AIMS Mathematics, 10(8): 18861–18886.
DOI:10.3934/math.2025843
Received: 16 May 2025
Revised: 28 July 2025
Accepted: 08 August 2025
Published: 20 August 2025

https://www.aimspress.com/journal/Math

Research article

A machine-learning approach to weight approximation for a new family of orthogonal polynomials

Varun Kumar[1], K. Laxminarayanamma[2], Abhishek Kumar Singh[3], Brajesh Shukla[4] and Saiful Rahman Mondal[5,*]

[1] Department of Pure and Applied Mathematics, Alliance School of Sciences, Alliance University, Bangalore 562106, Karnataka, India

[2] CSE Department, Institute of Aeronautical Engineering, Dundigal Hyderabad-500043, Telangana, India

[3] Department of Mathematics, Patna Science College, Patna University, Patna 800005, Bihar, India

[4] Department of Computer Science Engineering, SRM University Delhi–NCR, Sonepat, 131029, Haryana, India

[5] Department of Mathematics and Statistics, College of Science, King Faisal University, P. O. Box 400, Al-Ahsa 31982, Saudi Arabia

* **Correspondence:** Email: smondal@kfu.edu.sa.

Abstract: This research introduces a novel two-parameter family of orthogonal polynomials that emerge as solutions to a doubly confluent Heun-type differential equation. We investigate these polynomials, examining their geometric properties and analyzing the behavior and distribution of their zeros under varying parameter conditions. Leveraging machine learning techniques, we successfully derive symbolic expressions for the corresponding weight functions associated with these orthogonal polynomials. Our numerical results demonstrate the efficacy of this approach, achieving a maximum absolute error of order 10^{-4} in weight function approximation. Furthermore, we present a comparison between our proposed model and conventional approximation methods, including cubic spline interpolation and Lagrange polynomial interpolation, highlighting the advantages of our methodology.

Keywords: Heun equation; recurrence relation; orthogonal polynomials; zeros; machine learning; symbolic regression; weight approximation
Mathematics Subject Classification: 33C45, 34A30, 65H10, 68T20

1. Introduction

Differential equations (DEs) are the cornerstone of modeling dynamic systems across science and engineering. Their solutions describe how quantities evolve over time/space (e.g., heat diffusion, population dynamics), engineer systems like aircraft (Navier–Stokes equations) or circuits (Kirchhoff's laws), and encode physical principles (e.g., Newton's laws, Maxwell's equations, Schrodinger's equation). Representing DE solutions as series expansions in terms of orthogonal polynomials (e.g., Hermite, Legendre) or special functions (e.g., Bessel, hypergeometric) offers analytical tractability (the Schrodinger equation for a quantum harmonic oscillator is solved exactly using Hermite polynomials, yielding quantized energy levels), spectral accuracy (solving boundary-value problems with Chebyshev polynomials achieves exponential convergence for smooth solutions), capturing singular behavior (special functions adapt to irregular singular points (e.g., in Heun equations), where standard series fail), and connection to other branches of mathematics (Painleve equations (linked to orthogonal polynomials) describe universality in random matrix theory.)

The double confluent Heun equation (DCHE) is a second-order linear differential equation obtained by the confluence of one or more singularities present in the general Heun equation, which is represented as:

$$\frac{d^2y}{dx^2} + \left(\frac{a}{x} + \frac{b}{x-1} + \frac{c}{x-d}\right)\frac{dy}{dx} + \frac{\alpha\beta x - q}{x(x-1)(x-d)}y = 0.$$

This equation has four regular singular points at $\{0, 1, a, \infty\}$, in contrast to DCHE, which has two irregular singular points at zero and infinity [1]. It arises in mathematical physics and engineering, particularly in problems with strong singularities or non-Fuchsian behavior, such as quantum systems with singular potentials. The DCHE is challenging to solve due to the absence of standard Frobenius series expansions near its irregular singularities, necessitating advanced techniques like spectral methods, orthogonal polynomial expansions, or connection formulas involving special functions.

One of the techniques to deal with such DEs is the tridiagonal representation approach (TRA), laid down in the exemplary work of Alhaidari [2]. The roots of this approach date back to the pioneering works of Heller and Yamani [3]. Using this technique, the DEs of hypergeometric type [4], Bessel type [5], Heun type [6], etc., are solved, and their solutions are expressed as a series involving the product of cleverly chosen basis functions and orthogonal polynomials. In some of these studies, new families of orthogonal polynomials are obtained, and derivations of properties like Rodrigues' formula, determination of measure of orthogonality, etc., are formulated as open problems. This leaves a significant gap in the literature.

The construction of orthogonal polynomials from a given measure of orthogonality is classically achieved through the Gram–Schmidt orthogonalization process. However, the inverse problem–determining an appropriate measure or weight function corresponding to a given sequence of orthogonal polynomials—presents significant analytical challenges. While traditional interpolation methods, such as Lagrange or cubic spline techniques, can be employed when nodal data (typically at the zeros of the orthogonal polynomials) is available, these approaches often lack generality. In contemporary research, machine learning algorithms have emerged as powerful tools for extracting closed-form, human-interpretable expressions from empirical data, particularly in scenarios where analytical solutions are elusive. This work addresses this important gap in the literature by developing

a state-of-the-art machine learning framework for reconstructing measures of orthogonality. We demonstrate that our proposed algorithm possesses substantial scalability, extending its applicability beyond the specific cases examined in this study to encompass diverse families of orthogonal polynomials where analytical determination of weight functions proves intractable.

The versatility of orthogonal polynomials in modeling complex physical, chemical, and mathematical phenomena is evident from the available literature. For instance, Qiu et al. [7] and Si et al. [8] apply deep learning-based methods, including physics-informed neural networks (PINNs), to accurately model nonlocal nonlinear Schrodinger equations and soliton dynamics in optical systems. Similarly, Wan et al. [9] present a machine learning-assisted flexible sensor system for pesticide detection, combining biomimetic design with soliton principles. Mohammed et al. [10] propose an embedded kernel technique using orthogonal polynomials for 3D object recognition, bridging symbolic and geometric computing. Vakili–Nezhaad et al. [11] leverage orthogonal polynomial-based correlations to model CO_2-brine interfacial tensions in chemical systems. Last, Liu et al. [12] explore the orthogonal polynomial neural operator (OPNO) for solving PDEs with non-periodic boundaries, effectively combining operator learning with classical polynomial bases.

Recently, the single confluent Heun-type equation

$$\frac{d^2y}{dx^2} + \left(\frac{a}{x} - \frac{b}{1-x} + d\right)\frac{dy}{dx} + \frac{1}{x(1-x)}\left(\frac{A}{x} - \frac{B}{1-x} + Cx - D\right)y = 0 \tag{1.1}$$

is solved using the TRA method in [13]. The series solution is expressed by the product of basis functions involving Jacobi polynomials and modified continuous Hahn orthogonal polynomials. Motivated by this and various other studies stated above, in this paper, we extend the exploration of TRA to the solution of the double confluent Heun–type equation written as

$$\frac{d^2y}{dx^2} + \left(\frac{b}{x^2} + \frac{a}{x} + c\right)\frac{dy}{dx} + \frac{1}{x^2}\left(\frac{A}{x^2} + \frac{B}{x} + Ex + D\right)y = 0 \quad \text{or} \quad \mathcal{D}y(x) = 0,$$

$$\text{where } \mathcal{D} = \frac{d^2}{dx^2} + \left(\frac{b}{x^2} + \frac{a}{x} + c\right)\frac{d}{dx} + \frac{1}{x^2}\left(\frac{A}{x^2} + \frac{B}{x} + Ex + D\right), \tag{1.2}$$

where $\{a, b, c, A, B, D, E\}$ are real parameters.

Research has focused on finding solutions and connection formulas for this equation. The research [14] explored solutions for both confluent and double-confluent Heun equations, establishing relationships with quasi-exactly solvable problems. The study [15] used integral relations and transformation rules to derive new solutions for the DCHE, including a series of Coulomb wave functions. Applications of the Schrodinger equation with certain potentials were also discussed. The paper [16] proposed an algorithm to construct global solutions of the DCHE with specific behavior at singular points, expressing connection factors as quotients of Wronskians and using asymptotic expansions for computation. Solutions of DCHE near singularities using integral formulas and k-summability are given in [17]. These results complement the work [18]. Holomorphic solutions of DCHE associated with a physical model called RSJ are detailed in [19]. The relation of DCHE to Painleve transcendents can be obtained in random matrix theory and is discussed in [20].

The study of Heun and confluent Heun equations has been advanced through various solution techniques developed in the notable works of A. M. Ishkhanyan and T. A. Ishkhanyan. Their extensive contributions include representations of solutions to the biconfluent Heun equation expressed

through generalized hypergeometric functions [21] and Hermite functions [22], as well as solutions of the confluent Heun equation formulated in terms of confluent hypergeometric functions [23]. Furthermore, their research encompasses solutions of general Fuchsian linear ordinary differential equations using generalized hypergeometric functions [24] and innovative approaches to solving the general Heun equation through Appell hypergeometric functions [25]. Complementing these developments, polynomial solutions to the confluent Heun equation have been thoroughly investigated in [26].

However, in the present work, we develop an alternative approach to (1.2) through orthogonal polynomial expansions, providing new insights for this category of differential equations. The utility of series expansion methods becomes apparent when examining their capability to manage singular points effectively, as they enable solutions to be represented as convergent expansions centered at these critical locations. This approach proves particularly valuable for the confluent Heun equation, which typically lacks elementary solutions expressible in terms of standard special functions. Series representations stand among the limited techniques capable of yielding exact analytical solutions. A significant benefit of this methodology lies in its flexibility—the series can be tailored to various forms (including power series or asymptotic expansions) to accommodate different solution domains, such as neighborhoods of singularities or behavior at infinity.

The DCHE is closely related to other equations, such as the confluent Heun equation, and can be derived through limiting procedures [27]. Applications of the DCHE include solving the Schrodinger equation for certain potentials, particularly quasi-exactly solvable problems [28]. In some cases, the solutions can be reduced to confluent Heun polynomials, representing wavefunctions of bound states in quantum mechanical systems. Applications and various other treatments for DCHE can be found in [29–33] and references therein.

The remainder of this paper is structured as follows. In Section 2, we transform the differential Eq (1.2) into a three-term recurrence relation by applying the requirements of the TRA. We demonstrate that this recurrence relation corresponds to a special case of an established form in the literature. Consequently, the resulting polynomials form a novel two-parameter family of orthogonal polynomials denoted by $\mathcal{P}_n(\mu, \Sigma; z)$. The general solution of (1.2) is then expressed as a series expansion involving products of basis functions having generalized Bessel polynomials and the newly obtained family of orthogonal polynomials:

$$y(x) = \sum_{n=0}^{\infty} f_n(z)\phi_n(x). \tag{1.3}$$

Section 3 focuses on characterizing the spectrum by analyzing its nature (continuous, discrete, real, complex, or mixed) and geometric distribution in the complex plane as parameters vary. These investigations are supported by numerical experiments and graphical visualizations. In Section 4, we present a novel approach for deriving symbolic expressions for the orthogonality measure associated with our new polynomial family, utilizing the Python Symbolic Regressor (PySR) [34]. Through careful hyperparameter tuning, we achieve a remarkable accuracy of 10^{-4} when comparing actual weights (obtained by solving the discretized orthogonality system) with predicted weights (generated by the PySR-derived formula). We further provide a comparative analysis of this ML technique with deep symbolic regression (DSR) and conventional methods. The paper concludes with a summary of our findings and a discussion of their potential impact on future research directions.

Briefly, the study has two primary objectives:

1) Finding a series solution of DCHE (1.2) in terms of orthogonal polynomials.

2) Determination of the weight function for which these polynomials are orthogonal using a machine learning approach.

2. Main results

First of all, we attempt to convert (1.2) into a three-term recurrence relation (TTRR) in TRA. For this purpose, a careful selection of basis functions that can solve the DE is made. The structure of the singularities in (1.2) suggests choosing basis elements as follows:

$$\phi_n(x) = c_n x^{\alpha} e^{-\frac{\beta}{x}} \mathcal{J}_n^{\mu}(x), \tag{2.1}$$

where c_n is the normalization constant, α and β are arbitrary constants, and $\mathcal{J}_n^{\mu}(x)$ is the generalized Bessel polynomial [5, eqn. (A1)] (see also [35] for an equivalent form) given by

$$\mathcal{J}_n^{\mu}(x) = {}_2F_1\left(\begin{matrix} -n, n+2\mu+1 \\ - \end{matrix} \middle| -x \right), \quad x \geq 0, \quad \mu < -N - \frac{1}{2}, \quad n = 0, 1, \ldots, N.$$

The generalized Bessel polynomial satisfies the following recurrence relation [5, Eq (A2)]:

$$\begin{aligned} 2x\mathcal{J}_n^{\mu}(x) = & \frac{-\mu}{(n+\mu)(n+\mu+1)} \mathcal{J}_n^{\mu}(x) - \frac{n}{(n+\mu)(2n+2\mu+1)} \mathcal{J}_{n-1}^{\mu}(x) \\ & + \frac{n+2\mu+1}{(n+\mu+1)(2n+2\mu+1)} \mathcal{J}_{n+1}^{\mu}(x), \quad \text{where} \quad \mathcal{J}_0^{\mu}(x) = 1, \quad \mathcal{J}_{-1}^{\mu}(x) = 0. \end{aligned} \tag{2.2}$$

The first-order and second-order derivatives of basis elements can be obtained as follows:

$$\frac{d}{dx}\phi_n(x) = c_n x^{\alpha} e^{-\frac{\beta}{x}} \left(\frac{\alpha}{x} + \frac{\beta}{x^2} + \frac{d}{dx} \right) \mathcal{J}_n^{\mu}(x). \tag{2.3}$$

$$\frac{d^2}{dx^2}\phi_n(x) = c_n x^{\alpha} e^{-\frac{\beta}{x}} \left(\frac{\alpha}{x} + \frac{\beta}{x^2} + \frac{d}{dx} \right)^2 \mathcal{J}_n^{\mu}(x). \tag{2.4}$$

Theorem 2.1. *If $\alpha = \mu + 1 - \frac{a}{2}$, $2\beta = 1 - b$, and $c = E/2a$, then the solution of DCHE* (1.2) *is represented as:*

$$y(x) = \omega(x) \sum_{j=0}^{\infty} \mathcal{P}_n(\mu, \Sigma; z) \phi_j(x), \tag{2.5}$$

where $\omega(x)$ is the normalization factor (proportional to the square root of the weight function $w(x)$ to which $\mathcal{P}_n(\mu, \Sigma; z)$ are orthogonal), and polynomials $\mathcal{P}_n(\mu, \Sigma; z)$ satisfy the recurrence

$$\begin{aligned} \cos\theta \, \mathcal{P}_n(\mu, \Sigma; z) = z \sin\theta \left(\left(n + \mu + \frac{1}{2} \right)^2 - \Sigma^2 \right) \mathcal{P}_n(\mu, \Sigma; z) + \\ \mathcal{G}_{n-1} \mathcal{P}_{n-1}(\mu, \Sigma; z) + \mathcal{G}_n \mathcal{P}_{n+1}(\mu, \Sigma; z). \end{aligned} \tag{2.6}$$

where $\theta = \frac{\pi}{3}$,

$$\mathcal{G}_n = \frac{\sqrt{(n+1)(n+2\mu+1)}\,(n+\mu+1)}{2(n+\mu+1)\sqrt{(2n+2\mu+1)(2n+2\mu+3)}},\ \ \Sigma^2 = \left(\frac{a-1}{2}\right)^2 - \left(\frac{1-b}{2}\right)c - D,$$

and the orthogonality condition

$$\int \mathcal{P}_n(\mu,\Sigma;z)\mathcal{P}_m(\mu,\Sigma;z)w(z)dz = \delta_{mn}. \tag{2.7}$$

Proof. The action of operator $\mathcal{H} = x^2\mathcal{D}$ ($\mathcal{D}$ as in (1.2)) on the basis elements (2.1) with use of (2.3) and (2.4) yields

$$\mathcal{H}\phi_n(x) = c_n x^{\alpha+2}e^{-\frac{\beta}{x}}\left(\left(\frac{\alpha}{x}+\frac{\beta}{x^2}+\frac{d}{dx}\right)^2 + \left(\frac{b}{x^2}+\frac{a}{x}+c\right)\left(\frac{\alpha}{x}+\frac{\beta}{x^2}+\frac{d}{dx}\right) + \frac{1}{x^2}\left(\frac{A}{x^2}+\frac{B}{x}+Ex+D\right)\right)\mathcal{J}_n^{\mu}(x).$$

Expanding the power and multiplying the brackets, we have

$$\begin{aligned}\mathcal{H}\phi_n(x) = c_n x^{\alpha+2}e^{-\frac{\beta}{x}}\Bigg(&\frac{d^2}{dx^2} + \frac{\alpha^2}{x^2} + \frac{\beta^2}{x^4} + \frac{2\alpha\beta}{x^3} + \frac{2\alpha}{x}\frac{d}{dx} + \frac{2\beta}{x^2}\frac{d}{dx} + \frac{b\alpha}{x^3} + \frac{b\beta}{x^4} + \frac{b}{x^2}\frac{d}{dx}\\ &+ \frac{a\alpha}{x^2} + \frac{a\beta}{x^3} + \frac{a}{x}\frac{d}{dx} + \frac{c\alpha}{x} + \frac{c\beta}{x^2} + c\frac{d}{dx} + \frac{A}{x^4} + \frac{B}{x^3} + \frac{E}{x} + \frac{D}{x^2}\Bigg)\mathcal{J}_n^{\mu}(x).\end{aligned}$$

Collecting terms in descending order of derivatives, we obtain

$$\begin{aligned}\mathcal{H}\phi_n(x) = c_n x^{\alpha+2}e^{-\frac{\beta}{x}}\Bigg(&\frac{d^2}{dx^2} + \left(\frac{2\alpha+a}{x} + \frac{2\beta+b}{x^2} + c\right)\frac{d}{dx} + \frac{D+\alpha^2+a\alpha-\alpha+\beta c}{x^2}\\ &+ \frac{\alpha c+E}{x} + \frac{2\alpha\beta+b\alpha+a\beta-2\beta+B}{x^3} + \frac{\beta^2+\beta b+A}{x^4}\Bigg)\mathcal{J}_n^{\mu}(x).\end{aligned} \tag{2.8}$$

Reduction of (2.8) into TTRR makes the removal of derivatives $\frac{d^2}{dx^2}$ and $\frac{d}{dx}$ necessary. This is achieved using the differential equation (2.9) [5, Eq (A4)] and differential property (2.10) [5, Eq (A8)] for the generalized Bessel polynomial.

$$(x^2\frac{d^2}{dx^2} + (1+2x(\mu+1))\frac{d}{dx} - n(n+2\mu+1))y(x) = 0. \tag{2.9}$$

$$\begin{aligned}2x^2\frac{d}{dx}\mathcal{J}_n^{\mu}(x) = n(n+2\mu+1)\Bigg(&-\frac{\mathcal{J}_n^{\mu}(x)}{(n+\mu)(n+\mu+1)} + \frac{\mathcal{J}_{n-1}^{\mu}(x)}{(n+\mu)(2n+2\mu+1)}\\ &+ \frac{\mathcal{J}_{n+1}^{\mu}(x)}{(n+\mu+1)(2n+2\mu+1)}\Bigg).\end{aligned} \tag{2.10}$$

Taking the factor x^2 inside the brackets of (2.8) and using (2.9), we have

$$\mathcal{H}\phi_n(x) = c_n x^{\alpha}e^{-\frac{\beta}{x}}\left(\left(\frac{2\alpha+a-2\mu-2}{x} + \frac{2\beta+b-1}{x^2} + c\right)x^2\frac{d}{dx} + \frac{2\alpha\beta+b\alpha+a\beta-2\beta+B}{x} +\right.$$

$$D + \alpha^2 + a\alpha - \alpha + \beta c + n(n + 2\mu + 1) + \frac{\beta^2 + \beta b + A}{x^2} + (\alpha c + E)x\Big)\mathcal{J}_n^{\mu}(x). \tag{2.11}$$

The TRA requires only linear terms or constants to multiply with $\mathcal{J}_n^{\mu}(x)$ (and hence with $\phi_n(x)$). This suggests that the coefficient of the fractional terms $1/x$ and $1/x^2$ must vanish. These restrictions provide the following relation between the parameters of the basis and of the DE (1.2):

$$\beta(\beta + b) + A = 0 \Rightarrow \left(\frac{1-b}{2}\right)\left(\frac{1+b}{2}\right) + A = 0 \Rightarrow 1 - b^2 + 4A = 0.$$

$$\beta(2\alpha + a - 2) + b\alpha + B = 0 \Rightarrow \left(\mu + 1 - \frac{a}{2}\right)b + 2\mu\left(\frac{1-b}{2}\right) + B = 0 \Rightarrow \mu + \left(1 - \frac{a}{2}\right)b + B = 0$$

$$\Rightarrow \mu = \left(\frac{a}{2} - 1\right)b - B, \quad \text{and} \quad \alpha = (b-1)\left(\frac{a}{2} - 1\right) - B.$$

Putting these value in (2.11), we get

$$\mathcal{H}\phi_n(x) = c_n x^{\alpha} e^{-\frac{\beta}{x}}\left[cx^2\frac{d}{dx} + \underbrace{(D + \alpha(\alpha + a - 1) + n(n + 2\mu + 1) + \beta c)}\, + (\alpha c + E)x\right]\mathcal{J}_n^{\mu}(x). \tag{2.12}$$

The value of the middle term under the TRA assumptions is calculated as follows:

$$\begin{aligned}
&D + \alpha(\alpha + a - 1) + n(n + 2\mu + 1) + \beta c = \left(\mu + 1 - \frac{a}{2}\right)\left(\mu + \frac{a}{2}\right) + n^2 + 2nu + n \\
&+ \left(\frac{1-b}{2}\right)c + D = \mu^2 + \mu + \frac{a}{2} - \frac{a^2}{4} + n^2 + 2nu + n + \left(\frac{1-b}{2}\right)c + D \\
&= \left(\mu + \frac{1}{2}\right)^2 - \left(\frac{a-1}{2}\right)^2 + + n^2 + 2nu + n + \left(\frac{1-b}{2}\right)c + D \\
&= \left(n + \mu + \frac{1}{2}\right)^2 - \left(\frac{a-1}{2}\right)^2 + \left(\frac{1-b}{2}\right)c + D \\
&= \left(n + \mu + \frac{1}{2}\right)^2 - \Sigma^2.
\end{aligned}$$

Using the differential property (2.10) and recurrence relation for the generalized Bessel polynomial (2.2) in (2.12), we have,

$$\begin{aligned}
\mathcal{H}\phi_n(x) = c_n x^{\alpha} e^{-\frac{\beta}{x}}\Bigg[&\frac{cn(n + 2\mu + 1)}{2}\Bigg(- \frac{\mathcal{J}_n^{\mu}(x)}{(n + \mu)(n + \mu + 1)} + \frac{\mathcal{J}_{n-1}^{\mu}(x)}{(n + \mu)(2n + 2\mu + 1)} \\
&+ \frac{\mathcal{J}_{n+1}^{\mu}(x)}{(n + \mu + 1)(2n + 2\mu + 1)}\Bigg) + \left(\left(n + \mu + \frac{1}{2}\right)^2 - \Sigma^2\right)\mathcal{J}_n^{\mu}(x) \\
&+ \frac{\alpha c + E}{2}\Bigg(\frac{-\mu}{(n + \mu)(n + \mu + 1)}\mathcal{J}_n^{\mu}(x) - \frac{n}{(n + \mu)(2n + 2\mu + 1)}\mathcal{J}_{n-1}^{\mu}(x) \\
&\frac{n + 2\mu + 1}{(n + \mu + 1)(2n + 2\mu + 1)}\mathcal{J}_{n+1}^{\mu}(x)\Bigg)\Bigg].
\end{aligned}$$

Collecting the coefficients of $\mathcal{J}_n^{\mu}(x)$, $\mathcal{J}_{n-1}^{\mu}(x)$, and $\mathcal{J}_{n+1}^{\mu}(x)$, respectively, we obtain

$$\begin{aligned}\mathcal{H}\phi_n(x) = c_n x^{\alpha} e^{-\frac{\beta}{x}} \Bigg[& \left(\left(n+\mu+\frac{1}{2} \right)^2 - \Sigma^2 - \frac{cn(n+2\mu+1)}{2(n+\mu)(n+\mu+1)} - \frac{\mu(\alpha c+E)}{2(n+\mu)(n+\mu+1)} \right) \mathcal{J}_n^{\mu}(x) \\ & + \left(\frac{cn(n+2\mu+1)}{2(n+\mu)(2n+2\mu+1)} - \frac{n(\alpha c+E)}{2(n+\mu)(2n+2\mu+1)} \right) \mathcal{J}_{n-1}^{\mu}(x) \\ & + \left(\frac{cn(n+2\mu+1)}{2(n+\mu+1)(2n+2\mu+1)} + \frac{(\alpha c+E)(n+2\mu+1)}{2(n+\mu+1)(2n+2\mu+1)} \right) \mathcal{J}_{n+1}^{\mu}(x) \Bigg]. \end{aligned} \tag{2.13}$$

Simplifying the coefficients of $\mathcal{J}_n^{\mu}(x)$, $\mathcal{J}_{n-1}^{\mu}(x)$ and $\mathcal{J}_{n+1}^{\mu}(x)$ using the value of α, we have

$$\begin{aligned} & c\left(n^2+2n\mu+n+\mu^2+\mu-\frac{a}{2} \right) + \mu E = c\left(\left(n+\mu+\frac{1}{2} \right)^2 - \frac{1+2\mu a}{4} \right) + \mu E, \\ & nc\left(n+2\mu+1-\mu-1+\frac{a}{2} \right) - nE = nc\left(n+\mu+\frac{a}{2} \right) - nE. \end{aligned}$$

With a bit of elementary calculus, (2.13) reduces to

$$\begin{aligned} \frac{1}{c}\mathcal{H}\phi_n(x) = c_n x^{\alpha} e^{-\frac{\beta}{x}} \Bigg[& \left(\frac{1}{c}\left(n+\mu+\frac{1}{2} \right)^2 - \frac{\Sigma^2}{c} - \frac{\left(n+\mu+\frac{1}{2} \right)^2 - \frac{1+2\mu a}{4} + \frac{\mu E}{c}}{2(n+\mu)(n+\mu+1)} \right) \mathcal{J}_n^{\mu}(x) \\ & \mathcal{J}_{n-1}^{\mu}(x) \left(\frac{n\left(n+\mu+\frac{a}{2}-\frac{E}{c} \right)}{2(n+\mu)(2n+2\mu+1)} \right) + \mathcal{J}_{n+1}^{\mu}(x) \left(\frac{(n+2\mu+1)\left(n+\mu+1-\frac{a}{2}+\frac{E}{c} \right)}{2(n+\mu+1)(2n+2\mu+1)} \right) \Bigg]. \end{aligned}$$

Now, the multiplication of the expression outside the square brackets upon choosing the following normalization constant c_n produces terms proportional to $\phi_n(x)$, $\phi_{n-1}(x)$, and $\phi_{n+1}(x)$, i.e., if

$$c_n = \sqrt{(2n+2\mu+1)\frac{\Gamma(n+2\mu+1)}{\Gamma(n+1)}},$$

then,

$$\begin{aligned} \frac{1}{c}\mathcal{H}\phi_n(x) = & \left(\frac{1}{c}\left(n+\mu+\frac{1}{2} \right)^2 - \frac{\Sigma^2}{c} - \frac{\left(n+\mu+\frac{1}{2} \right)^2 - \frac{1+2\mu a}{4} + \frac{\mu E}{c}}{2(n+\mu)(n+\mu+1)} \right) \phi_n(x) \\ & + \left(\frac{\sqrt{n(n+2\mu)}\left(n+\mu+\frac{a}{2}-\frac{E}{c} \right)}{2(n+\mu)\sqrt{(2n+2\mu+1)(2n+2\mu-1)}} \right) \phi_{n-1}(x) \\ & + \left(\frac{\sqrt{(n+1)(n+2\mu+1)}\left(n+\mu+1-\frac{a}{2}+\frac{E}{c} \right)}{2(n+\mu+1)\sqrt{(2n+2\mu+1)(2n+2\mu+3)}} \right) \phi_{n+1}(x). \end{aligned} \tag{2.14}$$

Now, we have an expression of the form

$$c^{-1}\mathcal{H}\phi_n(x) = r_n\phi_j(x) + t_n\phi_{n+1}(x) + s_{n-1}\phi_{n-1}(x),$$

where $\phi_{-1}(x) = 0$. Note that (1.1) can also be represented as $\mathcal{H}y(x) = x^2\mathcal{D}y(x) = 0$ ($\because \mathcal{D}y(x) = 0$). Substituting (1.3) in (1.1) implies that the coefficients $f_n(z)$ should satisfy

$$u_n\lambda f_n(\lambda) = w_n f_n(z) + t_{n-1}f_{n-1}(z) + s_n f_{n+1}(z),$$

where $r_n = w_n - zu_n$ and z is some function of the parameters $\{\mu, \alpha, \beta, a, b, c, A, B, D, E\}$. The recurrence coefficients u_n, w_n, t_n, and s_n depend on n and the parameters $\{\mu, \alpha, \beta, a, b, c, A, B, D, E\}$, but are independent of z. Hence, the recurrence (2.14) together with the assumption $c = E/2a$ of Theorem 2.1 transforms to

$$\frac{1}{2}f_n(z) = \frac{a}{2E}\left(\left(n+\mu+\frac{1}{2}\right)^2 - \Sigma^2\right)f_n(z) + \left(\frac{\sqrt{n(n+2\mu)}\,(n+\mu)}{2(n+\mu)\sqrt{(2n+2\mu+1)(2n+2\mu-1)}}\right)f_{n-1}(z)$$
$$+\left(\frac{\sqrt{(n+1)(n+2\mu+1)}\,(n+\mu+1)}{2(n+\mu+1)\sqrt{(2n+2\mu+1)(2n+2\mu+3)}}\right)f_{n+1}(z).$$

This recurrence assumes the form of (2.6) for $f_n(z) = \mathcal{P}_n(\mu, \Sigma; z)$, $z = \frac{a}{\sqrt{3}E}$ and $\theta = \cos^{-1}\frac{1}{2}$, and the proof is complete. □

Remark 2.1. *We establish an identification between the two-parameter family of orthogonal polynomials $\mathcal{P}_n(\mu, \Sigma; z)$, which satisfy the recurrence relation* (2.6)*, and the four-parameter family of orthogonal polynomials discovered in the works of Alhaidari. This four-parameter family was presented in Eq (9) of [2], Eq (36) of [4], and Eqs (22) and (23) of [6], and whose recurrence relation is defined as:*

$$\cos\theta\,\mathcal{P}_n^{\mu,\nu}(z,\theta,\xi) = \left[z\sin\theta\left(\left(n+\frac{\mu+\nu+1}{2}\right)^2 - \xi^2\right) + \mathcal{F}_n\right]\mathcal{P}_n^{\mu,\nu}(z,\theta,\xi) +$$
$$2\mathcal{G}_{n-1}\mathcal{P}_{n-1}^{\mu,\nu}(z,\theta,\xi) + 2\mathcal{G}_n\mathcal{P}_{n+1}^{\mu,\nu}(z,\theta,\xi), \tag{2.15}$$

where

$$\mathcal{G}_n = \sqrt{\frac{(n+1)(n+\mu+1)(n+\nu+1)(n+\mu+\nu+1)}{(2n+\mu+\nu+1)(2n+\mu+\nu+2)^2(2n+\mu+\nu+3)}},\ \mathcal{F}_n = \frac{(\nu^2-\mu^2)}{(2n+\mu+\nu)(2n+\mu+\nu+2)}.$$

Clearly, for $\mu = \nu$ and $\xi = \Sigma$, (2.15) implies (2.6).

Remark 2.2. *When the limit $z \to 0$, the recurrence relation* (2.15) *takes the form of a recurrence relation satisfied by classical Jacobi polynomials $\mathcal{P}_n^{\mu,\nu}(x)$ [36] for $x = \cos\theta$.*

Remark 2.3. *The power series expansion* (2.5) *in this section is understood as formal, so that no questions of convergence are discussed.*

3. Analysis of spectrum of the polynomials $\mathcal{P}_n(\mu, \Sigma; z)$

The research [37] introduces a new family of polynomials orthogonal on the unit circle with a dense point spectrum, expressed using q-hypergeometric functions and associated with the wrapped

geometric distribution. The study [38] investigates the asymptotic behavior of orthogonal polynomials, demonstrating a connection between their behavior on and off the essential spectrum when the coefficients in the three-term recurrence relation form sequences of bounded variation. The work [39] investigated conditions for a denumerable spectrum unbounded above and below. Following this, the paper [40] explored conditions for spectral discreteness. While many orthogonal polynomials exhibit complex spectra, some may yield simpler, more predictable behaviors under specific conditions, highlighting the diversity in their spectral characteristics.

Building upon these profound investigations into the nature and distribution of zeros of orthogonal polynomials, we examine the polynomials $\mathcal{P}_n(\mu, \Sigma; z)$ from this perspective in the current work. Through systematic numerical experiments, we investigate both the spectral characteristics and the spatial distribution of their zeros. Our numerical analysis operates under the following constraints for the parameters:

- The degree n is restricted to natural numbers.
- The parameter μ is non-zero ($\mu \neq 0$).
- The parameter Σ may admit purely complex values, making $\sigma(= \Sigma^2)$ potentially positive, negative, or zero.

This parameter space generates six distinct cases for comprehensive examination.

$$\begin{array}{lll} \mu < 0 \text{ and } \sigma = 0, & \mu < 0 \text{ and } \sigma > 0, & \mu < 0 \text{ and } \sigma < 0. \\ \mu > 0 \text{ and } \sigma > 0, & \mu > 0 \text{ and } \sigma < 0, & \mu > 0 \text{ and } \sigma = 0. \end{array}$$

To get an in-depth insight into the location and nature of the spectrum, we further assume certain conditions within each case. These conditions are:

$$\begin{array}{lllll} \mu > \sigma, & \mu < \sigma, & |\mu| > \sigma, & |\mu| < \sigma, & \mu < |\sigma|, \\ \mu > |\sigma|, & |\mu| < |\sigma|, & |\mu| > |\sigma|, & n > \mu\sigma, & n > |\mu|\sigma, \\ n > \mu|\sigma|, & n > |\mu||\sigma|, & n < \mu\sigma, & n < |\mu|\sigma, & n < \mu|\sigma|. \end{array}$$

Based on various ranges of n, μ, and σ, several conclusions based on observations are tabulated below Table 1.

Table 1. Analysis of polynomial zeros for different cases of μ, σ, and n.

Case	Conditions	Zeros	Nature/Distribution
1.	$-1 < \mu < 0, \sigma = 0$	All real	Continuous spectrum.
2.	$-n < \mu < -1, \sigma = 0$	Both real and complex (R & C)	Continuous real spectrum clustering near origin, continuous complex spectrum within wedge-shaped region $\theta = \pm\pi/3$ (See Figure 1(a)).
3.	$\mu < -n, \sigma = 0$	R & C	Same as Case 2.
4.	$-1 < \mu < 1, 0 < \sigma < 1$	All real	Continuous spectrum.
5.	$-1 < \mu < 1, 1 < \sigma < n$	All real	Continuous spectrum.
6.	$-1 < \mu < 1, \sigma > n$	All real	Continuous spectrum.

Case	Conditions	Zeros	Spectrum/Distribution
7.	$-n < \mu < -1$, $0 < \sigma < 1$	R & C	Continuous real and complex spectrum, random distribution, no fixed geometry.
8.	$-n < \mu < -1$, $1 < \sigma < n$	Mostly real, sometimes complex depending on $\mu^2 < \sigma$ or $\mu^2 > \sigma$ (See Figure 4(a) and Figure 5)	Continuous real spectrum, discrete two-point complex spectrum.
9.	$-n < \mu < -1$, $\sigma > n$	R & C	Same as Case 7.
10.	$\mu < -n$, $0 < \sigma < 1$	R & C	Same as Case 7.
11.	$\mu < -n$, $1 < \sigma < n$	R & C	Same as Case 7.
12.	$\mu < -n$, $\sigma > n$	R & C	Continuous real and complex spectrum clustered near origin, spectrum lying within wedge-shaped region not centered at origin (Figure 4(b)).
13.	$-1 < \mu < n$, $-1 < \sigma < 0$	All real	Continuous spectrum.
14.	$-1 < \mu < n$, $-n < \sigma < -1$	All real	Continuous spectrum.
15.	$-1 < \mu < n$, $\sigma < -n$	All real	Continuous spectrum.
16.	$-n < \mu < -1$, $-1 < \sigma < 0$	R & C	No fixed geometry or specturm.
17.	$-n < \mu < -1$, $-n < \sigma < -1$	R & C	Same as Case 16.
18.	$-n < \mu < -1$, $\sigma < -n$	R & C	Same as Case 16.
19.	$\mu < -n$, $-1 < \sigma < 0$	R & C	Continuous real and complex spectrum, geometry resembling Figure 3 with no discrete parts.
20.	$\mu < -n$, $-n < \sigma < -1$	R & C	Same as Case 19.
21.	$\mu < -n$, $\sigma < -n$	R & C	Continuous complex and discrete real spectrum, geometry resembles with modified Folium of Descartes (Figure 6).
22.	$1 < \mu < n$, $0 < \sigma < 1$	All real	Continuous spectrum.

Case	Conditions	Zeros	Spectrum/Distribution
23.	$1 < \mu < n, 1 < \sigma < n$	Mostly real, sometimes complex depending on $\mu \times \sigma > n$ and $\mu >$ (or $<$)σ (See Figures 7 and 8)	Real spectrum consists of both discrete and continuous parts, complex spectrum is continuous only, geometry is Chaotic.
24.	$1 < \mu < n, \sigma > n$	All real	Continuous spectrum.
25.	$\mu > n, 0 < \sigma < 1$	R & C	Continuous real and complex spectrum, complex zeros form an oval-shaped region and real zeros form a slit as shown in Figure 1(b).
26.	$\mu > n, 1 < \sigma < n$	R & C	Same as Case 25.
27.	$\mu > n, \sigma > n$	R & C	Same as Case 25.
28.	$\mu > n, -1 < \sigma < 0$	R & C	Same as Case 19.
29.	$\mu > n, -n < \sigma < -1$	R & C	Same as Case 19.
30.	$\mu > n, \sigma < -n$	R & C	Same as Case 19.
31.	$0 < \mu < 1, \sigma = 0$	All real	Continuous spectrum.
32.	$1 < \mu < n, \sigma = 0$	Mostly real, sometimes complex depending on $\mu <$ (or $>$)$\frac{n}{3} + \frac{1}{2}$ (See Figures 2 and 3)	Purely real spectrum is continuous, mixed spectrum consists of both continuous and discrete parts.
33.	$\mu > n, \sigma = 0$	R & C	Same as Case 7.

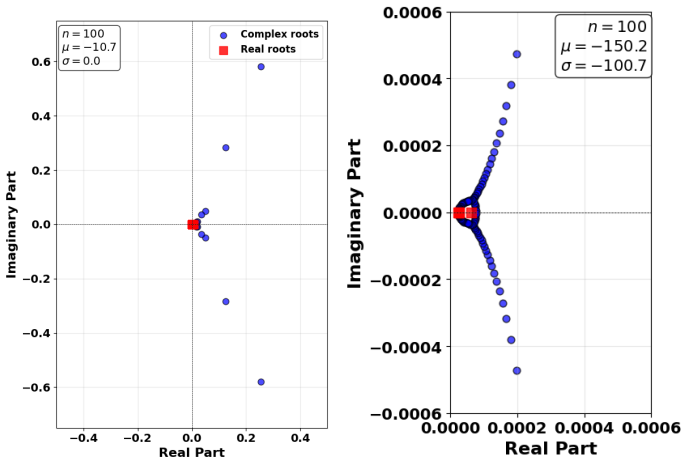

Figure 1. (a) Zeros of $\mathcal{P}_{100}(-10.7, 0; z)$, (b) Zeros of $\mathcal{P}_{100}(-150.2, -100.7; z)$.

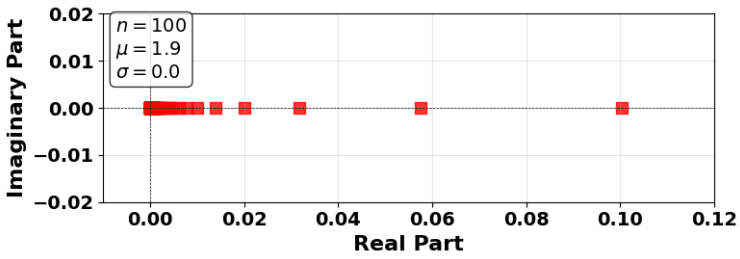

Figure 2. Zeros of $\mathcal{P}_{100}(1.9, 0; z)$.

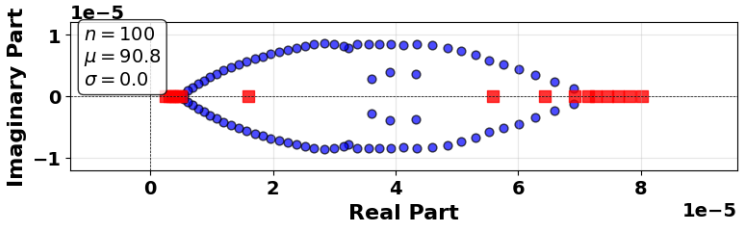

Figure 3. Zeros of $\mathcal{P}_{100}(90.8, 0; z)$.

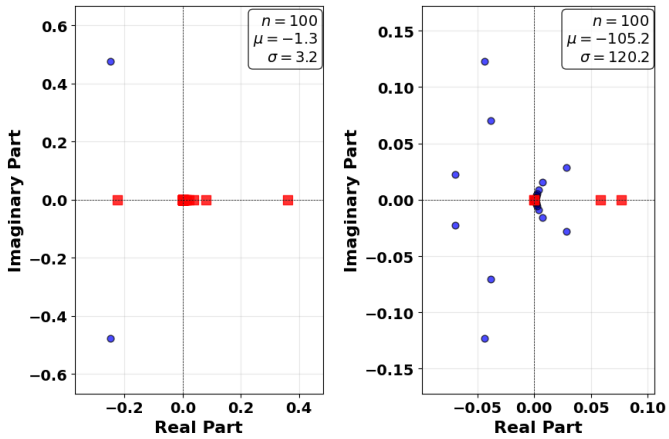

Figure 4. (a) Zeros of $\mathcal{P}_{100}(-1.3, 3.2; z)$, (b) Zeros of $\mathcal{P}_{100}(-105.2, 120.2; z)$.

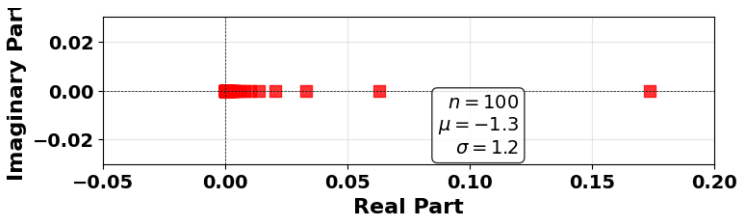

Figure 5. Zeros of $\mathcal{P}_{100}(-1.3, 1.2; z)$.

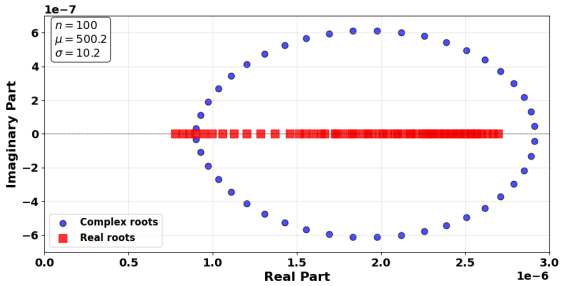

Figure 6. Zeros of $\mathcal{P}_{100}(500.2, 10.2; z)$.

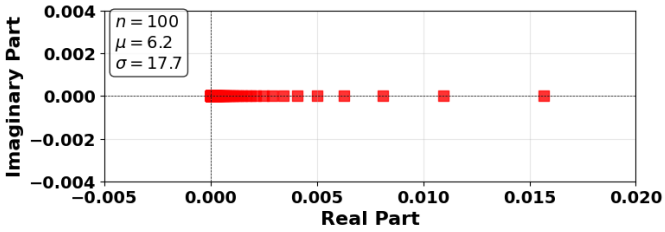

Figure 7. Zeros of $\mathcal{P}_{100}(6.2, 17.7; z)$.

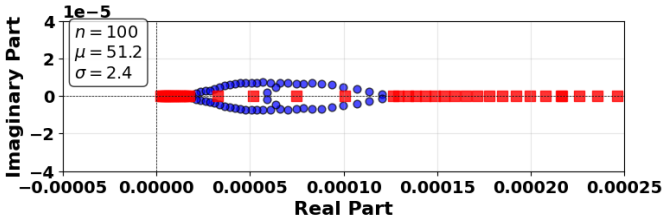

Figure 8. Zeros of $\mathcal{P}_{100}(51.2, 2.4; z)$.

4. Determination of the weight function $w(z)$ using machine learning

PySR (Python symbolic regression) is a specialized machine learning (ML) algorithm for symbolic regression, bridging data-driven learning and interpretable modeling. It's ideal when you need explainable mathematical relationships, parsimonious equations, or human-readable expressions. The study employs symbolic regression to derive an interpretable mathematical expression that approximates the weight function $w(z)$ associated with nodal points z. Unlike traditional curve-fitting methods constrained by predefined forms (e.g., linear, polynomial, spline, or Lagrange interpolation), this approach discovers both the structure and coefficients of the equation from data. It uses evolutionary algorithms (genetic programming) to search the space of possible equations. Stone—Weierstrass theorem guarantees that any well-behaved weight function can, in principle, be approximated arbitrarily closely by expressions built from basic mathematical operators—precisely the search space that PySR explores. Research shows that genetic programming can achieve convergence to optimal solutions under appropriate conditions. While the search space is vast, the evolutionary process is theoretically grounded in probability theory and has proven convergence properties for symbolic regression problems [41, 42].

Table 2. Symbolic regression workflow.

Step	Phase	Description	Key actions/rationale
1.	Custom Polynomial Construction	Generate quadrature nodes and weights from orthogonal polynomials	Define recurrence parameters μ and σ; prepare three-term recurrence and spectral method to get meaningful training data; build basis for supervised learning.
2.	Compute Roots (Nodes)	Use eigenvalue technique to find polynomial roots	Construct tri-diagonal Jacobi matrix J using recurrence coefficients α_n, β_n; compute eigenvalues λ_i as polynomial roots x_i.
3.	Evaluate Polynomial Squares	Evaluate $P_k^2(x_i)$ for orthogonal polynomials	Use recurrence to generate $P_k(x)$ values; store $P_k^2(x_i)$ as rows for later inclusion in least-squares constraints.
4.	Compute Weights	Solve for weights to enforce orthogonality	Use $\sum w(x_i)P_k^2(x_i) = 1$ for $k = 1, \ldots, k_{\max}$; solve overdetermined system $A^T Aw = A^T b$ via least-squares.
5.	Data Preparation	Format data for machine learning	Create input features $X = x_i$, targets $y = w(x_i)$; apply train/test split to facilitate model and generalization evaluation.
6.	Model Configuration & Training	Initialize `PySR` symbolic regression model	Select allowed operators: binary $(+, -, \times, /, \wedge)$, unary $(\exp, \log, \sqrt{}, 1/x)$; set population and evolution parameters; train model on $(X_{\text{train}}, y_{\text{train}})$.
7.	Extract Symbolic Expression	Retrieve discovered analytical form	Extract best symbolic formula $w(x)$ from fitted model; convert to `SymPy`/NumPy function for evaluation.
8.	Model Evaluation	Evaluate model prediction on test set	Predict $\hat{w}(x_i)$ on test data; compute absolute error, mean absolute error (MAE), and max error; tabulate all results for transparency.
9.	Train-Set Comparison	Estimate overfitting by comparing train/test error	Evaluate MAE on training data; compare with test MAE to gauge overfitting or underfitting.
10.	Visualization	Plot raw points and model curve	Overlay symbolic function on top of node-weight scatter; distinguish test vs train; assess smoothness and interpretability.

Our approach has particular theoretical merit because by using the orthogonality condition (4.1), we are constraining the search space to mathematically meaningful solutions. This transforms an unconstrained approximation problem into a constrained optimization problem with clear

mathematical foundations. Our system operates on a finite set of nodes, making the problem well-posed with unique solutions when the constraint matrix has full rank. Our implementation of train-test splits provides empirical stability validation, demonstrating that discovered weight functions generalize beyond the training data. Also, our finite-dimensional approach avoids the instabilities associated with infinite moment sequences, ensuring numerical stability in the reconstruction process [43]. For bounded intervals, the Hausdorff moment problem has well-established stability properties. Weight functions on $[a, b]$ are stable under small perturbations of the moment constraints [44]. An overview of the various steps involved in the process is detailed in Table 2.

4.1. Data preparation

The n zeros $\{z_1, z_2, \ldots, z_n\}$ of the polynomial $\mathcal{P}_n(\mu, \Sigma; z)$ are used as node points. Using these node points, the discretization of (2.7) when $m = n$ can be written as:

$$\sum_{i=1}^{n} \mathcal{P}_{i-1}^2(\mu, \Sigma; z_i) w(z_i) = 1, \tag{4.1}$$

where $w_i(z_i)$ for $i = 1, 2, \ldots, n$ are the n unknown weights. The orthogonality condition (4.1) yields a system of n equations with n unknowns. We have successfully reconstructed the symbolic weight function using datasets of 10, 20, and 30 node points. The data set is split into a 70-30 ratio, i.e., 70% of the nodes were used for training the model, and the test or validation is carried out on the remaining 30% of the nodes. Consider the nodes $\{z_1, z_2, \ldots, z_{10}\}$ with their corresponding weights $\{w(z_1), w(z_2), \ldots, w(z_{10})\}$. These nodes were specifically chosen as the zeros of the polynomial $\mathcal{P}_{10}(5.2, 3.5; z)$. The resulting linear system was solved numerically using Python's built-in least squares solver. The least-squares formulation with orthogonality constraints acts as implicit regularization, promoting stability. For clarity in subsequent discussion:

- The computed values $w(z_i)$ will be referred to as the actual weights.
- The values $\hat{w}(z_i)$ obtained from the machine-learned symbolic expression will be termed the predicted weights.

The node values spanned from 0.06×10^{-2} to 1.1976×10^{-2} and the corresponding weights spanned from $2.42549295 \times 10^{-1}$ to $4.92698827 \times 10^{-5}$. Data was used as-is without normalization, as the wide dynamic range of z_i and $w(z_i)$ is intrinsic to the problem.

Remark 4.1. *The parameters μ and Σ are carefully selected to ensure that the zeros are properly distributed along the real line, which leads to superior approximation quality. Our analysis reveals that when zeros are chosen too close together (resulting in high density), the approximation becomes noisy, and it proves challenging to maintain high accuracy between the actual weights and their predicted counterparts.*

Remark 4.2. *As demonstrated, the spectrum of $\mathcal{P}_n(\mu, \Sigma; z)$ can include both real and complex zeros. However, extending this methodology to derive symbolic expressions for the weight function via machine learning in these cases falls outside the scope of the current work.*

4.2. Hyperparameters

Genetic programming exhibits both genotypic and phenotypic convergence over extended runs. Studies demonstrate that meaningful convergence typically occurs after 1000+ generations [42, 45]. For symbolic regression problems, 2000 iterations provides sufficient exploration time to reach convergence with high probability [46]. Population genetics theory shows that population size N affects the probability of fixation of beneficial mutations. The optimal size balances exploration (larger N) with computational efficiency, with 20–40 populations being theoretically optimal for symbolic regression. Population diversity is crucial for preventing premature convergence and enabling exploration of diverse regions in the search space [47]. Multiple populations enhance the effectiveness of crossover operations by maintaining genetic diversity. PySR uses multiple populations in an "island methodology" where each population specializes in different regions of the search space. The optimal size balances exploration (larger N) with computational efficiency, with 20–40 populations being theoretically optimal for symbolic regression [48]. The operator selection +, *, -, /, ^ is mathematically justified as they form a functionally complete basis. These operations can construct polynomials of arbitrary degree. Combined with rational functions (via division), they can approximate any continuous function. exp and log enable the modeling of exponential processes common in physical systems. sqrt provides access to power functions with fractional exponents, etc. The complexity penalty prevents overfitting by penalizing unnecessarily complex constants. Hence, the PySR library was used with the hyperparameter settings presented in Table 3.

Table 3. Symbolic regression parameters.

Parameter	Value	Description
Iterations	2000	Number of evolutionary generations
Populations	30	Parallel populations for diversity
Max size	50	Maximum operations per equation
Parsimony	0.0005	Complexity penalty coefficient
Optimization cycles	1000	Optimization cycles per generation
Constant complexity	5	Allowed complexity for constant
Binary operators	+, *, -, /, ^	Basic arithmetic and power operations
Unary operators	exp, log, sqrt, inv, abs, sin, cos, tanh	Transcendental and activation functions

4.3. Results and discussion

The PySR ML algorithm discovers the following weight function:

$$w(z) = -281.1595z^2 + \frac{0.0060805}{|\ln(z) + 4.81454|^{2.04098}} \tag{4.2}$$

with respect to which the polynomials $\mathcal{P}_n(\mu, \Sigma; z)$, $n = 0, 1, \ldots, 10$ are orthogonal. Here, we would like to reiterate that $\{z_1, z_2, \ldots, z_7\}$ are used for training and discovery purpose, while $\{z_8, z_9, z_{10}\}$ are the test or validation nodes. The nodes $\{z_1, z_2, \ldots, z_{10}\}$ are substituted into the expression (4.2) to compute the predicted weights $\hat{w}(z_i)$. Figure 9 shows a comparative plot of both the actual weights and predicted weights against the nodal points. The visual inspection reveals an excellent agreement between them,

with discrepancies being practically negligible. Quantitatively, as evidenced by Table 4 (for 10 nodal points), the approximation error remains consistently of order 10^{-4} in the case of both the training and testing sets (see Figure 10). The superior performance is credited to fine-tuning of constants, deterministic sorting (Pareto-optimality), and population diversity. Field-relevant error metrics for the case when $n = 20$ and $n = 30$ are tabulated in Table 5.

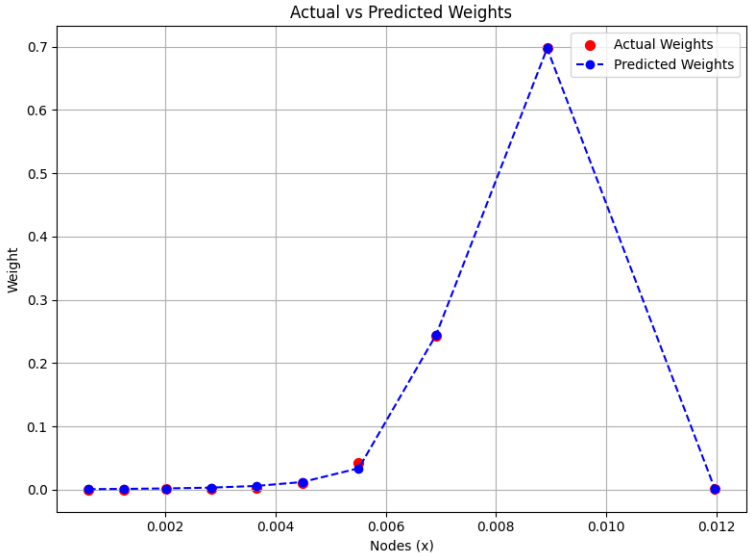

Figure 9. Plot for actual weights and predicted weights.

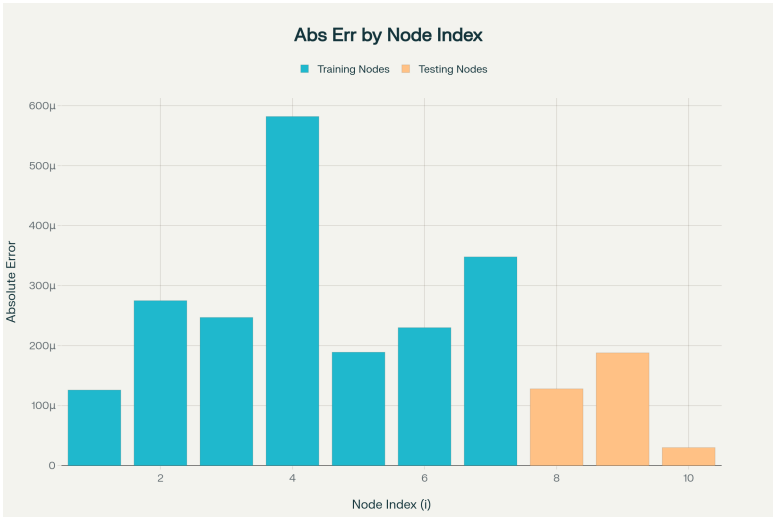

Figure 10. Absolute errors for training and testing nodes indicating prediction accuracy for each node index.

Table 4. Node-wise weight predictions and errors.

i	z_i	Actual $w(z_i)$	Predicted $\hat{w}(z_i)$	Absolute error
1	0.000600	0.000049	0.000176	0.000126
2	0.001247	0.000416	0.000692	0.000275
3	0.002011	0.000702	0.000456	0.000247
4	0.002833	0.001266	0.000684	0.000582
5	0.003651	0.003511	0.003700	0.000189
6	0.004486	0.010729	0.010959	0.000230
7	0.005511	0.042728	0.042380	0.000348
8	0.006916	0.242549	0.242677	0.000128
9	0.008932	0.697538	0.697349	0.000188
10	0.011976	0.000511	0.000541	0.000030

Table 5. Error metrics.

Metric	$n = 10$	$n = 20$	$n = 30$
Train Mean Absolute Error (MAE)	2.854e-4	4.936e-04	1.148e-04
Test Mean Absolute Error (MAE)	1.155e-4	3.946e-04	0.842e-04
Train Maximum Absolute Error	5.823e-4	9.183e-04	6.961e-04
Test Maximum Absolute Error	1.882e-4	8.391e-04	1.669e-04
Runtime (in seconds)	932	1815	2613

The weight function using Lagrange interpolation comes out to be

$$w(z) = 5.804 \times 10^{20} z^9 - 2.345 \times 10^{19} z^8 + 3.868 \times 10^{17} z^7 - 3.423 \times 10^{15} z^6 + 1.797 \times 10^{13} z^5 - 5.793 \times 10^{10} z^4 + 1.143 \times 10^{8} z^3 - 1.322 \times 10^{5} z^2 + 80.66z - 0.01919. \tag{4.3}$$

On the other hand, spline interpolation produces 9 piecewise cubic polynomials, requiring the determination of 36 constants for interpolation across 10 nodes. For conciseness, we refrain from presenting the detailed expressions of all these polynomials.

4.4. Comparison with traditional models

As demonstrated by Eq (4.2), the proposed algorithm offers significant advantages when increasing the number of node points. Notably, it maintains accuracy without requiring additional terms. In comparison, Eq (4.3) reveals that the Lagrange interpolation formula for 10 nodes already involves 10 terms, meaning scaling to 100 nodes would require 100 terms. Similarly, a spline-based approach would necessitate 99 piecewise polynomials. These observations highlight the scalability of our machine learning approach compared to traditional methods. Although the time taken by traditional models can be quite less, ranging between 15 and 60 seconds (depending on the number of nodes involved), the scalability and closed-form expression obtained using PySR compensates for the time consumed while applying this method.

Furthermore, the large leading coefficient in (4.3) indicates numerical instability, while splines may suffer from discontinuities or loss of differentiability near nodal points. In contrast, the symbolic

expressions generated by our PySR model inherently overcome these limitations by producing smooth, well-behaved functions that preserve both continuity and differentiability.

4.5. Comparison with an ML model

This section provides a comparison between the PySR symbolic regression algorithm and a Deep Symbolic Regression (DSR) approach, applied to the problem of discovering weight functions when $n = 10$. DSR is a deep learning approach that uses RNN with reinforcement learning to generate expressions. It excels at finding novel function forms but may overcomplicate simple relationships. It provides less consistent results on small datasets, but can be useful when the dataset has more than 100 node points and computational resources are abundant. Table 6 demonstrates that our choice of PySR is optimal for the weight function discovery problem involving small datasets.

Table 6. Performance Metrics for PySR vs. DSR.

Metric	PySR	DSR	Relative Difference
Test MAE	1.155e-4	1.299 e-4	PySR better by 12.5%
Test Max Absolute Error	1.882e-4	1.2797 e-4	DSR better by 30.2%
Training Time (minutes)	45	126	PySR 2.8× faster
Expression Complexity	Low	Medium–High	PySR simpler by 50%
Memory Usage	2GB	8GB	PySR better
Approach	Evolutionary	Deep RL	Depends on node points

4.6. Computational setup

The symbolic regression experiments were performed on a workstation equipped with an Intel Xeon E5-2680 v4 processor (14 cores, 2.4 GHz) and 64 GB of RAM, running Ubuntu 20.04 LTS. The Python environment utilized PySR version 0.11.4 with Julia 1.8.5 as the back-end computational engine. Each regression run was configured with 2000 evolutionary iterations across 30 parallel populations, requiring approximately 45 minutes of wall-clock time for complete convergence. The implementation leveraged NumPy 1.22.4 and SymPy 1.11.1 for numerical computations and symbolic manipulation, respectively. All experiments were conducted with fixed random seeds (seed=42) to ensure reproducibility of the evolutionary search process. The complete codebase, including parameter configurations and visualization routines, was managed through JupyterLab 3.5.0 with Python 3.9.15.

4.7. Challenges faced and limitations

The optimal tuning of hyperparameters, as detailed in Table 3, presented a significant challenge in achieving the reported accuracy. Extensive experimentation with various parameter configurations was required to attain the desired precision. These parameters are considered optimal in a specific sense: any reduction in the first five parameters—iterations, Population, Max Size, Parsimony, and Optimization Cycles—along with the Constant complexity, by increments of 100, 5, 5, 0.001, and 1, respectively, results in a degradation of accuracy, increasing the mean absolute error (MAE) from 10^{-4} to 10^{-3}. For instance, decreasing the *Iterations* from 2000 to 1900 while holding other parameters constant leads to an MAE of 10^{-3}, whereas increasing it to 2100 maintains the original MAE of 10^{-4}.

This sensitivity underscores the careful balance struck between computational efficiency and accuracy, ensuring that the chosen parameters represent the minimal configuration necessary to achieve the target precision without unnecessary computational overhead. Each parameter was meticulously adjusted to optimize performance, reflecting a trade-off between resource expenditure and error minimization.

It is evident from the theory of orthogonal polynomials that n-point quadrature rules are exact for approximating polynomials of degree $2n - 1$. Hence, we could expect a similar phenomenon in our study, i.e., the weight function discovered using 10 node points should work for polynomials up to degree 19. Specifically, $w(z)$, which is trained and tested on 10 node points and given by (4.2), should provide a good approximation of the weight obtained using 19 nodes, thereby saving approximately 50% of the computational cost. To verify this, one must test higher-degree polynomials by integrating them with respect to weight (4.2) and checking how close the result is to 1:

$$\int_{z_{\min}}^{z_{\max}} \mathcal{P}_{i-1}^2(\mu, \Sigma; z) w(z) dz \approx 1, \quad i = 11, \ldots, 19, \quad z_{\min} = \min\{z_1, \ldots, z_{19}\}, \quad z_{\max} = \max\{z_1, \ldots, z_{19}\}.$$

Our experimental results demonstrate that $w(z)$ performs well up to degree 12, but from degree 13 onwards, the accuracy begins to diminish. This behavior indicates that the formula remains stable up to a certain threshold degree, approximately 20% higher than the number of nodes actually used, beyond which it loses stability. While we believe that employing a larger dataset and high-performance computing environment could yield better results, achieving the ideal scenario of 50% computational savings appears particularly challenging. Consequently, we have chosen to leave further investigation of this aspect for future work.

5. Conclusions and future scope

A series solution to the differential equation (DCHE) (1.2) is derived using the TRA, expressed in terms of a novel two-parameter family of orthogonal polynomials. Through careful analysis, we identify these polynomials as a special case of a known four-parameter family of orthogonal polynomials from the literature. This four-parameter family generalizes the classical Jacobi polynomials, thereby connecting our results to well-established theory. We systematically investigate the behavior of polynomial zeros across different parameter ranges, supported by comprehensive numerical experiments. Furthermore, we address the inverse problem of determining closed-form expressions for the weight function associated with a given family of orthogonal polynomials. For this purpose, we employ modern ML techniques, demonstrating their effectiveness in this theoretical context. Our numerical validation shows that the ML-discovered weight function achieves satisfactory accuracy when evaluated at node points. To the best of our knowledge, this work represents the first interdisciplinary study bridging orthogonal polynomial theory with machine learning and artificial intelligence. The successful application of ML for weight function discovery opens new research directions in this field and establishes a novel methodological framework for future investigations.

Theoretically, the derivation of Rodrigues' type formula, generating function, quadrature rules on the real line, etc., for the polynomials $\mathcal{P}_n(\mu, \Sigma; z)$ remains open. Computationally, it would be interesting to compare its performance with other symbolic regression competitors like genetic programming (gplearn), SINDY (sparse Identification of Nonlinear Dynamics), Eureqa, AI Feynman, Operon, Deep Symbolic Regression (DSR), etc. Another direction is to explore neural

network approaches, e.g., MLP (Multilayer Perceptron) and Transformer-based Symbolic Regression (PySR+LLMs), etc., in the context of the present study.

Orthogonal polynomial research is poised for significant advancement, driven by its deep connections to numerical analysis, approximation theory, and spectral methods. As machine learning evolves, these polynomials offer powerful tools for interpretable model construction, efficient feature representations, and solving complex differential equations [45, 49, 50]. Future research will likely focus on integrating orthogonal polynomials with modern neural architectures, such as polynomial neural operators, to enhance the accuracy and generalization of machine-learned models in scientific computing. Applications will expand across physics, chemistry, signal processing, and data-driven scientific discovery, leveraging the ability of orthogonal polynomials to encode structure and symmetries in data.

Author contributions

Varun Kumar: methodology, conceptualization, validation, formal analysis, writing-original draft preparation; K. Laxminarayanamma: methodology, conceptualization, validation, formal analysis, writing-original draft preparation, writing-review and editing; Abhishek Kumar Singh: methodology, conceptualization, validation, writing-original draft preparation, writing-review and editing; Brajesh Shukla: methodology, conceptualization, validation, formal analysis, writing-original draft preparation; Saiful Rahman Mondal: conceptualization, validation, writing-review and editing, finance. All authors of this article have contributed equally. All authors have read and approved the final version of the manuscript for publication.

Use of Generative-AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Funding

This work was supported by the Deanship of Scientific Research, Vice Presidency for Graduate Studies and Scientific Research, King Faisal University, Saudi Arabia (Grant No. 252728).

Conflict of interest

The authors declare that they have no conflicts of interest.

References

1. A. Ronveaux, *Heun's differential equations*, Oxford: Oxford University Press, 1995. https://doi.org/10.1093/oso/9780198596950.001.0001
2. A. D. Alhaidari, Orthogonal polynomials derived from the tridiagonal representation approach, *J. Math. Phys.*, **59** (2018), 013503. https://doi.org/10.1063/1.5001168

3. E. J. Heller, H. A. Yamani, *J*-Matrix method: Application to S-wave electron-hydrogen scattering, *Phys. Rev. A*, **9** (1974), 1209–1214. https://doi.org/10.1103/PhysRevA.9.1209

4. A. D. Alhaidari, Series solutions of Laguerre- and Jacobi-type differential equations in terms of orthogonal polynomials and physical applications, *J. Math. Phys.*, **59** (2018), 063508. https://doi.org/10.1063/1.5027158

5. A. D. Alhaidari, H. Bahlouli, Solutions of a Bessel-type differential equation using the tridiagonal representation approach, *Rep. Math. Phys.*, **87** (2021), 313–327. https://doi.org/10.1016/S0034-4877(21)00039-2

6. A. D. Alhaidari, Series solutions of Heun-type equation in terms of orthogonal polynomials, *J. Math. Phys.*, **59** (2018), 113507. https://doi.org/10.1063/1.5045341

7. W.-X. Qiu, Z.-Z. Si, D.-S. Mou, C.-Q. Dai, J.-T. Li, W. Liu, Data-driven vector degenerate and nondegenerate solitons of coupled nonlocal nonlinear Schrödinger equation via improved PINN algorithm, *Nonlinear Dyn.*, **113** (2025), 4063–4076. https://doi.org/10.1007/s11071-024-09648-y

8. Z.-Z. Si, D.-L. Wang, B.-W. Zhu, Z.-T. Ju, X.-P. Wang, W. Liu, et al., Deep learning for dynamic modeling and coded information storage of vector-soliton pulsations in mode-locked fiber lasers, *Laser Photonics Rev.*, **18** (2024), 2400097. https://doi.org/10.1002/lpor.202400097

9. Y. Wan, Q. Wei, H. Sun, H. Z. B. Wu, Y. M. Zhou, C. W. Bi, et al., Machine learning assisted biomimetic flexible SERS sensor from seashells for pesticide classification and concentration prediction, *Chem. Eng. J.*, **507** (2025), 160813. https://doi.org/10.1016/j.cej.2025.160813

10. A. A. Mohammed, A. H. Al-sudani, A. M. Abdul-Hadi, A. Alwhelat, B. M. Mahmmod, S. H. Abdulhussain, et al., Three-dimensional object recognition Using orthogonal polynomials: An embedded kernel approach, *Algorithms*, **18** (2025), 78. https://doi.org/10.3390/a18020078

11. G. R. Vakili-Nezhaad, A. Al Shaaili, R. Yousefzadeh, A. Kazemi, A. Al Ajmi, CO2-Brine interfacial tension correlation based on the classical orthogonal polynomials: monovalent salts with common anion, *Chem. Pap.*, **78** (2024), 3483–3493. https://doi.org/10.1007/s11696-024-03321-9

12. Z. Y. Liu, H. F. Wang, H. Zhang, K. J. Bao, X. Qian, S. H. Song, Render unto numerics: Orthogonal polynomial neural operator for PDEs with nonperiodic boundary conditions, *SIAM J. Sci. Comput.*, **46** (2024), 323–348. https://doi.org/10.1137/23M1556320

13. S. R. Mondal, V. Kumar, An orthogonal polynomial solution to the confluent-type Heun's differential equation, *Mathematics*, **13** (2025), 1233. https://doi.org/10.3390/math13081233

14. L. J. El-Jaick, B. D. B. Figueiredo, Solutions for confluent and double-confluent Heun equations, *J. Math. Phys.*, **49** (2008), 083508. https://doi.org/10.1063/1.2970150

15. L. J. El-Jaick, B. D. B. Figueiredo, Confluent Heun equations: convergence of solutions in series of Coulomb wavefunctions, *J. Phys. A: Math. Theor.*, **46** (2013), 085203. https://doi.org/10.1088/1751-8113/46/8/085203

16. J. Abad, F. J. Gómez, J. Sesma, An algorithm to obtain global solutions of the double confluent Heun equation, *Numer. Algor.*, **49** (2008), 33–51. https://doi.org/10.1007/s11075-008-9197-4

17. A. Roseau, On the solutions of double confluent Heun equations, *Aequat. Math.*, **60** (2000), 116–136. https://doi.org/10.1007/s000100050140

18. D. Schmidt, G. Wolf, Double confluent Heun equation, In: *Heun's differential equations*, Oxford: Oxford University Press, 1995, 129–188.

19. V. M. Buchstaber, S. I. Tertychnyi, Holomorphic solutions of the double confluent Heun equation associated with the RSJ model of the Josephson junction, *Theor. Math. Phys.*, **182** (2015), 329–355. https://doi.org/10.1007/s11232-015-0267-1

20. J. D. Yu, C. Z. Li, M. K. Zhu, Y. Chen, Asymptotics for a singularly perturbed GUE, Painlevé III, double-confluent Heun equations, and small eigenvalues, *J. Math. Phys.*, **63** (2022), 063504. https://doi.org/10.1063/5.0062949

21. D. Y. Melikdzhanian, A. M. Ishkhanyan, Generalized-hypergeometric solutions of the biconfluent Heun equation, *Ramanujan J.*, **57** (2022), 37–53. https://doi.org/10.1007/s11139-021-00504-w

22. T. A. Ishkhanyan, A. M. Ishkhanyan, Solutions of the bi-confluent Heun equation in terms of the Hermite functions, *Ann. Phys.*, **383** (2017), 79–91. https://doi.org/10.1016/j.aop.2017.04.015

23. T. A. Ishkhanyan, V. P. Krainov, A. M. Ishkhanyan, Confluent hypergeometric expansions of the confluent Heun function governed by two-term recurrence relations, *J. Phys.: Conf. Ser.*, **1416** (2019), 012014. https://doi.org/10.1088/1742-6596/1416/1/012014

24. A. Ishkhanyan, C. Cesarano, Generalized-hypergeometric solutions of the general Fuchsian linear ODE having five regular singularities, *Axioms*, **8** (2019), 102. https://doi.org/10.3390/axioms8030102

25. A. M. Ishkhanyan, Appell hypergeometric expansions of the solutions of the general Heun equation, *Constr. Approx.*, **49** (2019), 445–459. https://doi.org/10.1007/s00365-018-9424-8

26. G. Lévai, Potentials from the polynomial solutions of the confluent Heun equation, *Symmetry*, **15** (2023), 461. https://doi.org/10.3390/sym15020461

27. M. Hortaçsu, Heun functions and some of their applications in physics, *Adv. High Energy Phys.*, **2018** (2018), 8621573. https://doi.org/10.1155/2018/8621573

28. B. D. B. Figueiredo, Schrödinger equation as a confluent Heun equation, *Phys. Scr.*, **99** (2024), 055211. https://doi.org/10.1088/1402-4896/ad3510

29. V. M. Buchstaber, S. I. Tertychnyi, Representations of the Klein group determined by quadruples of polynomials associated with the double confluent Heun equation, *Math. Notes*, **103** (2018), 357–371. https://doi.org/10.1134/S0001434618030033

30. A. A. Glutsyuk, On constrictions of phase-lock areas in model of overdamped Josephson effect and transition matrix of the double-confluent Heun equation, *J. Dyn. Control Syst.*, **25** (2019), 323–349. https://doi.org/10.1007/s10883-018-9411-1

31. T. Stoyanova, Stokes matrices of a reducible double confluent Heun equation via monodromy matrices of a reducible general Huen equation with symmetric finite singularities, *J. Dyn. Control Syst.*, **28** (2022), 207–245. https://doi.org/10.1007/s10883-021-09571-0

32. T. Grava, G. Mazzuca, Generalized Gibbs ensemble of the Ablowitz–Ladik lattice, circular β-ensemble and double confluent Heun equation, *Commun. Math. Phys.*, **399** (2020), 1689–1729. https://doi.org/10.1007/s00220-023-04642-8

33. S. I. Tertichniy, On the Monodromy-Preserving deformation of a double confluent Heun equation, *Proc. Steklov Inst. Math.* , **326** (2024), 303–338. https://doi.org/10.1134/S0081543824040151

34. A. Tonda, Review of PySR: high-performance symbolic regression in Python and Julia, *Genet. Program. Evolvable Mach.*, **26** (2025), 7. https://doi.org/10.1007/s10710-024-09503-4

35. E. H. Doha, H. M. Ahmed, Recurrences and explicit formulae for the expansion and connection coefficients in series of Bessel polynomials, *J. Phys. A: Math. Gen.*, **37** (2004), 8045. https://doi.org/10.1088/0305-4470/37/33/006

36. W. M. Abd-Elhameed, Novel formulae of certain generalized Jacobi polynomials, *Mathematics*, **10** (2022), 4237. https://doi.org/10.3390/math10224237

37. A. Zhedanov, An explicit example of polynomials orthogonal on the unit circle with a dense point spectrum generated by a geometric distribution, *Symmetry Integr. Geom.*, **16** (2020), 140. https://doi.org/10.3842/SIGMA.2020.140

38. W. Van Assche, J. S. Geronimo, Asymptotics for orthogonal polynomials on and off the essential spectrum, *J. Approx. Theory*, **55** (1988), 220–231. https://doi.org/10.1016/0021-9045(88)90088-3

39. T. S. Chihara, Orthogonal polynomials with discrete spectra on the real line, *J. Approx. Theory*, **42** (1984), 97–105. https://doi.org/10.1016/0021-9045(84)90059-5

40. T. S. Chihara, Orthogonal polynomials whose distribution functions have finite point spectra, *SIAM J. Math. Anal.*, **11** (1980), 358–364. https://doi.org/10.1137/0511033

41. W. B. Langdon, Genetic programming convergence, In: *Proceedings of the genetic and evolutionary computation conference companion*, New Youk: Association for Computing Machinery, 2022, 27–28. https://doi.org/10.1145/3520304.3534063

42. J. He, L. S. Kang, On the convergence rates of genetic algorithms, *Theor. Comput. Sci.*, **229** (1999), 23–39. https://doi.org/10.1016/S0304-3975(99)00091-2

43. F. Hjouj, M. S. Jouini, On orthogonal polynomials and finite moment problem, *The Open Chemical Engineering Journal*, **16** (2022), e2209260. http://doi.org/10.2174/18741231-v16-e2209260

44. R. Askey, I. J. Schoenberg, A. Sharma, Hausdorff's moment problem and expansions in Legendre polynomials, *J. Math. Anal. Appl.*, **86** (1982), 237–245. https://doi.org/10.1016/0022-247X(82)90267-0

45. Z. G. Xu, C. X. Liu, T. T. Liang, Tempered fractional neural grey system model with Hermite orthogonal polynomial, Âlex. Eng. J., **123** (2025), 403–414. https://doi.org/10.1016/j.aej.2025.03.037

46. G. Rudolph, Convergence analysis of canonical genetic algorithms, *IEEE T. Neural Networ.*, **5** (1994), 96–101. https://doi.org/10.1109/72.265964

47. D. Sudholt, The benefits of population diversity in evolutionary algorithms: A survey of rigorous runtime analyses, In: *Theory of evolutionary computation*, Cham: Springer, 2019, 359–404. https://doi.org/10.1007/978-3-030-29414-4_8

48. M. Cranmer, PySR: Tuning and workflow tips, Astroautomata, Available from: `https://astroautomata.com/PySR/tuning/`.

49. T. Taheri, A. A. Aghaei, K. Parand, An orthogonal polynomial kernel-based machine learning model for differential-algebraic equations, (2024), arXiv:2401.14382. https://doi.org/10.48550/arXiv.2401.14382

50. S. Dey, S. Dubey, Orthogonal polynomial-based neural network solution for differential equations with implicit boundary conditions, *Int. J. Numer. Method. H.*, (2025). https://doi.org/10.1108/HFF-11-2024-0901

AIMS Press