phi \le \pi/2$, is the rotation angle.

The effect of the choice of different base-lines can be appreciated in Figure 2, where the cubature nodes on a lune (difference of two disks) are located correspondingly to the y-axis (top-left), the diagonal (top-right), a suitable horizontal line (bottom-left) and the antidiagonal (bottom-right). Since we have chosen 9 equispaced control points on each of the two circular arcs defining the boundary, we have 16 overall control points, and $N = 48n^2 + 32n$ cubature nodes at a given degree of exactness $2n - 1$; in the figure, $n = 4$ and thus there are 896 overall cubature nodes. Observe that in the bottom-left and top-right figures the nodes are all located inside the domain.

Remark 4 (*shape of the integration domain*)

Observe that the shape of the integration domain Ω can be quite arbitrary, the only restriction being that its boundary is a piecewise regular and "simple" curve in a slightly generalized sense, i.e. self-intersections are allowed only at some vertices, that become the only multiple points of the boundary path. In particular, it is worth stressing that it is *not* required that the cartesian coordinate system can be chosen such that $\Omega = \{(x, y) \in \mathbb{R}^2 : a \le x \le b\,,\ \varphi(x) \le y \le \psi(x)\}$ for some functions $\varphi \le \psi$. For example, the domain can be an entire disk, with the boundary tracked by splines.

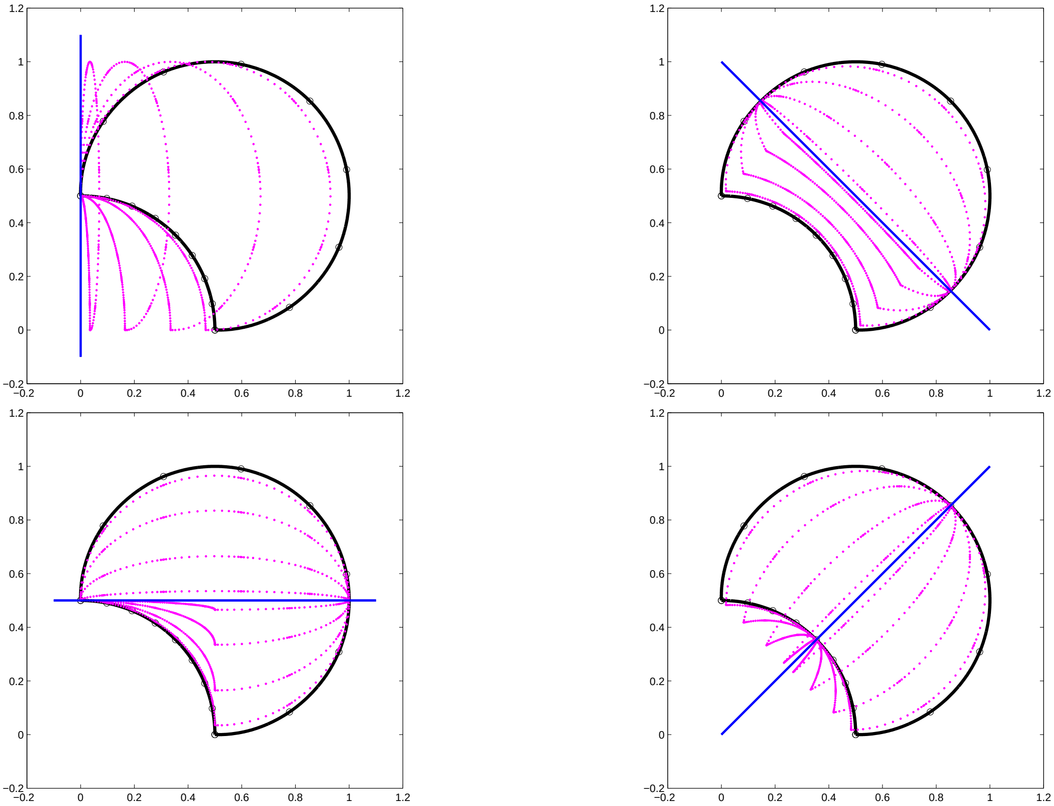

Fig. 2. Four examples of distribution of cubature points on a lune, generated by different choices of the “base-line” (algebraic degree of exactness = 7).

Moreover, the cubature formula (6) can be easily extended to multiply connected domains, via the corresponding extension of Green’s formula. Indeed, assume that the boundary of Ω be the union of an external boundary Γ^{ext} with a finite number of internal boundaries Γ_k^{int}, $k = 1, \ldots, s$ (describing holes). Then we have

$$\iint_{\Omega} f(x, y)\, dxdy = \oint_{\Gamma^{\mathrm{ext}}} \mathcal{F}(x, y)\, dy - \sum_{k=1}^{s} \oint_{\Gamma_k^{\mathrm{int}}} \mathcal{F}(x, y)\, dy \;, \tag{28}$$

where all line integrals are taken counterclockwise and can be computed by (6). We stress that in these cases the integrand f has to be continuous and computable (at least) in the convex hull, i.e. also in the holes.

3 Numerical tests.

In this section we present several numerical tests of cubature by formula (6) over two nonconvex domains with boundary tracked by spline curves. The terminology below concerning “sides”, “vertices” and “control points” refers to that adopted in Theorem 1. The cubature formula has been implemented by a Matlab code (termed `SplineGauss`, cf. [27]), which needs in input the complete sequence of control points (some of them are marked as vertices of the sides), the spline degree to be used on each side, the integrand function, and the parameter n (theoretical exactness at degree $2n-1$). In particular, Gauss-Legendre nodes and weights are computed by Gautschi's Matlab routines for orthogonal polynomials, see [10]. The user can choose to use the cumulative chordal spline parametrization as well as the equal increment one (cf. [2,29]). The interpolation additional conditions are chosen automatically, and are the classical periodic for a representation of the boundary by a single closed spline curve (one single side in our notation), or the "not-a-knot" conditions otherwise. All the tests have been done by an Intel-Centrino Duo T 2400 processor with 1 Gb RAM.

We have considered the following six bivariate test functions

$$f_1(x,y) = \frac{3}{4}\, e^{-\frac{1}{4}((9x-2)^2+(9y-2)^2)} + \frac{3}{4}\, e^{-\frac{1}{49}(9x+1)^2-\frac{1}{10}(9y+1)}$$

$$+\frac{1}{2}\, e^{-\frac{1}{4}((9x-7)^2+(9y-3)^2)} - \frac{1}{5}\, e^{-((9y-4)^2+(9y-7)^2)} \;,$$

$$f_2(x,y) = \sqrt{(x-0.5)^2+(y-0.5)^2}\;, \quad f_3(x,y) = (x+y)^{19}\;,$$

$$f_4(x,y) = e^{-((x-0.5)^2+(y-0.5)^2)}\;, \quad f_5(x,y) = e^{-100((x-0.5)^2+(y-0.5)^2)}\;,$$

$$f_6(x,y) = \cos\left(20(x+y)\right)\;, \tag{29}$$

which are in order the well-known Franke test function, the distance function from $(0.5, 0.5)$, a polynomial of degree 19, two Gaussians centered at $(0.5, 0.5)$ with different variance parameters, and a (moderately) oscillating function.

Tables 1-6 refer to numerical cubature over the nonconvex lune-like domain Ω_0 in Fig. 2, where four different choices of the base line are given.

In Table 1, we report the relative errors in computing the area of the lune, whose boundary is given by two circular arcs (the sides) and is tracked with 16, 32, 64 and 128 control points (that is 9, 17, 31 and 65 control points on each side, with two corner points that are the vertices). The exact value of the area is $1/4 + \pi/8 \approx 0.6426990816987241$. The errors correspond to three different choices of the spline degree p (the same for both sides), namely $p = 3$ (cubic), $p = 5$ (quintic), and $p = 7$, and are practically invariant with respect to the choice of the base line (for all spline curves we have used the "cumulative chordal" parametrization). Notice that the approximation improves by improving the tracking accuracy, either by increasing the number of control points or the spline degree.

The errors in Table 1 give the benchmarks for the numerical test in Table 2, which concerns integration of the Franke test function over the lune with the distribution of points in Fig. 2 top-right. Increasing the ADE (Algebraic

Degree of Exactness), we observe a stagnation of the error at a size close to the corresponding error in computing the area, as expected from estimate (25).

Table 3 shows that the stability estimate (14) is an overestimate, since very few of the weights are negative and these have small size. Indeed, the sum of the weights absolute values is practically invariant with the ADE, and in any case of the size of the domain area. When the choice of the base line guarantees that all the cubature points are inside the domain (top-right and bottom-left distributions in Fig. 2), the weights are even all positive, see Remark 3.

In Table 4 we compare the cubature errors of the Franke test function on the four distributions of points as in Fig. 2, for fixed number of control points ($m = 64$) and spline degree ($p = 7$). When the cubature points are all internal we have smaller errors at the same ADE, and all errors show a stagnation at the size of the corresponding error on the area (see Table 1). Observe that stagnation occurs at higher degrees for the choice of the base line in Fig. 2 top-left, where a lot of points fall outside the domain and the distribution is highly unsymmetric.

In Table 5 we fix again the number of control points ($m = 128$) and the spline degree ($p = 5$), and show the cubature errors of the six test functions in (29) on the lune Ω_0, at increasing ADE. The values of the integrals, with a relative error around 10^{-14}, are:

$$I_{\Omega_0}(f_1) = 0.20307626985342\ , \quad I_{\Omega_0}(f_2) = 0.20646770293563\ ,$$

$$I_{\Omega_0}(f_3) = 638.55743274702\ , \quad I_{\Omega_0}(f_4) = 0.57263720432530\ ,$$
$$I_{\Omega_0}(f_5) = 0.03137185199242\ , \quad I_{\Omega_0}(f_6) = 0.0062895812195655\ ,$$

which have been obtained representing the lune by unions and differences of sectors and squares, and using standard cubature formulas on such pieces. Observe that the absolute errors stagnate at a size compatible with the estimate (25) and the error on the area (see Table 1), for all the regular integrands f_1, f_3, f_4, f_5 and f_6 (more slowly for the peaked Gaussian f_5). We recall that in our notation Ω_0 is the lune and Ω its approximation via the spline boundary. The polynomial f_4 is integrated accurately already at low ADE, whereas the oscillating function f_6 needs an ADE compatible with the number of oscillations to push down the error. The behavior of the cubature formula is satisfactory even on the less regular function f_2, which has a singularity of the gradient at a point "in the middle" of the domain (where the cubature points cluster slowly).

In order to have a comparison with the performance of a standard integrator, in Table 6 we show the number of cubature points used by the Matlab `dblquad` automatic integrator (cf. [16]) and by our formula, to obtain an accuracy close to the best in each column of Table 5. Observe that `dblquad` is more efficient in treating internal singularities of the integrand at intermediate accuracies, like for f_2 and f_5, but at high accuracies suffers from introducing an artificial discontinuity across the boundary. In these cases the number of function evaluations required by `dblquad` can be from 5 to more than 10 times greater than that of `SplineGauss`. Moreover, bedides the considerations about the number of function evaluations at a given error tolerance, it should be recalled that `dblquad` implements adaptivity by recursion, which has a strong effect on the computing time. Indeed, the total CPU time for the examples of Tables 6 ranges from 1 to 3 seconds with `SplineGauss`, and from 9 to 360 seconds with `SplineGauss`.

Another application of the cubature formula (6) is given in Table 7, where we exploit one of its main features, that is the polynomial exactness. When it is applied to a polynomial of degree d, it is sufficient to choose $n \geq (d+1)/2$ to get the integral up to machine precision, irrespectively of the specific polynomial basis used to represent the polynomial. This flexibility could be very useful in applications where integrals of arbitrary polynomials have to be computed with very high accuracy. An important example of this kind is given by computation of polynomial moments over a planar region with complex shape, a problem arising typically in the pattern recognition and image analysis contexts. Recently, the method of tracking boundaries with spline curves has appeared in the literature on moment computation, see e.g. [13,21], but the formulas developed there seem to be restricted to monomial moments. On the other hand, the so-called orthogonal moments, i.e. moments corresponding to orthogonal polynomials, are important in many applications of pattern recognition and image analysis (see, e.g., [18,6,22,23] and references therein).

In Table 7 we report the maximum errors in the computation of the first 153 moments (up to degree 16) of three polynomial bases over an hand-shape domain Ω, whose boundary is tracked by 38 control points and a cubic spline together with a linear spline connecting the two points on the wrist. This example of tracking an hand by splines is taken from [17]. We have computed

$$I_{\Omega}(\phi_{p_1}(x)\phi_{p_2}(y)) \ , \quad 0 \leq p_1 + p_2 \leq 16 \ , \tag{30}$$

where $\phi_k(\cdot) = (\cdot)^k$ (standard monomial moments), $\phi_k(\cdot) = T_k(\cdot)$ (Chebyshev moments), and $\phi_k(\cdot) = P_k(\cdot)$ (Legendre moments); observe that $\Omega \subset [-1,1]^2$. The reference values for the moments have been computed by our formula with ADE = 21. As expected, in all cases the error jumps down abruptly close to machine precision when the ADE exceeds a threshold. This threshold turns out to be lower than the theoretical one (that is 16) for the monomial basis, but this is not surprising since, differently from the other two, the basis polynomials do not oscillate.

Here, we have restricted the attention to the computation of moments of a constant density over a spline curvilinear polygon. We stress that moment computation via (6) can be immediately extended to any bivariate continuous density function.

Table 1
Relative errors in computing the area of the lune in Fig. 2 (integration of $f \equiv 1$, any distribution of nodes): ADE is the algebraic degree of exactness, m the number of control points on the boundary, p the spline degree.

ADE	$m = 16$			$m = 32$		
	$p = 3$	$p = 5$	$p = 7$	$p = 3$	$p = 5$	$p = 7$
1	4.2E-04	8.1E-05	1.0E-05	5.8E-06	9.4E-07	3.9E-08
ADE	$m = 64$			$m = 128$		
	$p = 3$	$p = 5$	$p = 7$	$p = 3$	$p = 5$	$p = 7$
1	4.1E-07	8.1E-09	9.5E-11	5.0E-08	7.0E-11	1.4E-13

Table 2
Relative cubature errors for the Franke test function, with cubature points distributed as in Fig. 2 top-right: ADE is the algebraic degree of exactness, m the number of control points on the boundary, p the spline degree.

ADE	$m = 16$			$m = 32$		
	$p = 3$	$p = 5$	$p = 7$	$p = 3$	$p = 5$	$p = 7$
7	1.7E-03	9.8E-04	1.1E-03	1.1E-03	1.1E-03	1.1E-03
15	6.4E-04	9.3E-05	1.2E-05	1.9E-05	8.4E-07	4.1E-07
23	6.4E-04	9.4E-05	1.2E-05	1.9E-05	1.2E-06	5.2E-08
31	6.4E-04	9.4E-05	1.2E-05	1.9E-05	1.2E-06	5.2E-08
39	6.4E-04	1.0E-04	1.2E-05	1.9E-05	1.2E-06	5.2E-08
ADE	$m = 64$			$m = 128$		
	$p = 3$	$p = 5$	$p = 7$	$p = 3$	$p = 5$	$p = 7$
7	1.1E-03	1.1E-03	1.1E-03	3.8E-04	1.1E-03	1.1E-03
15	5.9E-07	3.5E-07	3.6E-07	4.9E-07	3.6E-07	3.6E-07
23	2.3E-07	1.1E-08	6.9E-11	1.9E-08	1.5E-10	6.7E-11
31	2.3E-07	1.1E-08	1.3E-10	1.9E-08	9.2E-11	3.5E-13
39	2.3E-07	1.1E-08	1.3E-10	1.9E-08	9.2E-11	3.9E-13

Table 3
Invariance with the ADE of the sum of the weights absolute values (rounded to 3 digits), with the four distributions of cubature points as in Fig. 2; the area of the domain rounded to 3 digits is 0.643.

ADE	top-left	top-right	bot-left	bot-right
7	1.14	0.643	0.643	0.785
> 7	1.14	0.643	0.643	0.785

Table 4
Relative cubature errors for the Franke test function, with the four distributions of cubature points as in Fig. 2: the number of control points is $m = 64$, the spline degree is $p = 7$.

ADE	top-left	top-right	bot-left	bot-right
7	3.2E-02	1.4E-03	1.1E-03	3.8E-04
15	7.6E-04	1.2E-06	3.6E-07	4.7E-07
23	5.7E-06	8.2E-11	6.9E-11	9.2E-11
31	1.3E-08	9.2E-11	1.3E-10	9.2E-11
39	2.5E-10	9.2E-11	1.3E-10	9.2E-11
47	9.3E-11	9.2E-11	1.3E-10	9.2E-11
55	9.2E-11	9.2E-11	1.3E-10	9.2E-11

Table 5
Relative cubature errors for the six test functions in (29), with the distribution of cubature points as in Fig. 2 bottom-left: the number of control points is $m = 128$, the spline degree is $p = 5$.

ADE	f_1	f_2	f_3	f_4	f_5	f_6
3	6.7E-03	2.2E-03	5.8E-02	8.0E-05	2.6E-02	4.8E+00
5	4.0E-03	2.6E-04	2.3E-03	3.2E-07	1.1E-01	1.2E+00
7	1.1E-03	5.0E-05	4.2E-05	1.0E-09	8.9E-03	1.3E-01
9	2.0E-04	1.4E-05	3.8E-07	6.1E-11	5.3E-03	8.8E-03
11	3.1E-05	5.1E-06	1.7E-09	6.1E-11	1.1E-03	3.9E-04
13	3.7E-06	2.2E-06	5.1E-11	6.1E-11	2.8E-05	1.2E-05
15	3.6E-07	1.0E-06	4.8E-11	6.1E-11	3.2E-05	2.9E-07
17	2.2E-08	5.2E-07	4.8E-11	6.1E-11	6.3E-06	6.7E-10
19	3.1E-10	2.9E-07	4.8E-11	6.1E-11	4.4E-07	4.7E-09
21	2.3E-10	1.7E-07	4.8E-11	6.1E-11	3.4E-08	4.6E-09

Table 6
Number of cubature points used by the Matlab **`dblquad`** integrator and by our formula (`SplineGauss`), to obtain an accuracy close to the best in each column of Table 5.

♯ pts.	f_1	f_2	f_3	f_4	f_5	f_6
`dblquad`	324186	32740	282738	225950	20782	1007426
`SplineGauss`	67840	67840	34048	17920	81664	55296

Table 7
Max errors in the computation of the first 153 moments (up to degree 16) of three polynomial bases over the hand-shape domain in Fig. 3, with horizontal base line (left column) and vertical base line (right column).

ADE	monomial		Chebyshev		Legendre	
5	2.1E-06	7.4E-10	8.6E-02	1.2E-02	2.1E-02	3.0E-03
9	1.1E-09	1.2E-14	6.5E-02	3.4E-03	1.3E-02	1.1E-03
13	3.1E-14	2.8E-16	3.3E-02	1.0E-04	8.3E-03	2.8E-05
17	2.2E-16	1.7E-16	3.4E-16	1.5E-16	2.2E-16	8.8E-17

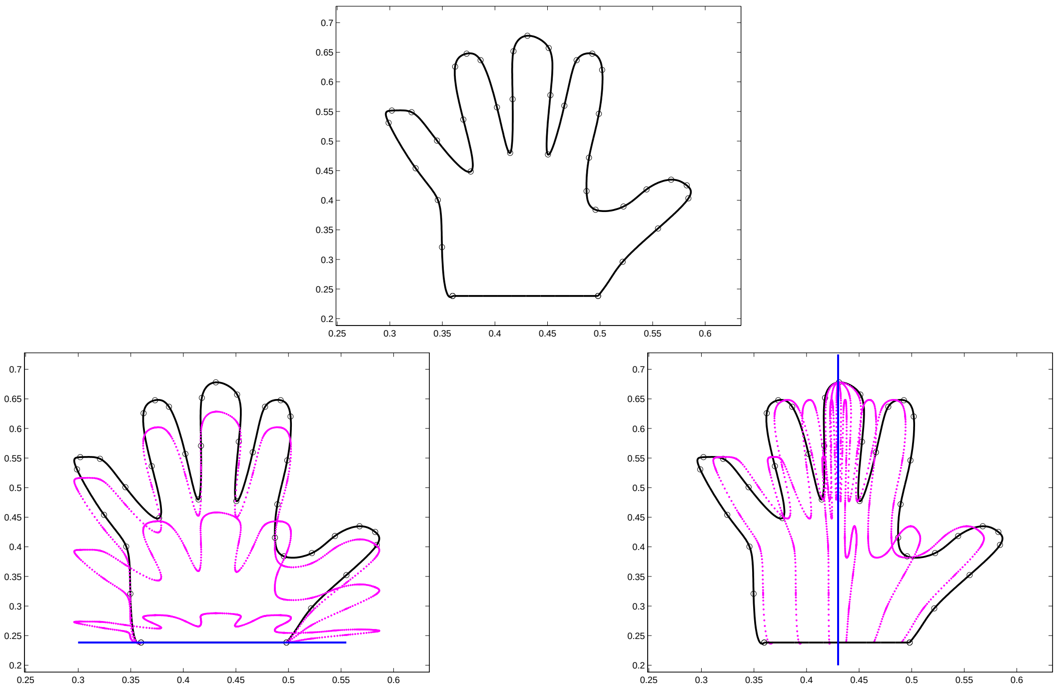

Fig. 3. Two examples (bottom) of distribution of cubature points for an hand-shape domain, generated by different choices of the “base-line” (algebraic degree of exactness = 5). The boundary is tracked by 38 control points and a cubic spline curve together with a linear spline connecting the two points on the wrist.

References

[1] T.M. Apostol, Calculus, vol. II, 2nd edition, Blaisdell, 1969.

[2] C. de Boor, A practical guide to splines, revised edition, Applied Mathematical Sciences, 27, Springer, New York, 2001.

[3] C. de Boor, Spline Toolbox 3 User's Guide, in: The Mathworks, MATLAB documentation set, 2007 version (available online at http://www.mathworks.com).

[4] R. Cools, D. Laurie & L. Pluym, Cubpack++: A C++ package for automatic two-dimensional cubature, ACM Trans. Math. Software 23 (1997), 1–15.

[5] M.P. Epstein, On the influence of parameterization in parametric interpolation, SIAM J. Numer. Anal. (13) 1976, 261–268.

[6] B. Fu, J. Zhou, Y. Li, G. Zhang, C. Wang, Image analysis by modified Legendre moments, Pattern Recognition 40 (2007), 691–704.

[7] C. F. Gauss, Theoria attractionis corporum sphaeroidicorum ellipticorum homogeneorum methodo novo tractate, Commentationes Societatis Regiae Scientiarum Gottingensis Recentiores, Vol. III, 1813.

[8] C.F. Gauss, Methodus nova integralium valores per approximationem inveniendi, Commentationes Societatis Regiae Scientiarum Gottingensis Recentiores, Vol. III, 1814.

[9] W. Gautschi, Numerical Analysis: An Introduction, Birkhäuser, Boston, 1997.

[10] W. Gautschi, Orthogonal polynomials: computation and approximation, Oxford University Press, Oxford, 2004 (software available at http://www.cs.purdue.edu/archives/2002/wxg/codes).

[11] W. Gautschi, Orthogonal polynomials (in Matlab), J. Comput. Appl. Math. 178 (2005), 215–234.

[12] G. Green, An Essay on the Application of Mathematical Analysis to the Theories of Electricity and Magnetism, Nottingham, 1828.

[13] M. Jacob, T. Blu & M. Unser, An Exact method for Computing the Area Moments of Wavelet and Spline Curves, IEEE Trans. on Pattern Analysis and Machine Intelligence 23 (2001), 633–642.

[14] V.I. Krylov, Approximate Calculation of Integrals, The Macmillan Co., New York-London, 1962.

[15] B.I. Kvasov, Methods of shape-preserving spline approximation, World Scientific Publishing Co., Inc., River Edge, NJ, 2000.

[16] The MathWorks, MATLAB documentation set, 2007 version (available online at http://www.mathworks.com).

[17] C.B. Moler, Numerical computing with Matlab, SIAM, Philadelphia, 2004.

[18] R. Mukundan & K. Ramakrishnan, Moment functions in image analysis. Theory and applications, World Scientific Publishing Co., Inc., River Edge, NJ, 1998.

[19] W. Pleśniak, Remarks on Jackson's theorem in $\mathbf{R}^N$, East J. Approx. 2 (1996), no. 3, 301–308.

[20] M. Sakai & R.A. Usmani, On orders of approximation of plane curves by parametric cubic splines, BIT Numerical Mathematics 30 (1990), 735–741.

[21] S. Sheynin & A. Tuzikov, Moment Computation for Objects with Spline Curve Boundary, IEEE Transactions on Pattern Analysis and Machine Intelligence 25 (2003), 1317–1322.

[22] H.Z. Shu, L.M. Luo, W.X. Xu & J.D. Zhou, Fast computation of Legendre moments of polyhedra, Pattern Recognition 34 (2001), 1119–1126.

[23] H.Z. Shu, J. Zhou, G.N. Han, L.M. Luo & J.L. Coatrieux, Image reconstruction from limited range projections using orthogonal moments, Pattern Recognition 40 (2007), 670–680.

[24] A. Sommariva & M. Vianello, Meshless cubature by Green's formula, Appl. Math. Comput. 183 (2006), 1098–1107.

[25] A. Sommariva & M. Vianello, Meshless cubature over the disk by Thin-Plate Splines, J. Comput Appl. Math. 221 (2008), 284–292.

[26] A. Sommariva & M. Vianello, Product Gauss cubature over polygons based on Green's integration formula, BIT Numerical Mathematics 47 (2007), 441–453.

[27] A. Sommariva & M. Vianello, `SplineGauss`: a Matlab code for Gauss-Green cubature over spline curvilinear polygons, software downloadable from: www.math.unipd.it/∼marcov/software.html.

[28] A.H. Stroud, Approximate calculation of multiple integrals, Prentice-Hall Series in Automatic Computation, Prentice-Hall, Inc., Englewood Cliffs, N.J., 1971.

[29] B. Su & D. Liu, Computational Geometry: Curve and Surface Modeling, Academic Press, New York, 1989.