Journal of Scientific Computing (2021) 88:53
https://doi.org/10.1007/s10915-021-01586-w

On the Computation of Recurrence Coefficients for Univariate Orthogonal Polynomials

Zexin Liu[1] · **Akil Narayan**[1]

Received: 30 January 2021 / Revised: 16 June 2021 / Accepted: 4 July 2021 /
Published online: 20 July 2021

Abstract
Associated to a finite measure on the real line with finite moments are recurrence coefficients in a three-term formula for orthogonal polynomials with respect to this measure. These recurrence coefficients are frequently inputs to modern computational tools that facilitate evaluation and manipulation of polynomials with respect to the measure, and such tasks are foundational in numerical approximation and quadrature. Although the recurrence coefficients for classical measures are known explicitly, those for nonclassical measures must typically be numerically computed. We survey and review existing approaches for computing these recurrence coefficients for univariate orthogonal polynomial families and propose a novel "predictor–corrector" algorithm for a general class of continuous measures. We combine the predictor–corrector scheme with a stabilized Lanczos procedure for a new hybrid algorithm that computes recurrence coefficients for a fairly wide class of measures that can have both continuous and discrete parts. We evaluate the new algorithms against existing methods in terms of accuracy and efficiency.

Keywords Orthogonal polynomials · Recurrence coefficients

Mathematics Subject Classification 33D45 · 42C10 · 65D15

1 Introduction

Univariate orthogonal polynomials are a mainstay tool in numerical analysis and scientific computing. These polynomials serve as theoretical foundations for numerical algorithms involving approximation and quadrature [4,12,13,22,31]. Given a positive measure μ on the

✉ Zexin Liu
zexin@math.utah.edu

Akil Narayan
akil@sci.utah.edu

[1] Department of Mathematics, and Scientific Computing and Imaging (SCI) Institute, The University of Utah, Salt Lake City, USA

real line $\mathbb{R}$, if μ has finite polynomial moments of all orders along with an infinite number of points of increase, then a family of orthonormal polynomials $\{p_n\}_{n=0}^{\infty}$ exists, satisfying $\deg p_n = n$, and

$$\int_{\mathbb{R}} p_n(x)\, p_m(x) \mathrm{d}\mu(x) = \delta_{m,n},$$

where $\delta_{m,n}$ is the Kronecker delta. If we further assume that each p_n has a positive leading coefficient, then these polynomials are unique. Such families are known to obey a three-term recurrence formula,

$$x p_n(x) = b_n p_{n-1}(x) + a_{n+1} p_n(x) + b_{n+1} p_{n+1}(x), \qquad n \geq 0, \tag{1}$$

with the starting conditions $p_{-1} \equiv 0$ and $p_0(x) = 1/b_0$. The coefficients $(a_n)_{n=1}^{\infty} \subset \mathbb{R}$ and $(b_n)_{n=0}^{\infty} \subset (0, \infty)$ depend only on the (polynomial) moments of μ. In practical settings, knowledge of these coefficients is the only requirement for implementing stable, accurate algorithms that achieve evaluation and manipulation of polynomials that are core components of approximation and quadrature algorithms. For example, the n eigenvalues of the $n \times n$ Jacobi matrix $\boldsymbol{J}_n$ are precisely the abscissae of a μ-Gaussian quadrature rule, with J_n the symmetric tridiagonal matrix given by

$$\boldsymbol{J}_n(\mu) = \begin{pmatrix} a_1 & b_1 & & & \\ b_1 & a_2 & b_2 & & \\ & \ddots & \ddots & \ddots & \\ & & b_{n-2} & a_{n-1} & b_{n-1} \\ & & & b_{n-1} & a_n \end{pmatrix}. \tag{2}$$

Therefore, the recurrence coefficients a_n and b_n must be computed stably and accurately.

Some classical probability measures μ give rise to classical families of orthogonal polynomials p_n: A Gaussian measure results in Hermite polynomials; the uniform measure on a compact interval results in Legendre polynomials; a Beta measure corresponds with Jacobi polynomials; and a one-sided exponential measure gives rise to Laguerre polynomials. These classical polynomial families are among a few for which explicit formulas are available for the recurrence coefficients a_n and b_n, see, e.g., [12, Tables 1.1, 1.2].

However, for even modestly complicated measures μ outside this classical collection, the task of determining these coefficients can be quite difficult. For example, an application in which this situation arises is in polynomial Chaos methods, which are techniques in scientific computing problems for modeling the effect of uncertainty in a model [35,38]. An output's dependence on a finite number of random variable inputs is modeled with polynomial dependence on those inputs. With one random input, the polynomial approximation is typically constructed using a basis of polynomials orthogonal to the distribution of the random input, which requires building orthogonal polynomials with respect to a given, often nonclassical, probability measure.

A simple example that illustrates how computation of orthogonal polynomials is difficult for even fairly simple measures is furnished by the class of *Freud weights*,

$$\mathrm{d}\mu(x) = \exp\left(-|x|^{\alpha}\right) \mathrm{d}x, \qquad \alpha > 0, \tag{3}$$

with support equal to all of $\mathbb{R}$. (In what follows, we will refer to μ as a measure and $\mathrm{d}\mu$ as a weight.) When $\alpha = 2$, corresponding to the Gaussian measure (and Hermite polynomial family), the three-term recurrence coefficients are known exactly. However, when $\alpha = 1$, no closed-form analytical formula for the coefficients a_n and b_n exists, even though the moments

of μ are known explicitly in terms of well-studied special functions. (For example, note that under a change of variable, the moments of the measure above correspond to evaluations of the Euler Gamma function.)

In such general cases when no known closed-form expression for the three-term recurrence coefficients exists, numerical methods are employed to approximate them. The main goal of this article is to survey and extend existing methods for computing these recurrence coefficients associated to measures for which explicit formulas are not available.

1.1 Existing Approaches

When μ is not a measure for which the coefficients have explicitly known formulas, one typically resorts to numerical methods to approximately compute these coefficients. A summary of the methods we consider in this article is presented in Table 1, which indicates later sections in this article where we give a formal description of each algorithm. A brief description of these procedures is given in Sect. 2, but an excellent and more detailed historical survey is provided in [12, Section 2.6]. Below we present a nontechnical summary of the approaches that we survey.

A classical approach to computing recurrence coefficients from moments is via determinants of Hankel matrices [12, Section 2.1.1]. A second classical approach, the Chebyshev algorithm, transforms monomial moments by expressing the recurrence coefficients in terms of moments involving monomials and p_n [2]. A more effective approach, the modified Chebyshev algorithm, uses moments involving p_n and another arbitrary set of polynomials [7,26,33]. Yet another procedure, the Stieltjes algorithm [28], computes recurrence coefficients directly assuming moments involving p_n can be computed. Finally, given a measure with discrete support, the Lanczos algorithm can be used to compute the Jacobi matrix for μ, yielding the recurrence coefficients; although this is typically unstable, a stable variant is given in [25].

For very special forms of weight functions, other procedures can be derived. A primary example of this are iterative recurrence-type algorithms resulting from discrete Painlevé equations when $\mathrm{d}\mu(x) \propto \exp(-x^\alpha)$ for $\alpha/2 \in \mathbb{N}$. These Painlevé equations, which determine the recurrence coefficients for p_n, are remarkably simple and direct to implement, but are quite unstable [32]. A final approach we consider amounts to using a linear orthogonalization procedure, such as (modified) Gram-Schmidt, to compute the expansion coefficients of p_n in terms of the monomials. However, this procedure is known to produce quite ill-conditioned matrices, especially for large n, making the computation of p_n, and hence the recurrence coefficients, suffer roundoff errors. Therefore, although this approach has often been used [36,37], it is less useful in the context of this article. Nevertheless, we consider one recent related approach, an "arbitrary polynomial chaos" approach suggested in [23], which amounts to solving a linear system involving a modified Hankel matrix.

1.2 Contributions of this Article

Several algorithms exist to compute the recurrence coefficients, but a few clear and direct recommendations are available for researchers without substantial experience and/or knowledge of the field. The main contribution of this paper is to summarize, evaluate, and extend existing methods for computing recurrence coefficients for univariate orthogonal polynomial families. We first provide a survey and comparison of many existing algorithms (see Sect. 2). In Sect. 3.1 we propose a novel "predictor–corrector" algorithm and evaluate its util-

Table 1 Abbreviation, subsection, and algorithm for each method

Method	Abbreviation	Section	Citation
Discrete Painlevé I equations method	`DP`	2.1	[32]
Hankel Determinants	`HD`	2.2	[12, Section 2.1.1]
Arbitrary polynomial chaos expansion method	`aPC`	2.3	[23, Section 3.1]
Modified Chebyshev algorithm	`MC`	2.4	[12, Section 2.1.7]
Stieltjes procedure	`SP`	2.5	[12, Section 2.2.3.1]
Stabilized Lanczos algorithm	`LZ`	2.6	[12, Section 2.2.3.2]
Predictor–corrector method	`PC`	3.1	—
Predictor–corrector-Lanczos method	`PCL`	3.3	—

Also included is a modern citation that explains each algorithm

ity. Finally, by modifying the "multiple component" approach in [8,10], we consider a new hybrid algorithm in Sect. 3.3 that combines our predictor–corrector scheme with a stabilized Lanczos procedure. Our algorithm can be used to compute recurrence coefficients for the fairly general class of measures whose differentials are given by

$$\mathrm{d}\mu(x) = \sum_{j=1}^{C} w_j(x)\mathbb{1}_{I_j}(x)\mathrm{d}x + \sum_{j=1}^{M} \nu_j \delta_{\tau_j}\mathrm{d}x, \tag{4}$$

where C and M are finite (either possibly 0), δ_{τ_j} is a Dirac mass located at $\tau_j \in \mathbb{R}$, $\{\nu_j\}_{j=1}^{M}$ are positive scalars, each I_j is a (possibly unbounded) nontrivial interval, and w_j is a continuous (ideally smooth) non-negative function on I_j. Specification of the w_j, I_j, τ_j, and ν_j is sufficient to utilize most of the algorithms we consider, but having extra information that characterizes w_j, particularly prescribed behavior at finite endpoints of I_j, will increase the accuracy of the procedures. In other words, with $I_j = [\ell_j, r_j]$ and either of the endpoints ℓ_j, r_j is finite, we assume knowledge of exponents $\beta_j, \alpha_j > -1$ such that w_j has polynomial singular strength β_j, α_j at endpoints ℓ_j, r_j, i.e.,

$$0 < \lim_{x\downarrow\ell_j} w_j(x)(x-\ell_j)^{-\beta_j} < \infty, \qquad 0 < \lim_{x\uparrow r_j} w_j(x)(r_j - x)^{-\alpha_j} < \infty. \tag{5}$$

Note that our assumption that $\alpha_j, \beta_j > -1$ is natural since if the inequality above is true with, say, $\alpha_j \le -1$, then μ is not a finite measure and therefore is not a probability measure.

Note that the form of μ we assume in (4) is quite general, and includes all classical measures, those with piecewise components, measures with discrete components, measures with unbounded support, and measures whose densities have integrable singularities.

This paper is structured as follows: In Sect. 2 we briefly survey the existing approaches summarized in Table 1. Section 3 contains the discussion that leads to our proposed hybrid "PCL" algorithm: Sect. 3.1 discusses the predictor–corrector scheme; Sect. 3.2 briefly describes how we compute moments, which leverages the specific form of the measure μ assumed in (4) and (5); Sect. 3.3 combines these with a stabilized Lanczos procedure. Finally, we present a wide range of numerical examples in Sect. 4, which compares many of the techniques in Table 1, and demonstrates the accuracy and efficiency of the "PCL" algorithm.

2 Existing Approaches

We review here some existing methods for computing recurrence coefficients. In order to compute the required coefficients, having some knowledge about the measure μ is neccessary. The following are two of the more common assumptions that one makes, with the latter assumption being stronger:

– The (monomial) moments of all orders of μ are known, i.e., the moment sequence

$$m_n := \int x^n \mathrm{d}\mu(x), \quad n \ge 0, \tag{6}$$

is known and available. In practice, the integrals can be obtained by the composite quadrature approach introduced in Sect. 3.2, but sometimes they can also be computed directly in terms of special functions, such as Gamma function given the Freud weights.

– General polynomial moments, i.e.,

$$\int q(x)\mathrm{d}\mu(x), \tag{7}$$

are computable for a general, finite-degree polynomial q that is often identified only partway through an algorithm.

No particular prescription exists for how the moments above are computed, but typically this is accomplished through a quadrature rule. In some "data-driven" scenarios, this quadrature rule often comes as a Monte Carlo rule from an empirical ensemble.

We discuss six procedures below; in practice, only the last two are computationally stable, but they are all useful for comparison purposes. The first procedure works only for very special Freud weights, i.e., those with exponential behavior.

2.1 DP: Freud Weights and Discrete Painlevé Equations

Freud weights, named after Géza Freud who studied them in the 1970s [5], have the following form:

$$\mathrm{d}\mu(x) = |x|^{\rho} \exp(-|x|^{\alpha})\mathrm{d}x, \quad \rho > -1, \alpha > 0. \tag{8}$$

Observe that Freud weights are symmetric, which implies that $a_n = 0$ for $n \geq 0$, and therefore only the b_n coefficients need be computed. Freud gave a recurrence relation for the recurrence coefficients b_n when $\alpha = 2, 4, 6$. The connection between Freud weights and discrete Painlevé equations was first pointed out by Magnus [21]. In the case of $\alpha = 4$, one can derive the following recurrence relation for $n \geq 1$ by letting $x_n := 2b_n^2$:

$$x_{n+1} = \frac{1}{x_n}\left(n + \frac{\rho}{2}\left(1 + (-1)^n\right)\right) - x_n - x_{n-1}, \qquad x_0 = 0, \qquad x_1 = \frac{2\Gamma(\frac{3+\rho}{4})}{\Gamma(\frac{1+\rho}{4})}. \tag{9}$$

See, e.g., [32, Section 2.2]. This recurrence relation is a discrete Painlevé I equation [21] that is useful for theoretical analysis. For example, it can be used to prove Freud's conjecture, which is a statement about asymptotic behavior of the b_n coefficients. For $\alpha = 4$ in this section, Freud's conjecture states

$$\lim_{n\to\infty} \frac{b_n}{n^{1/4}} = \frac{1}{\sqrt[4]{12}}. \tag{10}$$

A more general resolution of Freud's conjecture using alternative methods is provided in [20].

Similarly, when $\alpha = 6$, by letting $y_n := b_n^2$, a fourth-order nonlinear recurrence relation for $n \geq 2$ [32, Section 2.3] is given by

$$\begin{aligned} &6y_n\left(y_{n-2}y_{n-1} + y_{n-1}^2 + 2y_{n-1}y_n + y_{n-1}y_{n+1} + y_n^2 + 2y_n y_{n+1} + y_{n+1}^2 + y_{n+1}y_{n+2}\right) \\ &\quad = n + \frac{\rho}{2}\left(1 + (-1)^n\right), \end{aligned} \tag{11}$$

with initial condition

$$y_0 = 0, \qquad y_1 = \frac{\Gamma(\frac{3+\rho}{6})}{\Gamma(\frac{1+\rho}{6})},$$

$$y_2 = \frac{\Gamma(\frac{5+\rho}{6})}{\Gamma(\frac{3+\rho}{6})} - y_1, \quad y_3 = \frac{\Gamma(\frac{7+\rho}{6})}{y_2 y_1 \Gamma(\frac{1+\rho}{6})} - \frac{2(y_1+y_2)\Gamma(\frac{5+\rho}{6})}{y_2 y_1 \Gamma(\frac{1+\rho}{6})} + \frac{(y_1+y_2)^2\Gamma(\frac{3+\rho}{6})}{y_2 y_1 \Gamma(\frac{1+\rho}{6})}.$$

In this case, Freud's conjecture states

$$\lim_{n\to\infty} \frac{b_n}{n^{1/6}} = \frac{1}{\sqrt[6]{60}}. \tag{12}$$

Note the computation of recursion coefficients via (9) and (11) is quite straightforward, but is also very unstable. Nevertheless, there is a unique positive solution [19]; hence, a small (e.g., machine roundoff) error in x_1 or y_1 quickly results in the loss of positivity of x_n or y_n. Numerical solutions follow the exact asymptotic behavior well until large deviations from the true solution eventually appear, cf. Fig. 1.

2.2 `HD`: Hankel Determinants

Orthogonal polynomials as well as their recursion coefficients are expressible in determinantal form in terms of the moments of the underlying measure. Indeed, much of the classical theory of orthogonal polynomials is moment-oriented. One classical technique to express recurrence coefficients in terms of moments is via matrix determinants.

We introduce the Hankel determinant Δ_n of order n in terms of the finite moments (6), defined as

$$\Delta_{-1} = 1, \quad \Delta_0 = 1, \quad \Delta_n = \det \boldsymbol{H}_n, \quad \boldsymbol{H}_n := \begin{pmatrix} m_0 & m_1 & \cdots & m_{n-1} \\ m_1 & m_2 & \cdots & m_n \\ \vdots & \vdots & \ddots & \vdots \\ m_{n-1} & m_n & \cdots & m_{2n-2} \end{pmatrix}, \quad n \in \mathbb{N}. \tag{13}$$

These determinants of Gram matrices are associated to the μ-inner product, using a basis of monomials. In addition, we define determinants Δ'_n of modified Hankel matrices, where the modification is to replace the last column of $\boldsymbol{H}_n$ by the last column of $\boldsymbol{H}_{n+1}$ with the trailing entry removed,

$$\Delta'_0 = 0, \qquad \Delta'_1 = m_1, \qquad \Delta'_n = \begin{vmatrix} m_0 & m_1 & \cdots & m_{n-2} & m_n \\ m_1 & m_2 & \cdots & m_{n-1} & m_{n+1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ m_{n-1} & m_n & \cdots & m_{2n-3} & m_{2n-1} \end{vmatrix}, \qquad n = 2, 3, \ldots .$$

Along with $b_0 = \sqrt{m_0}$, the orthogonal polynomial recurrence coefficients can be computed explicitly from these determinants, cf.[12, Theorem 2.2],

$$a_n = \frac{\Delta'_n}{\Delta_n} - \frac{\Delta'_{n-1}}{\Delta_{n-1}}, \qquad b_n = \sqrt{\frac{\Delta_{n+1}\Delta_{n-1}}{\Delta_n^2}}, \qquad n \in \mathbb{N}. \tag{14}$$

The formulas (14) are not practically useful as an algorithm to compute reucrrence coefficients since the Hankel matrices above are typically ill-conditioned. In particular, the map that computes recurrence coefficients from moments can be severely ill-conditioned [12, Section 2.1.6].

2.3 `aPC`: "Arbitrary" Polynomial Chaos Expansions

The arbitrary polynomial chaos (`aPC`), like all polynomial chaos expansion techniques, approximates the dependence of simulation model output on model parameters by expansion

in an orthogonal polynomial basis. As shown in [23], aPC at finite expansion order demands the existence of only a finite number of moments and does not require the complete knowledge of a probability density function. Once we construct the polynomials such that they form an orthonormal basis for arbitrary distributions from the moment-based analysis, the recurrence coefficients can be derived using the aPC expansion coefficients.

Our goal is, firstly, to construct the polynomials in (15) such that they form an orthonormal basis for arbitrary distributions. Instead of the normality condition, we will first introduce an intermediate auxiliary condition by demanding that the leading coefficients of all polynomials be equal to 1.

We define the monic orthogonal polynomial $\pi_n(x)$ as

$$\pi_n(x) = \sum_{i=0}^{n} c_i^{(n)} x^i, \tag{15}$$

where $c_i^{(n)}$ are expansion coefficients, and specifically, $c_n^{(n)} = 1, \forall n$. The general conditions of orthogonality for $\pi_n(x)$ with respect to all lower order polynomials can be written in the following form [23, Section 3.1]:

$$\int_\Omega x^k \left(\sum_{i=0}^{n} c_i^{(n)} x^i \right) \mathrm{d}\mu(x) = 0, \qquad k = 0, 1, \ldots, n-1. \tag{16}$$

For each n, the system of equations given by (16) defines the unknown polynomial expansion coefficients in (15). Using finite moments in (6), the system can be reduced to

$$\sum_{i=0}^{n} c_i^{(n)} m_{i+k} = 0.$$

Alternatively, the system of linear equations can be written in the more convenient matrix form,

$$\begin{pmatrix} m_0 & m_1 & \cdots & m_n \\ m_1 & m_2 & \cdots & m_{n+1} \\ \vdots & \vdots & \ddots & \vdots \\ m_{n-1} & m_n & \cdots & m_{2n-1} \\ 0 & 0 & \cdots & 1 \end{pmatrix} \begin{pmatrix} c_0^{(n)} \\ c_1^{(n)} \\ \vdots \\ c_{n-1}^{(n)} \\ c_n^{(n)} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix}. \tag{17}$$

By defining the coefficient vector $\boldsymbol{c}^{(n)} = \left(c_0^{(n)}, c_1^{(n)}, \ldots, c_n^{(n)} \right)^T$, the normalized coefficients $\bar{c}_i^{(n)}$ can be expressed in terms of $\boldsymbol{c}^{(n)}$ and Hankel matrices $\boldsymbol{H}_{n+1}$,

$$\bar{c}_i^{(n)} = \frac{c_i^{(n)}}{\sqrt{\boldsymbol{c}^{(n)T} \boldsymbol{H}_{n+1} \boldsymbol{c}^{(n)}}}. \tag{18}$$

Together with $b_0 = \sqrt{m_0}$ and $c_{-1}^{(0)} := 0$, the recurrence coefficients can be obtained from (18) using (1),

$$a_n = \frac{\bar{c}_{n-2}^{(n-1)} - b_n \bar{c}_{n-1}^{(n)}}{\bar{c}_{n-1}^{(n-1)}}, \qquad b_n = \frac{\bar{c}_{n-1}^{(n-1)}}{\bar{c}_n^{(n)}}, \qquad n \in \mathbb{N}. \tag{19}$$

Thus, given the moments m_i, we first solve for the $c_k^{(n)}$ via (17) and subsequently uses (19) to compute the recurrence coefficients. As with the Hankel determinant procedure in Sect. 2.2, this procedure is susceptible to instability since the moment matrices in (17) are typically unstable.

2.4 MC: Modified Chebyshev Algorithm

The previous techniques have used (monomial) moments directly and suffer from numerical stability issues. The classical Chebyshev algorithm [2] still uses monomial moments, but it employs them through an iterative recursive approach to compute the recurrence coefficients. The technique in this section modifies the classical Chebyshev algorithm by using μ-moments computed with respect to some other set of polynomials $\{q_k\}$. Typically, q_k is chosen as a sequence of polynomials that are orthogonal with respect to another measure λ, where we require that the recurrence coefficients c_n, d_n for λ are known. The Modified Chebyshev algorithm is effective when λ is chosen "close" to μ.

We define the "mixed" moments as

$$\sigma_{n,k} = \int \pi_n(x) q_k(x) \mathrm{d}\mu(x), \qquad n, k > -1, \tag{20}$$

where $\pi_n(x)$ are the monic orthogonal polynomials with respect to μ. We denote a_n, b_n as the recurrence coefficients of orthonormal polynomials $p_n(x)$ with respect to μ. They can be used to formulate the three-term recurrence relation for monic orthogonal polynomials $\pi_n(x)$,

$$\pi_{n+1}(x) = (x - a_{n+1})\pi_n(x) - b_n^2 \pi_{n-1}(x). \tag{21}$$

We define c_k, d_k as recurrence coefficients of orthonormal polynomials $q_k(x)$. Plugging (21) into (20), the mixed moments $\sigma_{n,k}$, in turn, satisfies the recurrence relation below:

$$\begin{aligned} \sigma_{0,k} &= m_k, \\ \sigma_{n,k} &= d_k \sigma_{n-1,k-1} + (c_{k+1} - a_n)\sigma_{n-1,k} + d_{k+1}\sigma_{n-1,k+1} - b_{n-1}^2 \sigma_{n-2,k}. \end{aligned} \tag{22}$$

(22) gives a routine to compute the first N recurrence coefficients, which requires as input the first $2N-1$ modified moments $\{m_k\}_{k=0}^{2N-2}$ and $\{c_k, d_k\}_{k=0}^{2N-1}$.

Together with (21), (22) and the fact that $\sigma_{-1,k} = 0$, we have the expression of the recurrence coefficients,

$$\begin{aligned} a_1 &= c_1 + \frac{d_1 \sigma_{0,1}}{\sigma_{0,0}}, & a_n &= c_n + \frac{d_n \sigma_{n-1,n}}{\sigma_{n-1,n-1}} - \frac{d_{n-1}\sigma_{n-2,n-1}}{\sigma_{n-2,n-2}}, & n &= 2, 3, \ldots, \\ b_0 &= \sqrt{d_0 m_0}, & b_n &= \sqrt{\frac{d_n \sigma_{n,n}}{\sigma_{n-1,n-1}}}, & n &\in \mathbb{N}. \end{aligned} \tag{23}$$

Given a positive measure μ on $\mathbb{R}$, by choosing λ near μ in some sense, we expect the algorithm is well, or better, conditioned [12, Section 2.1.3].

2.5 SP: The Stieltjies Procedure

The previous procedures have used either monomial moments or general (mixed) moments with respect to a prescribed, fixed alternative basis q_k. In constrast, the Stieltjes procedure

[8,29] requires "on-demand" computation of moments, i.e., the moments required are determined during the algorithm. Starting with $b_0 = \left(\int \mathrm{d}\mu\right)^{1/2}$ and $p_0(x) = 1/b_0$, a_1 can be computed from (7) with $q(x) = xp_0(x)^2$, which allows us to evaluate $p_1(x)$ by means of (1). $p_1(x)$, in turn, can be used to generate b_1. The formulae [12, Section 2.2.3]

$$a_n = \int x p_{n-1}^2(x)\mathrm{d}\mu, \quad b_n = \left(\int ((x-a_n)p_{n-1}(x) - b_{n-1}p_{n-2}(x))^2\mathrm{d}\mu\right)^{\frac{1}{2}}, \quad n \in \mathbb{N}, \tag{24}$$

for the recursion coefficients provides a natural iterative framework for computing them.

2.6 LZ: A Lanczos-Type Algorithm

We assume that the measure $\mathrm{d}\mu$ is a discrete measure with finite support, i.e., (4) holds with $C = 0$ and $0 < M < \infty$. We wish to compute recurrence coefficients (a_n, b_n) up to $n < M$, ensuring that orthogonal polynomials up to this degree exist. We could also consider applying this procedure to a finite discretization of a continuous measure; see [12, Section 2.2.3.2 and Theorem 2.32].

The Lanczos procedure produces recurrence coefficients for the discrete measure μ, and utilizes the Lanczos algorithm that unitarily triangularizes a symmetric matrix. With $(\tau_j, \nu_j)_{j=1}^M$ the quadrature rule associated to the measure μ in (4), we define

$$\sqrt{\boldsymbol{\nu}} := \left(\sqrt{\nu_1},\ \sqrt{\nu_2},\ \ldots,\ \sqrt{\nu_M}\right)^T, \qquad \boldsymbol{D} := \mathrm{diag}\left(\tau_1,\ \tau_2,\ \ldots,\ \tau_M\right).$$

We define $\boldsymbol{Q}$ as a scaled $M \times M$ Vandermonde-like matrix,

$$\boldsymbol{Q} = \mathrm{diag}\left(\sqrt{\boldsymbol{\nu}}\right)\boldsymbol{V}, \qquad (\boldsymbol{V})_{j,k} = p_{j-1}(\tau_k),$$

for $j, k = 1, \ldots, M$. Then, $\boldsymbol{Q}$ is an orthogonal matrix by orthonormality of p_n. The orthogonality and the three-term recurrence further imply that,

$$\begin{pmatrix} 1 & \boldsymbol{0}^T \\ \boldsymbol{0} & \boldsymbol{Q} \end{pmatrix}\begin{pmatrix} 1 & \sqrt{\boldsymbol{\nu}}^T \\ \sqrt{\boldsymbol{\nu}} & \boldsymbol{D} \end{pmatrix}\begin{pmatrix} 1 & \boldsymbol{0}^T \\ \boldsymbol{0} & \boldsymbol{Q}^T \end{pmatrix} = \begin{pmatrix} 1 & b_0\boldsymbol{e}_1^T \\ b_0\boldsymbol{e}_1 & \boldsymbol{J}_M(\mu) \end{pmatrix},$$

where $\boldsymbol{e}_1 = (1, 0, 0, \ldots)^T \in \mathbb{R}^M$. The Lanczos algorithm, given the middle matrix on the left-hand side, computes the unitary triangularization above and outputs the right-hand side, which identifies the Jacobi matrix $\boldsymbol{J}_M$ in (2), and, hence, the recurrence coefficients. See [12, Section 2.2.3.2] for more details. It is well known that the standard Lanczos algortihm is numerically unstable, so that stabilization procedures must be employed [16,25]. We use a "double orthogonalization" stabilization technique to avoid instability. Our results suggest that, for discrete measures, this procedure is more accurate than all the alternatives, see Sect. 4.3.

3 PCL: A Hybrid Predictor–Corrector Lanczos Procedure

The main goal of this section is to describe a procedure by which we compute recurrence coefficients for μ of the form (4). The procedure entails knowledge of the continuous weights $\{w_j\}_{j=1}^C$ and their respective supporting intervals, $\{I_j\}_{j=1}^C$, along with the discrete part of the measure encoded by the nodes and weights $\left(\tau_j, \nu_j\right)_{j=1}^M$. In Sect. 3.2, we will also utilize the

singularity behavior of the weights w_j dictated by the constants α_j and β_j in (5) to compute moments.

Section 3.1 first introduces a new procedure to compute recurrence coefficients for a measure with a continuous density using polynomial moments. Section 3.2 then discusses our particular strategy for computing these moments. Finally, Sect. 3.3 introduces a procedure based on the multiple component approach in [8] for computing recurrence coefficients for a measure of general form (4).

3.1 PC: Predictor–Corrector Method

In this section, we describe a Stieltjes-like procedure for computing recurrence coefficients. Although this works for general measures, we are mainly interested in applying this technique for measures μ that have a continuous density. The high-level algorithm, like the previous ones we have discussed, is iterative. Suppose for some $n \geq 0$ we know the coefficient tableau,

$$\begin{array}{c} a_1(\mu)\ a_2(\mu)\ \cdots\ a_n(\mu) \\ b_0(\mu)\ b_1(\mu)\ b_2(\mu)\ \cdots\ b_n(\mu). \end{array}$$

These coefficients, via (1), define $p_0, \ldots, p_n$ that are orthonormal under a $\mathrm{d}\mu$-weighted integral. In order to compute a_{n+1} and b_{n+1}, we make educated guesses for these coefficients, and correct them using computed moments. The procedure is mathematically equivalent to the Stieltjes procedure: We define a new set of recurrence coefficients $\{\widetilde{a}_j, \widetilde{b}_j\}_{j=0}^{n+1}$, where

$$\widetilde{a}_j = a_j, \qquad \widetilde{b}_j = b_j,\ j = 0, \ldots, n, \tag{25a}$$

$$\widetilde{a}_{n+1} = a_n, \qquad \widetilde{b}_{n+1} = b_n, \tag{25b}$$

In particular, corrections $\Delta a_{n+1} \in \mathbb{R}$ and $\Delta b_{n+1} > 0$ exist such that

$$a_{n+1} = \widetilde{a}_{n+1} + \Delta a_{n+1}, \qquad b_{n+1} = \widetilde{b}_{n+1} \Delta b_{n+1}. \tag{25c}$$

Our procedure will compute the corrections Δa_{n+1} and Δb_{n+1}. The tableau of coefficients $\widetilde{a}_{n+1}$ and $\widetilde{b}_{n+1}$

$$\begin{array}{c} a_1(\mu)\ \cdots\ a_n(\mu)\ \widetilde{a}_{n+1}(\mu) \\ b_0(\mu)\ b_1(\mu)\ \cdots\ b_n(\mu)\ \widetilde{b}_{n+1}(\mu), \end{array}$$

can be used with (1) to generate the polynomials $p_0, \ldots, p_n$, along with $\widetilde{p}_{n+1}$, defined as

$$\widetilde{b}_{n+1} \widetilde{p}_{n+1} := (x - \widetilde{a}_{n+1}) p_n - b_n p_{n-1}. \tag{26}$$

Since $\widetilde{p}_{n+1}$ and p_{n+1} were generated using the same coefficients (a_j, b_j) up to index $j = n$, then they are both orthogonal to all polynomials of degree $n-1$ or less. However, $\widetilde{p}_{n+1}$ is not orthogonal to p_n in general. We can choose Δa_{n+1} to enforce this orthogonality, which requires computing a polynomial moment.

Once $a_{n+1} = \widetilde{a}_{n+1} + \Delta a_{n+1}$ is successfully computed, we can similarly define another degree-$(n+1)$ polynomial $\widehat{p}_{n+1}$ through the relation,

$$\widetilde{b}_{n+1} \hat{p}_{n+1} := (x - a_{n+1}) p_n - b_n p_{n-1}. \tag{27}$$

This polynomial differs from p_{n+1} by only a multiplicative constant, which can again be determined through a moment computation and used to compute Δb_{n+1}. We formalize the discussion above through the following result:

Lemma 1 *With $\widetilde{p}_{n+1}$ and $\hat{p}_{n+1}$ defined as in (26) and (27), respectively, let*

$$G_{n,n+1} := \int_{\mathbb{R}} p_n(x)\widetilde{p}_{n+1}(x)\mathrm{d}\mu(x), \tag{28a}$$

$$G_{n+1,n+1} := \int_{\mathbb{R}} \hat{p}^2_{n+1}(x)\mathrm{d}\mu(x), \tag{28b}$$

then,

$$\Delta a_{n+1} = G_{n,n+1}b_n, \qquad \Delta b_{n+1} = \sqrt{G_{n+1,n+1}}. \tag{29}$$

Proof Starting from the definition (26) for $\widetilde{p}_{n+1}$, we replace xp_n with the right-hand side of (1), yielding,

$$\begin{aligned}\widetilde{p}_{n+1} &= \Delta b_{n+1}\left[\frac{1}{b_{n+1}}(x-a_{n+1})\,p_n - b_n p_{n-1} + \Delta a_{n+1}\frac{1}{b_{n+1}}p_n\right] \\ &= \Delta b_{n+1}p_{n+1} + \frac{\Delta a_{n+1}\Delta b_{n+1}}{b_{n+1}}p_n. \end{aligned}\tag{30}$$

Thus, due to orthogonality of $\{p_j\}_{j\geq 0}$, we have

$$G_{n,n+1} = \int p_n(x)\widetilde{p}_{n+1}(x)\mathrm{d}\mu(x) \overset{(30)}{=} \frac{\Delta a_{n+1}\Delta b_{n+1}}{b_{n+1}} \overset{(25c)}{=} \frac{\Delta a_{n+1}}{b_n},$$

which shows the first relation in (29). To show the second relation, first we combine (1) and (27) to show,

$$\widetilde{b}_{n+1}\hat{p}_{n+1}(x) = (x-a_{n+1})p_n - b_n p_{n-1} = b_{n+1}p_{n+1},$$

so that

$$G_{n+1,n+1} = \int \hat{p}^2_{n+1}(x)\mathrm{d}\mu(x) = \left(\frac{b_{n+1}}{\widetilde{b}_{n+1}}\right)^2 \int p^2_{n+1}(x)\mathrm{d}\mu(x) = (\Delta b_{n+1})^2,$$

proving the second relation. □

The results (29) and (29) are the proposed approach: The moments $G_{n,n+1}$ and $G_{n+1,n+1}$ in (28a) and (28b) are polynomial moments that can be computed. We can subsequently use (29) and (25c) to compute the desired a_{n+1} and b_{n+1}.

The methodology of this section can then be iterated in order to compute as many recurrence coefficients a_n and b_n as desired. However, we must compute the $G_{n,n+1}$ and $G_{n+1,n+1}$ coefficients (which are similar to the moments required by the Stieltjes procedure). The main difference in our algorithm is that we use moments to compute $a_{n+1} - a_n$ and b_{n+1}/b_n that are typically close to 0 and 1, respectively, instead of simply a_n and b_n, which in general can be arbitrarily small or large numbers. We next summarize one particular strategy for computing these moments assuming that a type of characterization of μ is available.

3.2 Computation of Polynomial Moments

The previous section shows that we can compute recurrence coefficients for the measure μ if we can compute some of its moments, in particular $G_{n,n+1}$ and $G_{n+1,n+1}$. We briefly describe

in this section how we compute moments for measures of the form (4) with knowledge of the singularity behavior in (5). The moment of a polynomial q for μ can be written as

$$\int q(x)\mathrm{d}\mu(x) = \sum_{j=1}^{C}\int_{I_j} q(x)w_j(x)\mathrm{d}x + \sum_{j=1}^{M} v_j q(\tau_j),$$

so that the only difficult part is to compute $\int_{I_j} q(x)w_j(x)\mathrm{d}x$ for each j.

Suppose first that I_j is compact, i.e., that $I_j = [\ell, r]$ for finite ℓ, r. Then we rewrite the integral as

$$\int_{I_j} q(x)w_j(x)\mathrm{d}x = \frac{r-\ell}{2}\int_{-1}^{1} q(A(u))w_j(A(u))\mathrm{d}u, \quad A(u) := \left(\frac{r-\ell}{2}\right)u + \frac{r+\ell}{2}.$$

w_j obeying the limiting conditions (5) with constants α_j, β_j implies that $w_j(A(u))$ behaves like $(1-u^{\alpha_j})$ near $u = 1$, and like $(1+u^{\beta_j})$ near $u = -1$. When $\alpha_j = \beta_j = 0$, then a global $\mathrm{d}x$-Gaussian quadrature rule will be efficient in evaluating this integral, but the accuracy will suffer when either constant differs from 0. To address this problem, we can further rewrite the integral as:

$$\int_{I_j} q(x)w_j(x)\mathrm{d}x = \frac{r-\ell}{2}\int_{-1}^{1} q(A(u))\omega_j(u)\mathrm{d}\mu^{(\alpha_j,\beta_j)}(u),$$

where $\mu^{(\alpha_j,\beta_j)}$ is a Jacobi measure on $[-1, 1]$, and ω_j is w_j multiplied by the appropriate factors,

$$\mathrm{d}\mu^{(\alpha_j,\beta_j)}(u) = (1-u)^{\alpha_j}(1+u)^{\beta_j}\mathrm{d}x, \quad \omega_j(u) := w_j(A(u))(1-u)^{-\alpha_j}(1+u)^{-\beta_j}.$$

The advantage of this formulation is that ω_j is now smooth at the boundaries $u = \pm 1$, and if in addition it is smooth on the interior of $[-1, 1]$, then a Jacobi (α_j, β_j)-Gaussian quadrature rule will efficiently evaluate the integral. Therefore, if $(u_k, \lambda_k)_{k=1}^{K}$ is a K-point Jacobi (α_j, β_j)-Gaussian quadrature rule, we approximate the integral as

$$\int_{I_j} q(x)w_j(x)\mathrm{d}x \approx \sum_{k=1}^{K}\lambda_k\omega_j(u_k)q(A(u_k)),$$

where the nodes and weights can be computed through the spectrum of $J_K(\mu^{(\alpha_j,\beta_j)})$ since the recurrence coefficients of these measures are explicitly known. In particular, all the quadrature nodes u_k lie interior to $[-1, 1]$, so that the above procedure does *not* require evaluation of ω_j at $u = \pm 1$. We adaptively choose K, i.e., increasing K until the difference between approximations is sufficiently small.

3.3 PCL: A Hybrid Predictor–Corrector Lanczos Method

The full procedure we describe in this section combines the strategies in Sects. 3.1 and 3.2, along with the (stabilized) Lanczos procedure in Sect. 2.6. Assuming that we *a proiri* know that the first N recurrence coefficients $\{a_n, b_n\}_{n=0}^{N-1}$ are required for μ, then the main idea here is to construct a fully discrete measure ν whose moments up to degree $2N-2$ match those of μ.

We accomplish this as follows: Recall that the continuous densities $\{w_j\}_{j=1}^{C}$ of the measure μ in (4) are known, along with their boundary singularity behavior in (5). Then for each j,

the PC procedure in Sects. 3.1 and 3.2 can be used to compute the first $N+1$ recurrence coefficients for w_j, $\{a_{j,n}, b_{j,n}\}_{n=0}^{N}$. Using these recurrence coefficients, an N-point Gaussian quadrature rule $(x_{j,k}, \lambda_{j,k})_{k=1}^{N}$ can be computed that exactly integrates all polynomials up to degree $2N-1$ with respect to the weight w_j:

$$\int_{I_j} q(x) w_j(x) \mathrm{d}x = \sum_{k=1}^{N} \lambda_{j,k} q\left(x_{j,k}\right), \qquad \deg q \leq 2N-1.$$

After this quadrature rule is computed for every $j = 1, \ldots, C$, the discrete measure ν, defined as

$$\nu := \sum_{j=1}^{C} \sum_{k=1}^{N} \lambda_{j,k} \delta_{x_{j,k}} + \sum_{j=1}^{M} \nu_j \delta_{\tau_j}, \tag{31}$$

and has moments that match those of μ up to degree $2N-1$. Once this procedure is completed, we employ the Lanczos procedure in Sect. 2.6 to compute the first N recurrence coefficients for ν, which equal those for μ. The main reason we employ the Lanczos scheme (as opposed to any other approach) is that, for discrete measures, the Lanczos procedure appears more empirically stable than all other procedures we consider, cf. Sect. 4.5.

Note that if $C = 1$ and $M = 0$, then the Lanczos procedure is not needed at all since $(a_{1,n}, b_{1,n})_{n=0}^{N-1}$ are the desired coefficients, and if $C = 0$, then only the Lanczos procedure need be queried since no quadrature is required.

The above is essentially a complete description of the PCL algorithm. However, we include one additional adaptive procedure to ensure correct computation of the moments. Let $\{N_s\}_{s\geq 0}$ be an increasing sequence of positive integers. A strategy for determining the sequence of N_s can be found in [11,12],

$$\begin{aligned} N_0 &= N, & N_s &= N_{s-1} + \Delta_s, & s &= 1, 2, \ldots, \\ \Delta_1 &= 1, & \Delta_s &= 2^{\lfloor \frac{s}{5} \rfloor} N, & s &= 2, 3, \ldots . \end{aligned}$$

We define ν_s as the measure (31) with $N \leftarrow N_s$. We use PCL to compute numerical approximations $\{a_n^{[s]}, b_n^{[s]}\}_{n\geq 0}$ to the recurrence coefficients for ν_s. (I.e., we use PC to compute the N_s-point quadrature rule $(x_{j,k}, \lambda_{j,k})_{k=1}^{N_s}$ and subsequently use LZ to compute the recurrence coefficients for ν_s.) With the (approximate) coefficients for ν_s and ν_{s-1}, if the condition

$$\left| b_n^{[s]} - b_n^{[s-1]} \right| \leq \epsilon |b_n^{[s]}|, \qquad n = 0, 1, \ldots, N-1,$$

is satisfied, then we return the computed coefficients for ν_s. Otherwise, we set $s \leftarrow s+1$ and test the condition above again. This adaptive procedure is similar to those employed in [11,12]. In our computations we set $\epsilon = 10^{-12}$, and we set an upper limit of N_s as $N_s^{\max} = 10N$ for all s, which will usually be satisfactory.

4 Numerical Experiments

We now present numerical examples to illustrate the performance of our algorithm by computing the first N three-term recurrence coefficients for different types of measures μ. Our results will consider all the algorithms in Table 1: the first six in Sect. 2 and the last two new procedures proposed in Sect. 3. We implement all the algorithms in Python. All the

computations are carried out on a MacBook Pro laptop with a 3.1 GHz Intel(R) Core(TM) i5 processor and 8 GB of RAM.

Examples can be classified according to whether we have a way to compute the exact recurrence coefficients. When this is the case, we define $\{\hat{a}_n, \hat{b}_n\}_{n=0}^{N-1}$ as the first N exact coefficients and $\{a_n, b_n\}_{n=0}^{N-1}$ as coefficients that are computed from any particular algorithm. The error e_N can be denoted by an ℓ^2-type norm,

$$e_N = \left(\sum_{n=0}^{N-1} \left[(a_n - \hat{a}_n)^2 + \left(b_n - \hat{b}_n \right)^2 \right] \right)^{\frac{1}{2}}. \tag{32}$$

If the exact coefficients are not available, we consider another error metric. If $\{p_n(x)\}_{n=0}^{N-1}$ is a polynomial basis produced through the three-term recurrence (1) using the computed coefficients by $\{a_n, b_n\}_{n=0}^{N-1}$, then let $\boldsymbol{A}$ be an $N \times N$ matrix with entries

$$(\boldsymbol{A})_{m,n} = \int_{\mathbb{R}} p_{n-1}(x) p_{m-1}(x) \mathrm{d}\mu(x), \qquad n, m = 1, \ldots, N,$$

which equals $\delta_{n,m}$ if $\hat{a}_n = a_n$ and $\hat{b}_n = b_n$. The new error indicator f_N we compute is

$$f_N = \| \boldsymbol{A} - \boldsymbol{I} \|_F, \tag{33}$$

where $\| \cdot \|_F$ is the Frobenius norm on matrices and $\boldsymbol{I}$ is the $N \times N$ identity matrix.

The computational timing results that measure efficiency are averaged over 100 runs of any particular algorithm.

4.1 Freud Weights

One computational strategy for determining the recurrence coefficients for Freud weights of the form (3) on the entire real line is to use the ("non-modified") Chebyshev algorithm, which requires monomial moments and employs a recurrence similar to (22). The monomial moments of (3) are explicitly computable as simple evaluations of the Euler Gamma function, but numerical instabilities typically develop in such an approach due to roundoff error; to combat this limitation, computations may be completed in variable precision arithmetic, resulting in a procedure that correctly computes the recurrence coefficients [14]. In this section, we use this VPA procedure to generate recurrence coefficients treated as "exact" for use in computing errors. In particular, we employ the `sr_freud.m` routine from [6] that utilizes variable-precision arithmetic in Matlab [17].

We compute recurrence coefficients using the `DP`, `HD`, `aPC`, and `MC` methods for Freud exponents $\alpha = 4, 6$. The `DP` recursion for each of the two cases is simple, given by (9) and (11), respectively. For the `MC` method, we use Hermite orthogonal family for q_k in (20) that is orthogonal with respect to λ. The top two plots in Fig. 1 show that each of these methods is not computationally useful since instabilities develop quickly. In contrast, both the `SP` and `PC` approaches can effectively compute recurrence coefficients, which we show in the bottom two plots of Fig. 1. In terms of efficiency, Table 2 illustrates that the "exact" VPA procedure is several orders of magnitude more expensive than all other approaches, and that `SP` and `PC` are competitive. Code that reproduces this example is available in the routine `ex_freud_4.py` and `ex_freud_6.py` from [18].

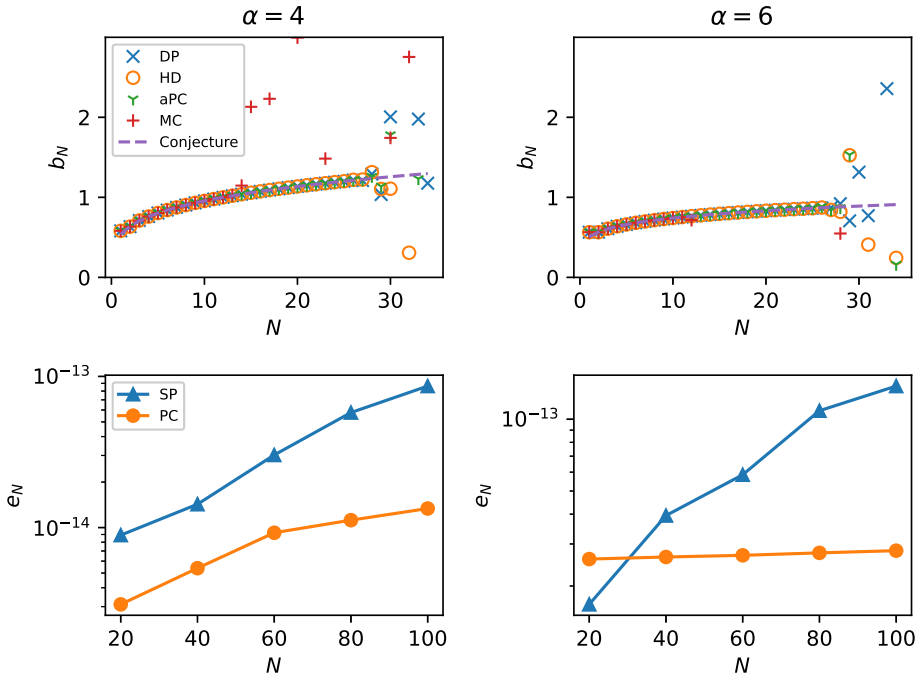

Fig. 1 Example for Sect. 2.1: the top two plots are recursion coefficients b_N computed by DP, HD, aPC, MC and Freud conjecture in (10) and (12). The two plots at the bottom show errors e_N of SP and PC

Table 2 Example for Sect. 2.1: elapsed time (s) for Freud weight when $\alpha = 4$ (subcolumns on the left) and $\alpha = 6$ (subcolumns on the right)

Method	$N = 20$		$N = 40$		$N = 60$		$N = 80$		$N = 100$	
VPA	18.86	19.13	99.38	101.10	293.44	300.41	631.20	633.20	1196.29	1362.86
SP	0.24	0.21	0.75	0.63	1.60	1.32	2.72	2.24	4.12	3.42
PC	0.25	0.22	0.75	0.65	1.60	1.34	2.72	2.27	4.12	3.40

4.2 Piecewise Smooth Weight

We consider the measure $\mathrm{d}\mu(x) = w(x)\mathrm{d}x$ on $[-1, 1]$, where

$$\omega(x) = \begin{cases} |\, x \,|^{\gamma}\ (x^2 - \xi^2)^p (1 - t^2)^q, & x \in [-1, -\xi] \cup [\xi, 1] \\ 0, elsewhere, & 0 < \xi < 1, p > -1, q > -1, \gamma \in \mathbb{R}. \end{cases}$$

For certain choices of γ, p, q, there is theory regarding the resulting orthogonal polynomials [1], and such weights arise in applications [34]. In the special cases $\gamma = \pm 1$, $p = q = \pm 1/2$, closed-form representations for the recurrence coefficients can be computed [9]. For example, the exact formula for the recurrence coefficients for the case $\gamma = 1$, $p = q = -1/2$, $\eta = (1 - \xi)/(1 + \xi)$ is given by

$$\hat{b}_0 = \sqrt{\pi}, \qquad \hat{b}_1 = \sqrt{\frac{1 + \xi^2}{2}},$$

$$\hat{b}_{2n} = \sqrt{\frac{(1-\xi^2)(1+\eta^{2n-2})}{4(1+\eta^{2n})}}, \qquad \hat{b}_{2n+1} = \sqrt{\frac{(1+\xi^2)(1+\eta^{2n+2})}{4(1+\eta^{2n})}}, \qquad n \in \mathbb{N},$$

with $\hat{a}_n = 0$ for all n.

A Legendre orthogonal family for q_k in (20) that is orthogonal with respect to λ is chosen for the `MC` method. For the choice $\gamma = 1$, $p = q = -1/2$ and $\xi = 1/10$, Table 3 illustrates the accuracy and cost of the algorithms `HD`, `aPC`, `MC`, `SP`, and `PC`. We observe that only the `SP` and `PC` approaches yield reasonable accuracy, with `PC` being slightly more accurate. We omit results for other choices of (γ, p, q), which produce nearly identical results. The results from this table can be produced from `ex_pws.py` in [18].

4.3 Transformed Discrete Chebyshev

In the previous example, we compute the recurrence coefficients of "continuous" orthogonal polynomials with respect to μ on bounded or unbounded supports. We now consider the support of μ that consists of a discrete set of points.

Given a positive number M, we define the nodes $\tau_j = (j-1)/M$ and $v_j = 1/M$ for $j = 1, 2, \ldots, M$. Then, the transformed discrete Chebyshev [12, Example 2.26] measure is given as

$$\mathrm{d}\mu(x) = \sum_{j=1}^{M} \frac{1}{M} \delta_{\frac{j-1}{M}} \mathrm{d}x, \qquad j = 1, 2, \ldots, M,$$

i.e., an equally spaced and equally weighted discrete measure on $[0, 1)$. The recurrence coefficients are known explicitly if a linear transformation of variables is applied to the discrete Chebyshev measure with canonical support points [12, Section 1.5.2]. For a given size of supports, M, with $\hat{b}_0 = 1$,

$$\hat{a}_n = \frac{M-1}{2M}, \qquad \hat{b}_n = \sqrt{\frac{1-(\frac{n}{M})^2}{4(4-(\frac{n}{M})^2)}}, \qquad n = 1, 2, \ldots, M-1.$$

In Fig. 2, the methods `HD`, `aPC` and `MC` are omitted since their instabilities develop very quickly. An NaN value appears when the required number of recurrence coefficients, N, is less than 20. We compare the `SP`, `LZ` and `PC` approaches on measure support sizes $M = 40, 80, 160, 320$. We observe that the `LZ` approach is effective for all choices of M, and when N is comparable to M, the `SP` and `PC` approaches become inaccurate. The lower two plots of Fig. 2 show that when M is notably larger than N, all three approaches produce good results. In particular, all the numerical results in this subsection are produced by `ex_discrete_cheb.py` in [18].

4.4 Discrete Probability Density Function

High-dimensional integration is a common problem in scientific computing arising from, for example, the need to estimate expectations in uncertainty quantification [27,30]. Many integrands for such integrals found in scientific computing applications map a large number of input variables to an output quantity of interest, but admit low-dimensional ridge structure that can be exploited to accelerate integration. A ridge function [24] is a function $f : \mathbb{R}^m \to \mathbb{R}$

Table 3 Example for Sect. 4.2: errors e_N (subcolumns on the left) and elapsed time (s) (subcolumns on the right) when $\gamma = 1$, $p = q = -1/2$

Method	$N = 20$		$N = 40$		$N = 60$		$N = 80$		$N = 100$	
HD	6.05e–02	0.003	—	—	—	—	—	—	—	—
aPC	6.05e–02	0.001	—	—	—	—	—	—	—	—
MC	2.34e–15	0.001	1.00e+00	0.006	—	—	—	—	—	—
SP	4.73e–14	0.10	2.85e–13	0.28	3.85e–13	0.57	3.99e–13	0.93	4.62e–13	1.39
PC	9.08e–15	0.10	1.80e–14	0.29	3.13e–14	0.57	5.14e–14	0.94	7.27e–14	1.40

Here — means a NaN value due to the numerical overflow from the instability of the corresponding method

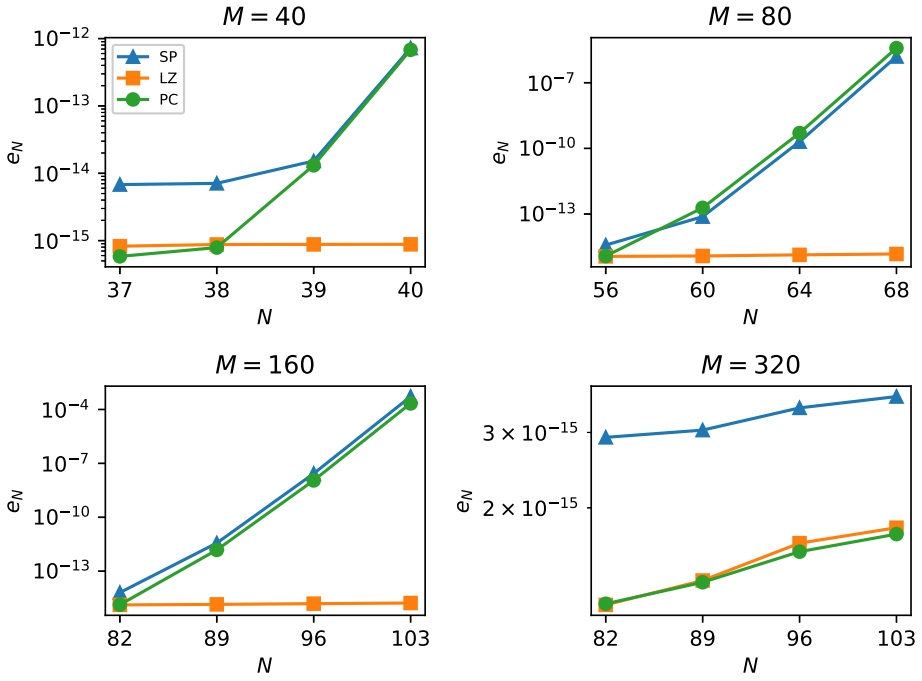

Fig. 2 Example for Sect. 4.3: the first three plots compute errors e_N for different N portion of distinct M and the last two plots for the same N but for distinct M

of the form

$$f(\boldsymbol{x}) = g(\boldsymbol{a}^T\boldsymbol{x}),$$

where $\boldsymbol{a} \in \mathbb{R}^m$ is a constant vector called the ridge direction and $g : \mathbb{R} \to \mathbb{R}$ is the ridge profile. For such functions, we clearly have that f depends only on a scalar variable $y := \boldsymbol{a}^T\boldsymbol{x}$. In applications, we frequently wish to integrate f with respect to some m-dimensional probability measure ρ on $\boldsymbol{x}$, which can be simplified by integrating over the scalar variable y with respect to the univariate measure μ that is the push-forward of ρ under the map $\boldsymbol{x} \mapsto \boldsymbol{a}^T\boldsymbol{x}$. Thus, the goal is to compute recurrence coefficients for μ.

In practice the multivariate measure ρ is known, but computing the univariate measure μ exactly is typically not feasible. However, an approximation to μ can be furnished using the procedure in [15, Section 2.2] that randomly generates M i.i.d. samples $\{\boldsymbol{x}_j\}_{j=1}^M$ from ρ, and defines μ as a discrete measure supported on the projection of these samples onto the real line:

$$\mathrm{d}\mu(x) = \sum_{j=1}^{M} \frac{1}{M}\delta_{\tau_j}\mathrm{d}x, \qquad \tau_j := \boldsymbol{a}^T\boldsymbol{x}_j.$$

To compute quadrature rules with respect to this measure, we take ρ as the uniform measure on the m-dimensional hypercube $[-1, 1]^m$. Let $m = 25$, and $\boldsymbol{a} \in \mathbb{R}^{25}$ is chosen randomly. We then test for $M = 100, 300$.

Since we do not have an expression for the exact recurrence coefficients, we measure errors using the metric f_N in (33). As shown in Table 4, the computed recursion coefficients

Table 4 Example for Sect. 4.4: errors f_N when $M = 100$ (subcolumns on the left) and $M = 300$ (subcolumns on the right)

Method	$N = 20$		$N = 40$		$N = 60$		$N = 80$		$N = 100$	
HD	1.69e–07	1.24e–07	—	—	—	—	—	—	—	—
aPC	7.72e–08	3.80e–08	2.03e+05	7.67e+05	1.35e+27	3.90e+25	5.85e+47	2.71e+55	4.75e+67	9.70e+72
MC	3.02e–09	3.60e–09	—	—	—	—	—	—	—	—
SP	2.74e–15	3.39e–15	9.26e–15	8.53e–15	5.94e–10	2.62e–14	4.00e+00	5.20e–14	7.48e+00	9.37e–14
LZ	4.75e–15	3.87e–15	2.95e–14	1.10e–14	3.45e–09	1.73e–14	1.86e+68	3.38e–14	2.50e+68	9.29e–14
PC	3.96e–15	4.54e–15	1.17e–14	9.57e–15	1.03e–09	1.50e–14	4.00e+00	2.47e–12	7.48e+00	1.41e–13

Here — means a NaN value due to the numerical overflow from the instability of the corresponding method

are not as accurate when N is closer to M, no matter what method is used. However, the methods SP, LZ and PC all perform better when M is large enough.Code that reproduces this example is available in the routine ex_discrete_convolution.py in [18].

4.5 Multiple Component: Chebyshev Weight Function Plus a Discrete Measure

The measure to be considered is the normalized Jacobi weight function on $[-1, 1]$ with a discrete M-point measure added to it,

$$\mathrm{d}\mu(x) = (\beta_0^J)^{-1}(1-x)^\alpha(1+x)^\beta \mathrm{d}x + \sum_{j=1}^{M} \nu_j \delta_{\tau_j}\mathrm{d}x, \qquad \alpha, \beta > -1, \qquad \nu_j > 0, \tag{34}$$

where $\beta_0^J = \int_{-1}^{1}(1-x)^\alpha(1+x)^\beta \mathrm{d}x$. The orthogonal polynomials belonging to the measure (34) are explicitly known only in very special cases. The case of one mass point at one end point, that is, $M = 1, \tau_1 = -1$, has been studied and the recurrence coefficients can be computed with rather technical formulas [3,11]. The exact recursion coefficients for $N = 1, 7, 18, 40$ are given in [12, Table 2.11]. For each of these particular N, we compute the fixed-N error, donated by $e_N^f = \left(\left(a_N - \hat{a}_N\right)^2 + \left(b_N - \hat{b}_N\right)^2\right)^{1/2}$.

Table 5 shows results for the HD, aPC, MC, SP, and PC approaches for the measure μ above. In addition, we compute results using the LZ approach; note that the LZ approach cannot directly be utilized on the measure (34) since this measure has an infinite number of support points. Instead, the LZ results shown in Table 5 first use the discretization approach as described in Sect. 2.6, which replaces the continuous part of μ with a discrete Gaussian quadrature measure. The reason we include this test in Table 5 is that it motivates the PCL algorithm: if one can discretize measures, then the LZ approach is frequently more accurate than alternative methods.

We generate the first 40 recursion coefficients for $\alpha = -0.6, \beta = 0.4$ of the Jacobi parameters in two cases: one mass at $\tau_1 = -1$ with strength $\nu_1 = 0.5$ and a single mass point of strength $\nu_1 = 1$ at $\tau_1 = 2$. The results, produced by routine ex_multi_component.py from [18], are shown in Table 5. SP, LZ, PC and even MC produce essentially identical results within machine precision in the first case. However, matters change significantly when a mass point is placed outside $[-1, 1]$, regardless of whether or not the other mass points on $[-1, 1]$ are retained [12, Example 2.39]. SP and PC become extremely unstable; this empirical superiority of the LZ approach for discrete measures is the reason why the last step of the PCL algorithm in Sect. 3.3 is to utilize the Lanczos algorithm.

4.6 General Multiple Component: Continuous Weight Function Plus a Discrete Measure

In the previous example, we studied the case of a combination of Chebyshev weight and discrete measure. A quadrature for Chebyshev is trivial because it is one of the classical weights so that we can obtain the quadrature by known recursion coefficients. However, if the continuous weight is not of classical form, then we employ the PCL algorithm in Sect. 3.3: We use PC to compute recursion coefficients, leading to Gaussian quadrature nodes and weights for the continuous part, which is then combined with the discrete part as input to the LZ algorithm.

Table 5 Example for Sect. 4.5: errors e_N^f with one mass at $\tau_1 = -1$ with $\nu_1 = 0.5$ (subcolumns on the left) and $\tau_1 = 2$ with $\nu_1 = 1$ (subcolumns on the right)

Method	$N = 1$		$N = 7$		$N = 18$		$N = 40$	
HD	3.71e–14	2.22e–11	3.64e–12	1.81e–09	1.72e–04	—	—	—
aPC	3.71e–14	2.22e–11	3.54e–12	1.81e–09	1.67e–04	—	—	—
MC	3.71e–14	2.22e–11	3.63e–12	8.90e–11	3.02e–12	2.90e+00	3.87e–12	1.84e+00
SP	3.71e–14	2.22e–11	3.63e–12	5.44e–13	3.03e–12	3.80e–12	3.90e–12	2.48e–06
LZ	3.70e–14	2.22e–11	3.63e–12	5.44e–13	3.03e–12	3.80e–12	3.90e–12	2.10e–12
PC	3.71e–14	2.22e–11	3.63e–12	5.44e–13	3.02e–12	3.80e–12	3.90e–12	2.49e–06

Here — means a NaN value due to the numerical overflow from the instability of the corresponding method

Table 6 Example for Sect. 4.6: errors f_N by procedure in 3.3 with $N_s = N$ for all s (subcolumns on the left) and by PCL, i.e. with a adaptive procedure (subcolumns on the right) when $M = 20, 40, 80, 160, 320$

M	$N = 20$		$N = 40$		$N = 60$		$N = 80$		$N = 100$	
20	1.09e–14	7.47e–15	6.48e–14	1.63e–14	1.46e–10	6.61e–13	2.41e–03	5.63e–12	1.66e+07	3.27e–09
40	6.50e–15	1.05e–14	2.50e–14	3.28e–14	9.34e–11	9.52e–14	8.54e–03	1.84e–13	1.95e+09	3.05e–11
80	8.80e–15	5.11e–15	1.39e–14	4.74e–14	1.68e–11	3.90e–14	4.48e–03	8.97e–14	5.10e+08	4.95e–11
160	7.73e–15	7.13e–15	1.43e–14	3.99e–14	2.90e–11	7.03e–14	1.88e–03	1.24e–13	2.34e+09	2.25e–11
320	7.24e–15	8.39e–15	1.98e–14	1.68e–14	3.80e–11	3.86e–14	6.82e–03	6.65e–14	9.63e+08	7.14e–11

We consider the positive half-range Hermite measure plus a transformed discrete Chebyshev measure defined on $(-1, 0]$,

$$\mathrm{d}\mu(x) = e^{-x^2} + \sum_{j=1}^{M} v_j \delta_{\tau_j} \mathrm{d}x, \qquad \tau_j := -\frac{j-1}{M}, \qquad v_j := \frac{1}{M}.$$

Using the `PCL` algorithm, for $M = 20, 40, 80, 160, 320$, we generate the first 100 recursion coefficients. Table 6 shows that the coefficients are more accurate when an adaptive procedure is applied to determine N_s, no matter what M is. The results here are produced by routine `ex_gmulti_component.py` in [18].

5 Summary and Extensions

In this paper, we summarize several existing numerical methods for computing these recurrence coefficients associated to measures for which explicit formulas are not available. We propose a novel "predictor–corrector" algorithm and study the accuracy and efficiency by comparing with existing methods for fairly general measures. The method makes predictions for the next coefficients and correct them iteratively. Finally, we introduce a hybrid algorithm that combines the "predictor–corrector" algorithm and the (stabilized) Lanczos procedure. It can be used to compute recurrence coefficients for a general measure with multiple continuous and discrete components.

The predictor–corrector algorithm outperforms many other methods and is competitive with the Stieltjes procedure when a continuous measure is given. For a discrete measure, it can compute accurate coefficients only when the discrete support M is large enough. However, the (stabilized) Lanczos procedure requires empirically appears to be superior for discrete measures. Based on this observation, we propose a "predictor–corrector-Lanczos" algorithm is that is a hybrid of the predictor–corrector and Lanczos schemes, and applies to a fairly general class of measures.

We focus on the computation of recurrence coefficients for univariate orthogonal polynomial families. Thus, a natural extension of this work would be to adapt the approaches to address the same problem for multivariate polynomials, for which the formulations can be substantially more complex. Such investigations are the focus of ongoing work.

Acknowledgements This work was supported by the National Institute of Biomedical Imaging and Bioengineering of the National Institutes of Health under grant number U24EB029012, and under National Science Foundation awards DMS-1720416 and DMS-1848508. This material is based upon work supported by both the National Science Foundation under Grant No. DMS-1439786 and the Simons Foundation Institute Grant Award ID 507536 while A. Narayan was in residence at the Institute for Computational and Experimental Research in Mathematics in Providence, RI, during the Spring 2020 semester

Declarations

Conflicts of interest The authors have no conflicts of interest to declare that are relevant to the content of this article.

References

1. Barkov, G.I.: Some systems of polynomials orthogonal in two symmetric intervals. Izvestiya Vysshikh Uchebnykh Zavedenii. Matematika **4**, 3–16 (1960)
2. Chebyshev, P..L.: Sur l'interpolation par la méthode des moindres carrés. Mémoires de lÁcadémie Impériale des sciences de St.-Pétersbourg **1**(15), 1–24 (1859)
3. Chihara, T.S.: An introduction to orthogonal polynomials. Courier Corporation (2011)
4. Freud, G.: Orthogonal Polynomials. Pergamon Press (1971)
5. Freud, G.: On the coefficients in the recursion formulae of orthogonal polynomials. In: Proceedings of the Royal Irish Academy. Section A: Mathematical and Physical Sciences, pp. 1–6. JSTOR (1976)
6. Gautschi, W.: https://www.cs.purdue.edu/archives/2002/wxg/codes/sr_freud.m
7. Gautschi, W.: A survey of gauss-christoffel quadrature formulae, em "eb christoffel-the influence of his work in mathematics and physical sciences"(pl butzer e f. fehér, eds.) pp. 72-147 (1981). https://doi.org/10.1007/978-3-0348-5452-8_6
8. Gautschi, W.: On generating orthogonal polynomials. SIAM J. Sci. Stat. Comput. **3**(3), 289–317 (1982). https://doi.org/10.1137/0903018
9. Gautschi, W.: On some orthogonal polynomials of interest in theoretical chemistry. BIT Numer. Math. **24**(4), 473–483 (1984)
10. Gautschi, W.: Algorithm 726: ORTHPOL—a package of routines for generating orthogonal polynomials and Gauss-type quadrature rules. ACM Trans. Math. Softw. **20**(1), 21–62 (1994). https://doi.org/10.1145/174603.174605
11. Gautschi, W.: Algorithm 726: Orthpol-a package of routines for generating orthogonal polynomials and gauss-type quadrature rules. ACM Trans. Math. Softw. (TOMS) **20**(1), 21–62 (1994)
12. Gautschi, W.: Orthogonal Polynomials: Computation and Approximation. Oxford University Press, USA (2004)
13. Gautschi, W.: Orthogonal polynomials, quadrature, and approximation: computational methods and software (in Matlab). In: Marcellán, F., Assche, W. V. (eds.) Orthogonal Polynomials and Special Functions, no. 1883 in Lecture Notes in Mathematics, pp. 1–77. Springer, Heidelberg (2006). https://doi.org/10.1007/978-3-540-36716-1_1
14. Gautschi, W.: Variable-precision recurrence coefficients for nonstandard orthogonal polynomials. Numer. Algorithms **52**(3), 409–418 (2009)
15. Glaws, A., Constantine, P.G.: Gaussian quadrature and polynomial approximation for one-dimensional ridge functions. SIAM J. Sci. Comput. **41**(5), S106–S128 (2019)
16. Gragg, W.B., Harrod, W.J.: The numerically stable reconstruction of jacobi matrices from spectral data. Numerische Mathematik **44**(3), 317–335 (1984)
17. https://www.mathworks.com/help/symbolic/vpa.html
18. https://github.com/ZEXINLIU/Univariate_ttr_examples
19. Lew, J.S., Quarles, D.A., Jr.: Nonnegative solutions of a nonlinear recurrence. J. Approx. Theory **38**(4), 357–379 (1983)
20. Lubinsky, D.S., Mhaskar, H.N., Saff, E.B.: A proof of Freud's conjecture for exponential weights. Constr. Approx. **4**(1), 65–83 (1988). https://doi.org/10.1007/BF02075448
21. Magnus, A.P.: Freud's equations for orthogonal polynomials as discrete painlevé equations. arXiv:math/9611218 pp. 7–8 (1996)
22. Nevai, P.G.: Orthogonal Polynomials. American Mathematical Society (1980)
23. Oladyshkin, S., Nowak, W.: Data-driven uncertainty quantification using the arbitrary polynomial chaos expansion. Reliab. Eng. Syst. Saf. **106**, 179–190 (2012). https://doi.org/10.1016/j.ress.2012.05.002
24. Pinkus, A.: Ridge Functions, vol. 205. Cambridge University Press (2015)
25. Rutishauser, H.: On Jacobi rotation patterns. Proc. Symp. Appl. Math. **15**, 219–239 (1963)
26. Sack, R.A., Donovan, A.F.: An algorithm for Gaussian quadrature given modified moments. Numerische Mathematik **18**(5), 465–478 (1971). https://doi.org/10.1007/BF01406683
27. Smith, R..C.: Uncertainty Quantification: Theory, Implementation, and Applications, vol. 12. Siam (2013)
28. Stieltjes, T.J.: Quelques recherches sur la théorie des quadratures dites mécaniques. Annales scientifiques de l'École Normale Supérieure **1**, 409–426 (1884)
29. Stieltjes, T.J.: Some research on the theory of so-called mechanical quadratures. Scientific annals of the 'Ecole Normale Sup é rieure **1**, 409–426 (1884)
30. Sullivan, T.J.: Introduction to Uncertainty Quantification, vol. 63. Springer (2015)
31. Szegö, G.: Orthogonal Polynomials, 4th edn. American Mathematical Soc (1975)
32. Van Assche, W.: Discrete Painlevé equations for recurrence coefficients of orthogonal polynomials. In: Difference Equations, Special Functions and Orthogonal Polynomials, pp. 687–725. World Scientific (2007)

33. Wheeler, J.C.: Modified moments and Gaussian quadratures. Rocky Mountain J. Math. **4**(2), 287–296 (1974). https://doi.org/10.1216/RMJ-1974-4-2-287
34. Wheeler, J.C.: Modified moments and continued fraction coefficients for the diatomic linear chain. J. Chem. Phys. **80**(1), 472–476 (1984)
35. Wiener, N.: The homogeneous chaos. Am. J. Math. **60**(4), 897–936 (1938)
36. Witteveen, J.A., Bijl, H.: Modeling arbitrary uncertainties using Gram–Schmidt polynomial chaos. In: 44th AIAA Aerospace Sciences Meeting and Exhibit, p. 896 (2006)
37. Witteveen, J.A., Sarkar, S., Bijl, H.: Modeling physical uncertainties in dynamic stall induced fluid-structure interaction of turbine blades using arbitrary polynomial chaos. Comput. Struct. **85**(11–14), 866–878 (2007)
38. Xiu, D., Karniadakis, G.E.: The Wiener–Askey polynomial chaos for stochastic differential equations. SIAM J. Sci. Comput. **24**(2), 619–644 (2002). https://doi.org/10.1137/S1064827501387826