# A new quasi-Monte Carlo technique based on nonnegative least squares and approximate Fekete points

Claudia Bittante[a], Stefano De Marchi[a,*], Giacomo Elefante[a]

[a] *University of Padova, Department of Mathematics, Via Trieste, 63, I-35121 Padova*

**Abstract**

The computation of integrals in higher dimensions and on general domains, when no explicit cubature rules are known, can be "easily" addressed by means of the quasi-Monte Carlo method. The method, simple in its formulation, becomes computationally inefficient when the space dimension is growing and the integration domain is particularly complex. In this paper we present two new approaches to the quasi-Monte Carlo method for cubature based on *nonnegative least squares* and *approximate Fekete points*. The main idea is to use less points and especially *good points* for solving the system of the moments. *Good points* are here intended as points with good interpolation properties, due to the strict connection between interpolation and cubature. Numerical experiments show that, in average, just a tenth of the points should be used mantaining the same approximation order of the quasi-Monte Carlo method. The method has been satisfactory applied to 2 and 3-dimensional problems on quite complex domains.

*Keywords:* cubature, quasi-Monte Carlo method, nonnegative least squares, approximate Fekete points.
*2010 MSC:* 11K45, 41A45, 65D30, 65D32.

## 1. Introduction

Consider the problem of calculating the integral $I(f) = \int_\Omega f(x)dx$, $\Omega \subset \mathbb{R}^d$. We know that if $\lambda_d(\Omega) < \infty$ (the $d$ dimensional Lebesgue measure of $\Omega$)

*Corresponding author
*Email addresses:* cbittant@math.unipd.it (Claudia Bittante), demarchi@math.unipd.it (Stefano De Marchi ), elefante.giacomo@gmail.com (Giacomo Elefante)

we can turn $\Omega$ into a probability space with probability measure $\mathrm{d}\mu(x) = \frac{1}{\lambda_d(\Omega)}\mathrm{d}x$. Then for $f \in L^1(\mu)$ we have

$$I(f) = \int_\Omega f(x)\mathrm{d}x = \lambda_d(\Omega)\int_\Omega f\mathrm{d}\mu(x) = \lambda_d(\Omega)E(f)$$

where $E(f)$ is the expected value of $f$.

The *Monte Carlo* (MC) method for numerical integration is obtained by taking $N$ independent $\mu$-distributed random samples $x_1, \ldots, x_N \in \Omega$ and, then approximating the integral as follows

$$I(f) \approx \lambda_d(\Omega)\frac{1}{N}\sum_{i=1}^{N} f(x_i) = I_N(f). \tag{1}$$

For the strong law of large numbers, as $N \to \infty$ we then know that the r.h.s. in (1) converges in the Lebesgue measure to the value of the integral.

Differently to the classical Monte Carlo method or Monte Carlo integration, which is based on sequences of pseudo-random numbers, a *quasi-Monte Carlo* (qMC) method is a method for numerical integration that uses the so-called *low-discrepancy sequences* (also known as *quasi-random sequences* or sub-random sequences). Well known low discrepancy sequences are *Halton* (also known as *Van de Corput-Halton*), *Hammersley* and the so-called $(t, s)$-sequences, such as the *Sobol* sequence. For the definition and properties of all these sequences we invite interested readers to refer to the book [15].

When dealing with a (quasi-)Monte Carlo method of integration we need to find a large number $N$ of points in order to approximate the value of the integral. This means a lot of flops and storage, which become unpractical when the space dimension $d$ grows. How can we avoid this?

In the paper we propose two techniques aimed to reduce the number of quasi-random nodes, while still keeping the same accuracy of the quasi-Monte Carlo approach. The new approaches are *"compressed cubature"* that we then apply to quite general domains in space dimensions $d = 2$, 3.

In the next section we introduce some useful results on low-discrepancy sequences and their use in cubature, then in Section 3 we describe the new approaches based on *nonnegative least squares* (NNLS) and *approximate Fekete points* (AFP). In Section 4 we provide an error analysis for the NNLS case that can be adapted to the case of the AFP when the *measure of stability*, $\rho$ (cf. formula (10)), is not too big. In Section 5 we present some numerical tests supporting the validity of our new approaches. We also point

out that all the algorithms have been implemented in Matlab for consistency with previous works done by collaborators at the CAA-research group for the construction of cubature formulas on various 2 and 3-dimensional domains (cf. e.g. [27, 30]). To conclude the Introduction, we observe that our approaches can be extended to any space dimension. The reason why we have confined ourselves to $d = 2$, 3 is mainly due to hardware limitations on which the numerical experiments have been performed.

## 2. Briefly on low discrepancy sequences

In what follows let $\Omega = [0,\, 1]^d$ be the $d$-dimensional unit cube, $f : \Omega \to \mathbb{R}$ a (continuous) function and $X = \{x_1, \ldots, x_N\}$ be a finite set of points on $\Omega$.

The *discrepancy* $D_N$ of $X$ is

$$D_N(X) := \sup_{B \in J} \left| \frac{\#(B, X)}{N} - \lambda_d(B) \right|$$

where $J$ is the family of sets of the form $\prod_{i=1}^{d} [a_i, b_i) = \{x \in \mathbb{R}^d : \, a_i \leq x_i < b_i\}$ and $\#(B, X) := \sum_{j=1}^{N} \chi_B(x_j)$ (i.e. the number of points $x_j$ falling in $B$).

The *star discrepancy* of $X$,

$$D_N^* := D_N(J^*; X)\,,$$

where $J^*$ is the family of subintervals of $\Omega$ of the form $\prod_{i=1}^{d} [0, a_i)$.

As we have already seen in the quasi-Monte Carlo method the set $X$ is chosen as a *low-discrepancy sequence*.

In Figure 1 we show the plots of 500 Halton and Sobol points on the square $[-1, 1]^2$. In Figure 2 we plot the corresponding star discrepancies in 2 and 3-dimensions for more than 6000 Halton, Sobol and Hammersley points that show the typical logaritmic decay to infinity. In fact, the exact lower bound of the star discrepancy $D_N^*$ is an open problem but it is believed that there exist some small positive constant $c_d$ such that for any point set $X$ consisting of $N$ distinct points in the $d$-dimensional unit-cube, the inequality

$$D_N^*(X) > c_d \frac{(\log N)^{d-1}}{N}\,,$$

holds [15]. To generate such sequences on the unit cube one can use the Matlab codes `haltonset, sobolset`, or those available at the Matlab Central

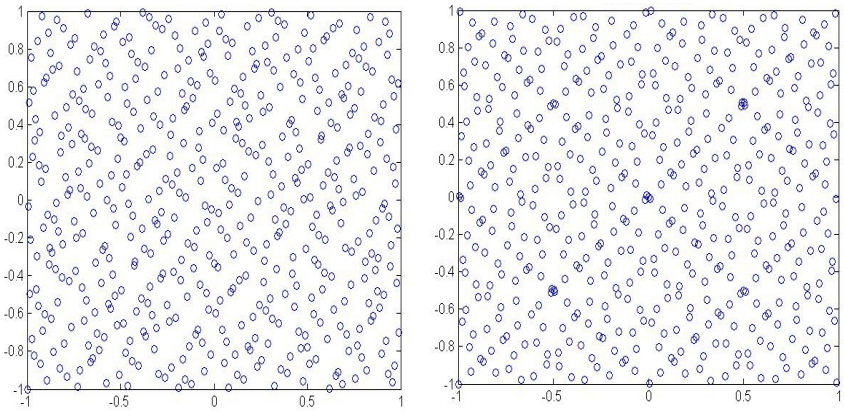

Figure 1: 500 Halton (left) and Sobol (right) points on $[-1,1]^2$

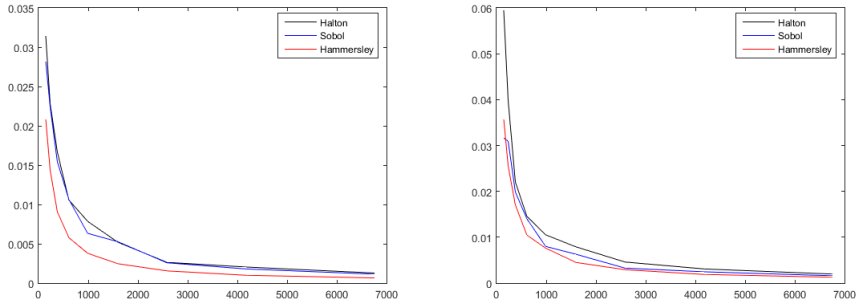


Figure 2: Star discrepancies in dimension 2 and dimension 3 for nearly 7000 points

File Exchange `http://www.mathworks.com/matlabcentral/fileexchange/`. Let $I_N(f)$ be the cubature rule given by the quasi-Monte Carlo method and

$$E_N(f) := \left| \int_\Omega f(x)dx - I_N(f) \right| , \tag{2}$$

the corresponding cubature error. If $f : \Omega \to \mathbb{R}$ is a bounded variation function, with variation $V(f)$, the *Koksma-Hlawka inequality* (cf. e.g. [24, 23]) says

$$E_N(f) \leq V(f) D_N^* . \tag{3}$$

The definition and analysis of the multidimensional bounded variation of a function, in the sense of Hardy-Krause, is well detailed in the paper [25].

Once we know $V(f)$, by the inequality (3), the quality of quasi-Monte Carlo integration rule depends on the star discrepancy. On the other hand,

the inequality (3) is not of practical use. As observed in [20] it has some problems

(i) $\mathcal{O}(N^{-1}\log^d(N))$ is smaller than $\mathcal{O}(N^{-1/2})$ when $d$ is small and $N$ is large;

(ii) for many functions $V(f)$ can be $+\infty$;

(iii) to compute $D_N$, $D_N^*$ and $V(f)$ is not always easy, while (3) gives only an upper bound.

Such drawbacks can be avoided by using a *radomized* qMC method, for example by random shift of the sequence $X$, as detailed in [34] or by the two new alternatives, presented in this article, aimed to reduce the computation efford essentially by reducing the number of points considered.

## 3. The new approaches

In the majority of real applications in which we need to approximate an integral, the domain of integration could have a quite complicated shape. For instance, in dimension $d = 2$, there exist many methods (and the corresponding algorithms) that allow to compute nodes and (positive) weights for the corresponding cubature rules. As an example for polygonal domains (convex or not convex), `polygauss` is a Matlab function that, thanks to Green's integration formula, allows to easily determine the cubature nodes and weights combining the Gauss-Legendre cubature formula with the Green's formula (cf. [27]). Other known examples for general 2-dimensional domains, with piecewise regular boundary, are those that make use again of the Gauss-Green approach and implemented in the Matlab functions `Splinegauss`, `ChebfunGauss` (cf. [30, 31]). In some of these domains, these formulas use a number of points greater or equal to the dimension of the underlying polynomial space, positive weights (for convergence reasons) and algebraic prefixed precision. Unluckily this is in general not the case! This is the reason why we are looking for cubature formulas also with some negative weights, risking instability, but gaining the possibility to work with more general domains avoiding, on the other hands, to use a lot of points as in the quasi-Monte Carlo approach.

### *3.1. Nonnegative Least Squares*

This is the purpose of the Matlab function `lsqnonneg`, based on a variant of the algorithm developed by Lawson and Hanson in [19]. Readers

interested to the use of the function `lsqnonneg` may refer to the Matlab's online documention.

NonNegative Least Squares (NNLS) problems are least squares problems that satisfy linear constraints inequalities (cf. [19, p. 161])

**Definition 1.** *Let $A$ be a $m \times n$ matrix and $b \in \mathbb{R}^m$ a column vector, $G$ a $r \times n$ matrix and $h \in \mathbb{R}^r$. A Linear System of Inequalities (LSI) problem is a least squares problem with linear constraints, that is the optimization problem*

$$\begin{aligned} \min_{x \in \mathbb{R}^n} & \|Ax - b\|_2 \\ & Gx \geq h\,. \end{aligned} \tag{4}$$

A special instance of the previous problem, used in curve fitting, consists in finding a solution with positive components.

**Definition 2.** *Let $A$ be a $m \times n$ matrix and $b \in \mathbb{R}^m$ a column vector. A NNLS problem is the LSI problem*

$$\begin{aligned} \min_{x \in \mathbb{R}^n} & \|Ax - b\|_2 \\ & x \geq 0\,. \end{aligned} \tag{5}$$

As described in [19, p. 161], the algorithm starts with a set of possible basis vectors and computes the associated dual vector, say $\lambda$. It then selects the basis vector corresponding to the maximum value in $\lambda$ in order to swap out of the basis in exchange for another possible candidate. This process continues until $\lambda_i \leq 0$, $\forall\, i$.
If in the previous definition we consider

- $A = V^T$, with $V$ the Vandermonde matrix at the sequence $X = \{x_i,\, i = 1, \ldots, n\}$ for the polynomial basis $\{p_j,\, j = 1, \ldots, m\}$;
- $b$ being the column vector of size $m$ of the moments, that is

  $$b_j = \int_\Omega p_j(x) d\mu(x)$$

  for some measure $\mu$ on $\Omega$
- $G = I$, i.e. the identity of order $n$ and $h = (0, \ldots, 0)^T$ of order $n$

then the LSI problem consists in finding the vector $x$ that minimizes $\|V^T x - b\|_2$ subject to $x \geq 0$. This will give the *nonnegative* weights $x$ for the cubature at the point set $X$. Hence, by using `lsqnonneg` we get the positive weights (given by the solution of the LSI problem) so that we can approximate the integrals at the corresponding point set $X$.

Notice that, from the Kuhn-Tucker theorem, the previous optimization problem has a solution $x$ with some components that are strictly positive and some other ones that vanish. Indeed, the residual of the solution of the NNLS problem, say

$$\epsilon = \|c - b\|_2, \;\; c = \{c_j\} = Ax^*, \;\; c_j = \sum_{k=1}^{n} w_k p_j(q_k)\,,$$

will not be zero in general (as before the $p_j$ are a polynomial basis on which we can write the solution). Then, the nodes $q_k$ and the weights $w = \{w_k\}$ are extracted correspondingly to the nonzero components of $x^*$.

*3.2. Approximate Fekete Points*

Another idea for approximating the integral, is by using the so-called *Approximate Fekete Points (AFP)* extracted from a suitable discretizion of the domain known as *Weakly Admissible Meshes (WAM)* (cf. e.g. [4, 5, 28]). For the definition and the properties of WAMs we refer to the paper [5].

The AFP are good approximation of the true Fekete points as proved in [4] and they are determined by a "simple" numerical procedure which turns out to be equivalent to the QR factorization with column pivoting of the transposed of the rectangular Vandermonde matrix associated to the approximation process.

More specifically, consider a WAM $\{\mathcal{A}_n\}$ of a compact set $K \subset \mathbb{R}^d$ (or $K \subset \mathbb{C}^d$), say $\mathcal{A}_n = \{a_1, \ldots, a_m\}$, $m \geq \nu_n = \dim(\mathbb{P}_n^d)$, and the associated *rectangular* Vandermonde-like matrix

$$V(\boldsymbol{a}; \boldsymbol{p}) = V(a_1, \ldots, a_m; p_1, \ldots, p_{\nu_n}) = [p_j(a_i)]\;, \;\; 1 \leq i \leq m\;,\; 1 \leq j \leq \nu_n\,, \tag{6}$$

where $\boldsymbol{a} = (a_i)$ is the array of mesh points, and $\boldsymbol{p} = (p_j)$ is the array of basis polynomials for $\mathbb{P}_n^d$ (both ordered in some manner). The AFP algorithm can be described in these simple Matlab-like notation

**algorithm AFP (Approximate Fekete Points):**
$W = (V(\boldsymbol{a}, \boldsymbol{p}))^t$; $\boldsymbol{b} = (1, \ldots, 1)^t \in \mathbb{C}^{\nu_n}$;
$\boldsymbol{w} = W \backslash \boldsymbol{b}$;
$ind = \texttt{find}(\boldsymbol{w} \neq \boldsymbol{0})$; $\boldsymbol{\xi} = \boldsymbol{a}(ind)$

For details about the AFP algorithm and its Matlab implementation we suggest the readers to refer to the papers [28, 4, 5]. Here we simply recall that at the web page `http://www.math.unipd.it/~marcov/CAAsoft.html` once can find all the necessary scripts for polynomial fitting and interpolation on WAMs.

Among the applications of the AFP, a *natural* one is *numerical cubature*. In fact, if in the algorithm AFP we take as right-hand side $\boldsymbol{b} = \boldsymbol{m} = \int_K \boldsymbol{p}(x)\, d\mu$ (the moments of the polynomials basis with respect to a given measure $\mu$), the vector $\boldsymbol{w}(ind)$ gives directly the weights of an algebraic cubature formula at the corresponding Approximate Fekete Points. As a remark, when the boundary of $K$ is approximated by polynomial splines, for $d\mu = dx$ the moments can be computed by the formulas developed in [32].

## 4. Error analysis

In [29, §2], the authors have provided an error analysis, estimating the effect of the moments error in integrating a function, at least when the integrand $f$ is defined on the whole domain $\Omega$. Just to give an idea on how this error analysis has been done, we consider a multivariate discrete measure $\nu$ supported at a finite set $X = \{X_i\} \subset \Omega \subset \mathbb{R}^d,\ i = 1, \ldots, N$ with correspondent (positive) weights $\omega_i$. The idea of "measure compression", which is essentially our goal, consists in computing an integral by a extracting a subset of the point set $X$ and the corresponding weights so that

$$\int_\Omega f(X) d\nu = \sum_{i=1}^{N} \omega_i f(X_i) \approx \sum_{j=1}^{M} w_j f(Y_j)\,, \tag{7}$$

where $Y = \{Y_j\} \subset X$ and $M = \texttt{card}(\{Y_j\}) \leq \texttt{dim}(\mathbb{P}_n^d) < N$ in such a way that the cubature formula is (nearly) exact on total-degree polynomials of degree $\leq n$ in $\mathbb{R}^d$.

Following [29],

$$\int_\Omega p(S) d\nu = \int_X p(S) d\nu = \langle \boldsymbol{c}, \boldsymbol{m} \rangle,\ \ \forall\, p \in \mathbb{P}_n^d$$

where the $c_j$ are the Fourier coefficients of $p$ in the orthogonal basis, say $\Phi = \{\phi_1, \ldots, \phi_M\}$ w.r.t. the $\mathrm{d}\lambda$, the measure of the domain, and $m_j$ the corresponding $\mathrm{d}\nu$-moments of the $\Phi$. Furthermore

$$\sum_{j=1}^{M} w_j p(Y_j) = \langle \boldsymbol{c}, \boldsymbol{\mu} \rangle$$

where the $\boldsymbol{\mu}$ are the approximate moments with moment error $\epsilon_{mom}$. Then, immediately we get

$$\left| \int_\Omega p(S)d\nu - \sum_{j=1}^{M} w_j p(Y_j) \right| = |\langle \boldsymbol{c}, \boldsymbol{m} - \boldsymbol{\mu} \rangle| \leq \|\boldsymbol{c}\|_2 \|\boldsymbol{m} - \boldsymbol{\mu}\|_2 \leq \|p\|_{L^2_{d\lambda}(\Omega)} \epsilon_{mom} \,.$$

**Theorem 1.** *For $f \in \mathcal{C}(\Omega)$, let $R_M(f) := \left| \int_\Omega f(S)d\nu - \sum_{j=1}^{M} w_j f(Y_j) \right|$ be the cubature error using the "compressed" point set $Y$ instead of $X$. We get*

$$R_M(f) \leq C\, E_n(f;\Omega) + \|f\|_{L^2_{d\lambda}(\Omega)}\, \epsilon_{mom}, \quad \forall\, f \in \mathcal{C}(\Omega)\,, \tag{8}$$

**Proof**. Let $p_n^*$ be the polynomial of best approximation of $f$ of degree not greater than $n$ in $\Omega$. Then

$$\begin{aligned} R_M(f) = \left| \int_\Omega f(S)d\nu - \sum_{j=1}^{M} w_j f(Y_j) \right| &\leq \left| \int_\Omega f(S)d\nu - \int_\Omega p_n^*(S)d\nu \right| \\ + \left| \int_\Omega p_n^*(S)d\nu - \sum_{j=1}^{M} w_j p_n^*(Y_j) \right| &+ \left| \sum_{j=1}^{M} w_j p_n^*(Y_j) - \sum_{j=1}^{M} w_j f(Y_j)d\nu \right| \\ &\leq \left( \nu(\Omega) + \sum_{j=1}^{M} |w_j| \right) E_n(f;\Omega) + \|p_n^*\|_{L^2_{d\lambda}(\Omega)} \epsilon_{mom}\,, \end{aligned}$$

where $E_n(f;\Omega) = \|f - p_n^*\|_{L^2_{d\lambda}(\Omega)}$ is the best polynomial approximation error. To conclude, it is enough using the inequality

$$\|p_n^*\|_{L^2_{d\lambda}(\Omega)} \leq \|p_n^* - f\|_{L^2_{d\lambda}(\Omega)} + \|f\|_{L^2_{d\lambda}(\Omega)} \leq \sqrt{\lambda(\Omega)} \|p_n^* - f\|_{L^\infty(\Omega)} + \|f\|_{L^2_{d\lambda}(\Omega)}$$

getting

$$R_M(f) \leq C\, E_n(f;\Omega) + \|f\|_{L^2_{d\lambda}(\Omega)}\, \epsilon_{mom}, \quad \forall\, f \in \mathcal{C}(\Omega)\,,$$

with

$$C = \nu(\Omega) + \sum_{j=1}^{M} |w_j| \sqrt{\lambda(\Omega)} \epsilon_{mom}\,. \tag{9}$$

This conclude the proof. □

Due to the assumed positivity of the weights, the constant $C$ in (9) can be written as follows $C = \nu(\Omega) + \rho \left| \sum_{j=1}^{M} w_j \right| \sqrt{\lambda(\Omega)} \epsilon_{mom}$ with

$$\rho = \frac{\sum_i |w_i|}{|\sum_i w_i|} \in [1, +\infty)\,, \tag{10}$$

that measures how many cubature weights of negative sign are present among all the weights. This quantity can be consider as a *measure of stability* of the method: if it assumes the value 1, then there is complete stability and so the capability of computing the integrals, with the prescribed precision, but with much less points (big values indicate a worsening of the process).

Therefore the cubature error depends on the moment error $\epsilon_{mom}$ and the stability constant $\rho$. Hence, the inequality (8) gives practical information on the error growth with NNLS (where $\rho = 1$) while with AFP it will be of practical use when the ratio $\rho$ is not too big.

## 5. Numerical tests

We present some examples of cubature in 2 and 3 dimensional domains. The domains we consider could be either convex and non-convex, discretized with Halton points (i.e. using low-discrepancy sequences), but can obviously be discretized with other low-discrepancy set of points, with random points or grids. Halton points on a given convex or union of convex can also be considered as a superset of a WAM for the domain. Then, by the properties P3 and P4 of WAMs, they are a "WAM" from which we can extract the corresponding AFP.

The functions we considered in the 2-dimensional domains are test functions used in many problems and applications (cf. e.g. [9, 13]):

$$\begin{aligned} f_1(x,y) &= \frac{3}{4} e^{-\frac{1}{4}((9x-2)^2+(9y-2)^2)} + \frac{3}{4} e^{-\frac{1}{49}(9x+1)^2 - \frac{1}{10}(9y+1)} \\ &\quad + \frac{1}{2} e^{-\frac{1}{4}((9x-7)^2+(9y-3)^2)} - \frac{1}{5} e^{-(9x-4)^2-(9y-7)^2} \qquad (11) \\ f_2(x,y) &= \sqrt{(x-0.5)^2 + (y-0.5)^2} \qquad (12) \\ f_3(x,y) &= \cos(30(x+y)). \qquad (13) \end{aligned}$$

The first one is the well-known *Franke* test function. The function $f_2$ has a singularity at $(0.5, 0.5)$, while the functions $f_3$ is infinitely differentiable with many ripples making the computation of its integral quite difficult.

For the 3-dimensional domains, we considered the following test functions (as already has been done in, e.g. [13, 14])

$$\begin{aligned} g_1(x,y,z) &= \frac{3}{4}e^{-\frac{1}{4}((9x-2)^2+(9y-2)^2+(9z-2)^2)} \\ &\quad + \frac{3}{4}e^{-\frac{1}{49}(9x+1)^2-\frac{1}{10}(9y+1)-\frac{1}{10}(9z+1)} \\ &\quad + \frac{1}{2}e^{-\frac{1}{4}((9x-7)^2+(9y-3)^2+(9z-5)^2)} \\ &\quad - \frac{1}{5}e^{-(9x-4)^2-(9y-7)^2-(9z-5)^2} \qquad (14) \\ g_2(x,y,z) &= \sqrt{(x-0.4)^2+(y-0.4)^2+(z-0.4)^2} \qquad (15) \\ g_3(x,y,z) &= \cos(4(x+y+z)) \qquad (16) \end{aligned}$$

which are similar to those considered in the 2-dimensional case.

All experiments have been performed on a laptop equipped with an Intel Core 2, 3.00 GHz processor, with 4GB of RAM by Matlab 7.10.0. Here we present only some experiments among the many more done in the Master's thesis of the first author [1].

*5.1. Experiments in* $\mathbb{R}^2$

To show that the cubature compression is working, we start with two simple convex domains: the square $[0,1]^2$ and the unit disk $x^2+y^2 \leq 1$. Notice that for these two domains cubature formulas with prescribed exactness are well-known (cf. e.g. [21, 6] and references therein). Both have been discretized with $10^4, 2\cdot 10^4$ and $5\cdot 10^4$ Halton points. In Tables 1–3 we display the relative errors obtained with the qMC method, the nonnegative least-squares (NNLS) and the extracted AFP for $n = 10, 20, 30$. The values of the integrals of the functions $f_1, f_2, f_3$ have been computed with the Matlab function `dblquad` giving the values (rounded to 4 decimal digits): $0.4070, 0.3826, 2.9\cdot 10^{-4}$ respectively. For the functions $f_1, f_2$ the validity of our new approaches is clearly confirmed by a decrease of the error with $n$. As we noticed, the function $f_3$ oscillates which makes difficult an accurate approximation, that is why errors are in general bigger.

In Figure 3 we display the extracted AFP for $n = 30$ from the discretization of the square with $5\cdot 10^4$ Halton points (not displayed). The distribution of the points reminds the arc-cosine distribution typical of nearly-optimal point sets [9]: indeed the AFP have asymptotically the same distribution of the true Fekete points, as proved in [4].

The results for the disk are in Tables 4–6 while in Figure 3) we show the corresponding AFP for $n = 30$. In this example, for almost all functions we

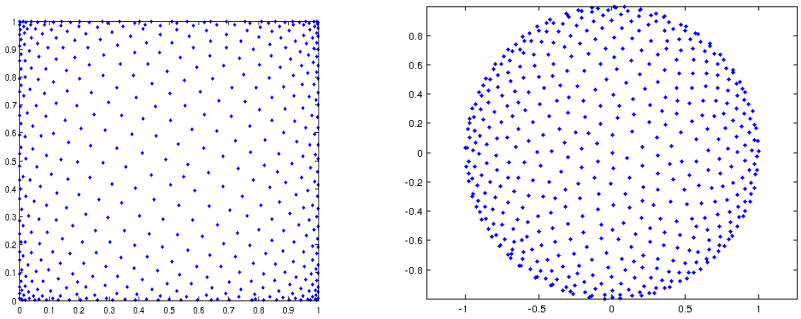

Figure 3: AFP on the square $[0,1]^2$ and the unit disk centered in the origin for $n = 30$ extracted from 5 $10^4$ Halton points

see that both methods do not need to increase either $N$ or $n$. In fact for the $N = 10^4$ and $n = 10$ we have almost the same results both with NNLS and AFP.

It is quite easy to observe that both methods compress the cubature, even if in average the AFP approach seems to perform better, even if the gain in precision is not sensible. For instance, Tables 4 and 5 show columns with the same values, because the differences are evident from the 4-th digit. In both these two tests, except for the function $f_3$, it is therefore reasonable to use the smallest $n$, i.e. $n = 10$.

| | method | $N = 10000$ | $N = 20000$ | $N = 50000$ |
|---|---|---|---|---|
| | qMC | 3.1e-04 | 1.3e-04 | 6.8e-05 |
| $n = 10$ | NNLS | 3.4e-03 | 6.3e-03 | 1.4e-03 |
| | AFP | 3.4e-03 | 2.9e-03 | 9.3e-04 |
| $n = 20$ | NNLS | 5.0e-04 | 5.4e-04 | 3.2e-05 |
| | AFP | 5.1e-04 | 2.0e-04 | 6.1e-05 |
| $n = 30$ | NNLS | 3.1e-04 | 1.3e-04 | 6.4e-05 |
| | AFP | 3.1e-04 | 1.3e-04 | 6.9e-05 |

Table 1: Relative errors for $f_1$ on the square $[0,1]^2$.

We present two other experiments on two more complicated domains.

The first one considers the domain whose shape is a lens (shown in Figure 4), consisting of the intersection of two disks with centers and radii $C_1 = (0,0)$, $r_1 = 5$ and $C_2 = (4,0)$, $r_2 = 3$, respectively. The initial grids used to extract "good points" are composed by Halton points. The

| | method | $N = 10000$ | $N = 20000$ | $N = 50000$ |
|---|---|---|---|---|
| | qMC | 4.4e-06 | 2.3e-05 | 9.1e-06 |
| $n = 10$ | NNLS | 5.4e-03 | 2.3e-03 | 3.2e-03 |
| | AFP | 2.2e-03 | 4.4e-03 | 3.7e-03 |
| $n = 20$ | NNLS | 5.6e-04 | 4.7e-04 | 2.8e-04 |
| | AFP | 3.9e-04 | 4.6e-04 | 5.2e-04 |
| $n = 30$ | NNLS | 7.8e-05 | 1.4e-04 | 1.8e-04 |
| | AFP | 1.2e-04 | 1.7e-04 | 1.3e-04 |

Table 2: Relative errors for $f_2$ on the square $[0, 1]^2$.

| | method | $N = 10000$ | $N = 20000$ | $N = 50000$ |
|---|---|---|---|---|
| | qMC | 3.7e-01 | 1.3e+0 | 4.0e-01 |
| $n = 10$ | NNLS | 2.2e+02 | 2.7e+02 | 9.7e+02 |
| | AFP | 2.4e+01 | 1.6e+02 | 1.7e+02 |
| $n = 20$ | NNLS | 8.3e+01 | 2.5e+02 | 6.1e+02 |
| | AFP | 2.2e+01 | 4.8e+01 | 4.1e+01 |
| $n = 30$ | NNLS | 2.1e+01 | 1.5e+01 | 3.3e+00 |
| | AFP | 1.4e+00 | 4.0e+00 | 1.3e+00 |

Table 3: Relative errors for $f_3$ on the square $[0, 1]^2$.

| | method | $N = 10000$ | $N = 20000$ | $N = 50000$ |
|---|---|---|---|---|
| | qMC | 7.4e-01 | 7.4e-01 | 7.4e-01 |
| $n = 10$ | NNLS | 7.3e-01 | 7.3e-01 | 7.4e-01 |
| | AFP | 7.4e-01 | 7.4e-01 | 7.4e-01 |
| $n = 20$ | NNLS | 7.4e-01 | 7.4e-01 | 7.4e-01 |
| | AFP | 7.4e-01 | 7.4e-01 | 7.4e-01 |
| $n = 30$ | NNLS | 7.4e-01 | 7.4e-01 | 7.4e-01 |
| | AFP | 7.4e-01 | 7.4e-01 | 7.4e-01 |

Table 4: Relative errors for $f_1$ on the unit disk.

exact moments have been computed using nodes and weights provided by the Matlab function `gqlens`, which uses subperiodic trigonometric gaussian formulas studied in [12] (the corresponding code can be found here `www.math.unipd.it/~marcov/CAAsoft.html`). The qMC moments have been computed by starting from an initial grid of $6 \cdot 10^5$ Halton points. In Figure 4 we show the nodes determined by `gqlens`, `NNLS` and the AFP. It is worth noticing that the nodes obtained with the `lsqnonneg` and the AFP

| | method | $N = 10000$ | $N = 20000$ | $N = 50000$ |
|---|---|---|---|---|
| | qMC | 6.7e-01 | 6.7e-01 | 6.7e-01 |
| $n = 10$ | NNLS | 6.7e-01 | 6.7e-01 | 6.7e-01 |
| | AFP | 6.7e-01 | 6.7e-01 | 6.7e-01 |
| $n = 20$ | NNLS | 6.7e-01 | 6.7e-01 | 6.7e-01 |
| | AFP | 6.7e-01 | 6.7e-01 | 6.7e-01 |
| $n = 30$ | NNLS | 6.7e-01 | 6.7e-01 | 6.7e-01 |
| | AFP | 6.7e-01 | 6.7e-01 | 6.7e-01 |

Table 5: Relative errors for $f_2$ on the unit disk.

| | method | $N = 10000$ | $N = 20000$ | $N = 50000$ |
|---|---|---|---|---|
| | qMC | 8.3e-03 | 9.1e-03 | 6.7e-03 |
| $n = 10$ | NNLS | 3.3e+00 | 3.7e+00 | 1.2e+01 |
| | AFP | 1.0e+02 | 4.2e+00 | 5.8e+00 |
| $n = 20$ | NNLS | 5.1e+00 | 9.9e-03 | 3.5e-01 |
| | AFP | 4.0e+00 | 2.9e+00 | 2.3e-01 |
| $n = 30$ | NNLS | 5.2e+00 | 4.7e+00 | 3.8e+00 |
| | AFP | 4.2e-01 | 1.1e+00 | 4.8e-01 |

Table 6: Relative errors for $f_3$ on the unit disk

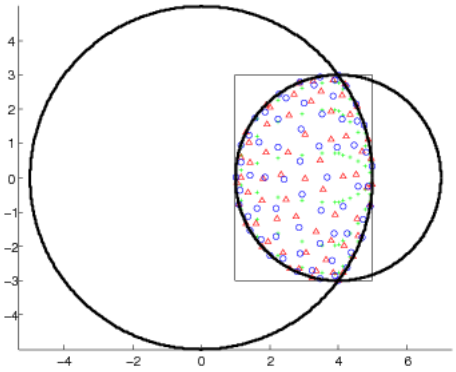


Figure 4: The lens approximated with $N = 2 \cdot 10^5$ Halton points and $n = 10$ (i.e. 66 points).The points with `gqlens` are indicated with (+), the ones with `lsqnonneg` with exact moments with ($\triangle$) and the AFP ($\circ$) .

accumulate along the boundary of the lens. In Table 7 we show, the num-

ber $n$ of the points extracted with different methods and the corresponding $N$ (number of points of the discretizion). Notice that `gqlens` uses slightly more nodes than the AFP, while `lsqnonneg` uses almost the same number of nodes as the AFP, except for the case in which the moments are exact (see the cases $n = 20, 30$).

We also computed the quantity $\rho$ (cf. (10)) which give information on the stability as detailed above. In Table 8 we present the values of the

| | method | $N = 10000$ | $N = 20000$ | $N = 50000$ |
|---|---|---|---|---|
| $n = 10$ | gqlens | 72 (1.00) | 72 (1.00) | 72 (1.00) |
| | qMC | 786 | 15179 | 37968 |
| | NNLS exact m. | 66 | 66 | 66 |
| | NNLS qMC m. | 66 | 66 | 66 |
| | AFP exact m. | 66 (1.01) | 66 (1.02) | 66 (1.02) |
| | AFP qMC m. | 66 (1.01) | 66 (1.02) | 66 (1.02) |
| $n = 20$ | gqlens | 242 (1.00) | 242 (1.00) | 242 (1.00) |
| | qMC | 7586 | 15179 | 37968 |
| | NNLS exact m. | 214 | 212 | 210 |
| | NNLS qMC m. | 231 | 231 | 231 |
| | AFP exact m. | 231 (1.02) | 231 (1.02) | 231 (1.03) |
| | AFP qMC m. | 231 (1.02) | 231 (1.02) | 231 (1.03) |
| $n = 30$ | gqlens | 512 (1.00) | 512 (1.00) | 512 (1.00) |
| | qMC | 7586 | 15179 | 37968 |
| | NNLS exact m. | 423 | 417 | 416 |
| | NNLS qMC m. | 496 | 496 | 495 |
| | AFP exact m. | 496 (1.06) | 496 (1.01) | 496 (1.01) |
| | AFP qMC m. | 496 (1.18) | 496 (1.02) | 496 (1.02) |

Table 7: Nodes on the lens extracted by `gqlens`, qMC, `lsqnonneg` with exact moments and approximated ones by qMC and the AFP, again with exact moments or approximated by qMC. In parentheses the ratio (10).

integrals of the functions $f_1, f_2, f_3$ at different values of $n$ by using `gqlens` starting from $5 \cdot 10^4$ Halton points to discretize the domain. The values of the integrals of $f_3$ show significant differences at different $n$ due, as noticed, to the oscillating behaviour of the function on the domain. We then expect that the corresponding relative errors will be quite big as well. In Tables 9–11 we provide the relative errors compared with the values of the integrals of Table 8.

As expected the errors $f_3$ are the biggest. For the functions $f_1$ and $f_2$ the errors are small and in agreement with the Koksma-Hlawka theorem. We notice that in all cases the integration with qMC shows little improvements

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | $n = 10$ | 5.2e-02 | | $n = 10$ | 5.5e+01 | | $n = 10$ | -5.3e-01 |
| $f_1$ | $n = 20$ | 5.4e-02 | $f_2$ | $n = 20$ | 5.5e+01 | $f_3$ | $n = 20$ | 5.2e-02 |
| | $n = 30$ | 5.3e-02 | | $n = 30$ | 5.5e+01 | | $n = 30$ | 1.6e+00 |

Table 8: Values of the integrals on the lens obtained with `gqlens` for the functions $f_1, f_2, f_3$ at $n = 10, 20, 30$.

| | method | $N = 10000$ | $N = 20000$ | $N = 50000$ |
|---|---|---|---|---|
| | qMC | 2.9e-06 | 1.2e-02 | 2.0e-02 |
| | NNLS exact m. | 5.6e-02 | 5.1e-02 | 3.0e-02 |
| $n = 10$ | NNLS qMC m. | 5.1e-02 | 1.6e-01 | 6.8e-02 |
| | AFP exact m. | 3.3e-03 | 2.7e-02 | 1.8e-02 |
| | AFP qMC m. | 3.6e-03 | 2.7e-02 | 1.8e-02 |
| | qMC | 3.3e-02 | 2.2e-02 | 1.4e-02 |
| | NNLS exact m. | 8.1e-03 | 8.5e-03 | 8.5e-04 |
| $n = 20$ | NNLS qMC m. | 2.7e-02 | 3.8e-03 | 3.9e-02 |
| | AFP exact m. | 1.0e-02 | 3.2e-02 | 6.0e-02 |
| | AFP qMC m. | 1.0e-02 | 3.2e-02 | 6.0e-02 |
| | qMC | 2.6e-02 | 1.5e-02 | 7.6e-03 |
| | NNLS exact m. | 4.5e-03 | 2.0e-03 | 3.3e-03 |
| $n = 30$ | NNLS qMC m. | 2.8e-02 | 1.3e-02 | 1.9e-02 |
| | AFP exact m. | 2.6e-02 | 3.0e-04 | 1.6e-03 |
| | AFP qMC m. | 3.8e-02 | 2.4e-03 | 1.4e-03 |

Table 9: Relative errors for $f_1$ on the lens, using qMC on Halton points. Errors are related to the results of Table 8 computed with `gqlens`.

on varying the cardinality $N$. In particular, by using the compression given by the NNLS and AFP with exact moments, the errors decrease with $n$, especially for the function $f_2$. Both the methods, when the moments are approximated with qMC, improve with $N$ and $n$, slowly for $f_1$ and faster for $f_2$. Actually for $f_1$ all methods behave in the same way. For $f_2$ the best results are those obtained with NNLS and AFP with exact moments.

We consider now the non-convex domain, illustrated in Figure 5, obtained by overlapping the disk with center $C = (0, 0)$ and radius $r = 3$, the square $[0, 4] \times [0, 4]$ and the closed polygon with vertices $V_1 = (1, 1), V_2 = (6, 2), V_3 = (7, 4), V_4 = (10, 3), V_5 = (9, 6), V_6 = (6, 7), V_7 = (4, 5), V_8 = (1, 6), V_9 = V_1$. For this domain we do not know,*indeed does not exist*, a cubature formula exact on the polynomials neither a way to compute the exact moments. The methods we compare are the ones that compute the

| | method | $N = 10000$ | $N = 20000$ | $N = 50000$ |
|---|---|---|---|---|
| $n = 10$ | qMC | 6.3e-04 | 1.4e-04 | 2.6e-04 |
| | NNLS exact m. | 3.4e-05 | 3.1e-05 | 7.4e-05 |
| | NNLS qMC m. | 6.6e-04 | 1.6e-04 | 2.2e-04 |
| | AFP exact m. | 5.1e-05 | 1.4e-05 | 3.0e-06 |
| | AFP qMC m. | 1.0e-04 | 3.5e-05 | 5.2e-05 |
| $n = 20$ | qMC | 6.4e-04 | 1.5e-04 | 2.5e-04 |
| | NNLS exact m. | 1.2e-06 | 3.3e-07 | 4.0e-07 |
| | NNLS qMC m. | 6.4e-04 | 1.5e-04 | 2.5e-04 |
| | AFP exact m. | 1.9e-07 | 2.4e-06 | 1.0e-06 |
| | AFP qMC m. | 4.8e-05 | 5.1e-05 | 5.0e-05 |
| $n = 30$ | qMC | 6.4e-04 | 1.5e-04 | 2.5e-04 |
| | NNLS exact m. | 1.1e-08 | 2.8e-08 | 2.9e-07 |
| | NNLS qMC m. | 6.4e-04 | 1.5e-04 | 2.5e-04 |
| | AFP exact m. | 4.5e-08 | 8.8e-09 | 1.1e-08 |
| | AFP qMC m. | 4.9e-05 | 4.9e-05 | 4.9e-05 |

Table 10: Relative errors for $f_2$ on the lens, using qMC on Halton points. Errors are related to the results of Table 8 computed with `gqlens`.

| | method | $N = 10000$ | $N = 20000$ | $N = 50000$ |
|---|---|---|---|---|
| $n = 10$ | qMC | 8.5e-01 | 6.9e-01 | 8.8e-01 |
| | NNLS exact m. | 4.5e+00 | 4.2e+00 | 1.9e+00 |
| | NNLS qMC m. | 4.8e-01 | 7.8e-01 | 7.3e+00 |
| | AFP exact m. | 1.6e+00 | 3.3e+00 | 3.1e+00 |
| | AFP qMC m. | 1.6e+00 | 3.3e+00 | 3.1e+00 |
| $n = 20$ | qMC | 2.5e+00 | 4.1e+00 | 2.2e+00 |
| | NNLS exact m. | 2.5e+01 | 2.6e+00 | 1.0e+01 |
| | NNLS qMC m. | 5.5e+00 | 1.1e+01 | 3.0e+01 |
| | AFP exact m. | 4.4e+01 | 2.6e+01 | 2.6e+01 |
| | AFP qMC m. | 4.4e+01 | 2.6e+01 | 2.6e+01 |
| $n = 30$ | qMC | 1.0e+00 | 1.1e+00 | 1.0e+00 |
| | NNLS exact m. | 1.3e+00 | 1.2e+00 | 4.1e-01 |
| | NNLS qMC m. | 1.7e+00 | 1.5e+00 | 7.2e-01 |
| | AFP exact m. | 1.7e+00 | 2.2e+00 | 1.3e+00 |
| | AFP qMC m. | 1.5e+00 | 2.0e+00 | 1.2e+00 |

Table 11: Relative errors for $f_3$ on the lens, using qMC on Halton points. Errors are related to the results of Table 8 computed with `gqlens`.

moments by qMC, the NNLS and the one that use the AFP with moments

approximated with qMC. The relative errors have been computed with respect to the first one (that computes the moments with the qMC). The qMC moments were computed using $6 \cdot 10^6$ Halton points.

In Figure 5 we show the points extracted by `lsqnoneneg` and the AFP for $n = 10$ from a set of $N = 50\,10^4$ Halton points (see also Table 12 for different values of $N$ and $n$). In Table 13 we show the integrals of the functions $f_1, f_2, f_3$ computed with the qMC method. Finally in Tables 14–16 we display the corresponding relative errors at different $N$ and $n$. At first glance, the errors computed with the AFP for $n = 30$ are not satisfactory, essentially because of the ratio $\rho$, as displayed in Table 12, that shows quite big values. This is an heuristic explaination of the important role of such a ratio in the comprehension of the approximation given by this approach. For $n = 10, 20$ the results with NNLS are almost equivalent with those obtained with the AFP.

In Table 17 we show the cputime for generating the Halton sequence in the qMC method (qMC tot), for extracting the positive weights with NNLS (NNLS tot) and the time for extracting the AFP (AFP tot). Moreover, in the same Table, we report the cputime for computing the integrals with the qMC, NNLS and AFP. To reduce the construction time for the NNLS and AFP is not so easy and it is not yet clear to us what can be a suitable way to solve this problem.

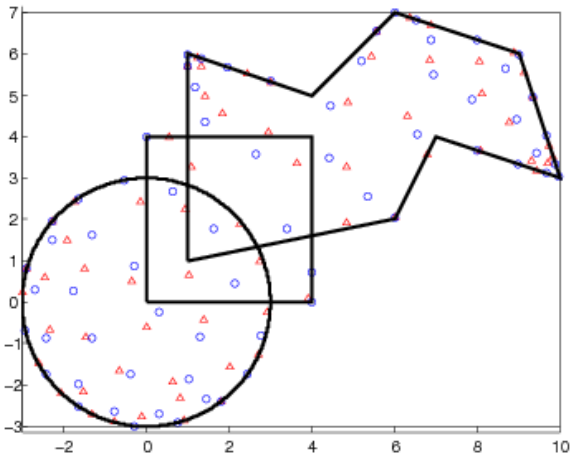


Figure 5: The composed domain approximated with $N = 2 \cdot 10^5$ Halton points and $n = 10$ (i.e. 66 points). The points selected with `lsqnonneg` are: with exact moments qMC with ($\triangle$) and the AFP ($\circ$) .

| | method | $N = 10000$ | $N = 20000$ | $N = 50000$ |
|---|---|---|---|---|
| $n = 10$ | qMC | 4658 | 9331 | 23323 |
| | NNLS | 66 | 66 | 66 |
| | AFP | 66 (1.02) | 66 (1.02) | 66 (1.03) |
| $n = 20$ | qMC | 4658 | 9331 | 23323 |
| | NNLS | 231 | 231 | 231 |
| | AFP | 231 (1.73) | 231 (1.43) | 231 (1.35) |
| $n = 30$ | qMC | 4658 | 9331 | 23323 |
| | NNLS | 496 | 496 | 496 |
| | AFP | 496 (2317.28) | 496 (1844.71) | 496 (3670.71) |

Table 12: For the composite domain of Fig. 5, varying the number $N$ of Halton points, we show the points extracted by `lsqnonneg` and the AFP at different $n$. We also show the ratio $\rho$ (in brackets).

| | | |
|---|---|---|
| $f_1$ | $N = 10000$ | 1.5e+01 |
| | $N = 20000$ | 1.5e+01 |
| | $N = 50000$ | 1.5e+01 |
| $f_2$ | $N = 10000$ | 2.5e+02 |
| | $N = 20000$ | 2.5e+02 |
| | $N = 50000$ | 2.5e+02 |
| $f_3$ | $N = 10000$ | 1.5e+00 |
| | $N = 20000$ | 1.6e+00 |
| | $N = 50000$ | 3.7e-01 |

Table 13: The integrals of $f_1, f_2, f_3$, computed with the qMC method with Halton points covering the rectangle that contains the composite domain.

| | method | $N = 10000$ | $N = 20000$ | $N = 50000$ |
|---|---|---|---|---|
| $n = 10$ | NNLS | 4.5e-02 | 1.7e-01 | 4.1e-02 |
| | AFP | 2.4e-02 | 9.1e-02 | 5.7e-02 |
| $n = 20$ | NNLS | 2.3e-02 | 8.0e-03 | 8.4e-03 |
| | AFP | 1.0e-02 | 1.0e-03 | 3.0e-02 |
| $n = 30$ | NNLS | 4.0e-03 | 1.0e-02 | 9.7e-03 |
| | AFP | 4.7e+00 | 2.8e-01 | 7.7e+00 |

Table 14: Relative errors for $f_1$ on the composite domain of Fig. 5.

| | method | $N = 10000$ | $N = 20000$ | $N = 50000$ |
|---|---|---|---|---|
| $n = 10$ | NNLS | 2.5e-03 | 1.6e-03 | 3.1e-03 |
| | AFP | 8.6e-06 | 3.4e-05 | 1.7e-04 |
| $n = 20$ | NNLS | 4.4e-04 | 3.6e-05 | 3.1e-04 |
| | AFP | 3.0e-04 | 9.3e-04 | 9.2e-04 |
| $n = 30$ | NNLS | 4.7e-05 | 7.0e-07 | 1.2e-04 |
| | AFP | 4.2e-01 | 1.2e-01 | 3.8e-01 |

Table 15: Relative errors for $f_2$ on the composite domain of Fig. 5.

| | method | $N = 10000$ | $N = 20000$ | $N = 50000$ |
|---|---|---|---|---|
| $n = 10$ | NNLS | 2.2e+00 | 2.3e-01 | 3.5e+00 |
| | AFP | 4.0e+00 | 3.8e+00 | 9.5e+00 |
| $n = 20$ | NNLS | 2.1e-01 | 3.4e+00 | 1.7e+01 |
| | AFP | 1.8e+00 | 4.4e-01 | 3.9e+00 |
| $n = 30$ | NNLS | 2.4e+00 | 2.2e+00 | 3.4e+00 |
| | AFP | 6.2e+02 | 8.6e+02 | 1.2e+04 |

Table 16: Relative errors for $f_3$ on the composite domain of Fig. 5.

*5.2. Experiments in* $\mathbb{R}^3$

We consider the domain consisting of the union of the cube $[0, 0.75]^3$ with the sphere centered in $C = (0.5, 0.5, 0.5)$ and radius $r = 0.5$ (see Figure 6). The moments qMC have been computed from a set of $1.5 \cdot 10^5$ Halton points. In Figure 7 we show the points extracted by `lsqnonneg` and the corresponding AFP. Both sets are well distributed in the domain except small clusterings close to the boundary of the domain. The test functions we considered are those given in (14)–(16). The results have been done taking a discretization of the domains with $N = 10^4, 2 \cdot 10^4$ and $5 \cdot 10^4$ Halton points and for $n \leq 9$ (the small values of $n$ depend on hardware limitations).

In Table 19 we report the number of points extracted on varying $N$ and $n$. Once again we recall that the AFP extracted are as many as the dimension of the 3-variate space of polynomials of degree $n$, i.e. $\eta_n$, while those determined by `lsqnonneg` sometimes are a little less (see for example the case $n = 9$ and $N = 5 \cdot 10^4$).

In Table 18 we show the values of the integrals of the functions obtained with the qMC at different sets of Halton points. The integrals have been computed by considering the parallelepiped surrounding the domain (its

| | | $f_1$ | $f_2$ | $f_3$ |
|---|---|---|---|---|
| $n = 10$ | qMC tot | 1.2e-01 | 1.5e-01 | 1.3e-01 |
| | NNLS tot | 2.3e-01 | 3.1e-01 | 2.7e-01 |
| | AFP tot | 4.3e-01 | 4.2e-01 | 4.0e-01 |
| | qMC | 2.1e-05 | 4.1e-05 | 1.8e-05 |
| | NNLS | 6.0e-06 | 9.0e-06 | 6.0e-06 |
| | AFP | 9.0e-06 | 5.0e-06 | 5.0e-06 |
| $n = 20$ | qMC tot | 1.4e-01 | 1.4e-01 | 1.2e-01 |
| | NNLS tot | 3.4e+00 | 2.8e+00 | 2.7e+00 |
| | AFP tot | 2.7e+00 | 2.7e+00 | 3.1e+00 |
| | qMC | 5.0e-05 | 2.8e-05 | 2.0e-05 |
| | NNLS | 1.0e-05 | 1.1e-05 | 8.0e-06 |
| | AFP | 8.0e-06 | 5.0e-06 | 7.0e-06 |
| $n = 30$ | qMC tot | 1.2e-01 | 1.3e-01 | 1.1e-01 |
| | NNLS tot | 1.7e+01 | 1.7e+01 | 1.6e+01 |
| | AFP tot | 9.2e+00 | 1.0e+01 | 9.9e+00 |
| | qMC | 3.0e-05 | 3.3e-05 | 1.7e-05 |
| | NNLS | 9.0e-06 | 1.1e-05 | 1.1e-05 |
| | AFP | 9.0e-06 | 9.0e-06 | 8.0e-06 |

Table 17: Cputime (in seconds) to compute the integrals on the composite domain of Fig. 5 starting from $N = 2 \cdot 10^4$ Halton points.

convex-hull) and taking the points falling into the domain. As we can see, with an approximation with 2 decimal digits, their values do not change with $N$.

In Tables 20–22 we show the relative errors of the cubature with the points determined by `lsqnonneg` and the AFP. In almost all examples, the approximation provided is quite good and the errors show a decreasing behavior. Moreover, errors computed with the `lsqnonneg` and the AFP are similar for all $N$.

In Table 23 we show the cputime for generating the Halton sequence in the qMC method (qMC tot), for extracting the positive weights with NNLS (NNLS tot) and the time for extracting the AFP (AFP tot). Moreover, in the same Table, we report the cputimes for computing the integrals with the qMC, NNLS and AFP. As it is clear, the more time spent in the construction is significantly gained in the computation of the integrals. This is the advange of the compression. As observed above, to reduce the construction time for the NNLS and AFP is not so easy and it is not yet clear to us which can be a suitable way to solve this problem.

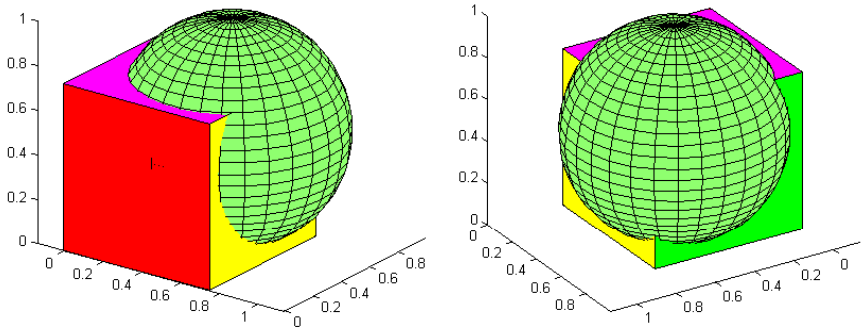

Figure 6: The 3-dimensional composite domain union of a cube and a sphere

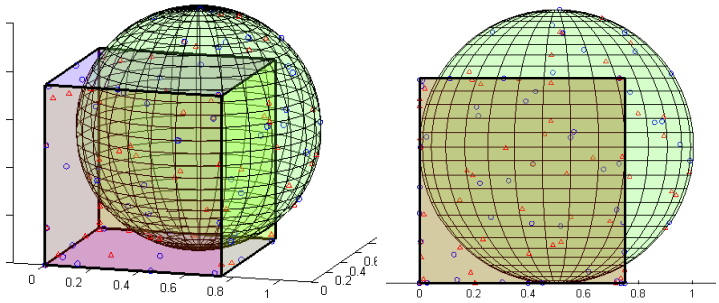

Figure 7: The points extracted by `lsqnonneg` and AFP (∘) for $n = 5$ from $N = 10^5$ Halton points.

| | | |
|---|---|---|
| $g_1$ | $N = 10000$ | 1.7e-01 |
| | $N = 20000$ | 1.7e-01 |
| | $N = 50000$ | 1.7e-01 |
| $g_2$ | $N = 10000$ | 2.7e-01 |
| | $N = 20000$ | 2.7e-01 |
| | $N = 50000$ | 2.7e-01 |
| $g_3$ | $N = 10000$ | 4.4e-02 |
| | $N = 20000$ | 4.4e-02 |
| | $N = 50000$ | 4.4e-02 |

Table 18: Integrals of the functions $g_1, g_2, g_3$ on the composite domain of Fig. 6 using the qMC method at different sets of Halton points.

| | method | $N = 10000$ | $N = 20000$ | $N = 50000$ |
|---|---|---|---|---|
| $n = 5$ | qMC | 6436 | 12882 | 32212 |
| | NNLS qMC m. | 56 | 56 | 56 |
| | AFP qMC m. | 56 (1.26) | 56 (1.20) | 56 (1.47) |
| $n = 7$ | qMC | 6436 | 12882 | 32212 |
| | NNLS qMC m. | 120 | 120 | 120 |
| | AFP qMC m. | 120 (1.32) | 120 (1.22) | 120 (1.48) |
| $n = 9$ | qMC | 6436 | 12882 | 32212 |
| | NNLS qMC m. | 220 | 220 | 218 |
| | AFP qMC m. | 220 (1.26) | 220 (1.42) | 220 (1.27) |

Table 19: For the composite domain of Fig. 6, varying the number $N$ of Halton points, we show the points extracted by `lsqnonneg`, the AFP at different $n$. We also show the ratio $\rho$ (in brackets).

| | method | $N = 10000$ | $N = 20000$ | $N = 50000$ |
|---|---|---|---|---|
| $n = 5$ | NNLS qMC m. | 4.21e-02 | 1.01e-02 | 2.53e-02 |
| | AFP qMC m. | 2.07e-02 | 2.98e-02 | 1.99e-02 |
| $n = 7$ | NNLS qMC m. | 1.53e-03 | 1.98e-02 | 2.80e-03 |
| | AFP qMC m. | 3.50e-03 | 1.46e-02 | 2.64e-02 |
| $n = 9$ | NNLS qMC m. | 5.37e-03 | 4.32e-04 | 5.50e-03 |
| | AFP qMC m. | 2.95e-03 | 7.73e-04 | 2.19e-03 |

Table 20: Relative errors for $g_1$ on the composite domain of Fig. 6. Errors are computed with respect to the qMC method.

| | method | $N = 10000$ | $N = 20000$ | $N = 50000$ |
|---|---|---|---|---|
| $n = 5$ | NNLS qMC m. | 1.16e-03 | 8.14e-03 | 5.25e-03 |
| | AFP qMC m. | 3.53e-03 | 4.20e-03 | 6.30e-03 |
| $n = 7$ | NNLS qMC m. | 2.13e-04 | 2.11e-03 | 3.44e-03 |
| | AFP qMC m. | 4.54e-04 | 1.15e-03 | 1.16e-03 |
| $n = 9$ | NNLS qMC m. | 1.16e-03 | 3.93e-04 | 5.38e-04 |
| | AFP qMC m. | 1.65e-03 | 9.84e-05 | 1.20e-04 |

Table 21: Relative errors for $g_2$ on the composite domain of Fig. 6. Errors are computed with respect to the qMC method.

| | method | $N = 10000$ | $N = 20000$ | $N = 50000$ |
|---|---|---|---|---|
| $n = 5$ | NNLS qMC m. | 3.05e-01 | 1.62e-01 | 6.51e-02 |
| | AFP qMC m. | 1.34e-01 | 2.73e-01 | 1.02e-03 |
| $n = 7$ | NNLS qMC m. | 1.42e-02 | 4.76e-03 | 2.63e-03 |
| | AFP qMC m. | 4.36e-03 | 1.16e-02 | 1.19e-03 |
| $n = 9$ | NNLS qMC m. | 4.30e-04 | 4.09e-05 | 2.28e-05 |
| | AFP qMC m. | 3.14e-03 | 2.07e-03 | 1.07e-03 |

Table 22: Relative errors for $g_3$ on the composite domain of Fig. 6. Errors are computed with respect to the qMC method.

| | | $g_1$ | $g_2$ | $g_3$ |
|---|---|---|---|---|
| $n = 5$ | qMC tot | 4.8e-02 | 3.7e-02 | 3.8e-02 |
| | NNLS tot | 3.5e-01 | 2.7e-01 | 2.6e-01 |
| | AFP tot | 8.1e+00 | 8.4e+00 | 8.1e+00 |
| | qMC | 4.8e-05 | 2.6e-05 | 2.9e-05 |
| | NNLS | 6.0e-06 | 8.0e-06 | 7.0e-06 |
| | AFP | 1.0e-05 | 7.0e-06 | 7.0e-06 |
| $n = 7$ | qMC tot | 4.5e-02 | 3.5e-02 | 5.3e-02 |
| | NNLS tot | 1.3e+00 | 1.1e+00 | 1.2e+00 |
| | AFP tot | 2.2e+01 | 2.1e+01 | 2.2e+01 |
| | qMC | 2.5e-05 | 2.0e-05 | 1.8e-05 |
| | NNLS | 9.0e-06 | 8.0e-06 | 8.0e-06 |
| | AFP | 5.0e-06 | 6.0e-06 | 6.0e-06 |
| $n = 9$ | qMC tot | 4.1e-02 | 3.5e-02 | 3.5e-02 |
| | NNLS tot | 3.8e+00 | 3.7e+00 | 3.6e+00 |
| | AFP tot | 5.1e+01 | 5.1e+01 | 5.1e+01 |
| | qMC | 2.1e-05 | 1.9e-05 | 1.8e-05 |
| | NNLS | 9.0e-06 | 7.0e-06 | 6.0e-06 |
| | AFP | 6.0e-06 | 5.0e-06 | 5.0e-06 |

Table 23: Cputime (in seconds) to compute the integrals on the composite domain of Figure 6 starting from $N = 2 \cdot 10^4$ Halton points.

We have done many other experiments in [1, Ch. 6] for classical domains, such as the unit cube, the cone (centered at the origin), the pyramid with square basis $[-0.5, 0.5] \times [-0.5, 0.5]$. Here we have presented the more interesting case where the domain is the union of two classical domains (cube and sphere).

We observe that the methods behave differently depending on $n$ and $N$.

For instance

- the tensor-product Gauss-Chebyshev points, used for the cube, depends only on $n$, and the number of nodes produced is always $\mathcal{O}(n^3)$;
- the cubature given by the Matlab function `3dWAM`, used for (generalized) cones and pyramids in [14], depends only on $n$.
- the cubature with qMC depends only on $N$, i.e. the cardinality of the Halton points;
- the cubature based on NNLS depends both on $n$ and $N$;
- the AFP depend both on $n$ and $N$. However, while the integrals using the AFP depend both on $n$ and $N$, the number of the nodes correspond to the dimension of the polynomial space.

Finally, since the *exact* moments are not known either for the cone, the pyramid and the composite domain here presented, we have used the qMC for approximating the integrals in `lsqnonneg` or, alternatively, the cubature weights were computed by the same function that extracts the AFP from a discretizion of the domain. In the case of the cone and the pyramid we actually know suitable WAMs (cf. [14]).

## 6. Conclusions and future works

The cubature on points extracted in a *clever way* from a set of $N$ quasi-random sequences, that is those obtained from nonnegative least squares (by using the Matlab function `lsqnonneg`) and the approximate Fekete points (by using the Matlab function `dexsets`), provide a general compression technique w.r.t. the quasi-Monte Carlo method. The method is quite flexible, in the sense that it applies to every space dimension and the number of points used to approximate the integrals is much less than those required by the classical quasi-Monte Carlo method. In fact, in general it uses only a number of points less or equal to the dimension of the polynomial space of degree, say $\nu_n$, with $\nu_n \ll N$. Both approaches can be applied once and for all for a big degree, say $n^*$,and than use the extracted points for computing the integrals for all $n \leq n^*$. Both approaches have shown a better behavior w.r.t quasi-Monte Carlo method, when the moments are exact except for functions with high variation within the integration domain. But this is also the case for the classical quasi-Monte Carlo method.

There are still open problems that we would like to address. The first consists in a new and faster way of finding AFP. This is indeed the bottleneck of the approach based on AFP. We are not sure if there will be a faster way to extract the AFP, but we are still convinced on the need of an *optimal* starting mesh, not simply a weakly admissible one. Optimal meshes are those whose cardinality grows like $\mathcal{O}(n^d)$ and, so far, are known only on some particular 2-dimensional domains (cf. [26, 18]). Another aspect to investigate more deeply is the error analysis especially for the AFP case. For the `NNLS` the recent error analysis provided in [29] and reported in §4, inequality (8), gives an overestimate for the cubature error. For the case of the moments approximated by AFP there is still room for such an analysis.

In any case, the new approaches presented in this work are promising, that is why it is worth to continue to investigate on them.

**Acknowledgments**. We thank an anonymous referee for the precise comments that have allowed to significantly improve the paper. This work has been supported by the University of Padova Project CPDA124755 "Multivariate approximation with applications to image reconstruction" and GNCS-INdAM funds.

## 7. References

[1] Bittante Claudia, *Una nuova tecnica di cubatura quasi-Monte Carlo su domini 2D e 3D* (in Italian), Master's thesis, University of Padua, March 2014.

[2] Briani Matteo, Sommariva Alvise, Vianello Marco, *Computing Fekete and Lebesgue points: simplex, square, disk*, J. Comput. Appl. Math. 236 (2012), no. 9, 2477–2486.

[3] L. Bos, J.-P. Calvi, N. Levenberg, A. Sommariva and M. Vianello, *Geometric weakly admissible meshes, discrete least squares approximation and approximate Fekete points*, Math. Comp. 80(275) (2011), 1623–1638.

[4] L. Bos, S. De Marchi, A. Sommariva and M. Vianello, *Computing multivariate Fekete and Leja points by numerical linear algebra*, SIAM J. Num. Anal. Vol. 48(5) (2010), 1984–1999.

[5] L. Bos, S. De Marchi, A. Sommariva and M. Vianello, *Weakly Admissible Meshes and Discrete Extremal Sets*, Numer. Math. Theor. Meth. Appl. Vol. 4(1) (2011), 1–12.

[6] Borislav Bojanov and Guergana Petrova, *Numerical integration over a disc. A new Gaussian quadrature formula*, Numer. Math. 80 (1998), 39–59.

[7] L. Bos and M. Vianello, *Low cardinality admissible meshes on quadrangles, triangles and disks*, Math. Inequal. Appl. 15 (2012), 229–235.

[8] R. E. Caflisch, *Monte Carlo and quasi-Monte Carlo methods*, Acta Numerica vol. 7, Cambridge University Press (1998), 1–49.

[9] Marco Caliari, Stefano De Marchi and Marco Vianello, *Bivariate polynomial interpolation on the square at new nodal sets*, Appl. Math. Comput. 165(2) (2005), 261–274

[10] J.P. Calvi and N. Levenberg, *Uniform approximation by discrete least squares polynomials*, J. Approx. Theory 152 (2008), 82–100.

[11] A. Civril and M. Magdon-Ismail, *On selecting a maximum volume submatrix of a matrix and related problems*, Theoretical Computer Science 410 (2009), 4801–4811.

[12] G. Da Fies and M. Vianello, *Agebraic cubature on planar lenses and bubbles*, Dolomites Res. Notes Approx. 5 (2012), 7–12.

[13] S. De Marchi, M. Marchiori and A. Sommariva, *Polynomial approximation and cubature at approximate Fekete and Leja points of the cylinder*, Appl. Math. Comput. Vol. 218(21) (2012), 10617–10629.

[14] S. De Marchi and M. Vianello, *Polynomial approximation on pyramids, cones and solids of rotation*, Dolomites Res. Notes Approx, Proceedings DWCAA12, Vol. 6 (2013), 20–26.

[15] Josef Dick and Friedrich Pillichshammer, Digital Nets and Sequences. Discrepancy Theory and Quasi-Monte Carlo Integration, Cambridge University Press, Cambridge, 2010.

[16] Michael Drmota and Robert F. Tichy, Sequences, discrepancies and applications, Lecture Notes in Math., 1651, Springer, Berlin, 1997.

[17] Achim Klenke, Probability Theory: A Comprehensive course. Springer-Verlag London, 2014.

[18] Kroó András, *On optimal polynomial meshes*, J. Approx. Theory 163 (2011), 1107–1124.

[19] Lawson, C. L. and R. J. Hanson, *Solving Least Squares Problems*, Prentice-0Hall 1974, p. 161.

[20] Christiane Lemieux, Monte Carlo and Quasi-Monte Carlo Sampling, Springer 2009.

[21] C. R. Morrow and T.N.L. Patterson, *Construction of algebraic cubatures rules using polynomial ideal theory*, SIAM J. Numer. Anal.15 (1978), 953–976.

[22] William J. Morokoff and Russel E. Caflisch, *Quasi-random sequences and their discrepancies*, SIAM J. Sci. Comput. 15 (1994), no. 6, 1251–1279.

[23] Harald Niederreiter. Random Number Generation and Quasi-Monte Carlo Methods. Society for Industrial and Applied Mathematics, 1992.

[24] Harald G. Niederreiter, *Quasi-Monte Carlo methods and pseudo-random numbers*, Bull. Amer. Math. Soc. 84 (1978), no. 6, 957–1041.

[25] Art B. Owen, *Multidimensional variation for quasi-Monte Carlo*, `http://finmath.stanford.edu/∼owen/reports/ktfang.pdf`.

[26] Piazzon F., Vianello M., *Analytic transformations of admissible meshes*, East J. Approx. 16 (2010), 389–398.

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

[28] Sommariva A., Vianello M., *Approximate Fekete points for weighted polynomial interpolation*. Electron. Trans. Numer. Anal. 37 (2010), 1–22.

[29] Sommariva A., Vianello M., *Compression of multivariate discrete measures and applications*. Numer. Funct. Anal. Optim. 36 (2015), 1198–1223.

[30] Santin G, Sommariva A., Vianello M., *An algebraic cubature formula on curvilinear polygons*. Appl. Math. Comput. 217 (2011), 10003–10015.

[31] Sommariva A., Vianello M., *ChebfunGauss: Matlab code for Gauss-Green cubature by the Chebfun package*, available at `www.math.unipd.it/∼marcov/CAAsoft.html`

[32] A. Sommariva and M. Vianello, *Gauss-Green cubature and moment computation over arbitrary geometries*, J. Comput. Appl. Math. 231 (2009), 886–896.

[33] Oto Strauch and Štefan Porubský, Distribution of Sequences: A Sampler, Peter Lang Publishing House, Frankfurt am Main 2005.

[34] Bruno Tuffin, *Radomization of quasi-Monte Carlo methods for error estimation: survey and normal approximation*, Monte Carlo Methods and Applications, 10(3-4) (2008), 617–628.