SIAM J. Matrix Anal. Appl.
Vol. 41, No. 2, pp. 413–431

STRUCTURED INVERSION OF THE BERNSTEIN MASS MATRIX[*]

LARRY ALLEN[†] AND ROBERT C. KIRBY[†]

Abstract. Bernstein polynomials, long a staple of approximation theory and computational geometry, have also increasingly become of interest in finite element methods. Many fundamental problems in interpolation and approximation give rise to interesting linear algebra questions. In [13], we gave block-structured algorithms for inverting the Bernstein mass matrix on simplicial cells but did not study fast algorithms for the univariate case. Here, we give several approaches to inverting the univariate mass matrix based on exact formulae for the inverse; decompositions of the inverse in terms of Hankel, Toeplitz, and diagonal matrices; and a spectral decomposition. In particular, the eigendecomposition can be explicitly constructed in $\mathcal{O}(n^2)$ operations, while its accuracy for solving linear systems is comparable to that of the Cholesky decomposition. Moreover, we study conditioning and accuracy of these methods from the standpoint of the effect of roundoff error in the L^2 norm on polynomials, showing that the conditioning in this case is far less extreme than in the standard 2-norm.

Key words. Bernstein polynomials, Legendre polynomials, Bernstein mass matrix, matrix inverse, Bezout matrix, spectral decomposition, fast algorithm, conditioning

AMS subject classifications. 65F05, 65F15, 65F35

DOI. 10.1137/19M1284166

1. Introduction.

Bernstein polynomials, long used in splines, computer-aided geometric design, and computer graphics, were first introduced more than a century ago [5]. More recently, they have also been considered as a tool for high-order approximation of PDEs via the finite element method. Tensorial decomposition of the simplicial Bernstein polynomials under the Duffy transform [6] has led to fast algorithms with comparable complexity to spectral element methods in rectangular geometry [1]. Moreover, Bernstein properties also lead to a special recursive blockwise structure for finite element matrices [13, 14]. Bernstein polynomials are not only a tool for standard C^0 finite element spaces but also can be used to represent bases for polynomial differential forms discretizing the de Rham complex [2, 12].

In this paper, we focus on the inversion of the one-dimensional mass or Gram matrix, which arises in computing L^2 projection and as an important local calculation in many discontinuous Galerkin methods. In [13], we gave recursive, block-structured fast algorithms for mass matrices on the d-simplex. These algorithms rely on the inversion of the one-dimensional mass matrix as an essential building block, but we did not consider fast algorithms or the underlying stability issues for that case. We turn to those issues here. Although the one-dimensional matrices we consider are not (currently) large enough to strictly require in practice, we recall that operations on Chebyshev polynomials have fast stable algorithms at very high orders of approximation [19, 21]. Whether analogous techniques can be found for Bernstein or other polynomial bases seems to be a very open question.

[*] Received by the editors August 28, 2019; accepted for publication (in revised form) by S. P. MacLachlan January 15, 2020; published electronically April 13, 2020.
https://doi.org/10.1137/19M1284166
Funding: The work of the second author was supported by National Science Foundation grant 1525697.

[†] Department of Mathematics, Baylor University, One Bear Place #97328, Waco, TX 76798-7328 (larry_alllen@baylor.edu, robert_kirby@baylor.edu).

In addition to studying the inverse matrix and fast algorithms for applying it, we also give an argument that the effective conditioning for Bernstein polynomials is less extreme than it might appear. This amounts to introducing a nonstandard matrix norm in which to consider the problem. In fact, our main result on conditioning mass matrices—that in the "right" norm the condition number is the square root of the standard matrix 2-norm—is generic and not limited to the particularities of either univariate polynomials or the Bernstein basis. In practice [3, 12], it has been observed that Bernstein polynomials of degree greater than 10 can give highly accurate finite element methods, and we believe this analysis gives a partial explanation of that phenomenon. It does also, however, suggest an eventual limit to the degree of polynomial absent algorithmic advances.

2. Notation and mass matrix. For integers $n \geq 0$ and $0 \leq i \leq n$, we define the Bernstein polynomial $B_i^n(x)$ as

$$B_i^n(x) = \binom{n}{i} x^i (1-x)^{n-i}. \tag{2.1}$$

Each $B_i^n(x)$ is a polynomial of degree n, and the set $\{B_i^n(x)\}_{i=0}^n$ forms a basis for the vector space of polynomials of degree at most n.

The mass (or Gram) matrix arises abstractly in the case of finding the best approximation onto a subspace of a generic Hilbert space. Given a Hilbert space H with inner product $(\cdot,\cdot)$ and a finite-dimensional subspace V^N with basis $\{\phi_i\}_{i=0}^N$, the problem of optimally approximating $u \in H$ by $\sum_{i=0}^N c_i \phi_i$ with respect to the norm of H requires solving the linear system

$$M\mathbf{c} = \mathbf{b}, \tag{2.2}$$

where $M_{ij} = (\phi_i, \phi_j)$ and $\mathbf{b}_i = (u, \phi_i)$.

The Bernstein mass matrix for polynomials of degree n is given by

$$M_{ij}^n = \int_0^1 B_i^n(x) B_j^n(x) dx, \tag{2.3}$$

which can be exactly computed [11] as

$$M_{ij}^n = \binom{n}{i}\binom{n}{j} \frac{(2n-i-j)!(i+j)!}{(2n+1)!}. \tag{2.4}$$

Note we have included a superscript n for the polynomial degree.

While choosing an orthogonal basis rather than Bernstein polynomials trivializes the inversion of M, matrices of this type also arise in other settings that constrain the choice of basis. For example, finite element methods typically require "geometrically decomposed" bases [4] that enable enforcement of continuity across mesh elements. Even when discontinuous bases are admissible for finite element methods, geometric decomposition can be used to simplify boundary operators [13]. Moreover, splines and many geometric applications make use of the Bernstein–Bézier basis as local representations.

If $m \leq n$, then 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 [7] that the entries of $E^{m,n}$ are given by

$$E^{m,n}_{ij} = \frac{\binom{m}{j}\binom{n-m}{i-j}}{\binom{n}{i}}, \tag{2.5}$$

with the standard convention that $\binom{n}{i} = 0$ for $i < 0$ or $i > n$.

We will also need reference to the Legendre polynomials, mapped from their typical home on $[-1, 1]$ to $[0, 1]$. We let $L^n(x)$ denote the Legendre polynomial of degree n over $[0, 1]$, scaled so that $L^n(1) = 1$ and

$$\|L^n\|^2_{L^2} = \frac{1}{2n+1}. \tag{2.6}$$

It was observed in [7] that $L^n(x)$ has the following representation:

$$L^n(x) = \sum_{i=0}^{n} (-1)^{n+i} \binom{n}{i} B^n_i(x). \tag{2.7}$$

The mass matrix (for Bernstein or any other family of polynomials) plays an important role in connecting the L^2 topology on the finite-dimensional space to linear algebra. To see this, we first define mappings connecting polynomials of degree n to $\mathbb{R}^{n+1}$. Given any $\mathbf{c} \in \mathbb{R}^{n+1}$, we let $\pi(\mathbf{c})$ be the polynomial expressed in the Bernstein basis with coefficients contained in $\mathbf{c}$:

$$\pi(\mathbf{c})(x) = \sum_{i=0}^{n} \mathbf{c}_i B^n_i(x). \tag{2.8}$$

We let $\mathbf{\Pi}$ be the inverse of this mapping, sending any polynomial of degree n to the vector of $n + 1$ coefficients with respect to the Bernstein basis.

Now let $p(x)$ and $q(x)$ be polynomials of degree n with expansion coefficients $\mathbf{\Pi}(p) = \mathbf{p}$ and $\mathbf{\Pi}(q) = \mathbf{q}$. Then the L^2 inner product of p and q is given by the M^n-weighted inner product of $\mathbf{p}$ and $\mathbf{q}$ for

$$\int_0^1 p(x)q(x)dx = \sum_{i,j=0}^{n} \mathbf{p}_i \mathbf{q}_j \int_0^1 B^n_i(x)B^n_j(x)dx = \mathbf{p}^T M^n \mathbf{q}. \tag{2.9}$$

Similarly, if

$$\|\mathbf{p}\|_{M^n} = \sqrt{\mathbf{p}^T M^n \mathbf{p}} \tag{2.10}$$

is the M^n-weighted vector norm, then we have, for $p = \pi(\mathbf{p})$,

$$\|p\|_{L^2} = \|\mathbf{p}\|_{M^n}. \tag{2.11}$$

It was shown in [11] that if $m \leq n$, then

$$M^m = \left(E^{m,n}\right)^T M^n E^{m,n}. \tag{2.12}$$

Further, in [11] it was observed that the mass matrix is, up to a row and column scaling, a Hankel matrix (constant along antidiagonals). Let Δ^n be the $(n+1)\times(n+1)$ diagonal matrix with binomial coefficients on the diagonal:

$$\Delta^n_{ij} = \begin{cases} \binom{n}{i}, & i = j; \\ 0, & \text{otherwise}. \end{cases} \tag{2.13}$$

Then let $\widetilde{M}^n$ be defined by

$$\widetilde{M}^n_{ij} = \frac{(2n-i-j)!(i+j)!}{(2n+1)!} \tag{2.14}$$

so that

$$M^n = \Delta^n \widetilde{M}^n \Delta^n. \tag{2.15}$$

Since $\widetilde{M}^n$ depends only on the sum $i+j$ and not i and j separately, it is a Hankel matrix. Therefore, $\widetilde{M}^n$ (and hence M^n) can be applied to a vector in $\mathcal{O}(n \log n)$ operations via a circulant embedding and FFTs [17], although the actual point where this algorithm wins over the standard one may be at a rather high degree.

In [13], we gave a spectral characterization of the mass matrix on simplices of any dimension. The following theorem characterizes the eigenvalues and eigenvectors for the one-dimensional case.

THEOREM 2.1. *The eigenvalues of M^n are $\{\lambda_i^n\}_{i=0}^n$, where*

$$\lambda_i^n = \frac{(n!)^2}{(n+i+1)!(n-i)!} \tag{2.16}$$

and the eigenvector of M^n corresponding to λ_i^n is $E^{i,n}\mathbf{\Pi}(L^i)$.

3. Characterizing the inverse matrix.

In this section, we present several approaches to constructing and applying the inverse of the mass matrix.

3.1. A formula for the inverse.

The following result (actually, a generalization) is derived in [15]:

THEOREM 3.1.

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

This result is obtained by characterizing (possibly constrained) Bernstein dual polynomials via Hahn orthogonal polynomials. Much earlier, Farouki [7] gave a characterization of the standard dual polynomials by more direct means. Lu [16] builds on the work in [15] to give a recursive formula for applying this inverse.

In this section, we summarize a proof of Theorem 3.1 based on Farouki's representation of the dual basis in subsection 3.1.1 and then give an alternative derivation of the result based on Bézoutians in subsection 3.1.2. Since the mass matrix is diagonal in the Legendre basis, we can also characterize the inverse operator by converting from Bernstein to Legendre representation, multiplying by a diagonal, and then converting back. This approach, equivalent to the spectral decomposition, is given in subsection 3.1.3.

3.1.1. Inversion via the dual basis.

The dual basis to $\{B_i^n(x)\}_{i=0}^n$ is the set of polynomials $\{d_i^n(x)\}_{i=0}^n$ satisfying

$$\int_0^1 B_i^n(x) d_j^n(x) dx = \begin{cases} 1, & i=j; \\ 0, & \text{otherwise}. \end{cases} \tag{3.2}$$

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 $\left(M^n\right)_{ij}^{-1} = \mathbf{d}_i^{n,j}$. Following Farouki's representation of the dual basis coefficients [7], this gives us that

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

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 subsection 3.2 to decompose the inverse in terms of diagonal, Hankel, and Toeplitz matrices.

If $u(t) = \sum_{i=0}^{n+1} u_i t^i$ and $v(t) = \sum_{i=0}^{n+1} v_i t^i$ are polynomials of degree at most $n+1$ expressed in the monomial basis, then the Bézout matrix generated by u and v, denoted $\operatorname{Bez}(u, v)$, is the $(n+1) \times (n+1)$ matrix with entries b_{ij} satisfying

$$\frac{u(s)v(t) - u(t)v(s)}{s - t} = \sum_{i,j=0}^{n} b_{ij} s^i t^j. \tag{3.5}$$

By comparing coefficients, we have that a closed-form expression for the entries is given by

$$b_{ij} = \sum_{k=0}^{\min\{i,n-j\}} \left(u_{j+k+1} v_{i-k} - u_{i-k} v_{j+k+1}\right). \tag{3.6}$$

Heinig and Rost [9] gave a formula for the inverse of a Hankel matrix in terms of a Bézout matrix.

THEOREM 3.2. *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*

$$H^{-1} = \frac{1}{v_{n+1}} \operatorname{Bez}(u, v), \tag{3.7}$$

where u is the polynomial whose coefficients are given by the last column of H^{-1} and v is the polynomial whose coefficients 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 u^n and v^{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 $\left(M^n\right)^{-1}$ 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 $\left(M^n\right)^{-1}$. The coefficients of u^n are then obtained via (2.15).

LEMMA 3.3. *The null space of* $\left(E^{n-1,n}\right)^T$ *is spanned by* $\mathbf{\Pi}(L^n)$.

Proof. By (2.5), we have that $(E^{n-1,n})^T$ is upper bidiagonal with nonzero entries on the main diagonal and the superdiagonal, and so the null space of $(E^{n-1,n})^T$ is one-dimensional. Therefore, it is enough to show that $\mathbf{\Pi}(L^n)$ belongs to the null space of $(E^{n-1,n})^T$.

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

$$0 = \int_0^1 p(x)L^n(x)dx. \tag{3.8}$$

Therefore, Theorem 2.1 and (2.9) imply that

$$0 = (E^{n-1,n}\mathbf{p})^T M^n \mathbf{\Pi}(L^n) = \lambda_n^n \mathbf{q}^T\mathbf{\Pi}(L^n) = \frac{(n!)^2}{(2n+1)!}\mathbf{q}^T\mathbf{\Pi}(L^n). \tag{3.9}$$

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 $\left(E^{n-1,n}\right)^T$. □

LEMMA 3.4. *For* $n \geq 0$*, let* $\mathbf{y}^n$ *be given by*

$$\mathbf{y}_i^n = (-1)^{n+i}(n+1)\binom{n+1}{i}. \tag{3.10}$$

Then, for $n \geq 1$*,*

$$\mathbf{y}^n = (2n+1)\mathbf{\Pi}(L^n) + E^{n-1,n}\mathbf{y}^{n-1}. \tag{3.11}$$

Proof. By (2.5) and (2.7), we have that, for each $0 \leq i \leq n$,

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

□

PROPOSITION 3.5. *Let* $\mathbf{y}^n$ *be as in Lemma* 3.4*. Then*

$$M^n\mathbf{y}^n = \mathbf{e}^n, \tag{3.12}$$

where

$$\mathbf{e}_i^n = \begin{cases} 1, & i = n; \\ 0, & \textit{otherwise}. \end{cases} \tag{3.13}$$

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

$$M^n\mathbf{y}^n = (2n+1)\lambda_n^n\mathbf{\Pi}(L^n) + M^nE^{n-1,n}\mathbf{y}^{n-1}. \tag{3.14}$$

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 (2.12) and the induction hypothesis gives us that

$$\left(E^{n-1,n}\right)^T M^n\mathbf{y}^n = \mathbf{e}^{n-1}. \tag{3.15}$$

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, (2.7) and (3.14) 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 (2.7) again, we have that

$$\mathbf{\Pi}(L^n)^T\mathbf{\Pi}(L^n) = \sum_{i=0}^{n}\binom{n}{i}^2 = \binom{2n}{n} = \frac{1}{(2n+1)\lambda_n^n}. \tag{3.16}$$

Therefore,

$$\mathbf{\Pi}(L^n)^T(M^n\mathbf{y}^n - \mathbf{e}^n) = \mathbf{\Pi}(L^n)^TM^nE^{n-1,n}\mathbf{y}^{n-1}. \tag{3.17}$$

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,

$$\mathbf{\Pi}(L^n)^TM^nE^{n-1,n}\mathbf{y}^{n-1} = \lambda_n^n\mathbf{\Pi}(L^n)^TE^{n-1,n}\mathbf{y}^{n-1} = 0, \tag{3.18}$$

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 $\left(E^{n-1,n}\right)^T$. □

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

$$u_i^n = \left(\Delta^n\mathbf{y}^n\right)_i = (-1)^{n+i}(n+1)\binom{n}{i}\binom{n+1}{i}. \tag{3.19}$$

We now find the coefficients of v^{n+1}. By Theorem 3.2, 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 v^{n+1} are given by elevating u^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

$$\mathbf{x}^P = P\begin{pmatrix}\mathbf{x}\\ 0\end{pmatrix}, \tag{3.20}$$

where

$$P = \begin{pmatrix} & & & & 1 \\ & & & 1 & \\ & & \iddots & & \\ & 1 & & & \\ 1 & & & & \end{pmatrix} \tag{3.21}$$

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

Lemma 3.6. *If H is an $(n+1) \times (n+1)$ nonsingular Hankel matrix of the form*

$$H = \begin{pmatrix} h_0 & h_1 & h_2 & \cdots & h_n \\ h_1 & h_2 & h_3 & \cdots & h_{n-1} \\ h_2 & h_3 & h_4 & \cdots & h_{n-2} \\ \vdots & \vdots & \vdots & \iddots & \vdots \\ h_n & h_{n-1} & h_{n-2} & \cdots & h_0 \end{pmatrix} \tag{3.22}$$

and $\mathbf{x}$ satisfies

$$H\mathbf{x} = \mathbf{e}^n \tag{3.23}$$

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

$$\widehat{H}\mathbf{x}^P = \mathbf{e}^{n+1}. \tag{3.24}$$

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

$$\begin{pmatrix} h_{n-1} & h_n & \cdots & h_3 & h_2 & h_1 & h_0 \\ h_{n-2} & h_{n-1} & \cdots & h_4 & h_3 & h_2 & h_1 \\ h_{n-3} & h_{n-2} & \cdots & h_5 & h_4 & h_3 & h_2 \\ h_{n-4} & h_{n-3} & \cdots & h_6 & h_5 & h_4 & h_3 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ \alpha & h_0 & \cdots & h_{n-3} & h_{n-2} & h_{n-1} & h_n \\ \beta & \alpha & \cdots & h_{n-4} & h_{n-3} & h_{n-2} & h_{n-1} \end{pmatrix} \begin{pmatrix} \mathbf{x}_0 \\ \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \\ \vdots \\ \mathbf{x}_n \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix}. \tag{3.25}$$

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

$$\alpha = -\frac{1}{\mathbf{x}_0} \sum_{i=0}^{n-1} h_i \mathbf{x}_{i+1} \tag{3.26}$$

and

$$\beta = \frac{1}{\mathbf{x}_0} \left(1 - \alpha \mathbf{x}_1 - \sum_{i=0}^{n-2} h_i \mathbf{x}_{i+2} \right) \tag{3.27}$$

gives the desired result. ∎

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

$$v_i^{n+1} = \left(\Delta^n \mathbf{y}^n\right)_i^P = (-1)^{i+1}(n+1)\binom{n+1}{i}\binom{n}{i-1}, \tag{3.28}$$

and hence Theorem 3.2 and (3.6) imply the following.

THEOREM 3.7. *The matrix $\widetilde{M}^n$ defined in* (2.14) *has an inverse given by*

$$\begin{aligned}\left(\widetilde{M}^n\right)^{-1}_{ij} &= \frac{1}{v^{n+1}_{n+1}} \sum_{k=0}^{\min\{i,n-j\}} \left(u^n_{j+k+1} v^{n+1}_{i-k} - u^n_{i-k} v^{n+1}_{j+k+1} \right) \\ &= (-1)^{i+j} \sum_k (2k+1-i+j) \binom{n+1}{i-k}^2 \binom{n+1}{j+k+1}^2 .\end{aligned}$$

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

$$M^n = Q^n \Lambda^n (Q^n)^T, \tag{3.29}$$

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 3.8.

$$\left\| E^{i,n} \mathbf{\Pi}(L^i) \right\|_2^2 = \frac{1}{(2i+1)\lambda^n_i}. \tag{3.30}$$

Proof. The result follows from (2.6), (2.10), and Theorem 2.1:

$$\frac{1}{2i+1} = \|L^i\|^2_{L^2} = \|E^{i,n}\mathbf{\Pi}(L^i)\|^2_{M^n} = \lambda^n_i \left\| E^{i,n}\mathbf{\Pi}(L^i) \right\|_2^2 . \tag{3.31}$$ □

Proposition 3.8 implies that

$$Q^n = \left(\sqrt{\lambda^n_0} E^{0,n} \mathbf{\Pi}(L^0) \quad | \quad \cdots \quad | \quad \sqrt{(2n+1)\lambda^n_n} \mathbf{\Pi}(L^n) \right) \tag{3.32}$$

and

$$\Lambda^n = \operatorname{diag}\left(\lambda^n_0, \ldots, \lambda^n_n\right). \tag{3.33}$$

Equation (3.32) characterizes Q^n and suggests an algorithm for its construction—begin with $\mathbf{\Pi}(L^i)$ 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 three-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^i has been constructed (for $1 \leq i < n$), the critical step in the recurrence is to multiply $L^i(x)$ by x. Given any polynomial

$$p(x) = \sum_{i=0}^n p_i B^n_i(x)$$

of degree n, we have that

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

where $\tilde{p}_0 = 0$ and $\tilde{p}_i = \frac{ip_{i-1}}{n+1}$ for $1 \leq i \leq n+1$. This $\mathcal{O}(n)$ process computes $xp(x)$ in the degree $n+1$ basis, but we need $xL^i(x)$ in the Bernstein basis of degree n. That is, if $\tilde{\mathbf{p}}$ holds the degree $n+1$ Bernstein coefficients, we need to find $\mathbf{q}$ such that

$$E^{n,n+1}\mathbf{q} = \tilde{\mathbf{p}}, \tag{3.34}$$

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 3.1 summarizes building the entire matrix Q^n using the three-term recurrence and the tridiagonal degree reduction, where E denotes the matrix $E^{n,n+1}$.

Algorithm 3.1 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 $j = 2$ to $n-1$ do
    $\widetilde{\mathbf{p}} \leftarrow \mathbf{\Pi}(x\pi(Q[:,j-1])(x))$
    $\mathbf{q} \leftarrow \left(E^T E\right)^{-1} E^T \widetilde{\mathbf{p}}$
    $Q^n[:,j] \leftarrow \frac{2j-1}{j}\left(2\mathbf{q} - Q^n[:,j-1]\right) - \frac{j-1}{j} Q^n[:,j-2]$
end for
$Q^n[:,n] \leftarrow \mathbf{\Pi}(L^n(x))$
for $j = 0$ to $n$ do
    $Q^n[:,j] \leftarrow \sqrt{(2j+1)\lambda_j^n} Q^n[:,j]$
end for
```

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

3.2. Decomposing the inverse.

The proof of Theorem 3.7 suggests the following decomposition of $(M^n)^{-1}$ into diagonal, Toeplitz, and Hankel matrices.

THEOREM 3.9. *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

$$(M^n)^{-1} = (\Delta^n)^{-1}\left[\widetilde{T}^n H^n - T^n \widetilde{H}^n\right](\Delta^n)^{-1}. \tag{3.35}$$

Proof. By Theorem 3.7, we have that

$$\begin{aligned}\left(\widetilde{M^n}\right)^{-1}_{ij} &= (-1)^{i+j}\sum_k (2k+1-i+j)\binom{n+1}{i-k}^2\binom{n+1}{j+k+1}^2\\ &= \sum_k\left[(-1)^{i-k}(i-k)\binom{n+1}{i-k}^2\right]\left[(-1)^{j+k+1}\binom{n+1}{j+k+1}^2\right]\\ &\quad -\sum_k\left[(-1)^{i-k}\binom{n+1}{i-k}^2\right]\left[(-1)^{j+k+1}(j+k+1)\binom{n+1}{j+k+1}^2\right].\end{aligned}$$

Therefore, $(\widetilde{M^n})^{-1} = \widetilde{T}^n H^n - T^n\widetilde{H}^n$, and so the result follows from (2.15). □

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

4. Applying the inverse.

Now we describe several approaches to applying M^{-1} to a vector.

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

$$M^n = LL^T, \tag{4.1}$$

where L is lower triangular with positive diagonal entries [20]. Widely available in libraries, computing L 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 start-up phase.

Exact inverse. In light of Theorem 3.1, we can directly form M^{-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 start-up and per-solve complexity as the Cholesky factorization, although the constants are different.

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

$$(M^n)^{-1} = Q^n\left(\Lambda^n\right)^{-1}\left(Q^n\right)^T. \tag{4.2}$$

The inverse can be applied by two (dense) matrix multiplications and a diagonal scaling, requiring $\mathcal{O}(n^2)$ operations. Thanks to Algorithm 3.1, we have only an $\mathcal{O}(n^2)$ start-up phase.

DFT-based application. Theorem 3.9 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. Moreover, the operations can be fused together so that some FFTs are reused and FFT/inverse FFT pairs cancel in (3.35). 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 superfast algorithm. Therefore, we do not go into much detail on this approach.

5. Conditioning and accuracy. As a corollary of Theorem 2.1, M^n is terribly ill conditioned as n increases. For each n, the minimal eigenvalue occurs when $i = n$, which is

$$\lambda_{\min}(M^n) = \frac{(n!)^2}{(2n+1)!}. \tag{5.1}$$

Similarly, the maximal eigenvalue occurs when $i = 0$:

$$\lambda_{\max}(M^n) = \frac{(n!)^2}{(n+1)!n!} = \frac{1}{n+1}. \tag{5.2}$$

Given these extremal eigenvalues, the 2-norm condition number is

$$\kappa_2(M^n) = \frac{(2n+1)!}{(n+1)!n!}. \tag{5.3}$$

While this conditioning seems spectacularly bad, in practice, the Bernstein basis seems to give entirely satisfactory results at moderately high orders of approximation [12]. Here, we hope to give at least a partial explanation of this phenomenon. Recall that solving $M^n\mathbf{c} = \mathbf{b}$ exactly yields the Bernstein expansion coefficients of the best L^2 polynomial approximation to a function f whose moments against the Bernstein basis are contained in $\mathbf{b}$. Consequently, if our solution process yields some $\mathbf{c}+\delta\mathbf{c}$ instead of $\mathbf{c}$, the L^2 norm of the polynomial encoded by $\delta\mathbf{c}$ has more direct relevance than the size of $\delta\mathbf{c}$ in the Euclidean norm. On the other hand, the perturbation $\delta\mathbf{b}$ exists as an array of numbers (say, the roundoff error in computing the moments of f by numerical integration), and we continue to use the Euclidean norm here.

In standard floating point analysis, suppose we have computed the solution to a perturbed system

$$M^n\left(\mathbf{c} + \delta\mathbf{c}\right) = \mathbf{b} + \delta\mathbf{b}, \tag{5.4}$$

and while a backward-stable algorithm yields a small $\delta\mathbf{b}$, the perturbation in the solution $\delta\mathbf{c}$ might itself be significant. Equivalently, the perturbation $\delta\mathbf{c}$ satisfies

$$M^n\delta\mathbf{c} = \delta\mathbf{b}. \tag{5.5}$$

While classical error perturbation and conditioning analysis would estimate the size of $\delta\mathbf{c}$ in the Euclidean norm, we wish to consider $\|\delta\mathbf{c}\|_{M^n}$, which is the L^2 norm of the perturbation in the computed polynomial.

Taking the inner product of (5.5) with $\delta\mathbf{c}$, we have

$$\|\delta\mathbf{c}\|_{M^n}^2 = \delta\mathbf{c}^T\delta\mathbf{b} \leq \|\delta\mathbf{c}\|_2\|\delta\mathbf{b}\|_2 \leq \|(M^n)^{-1}\|_2\|\delta\mathbf{b}\|_2^2 \tag{5.6}$$

so that

$$\|\delta\mathbf{c}\|_{M^n} \leq \left(\sqrt{\|(M^n)^{-1}\|_2}\right)\|\delta\mathbf{b}\|_2 = \sqrt{\lambda_{\min}^{-1}(M^n)}\|\delta\mathbf{b}\|_2. \tag{5.7}$$

In other words, for a perturbation $\delta\mathbf{b}$ of fixed 2-norm, the perturbation $\delta\mathbf{c}$ is much smaller when it is measured in the M^n norm rather than in the 2-norm—the amplification factor is only the square root as large.

This discussion suggests the following matrix norm. We think of M^n as an operator mapping $\mathbb{R}^{n+1}$ onto itself. However, we equip the domain with the M^n-weighted norm and the range with the Euclidean norm. Then we define the operator norm

$$\|A\|_{M^n\to 2} = \max_{\|\mathbf{c}\|_{M^n}=1} \|A\mathbf{c}\|_2, \tag{5.8}$$

and going the opposite direction,

$$\|A\|_{2\to M^n} = \max_{\|\mathbf{c}\|_2=1} \|A\mathbf{c}\|_{M^n}. \tag{5.9}$$

These two matrix norms naturally combine to define a new condition number

$$\kappa_{M^n\to 2}(A) = \|A\|_{M^n\to 2}\|A^{-1}\|_{2\to M^n}. \tag{5.10}$$

We will also need the matrix norm

$$\|A\|_{M^n} = \max_{\|\mathbf{c}\|_{M^n}=1} \|A\mathbf{c}\|_{M^n}, \tag{5.11}$$

using the M^n-weight in both the domain and the range.

In light of (5.9), we can interpret (5.7) as follows.

Proposition 5.1. *The norm of* $(M^n)^{-1}$ *satisfies*

$$\|(M^n)^{-1}\|_{2\to M^n} = \sqrt{\lambda_{\min}^{-1}(M^n)}. \tag{5.12}$$

We can, by the spectral decomposition, give a similar result for M^n in the $M^n \to 2$ norm as follows.

Proposition 5.2. *The norm of* M^n *satisfies*

$$\|(M^n)\|_{M^n\to 2} = \sqrt{\lambda_{\max}(M^n)}. \tag{5.13}$$

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

$$\begin{aligned}\|M^n\mathbf{c}\|_2^2 = (M^n\mathbf{c})^T(M^n\mathbf{c}) &= \mathbf{c}^T(M^n)^2\mathbf{c}\\ &= \left(\sqrt{M^n}\mathbf{c}\right)^T M^n \left(\sqrt{M^n}\mathbf{c}\right)\\ &= \|\sqrt{M^n}\mathbf{c}\|_{M^n}^2,\end{aligned}$$

and so

$$\|M^n\mathbf{c}\|_2 \le \|\sqrt{M^n}\|_{M^n}\|\mathbf{c}\|_{M^n}. \tag{5.14}$$

Now we can characterize the weighted norm of the square root matrix via the spectral decomposition $M^n = Q^n\Lambda^n(Q^n)^T$.

Note that, for any $\mathbf{c} \in \mathbb{R}^{n+1}$, we have its M^n norm as

$$\|\mathbf{c}\|_{M^n}^2 = \mathbf{c}^T M^n\mathbf{c} = \left((Q^n)^T\mathbf{c}\right)^T \Lambda^n \left((Q^n)^T\mathbf{c}\right). \tag{5.15}$$

Letting $\mathbf{d} = (Q^n)^T\mathbf{c}$,

$$\|\mathbf{c}\|_{M^n}^2 = \|\mathbf{d}\|_{\Lambda^n}^2 = \sum_{i=0}^{n} \lambda_i^n|\mathbf{d}_i|^2. \tag{5.16}$$

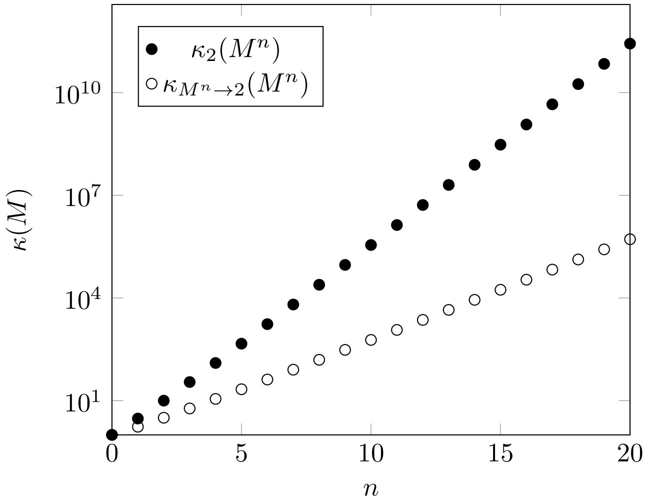


FIG. 1. *Bernstein mass matrix conditioning for degrees* 0 *through* 20 *in the* 2*-norm and the* $M^n \to 2$ *norms.*

Now we consider the M^n norm of $\sqrt{M^n}$. Again, for any $\mathbf{c} \in \mathbb{R}^{n+1}$, we have

$$\left\|\sqrt{M^n}\mathbf{c}\right\|_{M^n}^2 = \left(\sqrt{M^n}\mathbf{c}\right)^T M^n\sqrt{M^n}\mathbf{c} = \mathbf{c}^T(M^n)^2\mathbf{c} = \mathbf{c}^T Q^n \left(\Lambda^n\right)^2 \left(Q^n\right)^T\mathbf{c} \tag{5.17}$$

so that, with $\mathbf{d} = (Q^n)^T\mathbf{c}$, we have

$$\begin{aligned}\left\|\sqrt{M^n}\mathbf{c}\right\|_{M^n}^2 = \mathbf{d}^T(\Lambda^n)^2\mathbf{d} &= \sum_{i=0}^{n}(\lambda_i^n)^2|\mathbf{d}_i|^2 \\ &\leq \lambda_{\max}(M^n)\sum_{i=0}^{n}\lambda_i^n|\mathbf{d}_i|^2 \\ &= \lambda_{\max}(M^n)\|\mathbf{d}\|_{\Lambda^n}^2.\end{aligned}$$

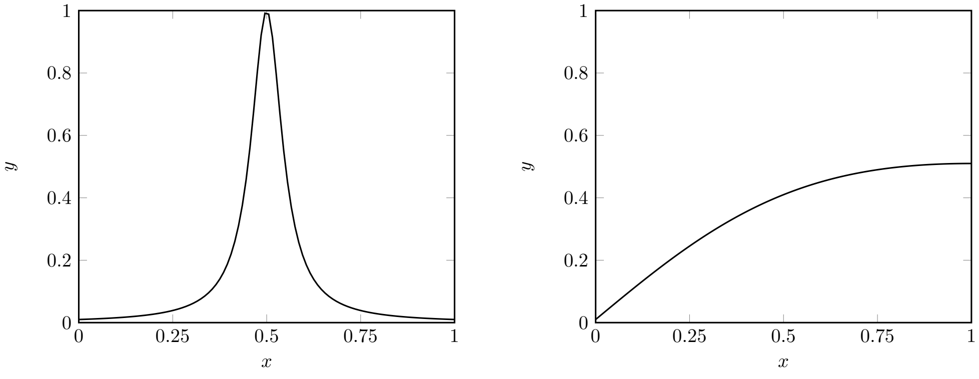
Consequently,

$$\left\|\sqrt{M^n}\right\|_{M^n} = \max_{\|\mathbf{c}\|_{M^n}=1}\left\|\sqrt{M^n}\mathbf{c}\right\|_{M^n} = \sqrt{\lambda_{\max}(M^n)}. \tag{5.18}$$ □

THEOREM 5.3. *The condition number of solving* $M^n\mathbf{c} = \mathbf{b}$ *measuring* $\mathbf{c}$ *in the* M^n*-norm and* $\mathbf{b}$ *in the* 2*-norm satisfies*

$$\kappa_{M^n\to 2}(M^n) = \sqrt{\kappa_2(M^n)} = \sqrt{\frac{(2n+1)!}{(n+1)!n!}}. \tag{5.19}$$

A plot of the condition numbers up to degree 20 in both norms is shown in Figure 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 (5.3) suggests. It is important to note that nothing in 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 [13] as well as the preconditioners for global mass matrices given in [3].



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

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

FIG. 2. *Plots of two functions being approximated with Bernstein polynomials. Figure* 2(a) *is smooth but has large derivatives and so needs high polynomial order to make the error small. The function in Figure* 2(b) *is much simpler to approximate but illustrates that the methods work on functions not symmetric about the interval midpoint.*

6. Numerical results.

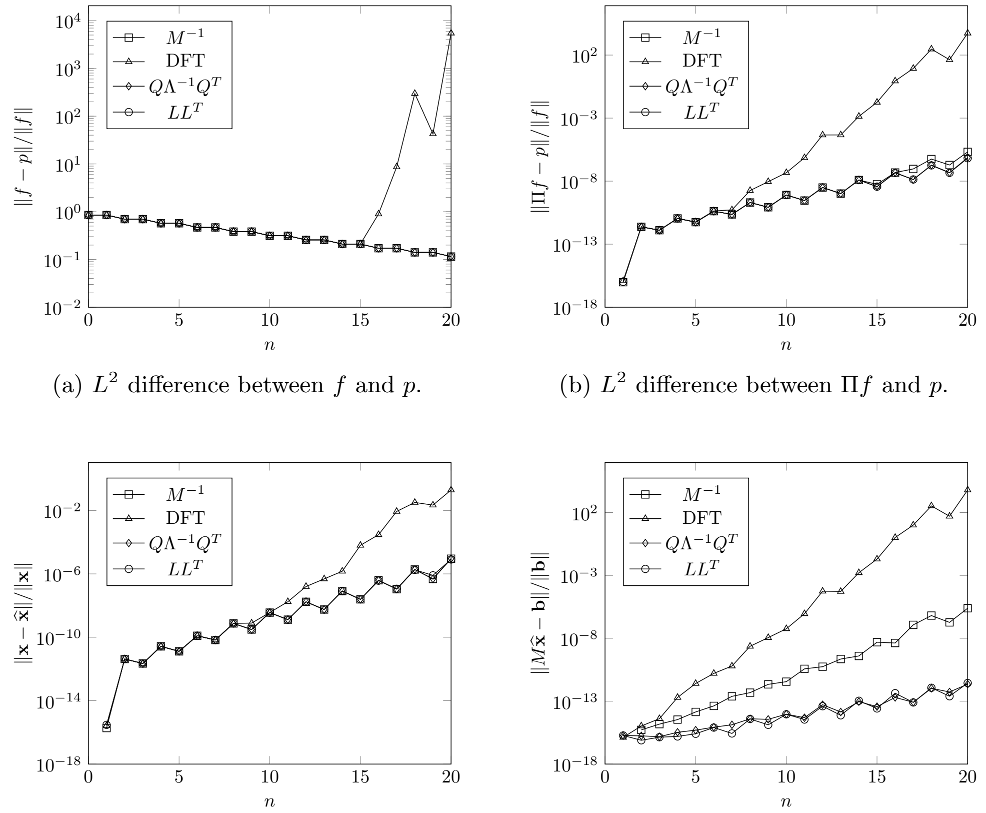
Now we consider the accuracy of the methods described above on a range of problems. All of our numerical results are run in double precision arithmetic on a 2014 MacBook Air running macOS 10.13 and using Python 2.7.13. Cholesky factorization and FFTs are performed using the `numpy` (v1.12.0) function 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 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 Figure 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 (2.2) accurately, the best polynomial approximation of degree n gives an L^2 error decreasing exponentially as a function of n. To demonstrate this, Figures 3(a) and 4(a) show the L^2 difference between the two functions and the L^2 projection computed with each of the four methods described in section 4. However, at various degrees, each of the methods deviates 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 difference between the exact and computed solution to (2.2). These plots appear in Figures 3(b) and 4(b), further demonstrating the instability of the DFT-based approach.

Differences between the three more stable methods become more apparent when we consider the Euclidean norm of the error (Figures 3(c) and 4(c)) and residual (Figures 3(d) and 4(d)) for each of our solution methods. The DFT-based approach again performs much worse than the other methods, with the exact inverse giving



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

(b) L^2 difference between Πf and p.

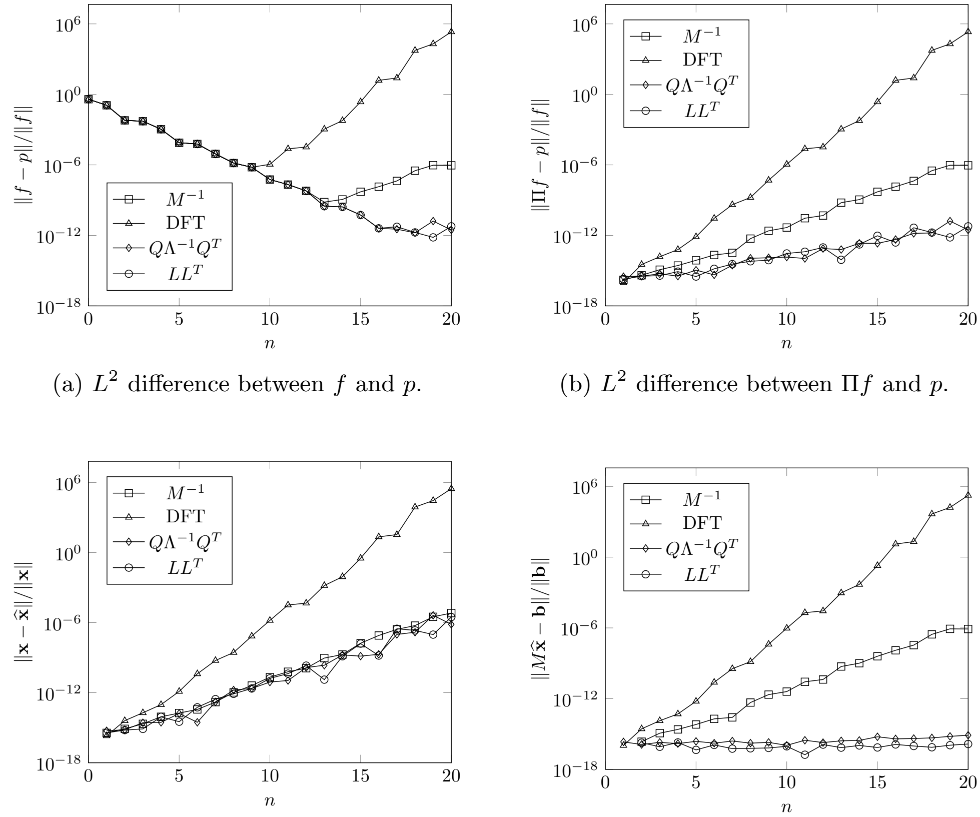
(c) Error in the 2-norm.

(d) Residual in the 2-norm.

FIG. 3. *Relative error/residual in using the methods described in section* 4 *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* $\mathbf{b}_i = \big(f(x), B_i^n(x)\big)_{L^2}$.

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 [8], 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 Figure 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. It is very interesting that three different vectors—the errors in using the exact inverse, Cholesky factorization, and spectral decomposition—should be so close in the Euclidean norm but vary so significantly in the M^n norm and that weighted by



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

(b) L^2 difference between Πf and p.

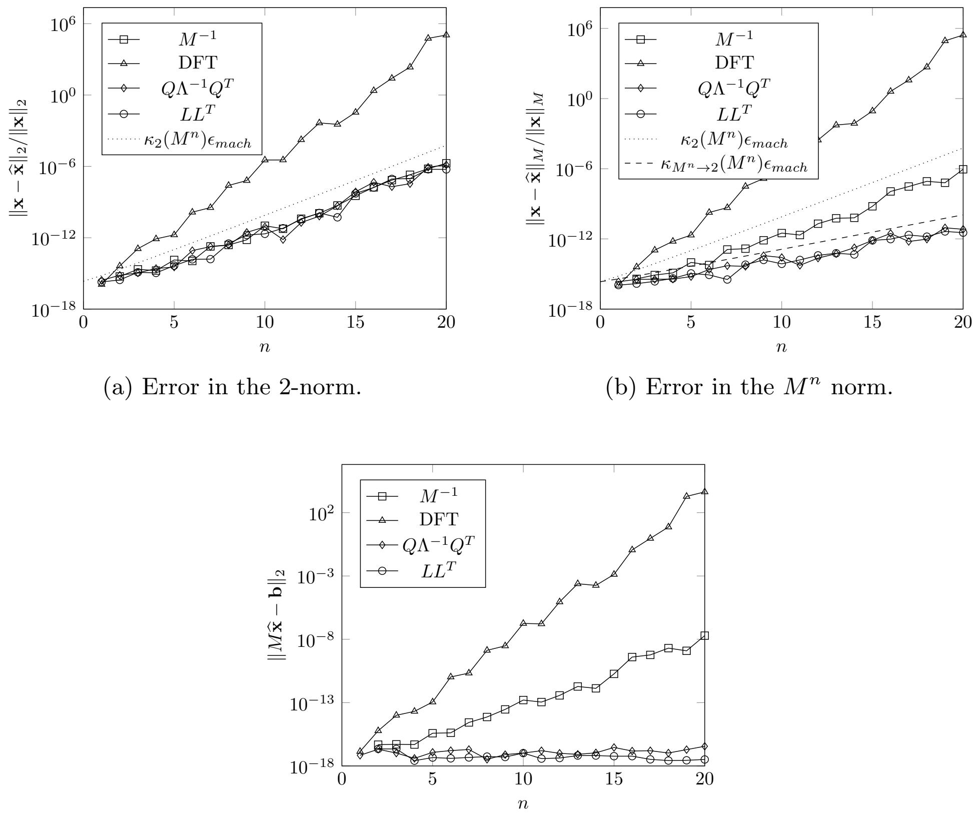
(c) Error in the 2-norm.

(d) Residual in the 2-norm.

FIG. 4. *Relative error/residual in using the methods described in section* 4 *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* $\mathbf{b}_i = \left(f(x), B_i^n(x)\right)_{L^2}$.

its inverse. This highlights that, although all finite-dimensional norms are equivalent, the "constants" need not be small and can grow quickly as a function of discretization parameters. (See [10, 18] for a discussion of this in the context of discrete PDEs.)

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 (Figures 3(b) and 4(b)) 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 5. 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 cost is higher—two dense matrix multiplications rather than two triangular solves.



(a) Error in the 2-norm.

(b) Error in the M^n norm.

(c) Residual in the 2-norm.

FIG. 5. *Error/residual in using the methods described in section* 4 *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. For reference, we have also included the condition numbers of* M^n *from Figure* 1 *scaled by the machine epsilon of about* 2.2×10^{-16} *as relevant.*

7. Conclusions. 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 DFT is unfortunately unstable. Moreover, we have given a new perspective on the conditioning of matrices for polynomial projection indicating that the problems are better conditioned with respect to the L^2 norm of the output than the Euclidean norm.

We comment briefly on the extension of this methodology to more general settings. In the case of a weighted mass matrix, such as can arise in certain variable coefficient PDEs, neither an explicit formula nor spectral decomposition is known, and so one must rely on the (thankfully stable!) Cholesky factorization. In the constant coefficient but multivariate case, our techniques can still find some application. On rectangular domains, one typically uses tensor products of Bernstein polynomials. Since the resulting mass matrix is a Kronecker product of univariate mass matrices, any of our algorithms discussed could be used repeatedly to solve the problem. We refer the reader to [13] for the case of simplicial geometry, which still requires inversion of univariate mass matrices. For multivariate weighted mass matrices, we again can only recommend the Cholesky decomposition at this time.

In the future, we hope to expand this perspective to other polynomial problems and continue the development of fast and accurate methods for problems involving Bernstein polynomials.

REFERENCES

[1] M. Ainsworth, G. Andriamaro, and O. Davydov, *Bernstein–Bézier finite elements of arbitrary order and optimal assembly procedures*, SIAM J. Sci. Comput., 33 (2011), pp. 3087–3109.

[2] M. Ainsworth and G. Fu, *Bernstein–Bézier bases for tetrahedral finite elements*, Comput. Methods Appl. Mech. Engrg., 340 (2018), pp. 178–201.

[3] M. Ainsworth, S. Jiang, and M. A. Sanchéz, *An $\mathcal{O}(p^3)$ hp-version FEM in two dimensions: Preconditioning and post-processing*, Comput. Methods Appl. Mech. Engrg., 350 (2019), pp. 766 – 802, https://doi.org/10.1016/j.cma.2019.03.020.

[4] D. N. Arnold, R. S. Falk, and R. Winther, *Geometric decompositions and local bases for spaces of finite element differential forms*, Comput. Methods Appl. Mech. Engrg., 198 (2009), pp. 1660–1672.

[5] S. Bernstein, *Démonstration du théorème de weierstrass fondèe sur le calcul des probabilités*, Commun. Soc. Math. Kharkov, 13 (1912), pp. 1–2.

[6] M. G. Duffy, *Quadrature over a pyramid or cube of integrands with a singularity at a vertex*, SIAM J. Numer. Anal., 19 (1982), pp. 1260–1262.

[7] R. T. Farouki, *Legendre–Bernstein basis transformations*, J. Comput. Appl. Math., 119 (2000), pp. 145–160.

[8] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Vol. 3, John Hopkins University Press, Baltimore, MD, 2012.

[9] G. Heinig and K. Rost, *Algebraic Methods for Toeplitz-Like Matrices and Operators*, Vol. 13, Birkhäuser-Verlag, Basel, Switzerland, 1984.

[10] R. C. Kirby, *From functional analysis to iterative methods*, SIAM Rev., 52 (2010), pp. 269–293.

[11] R. C. Kirby, *Fast simplicial finite element algorithms using Bernstein polynomials*, Numer. Math., 117 (2011), pp. 631–652.

[12] R. C. Kirby, *Low-complexity finite element algorithms for the de Rham complex on simplices*, SIAM J. Sci. Comput., 36 (2014), pp. A846–A868.

[13] R. C. Kirby, *Fast inversion of the simplicial Bernstein mass matrix*, Numer. Math., 135 (2017), pp. 73–95.

[14] R. C. Kirby and K. T. Thinh, *Fast simplicial quadrature-based finite element operators using Bernstein polynomials*, Numer. Math., 121 (2012), pp. 261–279.

[15] S. Lewanowicz and P. Woźny, *Bézier representation of the constrained dual Bernstein polynomials*, Appl. Math. Comput., 218 (2011), pp. 4580–4586.

[16] L. Lu, *Gram matrix of Bernstein basis: Properties and applications*, J. Comput. Appl. Math., 280 (2015), pp. 37–41.

[17] F. T. Luk and S. Qiao, *A fast eigenvalue algorithm for Hankel matrices*, Linear Algebra Appl., 316 (2000), pp. 171–182.

[18] K.-A. Mardal and R. Winther, *Preconditioning discretizations of systems of partial differential equations*, Numer. Linear Algebra Appl., 18 (2011), pp. 1–40.

[19] R. B. Platte and L. N. Trefethen, *Chebfun: A new kind of numerical computing*, in Progress in Industrial Mathematics at ECMI 2008, Springer, New York, 2010, pp. 69–87.

[20] G. Strang, *Linear algebra and its applications*, Thomson, 2006.

[21] L. N. Trefethen, *Approximation Theory and Approximation Practice*, Vol. 128, SIAM, Philadelphia, 2013.