Martina Zaccaron

Discrete orthogonal polynomials and hyperinterpolation over planar regions

M.Sc. Thesis

Advisor: Prof. M. Vianello
Co-advisor: Dott. A. Sommariva

Università degli Studi di Padova
Dipartimento di Matematica
July, 25th 2014

Contents

List of Figures

List of Tables

List of Algorithms

Abstract

We present an algorithm for recursively generating orthogonal polynomials on a discrete set of the real plane. The algorithm is a sort of two dimensional Hermite-Lanczos method and is based on short recurrence relations. Practical implementation details and numerical issues are considered.

We apply the method in the context of polynomial hyperinterpolation of a continuous function. This is a projection operator that generalizes the idea of polynomial interpolation. With the aid of known cubature formulas with positive weights and high algebraic degree of exactness, we successfully apply hyperinterpolation over some planar regions, namely polygons and circular sections.

Least Squares approximation on Weakly Admissible Meshes is presented. Moreover given a finite measure that has a good asymptotic behaviour we try to build a well conditioned polynomial basis over non-standard domains.

Sommario

Si presenta un algoritmo per generare ricorsivamente polinomi ortogonali su un insieme finito di punti del piano. L'algoritmo è una sorta di metodo di Hermite–Lanczos due dimensionale ed è basato sulle ricorrenze brevi. Vengono analizzati dettagli implementativi e problemi numerici.

Si usa poi l'algoritmo per l'iperinterpolazione polinomiale di una funzione continua. Si tratta di un operatore di proiezione che generalizza l'idea di interpolazione polinomiale. Utilizzando una formula di cubatura a pesi positivi e con un grado di esattezza abbastanza alto, applichiamo l'iperinterpolazione su alcune regioni del piano, in particolare poligoni e sezioni circolari.

Utilizziamo la base polinomiale prodotta per l'approssimazione ai minimi quadrati su insiemi di punti opportuni (Weakly Admissible Meshes). Infine, data una misura discreta che abbia un buon comportamento asintotico, cerchiamo di costruire una base polinomiale ben condizionata su regioni del piano non-standard.

Introduction

Chapter 1

Bivariate discrete Orthogonal Polynomials

In this chapter, following the approach of [39] and [38], we explain how to build Orthogonal bivariate Polynomials (OPs) on a discrete set $\mathcal{S} \in \mathbb{R}^2$. The idea is to use a Lanczos-like method for recursively computing orthogonal bivariate polynomials on a finite set $\mathcal{S} = \{(x_j, y_j)\}_{j=1}^M \subset \mathbb{R}^2$ with respect to weights $\{w_j\}_{j=1}^M$ which are positive and such that $\sum_{j=1}^M w_j = 1$. The resulting algorithm produces the polynomials without employing the standard monomial basis at any point, obtaining better numerical stability.
Moreover the length of the recurrence formulas, even though increasing, is bounded like $\sqrt{8d}$, where d is the number of polynomials generated so far.

In the articles the algorithms are thought as a scheme to build approximations to Eigenvalues of a given matrix, which is the main target of Lanczos Methods. Here instead we are interested in the generation of the orthogonal polynomials given a prescribed set of nodes (eigenvalues) and weights (starting vector). This is the reason why in first place we explain how Lanczos Method can be seen as a method for generating Orthogonal Polynomials in the simple one dimensional case.

1.1 Lanczos Iteration and Univariate Orthogonal Polynomials

In this section we summarize how Lanczos method works for a Hermitian matrix $A \in \mathbb{C}^{M\times M}$, i.e. a matrix that coincides with its conjugate transposed, and stress the connection between the algorithm and the discrete orthogonal univariate polynomials.

1.1.1 Lanczos Iteration

Given a matrix A and a vector $\boldsymbol{v}$ the linear space spanned by the sequence

$$\boldsymbol{v},\ A\boldsymbol{v},\ A^2\boldsymbol{v}, \dots,\ A^{n-1}\boldsymbol{v}$$

is called the n-Krylov subspace and denoted by $\mathcal{K}_n(\boldsymbol{v}; A)$ or simply $\mathcal{K}_n$. If the vectors are linearly independent the dimension of $\mathcal{K}_n$ is exactly n, it is less otherwise.
Consider the set of univariate real polynomials $\phi \in \mathbb{P}^1_{n-1}$ of degree less then or equal to

$n-1$. The link with the n-Krylov subspace is clear when we replace the indeterminate by the matrix A:

$$\mathcal{K}_n(\boldsymbol{v}; A) = \text{span}\{\phi(A)\boldsymbol{v}, \phi \in \mathbb{P}^1_{n-1}\}.$$

The Lanczos Algorithm was firstly introduced by Lanczos in [44] to reduce an Hermitian matrix A in a tridiagonal form, computing at the same time a basis of the n-Krylov subspace $\mathcal{K}_n$. The tridiagonal form allows an easy computation of the eigenvalues of a matrix, so the Algorithm is widely used to compute or approximate eigenvalues [35]. There are various ways to recover the Lanczos iteration. We present a simple but effective one. Recall that every matrix $A \in \mathbb{C}^{M\times M}$ can be reduced in a upper Hessemberg form H via Householder similarities $A = QHQ^*$, for a unitary matrix Q [63]. Moreover if the starting matrix A is Hermitian, the resulting matrix H is tridiagonal, and we write the factorization as $A = QTQ^*$.
We write the matrices T and Q as

$$T = \begin{bmatrix} \alpha_1 & \beta_1 & & & \\ \beta_1 & \alpha_1 & \beta_1 & & \\ & \beta_1 & \ddots & \ddots & \\ & & \ddots & \alpha_{M-1} & \beta_{M-1} \\ & & & \beta_{M-1} & \alpha_M \end{bmatrix} \qquad Q = \begin{bmatrix} \boldsymbol{q}_1 & \boldsymbol{q}_2 & \cdots & \boldsymbol{q}_M \end{bmatrix}.$$

Consider the matrix identity $AQ = QT$. If we take the i-th column of each side, for $1 < i < M$, we obtain the following 3-term recurrence relation

$$A\boldsymbol{q}_i = \beta_{n-1}\boldsymbol{q}_{i-1} + \alpha_i\boldsymbol{q}_i + \beta_i\boldsymbol{q}_{i+1}, \tag{1.1}$$

The relation, known as Lanczos iteration, allows to build the columns of Q iteratively, according to the procedure listed in Algorithm (1.1). The columns of Q are usually called Lanczos vectors.

Algorithm 1.1 Lanczos Algorithm

$\beta_0 = 0$, $\boldsymbol{q}_0 = 0$
$\boldsymbol{q}_1$ arbitrary of unitary norm
for $j = 1$ to $J < M$ **do**
  $\boldsymbol{u} = A\boldsymbol{q}_j$
  $\alpha_j = (\boldsymbol{q}_j, \boldsymbol{u})$
  $\boldsymbol{u} = \boldsymbol{u} - \beta_{j-1}\boldsymbol{q}_{j-1} - \alpha_j\boldsymbol{q}_j$
  $\beta_j = ||\boldsymbol{u}||$
  $\boldsymbol{q}_{j+1} = \boldsymbol{u}/\beta_j$
end for

To clarify how we are actually generating an orthonormal basis of the Krylov subspace $\mathcal{K}_n$ with Lanczos, a simple theorem is handy.

Theorem 1. *Let A be Hermitian, and such that $A = QTQ^*$, with Q unitary and T tridiagonal as above. The set $\{\boldsymbol{q}_1, \ldots, \boldsymbol{q}_j\}$ form an orthonormal basis of $\mathcal{K}_j(A; \boldsymbol{q}_1)$ for each $1 \le j \le M$.*

Proof. The $\boldsymbol{q}_i$s are orthonormal, being the columns of a unitary matrix.
The only thing left to show is

$$\text{span}\{\boldsymbol{q}_1, \ldots, \boldsymbol{q}_j\} = \mathcal{K}_j(A; \boldsymbol{q}_1) = \text{span}\{\phi(A)\boldsymbol{q}_1, \phi \in \mathbb{P}^1_{j-1}\},$$

for $1 \leq j \leq n$. We prove the statement by induction on $1 \leq j \leq n$. For $j = 1$, there is nothing to prove since $\text{span}\{\boldsymbol{q}_1\} = \mathcal{K}_1(A; \boldsymbol{q}_1)$.

Suppose the double inclusion is true for $1 < j \leq n$. For each $\boldsymbol{q}_i$, $i \leq j$ there exists a polynomial $\phi_{i-1} \in \mathbb{P}^1_{i-1}$ such that $\boldsymbol{q}_i = \phi_{i-1}(A)\boldsymbol{q}_1$, thanks to the inductive hypothesis. Rearranging the recurrence relation (1.1) and taking the index $i = j$, it yields

$$\beta_j \boldsymbol{q}_{j+1} = A\boldsymbol{q}_j - \beta_{j-1}\boldsymbol{q}_{j-1} - \alpha_j \boldsymbol{q}_j.$$

Thus

$$\boldsymbol{q}_{j+1} \in \text{span}\{A\boldsymbol{q}_j,\ \boldsymbol{q}_{j-1},\ \boldsymbol{q}_j\} = \text{span}\{A\phi_{j-1}(A)\boldsymbol{q}_1,\ \phi_{j-2}(A)\boldsymbol{q}_1,\ \phi_{j-1}(A)\boldsymbol{q}_1\},$$

and the last set is contained in $\mathcal{K}_{j+1}(A; \boldsymbol{q}_1)$ because the polynomials $x\phi_{j-1}(x)$, $\phi_{j-2}(x)$ and $\phi_{j-1}(x)$ have degree less then or equal to j.
On the other hand, from relation (1.1) we get

$$\beta_j \boldsymbol{q}_{j+1} = A\phi_{j-1}(A)\boldsymbol{q}_1 - \beta_{j-1}\phi_{j-2}(A)\boldsymbol{q}_1 - \alpha_j \phi_{j-1}(A)\boldsymbol{q}_1.$$

We define ϕ_j to be the polynomial satisfying the relation

$$\beta_j \phi_j(x) = x\phi_{j-1}(x) - \beta_{j-1}\phi_{j-2}(x) - \alpha_j \phi_{j-1}(x).$$

The polynomial has degree j, hence $A^j \boldsymbol{q}_1 \in \text{span}\{\boldsymbol{q}_1, \ldots, \boldsymbol{q}_{j+1}\}$ and also the inclusion

$$\mathcal{K}_{j+1}(A; \boldsymbol{q}_1) \subset \text{span}\{\boldsymbol{q}_1, \ldots, \boldsymbol{q}_{j+1}\}$$

holds true. □

Since we took a Hermitian matrix A, there exist a unitary matrix U and a diagonal real matrix D such that $A = UDU^*$. Let $D = \text{diag}(x_1, \ldots, x_M)$; the elements x_js are the eigenvalues of A, whereas the j-th column of U is the eigenvector of A relative to the eigenvalue x_j.

1.1.2 Orthogonal Univariate Polynomials

We change our point of view and the aim is no longer computing the eigenvalues of A. Suppose to start the Algorithm 1.1 with the matrix $A = \text{diag}(x_1, \ldots, x_M)$ and starting Lanczos vector $\boldsymbol{q}_1$ of unitary norm, such that $|(\boldsymbol{q}_1, \boldsymbol{e}_j)|^2 = w_j > 0$. We are proving that the resulting Lanczos vectors $\boldsymbol{q}_i$s are the evaluations of some polynomials in the eigenvalues of A. It turns out that these polynomials are the orthogonal polynomials with respect to a discrete weighted inner product we are introducing further.
Let us rearrange (1.1) to a more convenient form, moving all the terms involving $\boldsymbol{q}_{n+1}$ to the left hand side

$$\beta_n \boldsymbol{q}_{n+1} = A\boldsymbol{q}_n - \alpha_n \boldsymbol{q}_n - \beta_{n-1}\boldsymbol{q}_{n-1} \tag{1.2}$$

From the Lanczos Algorithm 1.1 we also have the formulas for the coefficients α_n and β_n

$$\alpha_n = (A\boldsymbol{q}_n - \beta_{n-1}\boldsymbol{q}_{n-1}, \boldsymbol{q}_n) \quad \beta_n = ||A\boldsymbol{q}_n - \alpha_n \boldsymbol{q}_n - \beta_{n-1}\boldsymbol{q}_{n-1}||.$$

From the relation (1.2) we can read the scalar recurrence involving $q_{j,n+1}$ the j-th component of $\boldsymbol{q}_{n+1}$

$$\beta_n q_{j,n+1} = x_j q_{j,n} - \alpha_n q_{j,n} - \beta_{n-1} q_{j,n-1}. \tag{1.3}$$

Note that the solution $\boldsymbol{q}_n$ to the recurrence (1.2) is uniquely determined, once is known the starting vector $\boldsymbol{q}_1$. To be precise we can state the following.

Lemma 1. *If $q_{j,n+1}$ is the j-th component of the $(n+1)$-th Lanczos basis vector $\boldsymbol{q}_{n+1}$, then there exists a real polynomial $\phi_n(x)$ such that* $\deg \phi_n = n$ *and*

$$q_{j,n+1} = \phi_n(x_j) q_{j,1} \quad n \geq 0 \tag{1.4}$$

where x_j is the j-th eigenvalue of A and $\boldsymbol{q}_1$ is the initial Lanczos basis vector.

Proof. Suppose the starting vector is $\boldsymbol{q}_1$. In case $n = 0$, we take $\phi_0(x) = 1$ and obtain $q_{j,1} = \phi_0(x_j) q_{j,1}$.
Suppose that $\phi_{-1}(x) \equiv 0$. Let $n \geq 1$. Given the scalar relation (1.3), at each step we take the polynomial $\phi_n(x)$ to satisfy the 3-term recurrence

$$\beta_n \phi_n(x) = (x - \alpha_n)\phi_{n-1}(x) - \beta_{n-1}\phi_{n-2}(x),$$

for all $x \in \{x_1, \ldots, x_M\}$. This gives a unique choice for the values of ϕ_n in the nodes x_j. Moreover since the $\boldsymbol{q}_i$s satisfy the recurrence (1.3), we have $q_{j,n+1} = \phi_n(x_j) q_{j,1}$. □

It follows immediately that the ϕ_ns satisfy a 3-term recurrence, namely

$$\begin{cases} \phi_{-1}(x) \equiv 0 \\ \phi_0(x) \equiv 1 \\ \beta_n \phi_n(x) = (x - \alpha_n)\phi_{n-1}(x) - \beta_{n-1}\phi_{n-2}(x) \quad n \geq 1 \end{cases} \tag{OP}$$

for all $x \in \{x_1, \ldots, x_M\}$.

Let $\{x_j\}_{j=1}^M$ be an arbitrary set of nodes and $\{w_j\}_{j=1}^M$ be positive weights such that $\sum_{j=1}^M w_j = 1$. We introduce a discrete weighted inner product on the space of univariate real polynomials $\mathbb{P}^1$ as follows

$$\langle \varphi, \rho \rangle_w = \sum_{j=1}^{M} w_j \varphi(x_j) \rho(x_j) \tag{1.5}$$

It is readily verified that it is an inner product because it is linear in each component, symmetric and positive definite. Remarkably the ϕ_n defined by the recurrence in (OP) are orthogonal with respect to this inner product.

Theorem 2. *The polynomials ϕ_n defined in (OP) are orthogonal with respect to the inner product $\langle \cdot, \cdot \rangle_w$ defined in (1.5). That is to say*

$$\langle \phi_l, \phi_h \rangle_w = \delta_{l,h}, \quad h, l \geq 0.$$

Proof. Recall that the Lanczos vectors are orthonormal, hence $(\boldsymbol{q}_s, \boldsymbol{q}_t) = \delta_{s,t}$ for all $s, t \geq 1$. Now given two any polynomials ϕ_l, ϕ_h

$$\begin{aligned} \langle \phi_l, \phi_h \rangle_w &= \sum_{j=1}^{n} w_j \phi_l(x_j)\phi_h(x_j) = \sum_{j=1}^{n} (\phi_h(x_j)\sqrt{w_j})(\phi_l(x_j)\sqrt{w_j}) = \\ &= \sum_{j=1}^{n} \left(\phi_h\left(x_j\right) q_{j,1}\right)\left(\phi_l\left(x_j\right) q_{j,1}\right) = (\boldsymbol{q}_{h+1}, \boldsymbol{q}_{l+1}) = \delta_{h+1,l+1} = \delta_{h,l}, \end{aligned}$$

thus concluding the proof. □

To summarize we have shown that the Lanczos iteration, in exact arithmetic, produces a family $\{\phi_n\}_{n \geq 0}$ of polynomials that are orthogonal with respect to the discrete inner product defined in (1.5). Moreover these polynomials satisfy the 3-term recurrence given by (OP).

1.2 Bivariate discrete Orthogonal Polynomials

The interested reader could find further references about the theory of discrete orthogonal polynomials presented here in [69].
We denote the set of bivariate polynomials with real coefficients and degree less then or equal to n as $\mathbb{P}_n^2$. Let $\{(x_j, y_j)\}_{j=1}^M$ be a set of nodes and $\{w_j\}_{j=1}^M$ be positive weights such that $\sum_{j=1}^M w_j = 1$. We introduce the weighted discrete inner product for any two polynomials φ, ρ in $\mathbb{P}^2$ as

$$\langle \varphi, \rho \rangle_w \equiv \sum_{j=1}^{M} w_j \varphi(x_j, y_j) \rho(x_j, y_j). \tag{1.6}$$

Note that the definition makes sense since the form is linear in each component, symmetric and positive definite.
We name bivariate discrete Orthogonal Polynomials (OPs) polynomials of a prescribed degree that are orthonormal with respect to the inner product $\langle \cdot, \cdot \rangle_w$. To be precise a given basis $\{\varphi_l\}_{l=1}^d$ of the space $\mathbb{P}_n^2$, where the dimension is $d = d_n = \frac{(n+1)(n+2)}{2}$, consists of OPs if

$$\langle \varphi_i, \varphi_j \rangle_w = \delta_{i,j}. \tag{1.7}$$

1.2.1 How to order bivariate polynomials

In the univariate case the increasing degree of the monomials provide also an order. Indeed each polynomial can be written as the sum of monomials ordered by their increasing degree. The ordered sequence of the coefficients of the monomials provide a unique representation in the space of polynomials of a fixed degree. The correspondence in the space $\mathbb{P}_n^1$ is

$$(a_0, a_1, \ldots, a_n) \longleftrightarrow \varphi(x) = \sum_{j=0}^{n} a_j x^j.$$

When we use two variables to obtain a similar bijection between ordered sequences and polynomials, it is necessary to introduce an order among the monomials $x^j y^l$. A common choice to set an order among monomials is to take the graded lexicographic order, denoted by $\succ$. We say that

$$x^{j_1} y^{l_1} \succ x^{j_2} y^{l_2}$$

if either $j_1 + l_1 > j_2 + l_2$ or if $j_1 + l_1 = j_2 + l_2$ and $j_1 > j_2$. We define the leading term (LT) of a polynomial φ to be greatest (with respect to $\succ$) monomial with a non zero coefficient together with its coefficient. Thus $LT(\varphi) = cx^j y^l$ if $c \neq 0$ and the monomial $x^j y^l$ is the greatest monomial with respect to relation $\succ$ in the sum that gives φ.

Example 1. Consider the polynomial $\varphi(x, y) = 3 + 5x + 2y^2 + 3xy + y^3x + 5x^4$. It is of degree 4 and it is a sum of 6 monomials. The monomials are ordered by the increasing graded lexicographic order. The leading term of φ is $5x^4$. The sequence that uniquely determines the polynomial in the space $\mathbb{P}_4^2$ (dimension 15) is $(3, 0, 5, 2, 3, 0, 0, 0, 0, 0, 0, 1, 0, 0, 5)$.

1.2.2 Computing OPs with the monomial basis

The goal is to compute OPs of degree up to n in a prescribed set of distinct nodes $\mathcal{S} = \{(x_j, y_j)\}_{j=1}^{M}$, with associated positive weights $\{w_j\}_{j=1}^{M}$ summing to one.
For the moment we consider a straightforward manner to compute them. First we evaluate the ordered monomial basis of bivariate polynomials of degree less then or equal to n in the nodes, the result is a set of vectors. Every vector is associated to a monomial $x^h y^l$ and contains in its j-th component the evaluation of the monomial in the j-th node, i.e. $x_j^h y_j^l$. The produced sequence is then

$$\{\mathbf{1},\, \boldsymbol{y},\, \boldsymbol{x},\, \boldsymbol{y}^2,\, \boldsymbol{xy},\, \boldsymbol{x}^2,\, \dots,\, \boldsymbol{y}^k,\, \boldsymbol{xy}^{k-1}, \dots, \boldsymbol{x}^{k-1}\boldsymbol{y},\, \boldsymbol{x}^k\}, \tag{1.8}$$

where we denote by $\mathbf{1}$ the vector that has all components equal to 1.
Using the sequence of vectors, we generate another set of vectors orthonormal with respect to the standard Euclidean Inner product $(\cdot,\cdot)$ of $\mathbb{R}^n$. The existence of such a basis is theoretically guaranteed by the Gram-Schmidt process [35].
After orthogonalizations and scaling, at each step we obtain the sequence

$$\{\boldsymbol{p}_{0,0},\, \boldsymbol{p}_{0,1},\, \boldsymbol{p}_{1,0},\, \boldsymbol{p}_{0,2},\, \boldsymbol{p}_{1,1},\, \boldsymbol{p}_{2,0},\, \dots,\, \boldsymbol{p}_{0,k},\, \boldsymbol{p}_{1,k-1}, \dots, \boldsymbol{p}_{k-1,1},\, \boldsymbol{p}_{k,0}\}. \tag{1.9}$$

Thanks to the property of the Gram-Schmidt process

$$\operatorname{span}\{\boldsymbol{x}^s \boldsymbol{y}^{t-s},\ 0 \le t \le k,\ 0 \le s \le t\} = \operatorname{span}\{\boldsymbol{p}_{s,t-s},\ 0 \le t \le k,\ 0 \le s \le t\},$$

on $\mathcal{S}$, for each $t \le k$.
The last step is consider the weights. We introduce the matrix

$$W = \operatorname{diag}(\sqrt{w_1}, \dots, \sqrt{w_M}).$$

We take two polynomials φ, ρ in $\mathbb{P}_k^2$ and denote by $\boldsymbol{p}$ and $\boldsymbol{q}$ in $\mathbb{R}^M$ the vectors such that $\boldsymbol{p}_j = \varphi(x_j, y_j)$ and $\boldsymbol{q}_j = \rho(x_j, y_j)$, then the following relation between the standard Euclidean inner product $(\cdot,\cdot)$ and the weighted discrete inner product holds

$$\langle \varphi, \rho \rangle_w = (W\boldsymbol{p}, W\boldsymbol{q}).$$

To finally obtain the polynomials evaluated in the nodes we scale the j-th component of each vector in the sequence (1.9) by the j-th weight.
In matrix term, denoting by P the matrix which has the vectors of (1.9) as columns, our final target is $Q = WP$.

We now observe something that allows interesting savings in computations. Notice that to expand the basis in the nodes and obtain the sequence in (1.8) we need only the vectors from the previous cycle. In particular the following proposition holds for each $(x_j, y_j) \in \mathcal{S}$.

Proposition 1. *The degree k monomials at the point $(x_0, y_0) \in \mathbb{R}^2$ are obtained from the degree $k-1$ monomials, multiplying them for the values x_0, y_0, for each $k \in \mathbb{N}$.*

Proof. We can prove the statement by induction. In the case $k = 1$ we just take the values x_0 and y_0. Now suppose we are given the ordered degree k monomials

$$\{y_0^k,\, x_0 y_0^{k-1}, \dots,\, x_0^{k-1} y_0,\, x_0^k\}.$$

We can obtain the degree $k+1$ monomials by first multiplying the y_0^k for y_0 and then all the degree k monomials by x_0. The result is indeed the ordered set

$$\{y_0^{k+1},\, x_0 y_0^k,\, x_0^2 y_0^{k-1} \dots,\, x_0^k y_0,\, x_0^{k+1}\},$$

thus concluding the proof. □

1.2.3 Computing OPs with Lanczos

The main drawback of the generation of OPs by computing the monomial basis in the nodes and then orthogonalizing is the lack of numerical stability. Therefore our concern is find a stable way of generating OPs. We should try to find a bivariate alternative to the univariate procedure seen in section 1.1. As we are working with polynomials we expect there is some kind of Krylov subspace lurking beneath.
Take H and K to be

$$H = \mathrm{diag}(x_1, \ldots, x_M) \qquad K = \mathrm{diag}(y_1, \ldots, y_M).$$

They are commuting real symmetric matrices such that the spectrum of $N = H + iK \in \mathbb{C}^{M\times M}$ equals $\mathcal{S} = \{(x_j, y_j)\}_{j=1}^{M} \subset \mathbb{R}^2$, after identifying $\mathbb{C}$ with $\mathbb{R}^2$.
Given a vector $\boldsymbol{v} \in \mathbb{R}^M$, for every $k \in \mathbb{N}$, we consider the following bivariate generalization of the definition of Krylov subspace

$$\mathcal{K}_k(H, K; \boldsymbol{v}) \equiv \mathrm{span}\{\boldsymbol{v},\, K\boldsymbol{v},\, H\boldsymbol{v},\, K^2\boldsymbol{v},\, HK\boldsymbol{v},\, H^2\boldsymbol{v}, \ldots, K^k\boldsymbol{v}, \ldots, H^k\boldsymbol{v}\},$$

where

$$d = d_k \equiv \frac{(k+1)(k+2)}{2}$$

denotes the dimension of the linear subspace when all the vectors are linearly independent. The connection with the bivariate polynomials $\phi(x, y) \in \mathbb{P}_k^2$ of degree less then or equal to k is clear once we replace the monomial x by H and the monomial y by K. Thus an equivalent definition is

$$\mathcal{K}_k(H, K; \boldsymbol{v}) \equiv \mathrm{span}\{\phi(H, K)\boldsymbol{v},\, \phi \in \mathbb{P}_k^2\}.$$

Observe that d_k is the dimension of the space $\mathbb{P}_k^2$. Furthermore let

$$\{\varphi_{j,l-j}(x, y), \quad l = 0, \ldots, k;\ j = 0, \ldots, l \\ \text{with} \quad LT(\varphi_{j,l-j}(x, y)) = c_{j,l-j}x^j y^{l-j} \text{ and } \quad c_{j,l-j} \neq 0\}$$

be any polynomial basis of $\mathbb{P}_k^2$, different form the monomial basis but ordered with respect to $\succ$. Then for each $0 \leq l \leq k$ the inclusion

$$\mathrm{span}\{H^j K^{t-j}\boldsymbol{v}\} = \mathrm{span}\{\varphi_{j,l-j}(H, K)\boldsymbol{v}\},$$

with $0 \leq t \leq l, 0 \leq j \leq t$ holds.

Take a vector $\boldsymbol{q}_0 \in \mathbb{R}^M$ of unit length such that $|(\boldsymbol{q}_0, \boldsymbol{e}_j)|^2 = w_j$, where as usually $\boldsymbol{e}_j$ is the j-th vector of the canonical basis of $\mathbb{R}^M$, and $(\cdot, \cdot)$ the standard Euclidean inner product of $\mathbb{R}^M$.
Let

$$Q \equiv [\boldsymbol{q}_0\ \boldsymbol{q}_1\ \boldsymbol{q}_2\ \cdots\ \boldsymbol{q}_{d_{k-1}-1}\ \boldsymbol{q}_{d_{k-1}}\ \cdots\ \boldsymbol{q}_{d_k-1}]$$

be the matrix having as columns the vectors obtained from applying the Gram-Schmidt procedure to the Krylov subspace $\mathcal{K}_k(H, K; \boldsymbol{q}_0)$. For each fixed degree $0 \leq t \leq k$ we have

$$\mathrm{span}\{H^j K^{l-j}\boldsymbol{q}_0,\ 0 \leq l \leq t, 0 \leq j \leq l\} = \mathrm{span}\{\boldsymbol{q}_0\ \ldots,\ \boldsymbol{q}_{d_t-1}\}.$$

This can be easily proved by induction, thanks to the property of Gram-Schmidt process.
Moreover for fixed indexes (j, l), with $0 \leq l \leq k$, $0 \leq j \leq l$ there exists a polynomial

$\phi_{j,l-j}(x,y) \in \mathbb{P}_k^2$ with leading term $x^j y^{l-j}$ such that a particular column of Q equals $\phi_{j,l-j}(H,K)\boldsymbol{q}_0$. Then the set

$$\{\phi_{j,l-j}(x,y)\}_{0\leq l\leq k, 0\leq j\leq l}$$

is an ordered basis of the bivariate polynomials $\mathbb{P}_k^2$.
More precisely let us denote with $l(h,s) = d_{h-1} + s$, with $0 \leq h \leq k$ and $0 \leq s \leq h$ the indexing. Then

$$\boldsymbol{q}_{l(h,s)} = \phi_{s,h-s}(H,K)\boldsymbol{q}_0,$$

for $0 \leq h \leq k$ and $0 \leq s \leq h$.
There can occur zero vectors among $\boldsymbol{q}_i$s and how this happens depends on the location of the nodes $\mathcal{S}$.

Proposition 2. *The vector $\boldsymbol{q}_{l(k,s)}$ equals the zero vector of $\mathbb{R}^M$, i.e. $\boldsymbol{q}_{l(k,s)} = \mathbf{0}$, if and only if the polynomial $\phi_{s,k-s}(x,y)$ is such that $\phi_{s,k-s}(x_j,y_j) = 0$ for each $1 \leq j \leq M$.*

Proof. Recall that $H = \mathrm{diag}(x_1,\dots,x_M)$ and $K = \mathrm{diag}(y_1,\dots,y_M)$. Since we choose $\boldsymbol{q}_0$ such that $|(\boldsymbol{q}_0,\boldsymbol{e}_j)|^2 = w_j > 0$, the starting vector is a linear combination of all the eigenvectors of both H and K, namely $\{\boldsymbol{e}_j\}_{j=1}^M$. If $\boldsymbol{q}_{l(k,s)} = \mathbf{0}$ then

$$0 = (\boldsymbol{e}_j, \boldsymbol{q}_{l(k,s)}) = (\boldsymbol{e}_j, \phi_{s,k-s}(H,K)\boldsymbol{q}_0) = \phi_{s,k-s}(x_j,y_j)(\boldsymbol{q}_0,\boldsymbol{e}_j),$$

for each $j = 1,\dots,M$. Being $(\boldsymbol{q}_0,\boldsymbol{e}_j)$ different from zero, $\phi_{s,k-s}(x_j,y_j)$ must be zero for each j. □

We required $\mathcal{S}$ to have M distinct points, namely $H + iK$ has distinct eigenvalues. The matrices H and K are diagonal, hence their eigenvectors are exactly the $\boldsymbol{e}_j$ of the canonical basis. The starting vector $\boldsymbol{q}_0$ is supported by every spectral subspace of N, i.e. $(\boldsymbol{q}_0,\boldsymbol{e}_j) \neq 0$ for all $1 \leq j \leq M$, thus by applying the Gram-Schmidt method to the columns of Q, we can produce at most M nonzero vectors $\boldsymbol{q}_j$. Being orthonormal, they yield an orthonormal basis of $\mathbb{R}^M$. Usually the degree of the polynomials k is such that $d_k < M$, going further lead only to zero vectors.
If the nodes in $\mathcal{S}$ have the additional property of not lying on an algebraic curve of $\mathbb{R}^2$ with degree lower then k, then there are not zero vectors among the columns of Q.

Proposition 3. *Suppose that $\{(x_j,y_j)\}_{j=1}^M$ do not lie on an algebraic curve of degree less then or equal to k, then the matrix Q has full column rank.*

Proof. Suppose we get a zero, $\boldsymbol{q}_i = \mathbf{0}$, $i < d_k$. This means that there exists a polynomial $\phi_{s,l-s}$, of degree $l \leq k$, such that $\phi_{s,l-s}(H,K)\boldsymbol{q}_0$ equals zero. Since $(\boldsymbol{q}_0,\boldsymbol{e}_j) \neq 0$, for each nodes (x_j,y_j) we have $\phi_{s,l-s}(x_j,y_j) = 0$, for a positive integer $l \leq k$, meaning the points are located in an algebraic curve $\phi_{s,l-s}(x,y) = 0$ of the plane of degree $l \leq k$. This contradicts the assumption. □

The purpose is clarify the link with the orthogonal polynomials when the matrix Q has full column rank. Take the polynomials $\phi_{s,i-s}$ and $\phi_{t,h-t}$ in $\mathbb{P}_k^2$ such that

$$\boldsymbol{q}_{l(s,i)} = \phi_{s,i-s}(H,K)\boldsymbol{q}_0$$

and

$$\boldsymbol{q}_{l(t,h)} = \phi_{t,h-t}(H,K)\boldsymbol{q}_0,$$

with $0 \leq s \leq i,\ 0 \leq i \leq k$ and $0 \leq t \leq h,\ 0 \leq h \leq k$. The columns of Q yields the evaluations in the nodes (x_j, y_j) of the bivariate discrete orthogonal polynomials

$$\{\phi_{t,h-t}(x,y)\}_{0\leq h\leq k; 0\leq t\leq h},$$

multiplied by the constant factor $\sqrt{w_j}$. Indeed

$$\begin{aligned}\langle \phi_{s,i-s}, \phi_{t,h-t}\rangle_w &= \sum_{j=1}^{M} w_j \phi_{s,i-s}(x_j,y_j)\phi_{t,h-t}(x_j,y_j) = \\ &= \sum_{j=1}^{M} (\phi_{s,i-s}(x_j,y_j)\sqrt{w_j})(\phi_{t,h-t}(x_j,y_j)\sqrt{w_j}) = \\ &= (\phi_{s,i-s}(H,K)\boldsymbol{q}_0, \phi_{t,h-t}(H,K)\boldsymbol{q}_0) = (\boldsymbol{q}_{l(s,i)}, \boldsymbol{q}_{l(t,h)}) = \\ &= \delta_{l(s,i),l(t,h)} = \delta_{s,t}\delta_{i,h}. \end{aligned} \tag{1.10}$$

Note that to expand the Krylov subspace we could have used the vectors resulting from the Gram-Schmidt process at the previous cycle, instead of the monomial basis. Expanding the generalized Krylov subspace using a set of orthonormal vectors yields more numerical stability; this procedure is precisely the two dimensional generalization of the Lanczos method we were looking for.
We now prove the statement that allows to build the Krylov subspace without passing through the monomial basis.

Proposition 4. *For each* $1 \leq t \leq k-1$,

$$\begin{aligned}span\{\boldsymbol{q}_0,\ K\boldsymbol{q}_0,\ H\boldsymbol{q}_0, \ldots, &K^{t-1}\boldsymbol{q}_0, \ldots, H^{t-1}\boldsymbol{q}_0, \\ &K^t\boldsymbol{q}_0,\ HK^{t-1}\boldsymbol{q}_0, \ldots, H^sK^{t-s}\boldsymbol{q}_0, \ldots, H^t\boldsymbol{q}_0\}\end{aligned}$$

equals

$$\begin{aligned}span\{\boldsymbol{q}_0,\ \boldsymbol{q}_{l(1,0)},\ \boldsymbol{q}_{l(1,1)}, \ldots, &\boldsymbol{q}_{l(t-1,0)}, \ldots, \boldsymbol{q}_{l(t-1,t-1)}, \\ &K\boldsymbol{q}_{l(t-1,0)},\ H\boldsymbol{q}_{l(t-1,0)}, \ldots, H\boldsymbol{q}_{l(t-1,s)}, \ldots, H\boldsymbol{q}_{l(t,t)}\}.\end{aligned}$$

Proof. We prove the statement by induction on $1 \leq t \leq k-1$. The case $t = 1$ is the inclusion

$$\text{span}\{\boldsymbol{q}_0,\ K\boldsymbol{q}_0,\ H\boldsymbol{q}_0\} = \text{span}\{\boldsymbol{q}_{l(0,0)},\ \boldsymbol{q}_{l(1,0)},\ \boldsymbol{q}_{l(1,1)}\},$$

which is true because the $\boldsymbol{q}_l$s form an orthonormal basis of the space on the left hand side.
Suppose the statement is true for t and prove it for $t+1$. By inductive hypothesis we have the leading monomial of $\phi_{s,l-s}$ equal to $x^s y^{l-s}$ for each $s = 0, \ldots, l,\ l = 0, \ldots, t$. In particular $LT(\phi_{s,t-s}) = x^s y^{t-s}$ for each $0 \leq s \leq t$. Moreover

- $LT(y\phi_{l(t,0)}(x,y)) = yy^t$
- $LT(x\phi_{l(t,s)}(x,y)) = xx^s y^t$ for $0 \leq s \leq t$.

But then replacing x by H and y by K the equality between linear space of the statement holds for $t+1$. Indeed

- $K\phi_{0,t}(H,K)\boldsymbol{q}_0 = K\boldsymbol{q}_{l(t,0)}$

- $H\phi_{s,t-s}(H,K)\boldsymbol{q}_0 = H\boldsymbol{q}_{l(t,s)}$ for $0 \le s \le t$.

□

With this idea, we expand the basis using the evaluations of the polynomials computed in the previous cycle. We generate a basis of $\mathbb{R}^M$ according to the rule

$$\begin{array}{cccccccccccccccc} I & K & H & K & H & H & K & H & H & H & K & H & H & H & H & \cdots \\ \boldsymbol{q}_0 & \boldsymbol{q}_0 & \boldsymbol{q}_0 & \boldsymbol{q}_1 & \boldsymbol{q}_1 & \boldsymbol{q}_2 & \boldsymbol{q}_3 & \boldsymbol{q}_3 & \boldsymbol{q}_4 & \boldsymbol{q}_5 & \boldsymbol{q}_6 & \boldsymbol{q}_6 & \boldsymbol{q}_7 & \boldsymbol{q}_8 & \boldsymbol{q}_9 & \cdots \\ \boldsymbol{q}_0 & \boldsymbol{q}_1 & \boldsymbol{q}_2 & \boldsymbol{q}_3 & \boldsymbol{q}_4 & \boldsymbol{q}_5 & \boldsymbol{q}_6 & \boldsymbol{q}_7 & \boldsymbol{q}_8 & \boldsymbol{q}_9 & \boldsymbol{q}_{10} & \boldsymbol{q}_{11} & \boldsymbol{q}_{12} & \boldsymbol{q}_{13} & \boldsymbol{q}_{14} & \cdots \end{array} \tag{1.11}$$

which is read from left to right. The first row indicates the multiplying matrix; the second row indicates the numbering of the vector which is multiplied by the matrix in the first row above and the third row indicates the numbering of the resulting vector, after orthogonalizing against all the previous vectors in the third row and scaling by its length.
There can occur zero vectors among $\boldsymbol{q}_i$, and how this happens depends on where the nodes $\mathcal{S}$ are located. More precisely if $\mathcal{S}$ lies on a low degree algebraic curve, then zero vectors will be generated by the orthogonalization process according to the following.

Proposition 5. *If some ϕ, with leading term of ϕ equal cx^jy^l vanishes on $\mathcal{S}$, then for $k \ge j+l$, we have $\boldsymbol{q}_{\frac{k(k+1)}{2}+s} = 0$ for $j \le s \le k-l$.*

Proof. The statement is obvious looking at the 1.11 for building the sequence of vectors $\boldsymbol{q}_j$ and the connection with the polynomials. □

1.2.4 Short Recurrences

The most interesting property of the Hermitian-Lanczos method is its computational efficiency, due to the 3-term recurrence arising from the procedure. In the bivariate case can we have a similar recurrence relation or do we have to give up with that property turning to a recurrence relation growing linearly with the iteration number? Fortunately it turns out that using the rule (1.11) the length of the recurrence is neither fixed nor it grows linearly with the iteration number. It is at most $2k+1$ while generating orthogonal bivariate polynomials of degree k and, consequently, the length of the recurrence grows asymptotically like the square root of the iteration number.

Theorem 3. *Let H, K in $\mathbb{R}^{M\times M}$ be symmetric and commuting such that the spectrum of $H+iK$ equals $\mathcal{S}$. Then with $\boldsymbol{q}_0 \in \mathbb{R}^M$, the length of recurrence for computing $\boldsymbol{q}_{\frac{k(k+1)}{2}+s}$ is at most $2k+1$ for $s = 0,\dots,k$.*

Proof. To generate the vectors of the kth cycle, we multiply first $\boldsymbol{q}_{\frac{k(k-1)}{2}}$ by K and then vectors $\boldsymbol{q}_j$ for $j = \frac{k(k-1)}{2}+s$, with $0 \le s \le k-1$, by H. Consider the standard inner product of $\mathbb{R}^M$

$$\left(K\boldsymbol{q}_{\frac{k(k-1)}{2}}, \boldsymbol{q}_{\frac{j(j+1)}{2}+t}\right) = \left(\boldsymbol{q}_{\frac{k(k-1)}{2}}, K\boldsymbol{q}_{\frac{j(j+1)}{2}+t}\right), \tag{1.12}$$

with $j \le k-1$ and $0 \le t \le j$. By construction every $\boldsymbol{q}_{\frac{k(k-1)}{2}+s}$ is orthogonal against $\rho(H,K)\boldsymbol{q}_0$ whenever $LT(r) < x^s y^{k-1-s}$. In particular, $K\boldsymbol{q}_{\frac{j(j+1)}{2}+t} = \rho(H,K)\boldsymbol{q}_0$ with $LT(\rho(x,y)) = c_{t,j-t}x^t y^{j-t+1}$. Thus if $j+1 < k-1$ or equivalently if $j < k-2$ then the inner product (1.12) is zero. Therefore, to compute $\boldsymbol{q}_{\frac{k(k-1)}{2}}$, it is only necessary to

orthogonalize against $2k-1$ vectors of the $(k-2)$-th and $(k-1)$-th cycles.

Consider for $0 \leq s \leq k-1$

$$\left(H\boldsymbol{q}_{\frac{k(k-1)}{2}+s}, \boldsymbol{q}_{\frac{j(j+1)}{2}+t}\right) = \left(\boldsymbol{q}_{\frac{k(k-1)}{2}+s}, H\boldsymbol{q}_{\frac{j(j+1)}{2}+t}\right), \tag{1.13}$$

with $j \leq k-1$ and $0 \leq t \leq j$. Now by the same reasoning as earlier, we get the sufficient conditions that either $j+1 \leq k-1$ or $j+1 = k-1$ and $t+1 < s$ for (1.13) to be zero. This means that orthogonalization is performed against $((k-1)-(s-2))+k+s = 2k+1$ vectors at most. □

Let $d_k = \frac{(k+1)(k+2)}{2}$ denote the number of vectors computed after completing the the k-th cycle, then this means saving and orthogonalizing against of order $\sqrt{8d}$ vectors per iteration. This is a very modest growth of the work and storage. In fact, the the length of the recurrence of the Algorithm 1.2 has the slowest growth for generating bivariate orthogonal polynomials we are aware of.

We can give an explicit algorithm of the Lanczos-like Algorithm in the bivariate case, with the Algorithm 1.2. This algorithm works in infinite precision arithmetic and some issues arise when we consider its practical implementation. All the next chapter will be devoted to explore how to remedy the lack of orthogonality, especially when very high degree polynomials and a lot of nodes are involved.

Algorithm 1.2 Bivariate Lanczos-Like Algorithm

H, K commuting real symmetric matrices, q_0 of unit length
for $k = 1$ to j **do**
  Define $k_0 \equiv \frac{(k-2)(k-1)}{2}$, $k_1 \equiv \frac{(k-1)k}{2}$, $k_2 \equiv \frac{k(k+1)}{2}$
  $\boldsymbol{q}_{k_2} = K\boldsymbol{q}_{k_1}$
  for $s = k_0 - 1$ to $k_2 - 1$ **do**
    $\alpha_s^{k_1} = (\boldsymbol{q}_{k_2}, \boldsymbol{q}_s)$, $\boldsymbol{q}_{k_2} = \boldsymbol{q}_{k_2} - \alpha_s^{k_1}\boldsymbol{q}_s$
  end for
  $\alpha_{k_2}^{k_1} = ||\boldsymbol{q}_{k_2}||$
  if $\alpha_{k_2}^{k_1} > 0$ **then**
    $\boldsymbol{q}_{k_2} = (1/\alpha_{k_2}^{k_1})\boldsymbol{q}_{k_2}$
  else
    $\boldsymbol{q}_{k_2} = 0$
  end if
  for $l = k_1$ to $k_2 - 1$ **do**
    $\boldsymbol{q}_{l+k+1} = H\boldsymbol{q}_l$
    for $s = l - k$ to $l + k$ **do**
      $\beta_s^l = (\boldsymbol{q}_{k+l+1}, \boldsymbol{q}_s)$, $\boldsymbol{q}_{k+l+1} = \boldsymbol{q}_{k+l+1} - \beta_s^l\boldsymbol{q}_s$
    end for
    $\beta_{k+l+1}^l = ||\boldsymbol{q}_{k+l+1}||$
    if $\beta_{k+l+1}^l > 0$ **then**
      $\boldsymbol{q}_{k+l+1} = (1/\beta_{k+l+1}^l)\boldsymbol{q}_{k+l+1}$
    else
      $\boldsymbol{q}_{k+l+1} = 0$
    end if
  end for
end for

We clarify the procedure to generate bivariate OPs with the following example.

Example 2. Let $\mathcal{S}$ consist of 8 nodes on the unit circle

$$(x_i, y_i) = \left(\cos\left(\frac{2\pi j}{8}\right), \sin\left(\frac{2\pi j}{8}\right)\right) \in \mathbb{R}^2 \quad j = 0, \ldots, 7.$$

Associate with $\mathcal{S}$ uniform weights $w_i = \frac{1}{8}$. Take $H = \text{diag}(x_1, \ldots, x_8)$, $K = \text{diag}(y_1, \ldots, y_8)$ and $\boldsymbol{q}_0 = \frac{1}{2\sqrt{2}}(1, \ldots, 1)^t$. Then by executing the algorithm 1.2 we obtain bivariate orthogonal polynomials

$$\begin{aligned}
&\phi_{0,0}(x, y) = 1,\\
&\phi_{0,1}(x, y) = \sqrt{2}y, \quad \phi_{1,0}(x, y) = \sqrt{2}x,\\
&\phi_{0,2}(x, y) = \sqrt{2}(2y^2 - 1), \quad \phi_{1,1}(x, y) = 2\sqrt{2}xy,\\
&\phi_{0,3}(x, y) = \sqrt{2}(4y^3 - 3y), \quad \phi_{1,2}(x, y) = \sqrt{2}(4xy^2 - x),\\
&\phi_{0,4}(x, y) = (8y^4 - 8y^2 + 1).
\end{aligned}$$

Note that $\phi_{2,0} = \phi_{2,1} = \phi_{3,0}$, due to the fact that the nodes lies on a second degree algebraic curve.

Chapter 2

Discrete Bivariate OPs in Finite Precision Arithmetic

We now explore the implementation aspects of the Algorithm 1.2 proposed in the First Chapter. The Algorithm was meant to compute not high degree polynomials and only in the nodes, thus we have to rearrange it for our purposes. We will see that Lanczos vectors produced becomes less and less orthogonal as the number of nodes and the polynomial degree increase. Loss of orthogonality is typical of Gram-Schmidt type of processes and it is even worse because of the short recurrences.
As orthogonality is crucial for our aim of building the approximation to a given function (see Chapter 3 and 4), we have to somehow remedy the loss of orthogonality. We consider different strategies, all based on the well known principle of "twice is enough". The first idea is to use a double QR after producing the evaluation of the polynomials according to the Algorithm; otherwise still remaining in the framework of the Gram-Schmidt algorithms, we can orthogonalize against all the previous vectors and in addition repeat the orthogonalization at each step to gain orthogonality among the generated vectors.
We choose the methods in case of non-uniform and uniform weights and then analyse "qualitatively" computational costs, efficiency and numerical stability of the various methods to make a suitable choice.

Notation

Below we list some symbols we use often in the sequel

Q matrix produced by Algorithm 1, Chapter 1

P matrix of the orthogonal polynomials

$\{\boldsymbol{x}_j\}_{j=1}^{M}$ nodes

$\{w_j\}_{j=1}^{M}$ weights

$\{\boldsymbol{y}_h\}_{h=1}^{H}$ finer mesh

$W = \text{diag}(\sqrt{w_1}, \ldots, \sqrt{w_M})$

$\{\phi_1, \ldots, \phi_d\}$ orthogonal polynomials

$\{\boldsymbol{p}_1, \dots, \boldsymbol{p}_d\}$ evaluations of orthogonal polynomials in the nodes

$\langle \cdot, \cdot \rangle_w$ discrete inner product

2.1 Some considerations about the weights

In the previous Chapter we ask the weights $\{w_j\}_{j=1}^M$ to be positive and such that $\sum_{j=1}^M w_j = 1$. The positivity is mandatory to make $\langle \cdot, \cdot \rangle_w$ an inner product, but it is not necessary that the weights sum to one. The case of positive weights summing to a number greater then one arises for example when the weights are obtained form a positive cubature rule (see Chapter 3).
Consider the case of positive weights $\{w_j\}_{j=1}^M$, with $\sum_{j=1}^M w_j = \sigma \geq 1$. Define

$$\tilde{w}_j = \frac{w_j}{\sigma}, \quad j = 1, \dots, M.$$

Then the set of new weights $\{\tilde{w}_j\}_{j=1}^M$ sums to one. Moreover if $\langle \cdot, \cdot \rangle_{\tilde{w}}$ is the inner product associated with the weights $\tilde{w}_j$ and $\langle \cdot, \cdot \rangle_w$ the inner product defined in (1.6), the following relation holds

$$\langle \cdot, \cdot \rangle_w = \sigma \langle \cdot, \cdot \rangle_{\tilde{w}}.$$

Given the set of nodes $\{(x_j, y_j)\}_{j=1}^n$ and positive weights $\{w_j\}_{j=1}^M$, from Algorithm 1.2 we obtain the matrix

$$Q = WP, \tag{2.1}$$

where $W = \operatorname{diag}(\sqrt{w_1}, \dots, \sqrt{w_M})$ and

$$P = [\phi_j(x_i, y_i)]_{ij} = [\boldsymbol{p}_1 \cdots \boldsymbol{p}_d], \quad 1 \leq j \leq d, 1 \leq i \leq M$$

contains the evaluation of the orthogonal polynomials with respect to $\langle \cdot, \cdot \rangle_w$ in the nodes. The goal is computing the polynomials $\{\phi_1, \dots, \phi_d\}$ at any point (x, y) of a certain compact domain $\Omega \subset \mathbb{R}^2$.
In the case of uniform weights the matrix Q equals P up to scaling, indeed

$$Q = \frac{1}{\sqrt{M}} P,$$

thus it is possible to compute P directly from Q, using the same recurrence used to compute Q. Another option is to slightly modify the algorithm for producing directly the matrix P; to this end observe that between the standard inner product $(\cdot, \cdot)$ of $\mathbb{R}^M$ and $\langle \cdot, \cdot \rangle_w$ the following relation

$$\langle \cdot, \cdot \rangle_w = (W \cdot, W \cdot)$$

holds. Hence if we orthogonalize with respect to the inner product $\langle \cdot, \cdot \rangle_w$, we obtain the matrix P. This second option is necessary in case W^{-1} has huge diagonal elements and computing P as $P = W^{-1}P$ is not possible, always in the case of non-uniform weights.

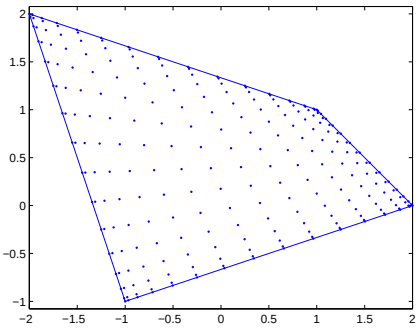


Figure 2.1: Quadrangle of vertices $\{(-1,-1),(2,0),(1,1),(-2,2)\}$. The mesh is produced by a WAM.

2.2 A domain for numerical tests

We want to compute the discrete orthogonal polynomials on the nodes of a cubature rule with positive weights (Chapter 3) or on good points for multivariate polynomial approximation, namely Weakly Admissible Meshes (WAMs) and Discrete Extremal Sets (see Chapter 4 for definitions). All these points are not contained in a low degree algebraic curve. Thus the matrix Q from the bivariate Lanczos Algorithm has full column rank and it is either square or with more rows then columns. Hereafter we denote by M the the number of rows of Q, that is equal to the cardinality of the nodes and by d the number of the columns, that is equal to the dimension of the space of bivariate polynomials of a fixed degree.

For testing the behaviour of the various algorithms, we consider a rather general quadrangle in the plane (Figure 2.1). The set of nodes in it is a WAM on the quadrangle; in the case of non-uniform weights we generate a (pseudo-) random vector of positive weights by the Matlab function `rand`. This domain is considered a non special domain because the orthogonality is not well preserved, and this is often the case for the set of nodes we are going to use.

2.3 Loss of orthogonality

We first test the behaviour of the algorithm with uniform weights. We implement the Algorithm exactly as presented in 1.2. There we used the Gram-Schmidt method in its numerical stable variant, called Modified Gram-Schmidt.
The orthogonality of the Lanczos vectors is desirable. Are we producing orthonormal vectors? A good index of the orthogonality of a rectangular matrix $Q \in \mathbb{R}^{M\times d}$ is the number

$$||Q^tQ - I_d||_\infty,$$

where the norm $||\cdot||_\infty$ is defined for a matrix $A = (a_{ij})$ to be

$$||A||_\infty = \max_j \max_i |a_{ij}|.$$

From Table 2.1 is clear that the Lanczos vectors are less and less orthogonal as the degree of the polynomials increase.

DEG	M	d	orthogonality
5	36	21	$6.661338e-16$
10	121	66	$1.176628e-13$
15	256	136	$2.623693e-11$
20	441	231	$9.841059e-09$
25	676	351	$3.256706e-06$
30	961	496	$6.097450e-03$
35	1296	666	$2.417971e-01$
40	1681	861	$3.804735e-01$
45	2116	1081	$3.951706e-01$
50	2601	1326	$4.348140e-01$
55	3136	1596	$3.968077e-01$
60	3721	1891	$4.991482e-01$

Table 2.1: Quadrangle of vertices $\{(-1,-1),(2,0),(1,1),(-2,2)\}$ with uniform weights $w_j = 1/M$. The last column contains the estimated orthogonality of the columns of $Q \in \mathbb{R}^{M\times d}$.

Experimenting we verify that the loss of orthogonality of Lanczos vectors depends strongly on the geometry of the domain and the location of the nodes. Different domains have different behaviours, in particular Table 2.2 shows different trends of orthogonality for nodes of WAMs over three different quadrangles with uniform weights.

DEG	M	d	orthogonality (A)	orthogonality (B)	orthogonality (C)
5	36	21	$7.008283e-16$	$9.367507e-16$	$6.349088e-16$
10	121	66	$2.097281e-15$	$1.138841e-12$	$1.341159e-12$
15	256	136	$3.688022e-15$	$1.272318e-09$	$8.194117e-10$
20	441	231	$1.241238e-14$	$7.751121e-08$	$7.981029e-07$
25	676	351	$1.046567e-14$	$1.060701e-03$	$1.263404e-02$
30	961	496	$1.948008e-14$	$1.281503e-01$	$2.986142e-01$
35	1296	666	$3.738503e-14$	$1.166014e-01$	$3.271176e-01$
40	1681	861	$4.036723e-14$	$2.024280e-01$	$4.080286e-01$
45	2116	1081	$6.319294e-14$	$2.050915e-01$	$5.035785e-01$
50	2601	1326	$8.054972e-14$	$1.745812e-01$	$4.979569e-01$
55	3136	1596	$4.529016e-14$	$1.964126e-01$	$5.175557e-01$
60	3721	1891	$6.010557e-14$	$2.227258e-01$	$7.407150e-01$

Table 2.2: Loss of orthogonality for 3 different quadrangle with uniform weights, using Algorithm 1.2. Quadrangle A has vertices $\{(-1,-1),(1,-1),(1,1),(-1,1)\}$; quadrangle B has vertices $\{(-1,-1),(1,-1),(0,1),(-1,1)\}$; quadrangle C has vertices $\{(-1,0),(0,-1),(1,0),(0,1)\}$. Orthogonality is measured by $||\cdot||_\infty$ norm.

2.4 Gram-Schmidt and loss of orthogonality

The Gram-Schmidt (GS) numerical algorithm deserves a wise and accurate analysis in a lot of different papers; see among others [35],[5]. There are two main variants, the Classical and Modified. Below we list the two algorithms, using a Matlab-like notation.

Let $A = [\boldsymbol{a}_1 \cdots \boldsymbol{a}_d]$ be the starting matrix, and $Q = [\boldsymbol{q}_1 \cdots \boldsymbol{q}_d]$, the matrix obtained after the process.

Algorithm 2.1 Classical GS

$r_{11} = ||\boldsymbol{a}_1||$, $Q = \boldsymbol{a}_1/r_{11}$
for $j = 2$ to d **do**
  Orthogonalize:
  $\boldsymbol{r}_j = Q^t \boldsymbol{a}_j$, $\boldsymbol{v} = \boldsymbol{a}_j - Q\boldsymbol{r}_j$
  Normalize:
  $r_{jj} = ||\boldsymbol{v}||$
  if $r_{jj} > 0$ **then**
    $\boldsymbol{r}_j = [\boldsymbol{r}_j^t; r_{jj}]^t$
    $Q = [Q; \boldsymbol{v}/r_{jj}]$
  end if
end for

Algorithm 2.2 Modified GS

$r_{11} = ||\boldsymbol{a}_1||$, $\boldsymbol{q}_1 = \boldsymbol{a}_1/r_{11}$
for $j = 2$ to d **do**
  Orthogonalize:
  $\boldsymbol{v} = \boldsymbol{a}_j$
  for $i = 1$ to $j - 1$ **do**
    $r_{ij} = \boldsymbol{q}_i^t \boldsymbol{v}$, $v = v - r_{ij}\boldsymbol{q}_i$
  end for
  Normalize:
  $r_{jj} = ||\boldsymbol{v}||$
  if $r_{jj} > 0$ **then**
    $\boldsymbol{r}_j = [\boldsymbol{r}_j^t; r_{jj}]^t$
    $Q = [Q; \boldsymbol{v}/r_{jj}]$
  end if
end for

The Classical GS (CGS) and Modified GS (MGS), despite being theoretically equivalent, have deep difference in numerical behaviour: MGS is numerically stable, CGS is not [35].
If we consider the loss of orthogonality is proportional to the square of the condition number for CGS, whereas it is proportional to the condition number for MGS (see [33]).
In our algorithm the matrix is built recursively and we have analogous cases in literature considering Arnoldi or Lanczos methods. Usually GS is applied to a starting matrix with a certain condition number, and only in this case precise bounds on the loss of orthogonality in terms of the condition number of the starting matrix are known. In the present case bounds are not available, this is the reason why in this section we do not give precise theoretical results on the loss of orthogonality, treating the problem only in its numerical aspect.

DEG	Monomial	Chebyshev	Lanczos
5	$8.545052e+02$	$2.936783e+02$	$1.000000e+00$
10	$1.873821e+06$	$2.470754e+05$	$1.000000e+00$
15	$4.518375e+09$	$1.994484e+08$	$1.000000e+00$
20	$1.107869e+13$	$1.578622e+11$	$1.000000e+00$
25	$2.756511e+16$	$1.233764e+14$	$1.000013e+00$
30	$9.586300e+18$	$1.244124e+16$	$1.027106e+00$
35	$6.220577e+20$	$1.254566e+16$	$1.088690e+02$
40	$2.730426e+22$	$2.296354e+16$	$8.569502e+04$
45	$1.019542e+24$	$2.983872e+16$	$1.345507e+07$
50	$3.445620e+25$	$3.638682e+16$	$1.507130e+09$
55	$1.659042e+27$	$4.840527e+16$	$1.002238e+10$
60	$3.029659e+29$	$9.197680e+16$	$4.988739e+11$

Table 2.3: Condition number of the Vandermonde-like matrices for the monomial basis, Chebyshev basis and Lanczos basis evaluated in the nodes of the WAM on the test quadrangle.

2.5 A Better Conditioned Basis

Even though not orthogonal, the matrix produced from the Algorithm 1.2 is better conditioned with respect to other known polynomial basis. Clearly the Lanczos basis is better conditioned then the standard monomial basis. But remarkably Lanczos basis is also better conditioned if compared to the total-degree Chebyshev basis of the minimal rectangle containing the domain.
A small condition number is particularly appreciated in the least-squares approximation and also for the extraction of Approximate Fekete Points (Chapter 4).
For testing the condition number we consider again the quadrangle of Figure 2.1. The condition number is computed by Matlab routine `cond`.

2.6 Remedies to the loss of orthogonality

With the Algorithm 1.2 we are orthogonalizing against an increasing number of vectors, but not all of them since we are using short recurrences. We expect a situation similar to the univariate Lanczos, where the short recurrences causes a loss of orthogonality. Indeed at each step of Lanczos the components in the direction of the eigenvectors corresponding to the greatest eigenvalues are amplified causing the phenomenon of the ghost eigenvalues [50],[18].

The first idea is to use the Lanczos vectors produced by the Algorithm 1.2 with uniform weights. The vectors are not orthogonal, but the matrix Q is not too ill conditioned. When the conditioning (in the 2-norm) is below (or not much greater) then the reciprocal of machine precision two QR factorization succeed in producing a matrix orthogonal up to an error close to machine precision. This phenomenon is known as "twice is enough" in the contest of numerical orthogonalization [34].
The advantage of this method is using a fast and optimized routine in Matlab, the economical QR factorization. Moreover the triangular matrices R_1 and R_2 (of the first and the second QR factorization respectively) yield the changes of basis that bring the polynomial basis to be orthogonal with respect to the inner product $\langle \cdot, \cdot \rangle_w$.

We summarize the procedure in Algorithm 2.3. Note that the first orthogonalization is needed only to make the matrix better conditioned, whereas multiplication by W is necessary to orthogonalize with respect to the inner product $\langle \cdot, \cdot \rangle_w$.

Algorithm 2.3 Computing OPs on the nodes - Method 1

Compute matrix Q from the Algorithm 1.2.
$Q_1 R_1 = Q$
$Q_2 R_2 = W Q_1$
$P = Q R_1^{-1} R_2^{-1}$

The other way to compute the matrix P of the orthogonal polynomials is to use the GS process with respect to the inner product $\langle \cdot, \cdot \rangle_w$. But -as we saw already in the case of uniform weights- we have to face the loss of orthogonality.
Even accepting recurrences growing linearly with the iteration number and losing the great efficiency of the algorithm, there is no substantial improvement of the orthogonality and the only way to obtain orthogonal vectors from a GS process is to repeat the orthogonalization twice at each step [33]. Note that also in this case we are using the "twice is enough" principle.
In this second case we rewrite the Algorithm 1.2 as shown in Algorithm 2.4. In the orthogonalization steps is possible to use either Modified GS or Classical GS. The two algorithms share the same trend as regarding the orthogonality. However, the CGS allows block computations, hence is faster in Matlab (and more generally in a block-operations) environment.

Algorithm 2.4 Computing OPs on the nodes - Method 2

INPUT

- $H = \text{diag}(x_1, \ldots, x_M)$
- $K = \text{diag}(y_1, \ldots, y_M)$
- $W = \text{diag}(\sqrt{w_1}, \ldots, \sqrt{w_M})$

for $k = 1$ to j **do**
  Define $k_1 \equiv \frac{(k-1)k}{2}$, $k_2 \equiv \frac{k(k+1)}{2}$
  $\boldsymbol{q}_{k_2} = K \boldsymbol{q}_{k_1}$
  for $s = 1$ to $k_2 - 1$ **do**
    Orthogonalise twice $\boldsymbol{q}_{k_2}$ against $\boldsymbol{q}_s$ with respect to $\langle \cdot, \cdot \rangle_w$
  end for
  Normalize $\boldsymbol{q}_{k_2}$
  for $l = k_1$ to $k_2 - 1$ **do**
    $\boldsymbol{q}_{l+k+1} = H \boldsymbol{q}_l$
    for $s = 1$ to $l + k$ **do**
      Orthogonalise twice $\boldsymbol{q}_{l+k+1}$ against $\boldsymbol{q}_s$ with respect to $\langle \cdot, \cdot \rangle_w$
    end for
    Normalize $\boldsymbol{q}_{k+l+1}$
  end for
end for

2.7 Computing OPs at any point

We would like to evaluate the computed OPs at any fixed point of $\mathbb{R}^2$, possibly different from the nodes.
Let $\{(x_j, y_j)\}_{j=1}^M$ be the set of nodes and $\{w_j\}_{j=1}^M$ the weights. As we have seen in the previous section we can produce the matrix P using two methods. In this section we will show that for both methods is possible to compute the OPs also outside the nodes.

If the matrix P is produced with the Algorithm 2.3, we store the recurrence coefficients used to produce Q, and the upper triangular matrices R_1 and R_2 of the changes of basis. Storing all these terms allows the computation of every polynomials also outside the nodes.
In the second case we store the recurrence coefficients produced by both the first and the second GS orthogonalizations. These coefficients are sufficient to reconstruct the orthogonal polynomials at any points in $\mathbb{R}^2$.

Algorithm 2.5 Computing OPs at any point - Method 1

INPUT:

- (x, y)
- recurrence coefficients to compute Q
- matrices R_1, R_2

Use the coefficients to find $Q = [\boldsymbol{q}_1 \cdots \boldsymbol{q}_d]$
$P = [\boldsymbol{p}_1 \cdots, \boldsymbol{p}_d] = QR_1^{-1}R_2^{-1}$

Algorithm 2.6 Computing OPs at any point - Method 2

INPUT:

- (x, y)
- recurrence coefficients to compute Q

Use the coefficients to find $P = [\boldsymbol{p}_1 \cdots \boldsymbol{p}_d]$

2.8 Computational Cost and Efficiency

In this section we analyse the computational cost and efficiency of the two methods proposed to remedy the loss of orthogonality. The computational cost is considered in terms of FLOPs (Floating Points Operations, namely additions, subtractions, multiplications and divisions) required for the computation of the orthogonal polynomials in the nodes. The efficiency is measured by the amount of space used in the memory to store the computed recurrence coefficients and the triangular matrices of the change of basis. All the expressions are obtained using the symbolic operations with Mathematica [67].

2.8.1 Computational cost for computing the OPs in the nodes

Method 1 The first Method as exposed in 2.3 has the computational cost of the basic Algorithm plus the two QR factorizations in economical form. Let M be the

number of nodes and d the dimension of the space of polynomials of degree less then or equal to n.
To produce the recurrence coefficients as in 1.2 we need the following number of operations

$$\sum_{k=1}^{n}\left\{M+\sum_{s=k_0-1}^{k_2-1}(2M-1+2M)+2M+M+\right.$$
$$\left.+\sum_{l=k_1}^{k_2-1}\left(M+\sum_{s=l-k}^{l+k}(2M-1+2M)+2M+M\right)\right\}.$$

Expanding the formula we obtain an expression containing M and n, namely

$$\frac{1}{6}\left(-11n-15n^2-4n^3+80nM+72n^2M+16n^3M\right).$$

Note that the computational cost to produce the OPs using the algorithm 1.2 is asymptotic to $\mathcal{O}(16Mn^3/6)$. Moreover we add: the cost of the two QR factorization [35], which is

$$2\left(Md^2-\frac{2}{3}d^3\right)$$

the cost of the multiplications by the weights matrix W (Md) and the cost of solving two triangular systems, with triangular matrices R_1 and R_2 $(Md^2$ each).
Thus the resulting cost is

$$-\frac{4}{3}+5M-\frac{47}{6}n+\frac{161}{6}Mn-\frac{27}{2}n^2+\frac{51}{2}Mn^2-$$
$$-\frac{67}{6}n^3+\frac{26}{3}Mn^3-\frac{11}{2}n^4+Mn^4-\frac{3}{2}n^5-\frac{1}{6}n^6.$$

Summarizing we proved that the computational cost of the Method 1 proposed has asymptotic behaviour equal to $\mathcal{O}(Mn^4)$.

Method 2 The second Method as proposed in 2.4 has the computational cost of the repeated GS process with respect to the discrete inner product $\langle\cdot,\cdot\rangle_w$. In both versions (with Modified and Classical GS) the cost of the algorithm is computed as

$$\sum_{k=1}^{n}\left\{M+\sum_{s=0}^{k_2-1}(2M-1+2M)+2M+M+\right.$$
$$\left.\sum_{l=k_1}^{k_2-1}\left(M+\sum_{s=0}^{l+k}(2M-1+2M)+2M+M\right)\right\}.$$

Since at each orthogonalization we consider the vector multiplied by W (recall that $\langle\cdot,\cdot\rangle_w=(W\cdot,W\cdot)$), we have to add $2Md$ operations.
Expanding this formula we obtain the following expression

$$2M-\frac{3}{2}n+15Mn-\frac{11}{4}n^2+14Mn^2-\frac{3}{2}n^3+6Mn^3-\frac{1}{4}n^4+Mn^4.$$

This algorithm has asymptotic behaviour equal to $\mathcal{O}(Mn^4)$ too, but looking at the second order term, this latter algorithm appears cheaper.

Despite the computational costs, the timings of the algorithms are quite different. We display in Table 2.4 the time spent for computing the OPs in the nodes with the Method 1 and Method 2 (both Modified and Classical GS variants).
The fact that the Classical GS allows block-columns operations makes the Method 2 more suitable for Matlab implementation. As regarding the orthogonality it is better preserved in the variants with the Modified Gram-Schmidt algorithm, but also in the case of Classical GS and Method 1 the level of orthogonality is satisfactory.

(a) Time to compute OPs in the nodes

DEG	Method 1	Method 2 (MGS)	Method 2 (CGS)
5	$5.018330e-03$	$1.507264e-02$	$2.815298e-02$
10	$1.558499e-02$	$5.349547e-02$	$1.880150e-02$
15	$7.597983e-02$	$2.258162e-01$	$3.051189e-02$
20	$1.559336e-01$	$6.808208e-01$	$4.496751e-02$
25	$2.510725e-01$	$2.105556e+00$	$1.446769e-01$
30	$6.432010e-01$	$4.797996e+00$	$1.487998e-01$
35	$1.897586e+00$	$5.660613e+00$	$4.047092e-01$
40	$3.402484e+00$	$1.099150e+01$	$1.715695e+00$
45	$7.130280e+00$	$4.458592e+01$	$5.102558e+00$
50	$1.265740e+01$	$8.806264e+01$	$9.958521e+00$
55	$2.017278e+01$	$1.091873e+02$	$1.574774e+01$
60	$3.101888e+01$	$2.163251e+02$	$1.832612e+01$

(b) Orthogonality of $Q = WP$

DEG	Method 1	Method 2 (MGS)	Method 2 (CGS)
5	$6.661338e-16$	$2.220446e-16$	$4.440892e-16$
10	$1.110223e-15$	$6.661338e-16$	$6.661338e-16$
15	$1.554312e-15$	$8.881784e-16$	$8.881784e-16$
20	$1.998401e-15$	$8.881784e-16$	$1.110223e-15$
25	$1.776357e-15$	$1.554312e-15$	$8.881784e-16$
30	$2.220446e-15$	$1.332268e-15$	$8.881784e-16$
35	$2.664535e-15$	$1.110223e-15$	$1.110223e-15$
40	$2.886580e-15$	$1.110223e-15$	$1.110223e-15$
45	$3.330669e-15$	$1.332268e-15$	$1.332268e-15$
50	$4.662937e-15$	$8.881784e-16$	$9.228729e-16$
55	$3.996803e-15$	$1.332268e-15$	$8.881784e-16$
60	$5.773160e-15$	$1.554312e-15$	$1.554312e-15$

Table 2.4: Quadrangle of vertices $\{(-1,-1),(2,0),(1,1),(-2,2)\}$ with non-uniform weights. Comparison among various methods to compute OPs. Measures of elapsed time in seconds for computing the OPs in the nodes. Orthogonality of the computed matrix in the $||\cdot||_\infty$ norm.

2.8.2 Computational cost for evaluating the OPs

Method 1 The first Method as exposed in 2.5 has the computational cost of reproducing the basic Algorithm plus the cost of the multiplication by the upper triangular matrices R_1^{-1}, R_2^{-1}. Let H be the number of points at which we want to evaluate the

d polynomials of degree less then or equal to n.
To reproduce the polynomials using the recurrence coefficients we need the following number of operations (we skip the computation of the coefficients)

$$\sum_{k=1}^{n}\left\{H+\sum_{s=k_0-1}^{k_2-1}(2H)+H+\sum_{l=k_1}^{k_2-1}\left(H+\sum_{s=l-k}^{l+k}(2H)+H\right)\right\}.$$

Expanding the formula we obtain an expression containing M and n, namely

$$\frac{20}{3}Mn+6Mn^2+\frac{4}{3}Mn^3.$$

Note that we save about half of the operations used to compute the OPs in the nodes. Moreover we add the cost of the two matrix multiplications, i.e. $2Hd^2$. Thus the resulting cost is

$$2H+\frac{38}{3}Hn+\frac{25}{2}Hn^2+\frac{13}{3}Hn^3+\frac{1}{2}Hn^4$$

We showed that the computational cost for evaluating the OPs, after producing the recurrence coefficients and the inverse triangular matrices has asymptotic behaviour equal to $\mathcal{O}(Mn^4)$.

Method 2 The second Method as proposed in 2.6 evaluate the OPs up to degree n in H points and has different computational cost. In both versions (with Modified and Classical GS) the algorithm costs can be computed as

$$\sum_{k=1}^{n}\left\{H+\sum_{s=0}^{k_2-1}(2H)+H+\sum_{l=k_1}^{k_2-1}\left(H+\sum_{s=0}^{l+k}(2H)+H\right)\right\}.$$

Expanding this formula we obtain the following expression

$$-\frac{3}{2}n+6Hn-\frac{11}{4}n^2+\frac{13}{2}Hn^2-\frac{3}{2}n^3+3Hn^3-\frac{1}{4}n^4+\frac{1}{2}Hn^4$$

This algorithm has asymptotic behaviour equal to $\mathcal{O}(Hn^4)$, but the number of operations needed are half of the ones needed to compute OPs in the nodes.
Again comparing the second order term this second algorithm appear more convenient.

Even in the case of evaluation of the polynomials, we consider the timings to have a concrete idea about the quality of the proposed Methods. To this end we compute the recurrence coefficients and (possibly) the triangular matrices for degrees 10, 20, 30 and 40 and then use these values to evaluate polynomials in an increasing number of points. The results are presented in the Table 2.5, again the block columns implementation of Classical GS algorithm causes the Method 2 with CGS to be the best choice. Note that the difference become smaller as the degree and the number of points increase. To explain this fact consider that the Matlab command `/` to compute the multiplication by the triangular inverse matrices are optimized causing a better usage of memory and faster computations.

(a) Degree 10

H	Method 1	Method 2 (MGS)	Method 2 (CGS)
10	$5.165458e-02$	$1.261318e-01$	$1.904451e-02$
10^2	$3.585983e-02$	$1.307626e-01$	$2.249871e-02$
10^3	$7.163346e-02$	$2.064558e-01$	$3.839979e-02$
10^4	$2.891325e-01$	$9.961866e-01$	$1.917382e-01$

(b) Degree 20

H	Method 1	Method 2 (MGS)	Method 2 (CGS)
10	$2.176916e-01$	$1.620216e+00$	$5.352824e-02$
10^2	$2.412967e-01$	$1.633113e+00$	$6.354765e-02$
10^3	$3.949167e-01$	$2.498941e+00$	$1.889257e-01$
10^4	$1.988462e+00$	$1.042253e+01$	$1.699664e+00$

(c) Degree 30

H	Method 1	Method 2 (MGS)	Method 2 (CGS)
10	$6.352388e-01$	$7.290339e+00$	$1.197903e-01$
10^2	$6.930041e-01$	$8.252970e+00$	$1.690478e-01$
10^3	$1.127305e+00$	$1.471608e+01$	$7.668499e-01$
10^4	$5.034270e+00$	$5.791837e+01$	$7.076889e+00$

(d) Degree 40

H	Method 1	Method 2 (MGS)	Method 2 (CGS)
10	$1.786669e+00$	$2.536993e+01$	$2.496776e-01$
10^2	$1.810302e+00$	$2.445212e+01$	$3.948120e-01$
10^3	$2.686948e+00$	$3.925531e+01$	$2.094131e+00$
10^4	$1.360306e+01$	$1.476062e+02$	$2.245733e+01$

Table 2.5: Quadrangle of vertices $\{(-1,-1),(2,0),(1,1),(-2,2)\}$ with non-uniform weights. Comparison among various methods to evaluate OPs. Measures of elapsed time in seconds in the subset of first H nodes taken from a degree 120 WAM on the quadrangle.

2.8.3 Efficiency

Consider now the quantity of floating numbers that we need to store to evaluate the orthogonal polynomials.
For the first Method we have to store the coefficients αs and βs, their number is

$$\frac{1}{3}\left(10n + 9n^2 + 2n^3\right).$$

Moreover we have to store the triangular matrices, which are of dimension $d \times d$ each, this means that we need to save

$$1 + \frac{19}{3}n + \frac{25}{4}n^2 + \frac{13}{6}n^3 + \frac{1}{4}n^4.$$

For the second Method we need to store the coefficients of both the orthogonalizations, this means storing

$$3n + \frac{13}{4}n^2 + \frac{3}{2}n^3 + \frac{1}{4}n^4$$

numbers.
In both ways the space to store the coefficients are of order $\mathcal{O}(n^4/4)$; one more time the second Method appears better then the first, looking at the second order term.

2.9 Stability

In this section we observe the response of the various method proposed to small perturbations of the nodes and the weights.
The analysis is done on a numerical point of view. Suppose the nodes and the weights are affected by an error of order ε. We denote the new weights by $\{\hat{w}_j\}_{j=1}^{M}$ and the new nodes by $\{(\hat{x}_j, \hat{y}_j)\}_{j=1}^{M}$. Let $\hat{P}$ be the matrix obtained by one of the orthogonalization methods proposed using the perturbed nodes and weights and let P be the matrix obtained using the original nodes and weights.
We produce artificially the nodes and weights affected by errors by adding to the original quantity a random error of the desired order, namely of order ε. We consider as index of stability the distance $||\hat{P} - P||_\infty$.

(a) $\varepsilon = 10^{-16}$

DEG	Method 1	Method 2 (MGS)	Method 2 (CGS)
5	$2.553513e-15$	$3.164136e-15$	$2.775558e-15$
10	$2.114975e-13$	$2.741141e-13$	$2.023243e-13$
15	$4.477965e-11$	$6.235895e-11$	$5.537974e-11$
20	$7.939116e-09$	$4.108985e-08$	$1.909450e-08$
25	$1.337670e-06$	$1.144611e-03$	$6.975289e-04$
30	$3.898145e-04$	$4.874856e-01$	$7.569423e-01$
35	$6.715763e-01$	$7.750173e-01$	$7.502276e-01$
40	$6.688278e-01$	$1.649483e+00$	$2.249518e+00$
45	$9.226162e-01$	$2.024141e+00$	$2.366285e+00$
50	$1.220909e+00$	$1.206194e+00$	$1.155290e+00$
55	$2.950014e+00$	$5.475262e+00$	$4.161549e+00$
60	$3.241659e+00$	$3.066390e+00$	$2.686518e+00$

(b) $\varepsilon = 10^{-14}$

DEG	Method 1	Method 2 (MGS)	Method 2 (CGS)
5	$1.687539e-13$	$1.705303e-13$	$1.690315e-13$
10	$4.208300e-13$	$4.079515e-13$	$4.349299e-13$
15	$2.815970e-11$	$8.134909e-11$	$5.330633e-11$
20	$7.186487e-09$	$4.256721e-08$	$1.559578e-08$
25	$1.181491e-06$	$9.274916e-04$	$8.317864e-04$
30	$5.150848e-04$	$8.569726e-01$	$5.864114e-01$
35	$6.731862e-01$	$7.312204e-01$	$7.179732e-01$
40	$7.231414e-01$	$1.183414e+00$	$1.403269e+00$
45	$1.289464e+00$	$1.511759e+00$	$1.634894e+00$
50	$1.314245e+00$	$1.291680e+00$	$1.389851e+00$
55	$2.899910e+00$	$6.452437e+00$	$4.091764e+00$
60	$2.727459e+00$	$3.132570e+00$	$2.404839e+00$

Table 2.6: Quadrangle of vertices $\{(-1,-1),(2,0),(1,1),(-2,2)\}$ with non-uniform weights. We add to the nodes coordinates and weights a vector of order ε. Measures of $||\hat{P}-P||_\infty$ for the various methods to compute OPs.

Chapter 3

Hyperinterpolation over some planar regions

In this chapter we present a generalization of polynomial interpolation. Given a continuous function f over a rather general planar region, hyperinterpolation is a linear approximation that uses values of f on a well chosen finite set. The approximation is a discrete least-squares approximation constructed with the aid of a high-order cubature rule. For a rule exact on polynomials of suitably high degree ($2n$ or greater), the L^2 error of the approximation is bounded by a constant factor of the error of best uniform approximation by polynomials of degree less then or equal to n. Therefore L^2 error converges to zero as the degree of the approximating polynomial approaches infinity. The theoretical background of this chapter can be found in [55].
After presenting the main result of the paper [55], we use the algorithm to produce OPs presented in the previous chapter to build the hyperinterpolant over non-standard domains. We are presenting hyperinterpolation over some planar regions, for which Gaussian rules of high-degree are known, namely circular sections and polygons.

3.1 Hyperinterpolation: Setting and Results

Let Ω be a compact region of $\mathbb{R}^2$. The region Ω is assumed to be a closure of a connected open domain of $\mathbb{R}^2$ with finite measure with respect to some positive measure $d\mu$, that is to say

$$\mu(\Omega) = \int_\Omega d\mu < \infty.$$

We wish to approximate $f \in C(\Omega)$ by a real bivariate polynomial of degree less then or equal to n, $L_n f$.
Assume that we are given also a positive cubature formula of polynomial degree of exactness at least $2n$. This means that there exist a finite set of points and related positive weights

$$\boldsymbol{x}_k \in \Omega \quad \text{and} \quad w_k > 0 \qquad 1 \le k \le M = M_{2n}, \tag{3.1}$$

with the property that

$$\sum_{k=1}^{M} w_k g(\boldsymbol{x}_k) = \int_\Omega g d\mu \quad \forall g \in \mathbb{P}^2_{2n}(\Omega). \tag{3.2}$$

The degree of exactness $2n$ of the cubature rule guarantees that the rule is exact over polynomials of degree less then or equal to $2n$.

Let $d = d_n = \dfrac{(n+2)(n+1)}{2}$ the dimension of $\mathbb{P}_n^2$. Consider a basis of $\mathbb{P}_n^2$

$$\{\phi_1, \dots \phi_d\} \in \mathbb{P}_n^2,$$

which is orthonormal in $L^2_{d\mu}(\Omega)$, that is

$$\int_\Omega \phi_i \, \phi_j d\mu = \delta_{i,j} \qquad 1 \leq i, j \leq d. \tag{3.3}$$

The sets of nodes and weights from the cubature rule induces a discrete weighted inner product $\langle \cdot, \cdot \rangle_w$ defined by

$$\langle f, g \rangle_w \equiv \sum_{k=1}^{M} w_k f(\boldsymbol{x}_k) g(\boldsymbol{x}_k). \tag{3.4}$$

Thanks to the degree of exactness $2n$, the basis $\{\phi_1, \dots \phi_d\}$ is orthonormal also with respect to this inner product. Indeed

$$\langle \phi_i, \phi_j \rangle_w = \sum_{k=1}^{M} w_k \phi_i(\boldsymbol{x}_k) \phi_j(\boldsymbol{x}_k) = \int_\Omega \phi_i \, \phi_j d\mu = \delta_{i,j}. \tag{3.5}$$

Note that also the converse is true: a polynomial basis orthonormal with respect to the discrete inner product it is orthonormal also in $L^2_{d\mu}(\Omega)$. It is natural to define the approximation $L_n f$ of f as the orthogonal projection of f onto $\mathbb{P}_n^2(\Omega)$, with respect to the discrete inner product $\langle \cdot, \cdot \rangle_w$.
Then $L_n f$, the approximation of f, is defined as

$$L_n f \equiv \sum_{j=1}^{d} c_j \phi_j, \quad c_j = \langle f, \phi_j \rangle_w. \tag{3.6}$$

Later on we prove that $L_n f$ is a good approximation to f. Asking the exactness up to degree $2n$ polynomials does not come for free and the resulting lower bound on the number M of quadrature points is presented in the next lemma [48].

Lemma 2. *If a quadrature rule is exact for all polynomials of degree less then or equal to $2n$, the number of quadrature points M satisfies $M \geq d$.*

Proof. Let Q be the $d \times m$ matrix with elements $q_{jk} = w_k^{1/2} p_j(x_k)$. Then (3.5) asserts that the rows of Q are orthogonal, and hence linearly independent, so that $\operatorname{rank}(Q) = d$. Since $\operatorname{rank}(Q) \leq M$, the result follows. □

Other lower bounds on the number of quadrature points are known for special situations. For a summary on these result see [49]. An M-point quadrature rule that is exact for all polynomials of degree less then or equal to $2n$ is said to be minimal if $M = d$.

The following lemma asserts that $L_n f = f$ if f is a polynomial of degree less then or equal to n, meaning that the approximation is exact for polynomials in $\mathbb{P}_n^2$.

Lemma 3. *If $f \in \mathbb{P}_n^2(\Omega)$, then $L_n f = f$.*

Proof. Since $f \in \mathbb{P}_n^2$ can be written as

$$f = \sum_{j=1}^{d} a_j \phi_j,$$

definition (3.6) yields

$$L_n f = \sum_{j=1}^{d} \langle \sum_{i=1}^{d} a_i \phi_i, \phi_j \rangle_w \phi_j = \sum_{j=1}^{d} \sum_{i=1}^{d} a_i \langle \phi_i, \phi_j \rangle_w \phi_j = \sum_{j=1}^{d} a_j \phi_j = f.$$

□

We are now ready to state the main result of the paper [55]. The norms are defined as usual

$$||g||_2 \equiv \left(\int_\Omega g^2 d\mu \right)^{1/2}, \quad g \in L^2_{d\mu}(\Omega)$$

and

$$||g||_\Omega \equiv \max_{\boldsymbol{x} \in \Omega} |g(\boldsymbol{x})|, \quad g \in C(\Omega).$$

Moreover we define $E_n(g; \Omega)$ to be the error of best uniform approximation of g by an element of $\mathbb{P}_n^2$ as

$$E_n(g; \Omega) \equiv \inf_{\chi \in \mathbb{P}_n^2} ||g - \chi||_\Omega, \quad g \in C(\Omega). \tag{3.7}$$

Theorem 4. *Given $f \in C(\Omega)$, let $L_n f \in \mathbb{P}_n^2$ be defined by (3.6), where the quadrature points and weights in the discrete inner product satisfy (3.1) and the quadrature rule is exact on polynomials of degree up to $2n$. Then*

$$||L_n f - f||_2 \leq 2\mu(\Omega)^{1/2} E_n(f; \Omega). \tag{3.8}$$

Thus $||L_n f - f||_2 \to 0$ as $n \to \infty$.

Proof. Under the hypothesis of the statement, let us prove that

$$\langle L_n f, L_n f \rangle_w \leq \langle f, f \rangle_w. \tag{3.9}$$

Write any polynomial $\chi \in \mathbb{P}_n^2$ as $\chi = \sum_{j=1}^{d} a_j \phi_j$. From definition 3.6 and discrete orthogonality relations among ϕ_i follows that $f - L_n f$ is orthogonal to the space $\mathbb{P}_n^2$. Indeed

$$\begin{aligned} \langle f - L_n f, \chi \rangle_w &= \sum_{j=1}^{d} a_j \langle f, \phi_j \rangle_w - \sum_{j=1}^{d} a_j \langle L_n f, \phi_j \rangle_w = \\ &= \sum_{j=1}^{d} a_j \langle f, \phi_j \rangle_w - \sum_{j=1}^{d} a_j \langle \sum_{i=1}^{d} \langle f, \phi_i \rangle_w \phi_i, \phi_j \rangle_w = 0. \end{aligned}$$

Since $L_n f \in \mathbb{P}_n^2$ and $\langle \cdot, \cdot \rangle_w$ is bilinear we get $\langle L_n f, L_n f \rangle_w = \langle f, L_n f \rangle_w$.
We finally obtain (3.9) by positivity and linearity of $\langle \cdot, \cdot \rangle_w$.

We now show that

$$||L_n f||_2 \leq \mu(\Omega)^{1/2} ||f||_\Omega. \tag{3.10}$$

Consider

$$||L_n f||_2^2 = \int_\Omega (L_n f)^2 d\mu = \langle L_n f, L_n f\rangle_w \leq \langle f, f\rangle_w = \\ = \sum_{k=1}^m w_k f(\boldsymbol{x}_k)^2 \leq \sum_{k=1}^m w_k ||f||_\Omega^2 = \mu(\Omega)||f||_\Omega^2,$$

where in the second equality we used the rule (3.2), which is applicable because $(L_f)^2 \in \mathbb{P}_{2n}^2$, then we used (3.9). In the last step we used again (3.2), with $g = 1$. The error bound (3.8) is finally achieved by the next argument. For any $\chi \in \mathbb{P}_n^2$, we have by (3.10),

$$||L_n f - f||_2 = ||L_n(f-\chi) - (f-\chi)||_2 \leq ||L_n(f-\chi)||_2 + ||f-\chi||_2 \leq \\ \leq \mu(\Omega)^{1/2}||f-\chi||_\Omega + \mu(\Omega)^{1/2}||f-\chi||_\Omega = 2\mu(\Omega)^{1/2}||f-\chi||_\Omega.$$

It follows, taking the $\inf_{\chi\in\mathbb{P}_n^2}$, that

$$||L_n f - f||_2 \leq 2\mu(\Omega)^{1/2} \inf_{\chi\in\mathbb{P}_n^2} ||f-\chi||_\Omega = 2\mu(\Omega)^{1/2} E_n(f;\Omega).$$

□

A relevant quantity is also the uniform norm of $L_n : C(\Omega) \to \mathbb{P}_n^2(\Omega)$, that is the operator norm with respect to $||f||_\Omega = \max_{\boldsymbol{x}\in\Omega} |f(\boldsymbol{x})|$. Observing that

$$L_n f(\boldsymbol{x}) = \sum_{j=1}^d c_j \phi_j(\boldsymbol{x}) = \sum_{j=1}^d \phi_j(\boldsymbol{x}) \sum_{i=1}^M w_i \phi_i(\boldsymbol{x}_i) f(\boldsymbol{x}_i) = \sum_{i=1}^M f(\boldsymbol{x}_i)\Psi_i(\boldsymbol{x}),$$

where

$$\Psi_i(\boldsymbol{x}) = K_n(\boldsymbol{x}, \boldsymbol{x}_i) = w_i \sum_{j=1}^d \phi_j(\boldsymbol{x})\phi_j(\boldsymbol{x}_i),$$

and $K_n(\boldsymbol{x}, \boldsymbol{y})$ is the reproducing kernel of $\mathbb{P}_n^2(\Omega)$, with the underlying inner product [21], one can prove that

$$||L_n|| = \sup \frac{||L_n f||_\Omega}{||f||_\Omega} = \max_{\boldsymbol{x}\in\Omega} \sum_{i=1}^M |\Psi_i(\boldsymbol{x})|. \tag{3.11}$$

Theoretical or numerical bounds for (3.11) allow to estimate the hyperinterpolation convergence rate

$$||L_n f - f||_\Omega \leq (1 + ||L_n||)\, E_n(f;\Omega), \tag{3.12}$$

as well as to study the response of hyperinterpolation to perturbations of the sampled values, that is its stability.

3.2 Positive Cubature Rules

A key feature for hyperinterpolation is the availability of a positive cubature formula on the given compact domain $\Omega \in \mathbb{R}^2$ exact up to degree $2n$ and with as least nodes as possible, in view of efficiency. Indeed all the matrix operations involved have computational complexity proportional to the number of nodes (see Chapter 2 for

details). The positivity is mandatory for the quadrature rule to generate a discrete inner product as defined in (3.4).
Unfortunately, minimal cubature formulas, are known in few multivariate instances, and mainly in standard geometries, see [16] and references therein.
In [54] they presented two classes of integrals for which Mysovskikh's characterization holds and to extend the one-dimensional results directly to the bivariate case. The integrals discussed allow an explicit computation of minimal formulae of an arbitrary even or odd degree of exactness. The domain bounded by two lines and a parabola presented there is used in [70] to build minimal cubature rules in the square $[-1,1]^2$. Other exact Gaussian formulas on the hexagon and triangle are presented in [45].

In recent years some attention has been devoted to the computation of minimal or near minimal formulas on virtually arbitrary geometries by suitable optimization algorithms, which are also able to impose positivity constraint [62],[68]. Cubature formulas with a given polynomial degree of exactness and cardinality equal to the dimension of the corresponding polynomial space can be constructed by discrete extremal sets on various non standard geometries (e.g. polygons), but in general, though numerically stable, they present some negative weights; see [32],[11] for the computation of multivariate extremal sets of Fekete and Leya type.

On the other hand, several positive quadrature formulas based on product Gaussian quadrature have been obtained for the ordinary area or surface measure on standard geometries, via suitable geometric transformation; see [23],[24],[26]. We present here some examples that exploit such product type formulas.

3.2.1 Circular Sections

In [9],[24] trigonometric interpolation and quadrature on subintervals of the period have been studied: in particular, "subperiodic" Gaussian-type quadrature formulas have been implemented. In [31], the quadrature problem has been inserted in the more general framework of sub-range Jacobi polynomials. These results have opened the way to the generation of product Gaussian quadrature formulas for the area measure on several domains constructed by circular arcs, such as circular segments, sectors, zones, and more general regions obtained by linear blending or circular and elliptical arcs.
The basic idea is that such regions are image of a rectangle by a suitable injective analytic transformation, say $\omega : \mathcal{R} \to \Omega$, whose components belong to the tensor-product space $\mathbb{P}^1 \bigotimes \mathbb{T}^1$ ($\mathbb{T}^d_k$ denotes the space of trigonometric polynomials of degree not exceeding k in d variables), and thus any polynomial $p \in \mathbb{P}^2_n(\Omega)$ corresponds to a tensor algebraic-trigonometric polynomial $p \circ \omega \in \mathbb{P}_n \bigotimes \mathbb{T}_n$. By a change of integration variables

$$\int_\Omega p(\boldsymbol{x})d\boldsymbol{x} = \int_{\mathcal{R}} p(\omega(\boldsymbol{y}))|\det J_\omega(\boldsymbol{y})|d\boldsymbol{y},$$

we can obtain a product formula, observing that $\det J_\omega \in \mathbb{P}^1 \bigotimes \mathbb{T}^1$ and has constant sign. For details and examples we refer to [23],[26].

The method has been also applied to regions related to a couple of overlapping disks with possibly different radii, such as lenses (intersection), double bubbles (union) and lunes (difference), see [24],[25]. The cardinality of such formulas for exactness degree n is $cn^2 + \mathcal{O}(n)$, with $c = 1$ or $c = 1/2$ (and even $c = 1/4$ by symmetry arguments in the special case of circular segments).
In the Table 3.1 we present a summary on the planar regions and the software available for this kind of formulas for a rather general circular sections.

Circular regions	Matlab code	Reference
circular sectors	`gqcircsect.m`	[26]
circular segments	`gqcircsegm.m`	[26]
circular zones	`gqcirczones.m`	[26]
symmetric lenses	`gqsymmlens.m`	[26]
bubbles	`gqdbubble.m`	[24]
lunes	`gqlune.m`	[25]
lenses	`gqlens.m`	[24]
linear blending of elliptical arcs	`gqellblend.m`	[23]

Table 3.1: Overview of the Matlab code available for building the cubature rules over rather general circular Sections, for the source code see [36].

3.2.2 Polygons

In [60] a triangulation-free positive cubature formual for the standard area measure on a wide class of polygons was presented. The formula, implemented in by the Matlab function `Polygauss.m` [36], is based on product Gaussian quadrature by decomposition into trapezoidal panels, and works on all convex polygons, as well as on a class of non-convex polygons with a simple geometric characterization. The cardinality for exactness degree n is bounded by $Ln^2/4+\mathcal{O}(n)$, where L is the number of polygons sides.

3.3 Numerical Examples

As we have seen above, there are two main ingredients: the availability of a positive cubature formula on the compact domain, and the possibility to compute the orthogonal polynomials with respect to $\langle\cdot,\cdot\rangle_w$ in the points of the domain. As regarding this second issue we use the algorithm implemented in the previous Chapter, by Method 2 (CGS).
We build the approximation by the following procedure.

1. Compute weights $\{\sqrt{w_1},\dots,\sqrt{w_M}\}$ and nodes $\{\boldsymbol{x}_1,\dots,\boldsymbol{x}_M\}$ of a cubature formula of degree of exactness at least $2n$ in the desired planar region Ω.
   We use the Software available to experiment hyperinterpolation on circular sections and polygons.

2. Compute the OPs $\mathbb{P}_n^2$ up to degree n with the Algorithm 2.4 with CGS. The discrete inner product to use $\langle\cdot,\cdot\rangle_w$ is the one induced by the cubature formula. We produce the matrix

   $$P = [\phi_j(\boldsymbol{x}_i)] \quad 1 \le j \le d, 1 \le i \le M,$$

   numerically orthogonal with respect to the weighted inner product.

3. Given a continuous function in the region Ω, we evaluate it at the nodes $\boldsymbol{f} = (f(\boldsymbol{x}_1),\dots,f(\boldsymbol{x}_M))^t$.

4. We compute the coefficients $\boldsymbol{c} = (c_1, \dots, c_d)^t$ of the expansion of f in the basis of OPs using the following formula

$$\boldsymbol{c} = P^t \boldsymbol{f},$$

thanks to the orthogonality.

5. Given a point $\boldsymbol{y} \in \Omega$, we can compute the evaluation of the hyperinterpolant at $\boldsymbol{y}$ as

$$L_n f(\boldsymbol{y}) = W\boldsymbol{c},$$

where $W = [\phi_1(\boldsymbol{y}) \dots \phi_d(\boldsymbol{y})]$, is the row vector of the evaluations of the OPs in $\boldsymbol{y}$, produced using the Algorithm 2.6.

3.3.1 Circular Sections

We consider five test functions with different regularity

$$\begin{aligned} f_1(x,y) &= (x+y+2)^{30} \\ f_2(x,y) &= \cos(x+y) \\ f_3(x,y) &= \cos(5x+5y) \\ f_4(x,y) &= ((x-0.5)^2 + (y-0.5))^{3/2} \\ f_5(x,y) &= |x-0.5| + |y-0.5| \end{aligned} \tag{3.13}$$

Note that f_1 is a polynomial, f_2 and f_3 are analytic functions, whereas f_4 belongs to C^3, with singular point $(0.5, 0.5)$ and f_5 belongs to C^1, with the lines $x = 0.5$ and $y = 0.5$ of singular points. Moreover the two analytic functions have different frequency, meaning that the function f_3 is more oscillatory, hence harder to approximate.

The relative reconstruction error by hyperinterpolation is estimated as

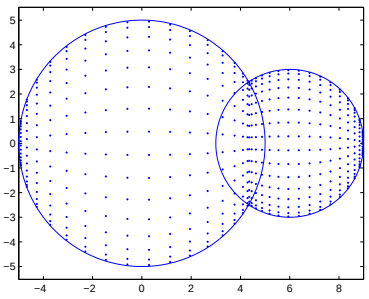
$$\frac{\left(\sum_{i=1}^{M} w_i (L_n f(\boldsymbol{x}_i) - f(\boldsymbol{x}_i))^2\right)^{1/2}}{\left(\sum_{i=1}^{M} w_i (f(\boldsymbol{x}_i))^2\right)^{1/2}} \approx \frac{||L_n f - f||_2}{||f||_2}. \tag{3.14}$$

In Figure 3.1 we show the numerical tests on a double bubble and on a circular zones. Observe that the singular point $(0.5, 0.5)$ of f_4 and the singular points with at least one coordinate equal to 0.5 of f_5 belong to the interior of both regions.
Note that the error on the polynomial f_1 becomes close to the machine precision as soon as exactness reaches the polynomial degree. This is expected, because L_n is a projection operator (Proposition 3). The other errors are influenced by the smoothness level of the functions as expected from (3.8) and the fact that circular sections are "Jackson Compacts". Indeed all these regions, as the other quoted above, admit a multivariate Jackson inequality, see [23],[25].

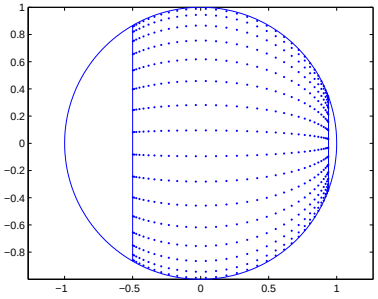
Recall that a compact $\Omega \in \mathbb{R}^2$ which is a closure of an open bounded set, is termed a Jackson Compact if it admits a Jackson inequality, namely for each $k \in \mathbb{N}$ there exist a positive constant c_k such that

$$n^k E_n(f;\Omega) \le c_k \sum_{|\boldsymbol{i}| \le m_k} ||D^{\boldsymbol{i}} f||_\Omega, \; n > k, \quad \forall f \in C^{m_k}(\Omega) \tag{3.15}$$

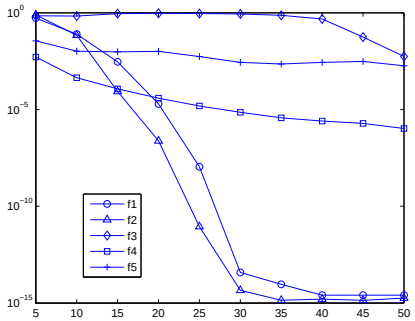
We refer to [51] for a recent survey on the multivariate Jackson inequality.



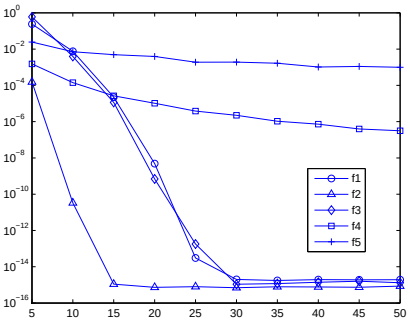
(a) Double Bubble



(b) Circular zone



(c) Hyperinterpolation Error



(d) Hyperinterpolation Error.

Figure 3.1: Circular Domains and Hyperinterpolation error for test functions f_i, $i = 1, 2, 3, 4, 5$.

In Figure 3.2 we report the uniform norm of the hyperinterpolation operator as a function of the degree n, for the two examples. We take as control mesh the cubature nodes for exactness degree $4n$, say the points $\mathcal{T} = \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_H\}$. Thus we estimate the operator norm computing

$$||L_n|| \approx \max_{\boldsymbol{x} \in \mathcal{T}} \sum_{i=1}^{M} |\Psi_i(\boldsymbol{x})|. \tag{3.16}$$

It is worth stressing that the increase rate experimentally observed $||L_n|| = \mathcal{O}(n \log n)$ is not surprising concerning the circular sections, since it is compatibe with known theoretical estimates for hyperinterpolation over the whole disk [37].

OPs in the cubature points Consider in the sequel the computation of OPs in the double bubble and in the circular zone. In the Table 3.2 we report for degree $n = 5, 10, \ldots, 50$ the cardinality M of the set of nodes required to make the cubature rule has degree of exactness $2n$. We also consider the orthogonality of the resulting matrix and the time elapsed for the computation of the matrix P (using the Method 2 with the CGS variant). Note that this is the heaviest part of the computational cost.

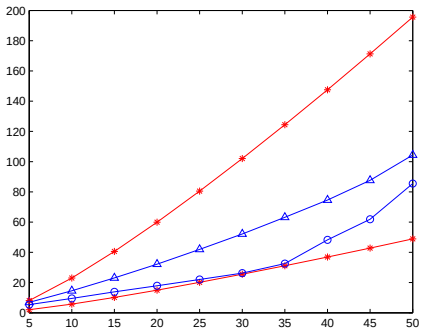


Figure 3.2: Numerically evaluated uniform norms of the hyperinterpolation operator for degree $n = 5, 10, \ldots, 60$ on the examples of bubbles ($\circ$) and circular zone ($\triangle$) compared with $n \log n$ and $\frac{1}{4} n \log n$ (upper and lower ($*$) curves).

3.3.2 Polygons

Consider the test polygons on the top of Figure 3.3, the first is convex, the second non-convex. We analyse the relative hyperinterpolation error estimates for the test function (3.13). Note that the singular points of f_4 and f_5 are still contained in the interior of the two domains.
In Figure 3.3 (bottom) we show the estimated relative error approximating f_i with $L_n f_i$ computed as in (3.14).
Note that the error, if the function to approximate is smooth enough, is close to the machine level with degree 20. This is explained by the high cardinality of the nodes in the Gaussian-type rule for polygons: such large sets impose also a large sampling of the function, reducing the errors.
Again the errors are influenced by the regularity of the test function. Moreover the polygons are Jackson compacts, being a finite union of triangles.

Since the cardinality of nodes is sufficiently large, we compute the norm of the hyperinterpolation operator taking $\mathcal{T} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_M\}$ and using the estimate in (3.16). In the Figure 3.4 we see that the operator norm grows as $\mathcal{O}(n \log n)$.

OPs in the cubature points Consider in the sequel the computation of OPs in the convex and non-convex polygons above. The large cardinality of the cubature nodes on the polygons forces to stop the computations at degree equal to 30. In the Table we report for degree $n = 5, 10, \ldots, 30$ the cardinality M of the set of nodes required to make the cubature rule has degree of exactness $2n$. We also consider the orthogonality of the resulting matrix and the time elapsed for the computation of the matrix P (with the Method 2 with the CGS variant). Note that this is the overall computing cost.

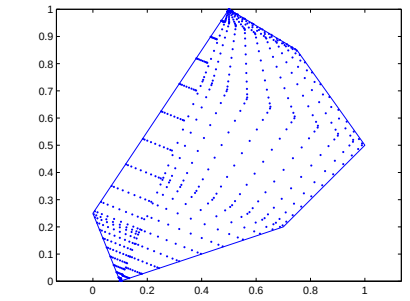
(a) Double Bubble

n	M	H	orthogonality	time
5	72	242	$4.440892e-16$	$1.259918e-02$
10	242	882	$1.110223e-15$	$7.477789e+00$
15	512	1922	$8.881784e-16$	$2.068398e+01$
20	882	3362	$2.220446e-15$	$3.013147e+01$
25	1352	5202	$1.332268e-15$	$4.850831e+01$
30	1922	7442	$1.554312e-15$	$7.707397e+01$
35	2592	10082	$1.110223e-15$	$1.231661e+02$
40	3362	13122	$8.881784e-16$	$1.726235e+02$
45	4232	16562	$6.439294e-15$	$2.301095e+02$
50	5202	20402	$1.001369e-15$	$2.624078e+02$

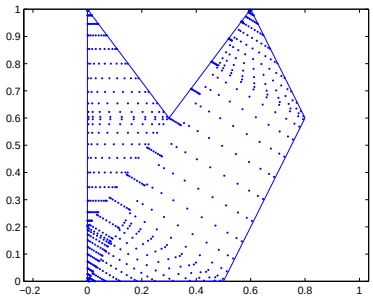
(b) Circular zone

n	M	H	orthogonality	time
5	78	253	$6.661338e-16$	$1.853134e-03$
10	253	903	$1.110223e-15$	$7.808692e+00$
15	528	1953	$8.881784e-16$	$1.824021e+01$
20	903	3403	$1.332268e-15$	$3.185490e+01$
25	1378	5253	$1.110223e-15$	$5.142113e+01$
30	1953	7503	$1.110223e-15$	$7.059854e+01$
35	2628	10153	$8.396062e-16$	$1.130459e+02$
40	3403	13203	$1.110223e-15$	$1.905071e+02$
45	4278	16653	$1.054712e-15$	$2.233412e+02$
50	5253	20503	$1.294104e-15$	$2.399346e+02$

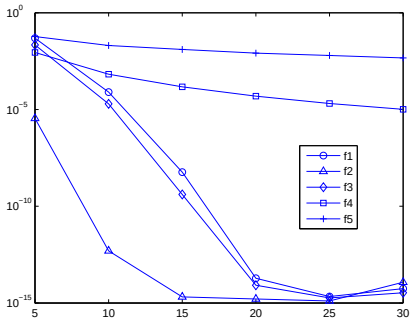
Table 3.2: Computation of OPs up to degree n on M nodes from a cubature rule on circular sections with degree of exactness $2n$. Orthogonality is measured with $||\cdot||_\infty$ norm.



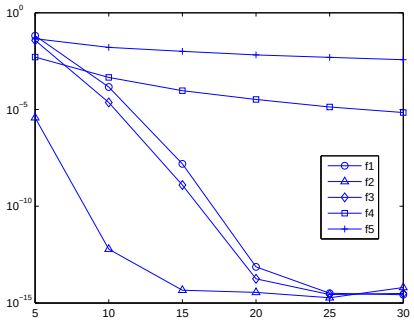
(a) Convex polygon



(b) Non-convex polygon



(c) Hyperinterpolation Error



(d) Hyperinterpolation Error.

Figure 3.3: Polygons and Hyperinterpolation error for test functions f_i, $i = 1, 2, 3, 4, 5$.

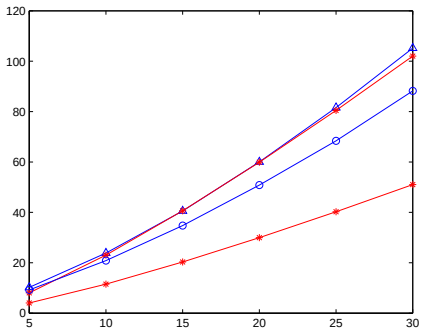


Figure 3.4: Numerically evaluated uniform norms of the hyperinterpolation operator for degree $n = 5, 10, \ldots, 30$ on the examples of a non-convex polygon ($\circ$) and a convex-polygon ($\triangle$) compared with $n \log n$ and $\frac{1}{2} n \log n$ (upper and lower ($*$) curves).

(a) Convex polygon

n	M	orthogonality	time
5	660	$6.661338e-16$	$1.637061e+00$
10	2520	$4.978656e-16$	$1.679057e+01$
15	5580	$6.661338e-16$	$2.342652e+01$
20	9840	$8.881784e-16$	$4.462161e+01$
25	15300	$9.298118e-16$	$5.943143e+01$
30	21960	$8.659740e-15$	$9.171649e+01$

(b) Non-convex polygon

n	M	orthogonality	time
5	770	$6.661338e-16$	$1.719121e+00$
10	2940	$1.110223e-15$	$1.074729e+01$
15	6510	$7.762888e-16$	$2.107101e+01$
20	11480	$8.604228e-16$	$4.540764e+01$
25	17850	$1.557782e-15$	$5.977046e+01$
30	25620	$3.996803e-15$	$8.573632e+01$

Table 3.3: Computation of OPs up to degree n on M nodes from a cubature rule on polygons with degree of exactness $2n$. Orthogonality is measured with $||\cdot||_\infty$ norm.

Chapter 4

Approximating Functions with Bivariate Polynomials

In this Chapter we are presenting the theory of Weakly Admissible Meshes (WAMs) and Discrete Extremal Sets. The idea is to replace a continuous compact set $\Omega \subset \mathbb{R}^2$ by these discrete sets of points belonging to the domain, thus compressing the informations in a minimal set of nodes. We can interpolate or approximate (in the least-squares sense) any continuous function using the function values in the points from these meshes, the properties of WAMs guarantee the success of the least square approximation. Moreover the methods proposed in Chapter 2 for computing OPs allows to produce a Vandermonde-like matrix on the points of the WAM which is not too ill conditioned, from that we can both have a good least-squares approximation and extract points that are good for the interpolation, which we call Discrete Extremal Sets.
The Discrete Extremal Sets have been extensively studied and have interesting asymptotic behaviour, namely they induce a discrete measure that is asymptotic to the pluripotential-theoretic equilibrium measure of Ω (see Theorem 5 for a precise statement). Such a property can be exploited to produce a good conditioned basis of OPs using that points as nodes and with uniform weights.
This theory -together with the theory of discrete orthogonal polynomials- gives new insights in different applications such as numerical cubature, digital filtering and high-order methods for Partial Differential Equations [**add references**].
We first present the general theory of polynomial interpolation, then we introduce the of Weakly Admissible Meshes and their properties. Numerical result on least-squares approximation an Discrete Extremal sets are also presented [36].

4.1 Polynomial Interpolation and Fekete Points

Given a compact set $\Omega \in \mathbb{R}^2$, let $S_N \equiv \text{span}\{\psi_j\}_{1\leq j\leq N}$ be a finite-dimensional space of continuous and linearly independent functions on Ω. Given a continuous function f defined in Ω we want to reconstruct it using the chosen basis S_N. We restrict ourselves to the case in which the ψ_j are polynomials, and we talk about polynomial interpolation. Recall that any continuous function $f \in C(\Omega)$ has sup-norm $||f||_\Omega \equiv \max_{\boldsymbol{x}\in\Omega} |f(\boldsymbol{x})|$.

Consider a set of points $X = \{\boldsymbol{x}_1, \dots, \boldsymbol{x}_N\} \in \Omega$, we can build the Vandermonde matrix

$$V(\boldsymbol{x}_1, \dots, \boldsymbol{x}_N) \equiv [\psi_j(\boldsymbol{x}_i)]_{ij}.$$

It is well known that if $\det(V(\boldsymbol{x}_1, \dots, \boldsymbol{x}_N)) \neq 0$ then the set X is unisolvent for the interpolation on S_N, meaning that two functions assuming the the same values on X coincide on S_N.
Moreover if we define

$$\varphi_j(\boldsymbol{y}) = \frac{\det V(\boldsymbol{x}_1, \dots, \boldsymbol{x}_{j-1}, \boldsymbol{y}, \boldsymbol{x}_{j+1}, \dots, \boldsymbol{x}_N)}{\det V(\boldsymbol{x}_1, \dots, \boldsymbol{x}_{j-1}, \boldsymbol{x}_j, \boldsymbol{x}_{j+1}, \dots, \boldsymbol{x}_N)}, \tag{4.1}$$

for $1 \leq j \leq N$, we have that φ_j is a cardinal basis for the space, i.e. $\varphi_j(\boldsymbol{x}_i) = \delta_{ji}$. It is possible to build the Lagrange interpolation polynomial as

$$L_{S_N} f(\boldsymbol{y}) \equiv \sum_{j=1}^{N} f(\boldsymbol{x}_j)\varphi(\boldsymbol{y}).$$

We can estimate the interpolation error with the following inequality (for an introduction on the theory see [**Mar36**, **Mar27**], for a more general setting see [**Mar33**, **pg 213-215**])

$$||f - L_{S_N} f||_\Omega \leq (1 + \Lambda_N) \inf_{p \in S_N} ||f - p||_\Omega, \tag{4.2}$$

where we denote with Λ_N the Lebesgue constant, defined as

$$\Lambda_n \equiv ||L_{S_N}|| = \max_{\boldsymbol{x} \in \Omega} \sum_{j=1}^{N} |\varphi_j(\boldsymbol{x})|.$$

From the inequality (4.2) is clear that we have to look for unisolvent points that have a minimal Lebesgue constant Λ_N. However looking for such optimal points it is a computationally challenging problem.
For this reason there was an attempt to find set of points for which the growth of Λ_N is not too high, for example looking for the so called Fekete Points.

The concept of Fekete points can be described in a very general, not necessarily polynomial, setting and for dimension greater then 2 [**add reference**]. In the case that the points maximize the absolute value of the denominator of (4.1) in Ω^2 we obtain the Fekete points. The sup-norm $||\varphi_j||_\Omega \leq 1$ for every j and thus the norm of the interpolation operator $L_{S_N} : C(\Omega) \to S_N$ is bounded by the dimension of the interpolation space,

$$\Lambda_N = \max_{\boldsymbol{x} \in \Omega} \sum_{j=1}^{N} |\varphi_j(\boldsymbol{x})| \leq N. \tag{4.3}$$

Fekete points as well as Λ_N are independent of the choice of the basis in S_N, since the determinant of the Vandermonde-like matrices changes by a factor independent on the points.
There are several open problems about Fekete points. They are analytically known only in few instances: interval, complex circle and the cube. They are asymptotically equidistributed with respect to the pluripotential equilibrium measure of Ω [3]. Moreover Fekete points are very difficult to compute, requiring an expensive and numerically challenging nonlinear multivariate optimization. Thus the problem has been solved numerically only in very special cases and for limited range of degrees.

4.2 Weakly Admissible Meshes and Discrete Extremal Sets

Looking for good points for polynomial interpolation leads to the introduction of the (Weakly) Admissible Meshes in [15].
As in the previous Chapter we denote the set of bivariate real polynomials of degree less then or equal to n, restricted to Ω, with the symbol $\mathbb{P}_n^2(\Omega)$. Recall the dimension of $\mathbb{P}_n^2(\Omega)$ equals $d = d_n = \frac{(n+1)(n+2)}{2}$.
Consider a polynomial determining compact set $\Omega \in \mathbb{R}^2$, i.e. polynomials vanishing there are identically zero.
A Weakly Admissible Mesh (WAM) is defined to be a sequence of discrete subsets $\mathcal{A}_n \in \Omega$ such that

$$||p||_\Omega \leq C(\mathcal{A}_n)||p||_{\mathcal{A}_n} \quad \forall p \in \mathbb{P}_n^2(\Omega), \tag{4.4}$$

where both the cardinality of the mesh $\text{card}(\mathcal{A}_n) \geq d_n$ and $C(\mathcal{A}_n)$ grows at most polynomially with n:

$$\begin{aligned} d_n \leq \text{card}(\mathcal{A}_n) = \mathcal{O}(n^\alpha) \quad \alpha > 0 \\ C(\mathcal{A}_n) = \mathcal{O}(n^\beta) \quad \beta > 0 \end{aligned}$$

When $C(\mathcal{A}_n)$ is bounded $C(\mathcal{A}_n) \leq C$, we speak of Admissible Meshes (AM). The adverb "Weakly" refers to the fact that to keep $\text{card}(\mathcal{A}_n)$ smaller we allow a less strict bound on the growth of the constant $C(\mathcal{A}_n)$.

Example 3.

For an overview of WAMs and their properties we refer the reader to [13]. Below we list the main features of WAMs in terms of ten properties:

P1 $C(\mathcal{A}_n)$ is invariant under affine mapping;

P2 any sequence of unisolvent interpolation sets (i.e. they have a unique solution to the interpolation problem) whose Lebesgue constant grows at most polynomially with n is a WAM, $C(\mathcal{A}_n)$ being the Lebesgue constant itself;

P3 any sequence of supersets of a WAM whose cardinalities grow polynomially with n is a WAM with the same constant $C(\mathcal{A}_n)$;

P4 a finite union of WAMs is a WAM for the corresponding union of compacts, $C(\mathcal{A}_n)$ being the maximum of the corresponding constant;

P5 a finite cartesian product of WAMs is a WAM for the corresponding union of compacts, $C(\mathcal{A}_n)$ being the product of the corresponding constants;

P6 in $\mathbb{C}^2$ a WAM of the boundary $\partial\Omega$ is a WAM of Ω (by the maximum principle);

P7 given a polynomial mapping π_s of degree s, then $\pi(\mathcal{A}_{ns})$ is a WAM for $\pi_s(\Omega)$ with constants $C(\mathcal{A}_{ns})$;

P8 any Ω satisfying a Markov polynomial inequality like

$$|| \bigtriangledown p||_\Omega \leq Mn^r||p||_\Omega$$

has an AM with $\mathcal{O}(n^{2r})$ points;

P9 convergence of least-squares polynomial approximation of $f \in C(\Omega)$: the least-squares polynomial $L_{\mathcal{A}_n} f$ on a WAM is such that

$$||f - L_{\mathcal{A}_n} f||_\Omega \lessapprox C(\mathcal{A}_n)\sqrt{\text{card}(\mathcal{A}_n)}E_n(f;\Omega), \tag{4.5}$$

where $E_n(f;\Omega)$ is the error of best uniform approximation of f by an element of $\mathbb{P}_n^2$;

P10 Fekete points: the Lebesgue constant of Fekete points extracted from a WAM can be bounded like $\Lambda_n \leq d_n C(\mathcal{A}_n)$, that is the elementary classical bound of the continuum Fekete points times a factor $C(\mathcal{A}_n)$. Moreover, their asymptotic distribution is the same of the continuum Fekete points, in the sense that the corresponding discrete probability measures converges weak-$*$ to the pluripotential equilibrium measure of Ω.

These ten properties give the basic tools for the construction and application of WAMs in polynomial approximation and interpolation. The present exposition takes advantage mainly of properties **P9** and **P10**, therefore they are re-examined in the next Sections. We are able to build WAMs for different standard domains such as quadrangles, disks and triangles [14],[43]. The properties of WAMs allow to produce WAMs -via quadrangulation and triangulation- also for arbitrary polygons [32]. All the software is available at [36].

4.3 Discrete Least-Squares Approximation on WAMs

Consider a WAM $\mathcal{A}_n \subset \Omega$, with Ω polynomial determining compact set. Let $\mathcal{A}_n = \{\boldsymbol{a}_1, \ldots, \boldsymbol{a}_M\}$ be the mesh of cardinality $M \geq d_n$ and $\boldsymbol{p} = \{\phi_1, \ldots, \phi_d\}$ the polynomial basis, both ordered in some manners. The associated rectangular Vandermonde-like matrix is

$$V = V(\boldsymbol{a}; \boldsymbol{p}) = [\phi_j(\boldsymbol{a}_i)]_{ij} \quad 1 \leq i \leq M, 1 \leq j \leq d. \tag{4.6}$$

Given the continuous function $f \in C(\Omega)$, let $\boldsymbol{f}$ be the vector containing the sampling of f in the nodes, i.e. $\boldsymbol{f} = (f(\boldsymbol{a}_1), \ldots, f(\boldsymbol{a}_M))^t$. Find the approximation is equivalent to compute the coefficients $\boldsymbol{c} = (c_1, \ldots, c_d)^t$ such that

$$V\boldsymbol{c} = \boldsymbol{f}.$$

We consider the least-squares solution of the over-determined linear system, namely the solution that minimizes the residual. For a survey on least-squares numerical methods see [4].
If the matrix V is orthonormal, i.e. $V^tV = I$, then the least square solution can be obtained by

$$\boldsymbol{c} = V^t\boldsymbol{f}.$$

Denoting by $L_{\mathcal{A}_n}$ the discrete least-squares projection operator on the WAM $\mathcal{A}_n$, we can write

$$L_{\mathcal{A}_n} f(\boldsymbol{x}) = \sum_{j=1}^{d} \left(\sum_{i=1}^{M} f(\boldsymbol{a}_i)\phi_i(\boldsymbol{a}_i) \right) \phi_j(\boldsymbol{x}) = \sum_{j=1}^{M} f(\boldsymbol{a}_i)\Psi_i(\boldsymbol{x}), \tag{4.7}$$

where

$$\Psi_i(\boldsymbol{x}) = K_n(\boldsymbol{x}, \boldsymbol{a}_i) \quad i = 1, \ldots, M, \tag{4.8}$$

and $K_n(\boldsymbol{x}, \boldsymbol{y}) = \sum_{j=1}^{d} \phi_j(\boldsymbol{x})\phi_j(\boldsymbol{y})$ is the reproducing kernel corresponding to the discrete inner product. Moreover, the norm of the least-squares operator is given by

$$||L_{\mathcal{A}_n}|| = \max_{\boldsymbol{x}\in\Omega} \sum_{i=1}^{M} |\Psi_i(\boldsymbol{x})|. \tag{4.9}$$

If $\mathcal{T} = \{\boldsymbol{y_1}, \ldots, \boldsymbol{y_H}\}$ is a (possibly) finer mesh then we approximate the uniform norm of the operator by

$$||L_{\mathcal{A}_n}|| \approx \max_{\boldsymbol{x}\in\mathcal{T}} \max_{1\le j\le d} |\phi_j(\boldsymbol{x})|. \tag{4.10}$$

Property **P9** guarantees that the WAMs can be used for least squares approximation with an near-optimal error, up to a factor $\mathcal{O}(n \log^2 n)$. The bound turns out to be a rough overestimate.

4.4 Discrete Extremal Sets

Thanks to the WAMs, we replace the compact set Ω with a discrete set of points. Property **P10** suggests WAMs are good candidates as starting meshes for extracting "Approximate" Fekete Points. Since the rows of the Vandermonde matrix $V = V(\boldsymbol{a}; \boldsymbol{p})$ correspond to the mesh points and the columns to the polynomial basis elements, computing the Approximate Fekete Points for a WAM amounts to select d rows of V such that the volume generated by these rows, i.e., the absolute value of the determinant of the resulting $d \times d$ submatrix is maximum. The points associated with the chosen rows are termed Approximate Fekete Points and denoted by $\boldsymbol{\xi} = \{\boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_d\}$ This problem, however, is known to be NP-hard, so heuristic or stochastic algorithms are mandatory. The volume of a general rectangular matrix V is defined to be $\text{vol}V = \sqrt{\det(A^t A)}$; cf. [40] for the notion of volume generated by a set of vectors. An approximate solution can be given by two different greedy algorithm, which compute what we call "Discrete Extremal Sets" [12],[11]. Below we present the two greedy algorithms in a Matlab-like notation.

Algorithm 4.1 Algorithm greedy 1 (Approximate Fekete Points)

$W = V(\boldsymbol{a}; \boldsymbol{p})^t$; $ind = [\quad]$;
for $k = 1 : d$ **do**
  select the index i_k that maximizes $\text{vol}V([ind, i_k], 1 : d)$
  $ind = [ind, i_k]$
end for
$\boldsymbol{\xi} = \boldsymbol{a}(i_1, \ldots, i_d)$

Algorithm 4.2 Algorithm greedy 2 (Discrete Leja Points)

$W = V(\boldsymbol{a}; \boldsymbol{p})^t$; $ind = [\quad]$;
for $k = 1 : d$ **do**
  select the index i_k that maximizes $|\det V([ind, i_k], 1 : k)|$;
  $ind = [ind, i_k]$
end for
$\boldsymbol{\xi} = \boldsymbol{a}(i_1, \ldots, i_d)$

The two greedy algorithms correspond to basic procedures of numerical linear algebra. The computation of Discrete Extremal Sets can be done by LU factorization with row pivoting of the Vandermonde matrix [53] or QR factorization with column pivoting of the transposed Vandermonde matrix [58]. This is summarized by the following Matlab-like scripts.

Algorithm 4.3 Approximate Fekete Points (AFP)

$W = V(\boldsymbol{a}; \boldsymbol{p})^t$
$\boldsymbol{w} = W \backslash (1, \ldots, 1)^t$
$ind = \texttt{find}(\boldsymbol{w} \neq \boldsymbol{0})$
$\boldsymbol{\xi} = \boldsymbol{a}(ind)$

Algorithm 4.4 Discrete Leja Points (DLP)

$V = V(\boldsymbol{a}; \boldsymbol{p})^t$
$[L, U, \boldsymbol{\sigma}] = \texttt{LU}(V, '\texttt{vector}')$
$ind = \boldsymbol{\sigma}(1, \ldots, d)$
$\boldsymbol{\xi} = \boldsymbol{a}(ind)$

Once the underlying extraction WAM has been fixed, Approximate Fekete Points depend on the choice of the basis, whereas Discrete Leja points depends also on its order. Nevertheless, such Discrete Extremal Sets share the same asymptotic behaviour, exactly that of the continuum Fekete points, as stated in the following theorem

Theorem 5. *Suppose that $\Omega \subset \mathbb{R}^2$ is compact, non-pluripolar, polynomially convex and regular (in the sense of Pluripotential theory) and that for $n \in \mathbb{N} = \{1, 2, \ldots\}$ $\mathcal{A}_n \subset \Omega$ is a WAM. Let $\boldsymbol{\xi} = \{\boldsymbol{\xi_1}, \ldots, \boldsymbol{\xi_d}\}$ be the Approximate Fekete Points selected from $\mathcal{A}_n$ by the greedy Algorithm AFP using any basis $\boldsymbol{p}$, or the Discrete Leja Points selected by $\mathcal{A}_n$ by the greedy algorithm DLP using any basis of the form $\boldsymbol{p} = L\boldsymbol{e}$, where $\boldsymbol{e} = \{\boldsymbol{e_1}, \ldots, \boldsymbol{e_d}\}$ is any ordering of the standard monomials $\boldsymbol{x}^\alpha$ such that $\deg(\boldsymbol{e_j}) \leq \deg(\boldsymbol{e_k})$ for $j \leq k$ that is consistent with the degree and $L \in \mathbb{R}^{d \times d}$ is lower triangular. Then*

- $$\lim_{n \to \infty} |\det V(\boldsymbol{\xi}; \boldsymbol{p})|^{1/m_n} = \tau(\Omega),$$

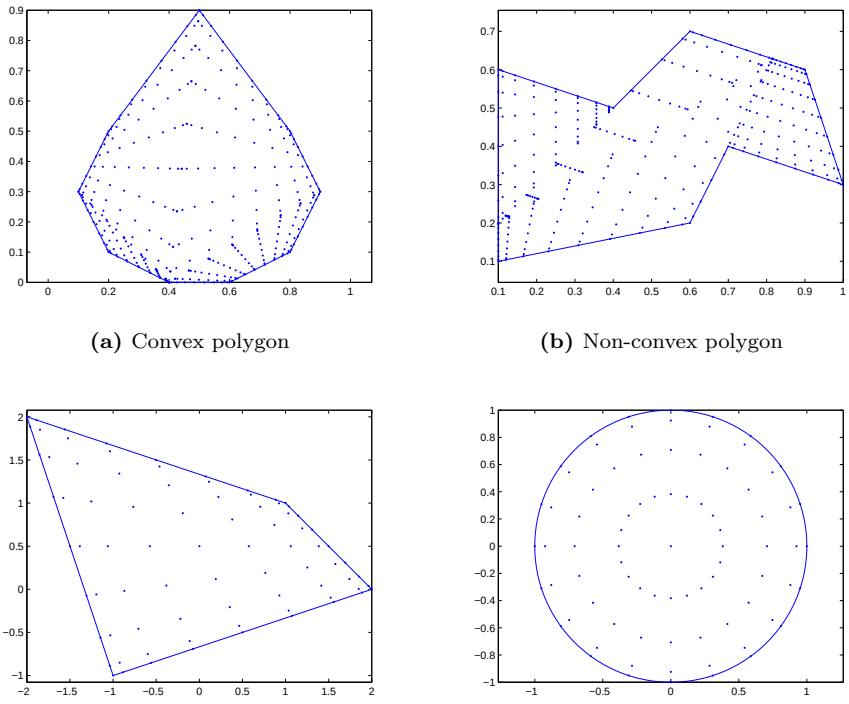
  where $\tau(\Omega)$ is the transfinite diameter of Ω and $m_n = \frac{2nd}{3}$ (the sum of the degrees of the d monomials of degree less then or equal to n)

- *the sequence of discrete probability measures $\mu_n \equiv \frac{1}{d} \sum_{j=1}^{d} \delta_{\boldsymbol{\xi_j}}$ converge to the pluripotential-theoretic equilibrium measure $d\mu_\Omega$ of Ω, in the sense that*

  $$\lim_{n \to \infty} \int_\Omega f(\boldsymbol{x}) d\mu_n = \int_\Omega f(\boldsymbol{x}) d\mu_\Omega,$$

  for every $f \in C(\Omega)$.

As polynomial basis in the WAM we can take the Chebyshev product basis of the minimal rectangle including the domain or the OPs basis computed in Chapter 2. In the next Sections we are experimenting these bases.



(a) Convex polygon

(b) Non-convex polygon

(c) Quadrangle

(d) Unit Disk

Figure 4.1: Domains to test least-squares Approximation and Interpolation. Points from a degree 8 WAM on the domain.

4.5 Numerical Examples

In this section we test the least-squares approximation on WAMs and the interpolation on Approximate Fekete Points or Discrete Leja Points. We are using the following test functions

$$\begin{aligned} f_2(x,y) &= \cos(x+y) \\ f_4(x,y) &= ((x-0.5)^2 + (y-0.5))^{3/2} \end{aligned} \tag{4.11}$$

and some different domains: the quadrangle, two polygons and the unit disk shown in Figure 4.1.

4.5.1 Least-squares Approximation on WAMs

We estimate numerically the error of approximation by computing the maximum error of approximation in the WAM $\mathcal{A}_n = \{\boldsymbol{a}_1, \ldots, \boldsymbol{a}_M\}$

$$\max_{\boldsymbol{x} \in \mathcal{A}_n} |L_{\mathcal{A}_n} f(\boldsymbol{x}) - f(\boldsymbol{x})| \tag{4.12}$$

n	5	10	15	20	25	30	35	40	45	50
convex polygon	3.9	4.8	5.9	6.4	6.9	9.4	13.9	16.3	19.9	30.3
non-convex polygon	4.3	6.0	6.8	7.3	7.7	13.7	19.2	24.2	29.2	34.2
quadrangle	2.8	3.5	3.9	4.2	4.4	4.7	4.8	19.8	59.6	202.6
disk	2.7	3.7	4.5	5.1	5.7	6.2	6.7	7.1	42.2	108.5

Table 4.1: Uniform norm of the least square projection operator on the WAM $\mathcal{A}_n$ for different domains, we use a finer mesh ($\mathcal{T} = \mathcal{A}_{2n}$) to compute the quantity (4.10).

or the relative least square error, computed as

$$\frac{||L_{\mathcal{A}_n} f(\boldsymbol{a}) - f(\boldsymbol{a})||}{||f(\boldsymbol{a})||} = \frac{\left(\sum_{i=1}^{M} \left(L_{\mathcal{A}_n} f(\boldsymbol{a}_i) - f(\boldsymbol{a}_i)\right)^2\right)^{1/2}}{\left(\sum_{i=1}^{M} f(\boldsymbol{a}_i)^2\right)^{1/2}}. \tag{4.13}$$

4.5.2 Discrete Extremal Sets

The first step is build a Vandermonde matrix on the WAM $\mathcal{A}_n$:

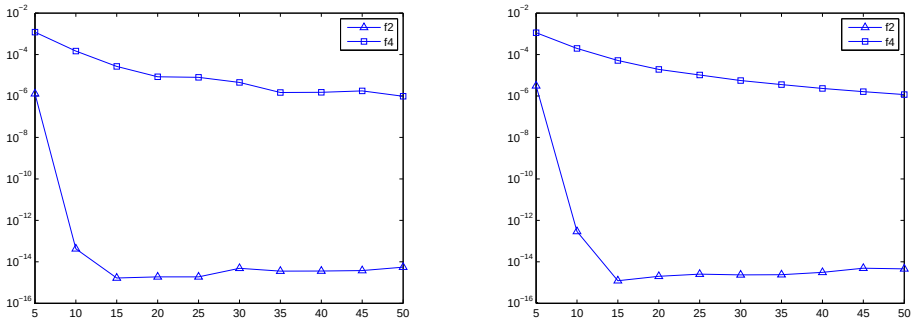
$$V(\boldsymbol{a}; \boldsymbol{p}) = [\boldsymbol{p}_j(\boldsymbol{a}_i)]_{\boldsymbol{ij}} \quad 1 \leq i \leq M, 1 \leq j \leq d. \tag{4.14}$$

We have some choices, with different computational cost, to compute the polynomials basis. We list below the possible ways to produce the Vandermonde matrices.

- Use the total-degree product Chebyshev basis of the minimal rectangle containing the domain; if the matrix V is too ill conditioned, we can still use it provided that we iterate (twice) an orthogonalization procedure, meaning we change the basis;
- Use the basis of orthogonal bivariate polynomials implemented as proposed in Chapter 2, with uniform weights in the nodes of a WAM in the domain. Note that this approach is quite expensive considering the computational cost for the production of the matrix.
- The last option is a hybrid method that takes advantage of the asymptotic distribution of the Discrete Leja Points. Since the discrete measure ... , we extract the DLP using the Chabyshev matrix, and then compute the orthogonal bivariate polynomial using uniform weights and DLP as nodes. Then we evaluate the orthogonal polynomial in all the points of a WAM and extract the Approximate Fekete Points.

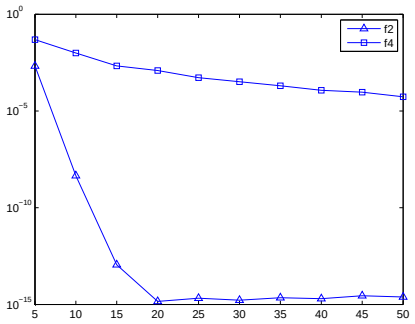
4.5.3 The unit square

The case of the square $[-1, 1]^2$ is particular because the orthogonality is preserved also with the basic algorithm with short recurrences. Thus, not only we can compute the Vandermonde matrix at the WAM and solve the computational problem with interesting computational savings, but also use directly the matrix for the extraction of Fekete and Leja points.

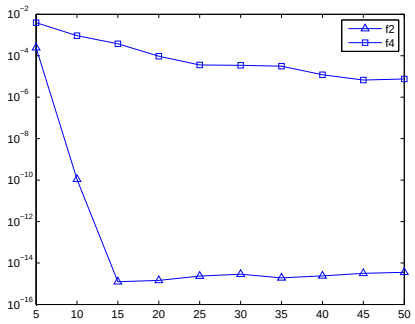


(a) Convex polygon

(b) Non-convex polygon



(c) Quadrangle



(d) Unit Disk

Figure 4.2: Maximum (Absoulte) Error by least-squares approximation of the test functions.

(a) Convex Polygon

	Cheb		OPs		OPs on Leja
DEG	AFP	DLP	AFP	DLP	AFP
5	6.2	8.6	6.2	8.0	8.0
10	13.2	28.5	13.2	18.3	18.3
15	32.5	35.6	32.5	46.9	46.9
20	38.8	79.4	38.8	54.8	54.8
25	52.5	189.9	52.5	76.7	76.7
30	88.4	133.4	102.8	109.9	151.7
35	–	–	87.5	192.6	215.6
40	–	–	152.1	298.1	189.3
45	–	–	170.4	231.6	293.5
50	–	–	260.0	390.8	459.9

(b) Non-convex polygon

	Cheb		OPs		OPs on Leja
DEG	AFP	DLP	AFP	DLP	AFP
5	7.0	17.4	7.0	6.0	6.0
10	13.3	22.3	13.3	18.6	18.6
15	25.8	37.7	25.8	31.2	31.2
20	42.2	65.2	42.2	89.3	89.3
25	77.5	159.8	69.6	130.8	186.1
30	83.9	205.0	81.1	175.0	123.2
35	–	–	86.0	170.0	132.3
40	–	–	190.6	226.3	215.3
45	–	–	227.6	249.7	351.9
50	–	–	193.2	332.8	375.1

Table 4.2

(a) Quadrangle

	Cheb		OPs		OPs on Leja
DEG	AFP	DLP	AFP	DLP	AFP
5	5.8	8.0	5.8	6.1	6.2
10	14.6	24.5	14.6	20.7	20.7
15	33.6	52.7	33.6	47.3	47.3
20	55.0	73.1	55.0	70.4	70.4
25	73.9	116.8	73.9	120.9	164.1
30	96.7	159.4	92.6	226.8	161.6
35	–	–	111.5	214.7	166.8
40	–	–	193.1	233.5	197.5
45	–	–	176.4	359.2	307.1
50	–	–	238.5	349.0	435.7

(b) Disk

	Cheb		OPs		OPs on Leja
DEG	AFP	DLP	AFP	DLP	AFP
5	11.0	11.0	11.0	11.0	11.0
10	13.5	27.6	13.5	28.3	21.0
15	38.4	40.8	38.3	49.8	35.0
20	52.5	89.5	52.5	85.4	63.7
25	75.1	118.8	71.7	102.4	90.3
30	98.7	225.7	98.7	105.2	202.5
35	133.9	442.8	131.8	257.2	239.2
40	159.0	387.2	195.3	266.2	246.2
45	–	–	198.5	396.4	255.6
50	–	–	270.5	396.9	423.1

Table 4.3

Conclusion

Appendix A

Software

Bibliography

[1] M. Barel and A. Chesnokov. "A method to compute recurrence relation coefficients for bivariate orthogonal polynomials by unitary matrix transformations". English. In: *Numerical Algorithms* 55.2-3 (2010), pp. 383–402. URL: http://dx.doi.org/10.1007/s11075-010-9392-y.

[2] B. J. Bauman and H. Xiao. "Gaussian quadrature for optical design with non-circular pupils and fields, and broad wavelengths". In: *Proc. SPIE* 7652.2 (2010), pp. 1–12.

[3] R. Berman and S. Boucksom. "Equidistribution of Fekete points on complex manifolds". In: *ArXiv e-prints* (2008). 0807.0035; Provided by the SAO/NASA Astrophysics Data System. URL: http://adsabs.harvard.edu/abs/2008arXiv0807.0035B.

[4] A. Björck. *Numerical Methods for Least Squares Problems*. SIAM, 3600 Market Street, Floor 6, Philadelphia, PA 19104: Society for Industrial and Applied Mathematics, 1996. ISBN: 9781611971484. URL: http://books.google.it/books?id=6YCC9U9_1UkC.

[5] Å. Björck. "Numerics of Gram-Schmidt Orthogonalization". In: *Linear Algebra Appl.* 197–198 (1994), pp. 297–316.

[6] C. de Boor. "Polynomial Interpolation in Several Variables". In: (1994).

[7] L. Bos and N. Levenberg. "On the Approximate Calculation of Fekete Points: the Univariate Case". In: *Electron. Trans. Numer. Anal* 30.377-397 (2008).

[8] L. Bos, N. Levenberg, and S. Waldron. "On the spacing of Fekete Points for a Sphere, Ball or Simplex". In: *Indag. Math.* 19 (2008), pp. 163–176.

[9] L. Bos and M. Vianello. "Subperiodic trigonometric interpolation and quadrature". In: *Applied Mathematics and Computation* 218.21 (2012), p. 10630. URL: http://www.sciencedirect.com/science/article/pii/S0096300312004237.

[10] L. Bos et al. "Bivariate Lagrange interpolation at the Padua points: the generating curve approach". In: *J. Approx.Theory* 143 (2006), pp. 15–25.

[11] L. Bos et al. "Computing Multivariate Fekete and Leja Points by Numerical Linear Algebra". In: *SIAM Journal on Numerical Analysis* 48.5 (2010). http://epubs.siam.org/doi/pdf/10.1137/090779024, pp. 1984–1999. URL: http://epubs.siam.org/doi/abs/10.1137/090779024.

[12] L. Bos et al. "Geometric Weakly Admissible Meshes, Discrete Least Squares Approximations and Approximate Fekete Points". In: *ArXiv e-prints* (Feb. 2009). arXiv: 0902.0215 [math.NA].

[13] L. Bos et al. “Weakly admissible meshes and discrete extremal sets”. In: *Numerical Mathematics.Theory, Methods and Applications* 4.1 (2011). 65D05 (65D30); 2862657; Daniela Roşca, pp. 1–12. URL: http://dx.doi.org/10.4208/nmtma.2011.m1003.

[14] Len Bos, Alvise Sommariva, and Marco Vianello. “Least-squares polynomial approximation on weakly admissible meshes: Disk and triangle”. In: *Journal of Computational and Applied Mathematics* 235.3 (2010), pp. 660 –668. ISSN: 0377-0427. DOI: http://dx.doi.org/10.1016/j.cam.2010.06.019. URL: http://www.sciencedirect.com/science/article/pii/S0377042710003717.

[15] J.-P. Calvi and N. Levenberg. “Uniform approximation by least squares polynomials”. In: *J. Approx. Theory* 152 (2008), pp. 82–100.

[16] R. Cools. “Constructing Cubature Formulae: The Science Behind the Art”. In: *Acta Numerica* 6 (1997), pp. 1–54.

[17] J. W. Daniel et al. “Reorthogonalization and Stable Algorithms Updating the Gram-Schmidt QR Factorization”. In: *Math. Comp.* 30.no.136 (1976), pp. 772–795.

[18] J. W. Demmel. *Applied numerical linear algebra*. ID: CALCOLO NUMERICO; ID: ALGEBRA LINEARE. Philadelphia: SIAM, 1997, XI, 419 p. ISBN: 0898713897.

[19] I. Dumitriu, A. Edelman, and G. Shuman. “MOPS: Multivariate orthogonal polynomials (symbolically)”. In: *Journal of Symbolic Computation* 42.6 (2007), p. 587. URL: http://www.sciencedirect.com/science/article/pii/S0747717107000090.

[20] C. Dunkl. “Orthogonal Polynomials on the Hexagon”. In: *SIAM Journal on Applied Mathematics* 47.2 (1987). http://epubs.siam.org/doi/pdf/10.1137/0147022, pp. 343–351. URL: http://epubs.siam.org/doi/abs/10.1137/0147022.

[21] Charles F. Dunkl and Yuan Xu. Cambridge University Press, 2001. URL: http://dx.doi.org/10.1017/CBO9780511565717.

[22] P. Erdös and P. Turán. “On interpolation. I. Quadrature and mean convergence in the Lagrange interpolation”. In: *Ann. of Math.* 38 (1937), pp. 142–155.

[23] G. Da Fies, A. Sommariva, and M. Vianello. “Algebraic cubature by linear blending of elliptical arcs”. In: *Applied Numerical Mathematics* 74.0 (2013), p. 49. URL: http://www.sciencedirect.com/science/article/pii/S0168927413000974.

[24] G. Da Fies and M. Vianello. “Algebraic cubature on planar lenses and bubbles”. In: *Dolomites Res. Notes Approx* 5 (2012), pp. 7–12.

[25] G. Da Fies and M. Vianello. *Product Gaussian quadrature on circular lunes*. 2013.

[26] G. Da Fies and M. Vianello. “Trigonometric Gaussian quadrature on subintervals of the period”. In: *Electron. Trans. Numer. Anal.* (2012).

[27] G. J. Gassner et al. “Polymorphic nodal elements and their application in discontinuous Galerkin methods”. In: *Journal of Computational Physics* 228.5 (2009), pp. 1573–1590.

[28] W. Gautschi. “Orthogonal polynomials: applications and computation”. In: *Acta Numerica* 5 (Jan. 1996), pp. 45–119. URL: http://journals.cambridge.org/article_S0962492900002622.

[29] W. Gautschi. *Orthogonal Polynomials: Computation and Approximation*. 2004556094. Oxford University Press, 2004. ISBN: 9780198506720. URL: http://books.google.it/books?id=rS35pXlqY9MC.

[30] W. Gautschi. "Orthogonal polynomials (in Matlab)". In: *Journal of Computational and Applied Mathematics* 178.1–2 (2005). Proceedings of the Seventh International Symposium on Orthogonal Polynomials,Special Functions and Applications, pp. 215–234. URL: http://www.sciencedirect.com/science/article/pii/S0377042704003760.

[31] W. Gautschi. "Sub-range Jacobi polynomials". English. In: *Numerical Algorithms* 61.4 (2012), pp. 649–657. URL: http://dx.doi.org/10.1007/s11075-012-9556-z.

[32] M. Gentile, A. Sommariva, and M. Vianello. "Polynomial interpolation and cubature over polygons". In: *Journal of Computational and Applied Mathematics* 235.17 (2011), p. 5232. URL: http://www.sciencedirect.com/science/article/pii/S0377042711002603.

[33] L. Giraud, J. Langou, and M. Rozložník. *On the loss of orthogonality in the Gram-Schmidt orthogonalization process.* Technical Report TR/PA/03/25. Preliminary version of the paper published in Computer and Mathematics with Applications, vol. 50, pp 1069-1075, 2005. 2003.

[34] L. Giraud et al. "Rounding error analysis of the classical Gram-Schmidt orthogonalization process". English. In: *Numerische Mathematik* 101.1 (2005), pp. 87–100. URL: http://dx.doi.org/10.1007/s00211-005-0615-4.

[35] Gene H. Golub and Charles F. Van Loan. *Matrix Computations (3rd Ed.)* Baltimore, MD, USA: Johns Hopkins University Press, 1996. ISBN: 0-8018-5414-8.

[36] CAA Padova-Verona Research group. *Numerical codes for multivariate interpolation, quadrature and hyperinterpolation, online at:* URL: http://www.math.unipd.it/~marcov/CAAsoft.html.

[37] Olaf Hansen, Kendall Atkinson, and David Chien. "On the norm of the hyperinterpolation operator on the unit disc and its use for the solution of the nonlinear Poisson equation". In: *IMA Journal of Numerical Analysis* (2009).

[38] M. Huhtanen. "Orthogonal polyanalytic polynomials and normal matrices". In: *Mathematics of Computation* 72.241 (2003). MCMPAF; 42C05 (15A57 65F15); 1933825 (2003f:42037); A. Bultheel, 355–373 (eetron). URL: http://dx.doi.org/10.1090/S0025-5718-02-01417-5.

[39] M. Huhtanen and R. M. Larsen. "On Generating Discrete Orthogonal Bivariate Polynomials". English. In: *BIT Numerical Mathematics* 42.2 (2002). J2: BIT Numerical Mathematics, pp. 393–407. URL: http://dx.doi.org/10.1023/A%3A1021907210628.

[40] Ali Çivril and Malik Magdon-Ismail. "On selecting a maximum volume submatrix of a matrix and related problems". In: *Theoretical Computer Science* 410.47–49 (2009), pp. 4801 –4811. ISSN: 0304-3975. DOI: http://dx.doi.org/10.1016/j.tcs.2009.06.018. URL: http://www.sciencedirect.com/science/article/pii/S0304397509004101.

[41] M. I. Klimek. *Pluripotential Theory.* lc91025014. Clarendon Press, 1991. ISBN: 9780198535683. URL: http://books.google.it/books?id=kwuoAAAAIAAJ.

[42] M. Kowalski. "The Recursion Formulas for Orthogonal Polynomials in n Variables". In: *SIAM Journal on Mathematical Analysis* 13.2 (1982), pp. 309–315. URL: http://epubs.siam.org/doi/abs/10.1137/0513022.

[43] Bos L. and Vianello M. "Low cardinality admissible meshes on quadrangles, triangles and disks". In: *Math. Inequal. Appl.* (2012).

[44] Cornelius Lanczos. "Solution of systems of linear equations by minimized iterations". In: *J.Res.Natl.Bur.Stand* 49 (1952), pp. 33–53.

[45] H. Li, J. Sun, and Y. Xu. "Discrete Fourier Analysis, Cubature, and Interpolation on a Hexagon and a Triangle". In: *SIAM Journal on Numerical Analysis* 46.4 (2008), pp. 1653–1681. DOI: 10.1137/060671851. eprint: http://epubs.siam.org/doi/pdf/10.1137/060671851. URL: http://epubs.siam.org/doi/abs/10.1137/060671851.

[46] S. De Marchi, M. Vianello, and Y. Xu. "New cubature formulae and hyperinterpolation in three variables". English. In: *BIT Numerical Mathematics* 49.1 (2009), pp. 55–73. URL: http://dx.doi.org/10.1007/s10543-009-0210-7.

[47] MathWorks. *MATLAB*. 2013. URL: http://www.mathworks.it/products/matlab/.

[48] I. P. Mysovskikh. "On the construction of cubature formulas with fewest nodes". In: *Soviet Math.Dokl.* 9 (1968), pp. 277–280.

[49] I. P. Mysovskikh. "The approximation of multiple integrals by using interpolatory cubature formulae". In: *Soviet Math.Dokl.* (1980), pp. 217–243.

[50] B. N. Parlett. *The Symmetric Eigenvalue Problem*. 79027221. Society for Industrial and Applied Mathematics, 1998. ISBN: 9780898714029. URL: http://books.google.it/books?id=uaYXlkHVPpEC.

[51] W. Pleśniak. "Multivariate Jackson Inequality". In: *Journal of Computational and Applied Mathematics* 233.3 (2009). 9th {OPSFA} Conference, p. 815. URL: http://www.sciencedirect.com/science/article/pii/S0377042709001307.

[52] L. Reichel and W. B. Gragg. "Algorithm 686: FORTRAN Subroutines for Updating the QR Decomposition". In: *ACM Trans.Math.Softw.* 16.4 (1990), pp. 369–377. URL: http://doi.acm.org/10.1145/98267.98291.

[53] Robert Schaback and Stefano De Marchi. "Nonstandard Kernels and Their Applications". In: *Dolomites Research Notes on Approximation* (2009).

[54] H. J. Schmid and Yuan Xu. "On bivariate Gaussian cubature formulae". In: *Proc. Amer. Math. Soc.* 122.3 (1994), pp. 833–841. ISSN: 0002-9939. DOI: 10.2307/2160762. URL: http://dx.doi.org/10.2307/2160762.

[55] I. H. Sloan. "Polynomial Interpolation and Hyperinterpolation over General Regions". In: *Journal of Approximation Theory* 83.2 (Nov. 1995), pp. 238–254.

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

[57] A. Sommariva and M. Vianello. "Approximate Fekete points for weighted polynomial interpolation". In: *Electronic Transactions on Numerical Analysis* (2009).

[58] A. Sommariva and M. Vianello. "Computing approximate Fekete points by QR factorizations of Vandermonde matrices". In: *Computers & Mathematics with Applications* 57.8 (2009), p. 1324. URL: http://www.sciencedirect.com/science/article/pii/S0898122109000625.

[59] A. Sommariva and M. Vianello. "Gauss-Green cubature and moment computation over arbitrary geometries". In: *Journal of Computational and Applied Mathematics* 231.2 (2009), p. 886. URL: http://www.sciencedirect.com/science/article/pii/S0377042709003173.

[60] A. Sommariva and M. Vianello. "Product Gauss cubature over polygons based on Green's integration formula". English. In: *BIT Numerical Mathematics* 47.2 (2007), pp. 441–453. URL: http://dx.doi.org/10.1007/s10543-007-0131-2.

[61] P. K. Suetin. *Orthogonal Polynomials in Two Variables*. 2002421778. Taylor & Francis, 1999. ISBN: 9789056991678. URL: http://books.google.it/books?id=oGi7hvzIcx4C.

[62] Mark A. Taylor, Beth A. Wingate, and Len P. Bos. "A Cardinal Function Algorithm for Computing Multivariate Quadrature Points". In: *SIAM J.Numer.Anal.* 45.1 (2007), pp. 193–205. URL: http://dx.doi.org/10.1137/050625801.

[63] Lloyd N. Trefethen and David Bau III. *Numerical linear algebra*. 50. Siam, 1997.

[64] T. Warburton. "An explicit construction of interpolation nodes on the simplex". English. In: *Journal of Engineering Mathematics* 56.3 (2006), pp. 247–262. URL: http://dx.doi.org/10.1007/s10665-006-9086-6.

[65] M. Weisfeld. "Orthogonal polynomials in several variables". English. In: *Numerische Mathematik* 1.1 (1959). J2: Numer. Math., pp. 38–40. URL: http://dx.doi.org/10.1007/BF01386371.

[66] Don R. Wilhelmsen. "A Markov inequality in several dimensions". In: *Journal of Approximation Theory* 11.3 (1974), pp. 216–220.

[67] Wolfram. *Mathematica*. 2013. URL: http://www.wolfram.com/mathematica/.

[68] Hong Xiao and Zydrunas Gimbutas. "A Numerical Algorithm for the Construction of Efficient Quadrature Rules in Two and Higher Dimensions". In: *Comput.Math.Appl.* 59.2 (2010), pp. 663–676. URL: http://dx.doi.org/10.1016/j.camwa.2009.10.027.

[69] Y. Xu. "On Multivariate Orthogonal Polynomials". In: *SIAM Journal on Mathematical Analysis* 24.3 (1993). doi: 10.1137/0524048; M3: doi: 10.1137/0524048; 09, pp. 783–794. URL: http://dx.doi.org/10.1137/0524048.

[70] Yuan Xu. "Minimal cubature rules and polynomial interpolation in two variables". In: *Journal of Approximation Theory* 164.1 (2012), pp. 6 –30. ISSN: 0021-9045. DOI: http://dx.doi.org/10.1016/j.jat.2011.09.003. URL: http://www.sciencedirect.com/science/article/pii/S0021904511001559.