Low cardinality Positive Interior cubature on NURBS-shaped domains

Alvise Sommariva and Marco Vianello

Università degli Studi di Padova

Software for Approximation 2022

February 3, 2022

Purpose

We present an algorithm that computes an algebraic cubature rule

$$\int_{\mathcal{S}} f(x,y)dxdy \approx \sum_{j=1}^{\eta} w_j f(Q_j)$$

over curvilinear polygons $\mathcal{S}$ defined by piecewise rational functions, that

- for n fixed, is exact for any $p \in \mathbb{P}_n$, being $\mathbb{P}_n$ the space of bivariate polynomials of total degree n (i.e. ADE=n);
- has positive weights $\{w_j\}_j$ and interior nodes $\{Q_j\}_j \subseteq \mathcal{S}$;
- has low cardinality, i.e. $\eta \leq (n+1)(n+2)/2$ nodes.

Examples are domains $\mathcal{S}$ such that $\partial\mathcal{S}$ is defined piecewise by

- NURBS curves,
- by composite Bezier curves,
- parametric splines.

Purpose

Key tools:

- overlooked theorem by Wilhelmsen (1976) on Tchakaloff sets,
  (sufficiently dense set on $\mathcal{S}$ contains nodes of an algebraic rule of PI-type with ADE$=n$),
- a new in-domain algorithm for such curvilinear polygons,
  (before available only on parametric spline curvilinear polygons or basic $\mathcal{S}$),
- the sparse nonnegative solution of underdetermined moment matching systems by the Lawson-Hanson NonNegative Least Squares solver,
  (extracts nodes and determines positive weights from the dense pointset and moments of a basis of $\mathbb{P}_n$).

Applications:

- NEFEM with NURBS-shaped curvilinear elements,
- VEM with NURBS-shaped curvilinear elements.

Examples of integration domains

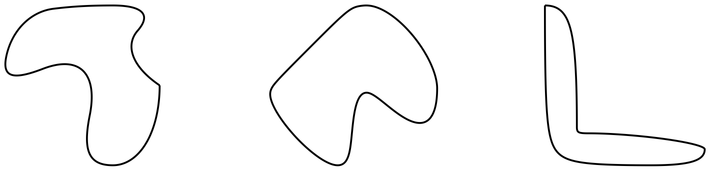

Figure: Examples of integration domains.

In-domain routine for rational spline curvilinear polygons

Assumptions:

the curvilinear polygon $\mathcal{S} \subset \mathbb{R}^2$ is a Jordan domain

(hence the domain has no holes and the boundary has no self-intersections);

- whose boundary $\partial\mathcal{S}$ is described by parametric equations $x = \tilde{x}(t)$, $y = \tilde{y}(t)$, $t \in [a, b]$, $\tilde{x}, \tilde{y} \in C([a, b])$, $\tilde{x}(a) = \tilde{x}(b)$ and $\tilde{y}(a) = \tilde{y}(b)$;

  (the boundary is described parametrically by two periodic continuous functions);
- for which there are partitions $\{I^{(k)}\}$, $k = 1, ..., M$ of $[a, b]$, and $\{I_j^{(k)}\}$ with $j = 1, ..., m_k$ of each $I^{(k)} := [t(k), t(k+1)]$, such that the restrictions of $\tilde{x}$, $\tilde{y}$ on each closed interval $I^{(k)}$ are rational splines, w.r.t. the subintervals $\{I_j^{(k)}\}$,

  (the boundary is described parametrically by M rational splines).

Example I: composite Bezier closed curves

I: composite Bezier closed curves:

- for specific points $\{\mathbf{P}_{i,k}\}_{1,\ldots,m_k} \subset \mathbb{R}^2$ choosen by the user;
- defined the *Bernstein polynomials*

$$b_{i,l}(t) = \left(\begin{array}{c} l \\ i \end{array} \right) t^i (1-t)^{l-i}, \;\; i = 0, \ldots, l-1, \quad t \in [0,1];$$

the k-th curve is of the form

$$\mathcal{B}(\tilde{t}) = \mathcal{B}(\omega_k(t)) = \sum_{i=0}^{m_k - 1} b_{i,m_k-1}(t) \mathbf{P}_{i+1,k},$$

where

$$\tilde{t} = \frac{t^{(k+1)} + t^{(k)}}{2} + \frac{t^{(k+1)} - t^{(k)}}{2} t := \omega_k(t), t \in [0,1],$$

(the boundary is described parametrically by continuous functions that are specific piecewise polynomials, often used in computer graphics).

Example II: NURBS

II: NURBS:

domains $\mathcal{S}$ in which $\partial\mathcal{S}$ is locally a p-th degree NURBS curve, i.e. defined in the curvilinear side $V_k \frown V_{k+1}$ as

$$\mathbf{C}(t) = \frac{\sum_{i=1}^{m_k} B_{i,p}(t)\, w_i \mathbf{P}_{i,k}}{\sum_{i=1}^{m_k} B_{i,p}(t)\, w_i}, \quad t \in [t^{(k)}, t^{(k+1)}]$$

where

- $\{\mathbf{P}_{i,k}\}_{i=1}^{m_k}$ are the control points, $\{\mathbf{w}_{i,k}\}_{i=1}^{m_k}$ are the weights,
- $\{B_{i,p}\}_{i=1}^{m_k}$ are suitable p-th degree B-spline basis functions defined on the nonperiodic (and nonuniform) knot vector

$$U = \{\underbrace{t^{(k)}, \ldots, t^{(k)}}_{p+1}, t^{(k)}_{p+1}, \ldots, t^{(k)}_{m_k-(p+1)}, \underbrace{t^{(k+1)}, \ldots, t^{(k+1)}}_{p+1}\}.$$

with $t^{(k)}_{p+j} \leq t^{(k)}_{p+j+1}$, $j = 1, \ldots, m_k - 1$,

(the boundary $\partial\mathcal{S}$ is described parametrically by continuous functions that are specific piecewise rational functions, often used in computer graphics).

In-domain algorithm: Jordan curve theorem

Jordan curve theorem:
a point P belongs to a Jordan domain $\mathcal{S}$ if and only if, having taken a point $P^* \notin \mathcal{S}$ then the segment $\overline{P^*P}$ crosses $\partial\mathcal{S}$ an odd number $c(P)$ of times.

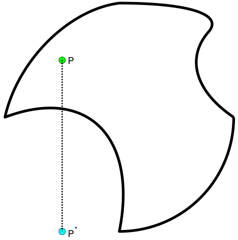

Figure: Points and boundary intersections. On the left $c(P) = 1$ and the point P is in the domain. On the right $c(P) = 2$ and the point P is outside the domain.

In-domain algorithm: Pathological cases

Pathological cases:

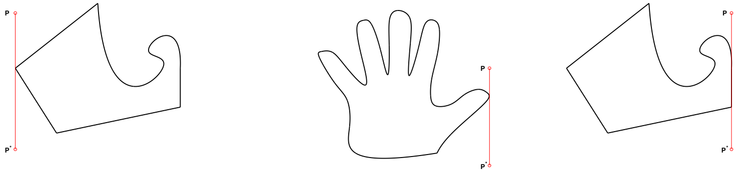

Figure: Critical situations for establishing the crossing number on curvilinear polygons.

In-domain algorithm: implementation

Basic idea:

- Cover the boundary $\partial\mathcal{S}$ by rectangles, with sides parallel to the axes, so that $x = \tilde{x}(t)$ and $y = \tilde{y}(t)$ are monotone (we will name them monotone boxes). Thus, the boundary is the graph of a local monotone Cartesian function in x and y.
- For each point P that is not in a pathological case, count the $c_0(P)$ monotone boxes *strictly below* P.
- If a point is inside some monotone boxes, count all the $c_1(P)$ times that is *over* the part of the boundary belonging to the box.
- Put $c(P) = c_0(P) + c_1(P)$. If $c(P)$ is odd then P is inside $\mathcal{S}$, otherwise it is not inside the curvilinear domain.
- For pathological cases, use alternative techniques, see [1].

In-domain algorithm: examples I

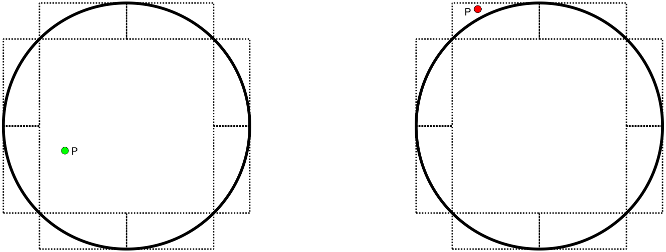

Figure: Monotone boxes and computation of the crossing number $c(P)$ when $\mathcal{P}$ is a disk. On the left figure, $c(P) = 1$, on the right one $c(P) = 2$.

In-domain algorithm: examples II

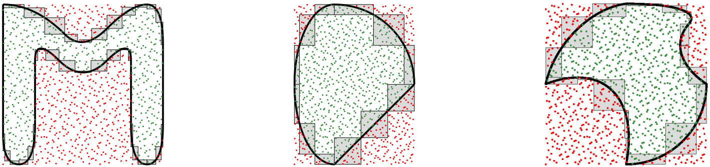

Figure: Monotone boxes and detection of the points inside the domain.

In-domain algorithm: main difficulties

1. Fast determination of the monotone boxes, from the piecewise rational splines $\tilde{x}$, $\tilde{y}$ (pre-processing);
2. analysis of the pathological points;
3. for each point P, fast determination of the monotone boxes necessary to the computation of $c(P)$;
4. fast determination if a point P belonging to a monotone box is over or below the curve relative to the monotone box;
5. deciding when a point P close to the boundary $\partial\mathcal{S}$ is *numerically inside* $\mathcal{S}$ or not.
6. for cubature purposes, we must be able to analyse 1000 points in less than 10^{-3} seconds, including the pre-processing cputime.

Remark

The fact that the boundary is described parametrically by piecewise rational splines is fundamental in items 1, 4, 5.

Moments computation

Having in mind to compute a rule with algebraic degree of precision $ADE = n$ by moments equations, we

- define a suitable basis $\{\phi_j\}$ of the polynomial space $\mathbb{P}_n$ (*tensorial Chebyshev* on the bounding box $\mathcal{R}^*$ of $\mathcal{S}$),
- compute the moments $\gamma_1, \ldots, \gamma_N$, where

$$\gamma_j := \int_{\mathcal{S}} \phi_j(x, y) dxdy, .$$

To this purpose:

1. By applying the Gauss-Green theorem, each γ_j is the sum of some line integrals, that after some computation are shown to require the integration in $[-1, 1]$ of continuous rational functions.
2. We compute these integrals in $[-1, 1]$ by high-order Gauss-Legendre rule (other techniques may be used).

Implementing Tchakaloff-like algebraic cubature rules

We extract the nodes and positive weights of a Tchakaloff-like algebraic cubature rule (i.e. a rule with ADE$=n$, positive weights, and cardinality at most equal to the dimension of $\mathbb{P}_n$, i.e. $N = (n+1)(n+2)/2$), by the following algorithm:

- compute the moments $\gamma = (\gamma_j)$ of a suitable basis of $\mathbb{P}_n$;

at the k-th iteration of the algorithm

- introduce a tensorial grid $\mathcal{M}_\ell$ in the rectangle $\mathcal{R}^* := [a_1, b_1] \times [a_2, b_2]$ containing $\mathcal{S}$;
- determine by the *in-domain* algorithm, at the ℓ-th iteration of the procedure, the set

$$\mathcal{P}_\ell = \mathcal{P}_{\ell-1} \cup (\mathcal{M}_\ell \cap \mathcal{S})$$

(the points of the analysed meshes, as well as of the present one, belong to $\mathcal{S}$);

- compute the Vandermonde matrix $V_{\mathcal{P}_\ell} = (\phi_j(\mathcal{P}_i^{(\ell)}))_{i,j}$ (relatively to the basis $\{\phi_j\}$ of $\mathbb{P}_n$ and the pointset $\mathcal{P}_\ell = \{\mathcal{P}_i^{(\ell)}\}$);

Implementing Tchakaloff-like algebraic cubature rules

- apply the Lawson-Hanson algorithm to attempt to find a solution $w^* \geq 0$ to the overdetermined linear system $V_{\mathcal{P}_\ell} w = \gamma$
  (any solution provides an alg. rule with pos. weights, internal nodes, $ADE = n$);
- find the nonnull components of w^*, say $\{w_i^{(\ell)}\}_{i=1,\ldots,\nu_\ell}$;
- determine the corresponding nodes $\{(x_i^{(\ell)}, y_i^{(\ell)})\}_{i=1,\ldots,\nu_\ell}$, $\nu_\ell \leq N$
  (if $w_l^* > 0$ then $(x_l^{(\ell)}, y_l^{(\ell)})$ is the relative node);
- for a fixed tolerance ε, check if the so obtained rule is such that

$$\gamma_j^{(\ell)} = \sum_{i=1}^{\nu_\ell} w_i^{(\ell)} \phi_j(x_i^{(\ell)}, y_i^{(\ell)}),\ j = 1, \ldots, N\ ,$$

  well approximates the set of moments $\gamma = \{\gamma_j\}$, i.e.

$$\|\gamma^{(\ell)} - \gamma\|_2 \leq \varepsilon \qquad (1)$$

  (the cubature rule numerically matches the moments at $ADE = n$);
- if (1) does not hold, iterate the procedure.

Implementing Tchakaloff-like algebraic cubature rules

Comment:

- The basic idea is to fill the domain $\mathcal{S}$ with points until one is able to determine the wanted ruled by Lawson-Hanson algorithm [5].
- Important: Lawson-Hanson will find a solution with at most $(n+1)(n+2)/2$ positive components!
- In VEM, the algebraic degree of precision $ADE = n$ is typically low, say $n \leq 5$, and all the process must take at most 10^{-2} seconds (many MATLAB tricks!).
- By construction the rules have internal nodes and positive weights.
- In exact arithmetic this procedure has finite termination in view of a theorem by Wilhelmsen mentioned above [5], since the set $\mathcal{P}_\ell$ becomes sufficiently dense after a finite number of iterations.
- Many technical details are skipped and can be found in [1].

Numerical examples: cubature

We have implemented in Matlab the ideas sketched above (cf.[3]).

In order to show the flexibility of our method, we consider the domains that are in the next figure from left to right,

1. a "M" shaped domain $\mathcal{S}_1$, in which $\partial\mathcal{S}_1$ is determined by a unique order 3 NURBS curve with 16 distinct control points,
2. a convex domain $\mathcal{S}_2$, where $\partial\mathcal{S}_2$ is obtained by joining a circular and an elliptical arc, followed by a segment,
3. a concave domain $\mathcal{S}_3$ whose boundary $\partial\mathcal{S}_3$ consists of a unique NURBS curve of order 3 with 9 distinct control points.

Numerical examples: cubature

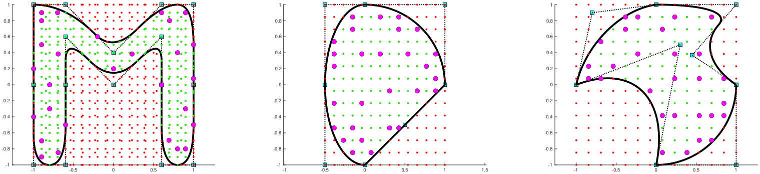

Figure: The curvilinear domains $\mathcal{S}_i$ with $i = 1, 2, 3$, the grid points P outside the domain or on its boundary (in red), those inside the domain (in green) and the nodes of a cubature formula of PI-type for $n = 6$ (28 magenta dots). The control points of the NURBS curve are represented as cyan squares, joined to represent the so called *control points polygon*.

Numerical examples: cubature

	n	#	# trial pts	cond	moment res	cpu
$\mathcal{S}_1$	2	6	28 (121)	1	$5e{-}16$	$1.3e{-}2$
	4	15	108 (377)	1	$1e{-}15$	$1.8e{-}2$
	6	28	225 (637)	1	$1e{-}15$	$2.2e{-}2$
	8	45	693 (1573)	1	$3e{-}15$	$3.4e{-}2$
	10	66	1304 (3077)	1	$5e{-}15$	$8.5e{-}2$
$\mathcal{S}_2$	2	6	65 (121)	1	$8e{-}16$	$4.8e{-}3$
	4	15	65 (121)	1	$2e{-}15$	$4.8e{-}3$
	6	28	109 (196)	1	$2e{-}15$	$6.6e{-}3$
	8	45	274 (484)	1	$2e{-}15$	$9.0e{-}3$
	10	66	609 (961)	1	$3e{-}15$	$1.5e{-}2$
$\mathcal{S}_3$	2	6	50 (121)	1	$5e{-}16$	$5.3e{-}3$
	4	15	50 (121)	1	$7e{-}16$	$6.1e{-}3$
	6	28	89 (196)	1	$1e{-}15$	$7.6e{-}3$
	8	45	239 (484)	1	$2e{-}15$	$1.1e{-}2$
	10	66	491 (961)	1	$4e{-}15$	$1.6e{-}2$

Table: Degree of precision $n = 2, 4, 6, 8, 10$ of the rule, cardinality # of the extracted nodes, cubature conditioning and moment residual of the rule on domains $\mathcal{S}_i$, $i = 1, 2, 3$, number of trial points used in the extraction, cubature condition number cond, moment residual of the rule and median of the cputime over 50 tests.

Numerical examples: cubature

As a further illustration, we report in the next Table the relative errors made by the Tchakaloff-like rules when approximating $\int_{\mathcal{S}_i} f_k(x,y)\,dx\,dy$, where

$$\begin{aligned} f_1(x,y) &= \exp(-(x^2+y^2)), \\ f_2(x,y) &= ((x-x_0)^2+(y-y_0)^2)^{11/2}\,,\ (x_0,y_0)=(0,0.4), \\ f_3(x,y) &= ((x-x_0)^2+(y-y_0)^2)^{1/2}\,,\ (x_0,y_0)=(0,0.4), \end{aligned}$$

- The functions f_k, $k=1,2,3$ are examples of functions with different degree of regularity on each domain $\mathcal{S}_i$, $i=1,2,3$.
- The reference values of these integrals are those obtained by the same routines with $ADE=20$.
- As expected, in both the domains the quality of the approximation worsens for less regular integrands (indeed $f_1 \in C^{\infty}(\mathcal{S}_i)$, whereas $(0,0.4) \in \mathcal{S}_i$ is a singular point for the first derivatives of f_3 and for 6-th derivatives of f_2).

Numerical examples: cubature

	$\mathcal{S}_3$			$\mathcal{S}_4$			$\mathcal{S}_5$		
ADE	f_1	f_2	f_3	f_1	f_2	f_3	f_1	f_2	f_3
2	$2e{-}02$	$4e{-}01$	$4e{-}02$	$4e{-}03$	$9e{-}01$	$6e{-}02$	$6e{-}03$	$2e{-}01$	$1e{-}02$
4	$3e{-}03$	$2e{-}01$	$9e{-}02$	$3e{-}04$	$2e{-}02$	$4e{-}02$	$9e{-}04$	$2e{-}01$	$1e{-}02$
6	$3e{-}04$	$4e{-}02$	$6e{-}03$	$4e{-}05$	$3e{-}02$	$2e{-}02$	$4e{-}05$	$1e{-}02$	$4e{-}03$
8	$3e{-}05$	$3e{-}03$	$2e{-}03$	$1e{-}06$	$8e{-}04$	$1e{-}03$	$2e{-}06$	$2e{-}03$	$3e{-}03$
10	$1e{-}06$	$8e{-}05$	$1e{-}03$	$8e{-}09$	$4e{-}05$	$2e{-}04$	$8e{-}08$	$3e{-}05$	$2e{-}04$

Table: Relative errors of the new rules on the domains $\mathcal{S}_i$, $i = 1, 2, 3$ with $ADE = 2, 4, 6, 8, 10$.

Numerical examples: indomain

#	algorithm	$\mathcal{S}_1$	$\mathcal{S}_2$	$\mathcal{S}_3$
10^3	*inRS*1	$4.3e{-}03s$	$1.9e{-}03s$	$2.1e{-}03s$
	*inRS*2	$2.8e{-}03s$	$1.3e{-}03s$	$1.3e{-}03s$
	speed-up	1.5	1.5	1.6
10^4	*inRS*1	$1.8e{-}02s$	$8.8e{-}03s$	$9.5e{-}03s$
	*inRS*2	$4.4e{-}03s$	$3.2e{-}03s$	$3.2e{-}03s$
	speed-up	4.1	2.8	3.0
10^5	*inRS*1	$1.6e{-}01s$	$7.3e{-}02s$	$8.0e{-}02s$
	*inRS*2	$1.7e{-}02s$	$2.1e{-}02s$	$2.0e{-}02s$
	speed-up	9.4	3.5	4.0

Table: The indomain algorithm named `inRS1` proposed in [1] has been improved in [2] by `inRS2`. In this table we list the CPU time of these routines on the three NURBS-shaped domains $\mathcal{S}_i$, $i = 1, 2, 3$, with # Halton points of the corresponding bounding box.

MATLAB Software

- All the MATLAB routines and demos are collected in the toolbox `CUB_RS` and can be downloaded at [3].
- We are not aware of the existence of an official built-in NURBS toolbox (though it can be retrieved by third-parties and MATLAB has a specific environment for rational splines). Thus we have implemented a set of routines to describe $\partial\mathcal{S} \subset \mathbb{R}^2$ by piecewise rational splines, including parametric splines or composite Bezier curves or NURBS.
- Finally we provide the routines
  - `inRS` that implement the faster in-domain algorithm introduced in [2] (of interest also in meshless methods),
  - `cubRS` that computes a PI-type Tchakaloff-like algebraic cubature rule of degree n,

  for the designed domain $\mathcal{S}$.

Future research

- Fast filling of the domain;
- generalisation to domains that are not simply connected;
- application to PDE problems with VEM and NEFEM;
- application to meshless methods;
- 3D instances (very difficult task!).

Bibliography

1. A. Sommariva and M. Vianello, Low cardinality Positive Interior cubature on NURBS-shaped domains (submitted).
   (Cubature and first indomain routine)
2. A. Sommariva and M. Vianello, inRS: implementing the indicator function for NURBS-shaped planar domains (submitted).
   (Faster indomain routine)
3. Matlab software: A. Sommariva homepage, `https://www.math.unipd.it/~alvise/software.html`.
4. A. Sommariva, M. Vianello, Compression of multivariate discrete measures and applications, *Numer. Funct. Anal. Optim.*, 36 (2015), 1198–1223.
   (Details on cubature compression)
5. D. R. Wilhelmsen, A Nearest Point Algorithm for Convex Polyhedral Cones and Applications to Positive Linear approximation, *Math. Comp.*, 30 (1976), 48-57.