ABSTRACT

Approximation and Interpolation with Bernstein Polynomials

Larry Allen, Ph.D.

Mentor: Robert C. Kirby, Ph.D.

Bernstein polynomials, long a staple of approximation theory and computational geometry, have also increasingly become of interest in finite element methods. In this dissertation, we investigate fundamental problems in approximation theory and numerical analysis involving Bernstein polynomials. We begin by developing a structured decomposition of the inverse of the matrices related to approximation and interpolation. These matrices are highly ill-conditioned, and so we introduce a non-standard matrix norm to study the conditioning of the matrices, showing that the conditioning in this case is better than in the standard 2-norm. We conclude by giving an algorithm for enforcing bounds constraints on the approximating polynomial. Extensions of the interpolation problem and constrained approximation problem to higher dimensions are also considered.

Approximation and Interpolation with Bernstein Polynomials

by

Larry Allen, B.S., M.S.

A Dissertation

Approved by the Department of Mathematics

_____

Dorina Mitrea, Ph.D., Chairperson

Submitted to the Graduate Faculty of
Baylor University in Partial Fulfillment of the
Requirements for the Degree
of

Doctor of Philosophy

Approved by the Dissertation Committee

_____

Robert C. Kirby, Ph.D., Chairperson

_____

Greg Hamerly, Ph.D.

_____

Lance Littlejohn, Ph.D.

_____

Andrei Martinez-Finkelshtein, Ph.D.

_____

Brian Simanek, Ph.D.

Accepted by the Graduate School

December 2021

_____

J. Larry Lyon, Ph.D., Dean

Page bearing signatures is kept on file in the Graduate School.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGMENTS

I'd first and foremost like to thank Dr. Alexei Kolesnikov for introducing me to mathematics research and getting me started on this incredible journey.

I would also like to thank my advisor, Dr. Robert Kirby. His passion and enthusiasm for our projects kept me motivated throughout all these years. I've enjoyed learning so much from him, and I appreciate how much he has helped me grow as a mathematician, especially with respect to numerical computing.

Lastly, I want to thank my friends and family. I am extremely grateful for their support and the many conversations we've had over the years; without them, I never would have made it this far.

CHAPTER ONE

Introduction

Bernstein polynomials were first introduced more than a century ago to give a constructive proof of the Weierstrass approximation theorem [8]. The Weierstrass approximation theorem (see, for example, [15, 54, 55, 60]) states that continuous functions on closed intervals can be uniformly approximated by polynomials; more precisely, if f is a continuous function on $[a, b]$, then for every $\varepsilon > 0$, there exists a polynomial p such that

$$|f(x) - p(x)| < \varepsilon \quad \text{for all} \quad x \in [a, b]. \tag{1.1}$$

By translating and scaling, it suffices to show that the result is true on $[0, 1]$. To accomplish this, Bernstein introduced the following polynomials for integers $n \geq 0$ and continuous functions f on $[0, 1]$:

$$B^n(f)(x) = \sum_{i=0}^{n} f\left(\frac{i}{n}\right) \binom{n}{i} x^i (1-x)^{n-i}. \tag{1.2}$$

It can be shown that the sequence $\{B^n(f)\}_{n=0}^{\infty}$ converges uniformly to f on $[0, 1]$, and hence the desired polynomial p can be found by choosing n sufficiently large.

More recently, Bernstein polynomials have been used as a tool in Computer Aided Geometric Design (CAGD) via Bézier curves. The field of CAGD deals with the mathematical description of shape for use in computer graphics and approximation theory. Bézier curves are parametric curves named after Pierre Bézier, who introduced

them in the early 1960s to give a computer representation of mechanical parts [24]. The connection to Bernstein polynomials was made by A.R. Forrest [27], who gave the representation

$$\mathbf{p}(t) = \sum_{i=0}^{n} \mathbf{p}_i \binom{n}{i} t^i (1-t)^{n-i}. \tag{1.3}$$

The points $\mathbf{p}_i$ are referred to as the control points and give information about the shape of the curve $\mathbf{p}$. While the curve starts at $\mathbf{p}_0$ and ends at $\mathbf{p}_n$, the rest of the control points are not generally contained in the curve $\mathbf{p}$; however, the curve $\mathbf{p}$ is completely contained in the convex hull of its control points, and so the control points can be used to manipulate the curve in a natural way. Combining (1.3) with de Casteljau's algorithm [10] for fast and stable evaluation of Bernstein polynomials has made Bézier curves a powerful resource in animation and other related fields.

Bernstein polynomials have also become a point of interest in the finite element method for numerically solving partial differential equations (PDEs). When discretizing a PDE, it is desirable to enforce certain displacement or continuity conditions. The Bernstein form facilitates the enforcement of these conditions and closed-form computation of the mass and stiffness matrices, which are used in the implementation of the finite element method [56]. The efficacy of Bernstein polynomials has led to an investigation of fast algorithms for high-order polynomials on simplicial domains, as seen in papers by Kirby, Ainsworth et al., and Kirby and Trinh [1, 37, 39].

The goal of this dissertation is to investigate fundamental problems in numerical analysis and approximation theory related to Bernstein polynomials. One of these problems is to find the best approximation (with respect to a given norm) of a function

by polynomials of a fixed degree. A classical version of this problem is to consider functions belonging to a Hilbert space and to find the best approximation in the norm induced by the corresponding inner product [21]; for example, given a continuous function f over the interval [0,1], find a polynomial p of degree at most m that minimizes $\|f - p\|_{L^2}$. In this setting, there exists a unique solution [16], and the coefficients of the solution with respect to a chosen basis can be characterized by a linear system involving the mass (or Gram) matrix [15]. Harder than a Hilbert space setting, one can also consider the best approximation in the norm of uniform convergence. While the norm is not induced by an inner product, there still exist unique solutions which are characterized by the Chebyshev Alternation Theorem [14]; however, the theorem does not provide the solution explicitly [17], and so solutions are often found for certain types of functions [18, 17, 34, 45].

Recently, attention has focused on the much harder version of these problems: approximation subject to bounds constraints; that is, given a continuous function f on [0,1], find a polynomial q of degree at most m with $q(x) \geq 0$ for all $x \in [0, 1]$ such that $\|f - q\|_{L^2}$ is minimized [12, 13, 20]. The set of nonnegative polynomials of degree at most m on $[0, 1]$ is a closed, convex subset of the space of polynomials of degree at most m (see Proposition 5.1), and so there exists a unique solution to this problem [16]; however, even representing bounds-constrained polynomials presents challenges. In [20], Després uses the Lukacs Theorem [58] to give a representation of all such polynomials. This representation is used in [13] to give a nonlinear projection algorithm for approximating bounded functions on an interval in the uniform norm by polynomials that satisfy the same bounds on the interval. Earlier, Nesterov [48]

classified nonnegative polynomials over an interval in terms of certain convex cones of their coefficients in the monomial basis. The coefficients of a polynomial lie in this cone if and only if the polynomial has a certain sum-of-squares type representation that, in the univariate case, is equivalent to nonnegativity. Nie and Demmel [49], for example, use this characterization to solve certain shape optimization problems via semidefinite programming. The cone constraint is equivalent to nonnegativity in the univariate case, but it encodes a type of sum-of-squares property that is only a sufficient condition in the multivariate settings.

Alternatively, one can consider characterizing bounds-constrained polynomials via Bernstein polynomials. Bernstein polynomials were first introduced more than a century ago to give a constructive proof of the Weierstrass approximation theorem [8]. The Bernstein basis forms a nonnegative partition of unity with a geometric decomposition, and the convex hull of a polynomial's coefficients in the Bernstein basis contains that polynomial. The converse of this last statement is not true, but Bernstein's Theorem [9] states that a polynomial satisfies some bounds on an interval if and only if there exists a degree (greater than or equal to the polynomial degree) in which the Bernstein coefficients satisfy the same bounds. While a general method for computing the exact number of the higher degree is unknown, an upper bound in terms of the minimum of the polynomial and the maximum absolute value of the coefficients in the monomial basis is given in [52]. A thorough discussion of certificates of positivity for polynomials in the Bernstein basis (on the simplex as well as the interval) is given in [43]. In addition, properties of Bernstein polynomials give a special structure of finite element matrices [36, 39] and a structured decomposition

of the inverse of matrices related to approximation and interpolation. We make use of many of these properties in our present work.

This leads us to consider the problem of approximating a function with a polynomial whose Bernstein coefficients (perhaps in a fixed higher degree) satisfy the function's bounds; that is, given a continuous function f on [0,1], find a polynomial q of degree at most m such that the degree $n \geq m$ Bernstein coefficients of q are nonnegative and the quantity $\|f - q\|_{L^2}$ is minimized. Similar to the nonnegative approximation, there exists a unique solution to this problem (see Proposition 5.2). While this approximation cannot be better than the best nonnegative approximation, it allows us to frame the approximation problem as a constrained optimization problem with linear inequality constraints. Although we do not include all nonnegative polynomials as in Nesterov's classification [48], our approach extends seamlessly to a multivariate setting.

By working in the L^2 norm rather than the uniform norm, we can make use of certain classical techniques. We introduce the quadratic cost functional

$$d_f(q) = \int_0^1 (f - q)^2 dx \tag{1.4}$$

and note that finding the coefficients of the best unconstrained polynomial approximation follows from solving a linear system with a Gram matrix. We will enforce bounds constraints on the polynomials as linear functions of the Bernstein coefficients, obtaining a quadratic program with linear inequality constraints. These constraints can be found explicitly and are sufficient conditions for the resulting polynomial to satisfy

the desired polynomial bounds. Exact nonnegativity can be obtained by enforcing quadratic cone constraints.

One particular application of bounds-constrained approximation comes in the numerical solution of partial differential equations (PDEs), especially hyperbolic equations [59]. Some approaches to limiting in discontinuous Galerkin (DG) methods for hyperbolic PDEs explicitly utilize the geometric properties of Bernstein polynomials to enforce maximum principles and other invariant properties [32, 40]. These methods are monolithic, with a built-in limiting process. Here, we pose a problem that is separate from any particular PDE method. While our method could be used as a limiter with existing DG methods, the problem is interesting and challenging in its own right. Although our method is focused on preserving a one-sided bound (non-negativity), its extension to two-sided bounds is straight-forward.

An important aspect of our approach compared to the Campos-Pinto, Charles, and Després (CPCD) algorithm [12] is the straightforward extension of Bernstein polynomials to the simplex. While the question is natural enough to pose (including, for example, limiters for DG methods), the Lukacs theorem is a univariate result, and generalizations, if possible, are likely to be highly nontrivial [53]. The Bernstein basis on a d-simplex consists of (suitably scaled) products of barycentric coordinates, maintaining the convex hull property [41]. This makes it possible to cleanly extend our approach to the simplicial multivariate case.

In this paper, we pose the problem of finding the best L^2 approximation of a function subject to positivity in terms of constrained quadratic programs for the approximant's Bernstein coefficients. In the univariate case, we can pose a quadratically

constrained problem over all positive polynomials of a fixed degree. Additionally, we can approximate this constraint in either the univariate or multivariate case by considering linear inequality constraints on the coefficients. Methods of semidefinite programming [61] may be used to efficiently solve these problems, although in the latter case we also give an (exponential) algorithm based on the KKT conditions, which finds the exact solution and also shows how special Bernstein structure can be incorporated in the process.

Another fundamental problem in numerical analysis and approximation theory is the interpolation problem. Given data $\{f_j\}_{j=0}^n$ and distinct nodes $\{x_j\}_{j=0}^n$, the interpolation problem consists of finding a polynomial p of degree n that satisfies

$$p(x_j) = f_j \tag{1.5}$$

for each $0 \leq j \leq n$. If a basis for the space of polynomials of degree at most n is chosen, then the interpolation problem can be expressed as a system of linear equations, where the coefficient matrix is a Vandermonde-like matrix [30]. Recently, Bernstein polynomials have been considered as a tool for high-order approximation of partial differential equations via the finite element method [1, 23], and the interpolant is often used as a polynomial approximation of initial or boundary data. The corresponding Bernstein–Vandermonde matrix is found to be highly ill-conditioned [15] in the 2 norm, but the structure of the matrix has led to fast algorithms that avoid some of the issues that arise from the ill-conditioning. For example, Marco and Martínez [46] used the fact that the Bernstein–Vandermonde matrix is strictly totally positive [28] to obtain a bidiagonal factorization of the inverse, and Ainsworth and Sanchez [3]

adapted the standard divided difference algorithm for the monomial Vandermonde matrix [11] to the Bernstein basis. In addition, it was observed in [19] that the collocation matrices of the Bernstein basis are optimally conditioned among all other totally positive bases when considering the conditioning in the infinity norm.

In this dissertation, we generalize an argument made by Kaplan [35] to obtain a decomposition of the inverse of Vandermonde-like matrices in terms of their transpose, a diagonal matrix, and the corresponding Bézout matrix. When applied to the Bernstein basis, this gives a decomposition of the inverse of the Bernstein–Vandermonde matrix in terms of Hankel, Toeplitz, and diagonal matrices, which in turn leads to a fast algorithm for solving the interpolation problem. Additionally, we use a nonstandard matrix norm to give an explanation for the relatively good performance of the Bernstein–Vandermonde matrix despite its massive condition number. While the application of the arguments from [35] to the Bernstein basis leads to an algorithm that is slightly less stable than the ones observed in [3, 46], the derivation is generic and can be applied to other bases and used to investigate the corresponding interpolation problems.

Bernstein polynomials also extend naturally to give a basis for multivariate polynomials of total degree n. Properties of Bernstein polynomials lead to special recursive blockwise-structure for finite element matrices [36, 39]. In this paper, we use this structure to obtain a block LU decomposition of the Bernstein–Vandermonde matrix associated to equispaced nodes on the d-simplex. Equispaced nodes on the d-simplex are commonly used in finite element methods, and so this decomposition (combined with fast, stable algorithms for the one-dimensional Bernstein–Vandermonde matrix)

leads to recursive, block-structured fast algorithms for the related interpolation problems.

The dissertation is structured as follows. In Chapter Two, we introduce the notation and discuss the mathematical preliminaries required by the remainder of the dissertation. Chapter Three focuses on approximation, and Chapter Four focuses on interpolation. In these chapters, we give numerical results for our methods and discuss the conditioning of the related matrices; in particular, we give an explanation for the relatively good performance of the matrices despite their massive condition numbers. Constrained approximation is discussed in Chapter Five, and general conclusions are given in Chapter Six.

CHAPTER TWO

Notation and Preliminaries

In this chapter, we introduce the notation and mathematical preliminaries needed to address the problems of approximation and interpolation with respect to Bernstein polynomials.

For an integer $n \geq 0$, let $\mathcal{P}^n$ denote the space of polynomials of degree at most n on $[0,1]$, and let $\mathcal{P}^{n,+}$ denote the subset of $\mathcal{P}^n$ given by

$$\mathcal{P}^{n,+} = \{p \in \mathcal{P}^n \ : \ p(x) \geq 0 \text{ for all } x \in [0,1]\}. \tag{2.1}$$

For each integer $0 \leq i \leq n$, the i^{th} Bernstein polynomial of degree n is given by

$$B_i^n(x) = \binom{n}{i} x^i (1-x)^{n-i}. \tag{2.2}$$

The Bernstein polynomials form a basis for $\mathcal{P}^n$; that is, every polynomial $p \in \mathcal{P}^n$ can be expressed as

$$p(x) = \sum_{i=0}^{n} \mathbf{\Pi}(p)_i B_i^n(x). \tag{2.3}$$

We use the notation $\mathbf{\Pi}(p)$ to emphasize the connection between a polynomial $p \in \mathcal{P}^n$ and its vector of coefficients $\mathbf{\Pi}(p) \in \mathbb{R}^{n+1}$; in a similar way, every vector $\mathbf{p} \in \mathbb{R}^{n+1}$ generates a polynomial in $\mathcal{P}^n$, which we will denote $\pi(\mathbf{p})$.

If $m \leq n$, then $\mathcal{P}^m \subseteq \mathcal{P}^n$, and so any polynomial expressed in the basis $\{B_i^m(x)\}_{i=0}^m$ can also be expressed in the basis $\{B_i^n(x)\}_{i=0}^n$. We denote by $E^{m,n}$ the $(n+1)\times(m+1)$ matrix that maps the coefficients of the degree m representation to the coefficients of

the degree n representation. It is remarked in [25] that the entries of $E^{m,n}$ are given by

$$E^{m,n}_{ij} = \frac{\binom{m}{j}\binom{n-m}{i-j}}{\binom{n}{i}} \tag{2.4}$$

with the standard convention that $\binom{n}{i} = 0$ whenever $i < 0$ or $i > n$. Note that E is bidiagonal for $n = m + 1$ and adds bands with increasing n.

Define

$$\mathcal{P}^{m,n} = \{p \in \mathcal{P}^m \ : \ E^{m,n}\mathbf{\Pi}(p) \geq \mathbf{0}^n\}, \tag{2.5}$$

where $\mathbf{0}^n$ denotes the vector of zeros of length $n+1$, and the inequality is understood to be component-wise.

We also reference the space of square integrable functions on $[0, 1]$,

$$L^2 = L^2[0,1] = \left\{ f : [0,1] \to \mathbb{R} \ : \ \int_0^1 (f(x))^2 dx < \infty \right\}, \tag{2.6}$$

together with the inner product

$$(f, g) = \int_0^1 f(x)g(x)dx \tag{2.7}$$

and associated norm

$$\|f\|_{L^2} = \sqrt{(f, f)}. \tag{2.8}$$

In addition, if $\mathbf{p} \in \mathbb{R}^{n+1}$, we use $\|\mathbf{p}\|_2$ to denote the standard Euclidean norm of $\mathbf{p}$.

For an integer $n \geq 0$, we let L^n denote the Legendre polynomial (see, for example, [6]) of degree n, mapped from its typical home on $[-1, 1]$ to $[0, 1]$ and scaled so that $L^n(1) = 1$ and

$$\|L^n\|^2_{L^2} = \frac{1}{2n+1}. \tag{2.9}$$

It was shown in [25] that the Legendre polynomials are related to the Bernstein polynomials via

$$L^n(x) = \sum_{i=0}^{n} (-1)^{n+i} \binom{n}{i} B_i^n(x); \tag{2.10}$$

that is,

$$\mathbf{\Pi}(L^n)_i = (-1)^{n+i} \binom{n}{i}. \tag{2.11}$$

Similarly, we let $\mathbf{\Theta}(p)$ refer to the vector of coefficients of a polynomial p with respect to the Legendre basis. we will exploit further connections between these basis later.

Given a function $f \in L^2$, a classical problem in approximation theory is to find a polynomial $p \in \mathcal{P}^n$ that minimizes the quantity $\|f - p\|_{L^2}$. The minimizer p is the orthogonal projection of f onto $\mathcal{P}^n$; that is, p satisfies

$$(f - p, q) = 0 \tag{2.12}$$

for every $q \in \mathcal{P}^n$. In particular, p must satisfy

$$(f - p, B_i^n) = 0 \tag{2.13}$$

for each $0 \leq i \leq n$. Expanding p in the Bernstein basis and using the linearity of the inner product leads to the system

$$\sum_{j=0}^{n} \mathbf{\Pi}(p)_j (B_i^n, B_j^n) = (f, B_i^n) \quad \text{for all} \quad 0 \leq i \leq n. \tag{2.14}$$

We recognize that the left side of (2.14) is a matrix-vector product, and so (2.14) can be compactly expressed as

$$M^n \mathbf{\Pi}(p) = \mathbf{f}, \tag{2.15}$$

where M^n is the $(n+1)\times(n+1)$ matrix given by

$$M^n_{ij} = (B^n_i, B^n_j) = \int_0^1 B^n_i(x)B^n_j(x)dx \tag{2.16}$$

and

$$\mathbf{f}_i = (f, B^n_i) = \int_0^1 f(x)B^n_i(x)dx. \tag{2.17}$$

The matrix M^n is called the Bernstein mass matrix of degree n. Its entries can be exactly computed [37] as

$$M^n_{ij} = \binom{n}{i}\binom{n}{j}\frac{(2n-i-j)!(i+j)!}{(2n+1)!}. \tag{2.18}$$

The mass matrix is symmetric and positive definite. The symmetry follow from (2.16); to see that it is positive definite, we observe that if $0 \neq \mathbf{p} \in \mathbb{R}^{n+1}$, then

$$\mathbf{p}^T M^n \mathbf{p} = \sum_{i,j=0}^{n} M^n_{ij}\mathbf{p}_i\mathbf{p}_j = \sum_{i,j=0}^{n} \mathbf{p}_i\mathbf{p}_j \int_0^1 B^n_i(x)B^n_j(x)dx = \int_0^1 \left(\pi(\mathbf{p})(x)\right)^2 dx > 0. \tag{2.19}$$

Therefore, M^n induces a norm on $\mathbb{R}^{n+1}$, which we denote by

$$\|\mathbf{p}\|_{M^n} = \sqrt{\mathbf{p}^T M^n \mathbf{p}}. \tag{2.20}$$

A consequence of (2.19) is that

$$\|p\|_{L^2} = \|\mathbf{\Pi}(p)\|_{M^n}, \tag{2.21}$$

and a similar argument shows that

$$\int_0^1 p(x)q(x)dx = \mathbf{\Pi}(p)^T M^n \mathbf{\Pi}(q) \tag{2.22}$$

for every $p, q \in \mathcal{P}^n$. By similar reasoning,

$$\|p\|_{L^2} = \|\mathbf{\Theta}(p)\|_{D^n}, \tag{2.23}$$

where $D^n = \operatorname{diag}(1/(2j+1))_{j=0}^n$.

If $m \leq n$, then the mass matrix of degree m is related to the mass matrix of degree n [37] by

$$M^m = \left(E^{m,n}\right)^T M^n \left(E^{m,n}\right). \tag{2.24}$$

In [37], it was also observed that

$$M^n = \Delta^n \widetilde{M}^n \Delta^n, \tag{2.25}$$

where $\Delta^n = \operatorname{diag}\left(\binom{n}{i}\right)_{i=0}^n$ and $\widetilde{M}^n$ is the $(n+1) \times (n+1)$ matrix given by

$$\widetilde{M}^n_{ij} = \frac{(2n-i-j)!(i+j)!}{(2n+1)!}. \tag{2.26}$$

Since $\widetilde{M}^n$ depends only on the sum $i+j$ and not i and j separately, it is a Hankel matrix (constant along antidiagonals).

Since M^n is symmetric and positive definite, its eigenvalues are positive real numbers. The following theorem characterizes the eigenvalues and eigenvectors of M^n as a special case of a result from [36]:

Theorem 2.1. *The eigenvalues of M^n are $\{\lambda_k^n\}_{k=0}^n$, where*

$$\lambda_k^n = \frac{(n!)^2}{(n+k+1)!(n-k)!}. \tag{2.27}$$

The eigenvector corresponding to λ_k^n is $E^{k,n}\mathbf{\Pi}(L^k)$.

Another classical problem in approximation theory is interpolation: given a vector of data $\mathbf{y} \in \mathbb{R}^{n+1}$ and a vector of nodes $\mathbf{x} \in \mathbb{R}^{n+1}$ satisfying $\mathbf{x}_i < \mathbf{x}_{i+1}$ for each $0 \leq i < n$, find $p \in \mathcal{P}^n$ that satisfies

$$p(\mathbf{x}_i) = \mathbf{y}_i \quad \text{for each} \quad 0 \leq i \leq n. \tag{2.28}$$

Since

$$p(\mathbf{x}_i) = \sum_{j=0}^{n} \mathbf{\Pi}(p)_j B_j^n(\mathbf{x}_i), \tag{2.29}$$

we can express (2.28) as the matrix equation

$$V^n(\mathbf{x})\mathbf{\Pi}(p) = \mathbf{y}, \tag{2.30}$$

where $V^n(\mathbf{x})$ is the $(n+1) \times (n+1)$ matrix given by

$$V_{ij}^n(\mathbf{x}) = B_j^n(\mathbf{x}_i). \tag{2.31}$$

The matrix $V^n(\mathbf{x})$ is called the Bernstein–Vandermonde matrix associated to the nodes $\mathbf{x}$.

When studying a matrix equation, we are often concerned with the condition number of the matrix. The condition number is defined to be the maximum ratio of the relative error in the input to the relative error in the output and gives a bound on how inaccurate the solution of the equation will be after approximation. If A is an $(n+1) \times (n+1)$ matrix, we can view A as an operator mapping $\mathbb{R}^{n+1}$ onto itself via $\mathbf{x} \mapsto A\mathbf{x}$ and measure the inputs and outputs in the Euclidean norm. This leads

us to define the matrix norm

$$\|A\|_2 = \max_{\mathbf{x}\neq\mathbf{0}} \frac{\|A\mathbf{x}\|_2}{\|\mathbf{x}\|_2} \tag{2.32}$$

and associated condition number

$$\kappa_2(A) = \|A\|_2\|A^{-1}\|_2. \tag{2.33}$$

However, we can also view A as an operator mapping $\mathcal{P}^n$ onto $\mathbb{R}^{n+1}$ via $p \mapsto A\mathbf{\Pi}(p)$ and measure the inputs in the L^2 norm and the outputs in the Euclidean norm. By (2.21), the inputs can also be measured in the M^n norm. This leads us to define the matrix norm

$$\|A\|_{M^n\to 2} = \max_{\mathbf{x}\neq\mathbf{0}} \frac{\|A\mathbf{x}\|_2}{\|\mathbf{x}\|_{M^n}}, \tag{2.34}$$

and going in the opposite direction,

$$\|A\|_{2\to M^n} = \max_{\mathbf{x}\neq\mathbf{0}} \frac{\|A\mathbf{x}\|_{M^n}}{\|\mathbf{x}\|_2}. \tag{2.35}$$

The corresponding condition number is

$$\kappa_{M^n\to 2}(A) = \|A\|_{M^n\to 2}\|A^{-1}\|_{2\to M^n}. \tag{2.36}$$

Lastly, we introduce Bézout matrices. Originally introduced by James Joseph Sylvester and Arthur Cayley, Bézout matrices have recently been shown to have connections to Hankel and Vandermonde matrices [33, 35] and will be a key component to our structured algorithms. Given polynomials $v, w \in \mathcal{P}^{n+1}$ expressed in a basis $\{b_j^{n+1}(x)\}_{j=0}^{n+1}$ of $\mathcal{P}^n$, the Bézout matrix generated by v and w, denoted $\mathrm{Bez}(v, w)$, is

the $(n+1) \times (n+1)$ matrix whose entries satisfy

$$\frac{v(s)w(t) - v(t)w(s)}{s-t} = \sum_{i,j=0}^{n} \mathrm{Bez}_{ij}(v,w) b_i^n(s) b_j^n(t). \tag{2.37}$$

Since the numerator of the fraction on the left side of (2.37) vanishes when $s = t$, the left side of (2.37) is a polynomial in s and t, and so the entries can be computed by comparing coefficients. The entries of a Bézout matrix depend on the basis chosen. When the monomial basis is used, the Bézout matrix has connections to Hankel matrices; when the Bernstein basis is used, the Bézout matrix has connections to the Bernstein–Vandermonde matrix.

CHAPTER THREE

Bernstein Mass Matrix

This chapter published as: Larry Allen and Robert C. Kirby. "Structured inversion of the Bernstein mass matrix". In: *SIAM Journal of Matrix Analysis and Applications* 41.2 (2020), pp. 413–431

3.1 Characterizing the Inverse Matrix

In this section, we present several approaches to constructing and applying the inverse of the mass matrix. The following result (actually, a generalization) is derived in [44]:

Theorem 3.1.

$$(M^n)^{-1}_{ij} = \frac{(-1)^{i+j}}{\binom{n}{i}\binom{n}{j}} \sum_{k=0}^{n} (2k+1-i+j) \binom{n+1}{i-k}^2 \binom{n+1}{j+k+1}^2. \tag{3.1}$$

In what follows, we summarize a proof of Theorem 3.1 based on Farouki's representation of the dual basis in Section 3.1.1, and then give an alternative derivation of the result based on Bézoutians in Section 3.1.2. This derivation leads to a factorization of the inverse, which is given in Section 3.1.4. A decomposition based on the spectral decomposition is given in Section 3.1.3.

3.1.1 Inversion via the Dual Basis

The dual basis to $\{B^n_i(x)\}^n_{i=0}$ is the set of polynomials $\{d^n_i(x)\}^n_{i=0}$ satisfying

$$\int_0^1 B^n_i(x) d^n_j(x) dx = \begin{cases} 1, & i = j; \\ 0, & \text{otherwise.} \end{cases} \tag{3.2}$$

If we consider $\mathbf{d}^{n,j} = \mathbf{\Pi}(d_j^n)$, then

$$\int_0^1 B_i^n(x) d_j^n(x) dx = \sum_{k=0}^{n} \mathbf{d}_k^{n,j} \int_0^1 B_i^n(x) B_k^n(x) dx = \left(M^n \mathbf{d}^{n,j}\right)_i, \tag{3.3}$$

and hence (3.2) implies that $(M^n)_{ij}^{-1} = \mathbf{d}_i^{n,j}$. Following Farouki's representation of the dual basis coefficients [25], this gives us that

$$(M^n)_{ij}^{-1} = \frac{(-1)^{i+j}}{\binom{n}{i}\binom{n}{j}} \sum_{k=0}^{n} (2k+1) \binom{n+k+1}{n-j} \binom{n-k}{n-j} \binom{n+k+1}{n-i} \binom{n-k}{n-i}. \tag{3.4}$$

Applying a few binomial identities gives the representation in Theorem 3.1.

3.1.2 Inversion via Bézoutians

In this section, we detail an alternate derivation of Theorem 3.1 via Bézout matrices, which we will use in Section 3.1.4 to decompose the inverse in terms of diagonal, Hankel, and Toeplitz matrices. We begin by establishing a closed form for the entries of a monomial Bézout matrix.

Theorem 3.2. *If $v(t) = \sum_{i=0}^{n+1} \mathbf{v}_i t^i$ and $w(t) = \sum_{i=0}^{n+1} \mathbf{w}_i t^i$ are polynomials of degree at most $n+1$, then the entries b_{ij} of the monomial Bézout matrix $\mathrm{Bez}(v, w)$ generated by v and w satisfy*

$$b_{ij} = b_{i-1,j+1} + \mathbf{v}_{j+1}\mathbf{w}_i - \mathbf{v}_i \mathbf{w}_{j+1}. \tag{3.5}$$

Proof. We have that

$$\begin{aligned}(s-t) \sum_{i,j=0}^{n} b_{ij} s^i t^j &= \sum_{i,j=0}^{n} b_{ij} s^{i+1} t^j - \sum_{i,j=0}^{n} s^i t^{j+1} \\ &= \sum_{i,j=0}^{n+1} \left(b_{i-1,j} - b_{i,j-1}\right) s^i t^j.\end{aligned}$$

On the other hand,

$$v(s)w(t) - v(t)w(s) = \sum_{i,j=0}^{n+1} (\mathbf{v}_i \mathbf{w}_j - \mathbf{v}_j \mathbf{w}_i) s^i t^j, \tag{3.6}$$

and so the result follows by comparing coefficients in (2.37). □

We can form the first row and last cloumn of $\mathrm{Bez}(v, w)$ by using (3.5) with $i = 0$ and $j = n$. The rest of the columns can then be built by applying the recurrence relation. However, we can also repeatedly apply (3.5) to obtain a closed form for the entries of the monomial Bézout matrix.

Corollary 3.3. *If* $v(t) = \sum_{i=0}^{n+1} \mathbf{v}_i t^i$ *and* $w(t) = \sum_{i=0}^{n+1} \mathbf{w}_i t^i$ *are polynomials of degree at most* $n+1$*, then the entries* b_{ij} *of the monomial Bézout matrix* $\mathrm{Bez}(v, w)$ *generated by* v *and* w *are given by*

$$b_{ij} = \sum_{k=0}^{m_{ij}} \left(\mathbf{v}_{j+k+1} \mathbf{w}_{i-k} - \mathbf{v}_{i-k} \mathbf{w}_{j+k+1} \right), \tag{3.7}$$

where $m_{ij} = \min\{i, n-j\}$.

Heinig and Rost [33] gave a formula for the inverse of a Hankel matrix in terms of a Bézout matrix.

Theorem 3.4. *If* H *is an* $(n+1) \times (n+1)$ *nonsingular Hankel matrix, and* $\widehat{H}$ *is any* $(n+2) \times (n+2)$ *nonsingular Hankel extension of* H *obtained by appending a row and column to* H*, then*

$$H^{-1} = \frac{1}{\mathbf{w}_{n+1}} \mathrm{Bez}(v, w), \tag{3.8}$$

where v is the polynomial whose coefficients in the monomial basis are given by the last column of H^{-1}, and w is the polynomial whose coefficients in the monomial basis are given by the last column of $\widehat{H}^{-1}$.

Since the matrices H and $\widehat{H}$ are of different sizes, the corresponding polynomials have different degrees. We reconcile this difference by elevating by one degree in the monomial basis (that is, appending a zero to the vector of coefficients).

The next few results are dedicated to finding the v^n and w^{n+1} that correspond to $\widetilde{M}^n$. We begin by showing that the null space of $\left(E^{n-1,n}\right)^T$ is spanned by the eigenvector corresponding to λ_n^n. We then use this to project our candidate $\mathbf{y}^n$ for the last column of $(M^n)^{-1}$ onto the null space and its orthogonal complement. This decomposition gives a recurrence relation for the $\mathbf{y}^n$, which will be the foundation for an inductive proof that $\mathbf{y}^n$ is the last column of $(M^n)^{-1}$. The coefficients of v^n are then obtained via (2.25).

Lemma 3.5. *The null space of $\left(E^{n-1,n}\right)^T$ is spanned by $\mathbf{\Pi}(L^n)$.*

Proof. By (2.4), we have that $\left(E^{n-1,n}\right)^T$ is upper bidiagonal with nonzero entries on the main diagonal and the superdiagonal, and so the null space of $\left(E^{n-1,n}\right)^T$ is one-dimensional. Therefore, it is enough to show that $\mathbf{\Pi}(L^n)$ belongs to the null space of $\left(E^{n-1,n}\right)^T$.

Let $\mathbf{q} = E^{n-1,n}\mathbf{p}$ be in the range of $E^{n-1,n}$. This means that $p = \pi(\mathbf{p})$ is a polynomial of degree at most $n-1$, and hence

$$0 = \int_0^1 p(x)L^n(x)dx. \tag{3.9}$$

Therefore, Theorem 2.1, (2.22), and (2.24) imply that

$$0 = (E^{n-1,n}\mathbf{p})^T M^n \mathbf{\Pi}(L^n) = \lambda_n^n \mathbf{q}^T \mathbf{\Pi}(L^n) = \frac{(n!)^2}{(2n+1)!}\mathbf{q}^T \mathbf{\Pi}(L^n). \tag{3.10}$$

This implies that $\mathbf{\Pi}(L^n)$ is orthogonal in the Euclidean inner product to the range of $E^{n-1,n}$, and so $\mathbf{\Pi}(L^n)$ belongs to the null space of $(E^{n-1,n})^T$. □

Lemma 3.6. *For $n \geq 0$, let $\mathbf{y}^n$ be given by*

$$\mathbf{y}_i^n = (-1)^{n+i}(n+1)\binom{n+1}{i}. \tag{3.11}$$

Then, for $n \geq 1$,

$$\mathbf{y}^n = (2n+1)\mathbf{\Pi}(L^n) + E^{n-1,n}\mathbf{y}^{n-1}. \tag{3.12}$$

Proof. By (2.4) and (2.10), we have that for each $0 \leq i \leq n$,

$$\begin{aligned}
\left[(2n+1)\mathbf{\Pi}(L^n) + E^{n-1,n}\mathbf{y}^{n-1}\right]_i &= (2n+1)\mathbf{\Pi}(L^n)_i + \frac{i}{n}\mathbf{y}_{i-1}^{n-1} + \frac{n-i}{n}\mathbf{y}_i^{n-1} \\
&= (2n+1)(-1)^{n+i}\binom{n}{i} + i(-1)^{n+i}\binom{n}{i-1} \\
&\quad - (n-i)(-1)^{n+i}\binom{n}{i} \\
&= (-1)^{n+i}\left[(n+i+1)\binom{n}{i} + i\binom{n}{i-1}\right] \\
&= (-1)^{n+i}(n+1)\binom{n+1}{i} \\
&= \mathbf{y}_i^n.
\end{aligned}$$

□

Proposition 3.7. *Let $\mathbf{y}^n$ be as in Lemma 3.6. Then*

$$M^n\mathbf{y}^n = \mathbf{e}^n, \tag{3.13}$$

where

$$\mathbf{e}_i^n = \begin{cases} 1, & i = n; \\ 0, & \textit{otherwise.} \end{cases} \tag{3.14}$$

Proof. We show the result by induction on n. Clearly, the result is true for $n = 0$. So suppose $M^{n-1}\mathbf{y}^{n-1} = \mathbf{e}^{n-1}$ for some $n \geq 1$. We show that $M^n\mathbf{y}^n = \mathbf{e}^n$ by showing that $M^n\mathbf{y}^n - \mathbf{e}^n$ belongs to both the null space of $\left(E^{n-1,n}\right)^T$ and its orthogonal complement with respect to the Euclidean inner product.

Multiplying (3.12) on the left by M^n and using Theorem 2.1 gives us that

$$M^n\mathbf{y}^n = (2n+1)\lambda_n^n\mathbf{\Pi}(L^n) + M^nE^{n-1,n}\mathbf{y}^{n-1}. \tag{3.15}$$

Since the null space of $\left(E^{n-1,n}\right)^T$ is spanned by $\mathbf{\Pi}(L^n)$, multiplying the previous equation on the left by $\left(E^{n-1,n}\right)^T$ and using (2.24) and the induction hypothesis gives us that

$$\left(E^{n-1,n}\right)^T M^n\mathbf{y}^n = \mathbf{e}^{n-1}. \tag{3.16}$$

Since $\mathbf{e}^{n-1} = \left(E^{n-1,n}\right)^T \mathbf{e}^n$, the previous equation implies that $M^n\mathbf{y}^n - \mathbf{e}^n$ belongs to the null space of $\left(E^{n-1,n}\right)^T$.

On the other hand, (2.10) and (3.15) imply that

$$\begin{aligned} \mathbf{\Pi}(L^n)^T(M^n\mathbf{y}^n - \mathbf{e}^n) &= \mathbf{\Pi}(L^n)^T((2n+1)\lambda_n^n\mathbf{\Pi}(L^n) + M^nE^{n-1,n}\mathbf{y}^{n-1} - \mathbf{e}^n) \\ &= (2n+1)\lambda_n^n\mathbf{\Pi}(L^n)^T\mathbf{\Pi}(L^n) + \mathbf{\Pi}(L^n)^TM^nE^{n-1,n}\mathbf{y}^{n-1} - 1. \end{aligned}$$

Using (2.10) again, we have that

$$\mathbf{\Pi}(L^n)^T\mathbf{\Pi}(L^n) = \sum_{i=0}^{n}\binom{n}{i}^2 = \binom{2n}{n} = \frac{1}{(2n+1)\lambda_n^n}. \tag{3.17}$$

Therefore,

$$\mathbf{\Pi}(L^n)^T(M^n\mathbf{y}^n - \mathbf{e}^n) = \mathbf{\Pi}(L^n)^T M^n E^{n-1,n}\mathbf{y}^{n-1}. \tag{3.18}$$

Since $\mathbf{\Pi}(L^n)$ is orthogonal in the Euclidean inner product to the range of $E^{n-1,n}$ and M^n is symmetric,

$$\mathbf{\Pi}(L^n)^T M^n E^{n-1,n}\mathbf{y}^{n-1} = \lambda_n^n \mathbf{\Pi}(L^n)^T E^{n-1,n}\mathbf{y}^{n-1} = 0, \tag{3.19}$$

and hence $\mathbf{\Pi}(L^n)^T(M^n\mathbf{y}^n - \mathbf{e}^n) = 0$. Therefore, $M^n\mathbf{y}^n - \mathbf{e}^n$ also belongs to the orthogonal complement of the null space of $(E^{n-1,n})^T$. □

Since $\Delta^n_{nn} = 1$, $M^n\mathbf{y}^n = \mathbf{e}^n$ if and only if $\widetilde{M}^n\Delta^n\mathbf{y}^n = \mathbf{e}^n$. Therefore, the coefficients of v^n are given by

$$\mathbf{v}^n_i = (\Delta^n\mathbf{y}^n)_i = (-1)^{n+i}(n+1)\binom{n}{i}\binom{n+1}{i}. \tag{3.20}$$

We now find the coefficients of w^{n+1}. By Theorem 3.4, the coefficients are given by the last column of the inverse of any nonsingular Hankel extension of $\widetilde{M}^n$. We use this freedom to choose an extension such that the coefficients of w^{n+1} are given by elevating v^n by one degree in the monomial basis and then reversing the order of the entries. To describe this process, we introduce the following notation: given a vector $\mathbf{x}$ of length $n+1$, denote by $\mathbf{x}^P$ the vector of length $n+2$ given by

$$\mathbf{x}^P = P\begin{pmatrix}\mathbf{x}\\ \\ 0\end{pmatrix}, \tag{3.21}$$

where

$$P = \begin{pmatrix} & & & & 1 \\ & & & 1 & \\ & & \iddots & & \\ & 1 & & & \\ 1 & & & & \end{pmatrix} \tag{3.22}$$

is the $(n+2) \times (n+2)$ exchange matrix.

Lemma 3.8. *If H is an $(n+1) \times (n+1)$ nonsingular Hankel matrix of the form*

$$H = \begin{pmatrix} h_0 & h_1 & h_2 & \cdots & h_n \\ h_1 & h_2 & h_3 & \cdots & h_{n-1} \\ h_2 & h_3 & h_4 & \cdots & h_{n-2} \\ \vdots & \vdots & \vdots & \iddots & \vdots \\ h_n & h_{n-1} & h_{n-2} & \cdots & h_0 \end{pmatrix}, \tag{3.23}$$

and $\mathbf{x}$ satisfies

$$H\mathbf{x} = \mathbf{e}^n \tag{3.24}$$

with $\mathbf{x}_0 \neq 0$, then there exists an $(n+2) \times (n+2)$ Hankel extension $\widehat{H}$ of H such that

$$\widehat{H}\mathbf{x}^P = \mathbf{e}^{n+1}. \tag{3.25}$$

Proof. Since $\mathbf{x}^P$ is the action of $\begin{pmatrix} \mathbf{x} & 0 \end{pmatrix}^T$ under the exchange matrix P, we can instead apply the exchange matrix to $\widehat{H}$ and show that there exist α and β such that

$$\begin{pmatrix} h_{n-1} & h_n & \cdots & h_3 & h_2 & h_1 & h_0 \\ h_{n-2} & h_{n-1} & \cdots & h_4 & h_3 & h_2 & h_1 \\ h_{n-3} & h_{n-2} & \cdots & h_5 & h_4 & h_3 & h_2 \\ h_{n-4} & h_{n-3} & \cdots & h_6 & h_5 & h_4 & h_3 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ \alpha & h_0 & \cdots & h_{n-3} & h_{n-2} & h_{n-1} & h_n \\ \beta & \alpha & \cdots & h_{n-4} & h_{n-3} & h_{n-2} & h_{n-1} \end{pmatrix} \begin{pmatrix} \mathbf{x}_0 \\ \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \\ \vdots \\ \mathbf{x}_n \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix}. \tag{3.26}$$

The first n equations are satisfied by assumption. Therefore, choosing

$$\alpha = -\frac{1}{\mathbf{x}_0} \sum_{i=0}^{n-1} h_i \mathbf{x}_{i+1} \tag{3.27}$$

and

$$\beta = \frac{1}{\mathbf{x}_0} \left(1 - \alpha \mathbf{x}_1 - \sum_{i=0}^{n-2} h_i \mathbf{x}_{i+2} \right) \tag{3.28}$$

gives the desired result. □

Therefore, the coefficients of w^{n+1} are given by

$$\mathbf{w}_i^{n+1} = (\Delta^n \mathbf{y}^n)_i^P = (-1)^{i+1}(n+1)\binom{n+1}{i}\binom{n}{i-1}, \tag{3.29}$$

and hence Theorem 3.4 and (3.7) imply that

Theorem 3.9.

$$\begin{aligned}\left(\widetilde{M}^n\right)^{-1}_{ij} &= \frac{1}{\mathbf{w}^{n+1}_{n+1}} \sum_k \left(\mathbf{v}^n_{j+k+1}\mathbf{w}^{n+1}_{i-k} - \mathbf{v}^n_{i-k}\mathbf{w}^{n+1}_{j+k+1}\right) \\ &= (-1)^{i+j} \sum_k (2k+1-i+j) \binom{n+1}{i-k}^2 \binom{n+1}{j+k+1}^2.\end{aligned}$$

Applying the identity $(M^n)^{-1} = (\Delta^n)^{-1}\left(\widetilde{M}^n\right)^{-1}(\Delta^n)^{-1}$ gives the result from Theorem 3.1.

3.1.3 Bernstein–Legendre Conversion

We can use Theorem 2.1 to construct the spectral (or eigenvector) decomposition

$$M^n = Q^n \Lambda^n (Q^n)^T, \tag{3.30}$$

where Λ^n contains the eigenvalues and Q^n is an orthogonal matrix of eigenvectors. Since M^n is symmetric and all of its eigenvalues are distinct, the eigenvectors are orthogonal. Therefore, all that remains is to normalize the eigenvectors in the 2-norm.

Proposition 3.10.

$$\left\| E^{k,n}\mathbf{\Pi}(L^k) \right\|_2^2 = \frac{1}{(2k+1)\lambda_k^n}. \tag{3.31}$$

Proof. The result follows from (2.9), (2.20), and Theorem 2.1:

$$\frac{1}{2k+1} = \|L^k\|^2_{L^2} = \|E^{k,n}\mathbf{\Pi}(L^k)\|^2_{M^n} = \lambda_k^n \left\| E^{k,n}\mathbf{\Pi}(L^k) \right\|_2^2. \tag{3.32}$$

□

Proposition 3.10 implies that

$$Q^n = \left(\sqrt{\lambda_0^n} E^{0,n} \mathbf{\Pi}(L^0) \quad | \quad \cdots \quad | \quad \sqrt{(2n+1)\lambda_n^n} \mathbf{\Pi}(L^n) \right) \tag{3.33}$$

and $\Lambda^n = \operatorname{diag}\left(\lambda_j^n\right)_{j=0}^n$.

Equation (3.33) characterizes Q^n and suggests an algorithm for its construction – begin with $\mathbf{\Pi}(L^k)$ and elevate it to degree n. This requires $\mathcal{O}(n^3)$ operations – there are $\mathcal{O}(n)$ columns and each one needs to be elevated $\mathcal{O}(n)$ times at $\mathcal{O}(n)$ operations per elevation. However, we can also adapt the classical 3-term recurrence for the Legendre polynomials to give a $\mathcal{O}(n^2)$ process for constructing Q^n.

We begin the process by constructing the degree n representation of L^0 – which is simply the vector of ones. We can similarly construct the coefficients for L^1. Then, we proceed inductively. Assuming that L^k has been constructed (for $1 \leq k < n$), the critical step in the recurrence is to multiply $L^k(x)$ by x. Given any polynomial

$$p(x) = \sum_{i=0}^{n} \mathbf{\Pi}(p)_i B_i^n(x)$$

of degree n, we have that

$$xp(x) = \sum_{i=0}^{n} \mathbf{\Pi}(p)_i \binom{n}{i} x^{i+1}(1-x)^{n-i} = \sum_{i=0}^{n+1} \widetilde{\mathbf{p}}_i B_i^{n+1}(x),$$

where $\widetilde{\mathbf{p}}_0 = 0$ and $\widetilde{\mathbf{p}}_i = \frac{i\mathbf{\Pi}(p)_{i-1}}{n+1}$ for $1 \leq i \leq n+1$. This $\mathcal{O}(n)$ process for computes $xp(x)$ in the degree $n+1$ basis, but we need $xL^k(x)$ in the Bernstein basis of degree n. That is, if $\widetilde{\mathbf{p}}$ holds the degree $n+1$ Bernstein coefficients, we need to find $\mathbf{q}$ such that $E^{n,n+1}\mathbf{q} = \widetilde{\mathbf{p}}$, which is a consistent system of $n+2$ equations in $n+1$ unknowns. Several $\mathcal{O}(n)$ algorithms for this are possible, but Gaussian elimination on

the (tridiagonal) normal equations seems stable in practice. Algorithm 1 summarizes building the entire matrix Q^n by using the three-term recurrence and the tridiagonal reduction, where E denotes the matrix $E^{n,n+1}$.

Algorithm 1 Builds the orthogonal matrix Q^n of eigenvectors of M^n

$Q^n[:,0] \leftarrow E^{0,n}\mathbf{\Pi}(L^0)$
$Q^n[:,1] \leftarrow E^{1,n}\mathbf{\Pi}(L^1)$
for $j \leftarrow 2, n-1$ **do**
  $\widetilde{\mathbf{p}} \leftarrow \mathbf{\Pi}(x\pi(Q[:,j-1])(x))$
  $\mathbf{q} \leftarrow \left(E^T E\right)^{-1} E^T \widetilde{\mathbf{p}}$
  $Q^n[:,j] \leftarrow \frac{2j-1}{j}\left(2\mathbf{q} - Q^n[:,j-1]\right) - \frac{j-1}{j} Q^n[:,j-2]$
$Q^n[:,n] \leftarrow \mathbf{\Pi}(L^n(x))$
for $j \leftarrow 0, n$ **do**
  $Q^n[:,j] \leftarrow \sqrt{(2j+1)\lambda_j^n}\, Q^n[:,j]$

Consequently, $(M^n)^{-1} = Q^n\left(\Lambda^n\right)^{-1}\left(Q^n\right)^T$ can be applied to a vector by multiplying by two dense orthogonal matrices and scaling by a diagonal matrix. This process is summarized in Algorithm 1.

3.1.4 Decomposing the Inverse

The statement of Theorem 3.9 suggests the following decomposition of $(M^n)^{-1}$ into diagonal, Toeplitz, and Hankel matrices:

Theorem 3.11. *Let T^n and $\widetilde{T}^n$ be the Toeplitz matrices given by*

$$T_{ij}^n = (-1)^{i-j}\binom{n+1}{i-j}^2 \qquad \text{and} \qquad \widetilde{T}_{ij}^n = (i-j)T_{ij}^n,$$

and let H^n and $\widetilde{H}^n$ be the Hankel matrices given by

$$H_{ij}^n = (-1)^{i+j+1}\binom{n+1}{i+j+1}^2 \qquad \text{and} \qquad \widetilde{H}_{ij}^n = (i+j+1)H_{ij}^n.$$

Then

$$(M^n)^{-1} = (\Delta^n)^{-1} \left[\widetilde{T}^n H^n - T^n \widetilde{H}^n \right] (\Delta^n)^{-1} . \tag{3.34}$$

This result implies a superfast $\mathcal{O}(n \log n)$ algorithm, but our numerical experiments suggest that it is highly unstable.

3.2 Applying the Inverse

Now, we describe several approaches to applying $(M^n)^{-1}$ to a vector.

Cholesky factorization M^n is symmetric and positive definite and therefore admits a Cholesky factorization

$$M^n = C^n (C^n)^T , \tag{3.35}$$

where C^n is lower triangular with positive diagonal entries [57]. Widely available in libraries, computing C^n requires $\mathcal{O}(n^3)$ operations, and each of the subsequent triangular solves require $\mathcal{O}(n^2)$ operations to perform. Our numerical results below suggest that it is one of the more stable and accurate methods under consideration, although our technique based on the eigendecomposition has similar accuracy and $\mathcal{O}(n^2)$ complexity for the startup phase.

Exact inverse In light of Theorem 3.1, we can directly form $(M^n)^{-1}$. Since the formula for each entry requires a sum, forming the inverse requires $\mathcal{O}(n^3)$ operations. The inverse matrix can then be applied to any vector in $\mathcal{O}(n^2)$ operations using the standard algorithm. This has the same startup and per-solve complexity as the Cholesky factorization, although the constants are different.

Spectral decomposition In Section 3.1.3, we showed how to compute the eigendecomposition of M^n and hence express its inverse as

$$(M^n)^{-1} = Q^n (\Lambda^n)^{-1} (Q^n)^T . \tag{3.36}$$

The inverse can be applied by two (dense) matrix multiplications and a diagonal scaling, requiring $\mathcal{O}(n^2)$ operations. Thanks to Algorithm 1, we have only an $\mathcal{O}(n^2)$ startup phase.

DFT-based application Theorem 3.11 implies that we can multiply by the inverse of M^n in $\mathcal{O}(n \log n)$ operations. We invert Δ^n onto a given vector, and then all of the Toeplitz and Hankel matrices can be applied via circulant embedding before inverting Δ^n onto the result. However, our numerical results reveal this approach to be quite unstable in practice, becoming wildly inaccurate long before one could hope to win from the super-fast algorithm. Therefore, we do not go into much detail on this approach.

3.3 Conditioning and Accuracy

Since M^n is symmetric and positive definite, its 2-norm condition number (2.33) is given by the ratio of the largest eigenvalue to the smallest eigenvalue. By Theorem 2.1, this means that M^n is terribly ill-conditioned as n increases. In particular,

$$\kappa_2(M^n) = \frac{\lambda_0^n}{\lambda_n^n} = \frac{(2n+1)!}{(n+1)!n!}. \tag{3.37}$$

While this conditioning seems spectacularly bad, in practice, the Bernstein basis seems to give entirely satisfactory results at moderately high orders of approxima-

tion [38]. Here, we hope to give at least a partial explanation of this phenomenon using the condition number $\kappa_{M^n \to 2}(M^n)$ defined in (2.36).

Proposition 3.12.

$$\|M^n\|_{M^n \to 2} = \sqrt{\lambda_0^n}. \tag{3.38}$$

Proof. Since M^n is symmetric and positive-definite, it has a well-defined positive square root via the spectral decomposition. In particular, for any $\mathbf{x} \in \mathbb{R}^{n+1}$, we have the relationship

$$\|\mathbf{x}\|_{M^n} = \left\| (M^n)^{1/2} \mathbf{x} \right\|_2. \tag{3.39}$$

Therefore, if $\mathbf{x} \neq \mathbf{0}$, then

$$\frac{\|M^n \mathbf{x}\|_2}{\|\mathbf{x}\|_{M^n}} \leq \frac{\left\| (M^n)^{1/2} \right\|_2 \left\| (M^n)^{1/2} \mathbf{x} \right\|_2}{\|\mathbf{x}\|_{M^n}} = \left\| (M^n)^{1/2} \right\|_2 = (\lambda_0^n)^{1/2}, \tag{3.40}$$

and so

$$\|M^n\|_{M^n \to 2} \leq (\lambda_0^n)^{1/2}. \tag{3.41}$$

On the other hand,

$$\begin{aligned} \|M^n\|_{M^n \to 2} = \max_{\mathbf{x} \neq \mathbf{0}} \frac{\|M^n \mathbf{x}\|_2}{\|\mathbf{x}\|_{M^n}} &\geq \frac{\|M^n E^{0,n} \mathbf{\Pi}(L^0)\|_2}{\|E^{0,n} \mathbf{\Pi}(L^0)\|_{M^n}} \\ &= \frac{\lambda_0^n \|E^{0,n} \mathbf{\Pi}(L^0)\|_2}{\left\| (M^n)^{1/2} E^{0,n} \mathbf{\Pi}(L^0) \right\|_2} \\ &= \frac{\lambda_0^n \|E^{0,n} \mathbf{\Pi}(L^0)\|_2}{(\lambda_0^n)^{1/2} \|E^{0,n} \mathbf{\Pi}(L^0)\|_2} \\ &= (\lambda_0^n)^{1/2}. \end{aligned}$$

□

Proposition 3.13.

$$\left\|(M^n)^{-1}\right\|_{2\to M^n} = \frac{1}{\sqrt{\lambda_n^n}}. \tag{3.42}$$

Proof. If $\mathbf{x} \neq \mathbf{0}$, then

$$\frac{\left\|(M^n)^{-1}\,\mathbf{x}\right\|_{M^n}}{\|\mathbf{x}\|_2} = \frac{\left\|(M^n)^{-1/2}\,\mathbf{x}\right\|_2}{\|\mathbf{x}\|_2} \leq \left\|(M^n)^{-1/2}\right\|_2 = (\lambda_n^n)^{-1/2}\,, \tag{3.43}$$

and so

$$\left\|(M^n)^{-1}\right\|_{2\to M^n} \leq (\lambda_n^n)^{-1/2}\,. \tag{3.44}$$

On the other hand,

$$\begin{aligned}
\left\|(M^n)^{-1}\right\|_{2\to M^n} = \max_{\mathbf{x}\neq\mathbf{0}} \frac{\left\|(M^n)^{-1}\,\mathbf{x}\right\|_{M^n}}{\|\mathbf{x}\|_2} &\geq \frac{\left\|(M^n)^{-1}\,\mathbf{\Pi}(L^n)\right\|_{M^n}}{\|\mathbf{\Pi}(L^n)\|_2} \\
&= \frac{\left\|(M^n)^{-1/2}\,\mathbf{\Pi}(L^n)\right\|_2}{\|\mathbf{\Pi}(L^n)\|_2} \\
&= \frac{(\lambda_n^n)^{-1/2}\,\|\mathbf{\Pi}(L^n)\|_2}{\|\mathbf{\Pi}(L^n)\|_2} \\
&= (\lambda_n^n)^{-1/2}\,.
\end{aligned}$$

□

Combining the two previous propositions gives us that

Theorem 3.14.

$$\kappa_{M^n\to 2}(M^n) = \sqrt{\kappa_2(M^n)} = \sqrt{\frac{(2n+1)!}{(n+1)!n!}}. \tag{3.45}$$

A plot of the condition numbers up to degree 20 in both norms is shown in Fig. 3.1. This discussion indicates that, when measuring the relevant L^2 norm rather than the Euclidean norm of the solution process, we can expect much better results than the alarming condition number in (3.37) suggests. It is important to note that nothing in

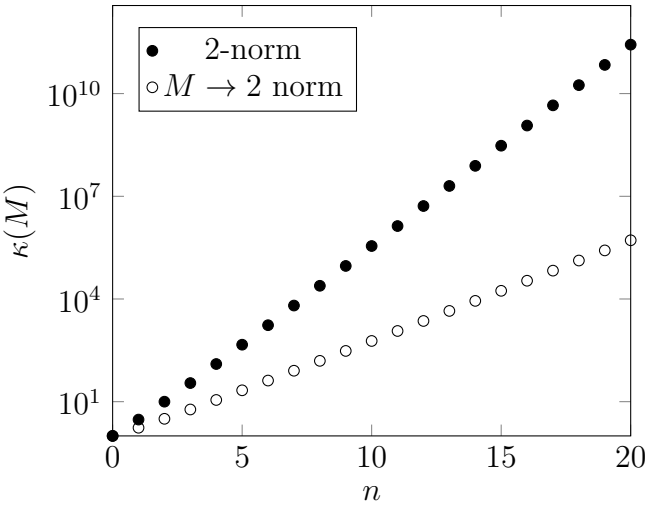
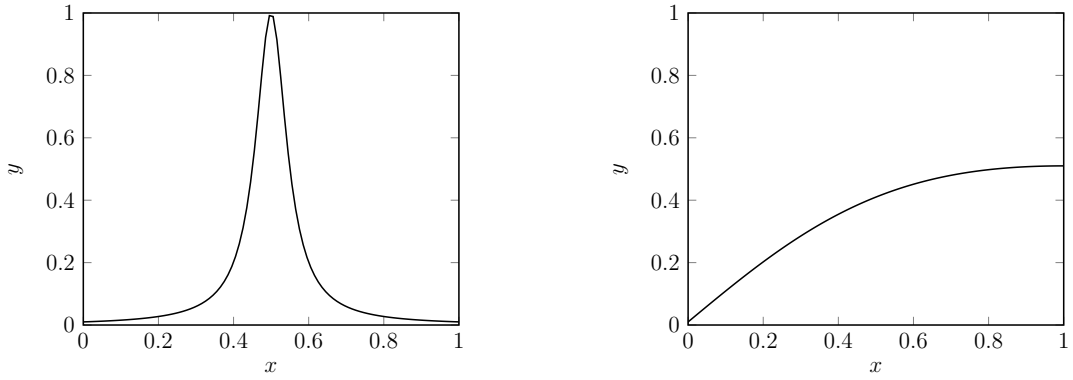


Figure 3.1: Bernstein mass matrix conditioning for degrees 0 through 20 in the 2-norm and the $M \to 2$ norms.

this discussion, other than the eigenvalues of M^n, is particular to the univariate mass matrix. Consequently, this discussion can inform the accuracy of the multivariate mass inversion process in [36] as well as the preconditioners for global mass matrices given in [2].

3.4 Numerical Results

Now, we consider the accuracy of the methods described above on a range of problems. All of our numerical results are run in double precision arithmetic on a 2014 MacBook Air running macOS 10.13 and using Python 2.7.13. Cholesky factorization and FFTs are performed using the `numpy` (v1.12.0) function calls. Also, because our fast algorithm turns out to be rather unstable and our code is a mix of pure Python and low-level compiled libraries, timings are not terribly informative. Consequently, we focus on assessing the stability and accuracy of our methods. If future work



(a) $1/(1+396(x-0.5)^2)$ (b) $0.01 + x/(1+x^2)$

Figure 3.2: Plots of two functions being approximating with Bernstein polynomials. Figure 3.2a is smooth, but has large derivatives and so needs high polynomial order to make the error small. The function in Fig. 3.2b is much simpler to approximate, but illustrates that the methods work on functions not symmetric about the interval midpoint.

leads to more stable fast algorithms, greater care will be afforded to tuning our implementations for performance.

We consider the best L^2 approximation of two smooth functions, $f(x) = 1/(1 + 396(x-0.5)^2)$ and $f(x) = 0.01 + x/(x^2+1)$ – see Fig. 3.2 for plots of these. Both functions are smooth; however, the first function has large derivatives and so requires high order of approximation to obtain small error. The second is not symmetric about $x = 0.5$ but is otherwise relatively easy to approximate with a polynomial.

Solving (2.15) accurately, the best polynomial approximation of degree n gives an L^2 error decreasing exponentially as a function of n. To demonstrate this, Figs. 3.3a and 3.4a show the L^2 error between the two functions and the L^2 projection computed with each of the four methods described above. However, at various degrees, each of the methods deviate from yielding exponential convergence as roundoff error

accumulates. We note that the DFT-based method seems to be the worst, followed by multiplication by the exactly computed inverse.

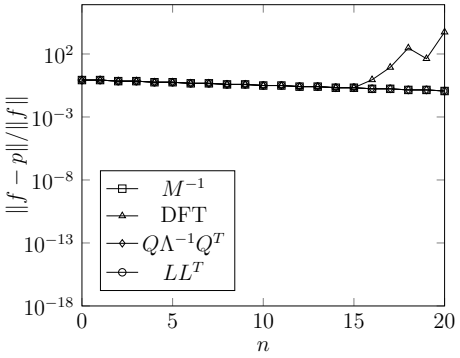
We also compute the best approximation using the Legendre polynomials (which only involves numerical integration) and compute the L^2 difference between that and the polynomial produced by each solution technique. Equivalently, this is the relative M^n-norm error between the exact and computed solution to (2.15). These plots appear in Figs. 3.3b and 3.4b, futher demonstrating the instability of the DFT-based approach.

Differences between the three more stable methods begin to appear when we consider the Euclidean norm of the error (Figs. 3.3c and 3.4c) and residual (Figs. 3.3d and 3.4d) for each of our solution methods. The DFT-based approach again performs much worse than the other methods, although the exact inverse gives 2-norm error comparable to the spectral and Cholesky approaches. The residual plots show that the latter two methods give very small residual errors, not much above machine precision. This strong backward stability of the Cholesky decomposition is a known property [31], and we suspect (but have not proven) that it holds for our spectral decomposition as well.

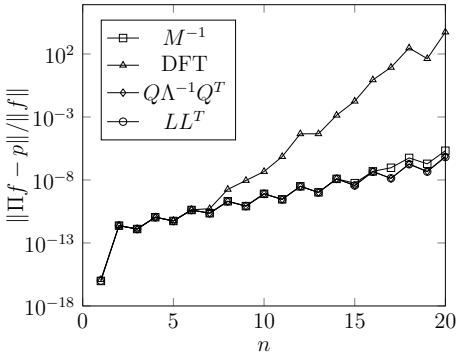
To illustrate that these properties of the solution algorithm do not seem to depend on approximating a smooth solution, we also chose random solution vectors, computed the right-hand side by matrix multiplication, and then applied our four solution algorithms to attempt to recover the solution. In Fig. 3.5, we see exactly the same behavior – the DFT-based method behaving very badly, the Cholesky and

spectral methods seemingly backward stable, and the exact matrix inverse somewhere in between.

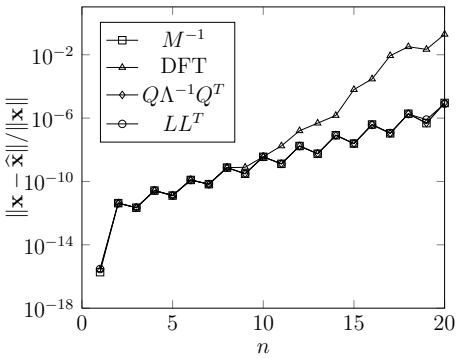
These numerical results highlight several points. Although finite-dimensional norms are equivalent, the bounding constants can dramatically vary as a function of the matrix size. Consequently, we can see comparable Euclidean norm errors but very different M^n-norm errors for, say the matrix inverse and Cholesky methods. Additionally, the ill conditioning of our method is real, although we in fact see that the M^n norm errors (Figs. 3.3b and 3.4b) in our solution process for the (at least empirically) backward-stable methods are in fact quite a bit lower than the 2-norm errors as predicted by the conditioning analysis in Section 3.3. It seems that the empirical performance of the spectral method and Cholesky factorization are the best, although they have slightly different associated costs. The spectral decomposition can be computed more quickly ($\mathcal{O}(n^2)$ versus $\mathcal{O}(n^3)$ for Cholesky), but the per-solve costs is higher – two dense matrix multiplications rather than two triangular solves.



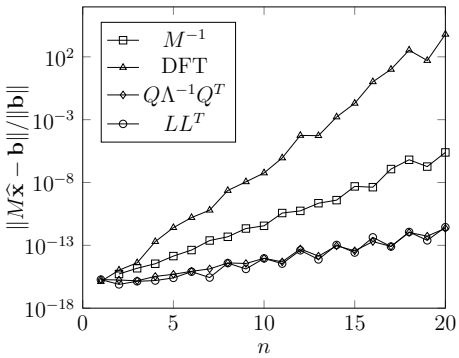
(a) L^2 error between f and p.



(b) L^2 error between Πf and p.

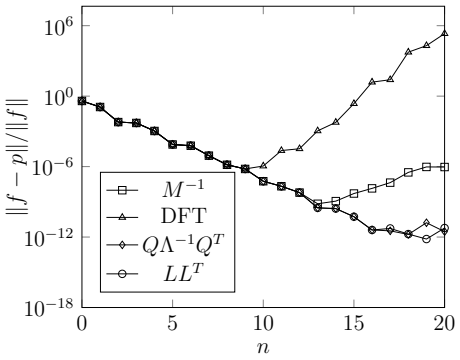


(c) Error in the 2-norm.

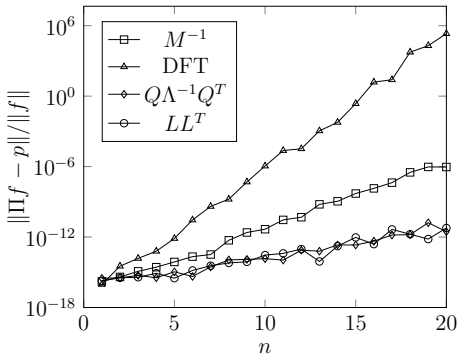


(d) Residual in the 2-norm.

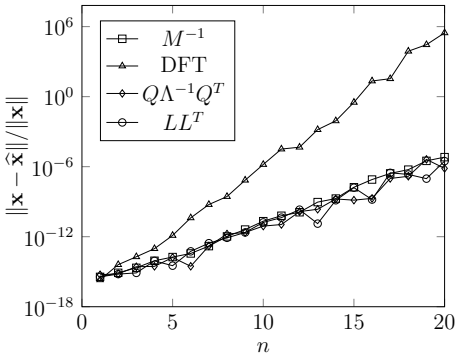
Figure 3.3: Relative error/residual in using the methods described in Section 3.2 to compute the degree n approximation of $f(x) = \frac{1}{1+396(x-0.5)^2}$. M^{-1} refers to the exact inverse, DFT refers to the factored inverse, $Q\Lambda^{-1}Q^T$ refers to the spectral decomposition, and LL^T refers to the Cholesky factorization. We use p to denote the computed approximation, Πf to denote the best approximation, and $\widehat{\mathbf{x}}$ and $\mathbf{x}$ to denote their respective Bernstein coefficients. The vector $\mathbf{b}$ is given by $\mathbf{b}_i = (f(x), B_i^n(x))_{L^2}$.



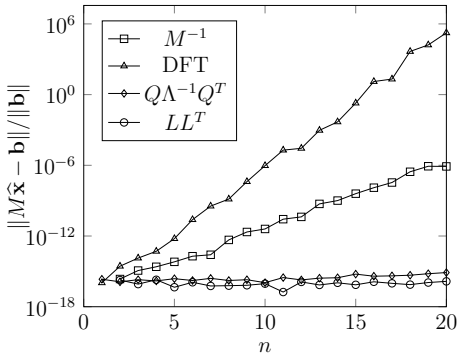
(a) L^2 error between f and p.



(b) L^2 error between Πf and p.

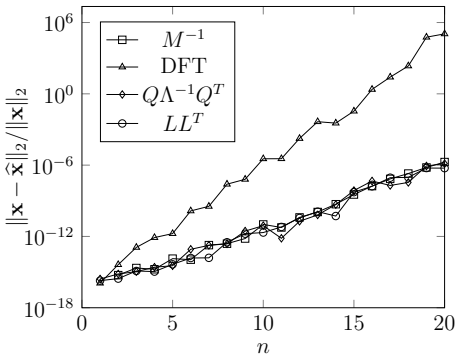


(c) Error in the 2-norm.

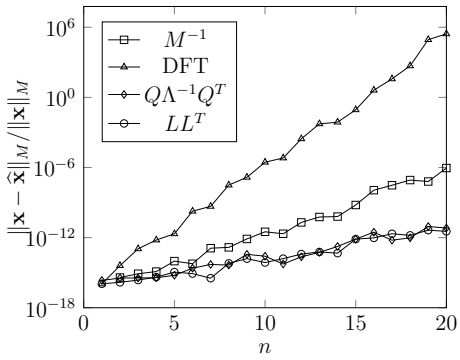


(d) Residual in the 2-norm.

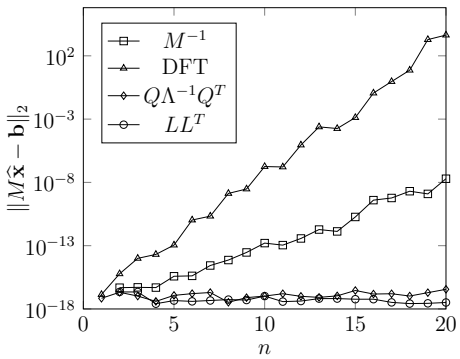
Figure 3.4: Relative error/residual in using the methods described in Section 3.2 to compute the degree n approximation of $f(x) = 0.01 + \frac{x}{x^2+1}$. M^{-1} refers to the exact inverse, DFT refers to the factored inverse, $Q\Lambda^{-1}Q^T$ refers to the spectral decomposition, and LL^T refers to the Cholesky factorization. We use p to denote the computed approximation, Πf to denote the best approximation, and $\widehat{\mathbf{x}}$ and $\mathbf{x}$ to denote their respective Bernstein coefficients. The vector $\mathbf{b}$ is given by $\mathbf{b}_i = (f(x), B_i^n(x))_{L^2}$.



(a) Error in the 2-norm.



(b) Error in the M norm.



(c) Residual in the 2-norm.

Figure 3.5: Error/residual in using the methods described in Section 3.2 to solve $M\mathbf{x} = \mathbf{b}$, where $\mathbf{b}$ is a random vector in $[-0.5, 0.5]^{n+1}$. M^{-1} refers to the exact inverse, DFT refers to the factored inverse, $Q\Lambda^{-1}Q^T$ refers to the spectral decomposition, and LL^T refers to the Cholesky factorization. We use $\widehat{\mathbf{x}}$ to denote the computed solution.

CHAPTER FOUR

Bernstein–Vandermonde Matrix

This chapter published as: Larry Allen and Robert C. Kirby. "Structured inversion of the Berstein Vandermonde matrix". In: *SIAM Journal of Matrix Analysis and Applications* 42.2 (2021), pp. 557–577

4.1 Inverse Formulas

4.1.1 General Formulas

In [35], it was shown that if the monomial basis is used, then the inverse of a Bézout matrix is a Hankel matrix. As a consequence of the arguments involved, one obtains a formula for the inverse of the monomial Vandermonde matrix in terms of its transpose, a diagonal matrix, and a particular Bézout matrix. Since none of the arguments require anything specific to the monomial basis, we can generalize the arguments to any basis, which we summarize here.

For $t \in \mathbb{R}$, let $\mathbf{b}^n(t)$ denote the column vector $(b_0^n(t), \ldots, b_n^n(t))^T$. As a consequence of (2.37), if $s \neq t$, then

$$\begin{aligned}(\mathbf{b}^n(s))^T \operatorname{Bez}(v,w)\mathbf{b}^n(t) &= \sum_{i,j=0}^{n} \operatorname{Bez}_{ij}(v,w) b_i^n(s) b_j^n(t) \\ &= \frac{v(s)w(t) - v(t)w(s)}{s-t}.\end{aligned}$$

This implies that

$$\begin{aligned}(\mathbf{b}^n(t))^T \operatorname{Bez}(v,w)\mathbf{b}^n(t) &= \lim_{\varepsilon\to 0}\left[(\mathbf{b}^n(t))^T \operatorname{Bez}(v,w)\mathbf{b}^n(t+\varepsilon)\right] \\ &= \lim_{\varepsilon\to 0}\frac{v(t+\varepsilon)w(t)-v(t)w(t+\varepsilon)}{\varepsilon} \\ &= v'(t)w(t)-v(t)w'(t).\end{aligned}$$

Therefore, if v has simple zeros at $\mathbf{x}_i$ and $w(\mathbf{x}_i)\neq 0$ for each $0\leq i\leq n$, then

$$V^n(\mathbf{x})\operatorname{Bez}(v,w)\left(V^n(\mathbf{x})\right)^T = \operatorname{diag}(v'(\mathbf{x}_j)w(\mathbf{x}_j))_{j=0}^n, \tag{4.1}$$

and so we have the following:

Theorem 4.1. *The inverse of $V^n(\mathbf{x})$ is given by*

$$\left(V^n(\mathbf{x})\right)^{-1} = \operatorname{Bez}(v,w)\left(V^n(\mathbf{x})\right)^T \operatorname{diag}\left(\frac{1}{v'(\mathbf{x}_j)w(\mathbf{x}_j)}\right)_{j=0}^n, \tag{4.2}$$

where v and w are any polynomials of degree at most $n+1$ satisfying $v(\mathbf{x}_i)=0$, $v'(\mathbf{x}_i)\neq 0$, and $w(\mathbf{x}_i)\neq 0$ for each $0\leq i\leq n$.

Corollary 4.2. *The inverse of $V^n(\mathbf{x})$ is given by*

$$\left(V^n(\mathbf{x})\right)^{-1} = \operatorname{Bez}(v,1)\left(V^n(\mathbf{x})\right)^T \operatorname{diag}\left(\frac{1}{v'(\mathbf{x}_j)}\right)_{j=0}^n, \tag{4.3}$$

where v is any polynomial of degree $n+1$ that has simple zeros at $\mathbf{x}_i$ for each $0\leq i\leq n$.

4.1.2 Bernstein–Vandermonde

We now focus on the case where $0\leq \mathbf{x}_i\leq 1$ for each $0\leq i\leq n$ and the basis consists of the Bernstein polynomials. In order to apply Theorem 4.1 or Corollary 4.2, we need a method for computing the entries of the Bernstein–Bézout matrix; in

fact, they satisfy a recurrence relation involving the Bernstein coefficients of the polynomials.

Theorem 4.3. *If* $v(t) = \sum_{i=0}^{n+1} \mathbf{v}_i B_i^{n+1}(t)$ *and* $w(t) = \sum_{i=0}^{n+1} \mathbf{w}_i B_i^{n+1}(t)$ *are polynomials of degree at most* $n+1$*, then the entries* b_{ij} *of the Bernstein–Bézout matrix* $\mathrm{Bez}(v, w)$ *generated by* v *and* w *satisfy*

$$b_{ij} = \frac{1}{(i+1)(n-j+1)} \left[j(n-i) b_{i+1,j-1} + (n+1)^2 \left(\mathbf{v}_{i+1}\mathbf{w}_j - \mathbf{v}_j \mathbf{w}_{i+1} \right) \right]. \tag{4.4}$$

Proof. By (2.2), we have that

$$t B_j^n(t) = \frac{j+1}{n+1} B_{j+1}^{n+1}(t) \tag{4.5}$$

and

$$B_j^n(t) = \frac{n-j+1}{n+1} B_j^{n+1}(t) + \frac{j+1}{n+1} B_{j+1}^{n+1}(t). \tag{4.6}$$

Therefore,

$$\begin{aligned} s \sum_{i,j=0}^{n} b_{ij} B_i^n(s) B_j^n(t) &= \sum_{i,j=0}^{n} \frac{i+1}{n+1} b_{ij} B_{i+1}^{n+1}(s) B_j^n(t) \\ &= \sum_{i,j=0}^{n} \frac{(i+1)(n-j+1)}{(n+1)^2} b_{ij} B_{i+1}^{n+1}(s) B_j^{n+1}(t) \\ &\quad + \sum_{i,j=0}^{n} \frac{(i+1)(j+1)}{(n+1)^2} b_{ij} B_{i+1}^{n+1}(s) B_{j+1}^{n+1}(t). \end{aligned}$$

Similarly, we have that

$$\begin{aligned} t\sum_{i,j=0}^{n} b_{ij}B_i^n(s)B_j^n(t) &= \sum_{i,j=0}^{n} \frac{j+1}{n+1} b_{ij}B_i^n(s)B_{j+1}^{n+1}(t) \\ &= \sum_{i,j=0}^{n} \frac{(n-i+1)(j+1)}{(n+1)^2} b_{ij}B_i^{n+1}(s)B_{j+1}^{n+1}(t) \\ &\quad + \sum_{i,j=0}^{n} \frac{(i+1)(j+1)}{(n+1)^2} b_{ij}B_{i+1}^{n+1}(s)B_{j+1}^{n+1}(t). \end{aligned}$$

This implies that

$$\begin{aligned} (s-t)\sum_{i,j=0}^{n} b_{ij}B_i^n(s)B_j^n(t) &= \sum_{i,j=0}^{n} \frac{(i+1)(n-j+1)}{(n+1)^2} b_{ij}B_{i+1}^{n+1}(s)B_j^{n+1}(t) \\ &\quad - \sum_{i,j=0}^{n} \frac{(n-i+1)(j+1)}{(n+1)^2} b_{ij}B_i^{n+1}(s)B_{j+1}^{n+1}(t), \end{aligned}$$

and so

$$\begin{aligned} (s-t)\sum_{i,j=0}^{n} b_{ij}B_i^n(s)B_j^n(t) &= \sum_{i,j=0}^{n+1} \frac{i(n-j+1)}{(n+1)^2} b_{i-1,j}B_i^{n+1}(s)B_j^{n+1}(t) \\ &\quad - \sum_{i,j=0}^{n+1} \frac{j(n-i+1)}{(n+1)^2} b_{i,j-1}B_i^{n+1}(s)B_j^{n+1}(t). \end{aligned}$$

On the other hand,

$$v(s)w(t) - v(t)w(s) = \sum_{i,j=0}^{n+1} (\mathbf{v}_i\mathbf{w}_j - \mathbf{v}_j\mathbf{w}_i)B_i^{n+1}(s)B_j^{n+1}(t), \tag{4.7}$$

and so the result follows by comparing coefficients in (2.37). □

We can form the first column and last row of $\mathrm{Bez}(v,w)$ by using (4.4) with $j=0$ and $i=n$. The rest of the columns can then be built by applying the recurrence relation and using the symmetry of $\mathrm{Bez}(v,w)$. Therefore,

Corollary 4.4. $\mathrm{Bez}(v,w)$ *can be constructed in* $\mathcal{O}(n^2)$ *operations.*

By repeatedly applying (4.4), we obtain a closed form for the entries of the Bernstein–Bézout matrix.

Corollary 4.5. *If* $v(t) = \sum_{i=0}^{n+1} \mathbf{v}_i B_i^{n+1}(t)$ *and* $w(t) = \sum_{i=0}^{n+1} \mathbf{w}_i B_i^{n+1}(t)$ *are polynomials of degree at most* $n+1$*, then the entries* b_{ij} *of the Bernstein–Bézout matrix* $\mathrm{Bez}(v, w)$ *generated by* v *and* w *are given by*

$$b_{ij} = \frac{1}{\binom{n}{i}\binom{n}{j}} \sum_{k=0}^{n_{ij}} \binom{n+1}{i+k+1} \binom{n+1}{j-k} \left(\mathbf{v}_{i+k+1}\mathbf{w}_{j-k} - \mathbf{v}_{j-k}\mathbf{w}_{i+k+1}\right), \tag{4.8}$$

where $n_{ij} = \min\{j, n-i\}$.

Constructing $\mathrm{Bez}(v, w)$ by using (4.8) would require $\mathcal{O}(n^3)$ operations, and so Corollary 4.4 implies that (4.8) is not optimal for building $\mathrm{Bez}(v, w)$; however, since $\binom{n}{i} = 0$ whenever $i > n$, the upper limit of the sum in (4.8) can be replaced with $k = n$, and so we recognize that $\mathrm{Bez}(v, w)$ is given by a difference of matrix products.

Corollary 4.6. *Let* $v(t) = \sum_{i=0}^{n+1} \mathbf{v}_i B_i^{n+1}(t)$ *and* $w(t) = \sum_{i=0}^{n+1} \mathbf{w}_i B_i^{n+1}(t)$ *be polynomials of degree at most* $n+1$*. Define the Toeplitz matrices* $T^{v,n}$ *and* $T^{w,n}$ *by*

$$T_{ij}^{v,n} = \binom{n+1}{j-i}\mathbf{v}_{j-i} \qquad \text{and} \qquad T_{ij}^{w,n} = \binom{n+1}{j-i}\mathbf{w}_{j-i},$$

define the Hankel matrices $H^{v,n}$ *and* $H^{w,n}$ *by*

$$H_{ij}^{v,n} = \binom{n+1}{i+j+1}\mathbf{v}_{i+j+1} \qquad \text{and} \qquad H_{ij}^{w,n} = \binom{n+1}{i+j+1}\mathbf{w}_{i+j+1},$$

and let $\Delta^n = \mathrm{diag}\left(\binom{n}{j}\right)_{j=0}^{n}$*. Then the Bernstein–Bézout matrix* $\mathrm{Bez}(v, w)$ *generated by* v *and* w *is given by*

$$\mathrm{Bez}(v, w) = (\Delta^n)^{-1}\left[H^{v,n}T^{w,n} - H^{w,n}T^{v,n}\right](\Delta^n)^{-1}. \tag{4.9}$$

It is interesting to note the similarities between (3.34) and (4.9) despite the different matrices and bases being considered.

Observe that

$$\begin{aligned}\prod_{i=0}^{n}(x-\mathbf{x}_i) &= \prod_{i=0}^{n}[-\mathbf{x}_i(1-x)+(1-\mathbf{x}_i)x] \\ &= \sum_{k=0}^{n+1}\frac{(-1)^{n-k+1}}{\binom{n+1}{k}}\sigma_{n-k+1}^{n+1}(\mathbf{x})B_k^{n+1}(x),\end{aligned}$$

where

$$\sigma_k^{n+1} = \sum_{\substack{A\subseteq\{\mathbf{x}_0,\ldots,\mathbf{x}_n\}\\ |A|=k}}\prod_{\mathbf{x}_i\in A}\mathbf{x}_i\prod_{\mathbf{x}_i\notin A}(1-\mathbf{x}_i). \tag{4.10}$$

In addition, since

$$1 = \sum_{i=0}^{n+1}B_i^{n+1}(x) \tag{4.11}$$

and

$$\frac{d}{dx}\left[\prod_{i=0}^{n}(x-\mathbf{x}_i)\right]_{x=x_j} = \prod_{i\in\{0,\ldots,n\}\setminus\{j\}}(\mathbf{x}_j-\mathbf{x}_i), \tag{4.12}$$

we can combine the previous discussion with Corollaries 4.2 and 4.6 to obtain a decomposition of the inverse of the Bernstein–Vandermonde matrix.

Theorem 4.7. *Define the Hankel matrices H^n and $\widetilde{H}^n(\mathbf{x})$ by*

$$H_{ij}^n = \binom{n+1}{i+j+1} \qquad \text{and} \qquad \widetilde{H}_{ij}^n(\mathbf{x}) = (-1)^{n-i-j}\sigma_{n-i-j}^{n+1}(\mathbf{x}),$$

define the Toeplitz matrices T^n and $\widetilde{T}^n(\mathbf{x})$ by

$$T_{ij}^n = \binom{n+1}{j-i} \qquad \text{and} \qquad \widetilde{T}_{ij}^n(\mathbf{x}) = (-1)^{n+i-j+1}\sigma_{n+i-j+1}^{n+1}(\mathbf{x}),$$

and define the diagonal matrices $D^n(\mathbf{x})$ *and* Δ^n *by*

$$D^n(\mathbf{x}) = \operatorname{diag}\left(\prod_{i\in\{0,\dots,n\}\setminus\{j\}} (\mathbf{x}_j - \mathbf{x}_i)\right)_{j=0}^{n} \qquad \textit{and} \qquad \Delta^n = \operatorname{diag}\left(\binom{n}{j}\right)_{j=0}^{n}.$$

In addition, let $\widetilde{V}^n(\mathbf{x})$ *be the scaled Bernstein–Vandermonde matrix*

$$\widetilde{V}^n_{ij}(\mathbf{x}) = \mathbf{x}_i^j (1-\mathbf{x}_i)^{n-j}.$$

Then

$$(V^n(\mathbf{x}))^{-1} = (\Delta^n)^{-1}\left[\widetilde{H}^n(\mathbf{x})T^n - H^n\widetilde{T}^n(\mathbf{x})\right]\left(\widetilde{V}^n(\mathbf{x})\right)^T (D^n(\mathbf{x}))^{-1}. \tag{4.13}$$

4.1.3 Equispaced Nodes

We now briefly focus on the case where $\mathbf{x}_i = i/n$ for each $0 \le i \le n$. For these nodes, we observe that

$$\begin{aligned}
\prod_{i=0}^{n}\left(x - \frac{i}{n}\right) &= -x(1-x)\prod_{i=1}^{n-1}\left[-\frac{i}{n}(1-x) + \left(1-\frac{i}{n}\right)x\right] \\
&= -x(1-x)\sum_{k=0}^{n-1}\frac{(-1)^{n-k-1}}{\binom{n-1}{k}}\sigma^{n-1}_{n-k-1}\left(\left(\frac{1}{n},\dots,\frac{n-1}{n}\right)\right)B_k^{n-1}(x) \\
&= -\frac{(n-1)!}{n^{n-1}}x(1-x)\sum_{k=0}^{n-1}\frac{(-1)^{n-k-1}}{\binom{n-1}{k}}\mu^{n-1}_{n-k-1}B_k^{n-1}(x) \\
&= \frac{(n-1)!}{n^{n-1}}\sum_{k=0}^{n+1}\frac{(-1)^{n-k+1}}{\binom{n+1}{k}}\mu^{n-1}_{n-k}B_k^{n+1}(x),
\end{aligned}$$

where

$$\mu_k^{n-1} = \begin{cases} \displaystyle\sum_{\substack{A\subseteq\{1,\dots,n-1\}\\ |A|=k}}\prod_{i\in A}\frac{i}{n-i}, & \text{if } 0\le k\le n-1; \\ 0, & \text{otherwise.} \end{cases} \tag{4.14}$$

In addition, we have that

$$\prod_{i\in\{0,\dots,n\}\setminus\{j\}} (j/n - i/n) = (-1)^{n-j}\frac{j!(n-j)!}{n^n}. \tag{4.15}$$

Corollary 4.8. *Define the Hankel matrices H^n and $\widetilde{H}^n$ by*

$$H^n_{ij} = \binom{n+1}{i+j+1} \qquad \text{and} \qquad \widetilde{H}^n_{ij} = \frac{(n-1)!}{n^{n-1}}(-1)^{n-i-j}\mu^{n-1}_{n-i-j-1},$$

define the Toeplitz matrices T^n and $\widetilde{T}^n$ by

$$T^n_{ij} = \binom{n+1}{j-i} \qquad \text{and} \qquad \widetilde{T}^n_{ij} = \frac{(n-1)!}{n^{n-1}}(-1)^{n+i-j+1}\mu^{n-1}_{n+i-j},$$

and define the diagonal matrices D^n and Δ^n by

$$D^n = \operatorname{diag}\left((-1)^{n-j}\left[j!(n-j)!\right]\right)_{j=0}^{n} \qquad \text{and} \qquad \Delta^n = \operatorname{diag}\left(\binom{n}{j}\right)_{j=0}^{n}.$$

In addition, let $\widetilde{V}^n$ be the scaled Bernstein–Vandermonde matrix

$$\widetilde{V}^n_{ij} = i^j(n-i)^{n-j}.$$

Then the inverse of $V^n = V^n(0, 1/n, \dots, 1)$ is given by

$$(V^n)^{-1} = (\Delta^n)^{-1}\left[\widetilde{H}^n T^n - H^n\widetilde{T}^n\right]\left(\widetilde{V}^n\right)^T (D^n)^{-1}. \tag{4.16}$$

4.2 Applying the Inverse

Now, we describe several approaches to applying $(V^n(\mathbf{x}))^{-1}$ to a vector.

***LU* factorization** We can decompose $V^n(\mathbf{x})$ as

$$V^n(\mathbf{x}) = R^n(\mathbf{x})U^n(\mathbf{x}), \tag{4.17}$$

where $R^n(\mathbf{x})$ and $U^n(\mathbf{x})$ are lower and upper triangular matrices, respectively. Widely available in libraries, computing $R^n(\mathbf{x})$ and $U^n(\mathbf{x})$ requires $\mathcal{O}(n^3)$ operations, and each of the subsequent triangular solves require $\mathcal{O}(n^2)$ operations to perform.

Exact inverse In light of Corollary 4.2, we can directly form $(V^n(\mathbf{x}))^{-1}$. By Theorem 4.3, we can form the inverse in $\mathcal{O}(n^2)$ operations. The inverse can then be applied to any vector in $\mathcal{O}(n^2)$ operations using the standard algorithm.

DFT-based application By Theorem 4.7, we can express the inverse of $V^n(\mathbf{x})$ in terms of its (scaled) transpose and Hankel, Toeplitz, and diagonal matrices. The matrices $H^n(\mathbf{x})$ and $T^n(\mathbf{x})$ can be formed in $\mathcal{O}(n)$ operations, but the matrices $\widetilde{H}^n(\mathbf{x})$ and $\widetilde{T}^n(\mathbf{x})$ cannot, since each entry involves a sum. Even though the Hankel and Toeplitz matrices can be applied to a vector in $\mathcal{O}(n \log n)$ operations via circulant embedding and a couple of FFT/iFFT [62], the scaled transpose still requires $\mathcal{O}(n^2)$ operations to apply to a vector.

Newton algorithm In [3], Ainsworth and Sanchez gave a fast algorithm for solving $V^n(\mathbf{x})\mathbf{c} = \mathbf{b}$ via a recursion relation for the Bézier control points of the Newton form of the Lagrangian interpolant. The numerical results given suggest that the algorithm is stable even for high polynomial degree, and so we include it for comparison.

Marco–Martínez algorithm In [46], Marco and Martínez gave a bidiagonal factorization of $(V^n(\mathbf{x}))^{-1}$ in the case where the nodes are contained in the interior of $[0, 1]$, which can be extended to the case where one or both of the endpoints are

contained in the list of nodes. The bidiagonal factors can be built in $\mathcal{O}(n^2)$ operations, and the inverse can subsequently be applied to a vector in $\mathcal{O}(n^2)$ operations. The startup and application costs are similar to the exact inverse method, and so we include it for comparison.

4.3 Conditioning and Accuracy

In Section 3.3, we gave a partial explanation of the relatively high accuracy, compared to its large 2-norm condition number, of working with the Bernstein mass matrix. We will use similar reasoning to investigate the performance of the Bernstein–Vandermonde matrix.

Lemma 4.9.

$$\kappa_{M^n \to 2}\left(V^n(\mathbf{x})\right) = \kappa_2\left(V^n(\mathbf{x})\left(M^n\right)^{-1/2}\right). \tag{4.18}$$

Proof. By (3.39), we that if $\mathbf{y} \neq \mathbf{0}$, then

$$\frac{\left\|\left(V^n(\mathbf{x})\right)^{-1}\mathbf{y}\right\|_{M^n}}{\|\mathbf{y}\|_2} = \frac{\left\|\left(M^n\right)^{1/2}\left(V^n(\mathbf{x})\right)^{-1}\mathbf{y}\right\|_2}{\|\mathbf{y}\|_2}, \tag{4.19}$$

and so

$$\left\|\left(V^n(\mathbf{x})\right)^{-1}\right\|_{2\to M^n} = \left\|\left(M^n\right)^{1/2}\left(V^n(\mathbf{x})\right)^{-1}\right\|_2. \tag{4.20}$$

Let $\mathbf{z} = \left(M^n\right)^{1/2}\mathbf{y}$. Then $\mathbf{z} \neq \mathbf{0}$ and

$$\begin{aligned}
\frac{\|V^n(\mathbf{x})\mathbf{y}\|_2^2}{\|\mathbf{y}\|_{M^n}^2} &= \frac{\mathbf{y}^T\left(V^n(\mathbf{x})\right)^T V^n(\mathbf{x})\mathbf{y}}{\mathbf{y}^T M^n \mathbf{y}} \\
&= \frac{\left(V^n(\mathbf{x})\left(M^n\right)^{-1/2}\mathbf{z}\right)^T\left(V^n(\mathbf{x})\left(M^n\right)^{-1/2}\mathbf{z}\right)}{\mathbf{z}^T\mathbf{z}} \\
&= \frac{\left\|V^n(\mathbf{x})\left(M^n\right)^{-1/2}\mathbf{z}\right\|_2^2}{\|\mathbf{z}\|_2^2},
\end{aligned}$$

and so

$$\|V^n(\mathbf{x})\|_{M^n\to 2} = \left\|V^n(\mathbf{x})\,(M^n)^{-1/2}\right\|_2. \tag{4.21}$$

It follows that

$$\begin{aligned}
\kappa_{M^n\to 2}\left(V^n(\mathbf{x})\right) &= \|V^n(\mathbf{x})\|_{M^n\to 2}\left\|(V^n(\mathbf{x}))^{-1}\right\|_{2\to M^n} \\
&= \left\|V^n(\mathbf{x})\,(M^n)^{-1/2}\right\|_2\left\|(M^n)^{1/2}\,(V^n(\mathbf{x}))^{-1}\right\|_2 \\
&= \left\|V^n(\mathbf{x})\,(M^n)^{-1/2}\right\|_2\left\|\left(V^n(\mathbf{x})\,(M^n)^{-1/2}\right)^{-1}\right\|_2 \\
&= \kappa_2\left(V^n(\mathbf{x})\,(M^n)^{-1/2}\right).
\end{aligned}$$

□

To analyze $\kappa_2\left(V^n(\mathbf{x})\,(M^n)^{-1/2}\right)$, we will need reference to the Legendre–Vandermonde matrix. The Legendre–Vandermonde matrix is the $(n+1)\times(n+1)$ matrix $\widehat{V}^n(\mathbf{x})$ given by

$$\widehat{V}^n_{ij}(\mathbf{x}) = L^j(\mathbf{x}_i). \tag{4.22}$$

Define T^n to be the $(n+1)\times(n+1)$ matrix satisfying $T^n\mathbf{\Theta}(p) = \mathbf{\Pi}(p)$ for all polynomials p of degree at most n. In particular, we have the relationships

$$\widehat{V}^n(\mathbf{x}) = V^n(\mathbf{x})T^n \tag{4.23}$$

and

$$Q^n = T^n\,\mathrm{diag}\left(\sqrt{(2j+1)\lambda^n_j}\right)_{j=0}^n. \tag{4.24}$$

The spectral decomposition of M^n combined with Lemma 4.9, (4.23), and (4.24) suggest a relationship between the $M^n \to 2$ condition number of $V^n(x)$ and the 2-norm condition number of $\widehat{V}^n(\mathbf{x})$ scaled by a diagonal matrix.

Theorem 4.10.

$$\kappa_{M^n \to 2}\left(V^n(\mathbf{x})\right) = \kappa_2\left(\widehat{V}^n(\mathbf{x}) \operatorname{diag}\left(\sqrt{2j+1}\right)_{j=0}^{n}\right). \tag{4.25}$$

Proof. We have that

$$\begin{aligned} V^n(\mathbf{x})\left(M^n\right)^{-1/2} &= V^n(\mathbf{x}) Q^n \left(\Lambda^n\right)^{-1/2} \left(Q^n\right)^T \\ &= V^n(\mathbf{x}) T^n \operatorname{diag}\left(\sqrt{(2j+1)\lambda_j^n}\right)_{j=0}^{n} \operatorname{diag}\left(\frac{1}{\sqrt{\lambda_j^n}}\right)_{j=0}^{n} \left(Q^n\right)^T \\ &= \widehat{V}^n(\mathbf{x}) \operatorname{diag}\left(\sqrt{2j+1}\right)_{j=0}^{n} \left(Q^n\right)^T. \end{aligned}$$

Since Q^n is orthogonal, the result follows from Lemma 4.9. □

In [29], Gautschi computed the condition number in the Frobenius norm $\|\cdot\|_F$ for Vandermonde matrices associated with families of orthonormal polynomials. Since the diagonal matrix in Theorem 4.10 contains the reciprocal of the L^2 norms of the Legendre polynomials, we can adapt his arguments to bound $\left(\widehat{V}^n(\mathbf{x}) \operatorname{diag}\left(\sqrt{2j+1}\right)_{j=0}^{n}\right)^{-1}$ in the Frobenius norm. In addition, the Legendre polynomials have the special property that $|L^j(x)| \leq 1$ for all $0 \leq x \leq 1$, and so we can bound the Frobenius norm of $\widehat{V}^n(\mathbf{x}) \operatorname{diag}\left(\sqrt{2j+1}\right)_{j=0}^{n}$.

For each integer $0 \leq j \leq n$, let $\ell^{j,n}$ be the j^{th} Lagrange polynomial with respect to $\mathbf{x}$; that is,

$$\ell^{j,n}(x) = \prod_{i \in \{0,\ldots,n\} \setminus \{j\}} \frac{x - \mathbf{x}_i}{\mathbf{x}_j - \mathbf{x}_i}. \tag{4.26}$$

Define $\mathbf{w}^n \in \mathbb{R}^{n+1}$ by

$$\mathbf{w}_j^n = \|\ell^{j,n}\|_{L^2}. \tag{4.27}$$

Corollary 4.11.

$$\kappa_{M^n \to 2}(V^n(\mathbf{x})) \leq (n+1)^{3/2} \|\mathbf{w}^n\|_2. \tag{4.28}$$

Proof. Since $|L^j(x)| \leq 1$ for all $0 \leq x \leq 1$, we have that

$$\begin{aligned}
\left\| \widehat{V}^n(\mathbf{x}) \operatorname{diag}\left(\sqrt{2j+1}\right)_{j=0}^n \right\|_F &= \left(\sum_{i,j=0}^n (2j+1) \left(L^j(\mathbf{x}_i)\right)^2 \right)^{1/2} \\
&\leq \left(\sum_{i,j=0}^n (2j+1) \right)^{1/2} \\
&= (n+1)^{3/2}.
\end{aligned}$$

Since

$$\ell^{j,n}(\mathbf{x}_i) = \begin{cases} 1, & \text{if } i = j; \\ 0, & \text{if } i \neq j; \end{cases} \tag{4.29}$$

we have that $\left(\widehat{V}^n(\mathbf{x})\right)^{-1}_{ij} = \boldsymbol{\Theta}\left(\ell^{j,n}\right)_i$. Therefore, by (2.9), we have that

$$\begin{aligned}
\|\mathbf{w}^n\|_2 &= \left(\sum_{j=0}^{n} \left(\mathbf{w}_j^n\right)^2\right)^{1/2} \\
&= \left(\sum_{j=0}^{n} \int_0^1 \left(\ell^{j,n}(x)\right)^2 dx\right)^{1/2} \\
&= \left(\sum_{j=0}^{n} \int_0^1 \sum_{i,k=0}^{n} \boldsymbol{\Theta}(\ell^{j,n})_i \boldsymbol{\Theta}(\ell^{j,n})_k L^i(x) L^k(x) dx\right)^{1/2} \\
&= \left(\sum_{j=0}^{n} \sum_{i,k=0}^{n} \boldsymbol{\Theta}(\ell^{j,n})_i \boldsymbol{\Theta}(\ell^{j,n})_k \int_0^1 L^i(x) L^k(x) dx\right)^{1/2} \\
&= \left(\sum_{i,j=0}^{n} \boldsymbol{\Theta}(\ell^{j,n})_i^2 \int_0^1 (L^i(x))^2 dx\right)^{1/2} \\
&= \left(\sum_{i,j=0}^{n} \frac{\boldsymbol{\Theta}(\ell^{j,n})_i^2}{2i+1}\right)^{1/2} \\
&= \left\| \operatorname{diag}\left(\frac{1}{\sqrt{2i+1}}\right)_{i=0}^{n} \left(\widehat{V}^n(\mathbf{x})\right)^{-1} \right\|_F \\
&= \left\| \left(\widehat{V}^n(\mathbf{x}) \operatorname{diag}\left(\sqrt{2j+1}\right)_{j=0}^{n}\right)^{-1} \right\|_F .
\end{aligned}$$

Since the condition number in the 2 norm is bounded above by the condition number in the Frobenius norm, the result follows from Theorem 4.10. □

Theorem 4.10 relates the condition number of the Bernstein–Vandermonde matrix in the $M^n \to 2$ norm to the 2 norm condition number of the (normalized) Legendre–Vandermonde matrix. Since the 2 norm conditioning of the Vandermonde matrix is better in the Legendre basis than the Bernstein basis [29], this result indicates that, when measuring the relevant L^2 norm rather than the Euclidean norm of the solution process, we can expect much better results than the ill-conditioning of the Bernstein–Vandermonde matrix suggests. Corollary 4.11 gives an estimate of this condition

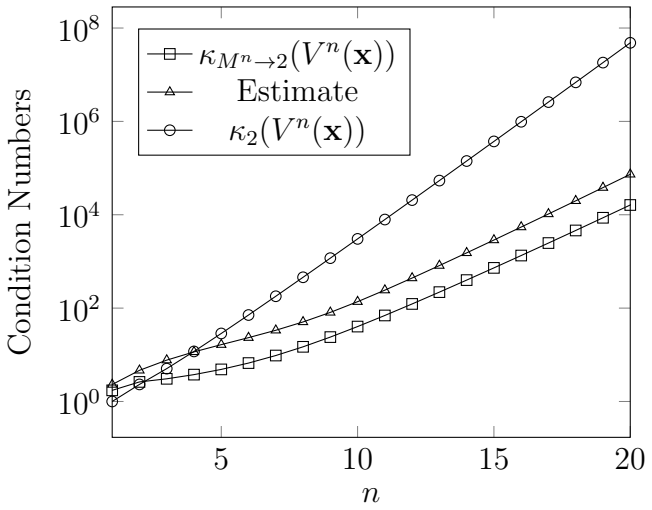


Figure 4.1: Comparsion of the condition number of V^n in the $M^n \to 2$ norm and the 2 norm for $1 \leq n \leq 20$, where V^n is the matrix described in Section 4.1.3. We also include the estimate given in Corollary 4.11. We use Theorem 4.10 to compute $\kappa_{M^n\to 2}(V^n(\mathbf{x}))$.

number that can be numerically evaluated. A plot of the condition numbers (and the estimate) up to degree 20 is given in Fig. 4.1.

4.4 Higher Dimensions

Now, in the case of equispaced nodes on the d-simplex, we can use block-recursive structure as discussed in [39] to give low-complexity algorithms for the inversion of simplicial Bernstein–Vandermonde matrices. This gives, at least for the equispaced lattice, an alternate algorithm for solving simplicial Bernstein interpolation problems to the one worked out in [1].

4.4.1 Notation and Preliminaries

For an integer $d \geq 1$, let S_d be a nondegenerate simplex in $\mathbb{R}^d$. Let $\{\mathbf{v}_i\}_{i=0}^d \subset \mathbb{R}^d$ be the vertices of S_d, and let $\{\mathbf{b}_i\}_{i=0}^d$ denote the barycentric coordinates of S_d. Each

$\mathbf{b}_i$ is an affine map from $\mathbb{R}^d$ to $\mathbb{R}$ such that

$$\mathbf{b}_i(\mathbf{v}_j) = \begin{cases} 1, & \text{if } i = j; \\ 0, & \text{if } i \neq j; \end{cases} \tag{4.30}$$

for each vertex $\mathbf{v}_j$. Each $\mathbf{b}_i$ is nonnegative on S_d, and

$$\sum_{i=0}^{d} \mathbf{b}_i = 1. \tag{4.31}$$

A multiindex $\boldsymbol{\beta}$ of length $d+1$ is a $(d+1)$-tuple of nonnegative integers, written

$$\boldsymbol{\beta} = (\boldsymbol{\beta}_0, \ldots, \boldsymbol{\beta}_d). \tag{4.32}$$

The order of $\boldsymbol{\beta}$, denoted $|\boldsymbol{\beta}|$, is given by

$$|\boldsymbol{\beta}| = \sum_{i=0}^{d} \boldsymbol{\beta}_i. \tag{4.33}$$

The factorial $\boldsymbol{\beta}!$ of a multiindex $\boldsymbol{\beta}$ is defined by

$$\boldsymbol{\beta}! = \prod_{i=0}^{d} \boldsymbol{\beta}_i. \tag{4.34}$$

For a multiindex $\boldsymbol{\beta}$ of length $d+1$, denote by $\boldsymbol{\beta}'$ the multiindex of length d given by

$$\boldsymbol{\beta}' = (\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_d). \tag{4.35}$$

Given a nonnegative integer b and a multiindex $\boldsymbol{\beta}' = (\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_d)$ of length d, define a new multiindex $b \vdash \boldsymbol{\beta}$ of length $d+1$ by

$$b \vdash \boldsymbol{\beta}' = (b, \boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_d). \tag{4.36}$$

In particular,

$$\boldsymbol{\beta} = \boldsymbol{\beta}_0 \vdash \boldsymbol{\beta}'. \tag{4.37}$$

Multiindices have a natural partial ordering given by

$$\boldsymbol{\beta} \leq \widetilde{\boldsymbol{\beta}} \qquad \text{if and only if} \qquad \boldsymbol{\beta}_i \leq \widetilde{\boldsymbol{\beta}}_i \quad \text{for all} \quad 0 \leq i \leq d. \tag{4.38}$$

The Bernstein polynomials of degree n on the d-simplex S_d are defined by

$$B^n_{\boldsymbol{\beta}} = \frac{n!}{\boldsymbol{\beta}!} \prod_{i=0}^{d} \mathbf{b}_i^{\boldsymbol{\beta}_i}. \tag{4.39}$$

The complete set of Bernstein polynomials $\{B^n_{\boldsymbol{\beta}}\}_{|\boldsymbol{\beta}|=n}$ form a basis for polynomials in d variables of complete degree at most n.

If $n_0 \leq n$, then any polynomial expressed in the basis $\{B^{n_0}_{\boldsymbol{\beta}}\}_{|\boldsymbol{\beta}|=n_0}$ can also be expressed in the basis $\{B^n_{\boldsymbol{\beta}}\}_{|\boldsymbol{\beta}|=n}$. We denote by $E^{d,n_0,n}$ the $\binom{n+d}{d} \times \binom{n_0+d}{d}$ matrix that maps the coefficients of the degree n_0 representation to the coefficients of the degree n representation. The matrix $E^{d,n_0,n}$ is sparse and can be applied matrix-free [36], if desired.

For a nonnegative integer m and a set of distinct nodes $\{\mathbf{x}^m_{\boldsymbol{\alpha}}\}_{|\boldsymbol{\alpha}|=m} \subset S_d$, define the Bernstein–Vandermonde matrix $V^{d,m,n}$ to be the $\binom{m+d}{d} \times \binom{n+d}{d}$ matrix given by

$$V^{d,m,n}_{\boldsymbol{\alpha}\boldsymbol{\beta}} = B^n_{\boldsymbol{\beta}}(\mathbf{x}^m_{\boldsymbol{\alpha}}) \tag{4.40}$$

Theorem 4.12. *Let d, m, n_0, and n be nonnegative integers with $d \geq 1$ and $n_0 \leq n$, and let $\{\mathbf{x}^m_{\boldsymbol{\alpha}}\}_{|\boldsymbol{\alpha}|=m}$ be distinct nodes contained in the d-simplex S_d. Then*

$$V^{d,m,n} E^{d,n_0,n} = V^{d,m,n_0}. \tag{4.41}$$

Proof. Given a polynomial $p \in \operatorname{span}\{B_{\boldsymbol{\beta}}^{n_0}\}_{|\boldsymbol{\beta}|=n_0}$, the matrix V^{d,m,n_0} evaluates p at each of the nodes $\mathbf{x}_{\boldsymbol{\alpha}}^m$. On the other hand, the matrix $V^{d,m,n}E^{d,n_0,n}$ evaluates the degree n representation of p at each of the nodes $\mathbf{x}_{\boldsymbol{\alpha}}^m$. Since we are evaluating at the same nodes and the polynomial has not been modified, both evaluations must give the same result. □

The partial ordering (4.38) implies a natural way to order the entries of $V^{d,m,n}$ and also imposes a block structure on $V^{d,m,n}$ by dividing the matrix into sections where $\boldsymbol{\alpha}_0$ and $\boldsymbol{\beta}_0$ are constant. For integers $0 \le \boldsymbol{\alpha}_0 \le m$ and $0 \le \boldsymbol{\beta}_0 \le n$, let $V_{\boldsymbol{\alpha}_0\boldsymbol{\beta}_0}^{d,m,n}$ denote the $\binom{m-\boldsymbol{\alpha}_0+d}{d} \times \binom{n-\boldsymbol{\beta}_0+d}{d}$ submatrix of $V^{d,m,n}$ whose entries satisfy

$$\left(V_{\boldsymbol{\alpha}_0\boldsymbol{\beta}_0}^{d,m,n}\right)_{\boldsymbol{\alpha}'\boldsymbol{\beta}'} = V_{(\boldsymbol{\alpha}_0 \vdash \boldsymbol{\alpha}')(\boldsymbol{\beta}_0 \vdash \boldsymbol{\beta}')}^{d,m,n}. \tag{4.42}$$

4.4.2 Dimension Reduction

We now consider the case of equispaced nodes on the d-simplex S^d. For this case, the entries of the Bernstein–Vandermonde matrix are given by

$$V_{\boldsymbol{\alpha}\boldsymbol{\beta}}^{d,m,n} = \frac{n!}{\boldsymbol{\beta}!}\frac{1}{m^n}\prod_{i=0}^{d}\boldsymbol{\alpha}_i^{\boldsymbol{\beta}_i}. \tag{4.43}$$

This representation of the entries allows us to represent the submatrix $V_{\boldsymbol{\alpha}_0\boldsymbol{\beta}_0}^{d,m,n}$ in terms of a lower-dimensional Bernstein–Vandermonde matrix.

Theorem 4.13. *Let $d > 1$ and $m, n \ge 0$ be integers, and let $\boldsymbol{\alpha}_0, \boldsymbol{\beta}_0$ be integers with $0 \le \boldsymbol{\alpha}_0 \le m$ and $0 \le \boldsymbol{\beta}_0 \le n$. Then*

$$V_{\boldsymbol{\alpha}_0\boldsymbol{\beta}_0}^{d,m,n} = m_{\boldsymbol{\alpha}_0\boldsymbol{\beta}_0}V^{d-1,m-\boldsymbol{\alpha}_0,n-\boldsymbol{\beta}_0}, \tag{4.44}$$

where

$$m_{\boldsymbol{\alpha}_0\boldsymbol{\beta}_0} = \binom{n}{\boldsymbol{\beta}_0}(\boldsymbol{\alpha}_0/m)^{\boldsymbol{\beta}_0}(1-\boldsymbol{\alpha}_0/m)^{n-\boldsymbol{\beta}_0}. \tag{4.45}$$

Proof. If $\boldsymbol{\alpha}_0 = m$ and $\boldsymbol{\beta}_0 \neq n$, then both sides equal zero and we are done. Otherwise, by (4.43), we have that

$$V^{d-1,m-\boldsymbol{\alpha}_0,n-\boldsymbol{\beta}_0}_{\boldsymbol{\alpha}'\boldsymbol{\beta}'} = \frac{(n-\boldsymbol{\beta}_0)!}{\boldsymbol{\beta}'!}\frac{1}{(m-\boldsymbol{\alpha}_0)^{n-\boldsymbol{\beta}_0}}\prod_{i=1}^{d}\boldsymbol{\alpha}_i^{\boldsymbol{\beta}_i}. \tag{4.46}$$

On the other hand,

$$\begin{aligned}\left(V^{d,m,n}_{\boldsymbol{\alpha}_0\boldsymbol{\beta}_0}\right)_{\boldsymbol{\alpha}'\boldsymbol{\beta}'} &= V^{d,m,n}_{\boldsymbol{\alpha}\boldsymbol{\beta}} \\ &= \frac{n!}{\boldsymbol{\beta}!}\frac{1}{m^n}\prod_{i=0}^{d}\boldsymbol{\alpha}_i^{\boldsymbol{\beta}_i} \\ &= \frac{n!}{\boldsymbol{\beta}_0!(n-\boldsymbol{\beta}_0)!}\frac{(n-\boldsymbol{\beta}_0)!}{\boldsymbol{\beta}'!}\frac{\boldsymbol{\alpha}_0^{\boldsymbol{\beta}_0}}{m^{\boldsymbol{\beta}_0}}\frac{(m-\boldsymbol{\alpha}_0)^{n-\boldsymbol{\beta}_0}}{m^{n-\boldsymbol{\beta}_0}}\frac{1}{(m-\boldsymbol{\alpha}_0)^{n-\boldsymbol{\beta}_0}}\prod_{i=1}^{d}\boldsymbol{\alpha}_i^{\boldsymbol{\beta}_i},\end{aligned}$$

where we have separated terms containing $\boldsymbol{\alpha}_0$ and $\boldsymbol{\beta}_0$ and multiplied and divided by $(n-\boldsymbol{\beta}_0)!$ and $(m-\boldsymbol{\alpha}_0)^{n-\boldsymbol{\beta}_0}$. Since

$$\frac{n!}{\boldsymbol{\beta}_0!(n-\boldsymbol{\beta}_0)!}\frac{\boldsymbol{\alpha}_0^{\boldsymbol{\beta}_0}}{m^{\boldsymbol{\beta}_0}}\frac{(m-\alpha_0)^{n-\boldsymbol{\beta}_0}}{m^{n-\boldsymbol{\beta}_0}} = \binom{n}{\boldsymbol{\beta}_0}(\boldsymbol{\alpha}_0/m)^{\boldsymbol{\beta}_0}(1-\boldsymbol{\alpha}_0/m)^{n-\boldsymbol{\beta}_0}, \tag{4.47}$$

we have the desired result. □

The fact that the blocks are multiples of lower-dimensional matrices is analogous to what has been observed for other matrices related to Bernstein polynomials. For example, the Bernstein–Vandermonde matrix associated with the zeroes of Legendre polynomials has a similar structure [39], and so does the Bernstein mass matrix [36].

We recognize that $m_{\boldsymbol{\alpha}_0\boldsymbol{\beta}_0}$ describes a univariate Bernstein polynomial of degree n being evaluated at equispaced points. Therefore, if V^n is the one-dimensional

Bernstein–Vandermonde matrix associated with equispaced nodes as described in Section 4.1.3, then we have that $V^{d,n,n}$ admits the block structure

$$V^{d,n,n} = \begin{pmatrix} V_{00}^n V^{d-1,n,n} & V_{01}^n V^{d-1,n,n-1} & \cdots & V_{0n}^n V^{d-1,n,0} \\ V_{10}^n V^{d-1,n-1,n} & V_{11}^n V^{d-1,n-1,n-1} & \cdots & V_{1n}^n V^{d-1,n-1,0} \\ \vdots & \vdots & \ddots & \vdots \\ V_{n0}^n V^{d-1,0,n} & V_{n1}^n V^{d-1,0,n-1} & \cdots & V_{nn}^n V^{d-1,0,0} \end{pmatrix}. \tag{4.48}$$

In [47], Marco, Martínez, and Viaña considered the bivariate interpolation problem with Lagrange-type data. They showed that when using the bivariate tensor product Bernstein basis, the corresponding coefficient matrix can be expressed as the Kronecker product of univariate Bernstein–Vandermonde matrices. While the case of equispaced nodes on the d-simplex does not fit in this framework, it is interesting to observe the similarity in the structure of (4.48) and the Kronecker product given in [47].

We can use Theorem 4.12 and the block structure given in (4.48) to perform block Gaussian elimination on $V^{d,n,n}$. This method was used in [36] to obtain the block LU decomposition of the Bernstein mass matrix. In the same way, we have the following:

Theorem 4.14. *Let V^n be the one-dimensional Bernstein–Vandermonde matrix associated with equispaced nodes as described in Section 4.1.3. Suppose $V^n = R^n U^n$ is the LU decomposition of V^n. Then*

$$V^{d,n,n} = R^{d,n} U^{d,n}, \tag{4.49}$$

where $R^{d,n}$ is the block lower triangular matrix with blocks given by

$$R^{d,n}_{\boldsymbol{\alpha}_0\boldsymbol{\beta}_0} = R^{n}_{\boldsymbol{\alpha}_0\boldsymbol{\beta}_0} V^{d-1,n-\boldsymbol{\alpha}_0,n-\boldsymbol{\beta}_0} \tag{4.50}$$

and $U^{d,n}$ is the block upper triangular matrix with blocks given by

$$U^{d,n}_{\boldsymbol{\alpha}_0\boldsymbol{\beta}_0} = U^{n}_{\boldsymbol{\alpha}_0\boldsymbol{\beta}_0} E^{d-1,n-\boldsymbol{\beta}_0,n-\boldsymbol{\alpha}_0}. \tag{4.51}$$

Proof. By Theorem 4.12, if $0 \leq \boldsymbol{\alpha}_0, \boldsymbol{\beta}_0 \leq n$ and $\boldsymbol{\gamma}_0 \leq \boldsymbol{\beta}_0$, then

$$\begin{aligned} R^{d,n}_{\boldsymbol{\alpha}_0\boldsymbol{\gamma}_0} U^{d,n}_{\boldsymbol{\gamma}_0\boldsymbol{\beta}_0} &= R^{n}_{\boldsymbol{\alpha}_0\boldsymbol{\gamma}_0} U^{n}_{\boldsymbol{\gamma}_0\boldsymbol{\beta}_0} V^{d-1,n-\boldsymbol{\alpha}_0,n-\boldsymbol{\gamma}_0} E^{d-1,n-\boldsymbol{\beta}_0,n-\boldsymbol{\gamma}_0} \\ &= R^{n}_{\boldsymbol{\alpha}_0\boldsymbol{\gamma}_0} U^{n}_{\boldsymbol{\gamma}_0\boldsymbol{\beta}_0} V^{d-1,n-\boldsymbol{\alpha}_0,n-\boldsymbol{\beta}_0}. \end{aligned}$$

Therefore,

$$\begin{aligned} \left(R^{d,n}U^{d,n}\right)_{\boldsymbol{\alpha}_0\boldsymbol{\beta}_0} &= \sum_{\boldsymbol{\gamma}_0=0}^{\boldsymbol{\beta}_0} R^{d,n}_{\boldsymbol{\alpha}_0\boldsymbol{\gamma}_0} U^{d,n}_{\boldsymbol{\gamma}_0\boldsymbol{\beta}_0} \\ &= \left(\sum_{\boldsymbol{\gamma}_0=0}^{\boldsymbol{\beta}_0} R^{n}_{\boldsymbol{\alpha}_0\boldsymbol{\gamma}_0} U^{n}_{\boldsymbol{\gamma}_0\boldsymbol{\beta}_0} \right) V^{d-1,n-\boldsymbol{\alpha}_0,n-\boldsymbol{\beta}_0} \\ &= V^{n}_{\boldsymbol{\alpha}_0\boldsymbol{\beta}_0} V^{d-1,n-\boldsymbol{\alpha}_0,n-\boldsymbol{\beta}_0} \\ &= V^{d,n,n}_{\boldsymbol{\alpha}_0\boldsymbol{\beta}_0}. \end{aligned}$$

□

4.5 Numerical Results

Now, we consider the accuracy of the methods described in Sections 4.2 and 4.4.2 on several problems. For all of the problems, we chose random solution vectors, computed the right-hand side by matrix multiplication, and then attempted to recover the solution. The numerical results are run in double precision arithmetic on a 2014

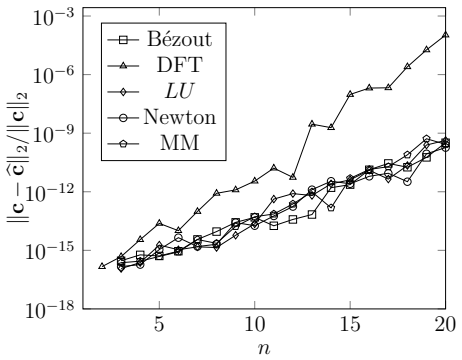
Macbook Air running macOS 10.13 and using Python 2.7.13. LU factorization and FFTs are performed using the `numpy` (v1.12.0) function calls. Also, because our code is a mix of pure Python and low-level compiled libraries, timings are not terribly informative. Consequently, we focus on assessing the stability and accuracy of our methods. If future work leads to more stable fast algorithms, greater care will be afforded to tuning our implementations for performance.

In Fig. 4.2, we consider the case of equispaced nodes. When considering the Euclidean norm of the error (Fig. 4.2a), the LU, Newton, MM, and Bézout algorithms have the best perfomance, followed by the DFT-based application. This is similar to what was observed in Section 3.2, where a similar DFT-based algorithm quickly became unstable. The same behavior can be observed when comparing the Euclidean norm of the residual (Fig. 4.2c), although some separation does occur between the methods. Since the solution vector can be viewed as the Bernstein coefficients of the interpolation polynomial, we also measure the L^2 difference between the exact and computed solutions (Fig. 4.2b); equivalently, we measure the relative M^n error, where M^n is the Bernstein mass matrix given in (2.16). All four solution methods have very similar behavior in the M^n norm as they do when considering the Euclidean norm of the residual.

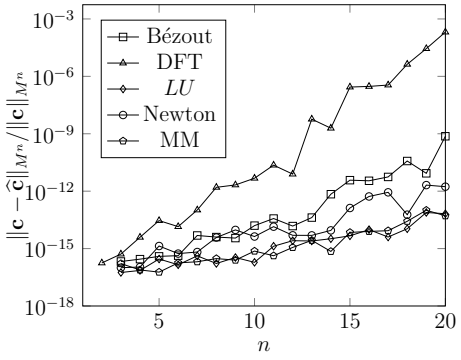
To ensure that the behavior of the solution methods does not depend on the choice of nodes, we also considered the case where the nodes $\mathbf{x}_j$ are randomly selected from $[j/(n+1), (j+1)/(n+1))$ for each $0 \leq j \leq n$ (Fig. 3.5); however, there was no significant difference in the quantities measured between this case and the equispaced case. In addition, we consider a case where the values of the right hand side are all

nonnegative (Fig. 4.4). We see some slight improvement for the DFT and Bézout methods when considering the error in the Euclidean and M^n norms, but overall, the methods have comparable performance to the previous case.

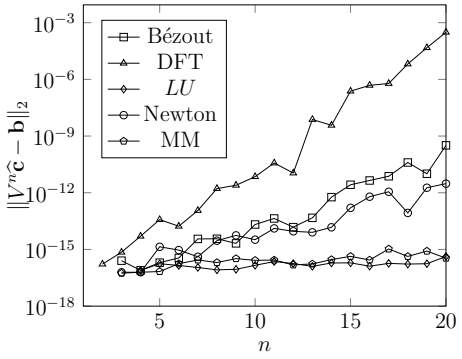
We also used the block LU decomposition given in Theorem 4.14 combined with the one-dimensional LU algorithm to solve the interpolation problem for equispaced nodes on the d-simplex for $d = 2$ and $d = 3$ (Fig. 4.5). Due to the recursive nature of the algorithm, the quantities measured are very similar to the ones observed for equispaced nodes.



(a) Error in the 2-norm.

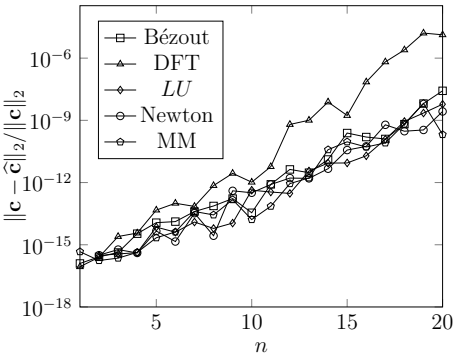


(b) Error in the M^n norm.

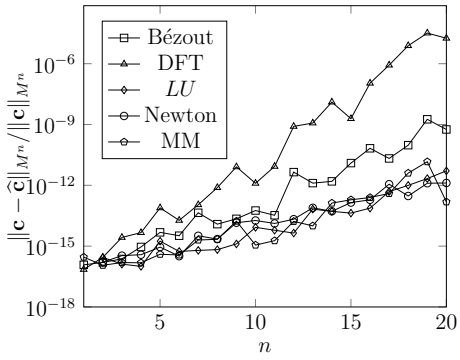


(c) Residual in the 2-norm.

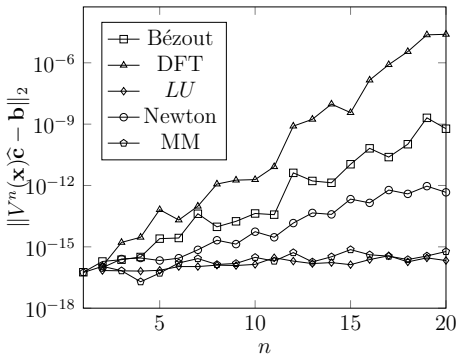
Figure 4.2: Error/residual in using the methods described in Section 4.2 to solve $V^n\mathbf{c} = \mathbf{b}$ for $1 \leq n \leq 20$, where V^n is the matrix described in Section 4.1.3 and $\mathbf{b}$ is a random vector in $[-1, 1]^{n+1}$. Bézout refers to Corollary 4.2, DFT refers to Corollary 4.8, *LU* refers to *LU* decomposition of V^n, Newton refers to the Ainsworth–Sanchez algorithm, and MM refers to the Marco–Martínez algorithm. We use $\widehat{\mathbf{c}}$ to denote the computed solution.



(a) Error in the 2-norm.

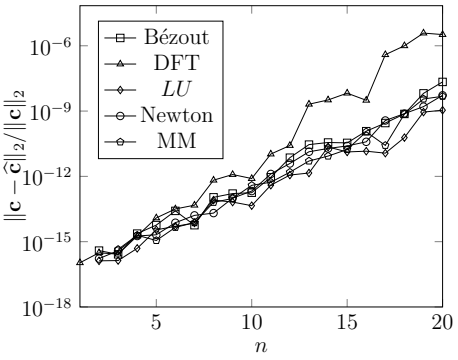


(b) Error in the M^n norm.

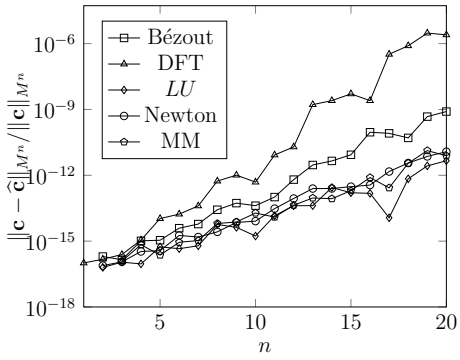


(c) Residual in the 2-norm.

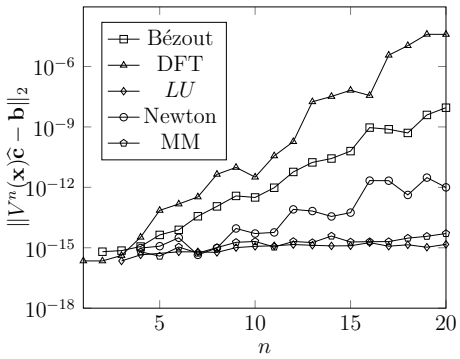
Figure 4.3: Error/residual in using the methods described in Section 4.2 to solve $V^n(\mathbf{x})\mathbf{c} = \mathbf{b}$ for $1 \leq n \leq 20$, where $V^n(\mathbf{x})$ is the Bernstein–Vandermonde matrix associated to $\mathbf{x}$, the nodes $\mathbf{x}_j$ are randomly selected from $[j/(n+1), (j+1)/(n+1))$ for $0 \leq j \leq n$, and $\mathbf{b}$ is a random vector in $[-1, 1]^{n+1}$. Bézout refers to Corollary 4.2, DFT refers to Theorem 4.7, *LU* refers to *LU* decomposition of V^n, Newton refers to the Ainsworth–Sanchez algorithm, and MM refers to the Marco–Martínez algorithm. We use $\widehat{\mathbf{c}}$ to denote the computed solution.



(a) Error in the 2-norm.

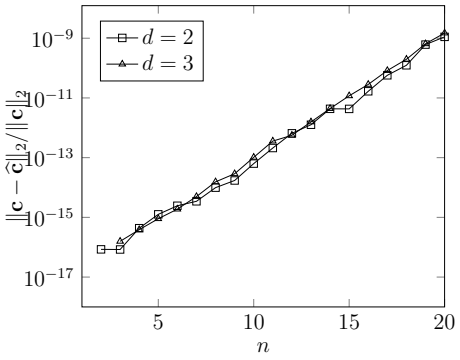


(b) Error in the M^n norm.

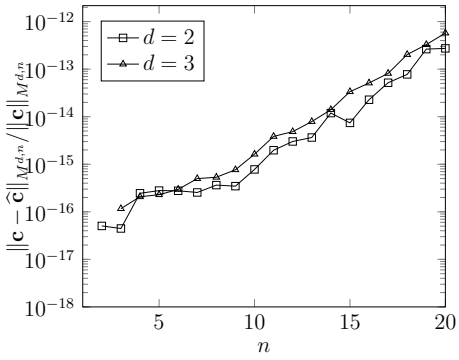


(c) Residual in the 2-norm.

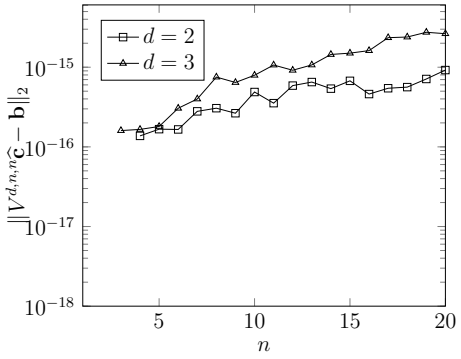
Figure 4.4: Error/residual in using the methods described in Section 4.2 to solve $V^n(\mathbf{x})\mathbf{c} = \mathbf{b}$ for $1 \leq n \leq 20$, where $V^n(\mathbf{x})$ is the Bernstein–Vandermonde matrix associated to $\mathbf{x}$, the nodes $\mathbf{x}_j$ are randomly selected from $[j/(n+1), (j+1)/(n+1))$ for $0 \leq j \leq n$, and $\mathbf{b}$ is a random vector in $[0, 4]^{n+1}$. Bézout refers to Corollary 4.2, DFT refers to Theorem 4.7, *LU* refers to *LU* decomposition of V^n, Newton refers to the Ainsworth–Sanchez algorithm, and MM refers to the Marco–Martínez algorithm. We use $\widehat{\mathbf{c}}$ to denote the computed solution.



(a) Error in the 2-norm.



(b) Error in the $M^{d,n}$ norm.



(c) Residual in the 2-norm.

Figure 4.5: Error/residual in using the block LU decomposition given in Theorem 4.14 to solve $V^{d,n,n}\mathbf{c} = \mathbf{b}$ for $1 \leq n \leq 20$ and $d = 2, 3$, where $V^{d,n,n}$ is the Bernstein–Vandermonde matrix given in (4.43) and $\mathbf{b}$ is a random vector in $[-1, 1]^{\binom{n+d}{d}}$. We use $\widehat{\mathbf{c}}$ to denote the computed solution.

CHAPTER FIVE

Constrained Approximation

5.1 Existence/Uniqueness of Solutions

We begin by showing that certain optimization problem over $\mathcal{P}^{m,+}$ and $\mathcal{P}^{m,n}$ have unique solutions. Since $\mathcal{P}^m$ is a Hilbert space, it suffices to show that $\mathcal{P}^{m,+}$ and $\mathcal{P}^{m,n}$ are closed, convex subsets of $\mathcal{P}^m$ [16].

Proposition 5.1. *$\mathcal{P}^{m,+}$ is a closed, convex subset of $\mathcal{P}^m$.*

Proof. If $p(x), q(x) \geq 0$ for all $x \in [0,1]$ and $\nu \in [0,1]$, then

$$\nu p(x) + (1-\nu)q(x) \geq 0 \quad \text{for all} \quad x \in [0,1], \tag{5.1}$$

and so $\mathcal{P}^{m,+}$ is convex.

Suppose $\{p_k\}_{k=0}^{\infty}$ is a sequence in $\mathcal{P}^{m,+}$ converging (in the L^2 norm) to a polynomial $p \in \mathcal{P}^m$. By (2.23), $\{\mathbf{\Theta}(p_k)\}_{k=0}^{\infty}$ converges to $\mathbf{\Theta}(p)$ in the D^n norm. Since all finite-dimensional norms are equivalent, the sequence $\{\mathbf{\Theta}(p_k)\}_{k=0}^{\infty}$ converges to $\mathbf{\Theta}(p)$ in the ℓ^1 norm. Recall that $|L^j(x)| \leq 1$ for all $x \in [0,1]$, and so

$$|p_k(x) - p(x)| \leq \|\mathbf{\Theta}(p_k) - \mathbf{\Theta}(p)\|_1 \to 0 \quad \text{as} \quad k \to \infty \tag{5.2}$$

for each $x \in [0,1]$. Therefore, for each $x \in [0,1]$, we have that $\{p_k(x)\}_{k=0}^{\infty}$ is a sequence of nonnegative numbers converging to $p(x)$. This implies that $p \in \mathcal{P}^{m,+}$, and so $\mathcal{P}^{m,+}$ is closed. □

Proposition 5.2. *$\mathcal{P}^{m,n}$ is a closed, convex subset of $\mathcal{P}^m$.*

Proof. If $p, q \in \mathcal{P}^{m,n}$ and $\nu \in [0,1]$, then

$$\begin{aligned}\left(E^{m,n}\left[\nu\mathbf{\Pi}(p) + (1-\nu)\mathbf{\Pi}(q)\right]\right)_i &= \nu\left(E^{m,n}\mathbf{\Pi}(p)\right)_i + (1-\nu)\left(E^{m,n}\mathbf{\Pi}(q)\right)_i \\ &\geq 0,\end{aligned}$$

and so $\mathcal{P}^{m,n}$ is convex.

Suppose $\{p_k\}_{k=0}^{\infty}$ is a sequence in $\mathcal{P}^{m,n}$ converging (in the L^2 norm) to a polynomial $p \in \mathcal{P}^m$. By (2.21), the sequence $\{E^{m,n}\mathbf{\Pi}(p_k)\}_{k=0}^{\infty}$ converges to $E^{m,n}\mathbf{\Pi}(p)$ in the M^n norm. Since $[0,\infty)^{n+1}$ is closed, we have that $E^{m,n}\mathbf{\Pi}(p) \in [0,\infty)^{n+1}$, and so $\mathcal{P}^{m,n}$ is closed. □

Because we are working in a Hilbert space, we can use Hilbert orthogonality to restrict the problem of approximating a function to one of approximating a polynomial. To see this, let $p^* \in \mathcal{P}^m$ be the best unconstrained approximation to some continuous nonnegative f on $[0,1]$. Then $f - p^*$ is orthogonal to $\mathcal{P}^m$. For any $q \in \mathcal{P}^m$, the Pythagorean Theorem lets us write

$$\|f - q\|^2 = \|f - p^*\|^2 + \|p^* - q\|^2. \tag{5.3}$$

Since $\|f - p^*\|^2$ is independent of the polynomial q, we can minimize the functional $\|p^* - q\|^2$ instead. This can be more efficient in practice since via (2.20), it requires only linear algebra to evaluate rather than evaluating/integrating the continuous function f during the optimization process.

This discussion (combined with Propositions 5.1 and 5.2) allows us to state well-posed optimization problems over sets of polynomials:

Theorem 5.3. *For any continuous nonnegative f defined on [0,1] and integer $m \geq 0$, there exists a unique nonnegative $p \in \mathcal{P}^{m,+}$ minimizing $\|f - p\|_{L^2}$.*

For univariate polynomials, membership in $\mathcal{P}^{m,+}$ can be defined in terms of a quadratic cone constraint on the coefficients in the monomial basis, as we describe below. Hence, the approximation problem may be posed as a quadratically constrained quadratic program and attacked via semidefinite programming. While the cone constraint possibly could be generalized to a multivariate setting, it is no longer equivalent to nonnegativity. Anticipating this, (2.5) says that membership in $\mathcal{P}^{m,n}$ is given explicitly by nonnegativity of a collection of linear functions of the Bernstein coefficients, and so the approximation problem over this set is a quadratic program with linear inequality constraints.

Theorem 5.4. *For any continuous nonnegative f defined on [0,1] and integers $m \geq 0$ and $n \geq m$, there exists a unique $p \in \mathcal{P}^{m,n}$ minimizing $\|f - p\|_{L^2}$.*

Enforcing two-sided bounds constraints, such as $f : [0,1] \to [0,1]$, is also well-posed and requires a simple extension. In this case, we seek $p \in \mathcal{P}^m$ such that $p \in \mathcal{P}^{m,+}$ and $1-p \in \mathcal{P}^{m,+}$, or as an approximation, $p \in \mathcal{P}^{m,n}$ and $1-p \in \mathcal{P}^{m,n}$. The resulting sets are still closed, convex subsets of $\mathcal{P}^m$, and so the optimization problems are well-posed.

An important question about these approximation problems is the accuracy of their solutions. By combining Bernstein's Theorem [9] with a result from [20], we can

give an error estimate for the best approximation in $\mathcal{P}^{m,n}$ provided that n is large enough.

Proposition 5.5. *There exists an $N \geq m$ such that for $n \geq N$*

$$\inf_{q \in \mathcal{P}^{m,n}} \|f - q\|_{L^2} \leq 2 \inf_{q \in \mathcal{P}^m} \|f - q\|_\infty. \tag{5.4}$$

Proof. By Bernstein's Theorem, there exists an $N \geq m$ such that the best approximation (in the L^2 norm) of f in $\mathcal{P}^{m,+}$ belongs to $\mathcal{P}^{m,n}$ for all $n \geq N$. This means that if $n \geq N$, then

$$\inf_{q \in \mathcal{P}^{m,n}} \|f - q\|_{L^2} = \inf_{q \in \mathcal{P}^{m,+}} \|f - q\|_{L^2} \leq \inf_{q \in \mathcal{P}^{m,+}} \|f - q\|_\infty. \tag{5.5}$$

The result then follows from Theorem 1.2 in [20]. □

While a method for determining the exact value of N in this theorem is unknown, an upper bound is given in [52]. Currently, an error estimate for degrees of elevation less than N is not known.

Now, we discuss certain linear algebraic structure that will be important in presenting a solution of the constrained optimization problem. Recall that M^n admits the spectral decomposition

$$M^n = Q^n \Lambda^n \left(Q^n\right)^T, \tag{5.6}$$

where

$$\Lambda^n = \operatorname{diag}(\lambda_0^n, \ldots, \lambda_n^n) \tag{5.7}$$

is the diagonal matrix of eigenvalues with

$$\lambda_j^n = \frac{(n!)^2}{(n+j+1)!(n-j)!}, \tag{5.8}$$

and

$$Q^n[:,j] = \sqrt{(2j+1)\lambda_j^n} E^{j,n} \mathbf{\Pi}(L^j) \tag{5.9}$$

is the orthogonal matrix of eigenvectors. Recall that we also showed the matrix Q^n can be constructed explicitly in $\mathcal{O}(n^2)$ operators.

In Section 5.2, we will provide an algorithm for finding the coefficients of the optimal polynomial in $\mathcal{P}^{m,n}$. This algorithm provides the degree n representation of a degree m polynomial q, and so as part of the algorithm, we must convert the degree n representation of q to its degree m representation. If the vector $\mathbf{y}$ contains the degree n representation, then the degree m representation is given by the least squares solution of $E^{m,n}\mathbf{\Pi}(q) = \mathbf{y}$; that is,

$$\mathbf{\Pi}(q) = \left((E^{m,n})^T E^{m,n} \right)^{-1} (E^{m,n})^T \mathbf{y}. \tag{5.10}$$

When $\mathbf{y}$ is in the range of $E^{m,n}$, it is also possible to solve $E^{m,n}\mathbf{x} = \mathbf{y}$ by Gaussian elimination on a rectangular matrix, but the least squares approach is more stable in practice.

The spectral decomposition of $(E^{m,n})^T E^{m,n}$ allows us to evaluate (5.10) efficiently. Since $(E^{m,n})^T E^{m,n}$ has a banded structure, it is more efficient to use Gaussian elimination on the least squares system if $n-m$ is small; however, in the interest of stating an algorithm that easily generalizes to higher dimensions, we choose to look at the spectral decomposition.

As a consequence of (2.24) and (5.9), we can characterize the eigenvalues and eigenvectors of $(E^{m,n})^T E^{m,n}$.

Proposition 5.6.

$$(E^{m,n})^T E^{m,n} = Q^m \Sigma^{m,n} (Q^m)^T, \tag{5.11}$$

where $\Sigma^{m,n} = \operatorname{diag}(\lambda_j^m/\lambda_j^n)_{j=0}^m$ *and* Q^m *is defined in* (5.9).

Proof.

$$\begin{aligned}
(E^{m,n})^T E^{m,n} E^{j,m}\mathbf{\Pi}(L^j) &= (E^{m,n})^T E^{j,n}\mathbf{\Pi}(L^j) \\
&= \frac{1}{\lambda_j^n} (E^{m,n})^T M^n E^{j,n}\mathbf{\Pi}(L^j) \\
&= \frac{1}{\lambda_j^n} (E^{m,n})^T M^n E^{m,n} E^{j,m}\mathbf{\Pi}(L^j) \\
&= \frac{1}{\lambda_j^n} M^m E^{j,m}\mathbf{\Pi}(L^j) \\
&= \frac{\lambda_j^m}{\lambda_j^n} E^{j,m}\mathbf{\Pi}(L^j).
\end{aligned}$$

□

5.2 Constrained Optimization

In this section, we pose optimization problems for finding the Bernstein coefficients of optimal polynomials in $\mathcal{P}^{m,+}$ or $\mathcal{P}^{m,n}$. In the latter case, we give an exact algorithm based on the KKT conditions for the latter case. Although the exponential nature of the algorithm makes it impractical for higher degrees, it provides an exact solution process for finding optimal polynomials that can be used as a comparison for other algorithms and demonstrates the role Bernstein structure could play in developing specialized algorithms.

To find the nonnegative polynomial approximation of some f satisfying the hypothesis of Theorem 5.4, we first compute the optimal unconstrained approximation $p \in \mathcal{P}^m$ with Bernstein coefficients $\mathbf{p} = \mathbf{\Pi}(p)$. Then, following (1.4) and (2.21), we pose the cost functional

$$d_p(\mathbf{q}) = (\mathbf{q} - \mathbf{p})^T M^m (\mathbf{q} - \mathbf{p}). \tag{5.12}$$

We can then pose the approximation problem as a quadratic program with linear inequality constraints:

$$\min_{E^{m,n}\mathbf{q} \geq \mathbf{0}^n} d_p(\mathbf{q}) \tag{5.13}$$

Note that we could also add constraints such as $E^{m,n}\mathbf{q} \leq u\mathbf{1}^n$, where $\mathbf{1}^n$ denotes the vector of ones of length $n+1$, to ensure that that the resulting approximation would satisfy an upper bound of u.

Some use cases (e.g. mass conservation) also require the approximating polynomial to preserve the integral average of the original function. Since

$$\int_0^1 p(x)dx = \frac{1}{m+1} \sum_{i=0}^{m} \mathbf{\Pi}(p)_i, \tag{5.14}$$

we can define $h_p(\mathbf{q}) = \frac{1}{m+1} \sum_{i=0}^{m} (\mathbf{\Pi}(p)_i - \mathbf{q}_i)$ and consider enforcing $h_p(\mathbf{q}) = 0$, in which case we pose the quadratic program with inequality and equality constraints:

$$\min_{\substack{E^{m,n}\mathbf{q} \geq \mathbf{0}^n \\ h_p(\mathbf{q})=0}} d_p(\mathbf{q}) \tag{5.15}$$

We can also pose exact nonnegativity via a cone constraint as per [48, 49], although since we are working with Bernstein polynomials, we must adapt the cone to work on

Bernstein coefficients. Define

$$K^{m,+} = \left\{ \mathbf{q} \in \mathbb{R}^{m+1} \ : \ \sum_{i=0}^{m} \mathbf{q}_i x^i \geq 0 \text{ for all } x \in [0,1] \right\}. \tag{5.16}$$

Let S^m denote the vector space of $m \times m$ symmetric matrices, let $S^{m,+}$ be the intersection of S^m and the positive semidefinite matrices, and let $H^{m,k} \in S^{m+1}$ be the Hankel matrix given by

$$H_{ij}^{m,k} = \begin{cases} 1, & \text{if } i+j=k; \\ \\ 0, & \text{otherwise.} \end{cases} \tag{5.17}$$

Observe that $H^{m,k}$ is the zero matrix if $k < 0$ or $k > 2m$. If $m = 2\ell$ is even, define the operators $\Omega_0 : \mathbb{R}^{m+1} \to S^{\ell+1}$ and $\Omega_1 : \mathbb{R}^{m+1} \to S^{\ell}$ by

$$\Omega_0(\mathbf{q}) = \sum_{k=0}^{2\ell} \mathbf{q}_k H^{\ell,k} \quad \text{and} \quad \Omega_1(\mathbf{q}) = \sum_{k=0}^{2\ell-2} \left[\mathbf{q}_{k+1} - \mathbf{q}_{k+2} \right] H^{\ell-1,k}; \tag{5.18}$$

if $m = 2\ell + 1$ is odd, define the operators $\Omega_0, \Omega_1 : \mathbb{R}^{m+1} \to S^{\ell+1}$ by

$$\Omega_0(\mathbf{q}) = \sum_{k=0}^{2\ell} \mathbf{q}_{k+1} H^{\ell,k} \quad \text{and} \quad \Omega_1(\mathbf{q}) = \sum_{k=0}^{2\ell} \left[\mathbf{q}_k - \mathbf{q}_{k+1} \right] H^{\ell,k}. \tag{5.19}$$

Let Ω_0^* and Ω_1^* denote the adjoint of these operators with respect to the inner product $(A, B) = \operatorname{tr}(AB)$ for symmetric matrices. By [48], we have the following characterization of $K^{m,+}$.

Theorem 5.7 (Nesterov)**.** *If $m = 2\ell$ is even, then*

$$K^{m,+} = \{ \mathbf{q} \in \mathbb{R}^{m+1} \ : \ \mathbf{q} = \Omega_0^*(A) + \Omega_1^*(B) \text{ for some } A \in S^{\ell+1,+}, B \in S^{\ell,+} \}; \tag{5.20}$$

if $m = 2\ell + 1$ is odd, then

$$K^{m,+} = \{ \mathbf{q} \in \mathbb{R}^{m+1} \ : \ \mathbf{q} = \Omega_0^*(A) + \Omega_1^*(B) \text{ for some } A, B \in S^{\ell+1,+} \}. \tag{5.21}$$

A straightforward calculation shows that if $m = 2\ell$ is even, then

$$\Omega_0^*(A)_k = \operatorname{tr}(AH^{\ell,k}) \tag{5.22}$$

and

$$\Omega_1^*(B)_k = \operatorname{tr}(B(H^{\ell-1,k-1} - H^{\ell-1,k-2})). \tag{5.23}$$

Similarly, if $m = 2\ell + 1$ is odd, then

$$\Omega_0^*(A)_k = \operatorname{tr}(AH^{\ell,k-1}) \tag{5.24}$$

and

$$\Omega_1^*(B)_k = \operatorname{tr}(B(H^{\ell,k} - H^{\ell,k-1})). \tag{5.25}$$

Therefore, we have a description of members of $K^{m,+}$, and so we can consider the optimization problem

$$\min_{\mathbf{q} \in K^{m,+}} d_p(T^m \mathbf{q}), \tag{5.26}$$

where T^m denotes the $(m+1) \times (m+1)$ matrix that maps the monomial coefficients of a polynomial $q \in \mathcal{P}^m$ to its corresponding Bernstein coefficients. This change of basis is triangular, but incredibly ill-conditioned. On the other hand, the monomial basis is also bad to work with directly.

Although these constrained quadratic programs may be efficiently solved via interior point methods in existing software, with linear constraints we also give a direct algorithm based on the Karush–Kuhn–Tucker (KKT) conditions [7] for optimal solutions in nonlinear programming. We desire to minimize $d_p : \mathbb{R}^{m+1} \to \mathbb{R}$ subject to linear inequality constraints. We let $g_i(\mathbf{q}) = (E^{m,n}\mathbf{q})_i$ and require $g_i(\mathbf{q}) \geq 0$ for

$0 \leq i \leq n$. The KKT conditions state that there exists $\boldsymbol{\mu} \in \mathbb{R}^{n+1}$ such that the optimum $\mathbf{q}$ satisfies the stationarity conditions

$$\nabla d_p(\mathbf{q}) - \sum_{i=0}^{n} \boldsymbol{\mu}_i \nabla g_i(\mathbf{q}) = \mathbf{0}^m \tag{5.27}$$

together with the complementary slackness conditions

$$\boldsymbol{\mu}_i g_i(\mathbf{q}) = 0 \quad \text{for each} \quad 0 \leq i \leq n \tag{5.28}$$

and the dual feasibility conditions

$$\boldsymbol{\mu}_i \geq 0 \quad \text{for each} \quad 0 \leq i \leq n. \tag{5.29}$$

Such algorithms, although of exponential complexity, do give an exact solution. They provide a baseline to compare with other optimization algorithms and demonstrate how Bernstein structure may be used within the process. In Section 5.2.1, we consider the problem of finding $q \in \mathcal{P}^{m,n}$ that solves the inequality-constrained problem, and extend the algorithm to include an equality constraint in Section 5.2.2, which only requires a slight modification to (5.27).

5.2.1 Finding Optimal Polynomials via KKT Theory

We begin by minimizing d_p subject to $g_i \geq 0$ for each $0 \leq i \leq n$, or equivalently, $E^{m,n}\mathbf{q} \geq \mathbf{0}^n$, componentwise. Since

$$\nabla d_p(\mathbf{q}) = 2M^m(\mathbf{q} - \boldsymbol{\Pi}(p)) \tag{5.30}$$

and

$$\nabla g_i(\mathbf{q}) = \left(E^{m,n}[i,:]\right)^T, \tag{5.31}$$

the KKT conditions translate to finding $\mathbf{q} \in \mathbb{R}^{m+1}$ and $\boldsymbol{\mu} \in \mathbb{R}^{n+1}$ that satisfy

$$2M^m(\mathbf{q} - \mathbf{\Pi}(p)) - \left(E^{m,n}\right)^T \boldsymbol{\mu} = \mathbf{0}^m, \tag{5.32}$$

$$\boldsymbol{\mu}_i \left(E^{m,n}\mathbf{q}\right)_i = 0 \quad \text{for each} \quad 0 \leq i \leq n, \tag{5.33}$$

and

$$\boldsymbol{\mu}_i \geq 0 \quad \text{for each} \quad 0 \leq i \leq n. \tag{5.34}$$

The complementary slackness conditions give us 2^{n+1} cases to consider. For each $J \subseteq \{0, \ldots, n\}$, define I_J^n to be the $(n+1) \times (n+1)$ diagonal matrix satisfying

$$\left(I_J^n\right)_{ii} = \begin{cases} 1, & \text{if } i \in J; \\ \\ 0, & \text{if } i \notin J. \end{cases} \tag{5.35}$$

The complementary slackness conditions can then be expressed in the form

$$I_J^n E^{m,n}\mathbf{q} = \mathbf{0}^n \quad \text{and} \quad I_{J^c}^n \boldsymbol{\mu} = \mathbf{0}^n, \tag{5.36}$$

where J^c is the relative complement of J; that is, $J^c = \{0, \ldots, n\} \setminus J$. Therefore, for each $J \subseteq \{0, \ldots, n\}$, we can consider the block matrix equation

$$\begin{pmatrix} M^m & -\frac{1}{2}\left(E^{m,n}\right)^T \\ \\ I_J^n E^{m,n} & I_{J^c}^n \end{pmatrix} \begin{pmatrix} \mathbf{q} \\ \\ \boldsymbol{\mu} \end{pmatrix} = \begin{pmatrix} M^m \mathbf{\Pi}(p) \\ \\ \mathbf{0}^n \end{pmatrix}. \tag{5.37}$$

Since M^m is invertible, we can use the Schur complement to eliminate the variable $\mathbf{q}$ and obtain the equation

$$\left(I_{J^c}^n + \frac{1}{2} I_J^n E^{m,n} \left(M^m\right)^{-1} \left(E^{m,n}\right)^T\right) \boldsymbol{\mu} = -I_J^n E^{m,n} \mathbf{\Pi}(p). \tag{5.38}$$

By (5.6),

$$E^{m,n}\left(M^m\right)^{-1}\left(E^{m,n}\right)^T = U^{m,n}\left(U^{m,n}\right)^T, \tag{5.39}$$

where $U^{m,n}$ is the $(n+1)\times(m+1)$ matrix with columns given by

$$U^{m,n}[:,j] = \sqrt{2j+1}E^{j,n}\mathbf{\Pi}(L^j). \tag{5.40}$$

By defining $W^{m,n} = \frac{1}{2}U^{m,n}\left(U^{m,n}\right)^T$, we can compactly express (5.38) as

$$\left(I^n_{J^c} + I^n_J W^{m,n}\right)\boldsymbol{\mu} = -I^n_J E^{m,n}\mathbf{\Pi}(p). \tag{5.41}$$

We remark that $W^{m,n}$ is a rank $m+1$ matrix with nonzero eigenvalues given by $\left(2\lambda^n_j\right)^{-1}$ for $0 \le j \le m$.

Equation (5.41) corresponds to the $|J|\times|J|$ system

$$\sum_{j\in J} W^{m,n}_{ij}\boldsymbol{\mu}_j = -(E^{m,n}\mathbf{\Pi}(p))_i \quad \text{for each} \quad i \in J. \tag{5.42}$$

We want to find a set $J \subseteq \{0,\dots,n\}$ for which a solution of (5.42) exists and satisfies the dual feasibility conditions. In such a case, we observe that (5.32) implies that

$$E^{m,n}\mathbf{q} = W^{m,n}\boldsymbol{\mu} + E^{m,n}\mathbf{\Pi}(p), \tag{5.43}$$

and so we can check whether the inequality constraints are satisfied. If both the inequality constraints and the dual feasibility conditions are satisfied, then

$$\begin{aligned}\mathbf{q} &= \left(\left(E^{m,n}\right)^T E^{m,n}\right)^{-1}\left(E^{m,n}\right)^T\left(W^{m,n}\boldsymbol{\mu} + E^{m,n}\mathbf{\Pi}(p)\right)\\ &= U^{m,m}\,\mathrm{diag}(\lambda^n_0,\dots,\lambda^n_m)\left(U^{m,n}\right)^T\left(W^{m,n}\boldsymbol{\mu} + E^{m,n}\mathbf{\Pi}(p)\right),\end{aligned}$$

where the last equality follows from Proposition 5.6.

The previous discussion is summarized in Algorithm 2. We remark that since solutions to the constrained optimization problem are unique, we can terminate the algorithm once the inequality constraints and dual feasibility conditions are satisfied.

Algorithm 2 Minimizes $d_p(\mathbf{q})$ subject to $g_i(\mathbf{q}) \geq 0$ for each $0 \leq i \leq n$.

for $J \subset \{0, \ldots, n\}$ **do**
  for $i \in J$ **do**
    $\mathbf{b}_i \leftarrow -(E^{m,n}\mathbf{\Pi}(p))_i$
    for $j \in J$ **do**
      $A_{ij} \leftarrow W_{ij}^{m,n}$
  if $\operatorname{rank}(A) = |J|$ **then**
    $\mathbf{x} \leftarrow A^{-1}\mathbf{b}$
    if $\mathbf{x} \geq \mathbf{0}^{|J|-1}$ **then**
      for $i \leftarrow 0, n$ **do**
        if $i \in J$ **then**
          $\boldsymbol{\mu}_i \leftarrow \mathbf{x}_i$
        else
          $\boldsymbol{\mu}_i \leftarrow 0$
      $\mathbf{y} \leftarrow W^{m,n}\boldsymbol{\mu} + E^{m,n}\mathbf{\Pi}(p)$
      if $\mathbf{y} \geq \mathbf{0}^n$ **then**
        $\mathbf{q} \leftarrow U^{m,m} \operatorname{diag}(\lambda_0^n, \ldots, \lambda_m^n) \left(U^{m,n}\right)^T \mathbf{y}$
        return $\mathbf{q}$

Since we are iterating over all subsets J of $\{0, \ldots, n\}$, the algorithm has exponential complexity. Each iteration requires $2|J|^3/3$ operations to solve for $\mathbf{x}$, $(n+1)(2n+2m+3)$ operations to form the vector $\mathbf{y}$, and $(n+1)(2m+1)+2(m+1)^2$ operations to form the vector $\mathbf{q}$. The vector $\mathbf{y}$ is only formed if the entries in $\mathbf{x}$ are nonnegative, and the vector $\mathbf{q}$ is only formed on the final iteration. Let N denote the number of times the vector $\mathbf{y}$ is formed. Even though the algorithm usually terminates early, this results in a worst-case operation count of

$$N(n+1)(2n+2m+3) + (n+1)(2m+1) + 2(m+1)^2 + \sum_{J \subset \{0,\ldots,n\}} \frac{2|J|^3}{3}, \quad (5.44)$$

which can be simplified to

$$N(n+1)(2n+2m+3)+(n+1)(2m+1)+2(m+1)^2+\frac{2^{n-1}}{3}(n+1)^2(n+4). \quad (5.45)$$

5.2.2 Enforcing Mass Preservation

Since

$$\nabla h_p(\mathbf{q}) = -\frac{1}{m+1}\mathbf{1}^m, \quad (5.46)$$

we can modify (5.32) and find $\mathbf{q} \in \mathbb{R}^{m+1}$, $\boldsymbol{\mu} \in \mathbb{R}^{n+1}$, and $\nu \in \mathbb{R}$ that satisfy

$$2M^m(\mathbf{q}-\mathbf{\Pi}(p)) - \left(E^{m,n}\right)^T \boldsymbol{\mu} - \frac{\nu}{m+1}\mathbf{1}^m = \mathbf{0}^m \quad (5.47)$$

together with (5.33) and (5.34). Similar to Section 5.2.1, for each $J \subseteq \{0,\ldots,n\}$, we consider the block matrix equation

$$\begin{pmatrix} M^m & -\frac{1}{2}\left(E^{m,n}\right)^T & -\frac{1}{2(m+1)}\mathbf{1}^m \\ I_J^n E^{m,n} & I_{J^c}^n & \mathbf{0}^n \\ \frac{1}{m+1}\left(\mathbf{1}^m\right)^T & \left(\mathbf{0}^n\right)^T & 0 \end{pmatrix} \begin{pmatrix} \mathbf{q} \\ \boldsymbol{\mu} \\ \nu \end{pmatrix} = \begin{pmatrix} M^m\mathbf{\Pi}(p) \\ \mathbf{0}^n \\ \frac{1}{m+1}\left(\mathbf{1}^m\right)^T\mathbf{\Pi}(p) \end{pmatrix}. \quad (5.48)$$

We express this as the augmented matrix

$$\left(\begin{array}{cccc} M^m & -\frac{1}{2}\left(E^{m,n}\right)^T & -\frac{1}{2(m+1)}\mathbf{1}^m & M^m\mathbf{\Pi}(p) \\ I_J^n E^{m,n} & I_{J^c}^n & \mathbf{0}^n & \mathbf{0}^n \\ \frac{1}{m+1}\left(\mathbf{1}^m\right)^T & \left(\mathbf{0}^n\right)^T & 0 & \frac{1}{m+1}\left(\mathbf{1}^m\right)^T\mathbf{\Pi}(p) \end{array}\right). \quad (5.49)$$

Since $\mathbf{1}^m$ is an eigenvector of M^m corresponding to the eigenvalue $\frac{1}{m+1}$ and $E^{m,n}\mathbf{1}^m = \mathbf{1}^n$, we can reduce the above system to

$$\begin{pmatrix} M^m & -\frac{1}{2}\left(E^{m,n}\right)^T & -\frac{1}{2(m+1)}\mathbf{1}^m & M^m\mathbf{\Pi}(p) \\ \mathbf{0}^n\left(\mathbf{0}^m\right)^T & I^n_{J^c} + I^n_J\left(W^{m,n} - \frac{1}{2}\mathbf{1}^n\left(\mathbf{1}^n\right)^T\right) & \mathbf{0}^n & -I^n_J E^{m,n}\mathbf{\Pi}(p) \\ \left(\mathbf{0}^m\right)^T & \left(\mathbf{1}^n\right)^T & 1 & 0 \end{pmatrix}. \quad (5.50)$$

Therefore, we search for $J \subseteq \{0, \ldots, n\}$ for which a solution of

$$\sum_{j\in J}\left(W^{m,n}_{ij} - \frac{1}{2}\right)\boldsymbol{\mu}_j = -(E^{m,n}\mathbf{\Pi}(p))_i \quad \text{for each} \quad i \in J \quad (5.51)$$

exists and satisfies the dual feasibility conditions. In such a case, we have that

$$\nu = -\sum_{i=0}^{n}\boldsymbol{\mu}_i. \quad (5.52)$$

Since (5.47) implies that

$$E^{m,n}\mathbf{q} = W^{m,n}\boldsymbol{\mu} + \frac{\nu}{2}\mathbf{1}^n + E^{m,n}\mathbf{\Pi}(p), \quad (5.53)$$

we can check whether the inequality constraints are satisfied. If both the inequality constraints and dual feasibility conditions are satisfied, then the desired solution is given by

$$\mathbf{q} = U^{m,m}\,\mathrm{diag}(\lambda^n_0, \ldots, \lambda^n_m)\left(U^{m,n}\right)^T\left(W^{m,n}\boldsymbol{\mu} + \frac{\nu}{2}\mathbf{1}^n + E^{m,n}\mathbf{\Pi}(p)\right). \quad (5.54)$$

The previous discussion is summarized in Algorithm 3. Similar to Section 5.2.1, the algorithm terminates once the inequality constraints and dual feasibility conditions are satisfied. We remark that Algorithms 2 and 3 can be combined by introducing a variable δ that is equal to 1 to enforce the equality constraint and 0 otherwise.

We also remark that these algorithms are derived by combining the standard KKT algorithm with Bernstein structure.

Algorithm 3 Minimizes $d_p(\mathbf{q})$ subject to $g_i(\mathbf{q}) \geq 0$ for each $0 \leq i \leq n$ and $h_p(\mathbf{q}) = 0$.

for $J \subset \{0, \ldots, n\}$ **do**
  for $i \in J$ **do**
    $\mathbf{b}_i \leftarrow -(E^{m,n}\mathbf{\Pi}(p))_i$
    for $j \in J$ **do**
      $A_{ij} \leftarrow W_{ij}^{m,n} - 1/2$
  if $\operatorname{rank}(A) = |J|$ **then**
    $\mathbf{x} \leftarrow A^{-1}\mathbf{b}$
    if $\mathbf{x} \geq \mathbf{0}^{|J|-1}$ **then**
      for $i \leftarrow 0, n$ **do**
        if $i \in J$ **then**
          $\boldsymbol{\mu}_i \leftarrow \mathbf{x}_i$
        else
          $\boldsymbol{\mu}_i \leftarrow 0$
      $\nu \leftarrow \sum_{i=0}^{n} \boldsymbol{\mu}_i$
      $\mathbf{y} \leftarrow W^{m,n}\boldsymbol{\mu} - \frac{\nu}{2}\mathbf{1}^n + E^{m,n}\mathbf{\Pi}(p)$
      if $\mathbf{y} \geq \mathbf{0}^n$ **then**
        $\mathbf{q} \leftarrow U^{m,m} \operatorname{diag}(\lambda_0^n, \ldots, \lambda_m^n) \left(U^{m,n}\right)^T \mathbf{y}$
        return $\mathbf{q}$

We summarize the discussions in Sections 5.2.1 and 5.2.2 in the following theorem.

Theorem 5.8. *Given a polynomial $p \in \mathcal{P}^m$, Algorithm 2 exactly computes the Bernstein coefficients of the unique polynomial $q \in \mathcal{P}^{m,n}$ that minimizes the quantity $\|p - q\|_{L^2}$ in at most*

$$M(n+1)(2n+2m+3) + (n+1)(2m+1) + 2(m+1)^2 + \frac{2^{n-1}}{3}(n+1)^2(n+4) \quad (5.55)$$

operations, and Algorithm 3 exactly computes the Bernstein coefficients of the unique polynomial $q \in \mathcal{P}^{m,n}$ that minimizes the quantity $\|p - q\|_{L^2}$ subject to $\int_0^1 p(x)dx = \int_0^1 q(x)dx$ in at most

$$N[2(n+1)(n+m+2)+n] + (n+1)(2m+1) + 2(m+1)^2 + \frac{2^{n-1}}{3}(n+1)^2(n+4) \quad (5.56)$$

operations, where M and N denote the number of times the systems (5.42) *and* (5.51) *have nonnegative solutions, resepctively.*

5.3 Higher Dimensions

Bernstein polynomials extend naturally to give a basis for multivariate polynomials of total degree n. Additionally, Bernstein polynomials form a geometrically decomposed partition of unity on the d-simplex and maintain the convex hull property. Analogs of Propositions 5.1 and 5.2 hold in the multivariate case, and so we can extend our analysis from Section 5.2 to nonnegative polynomials on the d-simplex. In this section, we discuss the generalization of Algorithms 2 and 3 to higher dimensions.

Although the linear inequality constraints for nonnegativity used in (5.13) transfer over naturally to the multivariate case, an exact characterization of nonnegativity via a quadratic or cone constraint is likely not possible. Such quadratic constraints typically imply that a polynomial is a sum-of-squares. This can typically be decided via semidefinite programmign in polynomial time, while the general problem if determining nonnegativity is NP-hard in general [42]. Approximating nonnegative polynomials with sums of squares may be possible, but introduces complications beyond the scope of the relatively simple quadratic programs we consider in this paper.

Similar to Section 4.4.1, for an integer $d \geq 1$, let S_d be the unit right simplex in $\mathbb{R}^d$. Let $\{\mathbf{v}_i\}_{i=0}^d \subset \mathbb{R}^d$ be the vertices of S_d, and let $\{\mathbf{b}_i\}_{i=0}^d$ denote the barycentric

coordinates of S_d. Each $\mathbf{b}_i$ is an affine map from $\mathbb{R}^d$ to $\mathbb{R}$ such that

$$\mathbf{b}_i(\mathbf{v}_j) = \begin{cases} 1, & \text{if } i = j; \\ 0, & \text{if } i \neq j; \end{cases} \tag{5.57}$$

for each vertex $\mathbf{v}_j$. Each $\mathbf{b}_i$ is nonnegative on S_d, and

$$\sum_{i=0}^{d} \mathbf{b}_i = 1. \tag{5.58}$$

A multiindex $\boldsymbol{\alpha}$ of length $d+1$ is a $(d+1)$-tuple of nonnegative integers, written

$$\boldsymbol{\alpha} = (\boldsymbol{\alpha}_0, \ldots, \boldsymbol{\alpha}_d). \tag{5.59}$$

The order of $\boldsymbol{\alpha}$, denoted $|\boldsymbol{\alpha}|$, is given by

$$|\boldsymbol{\alpha}| = \sum_{i=0}^{d} \boldsymbol{\alpha}_i. \tag{5.60}$$

The factorial $\boldsymbol{\alpha}!$ of a multiindex $\boldsymbol{\alpha}$ is defined by

$$\boldsymbol{\alpha}! = \prod_{i=0}^{d} \boldsymbol{\alpha}_i. \tag{5.61}$$

The Bernstein polynomials of degree n on the d-simplex S_d are defined by

$$B^n_{\boldsymbol{\alpha}} = \frac{n!}{\boldsymbol{\alpha}!} \prod_{i=0}^{d} \mathbf{b}_i^{\boldsymbol{\alpha}_i}. \tag{5.62}$$

The complete set of Bernstein polynomials $\{B^n_{\boldsymbol{\alpha}}\}_{|\boldsymbol{\alpha}|=n}$ form a basis for polynomials in d variables of total degree at most n.

If $m \leq n$, then any polynomial expressed in the basis $\{B^m_{\boldsymbol{\alpha}}\}_{|\boldsymbol{\alpha}|=m}$ can also be expressed in the basis $\{B^n_{\boldsymbol{\alpha}}\}_{|\boldsymbol{\alpha}|=n}$. We denote by $E^{d,m,n}$ the $\binom{d+n}{d} \times \binom{d+m}{d}$ matrix that maps the coefficients of the degree m representation to the coefficients of the degree

n representation. The matrix $E^{d,m,n}$ is sparse and can be applied matrix-free [36], if desired.

The Bernstein mass matrix for polynomials in d variables of total degree n is the $\binom{d+n}{d} \times \binom{d+n}{d}$ matrix $M^{d,n}$ whose entries are given by

$$M^{d,n}_{\boldsymbol{\alpha\beta}} = \int_{S^d} B^n_{\boldsymbol{\alpha}}(\mathbf{x}) B^n_{\boldsymbol{\beta}}(\mathbf{x}) d\mathbf{x}. \tag{5.63}$$

It was shown in [37] that the entries can be exactly computed as

$$M^{d,n}_{\boldsymbol{\alpha\beta}} = \frac{(n!)^2(\boldsymbol{\alpha}+\boldsymbol{\beta})!}{\boldsymbol{\alpha}!\boldsymbol{\beta}!(2n+d)!}. \tag{5.64}$$

Let $\mathcal{L}^{d,0}$ denote the space of constant polynomials, and for each integer $j \geq 1$, let $\mathcal{L}^{d,j}$ denote the space of d-variate polynomials that are L^2 orthogonal to all polynomials of degree less than j. In [26], a dimensionally recursive algorithm is given for constructing the Bernstein form of an orthogonal basis for $\mathcal{L}^{d,j}$. Therefore, for each nonnegative integer j, we can form the $\binom{d+j}{d} \times \binom{d+j-1}{d-1}$ matrix $L^{d,j}$ whose columns are the Bernstein coefficients of an orthogonal basis for $\mathcal{L}^{d,j}$. The following characterization of the eigenvalues of $M^{d,n}$ can be found in [36].

Theorem 5.9. *The eigenvalues of $M^{d,n}$ are $\{\lambda^{d,n}_j\}_{j=0}^n$, where*

$$\lambda^{d,n}_j = \frac{(n!)^2}{(n+j+d)!(n-j)!} \tag{5.65}$$

is an eigenvalue of multiplicity $\binom{d+j-1}{d-1}$, and the eigenvectors corresponding to $\lambda^{d,n}_j$ are the columns of $E^{d,j,n}L^{d,j}$.

For each $0 \leq j \leq n$, define $Q^{d,n,j}$ to be the $\binom{d+n}{d} \times \binom{d+j-1}{d-1}$ matrix whose columns are given by

$$Q^{d,n,j}[:,k] = \frac{1}{\|L^{d,j}[:,k]\|_{M^{d,j}}} \left(E^{d,j,n} L^{d,j}\right)[:,k]. \tag{5.66}$$

Then Theorem 5.9 implies that

$$E^{d,m,n} \left(M^{d,m,m}\right)^{-1} \left(E^{d,m,n}\right)^T = U^{d,m,n} \left(U^{d,m,n}\right)^T, \tag{5.67}$$

where $U^{d,m,n}$ is $\binom{d+n}{d} \times \binom{d+m}{d}$ matrix given by

$$U^{d,m,n} = \begin{pmatrix} Q^{d,n,0} & | & Q^{d,n,1} & | & \cdots & | & Q^{d,n,m} \end{pmatrix}. \tag{5.68}$$

We now turn our attention to the constrained optimization problem. Define $W^{d,m,n} = \frac{1}{2} U^{d,m,n} \left(U^{d,m,n}\right)^T$. Set $\delta = 1$ to enforce the equality constraints; otherwise, set $\delta = 0$. Following similar reasoning as Sections 5.2.1 and 5.2.2, we search for a set $J \subset \left\{0, \ldots, \binom{d+n}{d}\right\}$ for which the solution of

$$\sum_{j \in J} \left(W_{ij}^{d,m,n} - \frac{d!\delta}{2}\right) \boldsymbol{\mu}_j = -(E^{d,m,n} \boldsymbol{\Pi}(p))_i \quad \text{for each} \quad i \in J \tag{5.69}$$

satisfies the dual feasibility conditions. In such a case, we have that

$$E^{d,m,n} \mathbf{q} = W^{d,m,n} \boldsymbol{\mu} + \frac{\delta \nu}{2} \mathbf{1}^{d,n} + E^{d,m,n} \boldsymbol{\Pi}(p), \tag{5.70}$$

where

$$\nu = -d! \sum_{|\boldsymbol{\alpha}|=n} \boldsymbol{\mu}_{\boldsymbol{\alpha}}, \tag{5.71}$$

and so we can check whether the inequality constraints are satisfied. If both the inequality constraints and dual feasibility conditions are satisfied, then the desired

solution is given by

$$\mathbf{q} = U^{d,m,m} \operatorname{diag}(\lambda_0^{d,n}, \ldots, \lambda_m^{d,n}) \left(U^{d,m,n}\right)^T \left(W^{d,m,n} \boldsymbol{\mu} + \frac{\delta \nu}{2} \mathbf{1}^{d,n} + E^{d,m,n} \boldsymbol{\Pi}(p) \right), \tag{5.72}$$

where each eigenvalue $\lambda_j^{d,n}$ is repeated according to its multiplicity.

Algorithm 4 Minimizes $d_p(\mathbf{q})$ subject to $g_i(\mathbf{q}) \geq 0$ for each $0 \leq i \leq \binom{d+n}{d}$ and $h_p(\mathbf{q}) = 0$.

for $J \subset \{0, \ldots, \binom{d+n}{d}\}$ **do**
  for $i \in J$ **do**
    $\mathbf{b}_i \leftarrow -(E^{d,m,n} \boldsymbol{\Pi}(p))_i$
    for $j \in J$ **do**
      $A_{ij} \leftarrow W_{ij}^{d,m,n} - \frac{d! \delta}{2}$
  if $\operatorname{rank}(A) = |J|$ **then**
    $\mathbf{x} \leftarrow A^{-1} \mathbf{b}$
    if $\mathbf{x} \geq \mathbf{0}^{|J|-1}$ **then**
      for $i \leftarrow 0, n$ **do**
        if $i \in J$ **then**
          $\boldsymbol{\mu}_i \leftarrow \mathbf{x}_i$
        else
          $\boldsymbol{\mu}_i \leftarrow 0$
      $\nu \leftarrow d! \sum_{|\boldsymbol{\alpha}|=n} \boldsymbol{\mu}_{\boldsymbol{\alpha}}$
      $\mathbf{y} \leftarrow W^{d,m,n} \boldsymbol{\mu} - \frac{\delta \nu}{2} \mathbf{1}^{d,n} + E^{d,m,n} \boldsymbol{\Pi}(p)$
      if $\mathbf{y} \geq \mathbf{0}^{d,n}$ **then**
        $\mathbf{q} \leftarrow U^{d,m,m} \operatorname{diag}(\lambda_0^{d,n}, \ldots, \lambda_m^{d,n}) \left(U^{d,m,n}\right)^T \mathbf{y}$
        return $\mathbf{q}$

5.4 Numerical Results

In this section, we investigate the accuracy of approximating smooth functions $f : [0,1] \to [0,1]$ by polynomials whose Bernstein coefficients are nonnegative. Following the discussion in Chapter 2, we first compute the Bernstein coefficients of the optimal (in the L^2 norm) polynomial p^* in $\mathcal{P}^m$ via

$$\boldsymbol{\Pi}(p^*) = U^{m,m} \left(U^{m,m}\right)^T \mathbf{f}, \tag{5.73}$$

where

$$\mathbf{f}_i = \int_0^1 f(x) B_i^m(x) dx. \tag{5.74}$$

We can then find the best approximation in $\mathcal{P}^{m,+}$ or $\mathcal{P}^{m,n}$ by solving the various quadratic programs posed above. We use the Python package `cvxpy` [22] to solve our quadratic programs. The default solver works well for the linear inequality constraints in (5.13), and we use the interface to the Splitting Conic Solver `SCS` [50, 51] for (5.26). We also compare these results to those obtained with Algorithm 2 to solve the linearly constrained problems. Also, we have implemented the CPCD algorithm for approximation with nonnegative polynomials [12].

A few simpler approximation schemes also provide a baseline for comparison. We compute the error in the unconstrained L^2 best approximation. Then, the *Bernstein polynomial*

$$B_m(f)(x) = \sum_{i=0}^{m} f(i/m) B_i^m(x) \tag{5.75}$$

of a continuous function f converges uniformly to f on $[0, 1]$ as $m \to \infty$, although the rate is at best $\mathcal{O}(m^{-2})$ [41].

Interpolation of a function by continuous piecewise linear polynomials also preserves bounds constraints, and we do so by using the Bernstein control points as interpolation nodes. (Note, interpolation into this space exactly corresponds to the L^2 projection with mass lumping). This technique is used as a stage in subcell limiting methods for conservation laws such as [40].

We apply these methods to four functions:

$$\begin{aligned} f_0(x) &= \frac{1}{2}\left(\sin(2\pi x) + 1\right), \\ f_1(x) &= 0.01 + \frac{x}{x^2+1}, \\ f_2(x) &= \frac{26}{25}\left(\frac{1}{1+25(2x-1)^2}\right) - \frac{1}{26}, \\ f_3(x) &= \frac{\pi}{2} + \tan^{-1}\left(30\left(x - 1/2\right)\right). \end{aligned} \tag{5.76}$$

See Fig. 5.1 for plots of these. While all of the functions are smooth, some of the functions are more difficult to approximate by polynomials; for example, f_2 and f_3 have large derivatives and so require a higher order of approximation to obtain small error than f_0 and f_1. Superimposed with the functions are nonnegative polynomial approximations constructed by various techniques; additionaly, we include the best unconstrained polynomial approximation of each function.

We applied each of the approximation algorithms to these problems, showing the results in Fig. 5.2–Fig. 5.5. In the left subfigure, we show the results of linear approximation schemes – the unconstrained L^2 projection as well as the bounds-respecting Bernstein polynomial and interpolation into P^1. In the right subfigure, we show the nonlinear approximation algorithms – the various quadratic programs and the CPCD approximation.

In light of Proposition 5.5 above and Theorem 1.2 in [20], we might expect the error in the quadratic programs to track the unconstrained L^2 projection up to some constant factor, but this does not seem to be the case in every situation. For each function we approximate, we see that the cone-constrained quadratic program produces an error very close to the best approximation up to degree 5, but the convergence curves

typically flatten off after this. At higher degree, `cvxpy` reports an inaccurate solution is obtained, so ill-conditioning or some other numerical difficulty is preventing us from realizing the full theoretical accuracy.

The zero at $x = \frac{3}{4}$ in the function f_0 presents a major difficulty for our approximating algorithms, as we see in Fig. 5.2. Up to degree 5, the cone-constrained approximation seems to track the L^2 approximation quite closely, but after this, `cvxpy` reports an innaccurate solution. Moreover, the linearly-constrained optimization algorithms and the CPCD algorithm all struggle, not doing appreciably better than the Bernstein polynomial or P^1 interpolation.

The function f_1, with no zeros on the interval, presents less difficult for our numerical methods. Results are shown in Fig. 5.3. The linearly-constrained quadratic programs give error equal to the unconstrained approximation, which indicates that the L^2 projection has positive coefficients. The cone-constrained approximation also matches this until degree 5, after which it becomes inaccurate. For this problem, all of the nonlinear approximation algorithms fare much better than the Bernstein polynomial and P^1 interpolant.

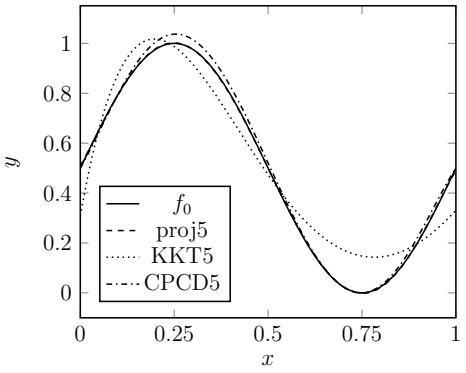
The function f_2 has simple zeros at the endpoints of the interval and has large derivatives within the interval and so is more difficult to approximate than f_1. In this case, we see that the P^1 interpolant roughly tracks the best L^2 approximation. Like out two previous problems, the cone-constrained approximation does nearly as well as the best L^2 approximation, but we lose accuracy as the degree increases. It and the other nonlinear approximations all fare much better than the Bernstein polynomial.

These results are shown in Fig. 5.4, and Fig. 5.5 shows similar performance for the sigmoid function $f_3(x)$.

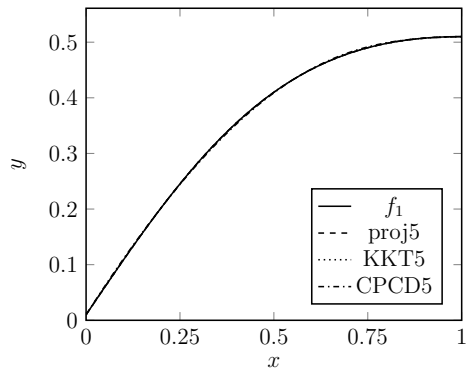
A note comparing the results obtained via quadratic programming to those obtained with the CPCD algorithm [12] is also in order. *A priori*, it is not possible to decide whether CPCD should produce better results than solving (5.13). On one hand, CPCD searches $\mathcal{P}^{m,+}$ rather than $\mathcal{P}^{m,n}$. On the other hand, it lacks a best approximation property within the search space. So, for any given function, it may be possible for either algorithm to outperform the other. We observe that CPCD gives smaller errors for f_0 and some higher degrees of approximation for f_2 and f_3, but larger errors in approximating f_1. We posit that if the best constrained L^2 approximation happens to lie in our search space, then solving (5.13) should win; however, if the best constrained algorithm requires a higher degree of elevation to achieve positive coefficients, then the CPCD algorithm could produce better results.

Finally, we investigate the accuracy of approximating smooth functions $f : S_2 \to [0, 1]$ by polynomials whose Bernstein coefficients are nonnegative. The L^2 norm of the errors in the approximations are shown in Fig. 5.6. We see that using `cvxpy` to solve the linearly-constrained problem (5.13) gives the same results as the KKT-based solver, but (as expected), the convergence is typically slower than for the unconstrained L^2 approximation.

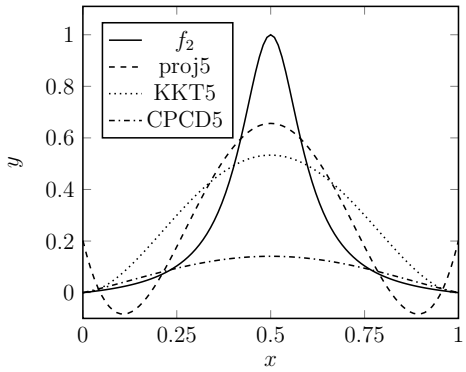
To summarize this discussion, we note that the cone-constrained problem (5.26) seems to consistently deliver very good approximations up to degree 5 and that no positivity-enforcing method consistently performs well at high polynomial degree. On the other hand, the quadratic program with linear constraints generalizes readily



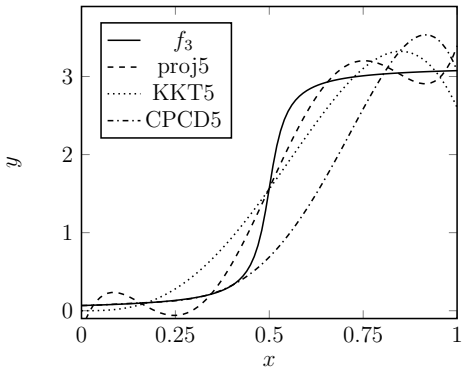
(a) $f_0(x) = \frac{\sin(2\pi x)+1}{2}$



(b) $f_1(x) = 0.01 + x/(1+x^2)$



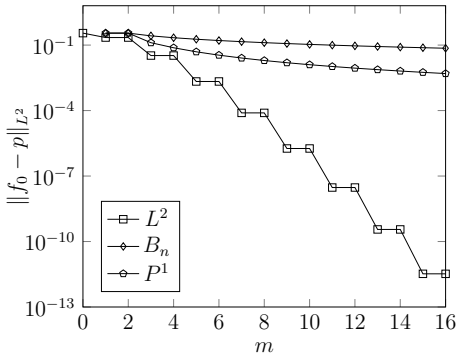
(c) $f_2(x) = \frac{26}{25}\left(\frac{1}{1+25(2x-1)^2} - \frac{1}{26}\right)$



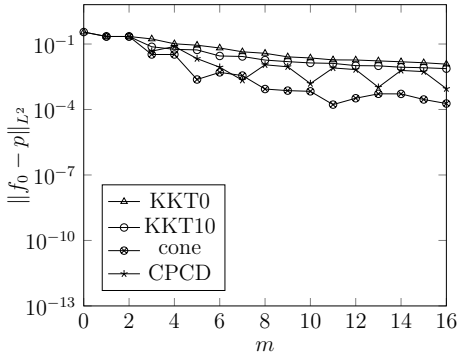
(d) $f_3(x) = \frac{\pi}{2} + \tan^{-1}(30(x-1/2))$

Figure 5.1: Plots of the functions being approximated and their degree 5 polynomial approximations. KKT5 and CPCD5 refer to the finding the degree 5 approximations through the KKT and CPCD algorithms, respectively; proj5 is the unconstrained best degree 5 approximation.

to approximations over the simplex, unlike CPCD or the cone-constrained quadratic program. More work will be needed to improve the numerical accuracy at high degree and to find better ways of enforcing positivity in the multivariate case.

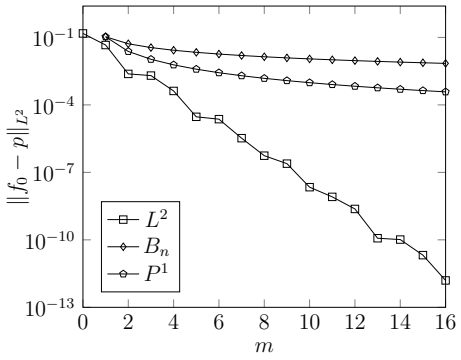


(a) Linear approximations of f_0: unconstrained L^2 projection, Bernstein quasi-interpolation, and interpolation onto P^1 submesh.

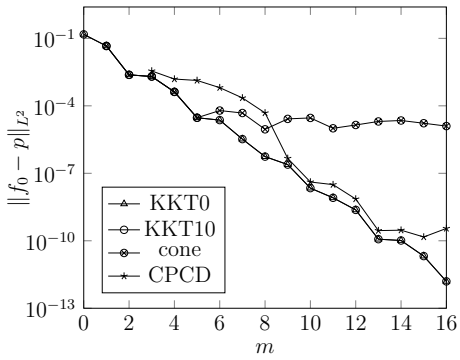


(b) Nonlinear approximations: Solution of (5.13) with 0 and 10 degrees of elevation (KKT0 and KKT10), solution of quadratic program (5.26) (cone), and the CPCD algorithm

Figure 5.2: Error in approximationing $f_0(x) = \frac{\sin(2\pi x)+1}{2}$ using a variety of methods.

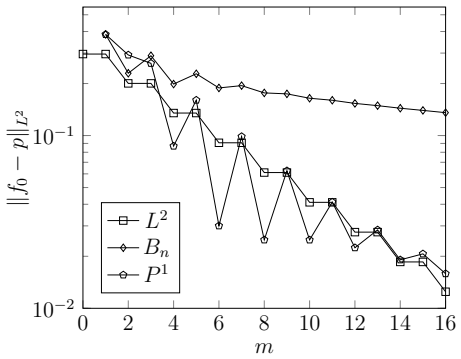


(a) Linear approximations of f_1: unconstrained L^2 projection, Bernstein quasi-interpolation, and interpolation onto P^1 submesh.

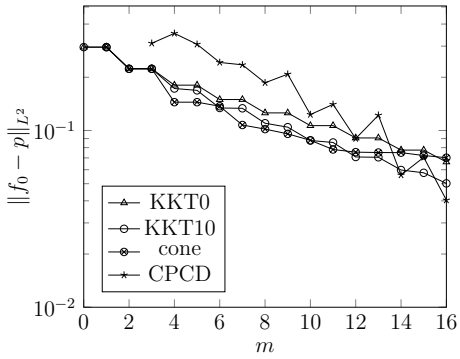


(b) Nonlinear approximations: Solution of (5.13) with 0 and 10 degrees of elevation (KKT0 and KKT10), solution of quadratic program (5.26) (cone), and the CPCD algorithm

Figure 5.3: Error in approximationing $f_1(x) = 0.01 + x/(x^2 + 1)$ using a variety of methods. The Bernstein operator and P^1 interpolant give very modest decrease in the error, while the solution of (5.13) with any elevation actually produces the best approximation. For low polynomial degrees, solving (5.26) also agrees with these, but the solver fails to find an accurate optimal solution after this point.

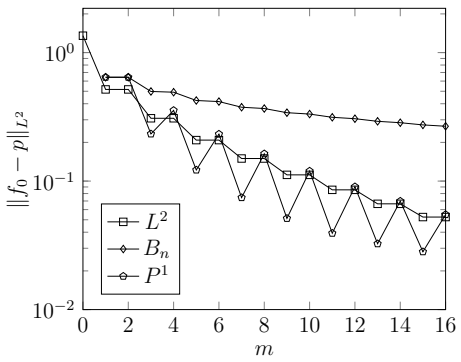


(a) Linear approximations of f_2: unconstrained L^2 projection, Bernstein quasi-interpolation, and interpolation onto P^1 submesh.

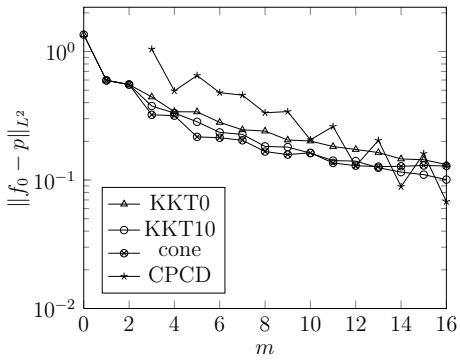


(b) Nonlinear approximations: Solution of (5.13) with 0 and 10 degrees of elevation (KKT0 and KKT10), solution of quadratic program (5.26) (cone), and the CPCD algorithm

Figure 5.4: Error in approximationing $f_2(x) = (26/25)(1/(1 + 25(2x - 1)^2) - 1/26)$ using a variety of methods.

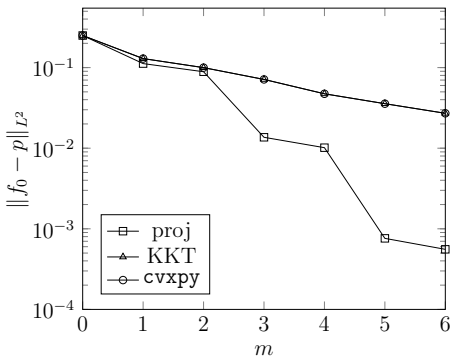


(a) Linear approximations of f_3: unconstrained L^2 projection, Bernstein quasi-interpolation, and interpolation onto P^1 submesh.

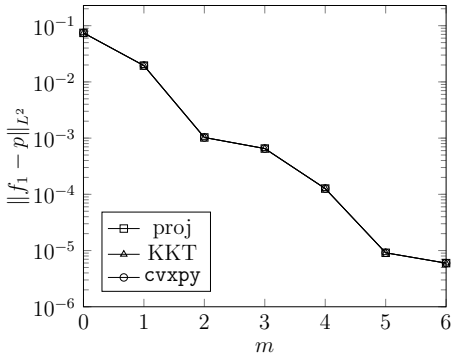


(b) Nonlinear approximations: Solution of (5.13) with 0 and 10 degrees of elevation (KKT0 and KKT10), solution of quadratic program (5.26) (cone), and the CPCD algorithm

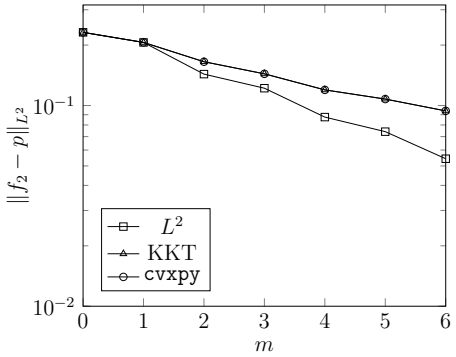
Figure 5.5: Error in approximationing $f_3(x) = \pi/2 + \tan^{-1}(30(x - 1/2))$ using a variety of methods.



(a) $f_0(x, y) = \frac{1-\sin(\pi(x-y))}{2}$



(b) $f_1(x, y) = \frac{1}{100} + \frac{2(x-y+1)}{(x-y+1)^2+4}$



(c) $f_2(x, y) = \frac{26}{25}\left(\frac{1}{1+25(x-y)^2} - \frac{1}{26}\right)$

Figure 5.6: L^2 error in approximating $f_j(x, y)$. We use L^2 to denote the unconstrained projection of f_j into $\mathcal{P}^m$; we use KKT to denote using Algorithm 4 to find the optimal polynomial in $\mathcal{P}^{m,m}$; and we use `cvxpy` to denote using `cvxpy` to find the optimal polynomial in $\mathcal{P}^{m,m}$.

CHAPTER SIX

Conclusions

In this dissertation, we have studied several algorithms for the inversion of the univariate Bernstein mass matrix. Our fast algorithm for inversion based on the spectral decomposition seems very stable in practice and has accuracy comparable to the Cholesky decomposition, while a superfast algorithm based on the discrete Fourier transform is unfortunately unstable.

Additionally, we have studied several algorithms for the inversion of the univariate Bernstein–Vandermonde matrix. These algorithms, while slightly less stable than algorithms discovered previously, provide insight into the structure of the Bernstein–Vandermonde matrix and are remarkably similar to algorithms derived for the Bernstein mass matrix. In addition, we have used a block LU decomposition of the Bernstein–Vandermonde matrix corresponding to equispaced nodes on the d-simplex to give a recursive, block-structured algorithm with comparable accuracy to the one-dimensional algorithm.

Moreover, we have given a new perspective on the conditioning of the Bernstein mass and Bernstein–Vandermonde matrices, indicating that the approximation and interpolation problems are better-conditioned with respect to the L^2 norm than the Euclidean norm.

Finally, we have proposed new techniques to pose bounds-constrained polynomial approximation over the simplex in terms of quadratic programming. Quadratic con-

straints can be used to exactly enforce nonnegativity for univariate polynomials, and the convex hull property of Bernstein polynomials allows us to give explicit sufficient conditions as linear inequality constraints. These techniques perform comparably to an existing method, and they extend naturally to multivariate polynomials on a simplex of any dimension. In the future, we hope to make a more thorough study of approximation properties of these algorithms and how to improve the accuracy of solving the quadratically constrained problem.

BIBLIOGRAPHY

[1] Mark Ainsworth, Gaelle Andriamaro, and Oleg Davydov. "Bernstein–Bézier finite elements of arbitrary order and optimal assembly procedures". In: *SIAM Journal on Scientific Computing* 33.6 (2011), pp. 3087–3109.

[2] Mark Ainsworth, Shuai Jiang, and Manuel A. Sanchéz. "An $\mathcal{O}(p^3)$ *hp*-version FEM in two dimensions: Preconditioning and post-processing". In: *Computer Methods in Applied Mechanics and Engineering* 350 (2019), pp. 766 –802.

[3] Mark Ainsworth and Manuel A. Sánchez. "Computing the Bézier control points of the Lagrangain Interpolant in arbitrary dimension". In: *SIAM Journal on Scientific Computing* 38.3 (2016), A1682–A1700.

[4] Larry Allen and Robert C. Kirby. "Structured inversion of the Bernstein mass matrix". In: *SIAM Journal of Matrix Analysis and Applications* 41.2 (2020), pp. 413–431.

[5] Larry Allen and Robert C. Kirby. "Structured inversion of the Berstein Vandermonde matrix". In: *SIAM Journal of Matrix Analysis and Applications* 42.2 (2021), pp. 557–577.

[6] Refaat El Attar. *Special Functions and Orthogonal Polynomials*. Lulu Press, 2006.

[7] Mokhtar S. Bazaraa, Hanif D. Sherali, and C.M. Shetty. *Nonlinear Programming: Theory and Algorithms*. Wiley, 2013.

[8] Serge Bernstein. "Démonstration du théorème de Weierstrass fondèe sur le calcul des probabilités". In: *Communications de la Société Mathématique de Kharkov* 13.1 (1912), pp. 1–2.

[9] Serge Bernstein. "Sur la représentation des polynômes positifs". In: *Communications de la Société mathématique de Kharkow* 14.2 (1915), pp. 227–228.

[10] Wolfgang Boehm, Marco Paluszny, and Hartmut Prautzsch. *Bézier and B-Spline Techniques*. Springer, 2002.

[11] Richard L. Burden and J. Douglas Faires. *Numerical Methods*. 4th ed. Cengage Learning, 2012.

[12] Martin Campos-Pinto, Frédérique Charles, and Bruno Després. "Algorithms for positive polynomial approximation". In: *SIAM Journal on Numerical Analysis* 57.1 (2019), pp. 148–172.

[13] Martin Campos-Pinto et al. "A projection algorithm on the set of polynomials with two bounds". In: *Numerical Algorithms* 85 (2020), pp. 1475–1498.

[14] E.W. Cheney. *Introduction to Approximation Theory*. 2nd ed. Chelsea, New York: McGraw-Hill, New York, 1982.

[15] Philip J. Davis. *Interpolation and Approximation*. New York: Dover Publications, Inc., 1975.

[16] Lokenath Debnath and Piotr Mikusiński. *Introduction to Hilbert Spaces*. 3rd ed. Elsevier Science, 2005.

[17] Mehdi Dehghan and M.R. Eslahchi. "Best uniform polynomial approximation of some rational functions". In: *Computers and Mathematics with Applications* 59.1 (2010), pp. 382–390.

[18] Mehdi Dehghan and M.R. Eslahchi. "The best uniform polynomial approximation to class of the form $1/(a^2 \pm x^2)$". In: *Nonlinear Analysis: Theory, Methods & Applications* 71.3–4 (2009), pp. 740–750.

[19] Jorge Delgado and J.M. Peña. "Optimal conditioning of Bernstein collocation matrices". In: *SIAM Journal of Matrix Analysis and Applications* 31.3 (2009), pp. 990–996.

[20] Bruno Després. "Polynomials with bounds and numerical approximation". In: *Numerical Algorithms* 76.3 (2017), pp. 829–859.

[21] Frank R. Deutsch. *Best Approximation in Inner Product Spaces*. Springer New York, 2012.

[22] Steven Diamond and Stephen Boyd. "CVXPY: A Python-embedded modeling language for convex optimization". In: *The Journal of Machine Learning Research* 17.1 (2016), pp. 2909–2913.

[23] Michael G Duffy. "Quadrature over a pyramid or cube of integrands with a singularity at a vertex". In: *SIAM journal on Numerical Analysis* 19.6 (1982), pp. 1260–1262.

[24] Gerald Farin. *Curves and Surfaces for CAGD: A Practical Guide*. 5th ed. Academic Press, Inc., 2002.

[25] Rida T. Farouki. "Legendre–Bernstein basis transformations". In: *Journal of Computational and Applied Mathematics* 119.1-2 (2000), pp. 145–160.

[26] Rida T. Farouki, T.N.T. Goodman, and Thomas Sauer. "Construction of orthogonal bases for polynomials in Bernstein form on triangular and simplex domains". In: *Computer Aided Geometric Design* 20.4 (2003), pp. 209–230.

[27] A.D. Forrest. "Interactive interpolation and approximation by Bézier polynomials". In: *The Computer Journal* 15.1 (1972), pp. 71–79.

[28] Mariano Gasca and J. M. Peña. "Total positivity and Neville elimination". In: *Linear Algebra and its Applications* 165 (1992), pp. 25–44.

[29] Walter Gautschi. "The condition of Vandermonde-like matrices involving orthogonal polynomials". In: *Linear Algebra and its Applications* 52/53 (1983), pp. 293–300.

[30] Walter Gautschi, Gene H. Golub, and Gerhard Opfer. *Applications and Computation of Orthogonal Polynomials*. Springer, 1999.

[31] Gene H. Golub and Charles F. Van Loan. *Matrix computations*. Vol. 3. JHU press, 2012.

[32] Hennes Hajduk. "Monolithic convex limiting in discontinuous Galerkin discretizations of hyperbolic conservation laws". In: *Computers & Mathematics with Applications* 87 (2021), pp. 120–138.

[33] Georg Heinig and Karla Rost. *Algebraic methods for Toeplitz-like matrices and operators*. Vol. 13. Birkhäuser, 1984.

[34] Sadegh Jokar and Bahman Mehri. "The best approximation of some rational functions in uniform norm". In: *Applied Numerical Mathematics* 55.2 (2005), pp. 204–214.

[35] Shmuel Kaplan, Alexander Shapiro, and Mina Teicher. *Several Applications of Bézout Matrices*. 2006. eprint: `arXiv:math/0601047`.

[36] Robert C. Kirby. "Fast inversion of the simplicial Bernstein mass matrix". In: *Numerische Mathematik* 135.1 (2017), pp. 73–95.

[37] Robert C. Kirby. "Fast simplicial finite element algorithms using Bernstein polynomials". In: *Numerische Mathematik* 117.4 (2011), pp. 631–652.

[38] Robert C. Kirby. "Low-complexity finite element algorithms for the de Rham complex on simplices". In: *SIAM Journal on Scientific Computing* 36.2 (2014), A846–A868.

[39] Robert C. Kirby and Kieu Tri Thinh. "Fast simplicial quadrature-based finite element operators using Bernstein polynomials". In: *Numerische Mathematik* 121.2 (2012), pp. 261–279.

[40] Dmitri Kuzmin and Manuel Quezada de Luna. "Subcell flux limiting for high-order Bernstein finite element discretizations of scalar hyperbolic conservation laws". In: *Journal of Computational Physics* 411 (2020), p. 109411.

[41] Ming-Jun Lai and Larry L. Schumaker. *Spline functions on triangulations*. Vol. 110. Encyclopedia of Mathematics and its Applications. Cambridge: Cambridge University Press, 2007, pp. xvi+592. ISBN: 978-0-521-87592-9; 0-521-87592-7.

[42] Jean B Lasserre. "A sum of squares approximation of nonnegative polynomials". In: *SIAM review* 49.4 (2007), pp. 651–669.

[43] Richard Leroy. "Ceritificates of positivity in the simplicial Bernstein basis". HAL-00589945.

[44] Stanisław Lewanowicz and Paweł Woźny. "Bézier representation of the constrained dual Bernstein polynomials". In: *Applied Mathematics and Computation* 218.8 (2011), pp. 4580–4586.

[45] D.S. Lubinsky. "Best approximation and interpolation of $1/(1+(ax)^2)$ and its transforms". In: *Journal of Approximation Theory* 125.1 (2003), pp. 106–115.

[46] Ana Marco and José-Javier Martínez. "A fast and accurate algorithm for solving Bernstein–Vandermonde linear systems". In: *Linear Algebra and its Applications* 422.2 (2007), pp. 616–628.

[47] Ana Marco, José-Javier Martínez, and Raquel Viaña. "Accurate polynomial interpolation by using the Bernstein basis". In: *Numerical Algorithms* 75 (2017), pp. 655–674.

[48] Yurii Nesterov. "Squared functional systems and optimization problems". In: *High performance optimization*. Springer, 2000, pp. 405–440.

[49] Jiawang Nie and James W Demmel. "Shape optimization of transfer functions". In: *Multiscale optimization methods and applications*. Springer, 2006, pp. 313–326.

[50] B. O'Donoghue et al. "Conic Optimization via Operator Splitting and Homogeneous Self-Dual Embedding". In: *Journal of Optimization Theory and Applications* 169.3 (2016), pp. 1042–1068. URL: `http://stanford.edu/~boyd/papers/scs.html`.

[51] B. O'Donoghue et al. *SCS: Splitting Conic Solver, version 2.1.4*. `https://github.com/cvxgrp/scs`. Nov. 2019.

[52] Victoria Powers and Bruce Reznick. "Polynomials that are positive on an interval". In: *Transactions of the American Mathematical Society* 352.10 (2000), pp. 4677–4692.

[53] Bruce Arie Reznick. *Sum of even powers of real linear forms*. American Mathematical Soc., 1992.

[54] H.L. Royden. *Real Analysis*. 3rd ed. Prentice-Hall, Inc., 1988.

[55] Walter Rudin. *Principles of Mathematical Analysis*. 3rd ed. McGraw-Hill, 1976.

[56] Larry L. Schumaker. "Computing bivariate splines in scattered data fitting and the finite-element method". In: *Numerical Algorithms* 48 (2008), pp. 237–260.

[57] Gilbert Strang. *Linear algebra and its applications*. Thomson, 2006.

[58] Gabor Szegö. *Orthogonal Polynomials*. American Mathematical Society, 1939.

[59] Eleuterio F. Toro. *Riemann Solvers and Numerical Methods for Fluid Dynamics: A Practical Introduction*. Springer Science & Business Media, 2013.

[60] Lloyd N Trefethen. *Approximation theory and approximation practice*. Vol. 128. SIAM, 2013.

[61] Lieven Vandenberghe and Stephen Boyd. "Semidefinite programming". In: *SIAM review* 38.1 (1996), pp. 49–95.

[62] Curtis R. Vogel. *Computational Methods for Inverse Problems*. SIAM, 2002.